Oracle Rdb™ for OpenVMS

Oracle RMU™ Reference Manual

Release 7.3.3.0

November 2018

ORACLE®

Oracle RMU Reference Manual

Release 7.3.3.0

Copyright © 1987, 2018, Oracle Corporation. **All rights reserved.**

Primary Author: Rdb Engineering and Documentation group

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle, Java, Oracle Rdb, Hot Standby, LogMiner for Rdb, Oracle SQL/Services, Oracle CODASYL DBMS, Oracle RMU, Oracle CDD/Repository, Oracle Trace, and Rdb7 are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Send Us Your Comments ix

Preface xi

1 Oracle RMU Command Syntax

1.1 Command Parameters 1–2
1.2 Command Qualifiers 1–2
1.3 Using Indirect Command Files 1–4
1.4 Required Privileges for Oracle RMU Commands 1–4
1.5 RMU Alter Command 1–12
1.6 RMU Analyze Command 1–14
1.7 RMU Analyze Cardinality Command 1–29
1.8 RMU Analyze Indexes Command 1–35
1.9 RMU Analyze Placement Command 1–50
1.10 RMU Backup Command 1–65
1.11 RMU Backup After_Journal Command 1–117
1.12 RMU Backup Plan Command 1–151
1.13 RMU Checkpoint Command 1–158
1.14 RMU Close Command 1–161
1.15 RMU Collect Optimizer_Statistics Command 1–167
1.16 RMU Convert Command 1–182
1.17 RMU Copy_Database Command 1–193
1.18 RMU Delete Optimizer_Statistics Command 1–216
1.19 RMU Dump Command 1–221
1.20 RMU Dump After_Journal Command 1–240
1.21 RMU Dump Audit Command 1–254
1.22 RMU Dump Backup_File Command 1–261
1.23 RMU Dump Export Command 1–281
1.24 RMU Dump Metadata_File Command 1–285
1.25 RMU Dump Recovery_Journal Command 1–288
1.26 RMU Dump Row_Cache Command 1–290

1.27 RMU Dump Symbols Command 1–293
1.28 RMU Extract Command 1–295
1.29 RMU Insert Optimizer_Statistics Command 1–341
1.30 RMU Librarian Command 1–346
1.31 RMU Load Command 1–349
1.32 RMU Load Plan Command 1–404
1.33 RMU Monitor Reopen_Log Command 1–412
1.34 RMU Monitor Start Command 1–414
1.35 RMU Monitor Stop Command 1–418
1.36 RMU Move_Area Command 1–421
1.37 RMU Open Command 1–443
1.38 RMU Optimize After_Journal Command 1–452
1.39 RMU Populate_Cache Command 1–468
1.40 RMU Reclaim Command 1–471
1.41 RMU Recover Command 1–476
1.42 RMU Recover Resolve Command 1–507
1.43 RMU Repair Command 1–510
1.44 RMU Resolve Command 1–523
1.45 RMU Restore Command 1–527
1.46 RMU Restore Only_Root Command 1–583
1.47 RMU Server After_Journal Reopen_Output Command 1–621
1.48 RMU Server After_Journal Start Command 1–623
1.49 RMU Server After_Journal Stop Command 1–625
1.50 RMU Server Backup_Journal Resume Command 1–627
1.51 RMU Server Backup_Journal Suspend Command 1–629
1.52 RMU Server Record_Cache Checkpoint Command 1–632
1.53 RMU Set After_Journal Command 1–634
1.54 RMU Set AIP Command 1–653
1.55 RMU Set Audit Command 1–658
1.56 RMU Set Buffer_Object Command 1–678
1.57 RMU Set Corrupt_Pages Command 1–681
1.58 RMU Set Database Command 1–687
1.59 RMU Set Galaxy Command 1–690
1.60 RMU Set Global_Buffers Command 1–692
1.61 RMU Set Logminer Command 1–694
1.62 RMU Set Privilege Command 1–696
1.63 RMU Set Row_Cache Command 1–709
1.64 RMU Set Server Command 1–715
1.65 RMU Set Shared_Memory Command 1–719
1.66 RMU Set Statistics Command 1–724
1.67 RMU Show Command 1–732
1.67.1 RMU Show After_Journal Command 1–733
1.67.2 RMU Show AIP Command 1–747

1.67.3 RMU Show Audit Command 1–752
1.67.4 RMU Show Corrupt_Pages Command 1–757
1.67.5 RMU Show Locks Command 1–759
1.67.6 RMU Show Logical_Names Command 1–777
1.67.7 RMU Show Optimizer_Statistics Command 1–780
1.67.8 RMU Show Privilege Command 1–787
1.67.9 RMU Show Statistics Command 1–790
1.67.10 RMU Show System Command 1–818
1.67.11 RMU Show Users Command 1–820
1.67.12 RMU Show Version Command 1–823
1.68 RMU Unload Command 1–827
1.69 RMU Unload After_Journal Command 1–861
1.70 RMU Verify Command 1–902

2 RdbALTER Utility Command Syntax

2.1 AREA . . . PAGE Command 2–4
2.2 ATTACH Command 2–6
2.3 COMMIT Command 2–8
2.4 DEPOSIT Command 2–9
2.5 DEPOSIT AREA_HEADER Command 2–15
2.6 DEPOSIT FILE Command 2–20
2.7 DEPOSIT ROOT Command 2–22
2.8 DEPOSIT ROOT UNIQUE_IDENTIFIER Command 2–24
2.9 DETACH Command 2–27
2.10 DISPLAY Command 2–28
2.11 DISPLAY AREA_HEADER Command 2–35
2.12 DISPLAY FILE Command 2–39
2.13 DISPLAY ROOT Command 2–41
2.14 DISPLAY ROOT UNIQUE_IDENTIFIER Command 2–43
2.15 EXIT Command 2–45
2.16 HELP Command 2–46
2.17 LOG Command 2–47
2.18 MAKE CONSISTENT Command 2–48
2.19 MOVE Command 2–50
2.20 NOLOG Command 2–51
2.21 PAGE Command 2–52
2.22 RADIX Command 2–53
2.23 ROLLBACK Command 2–54
2.24 UNCORRUPT Command 2–55
2.25 VERIFY Command 2–57

A RMU Load Record Definition File Language

A.1 DEFINE FIELD statement A–1
A.2 DEFINE RECORD Statement A–2
A.2.1 Usage Notes A–4
A.3 Additional Data Types A–6
A.3.1 Date-Time Syntax A–10

B Using LogMiner for Rdb

B.1 Restrictions and Limitations with LogMiner for Rdb B–2
B.2 Information Returned by LogMiner for Rdb B–3
B.3 Record Definition Prefix for LogMiner Fields B–5
B.4 SQL Table Definition Prefix for LogMiner Fields B–6
B.5 Segmented String Columns B–6
B.6 Using LogMiner to Minimize Application Downtime for Maintenance . . . B–6
B.7 Using an OpenVMS Pipe B–8

C Using Per-Process Monitoring for RMU Show Statistics

C.1 Per-Process Monitoring Operational Modes C–1
C.2 Per-Process Monitoring Facility Activation C–2
C.3 Per-Process Monitoring Facility Process Activation C–4
C.4 Per-Process Monitoring Run-Time Options C–6
C.5 Detached Process Monitoring C–8
C.6 Per-Process Monitoring Overview Information C–8

Index

Figures

1 A Sample Syntax Diagram xiv
1–1 Output from the RMU Show After_Journal Command 1–734
A–1 DEFINE FIELD Statement A–2
A–2 Data Types A–7

Tables

1–1	Privileges Required for Oracle RMU Commands	1–5
1–2	RMU$FLAGS Bits Used by the RMU Analyze Indexes Command	1–43
1–3	RMU$FLAGS Bits Used by the RMU Analyze Placement Command	1–56
1–4	RMU Backup Options	1–68
1–5	How Tapes are Relabeled During a Backup Operation	1–105
1–6	Statistics Gathered by the RMU Collect Optimizer_Statistics Command	1–169
1–7	System Tables Used to Store Optimizer Statistics	1–177
1–8	RMU Dump Command Header Options	1–224
1–9	RDBNSA$K types	1–256
1–10	Parameters for Generated Command File	1–306
1–11	Using Qualifiers to Determine Output Selection	1–322
1–12	Data Type Conversions Performed by Oracle Rdb	1–355
1–13	Columns in a Database Table for Storing Security Audit Journal Records	1–378
1–14	DACCESS Privileges for Database Objects (Part 1)	1–662
1–15	DACCESS Privileges for Database Objects (Part 2)	1–663
1–16	Buffer Object Control	1–679
1–17	Server Types and Logical Names	1–716
1–18	Lock Qualifier Combinations	1–760
1–19	RESOURCE_TYPE Keywords	1–765
1–20	Lock Mode Compatibility	1–768
1–21	Output Fields	1–867
1–22	Commit Record Contents	1–872
1–23	Parameter Record Contents	1–875
A–1	Character sets supported by Oracle RMU Load	A–8
B–1	Output Fields	B–4

Send Us Your Comments

Oracle Rdb for OpenVMS
Oracle RMU Reference Manual, Release 7.3

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title, chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail:InfoRdb_US@oracle.com
- FAX — 603-897-3825 Attn: Oracle Rdb
- Postal service:
  Oracle Corporation
  Oracle Rdb Documentation
  One Oracle Drive
  Nashua, NH 03062-2804
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Purpose of This Manual

Oracle Rdb is a general-purpose database management system based on the relational database model. Oracle RMU, the Oracle Rdb management utility, is the portion of Oracle Rdb that you can use to perform database maintenance tasks, and monitor and display information about Oracle Rdb databases.

This manual provides the syntax, semantics, and reference material for Oracle RMU through Release 7.3.3.0.

Intended Audience

To use this manual effectively, you should be familiar with data processing procedures, basic database management concepts and terminology, and the OpenVMS operating system.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Structure

This manual contains two chapters, three appendixes, and an index, as follows:

Chapter 1	Describes the syntax and semantics of Oracle RMU commands. Oracle RMU commands allow you to display information about, monitor, and manage Oracle Rdb databases.
Chapter 2	Describes the syntax and semantics of the RMU/ALTER commands available in the RdbALTER utility.
Appendix A	Describes the syntax generated for specially formatted output files (.rrd files) by some Oracle RMU commands.
Appendix B	Describes how to use the LogMiner for Rdb feature.
Appendix C	Describes how to use the RMU Per-Process statistics feature.

Oracle RMU Command Syntax Diagrams

This manual uses the following conventions to present the syntax of Oracle RMU commands:

- Oracle RMU commands and qualifiers appear in initial capitals. Commands and qualifiers can be entered in upper-, lower-, or mixed-case type.
- Command parameters and variables are represented in lowercase type These format elements represent parameters or variables for which you must supply a value.
- Horizontal ellipsis points (. . .) mean you can enter additional information, such as parameters or qualifier arguments.
- Brackets ([ ]) enclose optional clauses from which you can choose one or none.
- A comma (,) separating qualifier arguments means you can specify one or more arguments with the qualifier.
- A vertical bar (|) separating qualifier arguments in syntax diagrams means you can specify only one argument with the qualifier.

For example, the following is the format for the RMU Analyze Placement command:

```
RMU/Analyze/Placement root-file-spec [index-name[,...]]
```

The Analyze command and the Placement qualifier can be entered as upper-, lower-, or mixed-case type. The root-file-spec parameter indicates that you must supply a root file specification. The index-name enclosed in brackets (and followed by a comma and ellipses in brackets) indicates that you can supply one or more index-names, separated by commas.

RdbAlter Syntax Diagrams

This manual uses syntax diagrams to graphically present the syntax of RdbAlter statements. Syntax diagrams portray optional, required, and repeating characteristics of any Oracle Rdb statement. You can learn the syntax of a command by reading that command's syntax diagram.

To read a syntax diagram, start from the left and follow the arrows until you exit from the diagram at the right. When you come to a branch in the path, choose the branch that contains the option you want. If you want to omit an option, choose the path with no language elements. If a diagram occupies more than one horizontal line, the arrow returns to the left margin. Syntax diagrams can contain:

- Names of syntax diagrams

  If a diagram is named, the name appears in lowercase type above and to the left of the diagram. Syntax diagrams can refer to each other by name. The equal sign (=) indicates that the name is equivalent to the diagram and that the diagram can be substituted wherever the name appears.

  If the diagram contains the name of another diagram, substitute that other diagram where the name appears. Such a substitution is similar to putting the name of a field where "field-name" appears. Most named syntax diagrams appear as subdiagrams following the main diagram.

- Keywords

  Keywords appear in uppercase type. If a keyword is underlined, you must include it in the command. A keyword without underlining is optional, but the keyword makes the command more readable. Omitting an optional keyword does not change the meaning of a command.

- Punctuation marks

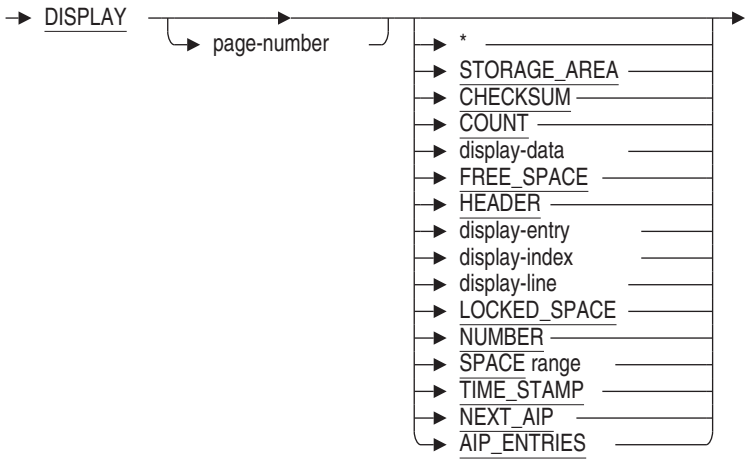
  Punctuation marks are included in the diagram when required by the syntax of the command.

- User-supplied elements

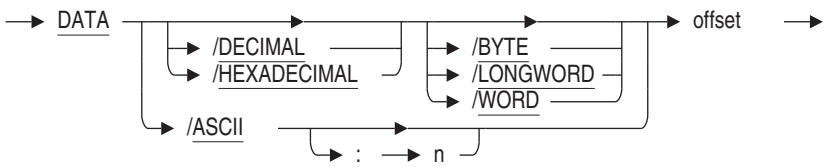
  User-supplied elements appear in lowercase type. These elements can include names, expressions, and literals. They are usually defined in text following the diagram.

Figure 1 shows a portion of the syntax diagram for the RdbALTER DISPLAY command.

Figure 1 A Sample Syntax Diagram



display-data =



DISPLAY	Is in uppercase and underlined on the main line of the diagram. Therefore, you must supply the keyword (which can usually be abbreviated).
page-number	Is in lowercase on a branch of the diagram. Therefore, the page-number clause is optional; if you include it, you must supply a substitute for page-number.
display-data	Is in lowercase on a branch. Because it parallels an empty branch, the display-data clause is optional. The subdiagram expands the definition of display-data.

Conventions

OpenVMS means both the OpenVMS Alpha and OpenVMS I64 operating systems.

In this manual, Oracle Rdb refers to Oracle Rdb for OpenVMS. Version 7.3 of Oracle Rdb software is often referred to as V7.3.

The SQL interface to Oracle Rdb is referred to as SQL. This interface is the Oracle Rdb implementation of the SQL standard adopted in 1999. This standard is referred to as the ANSI/ISO SQL standard or SQL:1999. See the Oracle Rdb Release Notes for more information.

Oracle CDD/Repository software is referred to as the dictionary, the data dictionary, or the repository.

In examples, an implied carriage return occurs at the end of each line, unless otherwise noted. You must press the Return key at the end of a line of input.

The following conventions are also used in this manual:

. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
...	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted.
e, f, t	Index entries in the printed manual may have a lowercase e, f, or t following the page number; the e, f, or t is a reference to the example, figure, or table, respectively, on that page.
[ ]	In format descriptions, brackets enclose optional clauses from which you can choose one or none. In a prompt, brackets indicate that the enclosed item is the default response. For example, [y] means the default response is Yes.
{ }	Braces in format descriptions enclose clauses from which you must choose at least one.
$	The dollar sign represents the OpenVMS Command Language (DCL) prompt.
boldface text	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
⇒	This symbol indicates that you choose a menu item.
menu name	An underlined character in a menu name indicates a mnemonic key you can use as an alternative to the mouse.

1
Oracle RMU Command Syntax

Oracle RMU, the Oracle Rdb management utility, lets database administrators manage Oracle Rdb databases. Oracle RMU commands are executed at the operating system prompt. Oracle RMU command syntax follows the rules and conventions of the OpenVMS Command Language (DCL).

Oracle RMU commands allow you to display the contents of database files, control the Oracle Rdb monitor process, verify data structures, perform maintenance tasks (such as backup and restore operations), and list information about current database users and database activity statistics.

Oracle RMU commands consist of words, generally verbs, that have parameters and qualifiers to define the action to be performed.

RMU commands that accept a TSN keyword or qualifier value now accept input formats as follows:

- A decimal string representing a quadword TSN value
- A hexadecimal string starting with "%X" representing a quadword TSN value
- A two-part decimal string separated by a colon representing a quadword TSN value as high and low longwords

Following are some example uses of input TSN values:

```
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=54321
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=123456234253245
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=%X7655
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=%X000000715F856AB
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=0:871251
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=3:53487
$ RMU /DUMP /AFTER_JOURNAL J1.AIJ /FIRST=TSN=21:653156
```

1.1 Command Parameters

One or more spaces separate command parameters and their qualifiers from the command keyword. Command parameters define the file, index, or entity on which the command will act. In most cases, you can omit the parameter from the command line and enter it in response to a prompt.

In the following sample command, RMU/DUMP is the command keyword and MF_PERSONNEL is the command parameter:

```
$ RMU/DUMP MF_PERSONNEL
```

When a storage area is a command parameter in an Oracle RMU command, use the *storage area name* instead of the *storage area file specification*. For example:

```
$ RMU/RESTORE/AREA MF_PERSONNEL.RBF EMPIDS_LOW/THRESHOLDS=(65,75,80)
```

Some commands, such as the RMU Backup command, require two or more command parameters. If you provide all parameters (for example, a root file specification and a backup file name), there are no prompts. Other commands, such as the RMU Restore command, have one required and one optional command parameter. In this case, there are no prompts if you provide the backup parameter but not the storage area parameter. However, if you do not provide either parameter, Oracle RMU prompts for both.

1.2 Command Qualifiers

Command qualifiers modify the behavior of an Oracle RMU command. Although similar in appearance, command qualifiers are different from the Oracle RMU commands themselves. The first (and sometimes the subsequent) word that follows the RMU keyword is the command itself. For instance, in the following example, /DUMP and /AFTER_JOURNAL are part of the Oracle RMU command and thus must appear in the order shown. /OPTION=STATISTICS and /LOG are command qualifiers and can appear in any order after the Oracle RMU command. You can determine which portions of an Oracle RMU command are the command itself, and which portions are command qualifiers by noting the documented name of the command,

```
$ RMU/DUMP/AFTER_JOURNAL aij_one.aij /OPTION=STATISTICS/LOG
```

Command qualifiers can be entered as upper-, lower-, or mixed-case type. They always begin with a slash (/) followed by a qualifier word.

In some cases, an equal sign (=) and a qualifier value follow the qualifier word. A qualifier value can be simple (a number, a string, or a keyword) or compound (a list of numbers, strings, or keywords separated by commas, enclosed in parentheses) or an indirect command file name. For information on using indirect command files, see Section 1.3.

A default value for a qualifier indicates what qualifier will be used if you omit the qualifier completely. Omitting a qualifier is not the same thing as specifying a qualifier with a default argument.

Command qualifiers influence the overall action of a command. Command qualifiers must be placed following the command keyword but before any parameters.

In the following example, the command qualifier, Users, immediately follows the Dump keyword and precedes the command parameter, mf_personnel:

```
$ RMU/DUMP/USERS MF_PERSONNEL
```

Parameter qualifiers (also referred to as file qualifiers or area qualifiers) affect the treatment of parameters in the command. If the command includes multiple instances of a given type of parameter, the placement of parameter qualifiers affects their scope of influence as follows:

- If you position the parameter qualifier after a particular parameter, the qualifier affects only that parameter. This is *local* use of a parameter qualifier.
- If you position the parameter qualifier before the first parameter, the qualifier applies to all instances of the parameter. This is *global* use of a parameter qualifier. Not all parameter qualifiers can be used globally. To identify such qualifiers, read the description of the qualifier.
- If you position the parameter qualifier after a parameter, the qualifier applies only to that instance of the parameter. Local parameter qualifiers take precedence over global parameter qualifiers, in most cases. Exceptions are documented in the qualifier descriptions for each Oracle RMU command.

The following example demonstrates the local use of the area qualifier, Thresholds, to change the threshold settings for the EMPIDS_LOW area:

```
$ RMU/RESTORE MF_PERSONNEL EMPIDS_LOW/THRESHOLDS=(70,80,90)
```

Note that if you specify a qualifier in both the negative and positive forms, the last occurrence of the qualifier is the one that takes effect. For example, the Nolog qualifier takes effect in this command:

```
$ RMU/BACKUP/LOG/NOLOG MF_PERSONNEL MF_PERS
```

This is consistent with DCL behavior for negative and positive qualifiers.

1.3 Using Indirect Command Files

Numerous Oracle RMU command operations accept lists of names as values for certain qualifiers, such as the Areas= or Lareas= qualifiers. The command syntax can easily exceed the maximum length of 1024 characters accepted by DCL. To overcome the problem of syntax that is too long, you can include the names in an indirect command file and specify the indirect command file following the qualifier. Throughout this manual, this is commonly referred to as using an indirect file reference. Note that indirect command files can be nested to any depth.

Each indirect command file (default file extension .opt) contains a list of names with one name per line. A comment, preceded by an exclamation point, can be appended to a name, or it can be inserted between lines. A reference to an indirect command file in the list must be preceded by an at sign (@) and enclosed in quotation marks (""). For example: "@EMPIDS".

The following example shows the contents of an indirect command file called empids.opt. It lists the EMPIDS_LOW, EMPIDS_MID, and EMPIDS_OVER storage areas. The last line in the example shows how you would reference the indirect command file in an Oracle RMU command line with the required quotation marks.

```
$ TYPE EMPIDS.OPT
   EMPIDS_LOW      ! Employee Areas
   EMPIDS_MID
   EMPIDS_OVER

$ RMU/ANALYZE/AREA="@EMPIDS" MF_PERSONNEL ! ANALYZE EMPLOYEE AREAS
```

1.4 Required Privileges for Oracle RMU Commands

An access control list (ACL) is created by default on the root file of each Oracle Rdb database. To be able to use a particular Oracle RMU command for the database, you must be granted the appropriate Oracle RMU privilege for that command in the database's root file ACL. For some Oracle RMU commands, you must have one or more OpenVMS privileges as well as the appropriate Oracle RMU privilege to be able to use the command.

Note that the root file ACL created by default on each Oracle Rdb database controls only your Oracle RMU access to the database (by specifying privileges that will allow a user or group of users access to specific Oracle RMU commands). Root file ACLs do not control your access to the database with SQL (structured query language) statements. See Section 1.67.8 for information on how to display your Oracle RMU access to the database.

Your access to a database with SQL statements is governed by the privileges granted to you in the database ACL (the ACL that is displayed when you use the SQL SHOW PROTECTION ON DATABASE command).

Table 1–1 shows the Oracle RMU privileges you must have to use each Oracle RMU command. When more than one Oracle RMU privilege appears in the Required Oracle RMU Privileges column, if you have any of the listed Oracle RMU privileges, you will pass the Oracle RMU privilege check for the specified Oracle RMU command.

If the Oracle RMU command requires a user to have one or more OpenVMS privileges in addition to the appropriate Oracle RMU privileges, the OpenVMS privileges are shown in the Required OpenVMS Privileges column of Table 1–1. When more than one OpenVMS privilege is listed in the Required OpenVMS Privileges column, you must have all of the listed OpenVMS privileges to pass the OpenVMS privilege check for the Oracle RMU command.

The OpenVMS Override Privileges column of Table 1–1 shows one or more OpenVMS privileges that allow a user without the appropriate required Oracle RMU and OpenVMS privileges for an Oracle RMU command to use the command anyway. When more than one OpenVMS privilege is listed in the OpenVMS Override Privileges column, you can use the specified Oracle RMU command if you have any of the listed privileges.

Table 1–1 Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Alter	RMU$ALTER[1]		SYSPRV, BYPASS
Analyze Areas	RMU$ANALYZE		SYSPRV, BYPASS

[1]You must have the OpenVMS SYSPRV or BYPASS privilege if you use an RMU/ALTER command to change a file name.

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Analyze Cardinality	RMU$ANALYZE		SYSPRV, BYPASS
Analyze Indexes	RMU$ANALYZE		SYSPRV, BYPASS
Analyze Placement	RMU$ANALYZE		SYSPRV, BYPASS
Backup	RMU$BACKUP		SYSPRV, BYPASS
Backup After_Journal	RMU$BACKUP		SYSPRV, BYPASS
Backup Plan	RMU$BACKUP		SYSPRV, BYPASS
Checkpoint	RMU$BACKUP, RMU$OPEN		WORLD
Close	RMU$OPEN		WORLD
Collect Optimizer_Statistics	RMU$ANALYZE		SYSPRV, BYPASS
Convert	RMU$CONVERT, RMU$RESTORE		SYSPRV, BYPASS
Copy_Database	RMU$COPY		SYSPRV, BYPASS
Delete Optimizer_Statistics	RMU$ANALYZE		SYSPRV, BYPASS
Dump After_Journal	RMU$DUMP		SYSPRV, BYPASS
Dump Areas	RMU$DUMP		SYSPRV, BYPASS
Dump Audit	RMU$SECURITY		BYPASS, SECURITY
Dump Backup_File	RMU$DUMP, RMU$BACKUP, RMU$RESTORE	READ[2]	BYPASS

[2]You must have OpenVMS READ access for the .rbf file.

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Dump Export		READ[3]	BYPASS
Dump Header	RMU$DUMP, RMU$BACKUP, RMU$OPEN		SYSPRV, BYPASS
Dump Lareas	RMU$DUMP		SYSPRV, BYPASS
Dump Recovery_Journal		READ[4]	BYPASS
Dump Row Cache	RMU$DUMP		SYSPRV, BYPASS
Dump Snapshots	RMU$DUMP		SYSPRV, BYPASS
Dump Users	RMU$DUMP, RMU$BACKUP, RMU$OPEN		WORLD
Extract	RMU$UNLOAD		SYSPRV, BYPASS
Insert Optimizer Statistics	RMU$ANALYZE		SYSPRV, BYPASS
Load	RMU$LOAD		SYSPRV, BYPASS
Load Audit	RMU$SECURITY		SECURITY, BYPASS
Load Plan	RMU$LOAD		SYSPRV, BYPASS
Monitor Reopen_Log		WORLD, CMKRNL, DETACH, PSWAPM, ALTPRI, SYSGBL, SYSNAM, SYSPRV, BYPASS	SETPRV

[3]You must have OpenVMS READ access for the .rbr or .unl file.

[4]You must have OpenVMS READ access for the .ruj file.

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Monitor Start		WORLD, CMKRNL, DETACH, PSWAPM, ALTPRI, PRMMBX, SYSGBL, SYSNAM, SYSPRV, BYPASS	SETPRV
Monitor Stop		WORLD, CMKRNL, DETACH, PSWAPM, ALTPRI, PRMMBX, SYSGBL, SYSNAM, SYSPRV, BYPASS	SETPRV
Move_Area	RMU$MOVE		SYSPRV, BYPASS
Open	RMU$OPEN		WORLD
Optimize After_Journal	RMU$BACKUP, RMU$RESTORE		SYSPRV, BYPASS
Reclaim	RMU$ALTER		SYSPRV, BYPASS
Recover	RMU$RESTORE		SYSPRV, BYPASS
Recover Resolve	RMU$RESTORE		SYSPRV, BYPASS
Repair	RMU$ALTER		SYSPRV, BYPASS
Resolve	RMU$RESTORE		SYSPRV, BYPASS

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Restore	RMU$RESTORE		SYSPRV, BYPASS
Restore Only_Root	RMU$RESTORE		SYSPRV, BYPASS
Server After_Journal Reopen_ Output	RMU$OPEN		WORLD
Server After_Journal Start	RMU$OPEN		WORLD
Server After_Journal Stop	RMU$OPEN		WORLD
Server Backup_Journal Resume	RMU$OPEN		WORLD
Server Backup_Journal Suspend	RMU$OPEN		WORLD
Server Record_Cache	RMU$OPEN		WORLD
Set After_Journal	RMU$ALTER, RMU$BACKUP, RMU$RESTORE		SYSPRV, BYPASS
Set AIP	RMU$ALTER		BYPASS, SECURITY
Set Audit	RMU$SECURITY		SECURITY, BYPASS
Set Buffer Object	RMU$ALTER		SYSPRV, BYPASS
Set Corrupt_Pages	RMU$ALTER, RMU$BACKUP, RMU$RESTORE		SYSPRV, BYPASS
Set Galaxy	RMU$ALTER		SYSPRV, BYPASS
Set Global Buffers	RMU$ALTER		SYSPRV, BYPASS
Set Logminer	RMU$ALTER, RMU$BACKUP, RMU$RESTORE		SYSPRV, BYPASS

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Set Privilege	RMU$SECURITY		SECURITY, BYPASS
Set Row_Cache	RMU$ALTER		SYSPRV, BYPASS
Set Shared Memory	RMU$ALTER		SYSPRV, BYPASS
Show After_Journal	RMU$BACKUP, RMU$RESTORE, RMU$VERIFY		SYSPRV, BYPASS
Show AIP	RMU$DUMP		SYSPRV, BYPASS
Show Audit	RMU$SECURITY		SECURITY, BYPASS
Show Corrupt_Pages	RMU$BACKUP, RMU$RESTORE, RMU$VERIFY		SYSPRV, BYPASS
Show Locks		WORLD	
Show Optimizer_Statistics	RMU$ANALYZE, RMU$SHOW		SYSPRV, BYPASS
Show Privilege	RMU$SECURITY		SECURITY, BYPASS
Show Statistics[8]	RMU$SHOW		SYSPRV, BYPASS, WORLD
Show System		WORLD	
Show Users[5]	RMU$SHOW, RMU$BACKUP, RMU$OPEN		WORLD
Show Version			

[5]You must have OpenVMS WORLD access *in addition to* the RMU$BACKUP, RMU$OPEN, or RMU$SHOW privilege for all databases on your node if you do not specify a database file name.

[8]You must have the OpenVMS WORLD privilege if you use this command to display statistics about other users (as opposed to database statistics). You must have both the OpenVMS WORLD and BYPASS privileges if you use this command to update fields in the Database Dashboard.

(continued on next page)

Table 1–1 (Cont.) Privileges Required for Oracle RMU Commands

Oracle RMU Command	Required Oracle RMU Privileges	Required OpenVMS Privileges	OpenVMS Override Privileges
Unload	RMU$UNLOAD[6]		SYSPRV, BYPASS
Unload After_Journal	RMU$DUMP		SYSPRV, BYPASS
Verify	RMU$VERIFY[7]		SYSPRV, BYPASS

[6]The appropriate Oracle Rdb privileges for accessing the database tables involved are also required.

[7]You must also have the SQL DBADM privilege.

1.5 RMU Alter Command

Invokes the RdbALTER utility for Oracle Rdb.

Note

Oracle Corporation recommends that the RdbALTER utility be used only as a last resort to provide a temporary patch to a corrupt database. The RdbALTER utility should not be used as a routine database management tool.

Use the RdbALTER utility only after you fully understand the internal data structure, know the information the database should contain, and know the full effects of the command. Because of the power of the RdbALTER utility and the cascading effects it can have, Oracle Corporation recommends that you experiment on a copy of the damaged database before applying the RdbALTER utility to a production database.

Complete information on the RdbALTER utility is found in Chapter 2, which provides a syntax diagram and description for each RdbALTER command.

To invoke the RdbALTER utility, enter the RMU Alter command in the following format:

```
$ RMU/ALTER [root-file-spec]
```

The optional root file parameter identifies the database you want to alter. If you specify this parameter, you automatically attach to the specified database. If you do not specify this parameter, you must use the RdbALTER ATTACH command. See Section 2.2 for more information on the ATTACH command.

The RMU Alter command responds with the following prompt:

```
RdbALTER>
```

This prompt indicates that the system expects RdbALTER command input.

To access the RdbALTER Help file, enter the following:

```
RdbALTER> HELP
```

1.5 RMU Alter Command

To use the RMU Alter command for a database, you must have the RMU$ALTER privilege in the root file ACL for the database or the OpenVMS SYSPRV or BYPASS privilege. You must have the OpenVMS SYSPRV or BYPASS privilege if you are using an RMU Alter command to change a file name.

1.6 RMU Analyze Command

Gathers and displays statistics on how the database uses storage, logical area, or page space.

Format

RMU/Analyze root-file-spec

Command Qualifiers	Defaults
/Areas[=storage-area-list]	/Areas
/[No]Binary_Output=file-option-list	/Nobinary_Output
/End=integer	/End=last-page
/Exclude=(options)	No logical areas excluded
/[No]Lareas[=logical-area-list]	/Lareas
/Option = {Normal | Full | Debug}	/Option=Normal
/Output=file-name	/Output=SYS$OUTPUT
/[No]Partitions[=(Tables,Indexes)]	/NoPartitions
/Start = integer	/Start=first-page

Description

The RMU Analyze command provides a maintenance tool for database administrators. It generates a formatted display of statistical information that describes storage utilization in the database. Information is displayed selectively for storage areas and logical areas, or for a range of pages in a storage area. You can use the RMU Analyze command to analyze the following:

- Space utilization for database pages
- Space utilization for storage areas
- Space utilization for logical areas

Command Parameters

root-file-spec
The file specification for the database root file to be analyzed. The default file extension is .rdb.

Command Qualifiers

Areas[=storage-area-list]
Specifies the storage areas to be analyzed. You can specify each storage area by name or by the area's ID number.

The default, the Areas qualifier, results in analysis of all storage areas. You can also specify the Areas=* qualifier to analyze all storage areas. If you specify more than one storage area, separate the storage area names or ID numbers in the storage-area-list parameter with a comma and enclose the list in parentheses. If you omit the Areas qualifier, information for all the storage areas is displayed.

You can use the Start and End qualifiers with the Areas qualifier to analyze specific pages. If you use the Start and End qualifiers when you specify more than one storage area in the storage-area-list parameter, the same specified range of pages are analyzed in each specified storage area.

The Areas qualifier can be used with an indirect command file. See Section 1.3 for more information.

Binary_Output=file-option-list
Nobinary_Output
Allows you to direct the summary results to a binary file, and to create a record definition file that is compatible with the data dictionary for the binary output file. The binary output file can be loaded into an Oracle Rdb database by using the RMU Load command with the Record_Definition qualifier for use by a user-written management application or procedure. The binary output can also be used directly by the user-written application or procedure.

Oracle reserves the right to add, remove or move fields in this record definition file (.RRD) for each major release of Rdb, as well as the output of the binary *.UNL file. Any customer applications that use these files that Rdb creates may need to be changed if this happens.

The valid file options are:

- File=file-spec

  The File option causes the Analyze command data to be stored in an RMS file that contains a fixed-length binary record for each storage area and logical area analyzed. The default file extension for the binary output file is .unl. The following command creates the binary output file analyze_ out.unl:

```
$ RMU/ANALYZE/BINARY_OUTPUT=FILE=ANALYZE_OUT MF_PERSONNEL.RDB
```

- Record_Definition=file-spec

  The Record_Definition option causes the Analyze command data record definition to be stored in an RMS file. The output file contains the definition in a subset of the data dictionary command format, a format very similar to RDO field and relation definitions. The default file extension for the record definition output file is .rrd. The following command creates the output file analyze_out.rrd:

```
$ RMU/ANALYZE/BINARY_OUTPUT=RECORD_DEFINITION=ANALYZE_OUT -
_$ MF_PERSONNEL.RDB
```

  You can specify both file options in one command by separating them with a comma and enclosing them within parentheses, for example:

```
$ RMU/ANALYZE/BINARY_OUTPUT= -
_$ (FILE=ANALYZE_OUT,RECORD_DEFINITION=ANALYZE_OUT) -
_$ MF_PERSONNEL.RDB
```

If you specify the Binary_Output qualifier, you must specify at least one of the options. The default is the Nobinary_Output qualifier, which does not create an output file.

End=integer

Specifies the ending page number for the analysis. The default is the end of the storage area file.

Exclude=options

Excludes information from the RMU Analyze command output. You can specify Exclude=System_Records or Exclude=Metadata, or both. If you specify both options, enclose them within parentheses and separate each option with a comma.

When you do not specify the Exclude qualifier, data is provided for all the logical areas in the database.

The options are as follows:

- System_Records

  Information on the RDB$SYSTEM_RECORDS logical areas is excluded from the Analyze command output.

- Metadata

  Information on all the Oracle Rdb logical areas (for example, the RDB$SYSTEM_RECORDS and RDB$COLLATIONS_NDX logical areas) is excluded from the RMU Analyze command output.

Data is accumulated for the logical areas excluded with the Exclude qualifier, but the data is excluded from the Analyze output.

You cannot use the Exclude qualifier and the Lareas qualifier in the same RMU Analyze command.

Lareas [=logical-area-list]
Nolareas
Specifies the logical areas to be analyzed. Each table in the database is associated with a logical area name. The default, the Lareas qualifier, results in analysis of all logical areas. You can also specify the Lareas=* qualifier to analyze all logical areas. If you specify more than one logical area name, separate the logical area names in the logical-area-list with a comma and enclose the list in parentheses.

The Lareas qualifier can be used with indirect command files. See Section 1.3 for more information.

Option=type
Specifies the type of information and level of detail the analysis will include. Three types of output are available:

- Normal

  Output includes only summary information. The Normal option is the default.

- Full

  Output includes histograms and summary information.

- Debug

  Output includes internal information about the data, as well as histograms and summary information. In general, use the Debug option for diagnostic support purposes. You can also use the Debug option to extract data and perform an independent analysis.

Output=file-name
Specifies the name of the file where output will be sent. The default file extension is .lis. If you do not specify the Output qualifier, the output is sent to SYS$OUTPUT.

Partitions[={Tables,Indexes}]
NoPartitions
Allows data to be collected and output for individual table and/or index partitions across multiple Oracle Rdb database storage areas when analyzing storage statistics. First statistics for each Oracle Rdb database storage area

containing partitions is output. Then statistics for each partition logical area defined for each partitioned table is output. Finally, statistics for each partition logical area defined for each partitioned index is output.

Please note the following:

- /NOPARTITIONS is the default
- /PARTITIONS=TABLES only outputs partitioned table statistics
- /PARTITIONS=INDEXES only outputs partitioned index statistics
- If only /PARTITIONS is specified, both partitioned table and partitioned index statistics are output

If /PARTITIONS=TABLES is specified, only statistics for partitioned tables is output. If /PARTITIONS=INDEXES is specified, only statistics for partitioned indexes is output. If the /LAREAS qualifier is used to specify a list of names of partitioned tables and/or partitioned indexes, only statistics for those tables and/or indexes will be output. If the /LAREAS qualifier is used to specify a list of logical area identifier numbers, only those logical area partitions for partitioned tables and/or indexes will be output. IF /BINARY_OUTPUT is specified with /PARTITIONS, record definition (*.RRD) and binary unload files (*.UNL) are created that can be used to load storage area and logical area data records for partitioned tables and/or indexes into an Oracle Rdb database table using the RMU/LOAD command.

The RMU/ANALYZE/PARTITIONS qualifier for storage statistics cannot be specified with the following RMU/ANALYZE qualifiers: /PLACEMENT, /CARDINALITY, /INDEXES, /AREAS, /START, /END and /EXCLUDE. If /LAREAS is used, it must specify a partitioned table name or index name or a logical area identifier number for a logical area defined for a partitioned table or partitioned index.

Start=integer
Specifies the starting page number for the analysis. The default is 1.

Usage Notes

- To use the RMU Analyze command for a database, you must have the RMU$ANALYZE privilege in the root file ACL for the database or the OpenVMS SYSPRV or BYPASS privilege.

  You must also have the SQL SELECT privilege on the database for which the analyze is being performed.

- When the RMU Analyze command is issued for a closed database, the command executes without other users being able to attach to the database.
- Detected asynchronous prefetch should be enabled to achieve the best performance of this command. Beginning with Oracle Rdb V7.0, by default, detected asynchronous prefetch is enabled. You can determine the setting for your database by issuing the RMU Dump command with the Header qualifier.

  If detected asynchronous prefetch is disabled, and you do not want to enable it for the database, you can enable it for your Oracle RMU operations by defining the following logicals at the process level:

  ```
  $ DEFINE RDM$BIND_DAPF_ENABLED 1
  $ DEFINE RDM$BIND_DAPF_DEPTH_BUF_CNT P1
  ```

  P1 is a value between 10 and 20 percent of the user buffer count.
- The following RMU Analyze command directs the results into a record definition file called db.rrd. This file is compatible with the syntax for creating new columns and tables in the data dictionary.

  ```
  $ RMU/ANALYZE/BINARY_OUTPUT=RECORD_DEFINITION=DB.RRD MF_PERSONNEL
  $! Display the db.rrd file created by the previous command:
  $ TYPE DB.RRD

  DEFINE FIELD RMU$DATE DATATYPE IS DATE.
  DEFINE FIELD RMU$AREA_NAME DATATYPE IS TEXT SIZE IS 32.
  DEFINE FIELD RMU$STORAGE_AREA_ID DATATYPE IS SIGNED WORD.
  DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
  DEFINE FIELD RMU$TOTAL_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$EXPANDED_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$FRAGMENTED_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$EXPANDED_FRAGMENT_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$TOTAL_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$FRAGMENTED_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$FRAGMENT_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$PAGE_LENGTH DATATYPE IS SIGNED WORD.
  DEFINE FIELD RMU$MAX_PAGE_NUMBER DATATYPE IS SIGNED LONGWORD.
  DEFINE FIELD RMU$FREE_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$OVERHEAD_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$AIP_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$ABM_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$SPAM_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$INDEX_COUNT DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$BTREE_NODE_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$HASH_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$DUPLICATES_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$OVERFLOW_BYTES DATATYPE IS F_FLOATING.
  DEFINE FIELD RMU$LOGICAL_AREA_ID DATATYPE IS SIGNED WORD.
  DEFINE FIELD RMU$RELATION_ID DATATYPE IS SIGNED WORD.
  ```

1.6 RMU Analyze Command

```
DEFINE FIELD RMU$RECORD_ALLOCATION_SIZE DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$TOTAL_SPACE DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_AREA.
   .
   .
   .
```

- The following list describes each of the fields in the db.rrd record definition:
  - RMU$DATE

    Contains the date that the Analyze operation was done
  - RMU$AREA_NAME

    Contains the name of the storage area that was analyzed
  - RMU$STORAGE_AREA_ID

    Contains the area ID of the storage area that was analyzed
  - RMU$FLAGS

    The three possible values in this field have the following meanings:
    * 0—Indicates that the record is a storage area record, not a logical area record
    * 1—Indicates that data compression is not enabled for the logical area
    * 3—Indicates that data compression is enabled for the logical area

- RMU$TOTAL_BYTES

  Contains the total size of the data stored in the logical area

- RMU$EXPANDED_BYTES

  Contains the total size of the stored data in the logical area after decompression

- RMU$FRAGMENTED_BYTES

  Contains the number of bytes in the stored fragments

- RMU$EXPANDED_FRAGMENT_BYTES

  Contains the number of bytes in the stored fragments after decompression

- RMU$TOTAL_COUNT

  Contains the total number of records stored

- RMU$FRAGMENTED_COUNT

- Contains the number of fragmented records

- RMU$FRAGMENT_COUNT

  Contains the number of stored fragments

- RMU$PAGE_LENGTH

  Contains the length in bytes of a database page in the storage area

- RMU$MAX_PAGE_NUMBER

  Contains the page number of the last initialized page in the storage area

- RMU$FREE_BYTES

  Contains the number of free bytes in the storage area

- RMU$OVERHEAD_BYTES

  Contains the number of bytes used for overhead in the storage area

- RMU$AIP_COUNT

  Contains the number of the area inventory pages (AIPs) in the storage area

- RMU$ABM_COUNT

  Contains the number of area bit map (ABM) pages in the storage area

- RMU$SPAM_COUNT

  Contains the number of space area management (SPAM) pages in the storage area

- RMU$INDEX_COUNT

  Contains the number of index records in the storage area

- RMU$BTREE_NODE_BYTES

  Contains the number of bytes for sorted indexes in the storage area

- RMU$HASH_BYTES

  Contains the number of bytes for hashed indexes in the storage area

- RMU$DUPLICATES_BYTES

  Contains the number of bytes for duplicate key values for sorted indexes in the storage area

- RMU$OVERFLOW_BYTES

  Contains the number of bytes for hash bucket overflow records in the storage area

- RMU$LOGICAL_AREA_ID

  Contains the logical area ID of the logical area that was analyzed

- RMU$RELATION_ID

  Contains the record type of the row in the logical area that was analyzed

- RMU$RECORD_ALLOCATION_SIZE

  Contains the size of a row when the table was initially defined

- RMU$TOTAL_SPACE

  Contains the number of bytes available for storing user data in the logical area (used space + free space + overhead)

Examples

Example 1

The following command analyzes the EMPIDS_LOW and EMP_INFO storage areas in the mf_personnel database:

```
$ RMU/ANALYZE/AREAS=(EMPIDS_LOW,EMP_INFO)/OUTPUT=EMP.OUT -
_$ MF_PERSONNEL.RDB
```

Example 2

Both of the following commands analyze the DEPARTMENTS and SALARY_HISTORY storage areas in the mf_personnel database:

```
$! Using storage area names to specify storage areas
$ RMU/ANALYZE/AREAS=(DEPARTMENTS,SALARY_HISTORY) MF_PERSONNEL.RDB -
$ /OUTPUT=DEP_SAL.OUT
$!
$! Using storage area ID numbers to specify storage areas
$ RMU/ANALYZE/AREAS=(2,9) MF_PERSONNEL.RDB /OUTPUT=DEP_SAL.OUT
```

Example 3

In the following example for the PART_DB database with one partitioned table T1 and one partitioned index I1, first statistics for the storage areas containing table and index partitions are output: DATA1.RDA, INDEX1.RDA and INDEX2.RDA. Then statistics for the partitioned table T1 defined with one partition SYS_P00059 in area DATA1 is output. Then statistics for the two partitioned index I1 partitions, P1 in the INDEX1 storage area and P2 in the INDEX2 storage area are output. Note that if /PARTITIONS=TABLES was specified for the PART_DB database, only statistics for the partitioned table T1 would be output and if /PARTITIONS=INDEXES was specified, only statistics for the partitioned index I1 would be output.

```
$ RMU/ANALYZE/PARTITIONS PART_DB

 Areas containing partitions for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 14:11:38.43

--------------------------------------------------------------------------------

 Storage analysis for storage area: DATA1 - file: DISK:[DIRECTORY]DATA1.RDA;1
 Area_id: 2,  Page length: 1024,  Last page: 703

 Bytes free: 366675 (51%), bytes overhead: 164457 (23%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 10000, bytes used: 188740 (26%)
   average length: 19, compression ratio: 0.90
   index records: 0, bytes used: 0 (0%)

--------------------------------------------------------------------------------

 Storage analysis for storage area: INDEX1 - file: DISK:[DIRECTORY]INDEX1.RDA;1
 Area_id: 3,  Page length: 1024,  Last page: 703

Bytes free: 685993 (95%), bytes overhead: 33451 (5%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 1, bytes used: 428 (0%)
   average length: 428, compression ratio: 1.00
   index records: 1, bytes used: 428 (0%)
     B-Tree: 428, Hash: 0, Duplicate: 0, Overflow: 0
```

1.6 RMU Analyze Command

```
------------------------------------------------------------------------------

 Storage analysis for storage area: INDEX2 - file: DISK:[DIRECTORY]INDEX2.RDA;1
 Area_id: 4,  Page length: 1024,  Last page: 703

 Bytes free: 685993 (95%), bytes overhead: 33451 (5%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 1, bytes used: 428 (0%)
   average length: 428, compression ratio: 1.00
   index records: 1, bytes used: 428 (0%)
     B-Tree: 428, Hash: 0, Duplicate: 0, Overflow: 0

------------------------------------------------------------------------------
------------------------------------------------------------------------------

 Partitioned Tables for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 14:11:38.43

------------------------------------------------------------------------------
 Storage analysis for Partitioned Table: T1

-----------------------------------

 Partition SYS_P00059 in area DATA1
 Logical area: T1 Logical area id : 59

 Larea id: 59,  Record type: 31, Record length: 26, Compressed

 Data records: 10000, bytes used: 188740 (26%)
   average length: 19, compression ratio: 0.90

------------------------------------------------------------------------------
------------------------------------------------------------------------------

 Partitioned Indexes for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 14:11:38.43

------------------------------------------------------------------------------

 Storage analysis for Partitioned Index: I1

-----------------------------------

 Partition P1 in area INDEX1
 Logical area: I1 Logical area id : 60

Larea id: 60,  Record type: 0, Record length: 215, Not Compressed

 Data records: 1, bytes used: 428 (0%)
   average length: 428

-----------------------------------

 Partition P2 in area INDEX2
 Logical area: I1 Logical area id : 61

 Larea id: 61,  Record type: 0, Record length: 215, Not Compressed

 Data records: 1, bytes used: 428 (0%)
   average length: 428
```

```
--------------------------------------------------------------------------------
```

Example 4

In the following example for the PART_DB database with one partitioned table T1 and one partitioned index I1, the /LAREA qualifier is specified to name the index I1. Therefore, only data for the partitioned index I1 is output.

```
$ RMU/ANALYZE/PARTITIONS=INDEXES/LAREA=I1 PART_DB
 Areas containing partitions for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 14:17:47.13

--------------------------------------------------------------------------------

 Storage analysis for storage area: INDEX1 - file: DISK:[DIRECTORY]INDEX1.RDA;1
 Area_id: 3,  Page length: 1024,  Last page: 703

 Bytes free: 685993 (95%), bytes overhead: 33451 (5%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 1, bytes used: 428 (0%)
   average length: 428, compression ratio: 1.00
   index records: 1, bytes used: 428 (0%)
     B-Tree: 428, Hash: 0, Duplicate: 0, Overflow: 0


--------------------------------------------------------------------------------

 Storage analysis for storage area: INDEX2 - file: DISK:[DIRECTORY]INDEX2.RDA;1
 Area_id: 4,  Page length: 1024,  Last page: 703

 Bytes free: 685993 (95%), bytes overhead: 33451 (5%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 1, bytes used: 428 (0%)
   average length: 428, compression ratio: 1.00
   index records: 1, bytes used: 428 (0%)
     B-Tree: 428, Hash: 0, Duplicate: 0, Overflow: 0

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------

 Partitioned Indexes for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 14:17:47.13

--------------------------------------------------------------------------------

 Storage analysis for Partitioned Index: I1

-----------------------------------

 Partition P1 in area INDEX1
 Logical area: I1 Logical area id : 60

 Larea id: 60,  Record type: 0, Record length: 215, Not Compressed

 Data records: 1, bytes used: 428 (0%)
   average length: 428

-----------------------------------
```

```
 Partition P2 in area INDEX2
 Logical area: I1 Logical area id : 61

 Larea id: 61,  Record type: 0, Record length: 215, Not Compressed
 Data records: 1, bytes used: 428 (0%)
   average length: 428

-------------------------------------------------------------------------------
```

Example 5

In the following example for the PART_DB database with one partitioned table T1 and one partitioned index I1, the /LAREA qualifier specifies the logical area identifier number "60". Therefore, only data for the single partition P1 for the index I1 is output.

```
$ RMU/ANALYZE/PARTITIONS/LAREA=60 PART_DB.RDB

 Areas containing partitions for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 15:25:23.68

-------------------------------------------------------------------------------

 Storage analysis for storage area: INDEX1 - file: DISK:[DIRECTORY]INDEX1.RDA;1
 Area_id: 3,  Page length: 1024,  Last page: 703

 Bytes free: 685993 (95%), bytes overhead: 33451 (5%)
 Spam count: 1, AIP count: 0, ABM count: 3
 Data records: 1, bytes used: 428 (0%)
   average length: 428, compression ratio: 1.00
   index records: 1, bytes used: 428 (0%)
     B-Tree: 428, Hash: 0, Duplicate: 0, Overflow: 0

-------------------------------------------------------------------------------
-------------------------------------------------------------------------------

 Partitioned Indexes for database - DISK:[DIRECTORY]PART_DB.RDB;1
 Created  9-JUN-2008 15:25:23.68

-------------------------------------------------------------------------------

 Storage analysis for Partitioned Index: I1

-----------------------------------

 Partition P1 in area INDEX1
 Logical area: I1 Logical area id : 60

 Larea id: 60,  Record type: 0, Record length: 215, Not Compressed

 Data records: 1, bytes used: 428 (0%)
   average length: 428

-------------------------------------------------------------------------------
```

Example 6

The following example shows that if /BINARY_OUTPUT is specified, a PART.UNL file and a PART.RRD file are created that can be used with the RMU/LOAD command to load storage partition data into an Oracle Rdb database table. The PART.RRD file contains the new fields RMU$TABLE_NAME, RMU$INDEX_NAME, RMU$PARTITION_NAME and RMU$ST_AREA_NAME for partition specific data. Note that RMU$AREA_NAME contains the logical area name and RMU$ST_AREA_NAME contains the storage area name that contains the partition or index.

```
$ RMU/ANALYZE/PARTITIONS -
 -$ /BINARY=(FILE=PART.UNL,RECORD=PART.RRD) -
 -$ /OUTPUT=PART.OUT PART_DB
$ TYPE PART.RRD
DEFINE FIELD RMU$DATE DATATYPE IS DATE.
DEFINE FIELD RMU$AREA_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$TABLE_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$INDEX_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$PARTITION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$ST_AREA_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$STORAGE_AREA_ID DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$TOTAL_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$EXPANDED_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$FRAGMENTED_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$EXPANDED_FRAGMENT_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$FRAGMENTED_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$FRAGMENT_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$PAGE_LENGTH DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$MAX_PAGE_NUMBER DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD RMU$FREE_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$OVERHEAD_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$AIP_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$ABM_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$SPAM_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$INDEX_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$BTREE_NODE_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$HASH_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATES_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$OVERFLOW_BYTES DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$LOGICAL_AREA_ID DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$RELATION_ID DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$RECORD_ALLOCATION_SIZE DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$TOTAL_SPACE DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_AREA.
    RMU$DATE.
    RMU$AREA_NAME.
    RMU$TABLE_NAME.
    RMU$INDEX_NAME.
    RMU$PARTITION_NAME.
```

1.6 RMU Analyze Command

```
    RMU$ST_AREA_NAME.
    RMU$STORAGE_AREA_ID.
    RMU$FLAGS.
    RMU$TOTAL_BYTES.
    RMU$EXPANDED_BYTES.
    RMU$FRAGMENTED_BYTES.
    RMU$EXPANDED_FRAGMENT_BYTES.
    RMU$TOTAL_COUNT.
    RMU$FRAGMENTED_COUNT.
    RMU$FRAGMENT_COUNT.
    RMU$PAGE_LENGTH.
    RMU$MAX_PAGE_NUMBER.
    RMU$FREE_BYTES.
    RMU$OVERHEAD_BYTES.
    RMU$AIP_COUNT.
    RMU$ABM_COUNT.
    RMU$SPAM_COUNT.
    RMU$INDEX_COUNT.
    RMU$BTREE_NODE_BYTES.
    RMU$HASH_BYTES.
    RMU$DUPLICATES_BYTES.
    RMU$OVERFLOW_BYTES.
    RMU$LOGICAL_AREA_ID.
    RMU$RELATION_ID.
    RMU$RECORD_ALLOCATION_SIZE.
    RMU$TOTAL_SPACE.
END RMU$ANALYZE_AREA RECORD.
```

1.7 RMU Analyze Cardinality Command

Generates a formatted display of the actual and stored cardinality values for specified tables and indexes. Also, if the stored cardinality values are different from the actual cardinality values, the RMU Analyze Cardinality command allows you to update the stored cardinality values.

Note

Beginning in Oracle Rdb Version 7.0, the RMU Analyze Cardinality command has been deprecated and might be removed in future versions of Oracle Rdb. The features available through this command are now available through the RMU Collect Optimizer_Statistics command and the RMU Show Optimizer_Statistics command.

In addition, updating cardinality information for indexes using the RMU Analyze Cardinality command may cause poor performance because the prefix cardinality information is not collected.

Therefore, Oracle Corporation recommends that you use the RMU Collect Optimizer_Statistics and RMU Show Optimizer_Statistics commands instead of the RMU Analyze Cardinality command.

See Section 1.15 and Section 1.67.7 for information on the RMU Collect Optimizer_Statistics and the RMU Show Optimizer_Statistics commands.

Format

RMU Analyze/Cardinality root-file-spec [table-or-index-name[,...]]

Command Qualifiers	Defaults
/[No]Confirm	/Noconfirm
/Output = file-name	/Output = SYS$OUTPUT
/Transaction_Type=option	/Transaction_Type=Automatic
/[No]Update	/Noupdate

1.7 RMU Analyze Cardinality Command

Description

The actual cardinality values for tables and indexes can be different from the stored cardinality values in your database's RDB$SYSTEM storage area if RDB$SYSTEM has been set to read-only access. When rows are added to or deleted from tables and indexes after the RDB$SYSTEM storage area has been set to read-only access, the cardinality values for these tables and indexes are not updated.

For indexes, the cardinality value is the number of unique entries for an index that allows duplicates. If the index is unique, Oracle Rdb stores zero for the cardinality, and uses the table cardinality instead. For tables, the cardinality value is the number of rows in the table. Oracle Rdb uses the cardinality values of indexes and tables to influence decisions made by the optimizer. If the actual cardinality values of tables and indexes are different from the stored cardinality values, the optimizer's performance can be adversely affected.

When you use the SQL ALTER DATABASE statement to set the RDB$SYSTEM storage area to read-only access for your database, the Oracle Rdb system tables in the RDB$SYSTEM storage area are also set to read-only access. When the Oracle Rdb system tables are set to read-only access:

- Automatic updates to table and index cardinality are disabled.
- Manual changes made to the cardinalities to influence the optimizer are not allowed.
- The I/O associated with the cardinality update is eliminated.

With the RMU Analyze Cardinality command, you can:

- Display the stored and actual cardinality values for the specified tables and indexes.
- Update the stored cardinality value for a specified table or index with either the actual value or an alternative value of your own choosing. Oracle Corporation recommends that you update the stored cardinality value with the actual cardinality value. Specifying a value other than the actual cardinality value can result in poor database performance.

Command Parameters

root-file-spec
The name of the database root file for which you want information. The default file extension is .rdb. This parameter is required.

table-or-index-name[,...]
The name of the table or index for which you want information about cardinality. The default is all tables and all enabled indexes. If you want information about a disabled index, you must specify it by name.

If you do not accept the default and instead specify a table name, the RMU Analyze Cardinality command and any qualifiers you specify will affect only the named table; the command will not result in a display or update (if the Update qualifier is specified) of the indexes associated with the table.

This parameter is optional. An indirect file reference can be used. See Section 1.3 for more information.

Command Qualifiers

Confirm
Noconfirm
Specify the Confirm qualifier with the Update qualifier to gain more control over the update function. When you specify the Confirm qualifier, you are asked whether the update should be performed for each selected table or index whose stored cardinality value is different from its actual cardinality value. You can respond with YES, NO, QUIT, or an alternative value for the stored cardinality.

Specifying YES means that you want to update the stored cardinality with the actual cardinality value. Specifying NO means that you do not want to update the stored cardinality value. Specifying QUIT aborts the RMU Analyze Cardinality command, rolls back any changes you made to stored cardinalities, and returns you to the operating system prompt. Specifying an alternative value updates the stored cardinality value with the alternative value.

When you specify the Noconfirm qualifier, you are not given the option of updating stored cardinality values with an alternative value of your own choosing. Instead, the stored cardinality values that differ from the actual cardinality values are automatically updated with the actual cardinality values.

The default is the Noconfirm qualifier.

1.7 RMU Analyze Cardinality Command

The Confirm and Noconfirm qualifiers are meaningless and are ignored if they are specified without the Update qualifier.

Output=file-name
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Transaction_Type=option
Allows you to specify the transaction mode for the transactions used to perform the analyze operation. Valid options are:

- Automatic
- Read_Only
- Noread_Only

You must specify an option if you use this qualifier.

If you do not specify any form of this qualifier, the Transaction_Type=Automatic qualifier is the default. This qualifier specifies that Oracle RMU is to determine the transaction mode used for the analyze operation. If any storage area in the database (including those not accessed for the analyze operation) has snapshots disabled, the transactions used for the analyze operation are set to read/write mode. Otherwise, the transactions are set to read-only mode.

The Transaction_Type=Read_Only qualifier specifies the transactions used to perform the analyze operation be set to read-only mode. When you explicitly set the transaction type to read-only, snapshots need not be enabled for all storage areas in the database, but must be enabled for those storage areas that are analyzed. Otherwise, you receive an error and the analyze operation fails.

You might select this option if not all storage areas have snapshots enabled and you are analyzing objects that are stored only in storage areas with snapshots enabled. In this case, using the Transaction_Type=Read_Only qualifier allows you to perform the analyze operation and impose minimal locking on other users of the database.

The Transaction_Type=Noread_Only qualifier specifies that the transactions used to for the analyze operation be set to read/write mode. You might select this option if you want to eradicate the growth of snapshot files that occurs during a read-only transaction and are willing to incur the cost of increased locking that occurs during a read/write transaction.

Update
Noupdate

Specify the Update qualifier to update the stored cardinality values of tables and indexes. You can perform an update only when the stored cardinality values differ from the actual cardinality values. When updating cardinality values, Oracle Corporation recommends that you update the stored cardinality values with the actual cardinality values, not with an alternative value of your own choosing. Specifying a value other than the actual cardinality value can result in poor database performance. The default is the Noupdate qualifier.

Using the Update qualifier allows you to update the stored cardinality values of the specified tables and indexes even when the RDB$SYSTEM storage area is designated for read-only access. If you have set the RDB$SYSTEM storage area to read-only access, Oracle RMU sets it to read/write during execution of the RMU Analyze Cardinality command with the Update qualifier. Oracle RMU resets the area to read-only when the operation completes.

If you are updating the stored cardinality for a table or index, and a system failure occurs before the RDB$SYSTEM storage area is changed back to read-only access, use the SQL ALTER DATABASE statement to manually change the database back to read-only access.

However, note that if you have set the area to read-only, the update operation specified with the Update qualifier commences only if the database is off line or the database is quiescent.

If you specify a table name parameter with an RMU Analyze Cardinality command that includes the Update qualifier, the associated indexes are not updated; you must specify each table and index you want to be updated or accept the default (by not specifying any table or index names) and have all items updated.

Oracle Corporation recommends that you use the Update qualifier during offline operations or during a period of low update activity. If you update a cardinality while it is changing (as a result of current database activity), the end result is unpredictable.

Specify the Noupdate qualifier when you want to display the stored and actual cardinality values only for the specified tables and indexes.

1.7 RMU Analyze Cardinality Command

Usage Notes

- To use the RMU Analyze Cardinality command for a database, you must have the RMU$ANALYZE privilege in the root file ACL for the database or the OpenVMS SYSPRV or BYPASS privilege.
- You must have the SQL ALTER privilege for the database to update a read-only RDB$SYSTEM storage area.
- If you specify a name for the table-or-index-name parameter that is both an index name and a table name, the RMU Analyze Cardinality command performs the requested operation for both the table and index.
- Although you can alter the cardinality of a unique index using the RMU Analyze Cardinality command, it has no effect. (A unique index has only unique keys and does not have any duplicate keys.) Because the cardinality of a unique index and the table it indexes are the same, Oracle Rdb uses the table cardinality value when performing operations that involve the cardinality of a unique index. Oracle Rdb does not use the cardinality value stored for a unique index, nor does it attempt to update this value as rows are stored or deleted.
- When the RMU Analyze Cardinality command is issued for a closed database, the command executes without other users being able to attach to the database.

Examples

Example 1

The following command provides information on the cardinality for all indexes and tables in the sample mf_personnel database:

```
$ RMU/ANALYZE/CARDINALITY/NOUPDATE MF_PERSONNEL.RDB /OUTPUT=CARD.LIS
```

Example 2

The following command provides information on the cardinality for the EMPLOYEES table in the mf_personnel database:

```
$ RMU/ANALYZE/CARDINALITY/NOUPDATE MF_PERSONNEL.RDB EMPLOYEES -
_$ /OUTPUT=EMP.LIS
```

1.8 RMU Analyze Indexes Command

Generates a formatted display of statistical information that describes the index structures for the database.

Format

RMU/Analyze/Indexes root-file-spec [index-name[,...]]

Command Qualifiers	Defaults
/[No]Binary_Output[=file-option-list]	/Nobinary_Output
/Exclude = Metadata	All index data displayed
/Option = {Normal | Full | Debug}	/Option=Normal
/Output = file-name	/Output=SYS$OUTPUT
/[No]Partitions	/NoPartitions
/Transaction_Type=option	/Transaction_Type=Automatic

Description

The RMU Analyze Indexes command provides a maintenance tool for analyzing index structures and generates a formatted display of this statistical information. Information is displayed selectively for storage areas and logical areas, or for a range of pages in a storage area. You can use the RMU Analyze Indexes command to analyze the structures of both sorted (including ranked sorted) and hashed indexes. The following shows sample output from the RMU Analyze Index command:

```
$ RMU/ANALYZE/INDEXES MF_PERSONNEL.RDB JH_EMPLOYEE_ID_RANKED
-------------------------------------------------------------------------------

 Indices for database  - RDBVMS_DISK1:[DB]MF_PERSONNEL.RDB;

-------------------------------------------------------------------------------
Index JH_EMPLOYEE_ID_RANKED for relation JOB_HISTORY duplicates allowed
Max Level: 3, Nodes: 34, Used/Avail: 8693/13532 (64%), Keys: 133, Records: 0
 Duplicate nodes:0, Used/Avail: 0/0 (0%), Keys: 100, Maps: 100, Records:4113
 Total Comp/Uncomp IKEY Size: 600/798, Compression Ratio:  .75

-------------------------------------------------------------------------------
```

Data included in the statistics display includes the following information:

- The first line of output identifies the database in which the analyzed index resides.

- The second line of output:
  - Specifies if the index is a hashed index. In the example, the index is not hashed, so the term hashed does not appear.
  - The index name
  - Whether or not duplicates are allowed.
- Third line of output:
  - Max Level

    The maximum number of levels in the index.
  - Nodes

    The total number of nodes in the index.
  - Used/Avail (%)

    The number of bytes used by the index/the number of bytes available. (The percentage of space used by the index.)
  - Keys

    The sum of the dbkeys that point directly to data records plus those that point to duplicate nodes.
  - Records

    The number of data records to which the Keys (in the previous list item) point directly.
- The fourth line of output:
  - Duplicate nodes

    For hashed and nonranked sorted indexes, this is the number of duplicate nodes in the index. For a ranked sorted index, this is the number of overflow nodes. With ranked sorted indexes, Oracle Rdb compresses duplicates using a byte-aligned bitmap compression. It compresses the list of dbkeys that point to duplicates and stores that list in the index key node. Oracle Rdb creates overflow nodes when the compressed list of duplicates does not fit in one index key node. This overflow node contains a bitmap compressed list of dbkeys and pointers to the next overflow node. Therefore, for ranked sorted indexes, the duplicate nodes count (overflow nodes) can be zero (0) if the compressed list of dbkeys that point to duplicates fits into one node.
  - Used/Avail (%)

The number of bytes used by duplicate nodes/number of bytes available in the duplicate nodes. (The percentage of space used within the duplicate nodes of the index.) This value can be zero (0) for a ranked sorted index if the number of duplicate nodes is zero.

- Keys

  The total number of dbkeys that point to a duplicate node or that point to the beginning of a duplicate node chain in the index.

- Maps (appears only if the index is a ranked sorted index)

  The number of duplicate key data record bit maps used by ranked sorted indexes to represent the duplicate index key data record dbkeys.

- Records

  The total number of data records pointed to by duplicate nodes. If the index is a ranked sorted index, Records refers to the number of data records pointed to by duplicate bit maps.

- The fifth line of output (appears only if the index is compressed):
  - Total Comp/Uncomp IKEY Size

    The total byte count of the compressed leaf index keys (level 1 nodes only)/the total byte count that would be consumed if the index were not compressed

  - Compression ratio.

    The calculated ratio of Total Comp/Uncomp. A compression ratio greater than 1.0 indicates that the compressed index keys occupy more space than the uncompressed index keys.

For more information on RMU Analyze Indexes and the display of index keys, refer to the *Oracle Rdb7 Guide to Database Performance and Tuning*.

Command Parameters

root-file-spec
The file specification for the database root file for which you want information. The default file extension is .rdb. This parameter is required.

index-name[,...]
The name of the index for which you want information. The default is all enabled indexes. If you want information about a disabled index, you must specify it by name. This parameter is optional. An indirect file reference can be used. See Section 1.3 for more information.

The wildcard characters "%" and "*" can be used in the index name specification. The following examples demonstrate various combinations of use of the wildcard characters.

```
$ RMU /ANALYZE /INDEX MF_PERSONNEL EMP*
$ RMU /ANALYZE /INDEX MF_PERSONNEL *LAST%NAME
$ RMU /ANALYZE /INDEX MF_PERSONNEL EMP%LAST%NAME
$ RMU /ANALYZE /INDEX MF_PERSONNEL *HASH, *LAST*
```

Command Qualifiers

Binary_Output[=file-option-list]
Nobinary_Output

Specifying the Binary_Output qualifier allows you to store the summary results in a binary file, and to create a record definition file that is compatible with the data dictionary for the binary output file. The binary output can be loaded into an Oracle Rdb database by using the RMU Load command with the Record_Definition qualifier for use by a user-written management application or procedure. The binary output can also be used directly by the user-written application or procedure.

The valid file options are:

- File=file-spec

  The File option causes the RMU Analyze Indexes command data to be stored in an RMS file that contains a fixed-length binary record for each index analyzed.

  The default file extension for the binary output file is .unl. The following command creates the binary output file analyze_out.unl:

  ```
  $ RMU/ANALYZE/INDEXES -
  _$ /BINARY_OUTPUT=FILE=ANALYZE_OUT MF_PERSONNEL.RDB
  ```

- Record_Definition=file-spec

  The Record_Definition option causes the RMU Analyze Indexes command data record definition to be stored in an RMS file. The output file contains the record definition in a subset of the data dictionary command format. The default file extension for the record definition output file is .rrd. Refer to Appendix A for a description of the .rrd files. The following command creates the output file analyze_out.rrd:

  ```
  $ RMU/ANALYZE/INDEXES -
  _$ /BINARY_OUTPUT=RECORD_DEFINITION=ANALYZE_OUT MF_PERSONNEL.RDB
  ```

You can specify both file options in one command by separating them with a comma and enclosing them within parentheses, as follows:

```
$ RMU/ANALYZE/INDEXES/BINARY_OUTPUT= -
_$ (FILE=ANALYZE_OUT,RECORD_DEFINITION=ANALYZE_OUT) -
_$ MF_PERSONNEL.RDB
```

If you specify the Binary_Output qualifier, you must specify at least one of the options. The default is the Nobinary_Output qualifier, which does not create an output file.

Exclude=Metadata

Excludes information from the RMU Analyze Indexes command output. When you specify the Exclude=Metadata qualifier, information on the Oracle Rdb indexes (for example, the RDB$NDX_REL_NAME_NDX and RDB$COLLATIONS_NDX indexes) is excluded from the RMU Analyze Indexes command output. When you do not specify the Exclude qualifier, data is provided for all indexes in the database.

Data is accumulated for the indexes excluded with the Exclude qualifier, but the data is excluded from the RMU Analyze Indexes command output.

You cannot specify the Exclude qualifier and one or more index names in the same RMU Analyze Indexes command.

Option=type

Specifies the type of information and the level of detail the analysis will include. Three types of output are available:

- Normal

  Output includes only summary information. The Normal option is the default.

- Full

  Output includes histograms and summary information. This option displays a summary line for each sorted index level.

- Debug

  Output includes internal information about the data, histograms, and summary information. Note the following when using this option to analyze compressed index keys:

  – The key lengths are from the compressed index keys.

  – The hexadecimal output for the keys is that of the uncompressed index keys.

– The output includes summary statistics about the compressed index keys.

In general, use the Debug option for diagnostic support purposes. You can also use the Debug option to extract data and perform an independent analysis.

Output=file-name
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify a file name.

Partitions
NoPartitions
Allows data to be collected and output for individual index partitions for sorted, sorted ranked and hashed indexes which are partitioned across multiple Oracle Rdb database storage areas. /NoPartitions is the default.

This qualifier will work with other RMU/ANALYZE/INDEXES qualifiers. RMU/ANALYZE/INDEXES/PARTITIONS can be used with /OPTIONS=FULL and /OPTIONS=DEBUG to include data for different numbered levels of individual index partitions and RMU/ANALYZE/INDEXES/PARTITIONS can be used with /BINARY_OUTPUT to output record definition (*.RRD) and unload files (*.UNL) that can be used to load partition data records into an Oracle Rdb database table using the RMU/LOAD command.

If /PARTITIONS is not specified or if /PARTITIONS is specified and an index is not partitioned, only index-wide data will be output by RMU/ANALYZE/INDEXES.

Transaction_Type=option
Allows you to specify the transaction mode for the transactions used to perform the analyze operation. Valid options are:

- Automatic
- Read_Only
- Noread_Only

You must specify an option if you use this qualifier.

If you do not use any form of this qualifier, the Transaction_Type=Automatic qualifier is the default. This qualifier specifies that Oracle RMU is to determine the transaction mode used for the analyze operation. If any storage area in the database (including those not accessed for the analyze operation) has snapshots disabled, the transactions used for the analyze operation are set to read/write mode. Otherwise, the transactions are set to read-only mode.

The Transaction_Type=Read_Only qualifier specifies the transactions used to perform the analyze operation be set to read-only mode. When you explicitly set the transaction type to read-only, snapshots need not be enabled for all storage areas in the database, but must be enabled for those storage areas that are analyzed. Otherwise, you receive an error and the analyze operation fails.

You might select this option if not all storage areas have snapshots enabled and you are analyzing objects that are stored only in storage areas with snapshots enabled. In this case, using the Transaction_Type=Read_Only qualifier allows you to perform the analyze operation and impose minimal locking on other users of the database.

The Transaction_Type=Noread_Only qualifier specifies that the transactions used to for the analyze operation be set to read/write mode. You might select this option if you want to eradicate the growth of snapshot files that occurs during a read-only transaction and are willing to incur the cost of increased locking that occurs during a read/write transaction.

Usage Notes

- To use the RMU Analyze Indexes command for a database, you must have the RMU$ANALYZE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- When the RMU Analyze Indexes command is issued for a closed database, the command executes without other users being able to attach to the database.
- The following RMU Analyze Indexes command produces an RMS record definition file called index.rrd that can be read by the RMU Load command and the data dictionary:

1.8 RMU Analyze Indexes Command

```
$ RMU/ANALYZE/INDEX/BINARY_OUTPUT=RECORD_DEFINITION=INDEX.RRD -
_$ MF_PERSONNEL
$!
$! Display the index.rrd file created by the previous command:
$ TYPE INDEX.RRD

DEFINE FIELD RMU$DATE DATATYPE IS DATE.
DEFINE FIELD RMU$INDEX_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$RELATION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$LEVEL DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$USED DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$AVAILABLE DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_USED DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_AVAILABLE DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_COMP_IKEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_IKEY_COUNT DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_INDEX.
```

- The following list describes each of the fields in the index.rrd record definition:
  - RMU$DATE

    Contains the date that the analyze operation was done
  - RMU$INDEX_NAME

    Contains the name of the index that was analyzed
  - RMU$RELATION_NAME

    Contains the name of the table for which the index is defined
  - RMU$LEVEL

    Contains the maximum number of index levels
  - RMU$FLAGS

    The eight possible values in this field have the following meanings:
    * 0—Index is sorted and not unique. A full report is not generated.
    * 1—Index is sorted and unique. A full report is not generated.
    * 2—Index is hashed and not unique. A full report is not generated.
    * 3—Index is hashed and unique. A full report is not generated.

* 4—Index is sorted and not unique. A full report is generated.
* 5– Index is sorted and unique. A full report is generated.
* 6– Index is hashed and not unique. A full report is generated.
* 7—Index is hashed and unique. A full report is generated.
* 8—Index is sorted ranked and not unique. A full report is not generated.
* 9—Index is sorted ranked and unique. A full report is not generated.
* 12—Index is sorted ranked and not unique. A full report is generated.
* 13—Index is sorted ranked and unique. A full report is generated.

The RMU Analyze Indexes command uses the RMU$FLAGS bits shown in Table 1–2 for describing specific index information.

Table 1–2 RMU$FLAGS Bits Used by the RMU Analyze Indexes Command

Bit Offset	Meaning
0	Unique index if true
1	Hashed index if true
2	Full report record if true
3	Ranked index if true

When RMU$FLAGS has bit 2 set it means that a full report is generated. A full report has records for each level of the index.

1.8 RMU Analyze Indexes Command

- RMU$COUNT

  Contains the number of index nodes
- RMU$USED

  Contains the amount of available space that is used
- RMU$AVAILABLE

  Contains the amount of space available in the index records initially
- RMU$DUPLICATE_COUNT

  Contains the number of duplicate records
- RMU$DUPLICATE_USED

  Contains the amount of available space used in the duplicate records
- RMU$DUPLICATE_AVAILABLE

  Contains the amount of space available in the duplicate records initially
- RMU$KEY_COUNT

  Contains the number of keys
- RMU$DATA_COUNT

  Contains the number of records
- RMU$DUPLICATE_KEY_COUNT

  Contains the number of duplicate keys
- RMU$DUPLICATE_DATA_COUNT

  Contains the number of duplicate records
- RMU$TOTAL_COMP_IKEY_COUNT

  Contains the number of compressed index key bytes
- RMU$TOTAL_IKEY_COUNT

  Contains the number of bytes that would be used by index keys, had they not been compressed

Examples

Example 1

The following command analyzes the JH_EMPLOYEE_ID and SH_ EMPLOYEE_ID indexes in the mf_personnel database:

```
$ RMU/ANALYZE/INDEXES MF_PERSONNEL.RDB JH_EMPLOYEE_ID,SH_EMPLOYEE_ID -
_$ /OUTPUT=EMP_ID_INDEX.LIS
```

Example 2

The following commands demonstrate the differences you see when you analyze a nonranked sorted index and a ranked sorted index. Note the differences in the values for the Duplicate nodes. The nonranked sorted index displays 80 duplicate nodes. The ranked sorted index (before more duplicates are added) displays 0 duplicate nodes for the same data. After hundreds of more duplicates are added, the ranked sorted index shows only 3 duplicate nodes. The differences you see are because of the different way duplicate records are stored for nonranked sorted indexes and ranked sorted indexes. See the Description section under this command for details on these differences.

```
$ ! Analyze a nonranked sorted index:
$ !
$ RMU/ANALYZE/INDEXES MF_PERSONNEL.RDB JH_EMPLOYEE_ID
-------------------------------------------------------------------------------

 Indices for database  - USER1:[DB]MF_PERSONNEL.RDB;1

-------------------------------------------------------------------------------
Index JH_EMPLOYEE_ID for relation JOB_HISTORY duplicates allowed
Max Level: 2, Nodes: 4, Used/Avail: 768/1592 (48%), Keys: 103, Records: 20
Duplicate nodes: 80, Used/Avail: 2032/4696 (43%), Keys: 80, Records: 254
-------------------------------------------------------------------------------
$ ! Analyze a ranked sorted index defined on the same column as the
$ ! nonranked sorted index:

$ RMU/ANALYZE/INDEXES MF_PERSONNEL.RDB JH_EMPLOYEE_ID_RANKED
-------------------------------------------------------------------------------

 Indices for database  - USER1:[DB]MF_PERSONNEL.RDB;1
```

```
-------------------------------------------------------------------------------
Index JH_EMPLOYEE_ID_RANKED for relation JOB_HISTORY duplicates allowed
Max Level: 2, Nodes: 11, Used/Avail: 2318/4378 (53%), Keys: 110, Records: 20
Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 80, Maps: 80, Records: 254
-------------------------------------------------------------------------------
$ !
$ ! Insert many duplicates and analyze the ranked sorted index again:
$ !
$ RMU/ANALYZE/INDEXES MF_PERSONNEL.RDB JH_EMPLOYEE_ID_RANKED
-------------------------------------------------------------------------------

 Indices for database  - USER1:[DB]MF_PERSONNEL.RDB;1

-------------------------------------------------------------------------------
Index JH_EMPLOYEE_ID_RANKED for relation JOB_HISTORY duplicates allowed
Max Level: 2, Nodes: 13, Used/Avail: 2705/5174 (52%), Keys: 112, Records: 20
Duplicate nodes:3, Used/Avail:850/1194 (71%), Keys:80, Maps: 83, Records:2964
-------------------------------------------------------------------------------
```

Example 3

The following example shows that only index-wide data is output for index I1 in database PART_IND_DB.RDB if /PARTITIONS is not specified.

```
$ RMU/ANALYZE/INDEX PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx

-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Max Level: 3, Nodes: 898, Used/Avail: 194731/357404 (54%), Keys: 10892,
 Records: 10000
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

-------------------------------------------------------------------------------
```

Example 4

The following example shows that data for each index partition is output following the index-wide data for index I1 in database PART_IND_DB.RDB if /PARTITIONS is specified. The partition names are P1 through P6 and the storage area names for the partitions are INDEX1 through INDEX6. The partition data is sorted by partition name.

```
$ RMU/ANALYZE/INDEX/PARTITION PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx
```

```
-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Max Level: 3, Nodes: 898, Used/Avail: 194731/357404 (54%), Keys: 10892,
 Records: 10000
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P1 in area INDEX1
 Max Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P2 in area INDEX2
 Max Level: 1, Nodes: 1, Used/Avail: 21/398 (5%), Keys: 1, Records: 1
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P3 in area INDEX3
 Max Level: 1, Nodes: 1, Used/Avail: 72/398 (18%), Keys: 4, Records: 4
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P4 in area INDEX4
 Max Level: 2, Nodes: 9, Used/Avail: 1789/3582 (49%), Keys: 103, Records: 95
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P5 in area INDEX5
 Max Level: 3, Nodes: 885, Used/Avail: 192849/352230 (54%), Keys: 10784,
 Records: 9900
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

 Partition P6 in area INDEX6
 Max Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

-------------------------------------------------------------------------------
```

Example 5

The following example shows that data for each level within each index partition is output following the index-wide data for index I1 in database PART_IND_DB.RDB if /OPTION=FULL is specified. The partition names are P1 through P6 and the storage area names for the partitions are INDEX1 through INDEX6. The levels are numbered in descending order.

```
$ RMU/ANALYZE/INDEX/PARTITION/OPTION=FULL PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx

-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Max Level: 3, Nodes: 898, Used/Avail: 194731/357404 (54%), Keys: 10892,
     Records: 10000
     Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0
```

```
    Level: 3, Nodes: 1, Used/Avail: 184/398 (46%), Keys: 23, Records: 0
    Level: 2, Nodes: 24, Used/Avail: 6197/9552 (64%), Keys: 869, Records: 0
    Level: 1, Nodes: 873, Used/Avail: 188350/347454 (54%), Keys: 10000,
    Records: 10000

Partition P1 in area INDEX1
Max Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0

Partition P2 in area INDEX2
Max Level: 1, Nodes: 1, Used/Avail: 21/398 (5%), Keys: 1, Records: 1
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 1, Nodes: 1, Used/Avail: 21/398 (5%), Keys: 1, Records: 1

Partition P3 in area INDEX3
Max Level: 1, Nodes: 1, Used/Avail: 72/398 (18%), Keys: 4, Records: 4
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 1, Nodes: 1, Used/Avail: 72/398 (18%), Keys: 4, Records: 4

Partition P4 in area INDEX4
Max Level: 2, Nodes: 9, Used/Avail: 1789/3582 (49%), Keys: 103, Records: 95
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 2, Nodes: 1, Used/Avail: 56/398 (14%), Keys: 8, Records: 0
    Level: 1, Nodes: 8, Used/Avail: 1733/3184 (54%), Keys: 95, Records: 95

Partition P5 in area INDEX5
Max Level: 3, Nodes: 885, Used/Avail: 192849/352230 (54%), Keys: 10784,
    Records: 9900
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 3, Nodes: 1, Used/Avail: 184/398 (46%), Keys: 23, Records: 0
    Level: 2, Nodes: 23, Used/Avail: 6141/9154 (67%), Keys: 861, Records: 0
    Level: 1, Nodes: 861, Used/Avail: 186524/342678 (54%), Keys: 9900,
    Records: 9900

Partition P6 in area INDEX6
Max Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0
    Duplicate nodes: 0, Used/Avail: 0/0 (0%), Keys: 0, Records: 0

    Level: 1, Nodes: 1, Used/Avail: 0/398 (0%), Keys: 0, Records: 0
```

Example 6

The following example shows that if /BINARY_OUTPUT is specified, a PARTIND.UNL file and a PARTIND.RRD file are created that can be used with the RMU/LOAD command to load partition data for the index I1 into an Oracle Rdb database table. The PARTIND.RRD file contains the fields RMU$PARTITION_NAME and RMU$AREA_NAME which will contain the

partition name and area name for each record in the PARTIND.UNL file which contains partition specific data.

```
$ RMU/ANALYZE/INDEX/PARTITION/BINARY=(FILE=PARTIND.UNL,RECORD=PARTIND.RRD) -
 /OUTPUT=I1.OUT PART_IND_DB I1
$ TYPE PARTIND.RRD
DEFINE FIELD RMU$DATE DATATYPE IS DATE.
DEFINE FIELD RMU$INDEX_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$RELATION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$PARTITION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$AREA_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$LEVEL DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$USED DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$AVAILABLE DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_MAP DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_USED DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_AVAILABLE DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_COMPRESSED_IKEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_IKEY_COUNT DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_INDEX.
    RMU$DATE.
    RMU$INDEX_NAME.
    RMU$RELATION_NAME.
    RMU$PARTITION_NAME.
    RMU$AREA_NAME.
    RMU$LEVEL.
    RMU$FLAGS.
    RMU$COUNT.
    RMU$USED.
    RMU$AVAILABLE.
    RMU$DUPLICATE_COUNT.
    RMU$DUPLICATE_MAP.
    RMU$DUPLICATE_USED.
    RMU$DUPLICATE_AVAILABLE.
    RMU$KEY_COUNT.
    RMU$DATA_COUNT.
    RMU$DUPLICATE_KEY_COUNT.
    RMU$DUPLICATE_DATA_COUNT.
    RMU$TOTAL_COMPRESSED_IKEY_COUNT.
    RMU$TOTAL_IKEY_COUNT.
END RMU$ANALYZE_INDEX RECORD.
```

1.9 RMU Analyze Placement Command

Generates a formatted display of statistical information describing the row placement relative to the index structures for the database.

Format

RMU/Analyze/Placement root-file-spec [index-name[,...]]

Command Qualifiers	Defaults
/Areas[=storage-area-list]	/Areas
/[No]Binary_Output[=file-option-list]	/Nobinary_Output
/Exclude = Metadata	All index data displayed
/Option = {Normal | Full | Debug}	/Option = Normal
/Output=file-name	/Output = SYS$OUTPUT
/[No]Partitions	/NoPartitions
/Transaction_Type=option	/Transaction_Type=Automatic

Description

The RMU Analyze Placement command provides a maintenance tool for analyzing row placement relative to index structures and generates a formatted display of this statistical information. Information is displayed selectively for any specified storage area.

You can use the RMU Analyze Placement command to determine:

- The maximum and average path length to a data record. (The maximum and average number of records touched to reach a data record.)
- The estimated maximum I/O path length to a data record.
- The estimated minimum I/O path length to a data record.
- The frequency distributions for the database key (dbkey) path lengths, maximum I/O path lengths, and minimum I/O path lengths for specified indexes.
- The distribution of data records on data pages in a storage area by logical area identifier (ID) and dbkey, the number of dbkeys needed to reach each data record, the maximum and minimum I/O path lengths needed to reach the data record, and the specific dbkey for the data record.

Command Parameters

root-file-spec
The file specification for the database root file to be analyzed. The default file extension is .rdb.

index-name[,...]
The name of the index for which you want information. The default is all enabled indexes. If you want information about a disabled index, you must specify it by name. This parameter is optional. An indirect file reference can be used.

Command Qualifiers

Areas[=storage-area-list]
Specifies the storage areas to be analyzed. You can specify each storage area by name or by the area's ID number.

If you are interested in the placement information for a particular index, specify the area where the data resides, not where the index resides. For example, if you are interested in the placement information for the SH_EMPLOYEE_ID index of the mf_personnel database, you should specify SALARY_HISTORY as the storage area (which is where the data resides), not RDB$SYSTEM (which is where the index resides).

If you do not specify the Areas qualifier, or if you specify the Areas qualifier but do not provide a storage-area-list, information for all the storage areas is displayed.

If you specify more than one storage area, separate the storage area names or ID numbers in the storage-area-list with a comma and enclose the list within parentheses.

If you specify more than one storage area with the Areas qualifier, the analysis Oracle RMU provides is a summary for all the specified areas. The analysis is not broken out into separate sections for each specified storage area. To get index information for a specific storage area, issue the RMU Analyze Placement command, specifying only that area with the Areas qualifier.

The Areas qualifier can be used with an indirect file reference. See Section 1.3 for more information.

The Areas qualifier (without a storage-area-list) is the default.

1.9 RMU Analyze Placement Command

Binary_Output[=file-option-list]
Nobinary_Output
Specifying the Binary_Output qualifier allows you to store the summary results in a binary file, and to create a record definition file that is compatible with the data dictionary for the binary output file. The binary output file can be loaded into an Oracle Rdb database by using the RMU Load command with the Record_Definition qualifier that can then be used by a user-written management application or procedure. The binary output can also be used directly by the user-written application or procedure.

The valid file options are:

- File=file-spec

  The File option causes the RMU Analyze Placement command data to be stored in an RMS file that contains a fixed-length binary record for each index analyzed. The default file extension for the binary output file is .unl. The following command creates the binary output file analyze_out.unl:

  ```
  $ RMU/ANALYZE/PLACEMENT -
  _$ /BINARY_OUTPUT=FILE=ANALYZE_OUT MF_PERSONNEL.RDB
  ```

- Record_Definition=file-spec

  The Record_Definition option causes the RMU Analyze Placement command data record definition to be stored in an RMS file. The output file contains the record definition in a subset of the data dictionary command format. The default file extension for the record definition output file is .rrd. Refer to Appendix A for a description of .rrd files. The following command creates the output file analyze_out.rrd:

  ```
  $ RMU/ANALYZE/PLACEMENT -
  _$ /BINARY_OUTPUT=RECORD_DEFINITION=ANALYZE_OUT MF_PERSONNEL.RDB
  ```

  You can specify both file options in one command by separating them with a comma and enclosing them within parentheses, as follows:

  ```
  $ RMU/ANALYZE/PLACEMENT/BINARY_OUTPUT= -
  _$ (FILE=ANALYZE_OUT,RECORD_DEFINITION=ANALYZE_OUT) -
  _$ MF_PERSONNEL.RDB
  ```

The default is the Nobinary_Output qualifier, which does not create an output file.

Exclude=Metadata
Excludes information from the RMU Analyze Placement command data. When you specify the Exclude=Metadata qualifier, information on all the Oracle Rdb indexes (for example, the RDB$NDX_REL_NAME_NDX and RDB$COLLATIONS_NDX indexes) is excluded from the RMU Analyze

Placement command output. When you do not specify the Exclude qualifier, data is provided for all indexes in the database.

Data is accumulated for the indexes excluded with the Exclude qualifier, but the data is excluded from the RMU Analyze Placement command output.

You cannot specify the Exclude qualifier and one or more index names in the same RMU Analyze Placement command.

Option=type
Specifies the type of information and level of detail the analysis will include. Three types of output are available:

- Normal

  Output includes only summary information. Normal is the default.

- Full

  Output includes histograms and summary information.

- Debug

  Output includes internal information about the data, histograms, and summary information. Output also displays uncompressed index keys from compressed indexes. The hexadecimal output is that of the uncompressed index key. However, the lengths shown are of the compressed index key. For more information on RMU Analyze Placement and the display of index keys, refer to the *Oracle Rdb7 Guide to Database Performance and Tuning*.

Output=file-name
Specifies the name of the file where output will be sent. The default file type is .lis. If you do not specify the Output qualifier, the default output is SYS$OUTPUT.

Partitions
NoPartitions
Collects and displays statistical information describing table row placement relative to the index structure for an Oracle Rdb database. This allows data to be collected and output for individual index partitions for sorted, sorted ranked and hashed indexes which are partitoned across multiple Oracle Rdb database storage areas.

The /PARTITIONS qualifier will work with existing RMU/ANALYZE/PLACEMENT qualifiers. RMU/ANALYZE/PLACEMENT/PARTITIONS can be used with /OPTIONS=FULL and /OPTIONS=DEBUG to include histogram displays for individual index partitions and RMU/ANALYZE/PLACEMENT/PARTITIONS can be used with /BINARY_OUTPUT to output record definition (*.RRD) and

unload files (*.UNL) that can be used to load partition placement data records into an Oracle Rdb database table using the RMU/LOAD command.

If /PARTITIONS is not specified or if /PARTITIONS is specified and an index is not partitioned, only index-wide placement data will be output by RMU/ANALYZE/PLACEMENT.

/NOPARTITIONS is the default.

Transaction_Type=option
Allows you to specify the transaction mode for the transactions used to perform the analyze operation. Valid options are:

- Automatic
- Read_Only
- Noread_Only

You must specify an option if you use this qualifier.

If you do not use any form of this qualifier, the Transaction_Type=Automatic qualifier is the default. This qualifier specifies that Oracle RMU is to determine the transaction mode used for the analyze operation. If any storage area in the database (including those not accessed for the analyze operation) has snapshots disabled, the transactions used for the analyze operation are set to read/write mode. Otherwise, the transactions are set to read-only mode.

The Transaction_Type=Read_Only qualifier specifies the transactions used to perform the analyze operation be set to read-only mode. When you explicitly set the transaction type to read-only, snapshots need not be enabled for all storage areas in the database, but must be enabled for those storage areas that are analyzed. Otherwise, you receive an error and the analyze operation fails.

You might select this option if not all storage areas have snapshots enabled and you are analyzing objects that are stored only in storage areas with snapshots enabled. In this case, using the Transaction_Type=Read_Only qualifier allows you to perform the analyze operation and impose minimal locking on other users of the database.

The Transaction_Type=Noread_Only qualifier specifies that the transactions used for the analyze operation be set to read/write mode. You might select this option if you want to eradicate the growth of snapshot files that occurs during a read-only transaction and are willing to incur the cost of increased locking that occurs during a read/write transaction.

Usage Notes

- To use the RMU Analyze Placement command for a database, you must have the RMU$ANALYZE privilege in the root file ACL for the database or the OpenVMS SYSPRV or BYPASS privilege.
- When the RMU Analyze Placement command is issued for a closed database, the command executes without other users being able to attach to the database.
- The following RMU Analyze Placement command directs the results into an RMS record definition file called placement.rrd that is compatible with the data dictionary:

```
$ RMU/ANALYZE/PLACEMENT/BINARY_OUTPUT=RECORD_DEFINITION=PLACEMENT.RRD -
_$ MF_PERSONNEL
$!
$! Display the placement.rrd file created by the previous command:
$ TYPE PLACEMENT.RRD

DEFINE FIELD RMU$DATE DATATYPE IS DATE.
DEFINE FIELD RMU$INDEX_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$RELATION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$LEVEL DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_KEY_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_PAGE_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_BUFFER_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MAX_KEY_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MAX_PAGE_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MIN_BUF_PATH DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_PLACEMENT.
```

- The following list describes each of the fields in the placement.rrd record definition:
  - RMU$DATE

    Contains the date that the analyze operation was done
  - RMU$INDEX_NAME

    Contains the name of the index that was analyzed
  - RMU$RELATION_NAME

Contains the name of the table for which the index is defined

- RMU$LEVEL

  Contains the maximum number of index levels

- RMU$FLAGS

  The six possible values in this field have the following meanings:

  * 0—Index is a sorted and not unique index
  * 1—Index is sorted and unique
  * 2—Index is hashed and not unique
  * 3—Index is hashed and unique
  * 4—Index is ranked sorted and not unique
  * 5—Index is ranked sorted and unique

  The RMU Analyze Placement command uses the RMU$FLAGS bits shown in Table 1–3 for describing specific index information.

Table 1–3 RMU$FLAGS Bits Used by the RMU Analyze Placement Command

Bit Offset	Meaning
0	Unique index if true
1	Hashed index if true
2	Ranked sorted index if true

- RMU$COUNT

  Contains the number of index nodes

- RMU$DUPLICATE_COUNT

  Contains the number of duplicate records

- RMU$KEY_COUNT

  Contains the number of keys

- RMU$DUPLICATE_KEY_COUNT

  Contains the number of duplicate keys

- RMU$DATA_COUNT

  Contains the number of records

- RMU$DUPLICATE_DATA_COUNT

Contains the number of duplicate records

- RMU$TOTAL_KEY_PATH

  Contains the total number of keys touched to access all the records

- RMU$TOTAL_PAGE_PATH

  Contains the total number of pages touched to access all the records

- RMU$TOTAL_BUFFER_PATH

  Contains the total number of buffers touched to access all the records

- RMU$MAX_KEY_PATH

  Contains the largest number of keys touched to access any of the records

- RMU$MAX_PAGE_PATH

  Contains the largest number of pages touched to access any of the records

- RMU$MIN_BUF_PATH

  Contains the smallest number of buffers touched to access any of the records

Examples

Example 1

The following command provides information on row storage relative to the DEPARTMENTS_INDEX index of the sample personnel database:

```
$ RMU/ANALYZE/PLACEMENT MF_PERSONNEL.RDB DEPARTMENTS_INDEX

------------------------------------------------------------------------

 Indices for database  - DISK1:[DB]MF_PERSONNEL.RDB;

------------------------------------------------------------------------
 Index DEPARTMENTS_INDEX for relation DEPARTMENTS duplicates not allowed
 Levels: 1, Nodes: 1, Keys: 26, Records: 26
 Maximum path length --  DBkeys: 2, IO range: 1 to 2
 Average path length --  DBkeys: 2.00, IO range: 1.00 to 1.65

------------------------------------------------------------------------
```

Example 2

The following example shows that only index wide placement data is output for index I1 in database PART_IND_DB.RDB if /PARTITIONS is not specified.

```
$ RMU/ANALYZE/PLACEMENT PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx

-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Levels: 3, Nodes: 898, Keys: 10892, Records: 10000
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 4, IO range: 2 to 4
 Average path length --  dbkeys: 3.99, IO range: 3.96 to 3.96

-------------------------------------------------------------------------------
```

Example 3

The following example shows that placement data for each index partition is output following the index wide placement data for index I1 in database PART_IND_DB.RDB if /PARTITIONS is specified. The partition names are P1 through P6 and the storage area names for the partitions are INDEX1 through INDEX6. The partition data is sorted by partition name.

```
$ RMU/ANALYZE/PLACEMENT/PARTITION PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx

-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Levels: 3, Nodes: 898, Keys: 10892, Records: 10000
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 4, IO range: 2 to 4
 Average path length --  dbkeys: 3.99, IO range: 3.96 to 3.96

 Partition P1 in area INDEX1
 Levels: 1, Nodes: 1, Keys: 0, Records: 0
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 0, IO range: 0 to 0
 Average path length --  dbkeys: 0.00, IO range: 0.00 to 0.00

 Partition P2 in area INDEX2
Levels: 1, Nodes: 1, Keys: 1, Records: 1
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 2, IO range: 2 to 2
 Average path length --  dbkeys: 2.00, IO range: 2.00 to 2.00
```

```
 Partition P3 in area INDEX3
 Levels: 1, Nodes: 1, Keys: 4, Records: 4
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 2, IO range: 2 to 2
 Average path length --  dbkeys: 2.00, IO range: 2.00 to 2.00

 Partition P4 in area INDEX4
 Levels: 2, Nodes: 9, Keys: 103, Records: 95
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 3, IO range: 2 to 3
 Average path length --  dbkeys: 3.00, IO range: 2.87 to 2.87

 Partition P5 in area INDEX5
 Levels: 3, Nodes: 885, Keys: 10784, Records: 9900
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 4, IO range: 3 to 4
 Average path length --  dbkeys: 4.00, IO range: 3.97 to 3.97

 Partition P6 in area INDEX6
Levels: 1, Nodes: 1, Keys: 0, Records: 0
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 0, IO range: 0 to 0
 Average path length --  dbkeys: 0.00, IO range: 0.00 to 0.00

-------------------------------------------------------------------------------
```

Example 4

The following example shows that placement data and histograms for each index partition are output following the index wide placement data and histograms for index I1 in database PART_IND_DB.RDB if /OPTION=FULL is specified. The partition names are P1 through P6 and the storage area names for the partitions are INDEX1 through INDEX6.

```
$ RMU/ANALYZE/PLACEMENT/PARTITION/OPTION=FULL PART_IND_DB I1
-------------------------------------------------------------------------------

 Indices for database  - DISK:[DIRECTORY]PART_IND_DB.RDB;
 Created dd-mmm-yyyy hh:mm:ss.xxxx

-------------------------------------------------------------------------------
 Index I1 for relation T1 duplicates allowed
 Levels: 3, Nodes: 898, Keys: 10892, Records: 10000
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 4, IO range: 2 to 4
 Average path length --  dbkeys: 3.99, IO range: 3.96 to 3.96

                       dbkey path length vs. frequency
```

```
 6 | (0)
 5 | (0)
 4 |=================================================== (9900)
 3 | (95)
 2 | (5)
 1 | (0)


                 MAX IO path length vs. frequency
 6 | (0)
 5 | (0)
 4 |=================================================== (9624)
 3 |== (359)
 2 | (17)
 1 | (0)


                 MIN IO path length vs. frequency
 6 | (0)
 5 | (0)
 4 |=================================================== (9624)
 3 |== (359)
 2 | (17)
 1 | (0)


Partition P1 in area INDEX1
Levels: 1, Nodes: 1, Keys: 0, Records: 0
  Dup nodes: 0, Dup keys: 0, Dup records: 0
Maximum path length --  dbkeys: 0, IO range: 0 to 0
Average path length --  dbkeys: 0.00, IO range: 0.00 to 0.00


Partition P2 in area INDEX2
Levels: 1, Nodes: 1, Keys: 1, Records: 1
  Dup nodes: 0, Dup keys: 0, Dup records: 0
Maximum path length --  dbkeys: 2, IO range: 2 to 2
Average path length --  dbkeys: 2.00, IO range: 2.00 to 2.00


                  dbkey path length vs. frequency
 6 | (0)
 5 | (0)
 4 | (0)
 3 | (0)
 2 |=================================================== (1)
 1 | (0)


                 MAX IO path length vs. frequency
```

```
    6 | (0)
    5 | (0)
    4 | (0)
    3 | (0)
    2 |================================================== (1)
    1 | (0)


                    MIN IO path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
   3  | (0)
    2 |================================================== (1)
    1 | (0)


Partition P3 in area INDEX3
Levels: 1, Nodes: 1, Keys: 4, Records: 4
  Dup nodes: 0, Dup keys: 0, Dup records: 0
Maximum path length --  dbkeys: 2, IO range: 2 to 2
Average path length --  dbkeys: 2.00, IO range: 2.00 to 2.00

                     dbkey path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
    3 | (0)
    2 |================================================== (4)
    1 | (0)


                    MAX IO path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
    3 | (0)
    2 |================================================== (4)
    1 | (0)


                    MIN IO path length vs. frequency
```

```
    6 | (0)
    5 | (0)
    4 | (0)
    3 | (0)
    2 |=================================================== (4)
    1 | (0)


Partition P4 in area INDEX4
Levels: 2, Nodes: 9, Keys: 103, Records: 95
  Dup nodes: 0, Dup keys: 0, Dup records: 0
Maximum path length --  dbkeys: 3, IO range: 2 to 3
Average path length --  dbkeys: 3.00, IO range: 2.87 to 2.87

                     dbkey path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
    3 |=================================================== (95)
    2 | (0)
    1 | (0)


                    MAX IO path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
    3 |=================================================== (83)
    2 |======= (12)
    1 | (0)


                    MIN IO path length vs. frequency
    6 | (0)
    5 | (0)
    4 | (0)
    3 |=================================================== (83)
    2 |======= (12)
    1 | (0)


Partition P5 in area INDEX5
Levels: 3, Nodes: 885, Keys: 10784, Records: 9900
  Dup nodes: 0, Dup keys: 0, Dup records: 0
Maximum path length --  dbkeys: 4, IO range: 3 to 4
Average path length --  dbkeys: 4.00, IO range: 3.97 to 3.97

                     dbkey path length vs. frequency
```

```
6 | (0)
5 | (0)
4 |================================================== (9900)
3 | (0)
2 | (0)
1 | (0)

                  MAX IO path length vs. frequency

6 | (0)
5 | (0)
4 |================================================== (9624)
3 |= (276)
2 | (0)
1 | (0)


                  MIN IO path length vs. frequency

6 | (0)
5 | (0)
4 |================================================== (9624)
3 |= (276)
2 | (0)
1 | (0)


 Partition P6 in area INDEX6
 Levels: 1, Nodes: 1, Keys: 0, Records: 0
   Dup nodes: 0, Dup keys: 0, Dup records: 0
 Maximum path length --  dbkeys: 0, IO range: 0 to 0
 Average path length --  dbkeys: 0.00, IO range: 0.00 to 0.00
```

Example 5

The following example shows that if /BINARY_OUTPUT is specified, a PARTIND.UNL file and a PARTIND.RRD file are created that can be used with the RMU/LOAD command to load partition placement data for the index I1 into an Oracle Rdb database table. The PARTIND.RRD file contains the fields RMU$PARTITION_NAME and RMU$AREA_NAME which will contain the partition name and area name for each record in the PARTIND.UNL file which contains partition specific data.

1.9 RMU Analyze Placement Command

```
$ RMU/ANALYZE/PARTITION/PARTITION/BINARY=(FILE=PARTIND.UNL,RECORD=PARTIND.RRD) -
 /OUTPUT=I1.OUT PART_IND_DB I1
$ TYPE PARTIND.RRD
DEFINE FIELD RMU$DATE DATATYPE IS DATE.
DEFINE FIELD RMU$INDEX_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$RELATION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$PARTITION_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$AREA_NAME DATATYPE IS TEXT SIZE IS 32.
DEFINE FIELD RMU$LEVEL DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$FLAGS DATATYPE IS SIGNED WORD.
DEFINE FIELD RMU$COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_MAP_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_KEY_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$DUPLICATE_DATA_COUNT DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_KEY_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_PAGE_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$TOTAL_BUFFER_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MAX_KEY_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MAX_PAGE_PATH DATATYPE IS F_FLOATING.
DEFINE FIELD RMU$MIN_BUF_PATH DATATYPE IS F_FLOATING.
DEFINE RECORD RMU$ANALYZE_PLACEMENT.
    RMU$DATE.
    RMU$INDEX_NAME.
    RMU$RELATION_NAME.
    RMU$PARTITION_NAME.
    RMU$AREA_NAME.
    RMU$LEVEL.
    RMU$FLAGS.
    RMU$COUNT.
    RMU$DUPLICATE_COUNT.
    RMU$DUPLICATE_MAP_COUNT.
    RMU$KEY_COUNT.
    RMU$DUPLICATE_KEY_COUNT.
    RMU$DATA_COUNT.
   RMU$DUPLICATE_DATA_COUNT.
    RMU$TOTAL_KEY_PATH.
    RMU$TOTAL_PAGE_PATH.
    RMU$TOTAL_BUFFER_PATH.
    RMU$MAX_KEY_PATH.
    RMU$MAX_PAGE_PATH.
    RMU$MIN_BUF_PATH.
END RMU$ANALYZE_PLACEMENT RECORD.
```

1.10 RMU Backup Command

Creates a backup copy of the database and places it in a file. If necessary, you can later use the RMU Restore command to restore the database to the condition it was in at the time of the backup operation.

Format

RMU/Backup root-file-spec backup-file-spec

Command Qualifiers	Defaults
/[No]Accept_Label	/Noaccept_Label
/[No]Acl	/Acl
/Active_IO=max-writes	/Active_IO=3
/Allocation=blocks	None
/[No]Assist	See description
/Block_Size=integer	See description
/[No]Checksum_Verification	/Checksum_Verification
/[No]Compression[=options]	/Nocompression
/Crc[=Autodin_II]	See description
/Crc=Checksum	See description
/Nocrc	See description
/[No]Database_Verification	/Database_Verification
/Density=(density-value,[No]Compaction)	See description
/Disk_File[=options]	None
/Encrypt=({Value=lName=}[,Algorithm=])	See description
/Exclude[=storage-area[,...]]	See description
/[No]Execute	See description
/Extend_Quantity=number-blocks	/Extend_Quantity=2048
/[No]Group_Size=interval	See description

1.10 RMU Backup Command

/Include[=storage-area[,...]]	See description
/[No]Incremental	/Noincremental
/Incremental={By_area|Complete}	None
/Journal=file-name	See description
/Label=(label-name-list)	See description
/Librarian[=options]	None
/List_Plan=output-file	See description
/Loader_Synchronization[=Fixed]	See description
/Lock_Timeout[=seconds]	See description
/[No]Log[=Brief|Full]	Current DCL verify switch
/Master	See description
/[No]Media_Loader	See description
/No_Read_Only	See description
/[No]Record	Record
/[No]Online	/Noonline
/Output=file-spec	See description
/Owner=user-id	See description
/Page_Buffers=number-buffers	/Page_Buffers=2
/Parallel=(Executor_Count=n[,options])	See description
/Progress_Report[=n]	See description
/Prompt={Automatic|Operator|Client}	See description
/Protection[=file-protection]	See description

/[No]Quiet_Point	/Quiet_Point
/Reader_Thread_Ratio=integer	See description
/Restore_Options=file-name	None
/[No]Rewind	/Norewind
/[No]Scan_Optimization	See description
/Tape_Expiration=date-time	The current time
/Threads=n	See description

Description

The RMU Backup command copies information contained in a database to a file. It provides a number of options that allow you to determine the following:

- Whether to perform a parallel backup operation.

  When you specify a parallel backup operation, you must back up to tape or multiple disks. The Parallel Backup Monitor allows you to monitor the progress of a parallel backup operation.

- Whether to back up the database to disk or tape.
- The extent (how much of the database) to back up.

The backup operation uses a multithreaded process to optimize the performance of the backup operation. See the *Oracle Rdb Guide to Database Maintenance* for a complete description of how multithreading works.

A parallel backup operation, in addition to using multithreaded processes, uses a coordinator executor and multiple worker executors (subprocesses) to enhance the speed of the backup operation. You can also direct each worker executor to run on a different node within a cluster to further enhance the speed of the operation. You must have Oracle SQL/Services installed and running to perform a parallel backup operation.

See the *Oracle Rdb Guide to Database Maintenance* for information on when a parallel backup operation is most useful.

Use the Parallel qualifier to indicate to Oracle RMU that you want to perform a parallel backup operation. Use the Noexecute and List_Plan qualifiers to generate a Backup plan file. A Backup plan file records the backup options and specifications you enter on the command line in a text file. You can edit this text file to fine-tune your parallel backup operation and execute it, as needed, with the RMU Backup Plan command. Use the Statistics option to the Parallel qualifier if you want to monitor the progress of the parallel backup operation with the Parallel Backup Monitor. See the description of the Parallel, List_Plan, and Noexecute qualifiers, and the RMU Backup Plan command for details.

You cannot use the Parallel Backup Monitor to monitor the progress of a non-parallel backup operation. However, you can achieve a close approximation of this by specifying the Executor_Count=1 and the Statistics options with the Parallel qualifier. This results in a parallel backup operation with one executor and one controller that you can monitor with the Parallel Backup Monitor.

Both parallel and non-parallel backup operations allow you to perform different types of backup operations with respect to the portions of the database to be backed up, as described in Table 1–4.

1.10 RMU Backup Command

Table 1–4 RMU Backup Options

	Storage Area Selection	
Database Page Selection	**Complete (All Areas)**	**By-Area (Selected Areas)**
Full	Copies the database root (.rdb) file and all the database pages in all the storage areas in the database. This is the default backup operation. Note that you must use this type of backup prior to upgrading to a newer version of Oracle Rdb. Because this is the default operation, no qualifiers are needed to specify a full backup.	Copies the database root (.rdb) file and backs up only the database pages in the storage areas that you specify on the backup command line. All the storage areas in the database are backed up only if you specify them all (or perform a full and complete backup operation). Use the Include or Exclude qualifiers to specify the storage areas for a full by-area backup operation.
Incremental	Copies all database pages that have been updated since the latest full backup operation and the database root file. Use the Incremental (or Incremental=Complete) qualifier to specify an incremental and complete backup operation.	Copies the database root (.rdb) file and only the database pages for the specified storage areas that have changed since the latest full backup operation. Use the Include or Exclude qualifier along with the Incremental=By_Area qualifier to specify an incremental, by-area, backup operation.

Oracle Corporation recommends that you use a full backup operation to back up a database if you have made changes in the physical or logical design. Performing an incremental backup operation under these circumstances can lead to the inability to recover the database properly.

If you choose to perform a by-area backup operation, your database can be fully recovered after a system failure *only* if after-image journaling is enabled on the database. If your database has both read/write and read-only storage areas but does not have after-image journaling enabled, you should do complete backup operations (backup operations on all the storage areas in the database) at all times. Doing complete backup operations when after-image journaling is not enabled ensures that you can recover the entire database to its condition at the time of the previous backup operation.

When a full backup file is created for one or more storage areas, the date and time of the last full backup file created for those storage areas (as recorded in the backup (.rbf) file) is updated. You can display the date and time of the last full backup operation on each of the storage areas in a database by executing an RMU Dump command with the Header qualifier on the latest backup (.rbf) file for the database. The date and time displayed by this command is the date and time of the last *full* backup operation performed for the area.

Note that an *incremental* backup operation on a storage area does *not* update the date and time for the last *full* backup operation performed on the storage area that is recorded in the backup file.

In the event of subsequent damage to the database, you can specify backup files in an RMU Restore command to restore the database to the condition it was in when you backed it up.

The RMU Backup command writes backup files in compressed format to save space. Available or free space in the database root (.rdb) file and on each database page in a storage area (.rda) file is not written to the backup file.

Note

Use only the RMU Backup command to back up all Oracle Rdb databases. Do not back up a database by using any other method (such as the DCL BACKUP command). The database root of a database is updated only when the RMU Backup command is used.

For detailed information on backing up a database to tape, see the *Oracle Rdb Guide to Database Maintenance*.

Command Parameters

root-file-spec
The name of the database root file. The root file name is also the name of the database. The default file extension is .rdb.

backup-file-spec
The file specification for the backup file. The default file extension is .rbf. Depending on whether you are performing a backup operation to magnetic tape, disk, or multiple disks, the backup file specification should be specified as follows:

- If you are backing up to magnetic tape
  - Oracle Corporation recommends that you supply a backup file name that is 17 or fewer characters in length. File names longer than 17 characters might be truncated. See the Usage Notes section for more information about backup file names that are longer than 17 characters.

- If you use multiple tape drives, the backup-file-spec parameter must be provided with (and only with) the first tape drive name. Additional tape drive names must be separated from the first and subsequent tape drive names with commas.

See the *Oracle Rdb Guide to Database Maintenance* for more information about using multiple tape drives.

- If you are backing up to multiple or single disk files
  - It is good practice to write backup files to a device other than the devices where the database root, storage area, and snapshot files of the database are located. This way, if there is a problem with the database disks, you can still restore the database from a backup file.
  - If you use multiple disk files, the backup-file-spec parameter must be provided with (and only with) the first disk device name. Additional disk device names must be separated from the first and subsequent disk device names with commas. You must include the Disk_File qualifier. For example:

```
$ RMU/BACKUP/DISK_FILE MF_PERSONNEL.RDB -
_$ DEVICE1:[DIRECTORY1]MFP.RBF,DEVICE2:[DIRECTORY2]
```

As an alternative to listing the disk device names on the command line (which, if you use several devices, can exceed the line-limit length for a command line), you can specify an options file in place of the backup-file-spec. For example:

```
$ RMU/BACKUP/DISK_FILE LARGE_DB "@DEVICES.OPT"
```

The contents of devices.opt might appear as follows:

```
DEVICE1:[DIRECTORY1]LARGE_DB.RBF
DEVICE2:[DIRECTORY2]
```

The resulting backup files created from such an options file would be:

```
DISK1:[DIRECTORY1]LARGE_DB.RBF
DISK2:[DIRECTORY2]LARGE_DB01.RBF
```

Note that the same directory must exist on each device before you issue the command. Also, if you forget to specify the Disk_File qualifier, you receive an error message similar to the following:

```
$ RMU/BACKUP MF_PERSONNEL DEVICE1:[DIRECTORY1]MFP.RBF, -
_$ DEVICE2:[DIRECTORY2]
%RMU-F-NOTBACFIL, DEVICE1:[DIRECTORY1]MFP.RBF; is not a valid
backup file
%RMU-F-FTL_BCK,Fatal error for BACKUP operation at 2-MAY-2001
09:44:57.04
```

Command Qualifiers

Accept_Label
Specifies that RMU Backup should keep the current tape label it finds on a tape during a backup operation even if that label does not match the default label or that specified with the Label qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) and written to the backup journal file (assuming you have specified the Journal qualifier) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your backup operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes. However, you are responsible for remembering the order in which tapes were written. For this reason, it is a good idea to use the Journal qualifier when you use the Accept_Label qualifier.

If you do not specify this qualifier, the default behavior of RMU Backup is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the Label qualifier).

See the description of the Labels qualifier in this section for information on default labels. See Table 1–5 in the Usage Notes section for a summary of which labels are applied under a variety of circumstances.

Acl
Noacl
Specifies whether to back up the root file access control list (ACL) for a database when you back up the database. The root file ACL controls users privileges to issue Oracle RMU commands.

If you specify the Acl qualifier, the root file ACL will be backed up with the database.

If you specify the Noacl qualifier, the root file ACL will not be backed up with the database. The Noacl qualifier can be useful if you plan to restore the database on a system where the identifiers in the current root file ACL will not be valid.

The default is the Acl qualifier.

Active_IO=max-writes
Specifies the maximum number of write operations to a backup device that the RMU Backup command will attempt simultaneously. This is not the maximum number of write operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

Allocation=blocks
Specifies the size, in blocks, which the backup file is initially allocated. The minimum value for the number-blocks parameter is 1; the maximum value allowed is 2147483647. If you do not specify the Allocation_Quantity qualifier, the Extend_Quantity value effectively controls the file's initial allocation.

This qualifier cannot be used with backup operations to tape.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Block_Size=integer
Specifies the maximum record size for the backup file. The size can vary between 2048 and 65,024 bytes. The default value is device dependent. The appropriate block size is a compromise between tape capacity and error rate. The block size you specify must be larger than the largest page length in the database.

Checksum_Verification
Nochecksum_Verification
The Checksum_Verification qualifier requests that the RMU Backup command verify the checksum stored on each database page before the backup operation is applied, thereby providing end-to-end error detection on the database I/O. The default value is Checksum_Verification.

Oracle Corporation recommends that you accept this default behavior for your applications. The default behavior prevents you from including corrupt database pages in backup files and optimized .aij files. Without the checksum verifications, corrupt data pages in these files are not detected when the files are restored. The corruptions on the restored page may not be detected until weeks or months after the backup file is created, or it is possible the corruption may not be detected at all.

The Checksum_Verification qualifier uses additional CPU resources but provides an extra measure of confidence in the quality of the data that is backed up.

Note that if you specify the Nochecksum qualifier, and undetected corruptions exist in your database, the corruptions are included in your backup file and are restored when you restore the backup file. Such a corruption might be difficult to recover from, especially if it is not detected until long after the restore operation is performed.

Compression
Compression=LZSS
Compression=Huffman
Compression=ZLIB
Nocompression
Allows you to specify the compression method to use before writing data to the backup file. This reduces performance, but may be justified when the backup file is a disk file, or is being backed up over a busy network, or is being backed up to a tape drive that does not do its own compression. You probably do not want to specify the Compression qualifier when you are backing up a database to a tape drive that does its own compression; in some cases doing so can actually result in a larger file.

If you specify the Compression qualifier without a value, the default is COMPRESSION=ZLIB=6.

The level value (ZLIB=level) is an integer between 1 and 9 specifying the relative compression level with one being the least amount of compression and nine being the greatest amount of compression. Higher levels of the compression use increased CPU time while generally providing better compression. The default compression level of 6 is a balance between compression effectiveness and CPU consumption.

Crc[=Autodin_II]
Uses the AUTODIN-II polynomial for the 32-bit cyclic redundancy check (CRC) calculation and provides the most reliable end-to-end error detection. This is the default for NRZ/PE (800/1600 bits/inch) tape drives.

If you enter only Crc as the qualifier, RMU Backup assumes you are specifying Crc=Autodin_II.

Crc=Checksum
Uses one's complement addition, which is the same computation used to do a checksum of the database pages on disk. This is the default for GCR (6250 bits/inch) tape drives and for TA78, TA79, and TA81 tape drives.

The Crc=Checksum qualifier allows detection of data errors.

Nocrc
Disables end-to-end error detection. This is the default for TA90 (IBM 3480 class) drives.

Note

The overall effect of the Crc=Autodin_II, Crc=Checksum, and Nocrc qualifier defaults is to make tape reliability equal to that of a disk. If you retain your tapes longer than 1 year, the Nocrc default might not be adequate. For tapes retained longer than 1 year, use the Crc=Checksum qualifier.

If you retain your tapes longer than 3 years, you should always use the Crc=Autodin_II qualifier.

Tapes retained longer than 5 years could be deteriorating and should be copied to fresh media.

See the *Oracle Rdb Guide to Database Maintenance* for details on using the Crc qualifiers to avoid underrun errors.

Database_Verification
Nodatabase_Verification
The RMU /BACKUP command performs a limited database root file verification at the start of the backup operation. This verification is intended to help prevent backing up a database with various detectable corruptions or inconsistancies of the root file or associated database structures. However, in some limited cases, it can be desirable to avoid these checks.

The qualifier /NODATABASE_VERIFICATION may be specified to avoid the database root file verification at the start of the backup. The default behavior is /DATABASE_VERIFICATION. Oracle strongly recommends accepting the default of /DATABASE_VERIFICATION.

Density=(density-value,[No]Compaction)
Specifies the density at which the output volume is to be written. The default value is the format of the first volume (the first tape you mount). You do not need to specify this qualifier unless your tape drives support data compression or more than one recording density.

The Density qualifier is applicable only to tape drives. Oracle RMU returns an error message if this qualifier is used and the target device is not a tape drive.

The following table lists the tape device density and compaction values supported by Oracle RMU.

DEFAULT	800	833	1600
6250	3480	3490E	TK50
TK70	TK85	TK86	TK87
TK88	TK89	QIC	8200
8500	8900	DLT8000	
SDLT	SDLT320	SDLT600	
DDS1	DDS2	DDS3	DDS4
AIT1	AIT2	AIT3	AIT4
LTO2	LTO3	COMPACTION	NOCOMPACTION

For the OpenVMS tape device density values that accept tape compression, you can use the following syntax:

```
/DENSITY = (new_density_value,[No]Compaction)
```

In order to use the Compaction or NoCompaction keyword, you must use one of the following density values that accepts compression:

DEFAULT	3480	3490E	8200
8500	8900	TK87	TK88
TK89	DLT8000	SDLT	SDLT320
AIT1	AIT2	AIT3	AIT4
DDS1	DDS2	DDS3	DDS4
SDLT600	LTO2	LTO3	

Refer to the OpenVMS documentation for more information about density values.

Disk_File[=(options)]
Specifies that you want to perform a multithreaded backup operation to disk files, floppy disks, or other backup media devices that support VMS directories

and are currently mounted on the VMS system. You can use the following keyword with the Disk_File qualifier:

- Writer_Threads

  Specifies the number of threads that Oracle RMU should use when performing a multithreaded backup operation to disk files. By default, one writer thread is used. If you specify one writer thread though, you lose concurrency because the one writer thread can only write to one disk file at a time. See example below. So /WRITER_THREADS=# specifies the number of writer threads and the number of files you specify at the end of the command specifies the number of RBF files in the RBF file set you are creating. To take advantage of concurrency, you would specify one writer thread per RBF file. One should be aware though that too many writer threads operating at the same time will add thread management overhead.

```
$!
$!
$! Back up MF_PERSONNEL into 3 RBF files, one at a time.
$ define/user_mode sys$output backup1.dat
$ RMU/BACKUP/LOG/JOURNAL=B31A/DISK_FILE=(WRITER=1) MF_PERSONNEL -
 B31A.RBF,TEST$SCRATCH:,TEST$SCRATCH:
$! sear backup1.dat completed
$!
$! Back it up into 3 RBF files, all 3 at once.
$ define/user_mode sys$output backup2.dat
$ RMU/BACKUP/LOG/DISK_FILE=(WRITER=3) MF_PERSONNEL -
 BB33A.RBF,TEST$SCRATCH:,TEST$SCRATCH:
```

Encrypt=({Value=|Name=}[,Algorithm=])

The Encrypt qualifier encrypts the save set file of a database backup.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Encryption Key

If you cannot remember the encryption key you have effectively lost all data in the encrypted file. The encryption key used on the backup command will be REQUIRED on the RESTORE command of that backup file.

Exclude[=storage-area[,...]]
Specifies the storage areas that you want to exclude from the backup file. If you specify neither the Exclude nor the Include qualifier with the RMU Backup command, or if you specify the Exclude qualifier but do not specify a list of storage area names, a full and complete backup operation is performed on the database. This is the default behavior.

If you specify a list of storage area names with the Exclude qualifier, RMU Backup excludes those storage areas from the backup file and includes all of the other storage areas. If you specify more than one database storage area in the Exclude qualifier, place a comma between each storage area name and enclose the list of names within parentheses.

Use the Exclude=* qualifier to indicate that you want only the database root file to be backed up. Note that a backup file created with the Exclude=* qualifier can be restored only with the RMU Restore Only_Root command.

You can use an indirect command file as shown in the following example:

```
$ RMU/BACKUP/EXCLUDE="@EXCLUDE_AREAS.OPT" -
_$ MF_PERSONNEL.RDB PARTIAL_MF_PERS.RBF
%RMU-I-NOTALLARE, Not all areas will be included in this backup file
```

See Section 1.3 for more information on indirect command files.

If you use the Exclude qualifier with a list of storage area names, your backup file will be a by-area backup file because the Exclude qualifier causes database storage areas to be excluded from the backup file. The following example shows the informational message you receive if you do not back up all of the areas in the database:

```
%RMU-I-NOTALLARE, Not all areas will be included in this backup file
```

By using the RMU Backup and RMU Restore commands, you can back up and restore selected storage areas of your database. This Oracle RMU backup and restore by-area feature is designed to:

- Speed recovery when corruption occurs in some (not all) of the storage areas of your database
- Reduce the time needed to perform backup operations because some data (data in read-only storage areas, for example) does not need to be backed up with every backup operation performed on the database

If you plan to use the RMU Backup and RMU Restore commands to back up and restore only selected storage areas for a database, you should perform full and complete backup operations on the database at regular intervals.

1.10 RMU Backup Command

If you plan to back up and restore only selected storage areas of a database, Oracle Corporation also strongly recommends that you enable after-image journaling for the database. This ensures that you can recover all of the storage areas in your database if a system failure occurs.

If you *do not* have after-image journaling enabled and one or more of the areas restored with the RMU Restore command are not consistent with the unrestored storage areas, Oracle Rdb does not allow any transaction to use the storage areas that are not consistent in the restored database. In this situation, you can return to a working database by restoring the database, using the backup file from the last full and complete backup operation of the database storage areas. However, any changes made to the database since the last full and complete backup operation are not recoverable.

If you *do* have after-image journaling enabled, use the RMU Recover command (or the Restore command with the Recover qualifier) to apply transactions from the .aij file to storage areas that are not consistent after the RMU Restore command completes; that is, storage areas that are not in the same state as the rest of the restored database. You cannot use these areas until you recover the database. When the RMU Recover command completes, your database will be consistent and usable.

Using the Exclude or Include qualifier gives you greater flexibility for your backup operations, along with increased file management and recovery complexity. Users of large databases might find the greater flexibility of the backup operation to be worth the cost of increased file management and recovery complexity.

You cannot specify the Exclude=area-list and Include=area-list qualifiers in the same RMU Backup command.

Execute
Noexecute

Use the Execute and Noexecute qualifiers with the Parallel and List_Plan qualifiers to specify whether or not the backup plan file is to be executed.

The following list describes the effects of using the Execute and Noexecute qualifier:

- Execute

  Creates, verifies, and executes a backup list plan

- Noexecute

  Creates and verifies, but does not execute a backup list plan.

The verification determines such things as whether the storage areas listed in the plan file exist in the database.

The Execute and Noexecute qualifiers are only valid when the Parallel and List_Plan qualifiers are also specified.

If you specify the Execute or Noexecute qualifier without the List_Plan and Parallel qualifiers, RMU Backup generates and verifies a temporary backup list plan, but then deletes the backup list plan and returns a fatal error message.

By default, the backup plan file is executed when you issue an RMU Backup command with the Parallel and List_Plan qualifiers.

Extend_Quantity=number-blocks
Sets the size, in blocks, by which the backup file can be extended. The minimum value for the number-blocks parameter is 1; the maximum value is 65535. If you do not specify the Extend_Quantity qualifier, the default number of blocks by which an on-disk backup file can be extended is 2048 blocks.

This qualifier cannot be used with backup operations to tape.

Group_Size=interval
Nogroup_Size
Specifies the frequency at which XOR recovery blocks are written to tape. The group size can vary from 0 to 100. Specifying a group size of zero or specifying the Nogroup_Size qualifier results in no XOR recovery blocks being written. The Group_Size qualifier is only applicable to tape, and its default value is 10. RMU Backup returns an error message if this qualifier is used and the target device is not a tape device.

Include[=storage-area[,...]]
Specifies storage areas that you want to include in the backup file. If you specify neither the Include nor the Exclude qualifier with the RMU Backup command, a full and complete backup operation is performed on the database by default. You can specify the Include=* qualifier to indicate that you want all storage areas included in the backup file, but this is unnecessary because this is the default behavior. The default behavior is performed also when you specify the Include qualifier without specifying a list of storage area names.

If you specify a list of storage area names with the Include qualifier, Oracle RMU includes those storage areas in the backup operation and excludes all of the other storage areas. If you specify more than one database storage area in the Include qualifier, place a comma between each storage area name and enclose the list of names within parentheses.

You cannot specify the Exclude=area-list and Include=area-list qualifiers in the same RMU Backup command.

If you use the Include qualifier, your backup operation will be a by-area backup operation because the areas not specified with the Include qualifier are excluded from the backup file. If you do not back up all of the areas in the database, you receive the following informational message:

```
%RMU-I-NOTALLARE, Not all areas will be included in this backup file
```

By using the RMU Backup and RMU Restore commands, you can back up and restore selected storage areas of your database. This Oracle RMU backup and restore by area feature is designed to:

- Speed recovery when corruption occurs in some (not all) of the storage areas of your database
- Reduce the time needed to perform backup operations because some data (data in read-only storage areas, for example) does not need to be backed up with every backup operation performed on the database

See the description of the Exclude qualifier for information on the implications of using these commands to back up and restore selected areas of your database.

The Include qualifier can be used with indirect file references. See Section 1.3 for more information.

Incremental[=By_Area or Complete]
Noincremental

Determines the extent of the backup operation to be performed. The four possible options are:

- Noincremental

  If you do not specify any of the possible Incremental qualifier options, the default is the Noincremental qualifier. With the Noincremental qualifier, a full backup operation is performed on the database.

- Incremental

  If you specify the Incremental qualifier, an incremental backup of all the storage areas that have changed since the last full and complete backup operation on the database is performed.

- Incremental=By_Area

If you specify the Incremental=By_Area qualifier, an incremental backup operation is performed. The Incremental=By_Area qualifier backs up those database pages that have changed in each selected storage area since the last full backup operation was performed on the area. The last full backup operation performed on the area is the later of the following:

- The last full and complete backup operation performed on the database
- The last full by-area backup operation performed on the area

With an incremental by-area backup operation, each storage area backed up might contain changes for a different time interval, which can make restoring multiple storage areas more complex.

- Incremental=Complete

  If you specify the Incremental=Complete qualifier, an incremental backup operation on all of the storage areas that have changed since the last full and complete backup operation on the database is performed. Selecting the Incremental=Complete qualifier is the same as selecting the Incremental qualifier.

Following a full database backup operation, each subsequent incremental backup operation replaces all previous incremental backup operations.

The following two messages are meant to provide an aid for designing more effective backup strategies. They are printed as part of the per-area summary statistics, and they provide a guide to the incremental benefit of the incremental operation:

- "Est. cost to backup relative to a full backup is x.yy"
- "Est. cost to restore relative to a full restore is x.yy"

These estimates are only approximate and reflect the disk input/output (I/O) cost for the backup or restore operations of that area. Tape I/O, CPU, and all other costs are ignored. The disk I/O costs take into account the number of I/O operations needed and the requirement for a disk head seek to perform the I/O. Each disk type has its own relative costs—transfer rate, latency, seek time—and the cost of a given sequence of I/Os is also affected by competition for the disk by other processes. Consequently, the estimates do not translate directly into "clock time." But they should nevertheless be useful for determining the point at which the incremental operation is becoming less productive.

The relative costs can vary widely, and can be much higher than 1.00. The actual cost depends on the number and location of the pages backed up. An incremental restore operation must always follow a full restore operation, so the actual estimate of restoring the area is actually 1.00 higher than reported when that full restore operation is accounted for. The guideline that

1.10 RMU Backup Command

Oracle Corporation recommends is, "Perform full backup operations when the estimated cost of a restore operation approaches 2.00."

Journal=file-name
Allows you to specify a journal file to be used if the backup is to a single disk or tape drive or to multiple disk (/DISK_FILE) or tape drives. It is useful for recording details about backups, especially if using multiple tape or disk drives because it records which drives were actually used and which areas were on which drives. (This is not to be confused with an after-image journal file.)

As the backup operation progresses, RMU Backup creates the journal file and writes to it a description of the backup operation containing identification of the tape or disk drive names and the volumes and their contents. The default file extension is .jnl.

The journal file must be written to disk; it cannot be written to tape along with the backup file. (Although you can copy the disk file to tape after it is written, if desired.)

This journal file is used with the RMU Restore and the RMU Dump Backup commands to optimize their tape or disk utilization.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file are to be labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

If you do not specify the Label (or Accept_Label) qualifier, RMU Backup labels the first tape used for a backup operation with the first 6 characters of the backup file name. Subsequent default labels are the first 4 characters of the backup file name appended with a sequential number. For example, if your backup file is my_backup.rbf, the default tape labels are my_bac, my_b01, my_b02, and so on.

When you reuse tapes, RMU Backup compares the label currently on the tape to the label or labels you specify with the Label qualifier. If there is a mismatch between the existing label and a label you specify, RMU Backup sends a message to the operator asking if the mismatch is acceptable (unless you also specify the Accept_Labels qualifier).

If desired, you can explicitly specify the list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses. If you are reusing tapes be certain that you load the tapes so that the label RMU Backup expects and the label on each tape will match, or be prepared for a high level of operator intervention. Alternatively, you can specify the Accept_Label qualifier. In this

case, the labels you specify with the Label qualifier are ignored if they do not match the labels currently on the tapes and no operator intervention occurs.

If you specify fewer labels than are needed, RMU Backup generates labels based on the format you have specified. For example, if you specify Label=TAPE01, RMU Backup labels subsequent tapes as TAPE02, TAPE03, and so on up to TAPE99. Thus, many volumes can be preloaded in the cartridge stacker of a tape drive. The order is not important because RMU Backup relabels the volumes. An unattended backup operation is more likely to be successful if all the tapes used do not have to be mounted in a specific order.

Once the backup operation is complete, externally mark the tapes with the appropriate label so that the order can be maintained for the restore operation. Be particularly careful if you are allowing RMU Backup to implicitly label second and subsequent tapes and you are performing an unattended backup operation. Remove the tapes from the drives in the order in which they were written. Apply labels to the volumes following the logic of implicit labeling (for example, TAPE02, TAPE03, and so on).

Oracle recommends you use the Journal qualifier when you employ implicit labeling in a multidrive, unattended backup operation. The journal file records the volume labels that were written to each tape drive. The order in which the labels were written is preserved in the journal. Use the RMU Dump Backup command to display a listing of the volumes written by each tape drive.

You can use an indirect file reference with the Label qualifier. See Section 1.3 for more information on using indirect file references. See Table 1–5 in the Usage Notes section for a summary of which labels are applied under a variety of circumstances.

Librarian[=options]

Use the Librarian qualifier to back up files to data archiving software applications that support the Oracle Media Management interface. The backup file name specified on the command line identifies the stream of data to be stored in the Librarian utility. If you supply a device specification or a version number it will be ignored.

You can use the Librarian qualifier for parallel backup operations. The Librarian utility should be installed and available on all nodes on which the parallel backup operation executes.

The Librarian qualifier accepts the following options:

- Writer_Threads=n

Use the Writer_Threads option to specify the number of backup data streams to write to the Librarian utility. The value of *n* can be from 1 to 99. The default is one writer thread.

Each writer thread for a backup operation manages its own stream of data. Therefore, each thread uses a unique backup file name. The unique names are generated by incrementing the number added to the end of the backup file name. For example, if you specify the following Oracle RMU Backup command:

```
$RMU/ BACKUP /LIBRARIAN=(WRITER_THREADS=3) /LOG DB FILENAM.RBF
```

The following backup file data stream names are generated:

```
FILENAME.RBF
FILENAME.RBF02
FILENAME.RBF03
```

Because each data stream must contain at least one database storage area, and a single storage area must be completely contained in one data stream, if the number of writer threads specified is greater than the number of storage areas, it is set equal to the number of storage areas.

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  – Level 0 traces all error conditions. This is the default.

  – Level 1 traces the entry and exit from each Librarian function.

  – Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian utility.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

  This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed. For a parallel RMU backup, define RMU$LIBRARIAN_PATH as a system-wide logical name so that the multiple processes created by a parallel backup can all translate the logical.

```
$ DEFINE /SYSTEM /EXECUTIVE_MODE -
_$ RMU$LIBRARIAN_PATH librarian_shareable_image.exe
```

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image. For a parallel RMU backup, the RMU$DEBUG_SBT logical should be defined as a system logical so that the multiple processes created by a parallel backup can all translate the logical.

The following lines are from a backup plan file created by the RMU Backup/Parallel/Librarian command:

```
Backup File = MF_PERSONNEL.RBF
Style = Librarian
Librarian_trace_level = #
Librarian_logical_names = (-
         logical_name_1=equivalence_value_1, -
         logical_name_2=equivalence_value_2)
Writer_threads = #
```

The "Style = Librarian" entry specifies that the backup is going to a Librarian utility. The "Librarian_logical_names" entry is a list of logical names and their equivalence values. This is an optional parameter provided so that any logical names used by a particular Librarian utility can be defined as process logical names before the backup or restore operation begins. For example, some Librarian utilities provide support for logical names for specifying catalogs or debugging.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

List_Plan=output-file
Specifies that RMU Backup should generate a backup plan file for a parallel backup operation and write it to the specified output file. A backup plan file is a text file that contains qualifiers that can be specified on the RMU Backup command line. Qualifiers that you do *not* specify on the command line appear as comments in the backup list plan file. In addition, the backup plan file specifies the worker executor names along with the system node, storage areas, and tape drives assigned to each worker executor.

You can use the generated backup plan file as a starting point for building a parallel backup operation to tape that is tuned for your particular configuration. The output file can be customized and then used with the RMU Backup Plan command. See Section 1.12 for details.

If you specify the Execute qualifier with the List_Plan qualifier, the backup plan file is generated, verified, and executed. If you specify the Noexecute qualifier with the List_Plan qualifier, the backup plan file is generated and verified, but not executed.

By default, the backup plan file is executed.

The List_Plan qualifier is only valid when the Parallel qualifier is also specified.

Loader_Synchronization[=Fixed]
Allows you to preload tapes and preserve tape order to minimize the need for operator support. When you specify the Loader_Synchronization qualifier and specify multiple tape drives, the backup operation writes to the first set of tape volumes concurrently then waits until each tape in the set is finished before assigning the next set of tape volumes. This ensures that the tape order can be preserved in the event that a restore operation from these tapes becomes necessary.

One disadvantage with using the Loader_Synchronization qualifier with the Label qualifier is that because not all tape threads back up equal volumes of data, some threads may not need a subsequent tape to back up the assigned volume of data. In order to preserve the tape order, operator intervention may be needed to load the tapes in stages as backup threads become inactive. Use the keyword Fixed to force the assignment of tape labels to the drives regardless of how many tapes each drive actually uses.

The Loader_Synchronization qualifier does result in reduced performance. For maximum performance, no drive should remain idle, and the next identified volume should be placed on the first drive that becomes idle. However, because the order in which the drives become idle depends on many uncontrollable factors and cannot be predetermined, without the Loader_Synchronization qualifier, the drives cannot be preloaded with tapes. (If you do not want to relabel tapes, you might find that the Accept_Label qualifier is a good alternative to using the Loader_Synchronization qualifier. See the description of the Accept_Label qualifier for details.)

Because the cost of using the Loader_Synchronization qualifier is dependent on the hardware configuration and the system load, the cost is unpredictable. A 5% to 20% additional elapsed time for the operation is typical. You must determine whether the benefit of a lower level of operator support compensates for the loss of performance. The Loader_Synchronization qualifier is most useful for large backup operations.

See the *Oracle Rdb Guide to Database Maintenance* for more information on using the Loader_Synchronization qualifier, including information on when this qualifier might lead to unexpected results, and details on how this qualifier interacts with other RMU Backup command qualifiers.

For very large backup operations requiring many tape volumes, managing the physical marking of tape volumes can be difficult. In such a case, you might consider using a library or archiving to automatically manage tape labeling for you.

Lock_Timeout[=seconds]

Determines the maximum time the backup operation will wait for the quiet-point lock and any other locks needed during online backup operations.

The /LOCK_TIMEOUT qualifier can also be specified without a value. In this case, the default lock timeout value specified for the database will be used. Specifically, the default lock timeout value used will be the value of the logical name RDM$BIND_LOCK_TIMEOUT_INTERVAL if it has been specified, or, if RDM$BIND_LOCK_TIMEOUT_INTERVAL has not been specified, the "LOCK TIMEOUT INTERVAL" specified by the SQL CREATE DATABASE or ALTER DATABASE command will be used. If neither value has been specified, the value used will be the maximum possible lock timeout value which can be specified for an Oracle Rdb database.

Note that /LOCK_TIMEOUT is not the default and must be specified. The default if /LOCK_TIMEOUT is not specified is to wait indefinitely to acquire the QUIET POINT or other database locks requested by RMU.

The Lock_Timeout=seconds qualifier is ignored for offline backup operations.

1.10 RMU Backup Command

Log
Log=Brief
Log=Full
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request that the progress of the restore operation be written to SYS$OUTPUT, or the Nolog qualifier to suppress this report. If you specify the Log=Brief option, which is the default if you use the Log option without a qualifier, the log contains the start and completion time of each storage area. If you specify the Log=Full option, the log also contains thread assignment and storage area statistics messages.

If you do not specify the Log or the Nolog qualifier, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Master
Controls the assignment of tape drives to output threads by allowing you to specify a tape drive as a master tape drive. This is a positional qualifier specified with a tape drive. When the Master qualifier is used, it must be used on the first tape drive specified. When the Master qualifier is specified, all additional tape drives become slaves for that tape drive until the end of the command line, or until the next Master qualifier, whichever comes first.

If you specify the Master qualifier (without also specifying the Loader_ Synchronization qualifier) on sets of tape drives, each master/slave set of tape drives will operate independently of other master/slave sets. If the Master qualifier is used on a tape drive that is not physically a master tape drive, the output performance of the backup operation will decrease.

See the *Oracle Rdb Guide to Database Maintenance* for complete details on the behavior of the master qualifier.

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device receiving the backup file has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, RMU Backup should recognize this fact. However, occasionally RMU Backup does not recognize that a tape device has a loader or stacker. Therefore, when the first backup tape fills, RMU Backup issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes RMU

Backup behaves as though a tape device has a loader or stacker when actually it does not.

If you find that RMU Backup is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that RMU Backup expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

No_Read_Only
Allows you to specify that you do not want any of the read-only storage areas in your database to be backed up when you back up the database.

If you do not specify the No_Read_Only qualifier, any read-only storage area not specified with the Exclude qualifier will be included in the backup file. The No_Read_Only qualifier allows you to back up a database with many read-only storage areas without having to type a long list of read-only storage area names with the Exclude qualifier.

If you specify the No_Read_Only qualifier, read-only storage areas are not backed up even if they are explicitly listed by the Include qualifier.

There is no Read_Only qualifier.

Record
Norecord
The Record qualifier is set by default. Using the Norecord qualifier allows you to avoid the modification of the database with recent backup information. Hence the database appears as if it had not been backed up at this time.

The main purpose of this qualifier is to allow a backup of a Hot Standby database without modifying the database files.

The Norecord qualifier can be negated with the Record qualifier.

Online
Noonline
Specifying the Online qualifier permits users running active transactions at the time the command is entered to continue without interruption (unless the Noquiet_Point qualifier is also specified).

Any subsequent transactions that start during the online backup operation are permitted as long as the transactions do not require exclusive access to the database, a table, or any index structure currently being backed up.

To perform an online database backup operation, snapshots (either immediate or deferred) must be enabled. You can use the Online qualifier with the Incremental or Noincremental qualifiers.

If you use the default, the Noonline qualifier, users cannot be attached to the database. If a user has invoked the database and the RMU Backup command is entered with the Noonline qualifier (or without the Online qualifier), an Oracle RMU error results. For example:

```
%RMU-I-FILACCERR, error opening database root file DB_DISK:MF_PERSONNEL.RDB;1
-SYSTEM-W-ACCONFLICT, file access conflict
```

The offline backup process (specified with the Noonline qualifier) has exclusive access to the database and does not require snapshot (.snp) files in order to work. The snapshot files can be disabled when the Noonline qualifier is used.

Oracle Corporation recommends that you close the database (with the RMU Close command) when you perform the offline backup operation on a large database. If the database was opened with the SQL OPEN IS MANUAL statement, the RMU Backup command will fail unless the RMU Close command is used. If the database was opened with the SQL OPEN IS AUTOMATIC statement, the RMU Backup command might fail if the activity level is high (that is, users might access the database before the database is taken off line). Issuing the RMU Close command can force the users out of the database and give the RMU Backup command a chance to start; however, although recommended, issuing the RMU Close command is not required in this case.

Output=file-spec

This qualifier redirects log output from SYS$OUTPUT to the specified file. If a file extension is not specified in the output file specification, the output file extension will be ".LIS".

The default if this qualifier is not specified is to send output from the BACKUP command to SYS$OUTPUT. This qualifier must specify a valid file specification and cannot be negated. The /NOLOG qualifier cannot be specified on the same command line as the /OUTPUT qualifier. The qualifer /LOG=BRIEF or /LOG=FULL can be specified with the /OUTPUT=file_spec qualifier to control the amount of information written to the designated output file. The /LOG=BRIEF (the default) qualifier will display the start and completion time of the backup or restore of each database storage area. The /LOG=FULL qualifier will also display thread assignment information and statistical information for each storage area.

In this example, the /LOG=BRIEF qualifier is used first with the RMU/RESTORE command and then with the RMU/BACKUP command to direct output to the TEST.LIS file.

```
$ rmu/backup/log=brief/output=test mf_personnel mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, BACKUP operation completed at 27-JUN-2018 16:41:00.35
$ delete test.lis;*
$!
$ sql
drop database fi disk:[directory]mf_personnel;
$ rmu/restore/nocdd/dir=disk:[directory]/log=brief/output=test mfp.rbf
$ dir test.lis
Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, RESTORE operation completed at 27-JUN-2018 16:43:00.37
```

In this example, the /LOG=FULL qualifier is used first with the RMU/RESTORE command and then with the RMU/BACKUP command to direct output to the TEST.LIS file.

```
$ rmu/backup/log=full/output=test mf_personnel mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, BACKUP operation completed at 27-JUN-2018 16:45:00.23
$ delete test.lis;*
$!
$ sql
drop database fi disk:[directory]mf_personnel;
$ rmu/restore/nocdd/dir=disk:[directory]/log=full/output=test mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, RESTORE operation completed at 27-JUN-2018 16:48:00.22
```

1.10 RMU Backup Command

Owner_Uic=user-id
Synonymous with the Owner qualifier. See the description of the Owner qualifier.

Owner=user-id
Specifies the owner of the tape volume set. The owner is the user who will be permitted to restore the database. The user-id parameter must be one of the following types of identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A user identification code (UIC) in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP

The Owner qualifier cannot be used with a backup operation to disk. When used with tapes, the Owner qualifier applies to all continuation volumes. The Owner qualifier applies to the first volume only if the Rewind qualifier is also specified.

If the Rewind qualifier is not specified, the backup operation appends the file to a previously labeled tape, so the first volume can have a protection different from the continuation volumes.

See the *Oracle Rdb Guide to Database Maintenance* for information on tape label processing.

Page_Buffers=number-buffers
Specifies the number of disk buffers assigned to each storage area thread.

The range is 2 to 5 with a default of 2.

The higher values speed up scans for changed pages during an incremental backup operation, but they exact a cost in memory usage and larger working set requirements.

Parallel=(Executor_Count=n[,options])
Specifies that you want to perform a parallel backup operation. When you issue an RMU Backup command with the parallel qualifier, RMU Backup generates a plan file. This plan file describes how the parallel backup operation should be executed. If you specify the Noexecute qualifier, the plan file is generated, but not executed. If you specify the Execute qualifier (or accept the default), the plan file is executed immediately after RMU Backup creates it.

The Executor_Count specifies the number of worker executors you want to use for the parallel backup operation. The number of worker executors must be equal to or less than the number of tape drives you intend to use. If you specify Executor_Count=1, the result is a non-parallel backup operation that is executed using the parallel backup procedure, including creation of the plan file and a dbserver process.

You can specify one, both, or none of the following options:

- Node=(node-list)

  The Node=(node-list) option specifies the names of the nodes in the cluster where the worker executors are to run. If more than one node is specified, all nodes must be in the same cluster and the database must be accessible from all nodes in the cluster.

  In addition, for a backup operation across nodes in a cluster to be successful, whoever starts SQL/Services must have proxy access among all nodes involved in the backup operation (assuming you are using DECnet). For example, if you specify the Nodes=(NODE1, NODE2, NODE3) as an option to the Parallel qualifier, whomever started SQL/Services must have access from NODE1 to NODE2, NODE1 to NODE3, NODE2 to NODE1, NODE2 to NODE3, NODE3 to NODE1, and NODE3 to NODE2.

  Separate node names in the node-list with commas. If you do not specify the Nodes option, all worker executors run on the node from which the parallel backup plan file is executed.

- Server_Transport=(DECnet | TCP)

  To execute a parallel backup operation, SQL/Services must be installed on your system. By default, the RMU Backup command uses DECnet to access SQL/Services; if DECnet is not available, RMU Backup tries to use TCP/IP. Use the Server_Transport option to set the default behavior such that RMU Backup tries TCP/IP first. You can also use the SQL_NETWORK_TRANSPORT_TYPE configuration parameter to modify the default behavior. See the *Oracle Rdb Installation and Configuration Guide* for details on setting the SQL_NETWORK_TRANSPORT_TYPE configuration parameter.

- Statistics

  Specifies that you want RMU Backup to gather statistics on the parallel backup operation for use with the Parallel Backup Monitor. You must invoke the Parallel Backup Monitor, a Windowing interface, to view these statistics.

1.10 RMU Backup Command

To execute a parallel backup operation, SQL/Services must be installed on your system. By default, the RMU Backup command uses DECnet to access SQL/Services; if DECnet is not available, RMU Backup tries to use TCP/IP. You can use the SQL_NETWORK_TRANSPORT_TYPE configuration parameter to set the default behavior such that RMU Backup tries TCP/IP first. See the *Oracle Rdb Installation and Configuration Guide* for details on setting the SQL_NETWORK_TRANSPORT_TYPE configuration parameter.

Note that during a parallel backup operation, all tape requests are sent to the Operator; the parallel backup operation does not send tape requests to the user who issues the Backup command. Therefore, you should issue the DCL REPLY/ENABLE=TAPES command from the terminal that serves the operator before issuing the RMU Backup command.

Progress_Report=n
Displays the performance and progress of the backup operation. This feature can be activated by typing Control-T after the RMU Backup operation has been started from the command line or by adding /PROGRESS_REPORT to the RMU command line.

The use of Control-T has to be enabled at the DCL level using:

```
$ SET CONTROL=T
```

While a Control-T just displays the information once, the /PROGRESS_ REPORT qualifier can be used to periodically print the information to SYS$OUTPUT in a batch job.

The /PROGRESS_REPORT qualifier will default to 60 seconds.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Protection[=file-protection]
Specifies the system file protection for the backup file produced by the RMU Backup command.

The default file protection varies, depending on whether you backup the file to disk or tape. This is because tapes do not allow delete or execute access and the SYSTEM account always has both read and write access to tapes.

In addition, a more restrictive class accumulates the access rights of the less restrictive classes.

If you do not specify the Protection qualifier, the default protection is as follows:

- S:RWED,O:RE,G,W if the backup is to disk
- S:RW,O:R,G,W if the backup is to tape

If you specify the Protection qualifier explicitly, the differences in protection applied for backups to tape or disk as noted in the preceding paragraph are applied. Thus, if you specify Protection=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

Quiet_Point
Noquiet_Point
Allows you to specify that the database backup operation is to occur either immediately or when a quiet point for database activity occurs. A quiet point is defined as a point where no active update transactions are in progress in the database. Therefore, this qualifier is used with the Online qualifier.

When you specify the Noquiet_Point qualifier, RMU Backup proceeds with the backup operation as soon as the RMU Backup command is issued, regardless of any update transaction activity in progress in the database. Because RMU Backup must acquire concurrent-read locks on all physical and logical areas, the backup operation will fail if there are any active transactions with exclusive locks on a storage area. However, once RMU Backup has successfully acquired all concurrent-read storage area locks it should not encounter any further lock conflicts. If a transaction that causes Oracle Rdb to request exclusive locks is started while the backup operation is proceeding, that transaction will either wait or receive a lock conflict error, but the RMU Backup command will continue unaffected.

See the Usage Notes section for recommendations on using the Quiet_Point and Noquiet_Point qualifiers.

The default is the Quiet_Point qualifier.

Reader_Thread_Ratio=integer
This qualifier has been deprecated. Use the /Threads qualifier instead.

Restore_Options=file-name
Generates an options file designed to be used with the Options qualifier of the RMU Restore command. If you specify a full backup operation, all the storage areas will be represented in the options file. If you specify a by-area backup

operation, only those areas included in the backup will be represented in the options file.

The Restore_Options file is created at the end of the backup operation.

By default, a Restore_Options file is not created. If you specify the Restore_ Options qualifier and a file, but not a file extension, RMU Backup uses an extension of .opt by default.

Rewind
Norewind
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The tape will be initialized according to the Label and Density qualifiers. The Norewind qualifier is the default and causes the backup file to be created starting at the current logical end-of-tape (EOT).

The Rewind and Norewind qualifiers are applicable only to tape devices. RMU Backup returns an error message if these qualifiers are used and the target device is not a tape device.

Scan_Optimization
Noscan_Optimization
Specifies whether or not RMU Backup should employ scan optimizations during incremental backup operations.

By default, RMU Backup optimizes incremental backup operations by scanning regions of the database that have been updated since the last full backup operation. The identity of these regions is stored in the database. Only these regions need to be scanned for updates during an incremental backup operation. This provides a substantial performance improvement when database activity is sufficiently low.

However, there is a cost in recording this information in the database. In some circumstances the cost might be too high, particularly if you do not intend to use incremental backup operations.

The Scan_Optimization qualifier has different effects, depending on the type of backup operation you perform. In brief, you can enable or disable the scan optimization setting only when you issue a full offline backup command, and you can specify whether to use the data produced by a scan optimization only when you issue an incremental backup command. The following list describes this behavior in more detail:

- During an offline full backup operation, you can enable or disable the scan optimization setting.

Specify the Scan_Optimization qualifier to enable recording of the identities of areas that change after this backup operation completes.

Specify the Noscan_Optimization qualifier to disable recording of the identities of areas that change after this backup operation completes.

By default, the recording state remains unchanged (from the state it was in prior to execution of the Backup command) during a full backup operation.

Note that specifying the Scan_Optimization or Noscan_Optimization qualifier with an offline full backup operation has no effect on the backup operation itself, it merely allows you to change the recording state for scan optimization.

- During an online full backup operation, the qualifier is ignored.

  The recording state for scan optimization remains unchanged (from the state it was in prior to execution of the Backup command). If you execute an online full backup operation and specify the Scan_Optimization or Noscan_Optimization qualifier, RMU Backup returns an informational message to indicate that the qualifier is being ignored.

- During an incremental backup operation, the qualifier directs whether the scan optimization data (if recorded previously) will be used during the operation.

  If you specify the Scan_Optimization qualifier, RMU Backup uses the optimization if Oracle Rdb has been recording the regions updated since the last full backup operation.

  If you specify the Noscan_Optimization qualifier, RMU Backup does not use the optimization, regardless of whether Oracle Rdb has been recording the identity of the regions updated since the last full backup operation.

  You cannot enable or disable the setting for scan optimizations during an incremental backup operation.

  By default, the Scan_Optimization qualifier is used during incremental backup operations.

Tape_Expiration=date-time

Specifies the expiration date of the backup (.rbf) file. Note that when RMU Backup reads a tape, it looks at the expiration date in the file header of the first file on the tape and assumes the date it finds in that file header is the expiration date for the entire tape. Therefore, if you are backing up an .rbf file to tape, specifying the Tape_Expiration qualifier only has meaning if the .rbf is the first file on the tape. You can guarantee that the .rbf file will be the first file on the tape by specifying the Rewind qualifier and overwriting any existing files on the tape.

When the first file on the tape contains an expiration date in the file header, you cannot overwrite the tape before the expiration date unless you have the OpenVMS SYSPRV or BYPASS privilege.

Similarly, when you attempt to restore a .rbf file from tape, you cannot perform the restore operation after the expiration date recorded in the first file on the tape unless you have the OpenVMS SYSPRV or BYPASS privilege

By default, no expiration date is written to the .rbf file header. In this case, if the .rbf file is the first file on the tape, the tape can be overwritten immediately. If the .rbf file is not the first file on the tape, the ability to overwrite the tape is determined by the expiration date in the file header of the first file on the tape.

You cannot explicitly set a tape expiration date for an entire volume. The volume expiration date is always determined by the expiration date of the first file on the tape.

The Tape_Expiration qualifier cannot be used with a backup file written to disk.

See the *Oracle Rdb Guide to Database Maintenance* for information on tape label processing.

Threads=number
Specifies the number of reader threads to be used by the backup process.

RMU creates so called internal 'threads' of execution to read data from one specific storage area. Threads run quasi-parallel within the process executing the RMU image. Each thread generates its own I/O load and consumes resources like virtual address space and process quotas (e.g. FILLM, BYTLM). The more threads, the more I/Os can be generated at one point in time and the more resources are needed to accomplish the same task.

Performance increases with more threads due to parallel activities which keeps disk drives busier. However, at a certain number of threads, performance suffers because the disk I/O subsystem is saturated and I/O queues build up for the disk drives. Also the extra CPU time for additional thread scheduling overhead reduces the overall performance. Typically 2-5 threads per input disk drive are sufficient to drive the disk I/O susbsystem at its optimum. However, some controllers may be able to handle the I/O load of more threads, for example disk controllers with RAID sets and extra cache memory.

In a backup operation, one writer thread is created per output stream. An output stream can be either a tape drive, a disk file or, a media library manager stream. In addition, RMU creates a number of reader threads and their number can be specified. RMU assigns a subset of reader threads to writer threads. RMU calculates the assignment so that roughly the same

amount of data is assigned to each output stream. By default, five reader threads are created for each writer thread. If the user has specified the number of threads, then this number is used to create the reader thread pool. RMU always limits the number of reader threads to the number of storage areas. A threads number of 0 causes RMU to create one thread per storage area which start to run all in parallel immediately. Even though this may sound like a good idea to improve performance, this approach suffers performance for databases with a larger number (>10) of storage areas. For a very large number of storage areas (>800), this fails due to hard limitations in system resources like virtual address space.

For a backup operation, the smallest threads number you can specify is the number of output streams. This guarantees that each writer thread has at least one reader thread assigned to it and does not produce an empty save set. Using a threads number equal to the number of output streams generates the smallest system load in terms of working set usage and disk I/O load. Disk I/O subsystems most likely can handle higher I/O loads. Using a slightly larger value than the number of output streams (for example, assigning more reader threads to a writer thread) typically results in faster execution time.

Usage Notes

- To use the RMU Backup command for a database, you must have the RMU$BACKUP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- If you attempt to back up an area with detected corruptions (or which has corrupt pages logged to the CPT), the backup operation fails immediately. If you attempt to back up an area that contains an undetected corruptions (a corruption that has not been logged to the CPT), the backup operation proceeds until a corruption is found. These undetected corruptions are found only if you specify the Checksum qualifier with the Backup command.
- The following list provides usage information for parallel backup operations:
  - When performing a parallel backup operation, do not allocate or mount any tapes manually; this is done automatically by RMU Backup.
  - You can monitor the progress of a backup operation to tape on your Windows system using the Parallel Backup Monitor.

  - You can use the Parallel Backup Monitor to monitor the progress of a parallel backup operation to tape. Specify your backup operation using the Parallel qualifier with the Executor_Count=1 option to approximate a non-parallel backup operation. Non-parallel backup operations (backup commands without the Parallel qualifier) cannot be monitored with the Parallel Backup Monitor.
  - If a parallel backup operation is issued from a server node, then RMU Backup communicates with SQL/Services to start the Coordinator. SQL/Services creates a Coordinator process.
  - If a parallel backup operation is issued from a client node (for example, using RMUwin), then the same SQL/Services process that is created to execute client/server RMU Backup commands is used as the Coordinator process.
  - You cannot use the Storage Library System (SLS) for OpenVMS with an RMU parallel backup.

- Logical area threshold information for storage areas with uniform page format is recorded in the backup file. See the *Oracle Rdb SQL Reference Manual* for more information on logical area threshold information.
- See the *Oracle Rdb Guide to Database Maintenance* for information on determining the working set requirements for a non-parallel backup operation.
- The following list provides usage information for the Quiet_Point and Noquiet_Point qualifiers
  - If the operation stalls when you attempt a quiet-point Oracle RMU backup operation, it may be because another user is holding the quiet-point lock. In some cases, there is no way to avoid this stall. In other cases you may find the stall is caused by a user who has previously issued and completed a read/write transaction, and is currently running a read-only transaction. When this user started the read/write transaction, the process acquired the quiet-point lock. Ordinarily, such a process retains this lock until it detaches from the database.

    You can set the RDM$BIND_SNAP_QUIET_POINT logical name to control whether or not such a process retains the quiet-point lock. Set the value of the logical name to "1" so that all transactions hold the quiet point lock until a backup process requests it. Read-only transactions will not obtain the quiet point lock; only read/write transactions will obtain the quiet point lock. Set the value of the logical name to "0" so that read-only transactions always release the quiet point lock at the beginning of the transaction, regardless of the

existence of a backup process. All modified buffers in the buffer pool have to be written to disk before the transaction proceeds. Applications that utilize the fast commit feature and often switch between read-only and read/write transactions within a single attach may experience performance degradation if the logical is defined to "0".

Oracle recommends that you do not define the RDM$BIND_SNAP_QUIET_POINT logical for most applications.

- If you intend to use the Noquiet_Point qualifier with a backup procedure that previously specified the Quiet_Point qualifier (or did not specify either the Quiet_Point or Noquiet_Point qualifier), you should examine any applications that execute concurrently with the backup operation. You might need to modify your applications or your backup procedure to handle the lock conflicts that might occur when you specify Noquiet_Point.

  When you specify the Quiet_Point qualifier, the backup operation begins when a quiet point is reached. Other update transactions that are started after the database backup operation begins are prevented from executing until after the root file for the database has been backed up (the backup operation on the database storage areas begins after the root file is backed up).

- When devising your backup strategy for both the database and the after-image journal files, keep in mind the trade-offs between performing quiet-point backup operations and noquiet-point backup operations. A noquiet-point backup operation is quicker than a quiet-point backup operation, but usually results in a longer recovery operation. Because transactions can span .aij files when you perform noquiet-point .aij backup operations, you might have to apply numerous .aij files to recover the database. In a worst-case scenario, this could extend back to your last quiet-point .aij or database backup operation. If you rarely perform quiet-point backup operations, recovery time could be excessive.

  One method you can use to balance these trade-offs is to perform regularly scheduled quiet-point .aij backup operations followed by noquiet-point database backup operations. (You could do the converse, but a quiet-point backup of the .aij file improves the performance of the recovery operation should such an operation become necessary.) Periodically performing a quiet-point .aij backup operation helps to ensure that your recovery time will not be excessive.

- Do not add new logical areas in the context of an exclusive transaction during an online backup operation.

When new logical areas are added during an online backup operation such that new records are physically placed in a location that the backup operation has not processed yet, Oracle Rdb returns the following error:

```
%RMU-F-CANTREADDBS, error reading pages !UL:!UL-!UL
```

Logical areas that cause this problem are created when you do either of the following:

- Create a new table, start a transaction that reserves the new table in exclusive mode, and load the table with rows.
- Create a new table, start a transaction that reserves the new table in exclusive mode, and create an index for the table.

Creating new tables and populating them, or creating new indexes do not pose a problem if the table is not reserved in exclusive mode.

- If you back up a database without its root file ACL (using the Noacl qualifier of the RMU Backup command, for example), a user who wants to restore the database must have the OpenVMS SYSPRV or BYPASS privilege.
- You might receive the RMU-I-WAITOFF informational message when you try to back up your database if the database was manually opened with the RMU Open command and has not been manually closed with the RMU Close command. You also receive this message when you issue an RMU Close command with the Nowait qualifier and users are still attached to the database. To back up your database, you must have exclusive access to the database root file. This error message usually indicates that you do not have exclusive access to the database root file because the operating system still has access to it. If your database was manually opened with the RMU Open command, you should be able to gain exclusive access to the database root file by manually closing the database with an RMU Close command.

  You can also receive this error message when you attempt other operations for which you must have exclusive access to the database root file. The solution in those cases is to attempt the operation again, later. Until you have exclusive access to the database root file, meaning that no other user gained access to the database between the time you issued the command and the time the command takes effect, you cannot complete those operations.
- Backup files are typically smaller in size than the actual database. They exclude free space and redundant structural information that can be reconstructed with a restore operation. However, backup files also contain some overhead to support the backup format. Compression factors range from approximately 1.2 to 3 depending on the organization and fullness

of the database. The compression factor achieved for a given database is generally quite stable and usually only changes with structural or logical reorganization.

Do not use the size of the backup file as an indication of the size of the database files. Use the RMU Analyze command to determine the actual data content.

- Backup performance is strongly affected by the job priority of the process running it. For best performance, a backup operation should execute at interactive priority, even when it is operating as a batch job.
- The following list contains information of interest if you are performing a backup operation to tape:
  - If you back up the database to tape, and you do not specify the Parallel qualifier, you must mount the backup media by using the DCL MOUNT command before you issue the RMU Backup command. The tape must be mounted as a FOREIGN volume. Like the OpenVMS Backup utility (BACKUP), the RMU Backup command performs its own tape label processing. This does not prohibit backing up an Oracle Rdb database to an RMS file on a Files-11 disk.

    When you specify the Parallel qualifier, you need not mount the backup media because the parallel executors allocate and mount the drive and labels for you.
  - When RMU Backup creates a multivolume backup file, you can only append data to the end of the last volume. You cannot append data to the end of the first or any intermediate volumes.
  - The RMU Backup command uses asynchronous I/O. Tape support provided includes support for multifile volumes, multivolume files, and multithreaded concurrent tape processing.
  - If you allow RMU Backup to implicitly label tapes and you are using a tape drive that has a display (for example, a TA91 tape drive), the label displayed is the original label on the tape, not the label generated by RMU Backup.
  - Oracle Corporation recommends that you supply a name for the backup file that is 17 or fewer characters in length. File names longer than 17 characters can be truncated. The system supports four file-header labels: HDR1, HDR2, HDR3, and HDR4. In HDR1 labels, the file identifier field contains the first 17 characters of the file name you supply. The remainder of the file name is written into the HDR4 label, provided that this label is allowed. If no HDR4 label is supported, a file name longer than 17 characters will be truncated.

1.10 RMU Backup Command

The following Oracle RMU commands are valid. The terminating period for the backup file name is not counted as a character, and the default file type of .rbf is assumed. Therefore, the system interprets the file name as wednesdays_backup, which is 17 characters in length:

```
$ RMU/BACKUP/REWIND/LABEL=TAPE MF_PERSONNEL MUA0:WEDNESDAYS_BACKUP.
$ RMU/RESTORE/REWIND/LABEL=TAPE MUA0:WEDNESDAYS_BACKUP.
```

The following Oracle RMU commands create a backup file that cannot be restored. Because no terminating period is supplied, the system supplies a period and a file type of .rbf, and interprets the backup file name as wednesdays_backup.rbf, which is 20 characters in length. RMU truncates the backup file name to wednesdays_backup. When you attempt to restore the backed up file, RMU assumes the default extension of .rbf and returns an error when it cannot find the file wednesdays_backup.rbf on tape.

```
$ RMU/BACKUP/REWIND/LABEL=TAPE MF_PERSONNEL MUA0:WEDNESDAYS_BACKUP
$ RMU/RESTORE/REWIND/LABEL=TAPE MUA0:WEDNESDAYS_BACKUP
```

- See the *Oracle Rdb Guide to Database Maintenance* for information on the steps RMU Backup follows in tape label checking for the RMU Backup command.
- The RMU Backup command works correctly with unlabeled or nonstandard formatted tapes when the Rewind qualifier is specified. However, tapes that have never been used or initialized, and nonstandard tapes sometimes produce errors that make OpenVMS mount attempts fail repeatedly. In this situation, RMU Backup cannot continue until you use the DCL INITIALIZE command to correct the error.
- Table 1–5 summarizes the tape labeling behavior of RMU Backup under a variety of circumstances. For example, the last row of the table describes what labels are applied when you specify both the Label=back qualifier and the Accept_Label qualifier and all the tapes (except the second) are already labeled and used in the following order: aaaa, no label, bbbb, dddd, cccc. The table shows that these tapes will be relabeled in the following order, with no operator notification occurring: aaaa, back02, bbbb, dddd, eeee. Table 1–5 assumes the backup file name is mf_personnel.rbf:

Table 1–5 How Tapes are Relabeled During a Backup Operation

Qualifiers Specified	Current Labels	Resulting Labels	Operator Notification
Neither Label nor Accept_ Label	mf_per	mf_per	None
	mf_p05	mf_p05	
	mf_p06	mf_p06	
	mf_p02	mf_p02	
	mf_p03	mf_p03	
Neither Label nor Accept_ Label	aaaa	mf_per	All tapes except second tape
	no label	mf_p02	
	bbbb	mf_p03	
	dddd	mf_p04	
	cccc	mf_p05	
Label=back	aaaa	back	All tapes except second
	no label	back02	
	bbbb	back03	
	dddd	back04	
	cccc	back05	
Label=(back01, back02)	aaaa	back01	All tapes except second
	no label	back02	
	bbbb	back03	
	dddd	back04	
	cccc	back05	

(continued on next page)

Table 1–5 (Cont.) How Tapes are Relabeled During a Backup Operation

Qualifiers Specified	Current Labels	Resulting Labels	Operator Notification
Accept_Label			None
	aaaa	aaaa	
	no label	mf_p02	
	bbbb	bbbb	
	dddd	dddd	
	cccc	cccc	
Accept_Label, Label=back			None
	aaaa	aaaa	
	no label	back02	
	bbbb	bbbb	
	dddd	dddd	
	cccc	cccc	

- When you use more than one tape drive for a backup operation, ensure that all of the tape drives are the same type (for example, all of the tape drives must be TA90s or TZ87s or TK50s). Using different tape drive types (for example, one TK50 and one TA90) for a single database backup operation may make database restoration difficult or impossible.

  Oracle RMU attempts to prevent you from using different tape drive densities during a backup operation but is not able to detect all invalid cases and expects that all tape drives for a backup are of the same type.

  As long as all of the tapes used during a backup operation can be read by the same type of tape drive during a restore operation, the backup is likely to be valid. This may be the case, for example, when you use a TA90 and a TA90E.

  Oracle Corporation recommends that, on a regular basis, you test your backup and recovery procedures and environment using a test system. You should restore the database and then recover using after-image journals (AIJs) to simulate failure recovery of the production system.

  Consult the *Oracle Rdb Guide to Database Maintenance* and the *Oracle Rdb Guide to Database Design and Definition* for additional information about Oracle Rdb backup and restore operations.

- If you want to use an unsupported density value, use the VMS INITIALIZE and MOUNT commands to set the tape density. Do not use the Density qualifier.
- The density syntax used on the command can also be used in the plan file for the Parallel RMU backup to tape process.
- Oracle Rdb cannot continue a single .rda file across multiple disks. This means that, during a multidisk backup operation, each device must have enough free space to hold the largest storage area in the database. If the storage areas are on stripe sets and are larger than any actual single disk, then the devices specified for the backup file must be striped also.

  It is not possible to indicate which storage area should be backed up to a given device.
- Because data stream names representing the database are generated based on the backup file name specified for the Oracle RMU backup command, you must either use a different backup file name to store the next backup of the database to the Librarian utility or first delete the existing data streams generated from the backup file name before the same backup file name can be reused.

  To delete the existing data streams stored in the Librarian utility, you can use a Librarian management utility or the Oracle RMU Librarian/Remove command.
- If you are backing up to multiple disk devices using thread pools, the following algorithm to assign threads is used by the backup operation:
  - The size of each area is calculated as the product of the page length in bytes multiplied by the highest page number used (maximum page number) for that area.
  - The area sizes are sorted by descending size and ascending device name. For internal processing reasons, the system area is placed as the first area in the first thread.
  - Each of the remaining areas is added to the thread that has the lowest byte count.

  The same algorithm is used for tape devices, but the areas are partitioned among writer threads, not disk devices.
- The partitioning for backup to multiple disk devices is done by disk device, not by output thread, because there will typically be more disk devices than output threads, and an area cannot span a device.

Examples

Example 1

The following command performs a full backup operation on the mf_personnel database and displays a log of the session:

```
$ RMU/BACKUP MF_PERSONNEL -
_$ DISK2[USER1]MF_PERS_FULL_BU.RBF /LOG
```

Example 2

To perform an incremental backup operation, include the Incremental qualifier. Assume a full backup operation was done late Monday night. The following command performs an incremental backup operation on the database updates only for the following day:

```
$ RMU/BACKUP/INCREMENTAL MF_PERSONNEL.RDB -
_$ $222$DUA20:[BCK]TUESDAY_PERS_BKUP/LOG
```

Example 3

To back up the database while there are active users, specify the Online qualifier:

```
$ RMU/BACKUP/ONLINE MF_PERSONNEL.RDB -
_$ $222$DUA20:[BACKUPS]PERS_BU.RBF /LOG
```

Example 4

The following RMU Backup command includes only the EMPIDS_LOW and EMPIDS_MID storage areas in the backup file of the mf_personnel database. All the other storage areas in the mf_personnel database are excluded from the backup file:

```
$ RMU/BACKUP/INCLUDE=(EMPIDS_LOW,EMPIDS_MID) -
_$ MF_PERSONNEL.RDB $222$DUA20:[BACKUPS]MF_PERS_BU.RBF
```

Example 5

The following command backs up the mf_personnel database but not the root file ACL for the database:

```
$ RMU/BACKUP/NOACL MF_PERSONNEL MF_PERS_NOACL
```

Example 6

The following command backs up the mf_personnel database without waiting for a quiet point in the database:

```
$ RMU/BACKUP/NOQUIET_POINT MF_PERSONNEL MF_PERS_NQP
```

Example 7

The following command creates a journal file, pers_journal.jnl, and a backup file, pers_backup.rbf.

```
$ RMU/BACKUP/JOURNAL=PERS_JOURNAL MF_PERSONNEL PERS_BACKUP
```

Example 8

The following example backs up all the storage areas in the mf_personnel database except for the read-only storage areas.

```
$ RMU/BACKUP/NO_READ_ONLY MF_PERSONNEL MF_PERSONNEL_BU
```

Example 9

The following example assumes that you are using multiple tape drives to do a large backup operation. By specifying the Loader_Synchronization qualifier, this command does not require you to load tapes as each becomes full. Instead, you can load tapes on a loader or stacker and RMU Backup will wait until all concurrent tape operations have concluded for one set of tape volumes before assigning the next set of tape volumes.

Using this example, you:

1. Verify the database.
2. Allocate each tape drive.
3. Manually place tapes BACK01 and BACK05 on the 111MUA0: drive.
4. Manually place tapes BACK02 and BACK06 on the 222MUA1: drive.
5. Manually place tapes BACK03 and BACK07 on the 333MUA2: drive.
6. Manually place tapes BACK04 and BACK08 on the 444MUA3: drive.
7. Mount the first volume.
8. Perform the backup operation.
9. Dismount the last tape mounted. (This example assumes it is on the 444MUA3: drive.)
10. Deallocate each tape drive.

```
$ RMU/VERIFY DB_DISK:[DATABASE]MF_PERSONNEL.RDB
$
$ ALLOCATE $111$MUA0:
$ ALLOCATE $222$MUA1:
$ ALLOCATE $333$MUA2:
$ ALLOCATE $444$MUA3:
```

1.10 RMU Backup Command

```
$
$ MOUNT/FOREIGN $111$MUA0:
$
$ RMU/BACKUP /LOG/REWIND/LOADER_SYNCHRONIZATION             -
_$ /LABEL=(BACK01, BACK02, BACK03, BACK04, BACK05,          -
_$ BACK06, BACK07, BACK08)                                  -
_$ DB_DISK:[MFPERS]MF_PERSONNEL.RDB                         -
_$ $111$MUA0:PERS_FULL_MAR30.RBF/Master, $222$MUA1:         -
_$ $333$MUA1:/MASTER, $444$MUA3
$
$ DISMOUNT $444$MUA3:
$
$ DEALLOCATE $111$MUA0:
$ DEALLOCATE $222$MUA1:
$ DEALLOCATE $333$MUA2:
$ DEALLOCATE $444$MUA4:
```

Example 10

The following example generates a parallel backup plan file, but does not execute it. The result is a backup plan file. See the next example for a description of the plan file.

```
$ RMU/BACKUP/PARALLEL=(EXEC=4, NODE=(NODE1, NODE2)) -
_$ /LIST_PLAN=(PARTIAL.PLAN)/NOEXECUTE/INCLUDE=(RDB$SYSTEM, EMPIDS_LOW, -
_$ EMPIDS_MID, EMPIDS_OVER, SALARY_HISTORY, EMP_INFO) -
_$ /LABEL=(001, 002, 003, 004, 005, 006, 007, 008, 009) -
_$ /CHECKSUM_VERIFICATION -
_$ MF_PERSONNEL TAPE1:MF_PARTIAL.RBF, TAPE2:, TAPE3:, TAPE4:
```

Example 11

The following display shows the contents of the plan file, PARTIAL.PLAN created in the preceding example. The following callouts are keyed to this display:

❶ The Plan Parameters include all the parameters specified on the RMU BACKUP command line and all possible command qualifiers.

❷ Command qualifiers that are not specified on the command line are represented as comments in the plan file. This allows you to edit and adjust the plan file for future use.

❸ Command qualifiers that are explicitly specified on the command line are represented in the plan file as specified.

❹ Executor parameters are listed for each executor involved in the backup operation.

```
! Plan created on 28-JUN-1996 by RMU/BACKUP.

Plan Name = PARTIAL
Plan Type = BACKUP

Plan Parameters: ❶
    Database Root File = DISK1:[DB]MF_PERSONNEL;1
    Backup File = PARTIAL.RBF
    ! Journal = specification for journal file ❷
    ! Tape_Expiration = dd-mmm-yyyy
    ! Active_IO = number of buffers for each tape
    ! Protection = file system protection for backup file
    ! Block_Size = bytes per tape block
    ! Density = tape density
    ![No]Group_Size = number of blocks between XOR blocks
    ! Lock_Timeout = number of seconds to wait for locks
    ! Owner = identifier of owner of the backup file
    !Page_Buffers = number of buffers to use for each storage area
    Checksum_Verification ❸
    CRC = AUTODIN_II
    NoIncremental
    ! Accept_labels preserves all tape labels
    Log
    ! Loader_synchronization labels tapes in order across drives
    ! Media_loader forces support of a tape media loader
    NoOnline
    Quiet_Point
    NoRewind
    Statistics
    ACL
    ![No]Scan_Optimization
    Labels = (-
             001          -
             002          -
             003          -
             004          -
             005          -
             006          -
             007          -
             008          -
             009         )
End Plan Parameters
Executor Parameters :
    Executor Name = COORDINATOR
    Executor Type = Coordinator
End Executor Parameters
```

1.10 RMU Backup Command

```
Executor Parameters : ❹
    Executor Name = WORKER_001
    Executor Type = Worker
    Executor Node = NODE1
    Start Storage Area List
        EMPIDS_LOW,
        SALARY_HISTORY
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE1:
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_002
    Executor Type = Worker
    Executor Node = NODE2
    Start Storage Area List
        EMPIDS_MID,
        RDB$SYSTEM
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE2:
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_003
    Executor Type = Worker
    Executor Node = NODE1
    Start Storage Area List
        EMPIDS_OVER
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE3
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_004
    Executor Type = Worker
    Executor Node = NODE2
    Start Storage Area List
        EMP_INFO
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE4
        MASTER
    End Tape Drive List
End Executor Parameters
```

Example 12

The following example demonstrates the use of the Restore_Options qualifier. The first command backs up selected areas of the mf_personnel database and creates an options file. The second command shows the contents of the options file. The last command demonstrates the use of the options file with the RMU Restore command.

```
$ RMU/BACKUP MF_PERSONNEL.RDB MF_EMPIDS.RBF/INCLUDE=(EMPIDS_LOW, -
_$ EMPIDS_MID, EMPIDS_OVER) /RESTORE_OPTIONS=MF_EMPIDS.OPT
%RMU-I-NOTALLARE, Not all areas will be included in this backup file
$ !
$ !
$ TYPE MF_EMPIDS.OPT
!  Options file for database USER1:[MFDB]MF_PERSONNEL.RDB;1
!  Created 18-JUL-1995 10:31:08.82
!  Created by BACKUP command

EMPIDS_LOW -
        /file=USER2:[STOA]EMPIDS_LOW.RDA;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=100, -
                  file=USER2:[SNP]EMPIDS_LOW.SNP;1)

EMPIDS_MID -
        /file=USER3:[STOA]EMPIDS_MID.RDA;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=100, -
                  file=USER3:[SNP]EMPIDS_MID.SNP;1)

EMPIDS_OVER -
        /file=USER4:[STOA]EMPIDS_OVER.RDA;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=100, -
                  file=USER4:[SNP]EMPIDS_OVER.SNP;1)
$ !
$ !
$ !
$ RMU/RESTORE MF_EMPIDS.RBF /AREA/OPTIONS=MF_EMPIDS.OPT
```

1.10 RMU Backup Command

Example 13

The following example uses a density value with compression:

```
$RMU/BACKUP/DENSITY=(TK89,COMPACTION)/REWIND/LABEL=(LABEL1,LABEL2)  -
_$ MF_PERSONNEL TAPE1:MFP.BCK, TAPE2:
```

Example 14

The following example shows how to perform a multidisk backup operation.

```
$ RMU/BACKUP/DISK MF_PERSONNEL DEVICE1:[DIRECTORY1]MFP.RBF, -
_$ DEVICE2:[DIRECTORY2]
.
.
.
%RMU-I-COMPLETED, BACKUP operation completed at  1-MAY-2001 17:40:53.81
```

Example 15

The following example shows the use of the Librarian qualifier with a plan file.

```
$RMU/BACKUP/PARALLEL=EXECUTOR=3/LIBRARIAN=WRITER_THREADS=3 -
_$ /LIST_PLAN=FILENAME.PLAN/NOEXECUTE/LOG DATABASE FILENAM.RBF
$RMU/BACKUP/PLAN FILENAME.PLAN
$RMU/RESTORE/LIBRARIAN=(READER_THREADS=9)/LOG FILENAME.RBF
```

The first backup command creates the plan file for a parallel backup, but does not execute it. The second backup command executes the parallel backup using the plan file. Three worker processes are used; each process uses the three writer threads specified with the Librarian qualifier. Each writer thread in each process writes one stream of backup data to the Librarian utility; a total of nine streams is written.

Example 16

This example shows the use of the Compression qualifier ZLIB.

```
$ RMU /BACKUP /COMPRESS=ZLIB:9 /LOG=FULL FOO BCK
    .
    .
    .
BACKUP summary statistics:
        Data compressed by 53% (9791 KB in/4650 KB out)
```

Example 17

The following example shows the use of the Norecord qualifier. This would be used to backup a Hot Standby database without modifying the database files.

```
$ RMU /BACKUP /NORECORD FOO BCK
```

Example 18

The following example shows the use of the /Progress_Report qualifier to display backup performance every 10 seconds.

```
$ RMU/BACKUP $1$DGA10:[DB]SAMPLE $1$DGA20:[BCK]SAMPLE /DISK /PROGRESS_REPORT=10
Read    18 MB ( 0%) at  18 MB/s, estimated completion time 14:10:41.15
   .
   .
   .
Read  3934 MB (99%) at  28 MB/s, estimated completion time 14:10:39.86
```

Read n MB = raw blocks read so far
(n%) = percent of total blocks read
n MB/s = transfer rate since last display
estimated completion time = recalculated using the current transfer rate

Restores from other than disk files do not display the percentage completed nor the estimated completion time:

```
$ RMU/RESTORE/NOCDD $1$MGA500:SAMPLE,$1$MGA600: /REWIND /PROG=10
Read    72 MB at   14 MB/s
Read   135 MB at   15 MB/s
   .
   .
   .
Read  1441 MB at   12 MB/s
```

Example 19

The following example shows the different RMU commands which accept the /LOCK_TIMEOUT qualifier. In the first command, the specified lock timeout value of 100 seconds will be used. In the other commands, since a lock timeout value is not specified, the default database lock timeout value will be used.

```
$ RMU/BACKUP/ONLINE/LOCK_TIMEOUT=100/NOLOG MF_PERSONNEL MFP.RBF
$ RMU/BACKUP/ONLINE/LOCK_TIMEOUT/NOLOG MF_PERSONNEL MFP.RBF
$ RMU/COPY/ONLINE/LOCK_TIMEOUT/ROOT=DISK:[DIRECTORY]MF_PERSONNEL-
  /NOAFTER/NOLOG MF_PERSONNEL
$ RMU/BACKUP/AFTER/NOLOG/QUIET_POINT/LOCK_TIMEOUT -
  DISK:[DIRECTORY]MF_PERSONNEL MFP_AIJ_1
```

Example 20

The following example shows that the PLAN file used with the RMU PARALLEL BACKUP command will accept either the specified lock timeout value of 100 seconds or will accept the default database lock timeout value.

1.10 RMU Backup Command

```
$ RMU/BACKUP/PARALLEL=EXEC=1/DISK=WRITER=2/LIST=TMP.PLAN/ONLINE-
  /LOCK_TIMEOUT=100 MF_PERSONNEL DISK:[DIRECTORY]MFP,DISK:[DIRECTORY]
$ SEAR TMP.PLAN LOCK_TIMEOUT
    Lock_Timeout = 100
$ RMU/BACKUP/PARALLEL=EXEC=1/DISK=WRITER=2/LIST=TMP.PLAN/ONLINE-
  /LOCK_TIMEOUT MF_PERSONNEL DISK:[DIRECTORY]MFP,DISK:[DIRECTORY]
$ SEAR TMP.PLAN LOCK_TIMEOUT
    Lock_Timeout
$ RMU/BACKUP/PLAN TMP.PLAN
```

1.11 RMU Backup After_Journal Command

Creates a backup file of the database after-image journal (.aij) file or files.

Oracle Rdb supports two types of after-image journaling mechanisms: one that employs a single, extensible .aij file and another that employs multiple, fixed-size .aij files. The type of journaling mechanism being used at the time the backup operation starts can affect how you should specify the backup command. Further information on how these two journaling mechanisms affect the backup operation appears in the Description section.

The backup .aij file is an actual, usable .aij file that can be applied to the appropriate Oracle Rdb database in a recovery operation.

The RMU Backup After_Journal command can be used while users are attached to the database.

Format

RMU/Backup/After_Journal root-file-spec {backup-file-spec | ""}

Command Qualifiers	Defaults
/[No]Accept_Label	/Accept_Label
/Active_IO=max-writes	/Active_IO=3
/[No]Assist	See description
/Block_Size=integer	See description
/[No]Compression[=options]	/Nocompression
/[No]Continuous=(n)	/Nocontinuous
/[No]Crc	See description
/Crc[=Autodin_II]	See description
/Crc=Checksum	See description
/Density=(density-value, [No]Compaction)	See description
/[No]Edit_Filename=(options)	/Noedit_Filename
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/Format={Old_File|New_Tape}	/Format=Old_File
/[No]Group_Size[=interval]	See description
/[No]Interval=number-seconds	/Nointerval
/Label=(label-name-list)	See description
/Librarian[=options]	None
/Lock_Timeout[=seconds]	See description

1.11 RMU Backup After_Journal Command

/[No]Log	Current DCL verify value
/[No]Media_Loader	See description
/Owner=user-id	See description
/Prompt={Automatic|Operator|Client}	See description
/Protection=openvms-file-protection	See description
/[No]Quiet_Point	See description
/[No]Rename	/Norename
/[No]Rewind	/Norewind
/[No]Sequence=(n,m)	/Nosequence
/[No]Space_Check	/Space_Check
/Tape_Expiration=date-time	The current time
/[No]Threshold=disk-blocks	/Nothreshold
/Until=time	See description
/[No]Wait=n	See description

Description

The backup .aij file you create can be used with the RMU Recover command to recover (roll forward) journaled transactions. In some cases, you might have to issue additional Recover commands: one for the backup .aij file and a second for the more recent .aij files.

Oracle Rdb supports the following two types of .aij file configurations:

- A configuration that uses a single, extensible .aij file

  This is the method always used prior to Version 6.0 and is also the default (for compatibility with versions of Oracle Rdb prior to Version 6.0).

  When an extensible .aij file is used, one .aij file is written to and extended, as needed, by the number of blocks specified when the .aij file was created. The .aij file continues to be extended until it is backed up (or the device on which it resides is full).

  The RMU Backup After_Journal command copies transactions recorded in the current .aij file (always on a disk device) to the backup .aij file (which might be on a tape or disk device). On completion, the current .aij file is truncated and used again. During periods of high update activity, the truncation of the active .aij file might not be performed because of conflicting access to the .aij file by other users, but the storage allocated to the active .aij file is still used again when the backup operation completes.

- A configuration that uses two or more fixed-size .aij files

  When fixed-size .aij files are used, the database maintains multiple .aij files; however, only one .aij file is written to at a time. This .aij file is considered the *current* journal. When this .aij file is filled, a switchover occurs to allow journaling to continue in another available .aij file.

1.11 RMU Backup After_Journal Command

The RMU Backup After_Journal command works as follows with fixed-size .aij files:

- Backs up any full .aij files

  The backup operation first backs up the .aij file with the lowest AIJ sequence number (that needs backing up), the operation continues to back up .aij files in ascending AIJ sequence number. If a lot of .aij files need to be backed up when the RMU Backup After_Journal command is issued, one backup file might contain the contents of all the .aij files being backed up.

- Backs up the current .aij file

  Even if there are active transactions at the time of the backup operation, the RMU Backup After_Journal command will start to backup the current active .aij file. If you have specified the Quiet_Point qualifier, the backup operation stalls at some point waiting for all the current transactions to complete.

- Switches to the next available .aij file

  An **available** .aij file is one for which both of the following are true:

  * It is not currently being used to record transactions.
  * It is not needed for a redo operation.

  Such an .aij file might be one that has never been used, or one that has already been backed up.

Once a specified .aij file has been completely backed up, it is initialized and marked as available for reuse.

Note

The method employed, fixed-size .aij files or an extensible .aij file, cannot be set explicitly by the user. Any event that reduces the number of .aij files to one results in an extensible .aij file being used. Any event that increases the number .aij files to two or more results in fixed-size .aij files being used. An inaccessible .aij file is counted in these equations. Therefore, if you have one accessible .aij file and one inaccessible .aij file (perhaps because it has been suppressed), fixed-size .aij journaling is still used.

Because some of the RMU Backup After_Journal qualifiers are valid only when one or the other journaling mechanism is employed, you might need to issue an RMU Dump command to determine which journaling mechanism is currently being employed before you issue an RMU Backup After_Journal command.

1.11 RMU Backup After_Journal Command

Also note that once a backup operation begins, .aij file modification is not allowed until the backup operation is complete. However, if the type of journaling changes between the time you issue an RMU Dump command and the time you issue the RMU Backup After_Journal command, you receive an error message if you have specified qualifiers that are only valid with a particular type of journaling mechanism. (The Threshold qualifier, for example, is valid only when the extensible journaling mechanism is being used.)

If you back up the .aij file or files to tape, you must mount the backup media by using the DCL MOUNT command before you issue the RMU Backup After_Journal command. If you specify the default, Format=Old_File, the RMU Backup After_Journal command uses RMS to write to the tape and the tape must be mounted as an OpenVMS volume. (That is, do not specify the FOREIGN qualifier with the MOUNT command.) If you specify the Format=New_Tape qualifier, the RMU Backup After_Journal command writes backup files in a format similar to that used by the RMU Backup command, and you must mount the tape as a FOREIGN volume.

If you back up an .aij file to disk, you can then use the OpenVMS Backup utility (BACKUP) to archive the .aij backup file.

The RMU Backup After_Journal command can be used in a batch job to avoid occupying an interactive terminal for long periods of time. The Continuous, Interval, Threshold, and Until qualifiers control the duration and frequency of the backup process. When you use the Continuous qualifier, the command can occupy a terminal indefinitely. Therefore, it is good practice to issue the command through a batch process when executing a continuous .aij file backup operation. However, remember that the portion of the command procedure that follows the RMU Backup After_Journal command is not executed until after the time specified by the Until qualifier.

When the RMU Backup After_Journal command completes, it records information about the state of the backup files in the global process symbols presented in the following list. You can use these symbols in DCL command procedures to help automate the backup operation.

These symbols are not set, however, if you have issued a DCL SET SYMBOL/SCOPE=(NOLOCAL, NOGLOBAL) command.

- RDM$AIJ_SEQNO

Contains the sequence number of the last .aij backup file written to tape. This symbol has a value identical to RDM$AIJ_BACKUP_SEQNO. RDM$AIJ_SEQNO was created prior to Oracle Rdb Version 6.0 and is maintained for compatibility with earlier versions of Oracle Rdb.

- RDM$AIJ_CURRENT_SEQNO

  Contains the sequence number of the currently active .aij file. A value of –1 indicates that after-image journaling is disabled.

- RDM$AIJ_NEXT_SEQNO

  Contains the sequence number of the next .aij file that needs to be backed up. This symbol always contains a positive integer value (which can be 0).

- RDM$AIJ_LAST_SEQNO

  Contains the sequence number of the last .aij file ready for a backup operation, which is different from the current sequence number if fixed-size journaling is being used. A value of –1 indicates that no journal has ever been backed up.

  If the value of the RDM$AIJ_NEXT_SEQNO symbol is greater than the value of the RDM$AIJ_LAST_SEQNO symbol, no more .aij files are currently available for the backup operation.

- RDM$AIJ_BACKUP_SEQNO

  Contains the sequence number of the last .aij file backed up by the backup operation. This symbol is set at the completion of an .aij backup operation. A value of –1 indicates that this process has not yet backed up an .aij file.

  The RMU Backup After_Journal command provides an informational message that describes the exact sequence number for each .aij backup file operation.

- RDM$AIJ_COUNT

  Contains the number of available .aij files.

- RDM$AIJ_ENDOFFILE

  Contains the end of file block number for the current AIJ journal.

- RDM$AIJ_FULLNESS

  Contains the percent fullness of the current AIJ journal.

Note that these are string symbols, not integer symbols, even though their equivalence values are numbers. Therefore performing arithmetic operations with them produces unexpected results.

If you need to perform arithmetic operations with these symbols, first convert the string symbol values to numeric symbol values using the OpenVMS F$INTEGER lexical function. For example:

```
$ SEQNO_RANGE = F$INTEGER(RDB$AIJ_LAST_SEQNO)
                - F$INTEGER(RDB$AIJ_NEXT_SEQNO)
```

Command Parameters

root-file-spec
The name of the database root file. The root file name is also the name of the database. An error results if you specify a database that does not have after-image journaling enabled. The default file extension is .rdb.

backup-file-spec
A file specification for the .aij backup file. The default file extension is .aij unless you specify the Format=New_Tape qualifier. In this case, the default file extension is .aij_rbf.

""
Double quotes indicate to Oracle RMU that you want the default .aij backup file specification to be used. The default .aij backup file specification is defined with the SQL ALTER DATABASE statement or the RMU Set After_Journal command.

Command Qualifiers

Accept_Label
Specifies that Oracle RMU should keep the current tape label it finds on a tape during a backup operation even if that label does not match the default label or that specified with the Label qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) and written to the backup journal file (assuming you have specified the Journal qualifier) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your backup operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes.

If you do not specify this qualifier, the default behavior of Oracle RMU is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the Label qualifier).

See the description of the Labels qualifier in this section for information on default labels. See Table 1–5 for a summary of which labels are applied under a variety of circumstances.

Active_IO=max-writes
Specifies the maximum number of write operations to a backup device that the RMU Backup After_Journal command attempts simultaneously. This is not the maximum number of write operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Block_Size=integer
Specifies the maximum record size for the backup file. The size can vary between 2048 and 65,024 bytes. The default value is device dependent. The appropriate block size is a compromise between tape capacity and error rate.

Compression
Compression=LZSS
Compression=Huffman
Compression=ZLIB
Nocompression
Allows you to specify the compression method to use before writing data to the AIJ backup file. This reduces performance, but may be justified when the AIJ backup file is a disk file, or is being backed up over a busy network, or is being backed up to a tape drive that does not do its own compression. You probably do not want to specify the Compression qualifier when you are backing up an aIJ file to a tape drive that does its own compression; in some cases doing so can actually result in a larger file.

Compression methods LZSS and Huffman only work with /Format=New_Tape. Compression method ZLIB works with /Format=New_Tape and /Format=Old_ file.

If you specify the Compression qualifier without a value, the default is COMPRESSION=ZLIB=6.

The level value (ZLIB=level) is an integer between 1 and 9 specifying the relative compression level with one being the least amount of compression and nine being the greatest amount of compression. Higher levels of the compression use increased CPU time while generally providing better compression. The default compression level of 6 is a balance between compression effectiveness and CPU consumption.

If the Compression qualifier is specified, the /SPACE_CHECK qualifier is not turned on by default. It can be specified on the same command line as /COMPRESSION but will not be turned on by default.

Continuous=(n)
Nocontinuous
Specifies whether the .aij backup process operates continuously. You specify termination conditions by specifying one or both of the following:

- The Until qualifier

  Specifies the time and date to stop the continuous backup process.

- The value for *n*

  Specifies the number of iterations Oracle RMU should make through the set of active .aij files before terminating the backup operation.

When you use the Continuous qualifier, you must use either the Until or the Interval qualifier or provide a value for *n* (or both) to specify when the backup process should stop. You can also stop the backup process by using the DCL STOP command when backing up to disk.

If you specify the Continuous qualifier, Oracle Rdb does not terminate the backup process after truncating the current .aij file (when an extensible journal is used) or after switching to a new journal (when fixed-size journals are used). Instead, the backup process waits for the period of time that you specify in the argument to the Interval qualifier. After that time interval, the backup process tests to determine if the threshold has been reached (for an extensible journal) or if the journal is full (for fixed-size journals). It then performs backup operations as needed and then waits again until the next interval break, unless the number of iterations or the condition specified with the Until qualifier has been reached.

If you specify the Continuous qualifier, the backup process occupies the terminal (that is, no system prompt occurs) until the process terminates. Therefore, you should usually enter the command through a batch process.

If you specify the default, the Nocontinuous qualifier, the backup process stops as soon as it completely backs up the .aij file or files. The default value for the number of iterations (*n*) is 1.

If you specify both the Until qualifier and the Continuous=*n* qualifier, the backup operation stops after whichever completes first. If you specify the Until=12:00 qualifier and the Continuous=5 qualifier, the backup operation terminates at 12:00 even if only four iterations have completed. Likewise, if five iterations are completed prior to 12:00, the backup operation terminates after the five iterations are completed.

The Continuous qualifier is not recommended when you are backing up to tape, particularly when the Format=New_Tape qualifier is used. If your tape operations complete successfully, you do not want the backup operation to continue in an infinite loop.

Using the DCL STOP command to terminate a backup operation to tape might result in an incomplete or corrupt backup file. However, do not delete this backup file; it is extremely important that you preserve *all* .aij backup files, even those produced by failed or terminated backup processes. If the resultant .aij backup file is discarded, the next .aij backup file could contain a "gap" in transactions, so that no transactions would ever be rolled forward from that point on.

Crc[=Autodin_II]
Uses the AUTODIN-II polynomial for the 32-bit CRC calculation and provides the most reliable end-to-end error detection. This is the default for NRZ/PE (800/1600 bits/inch) tape drives.

Typing Crc is sufficient to select the Crc=Autodin_II qualifier. It is not necessary to type the entire qualifier.

Crc=Checksum
Uses one's complement addition, which is the same computation used to do a checksum of the database pages on disk. This is the default for GCR (6250 bits/inch) tape drives and for TA78, TA79, and TA81 tape drives.

The Crc=Checksum qualifier allows detection of errors.

Nocrc
Disables end-to-end error detection. This is the default for TA90 (IBM 3480 class) drives.

Note

The overall effect of the Crc=Autodin_II, Crc=Checksum, and Nocrc qualifier defaults is to improve tape reliability so that it is equal to that of a disk. If you retain your tapes longer than 1 year, the Nocrc default might not be adequate. For tapes retained longer than 1 year, use the Crc=Checksum qualifier.

If you retain your tapes longer than 3 years, you should always use the Crc=Autodin_II qualifier.

Tapes retained longer than 5 years could be deteriorating and should be copied to fresh media.

See the *Oracle Rdb Guide to Database Maintenance* for details on using the Crc qualifiers to avoid underrun errors.

Density=(density-value,[No]Compaction)
Specifies the density at which the output volume is to be written. The default value is the format of the first volume (the first tape you mount). You do not need to specify this qualifier unless your tape drives support data compression or more than one recording density.

The Density qualifier is applicable only to tape drives. Oracle RMU returns an error message if this qualifier is used and the target device is not a tape drive.

The following table lists the tape device density and compaction values supported by Oracle RMU.

DEFAULT	800	833	1600
6250	3480	3490E	TK50
TK70	TK85	TK86	TK87
TK88	TK89	QIC	8200
8500	8900	DLT8000	
SDLT	SDLT320	SDLT600	
DDS1	DDS2	DDS3	DDS4
AIT1	AIT2	AIT3	AIT4

LTO2	LTO3	COMPACTION	NOCOMPACTION

For the OpenVMS tape device density values that accept tape compression, you can use the following syntax:

```
/DENSITY = (new_density_value,[No]Compaction)
```

In order to use the Compaction or NoCompaction keyword, you must use one of the following density values that accepts compression:

DEFAULT	3480	3490E	8200
8500	8900	TK87	TK88
TK89	DLT8000	SDLT	SDLT320
AIT1	AIT2	AIT3	AIT4
DDS1	DDS2	DDS3	DDS4
SDLT600	LTO2	LTO3	

Refer to the OpenVMS documentation for more information about density values.

Edit_Filename=(options)
Noedit_Filename

When the Edit_Filename=(options) qualifier is used, the specified backup file name is edited by appending any or all of the values specified by the following options to the backup file name:

- Day_Of_Week

  The current day of the week expressed as a 1-digit integer (1 to 7). Sunday is expressed as 1; Saturday is expressed as 7.

- Day_Of_Year

  The current day of the year expressed as a 3-digit integer (001 to 366).

- Hour

  The current hour of the day expressed as a 2-digit integer (00 to 23).

- Julian_Date

  The number of days passed since 17-Nov-1858.

- Minute

  The current minute of the hour expressed as a 2-digit integer (00 to 59).

- Month

  The current month expressed as a 2-digit integer (01 to 12).

- Sequence

  The journal sequence number of the first journal in the backup operation.

- Vno

  Synonymous with the Sequence option. See the description of the Sequence option.

- Year

  The current year (A.D.) expressed as a 4-digit integer.

If you specify more than one option, place a comma between each option.

The edit is performed in the order specified. For example, the file backup.aij and the qualifier /EDIT_FILENAME=(HOUR, MINUTE, MONTH, DAY_OF_MONTH, SEQUENCE) creates a file with the name backup_160504233.aij when journal 3 is backed up at 4:05 P.M. on April 23rd.

You can make the name more readable by inserting quoted strings between each Edit_Filename option. For example, the following qualifier adds the string "$30_0155-2" to the .aij file name if the day of the month is the 30th, the time is 1:55 and the version number is 2:

```
/EDIT_FILENAME=("$",DAY_OF_MONTH,"_",HOUR,MINUTE,"-",SEQUENCE)
```

This qualifier is useful for creating meaningful file names for your backup files and makes file management easier.

The default is the Noedit_Filename qualifier.

Encrypt=({Value= | Name=}[,Algorithm=])

The Encrypt qualifier encrypts the backup file of the after image journal.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

This feature only works for a newer format backup file which has been created using /FORMAT=NEW_TAPE. Therefore you have to specify /FORMAT=NEW_TAPE with this command if you also use /ENCRYPT.

Encryption Key

If you cannot remember the encryption key you have effectively lost all data in the encrypted file. The encryption key used on the backup

command will be REQUIRED on the RESTORE/RECOVER command of that backup file.

Format=Old_Rms
Synonymous with Format=Old_File qualifier. See the description of that qualifier.

Format=Old_File
Format=New_Tape
Specifies the format in which the backup file is to be written. Oracle Corporation recommends that you specify the Format=Old_File qualifier (or accept the default) when you back up your .aij file to disk and that you specify the Format=New_Tape qualifier when you back up your .aij file to tape.

If you specify the default, the Format=Old_File qualifier, the RMU Backup command writes the file in a format that is optimized for a file structured disk. If you specify the Format=New_Tape qualifier, the Oracle RMU command writes the file in a format that is optimized for tape storage, including ANSI/ISO labeling and end-to-end error detection and correction. When you specify the Format=New_Tape qualifier and back up the .aij file to tape, you must mount the backup media by using the DCL MOUNT command before you issue the RMU Backup After_Journal command. The tape must be mounted as a FOREIGN volume. If you mount the tape as an OpenVMS volume (that is, you do not mount it as a FOREIGN volume) and you specify the Format=New_Tape qualifier, you receive an RMU-F-MOUNTFOR error.

When you back up your .aij file to tape and specify the Format=New_Tape qualifier you can create a backup copy of the database (using the RMU Backup command) and a backup of the .aij file (using the RMU Backup After_Journal command) without dismounting your tape.

The following tape qualifiers have meaning only when used in conjunction with the Format=New_Tape qualifier:

Active_IO
Block_Size
Crc
Density
Group_Size
Label
Owner
Protection
Rewind

Tape_Expiration

The Format=New_Tape and the Noquiet_Point qualifiers cannot be used on the same Oracle RMU command line. See the Usage Notes section for an explanation.

The default file specification, when you specify the Format=New_Tape qualifier is .aij_rbf. The default file specification, when you specify the Format=Old_File qualifier is .aij.

Although Oracle Corporation recommends that you specify the Format=New_Tape qualifier for .aij backup operations to tape and the Format=Old_File qualifier for .aij backup operations to disk, Oracle RMU does not enforce this recommendation. This is to provide compatibility with prior versions of Oracle Rdb. See the Usage Notes section for issues and problems you can encounter when you do not follow this recommendation.

Group_Size[=interval]
Nogroup_Size
Specifies the frequency at which XOR recovery blocks are written to tape. The group size can vary from 0 to 100. Specifying a group size of zero or specifying the Nogroup_Size qualifier results in no XOR recovery blocks being written. The Group_Size qualifier is only applicable to tape, and its default value is device dependent. Oracle RMU returns an error message if this qualifier is used and the target device is not a tape device.

Interval=number-seconds
Nointerval
Specifies the number of seconds for which the backup process waits. Use this qualifier in conjunction with the Continuous qualifier and the extensible journaling method. The interval determines how often to test the active .aij file to determine if it contains more blocks than the value of the Threshold qualifier.

If you specify the Interval qualifier without specifying the number of seconds, or if you omit this qualifier, the default number of seconds is 60.

Oracle Corporation recommends using the default (Interval=60) initially because the interval that you choose can affect the performance of the database. In general, you can arrive at a good interval time on a given database only by judgment and experimentation.

If you specify the Nointerval qualifier, the active .aij file is tested repeatedly with no interval between finishing one cycle and beginning the next.

You must specify the Continuous qualifier if you specify either the Interval or Nointerval qualifier.

If you specify both the Interval and Nocontinuous qualifiers, the Interval qualifier is ignored.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file are to be labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses.

If you do not specify the Label (or Accept_Label) qualifier, Oracle RMU labels the first tape used for a backup operation with the first 6 characters of the backup file name. Subsequent default labels are the first 4 characters of the backup file name appended with a sequential number. For example, if your backup file is my_backup.rbf, the default tape labels are my_b, my_b01, my_b02, and so on.

When you reuse tapes, Oracle RMU compares the label currently on the tape to the label or labels you specify with the Label qualifier. If there is a mismatch between the existing label and a label you specify, Oracle RMU sends a message to the operator asking if the mismatch is acceptable (unless you also specify the Accept_Labels qualifier).

If desired, you can explicitly specify the list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses. If you are reusing tapes be certain that you load the tapes so that the label Oracle RMU expects and the label on each tape will match, or be prepared for a high level of operator intervention.

If you specify fewer labels than are needed, Oracle RMU generates labels based on the format you have specified. For example, if you specify Label=TAPE01, Oracle RMU labels subsequent tapes as TAPE02, TAPE03, and so on up to TAPE99. Thus, many volumes can be preloaded in the cartridge stacker of a tape drive. The order is not important because Oracle RMU relabels the volumes. An unattended backup operation is more likely to be successful if all the tapes used do not have to be mounted in a specific order.

Once the backup operation is complete, externally mark the tapes with the appropriate label so that the order can be maintained for the restore operation. Be particularly careful if you are allowing Oracle RMU to implicitly label second and subsequent tapes and you are performing an unattended backup operation. Remove the tapes from the drives in the order in which they were

written. Apply labels to the volumes following the logic of implicit labeling (for example, TAPE02, TAPE03, and so on).

Oracle Corporation recommends you use the Journal qualifier when you employ implicit labeling in a multidrive, unattended backup operation. The journal file records the volume labels that were written to each tape drive. The order in which the labels were written is preserved in the journal. Use the RMU Dump Backup command to display a listing of the volumes written by each tape drive.

You can use an indirect file reference with the Label qualifier. See Section 1.3 for more information on using indirect file references. See Table 1–5 in the Usage Notes section for a summary of which labels are applied under a variety of circumstances.

Librarian[=options]

Use the Librarian qualifier to back up files to data archiving software applications that support the Oracle Media Management interface. The backup file name specified on the command line identifies the stream of data to be stored in the Librarian utility. If you supply a device specification or a version number it will be ignored.

The Librarian qualifier accepts the following options:

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The

list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian utility.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

  This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed.

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Lock_Timeout[=seconds]

Determines the maximum time the .aij file backup operation will wait for the quiet-point lock and any other locks needed during online backup operations.

The /LOCK_TIMEOUT qualifier can also be specified without a value. In this case, the default lock timeout value specified for the database will be used. Specifically, the default lock timeout value used will be the value of the logical name RDM$BIND_LOCK_TIMEOUT_INTERVAL if it has been specified, or, if RDM$BIND_LOCK_TIMEOUT_INTERVAL has not been specified, the "LOCK TIMEOUT INTERVAL" specified by the SQL CREATE DATABASE or ALTER DATABASE command will be used. If neither value has been specified, the value used will be the maximum possible lock timeout value which can be specified for an Oracle Rdb database.

Note that /LOCK_TIMEOUT is not the default and must be specified. The default if /LOCK_TIMEOUT is not specified is to wait indefinitely to acquire the QUIET POINT or other database locks requested by RMU.

The Lock_Timeout=seconds qualifier is ignored if the Noquiet_Point qualifier is specified.

1.11 RMU Backup After_Journal Command

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device receiving the backup file has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, Oracle RMU should recognize this fact. However, occasionally Oracle RMU does not recognize that a tape device has a loader or stacker. Therefore, when the first backup tape fills, Oracle RMU issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes Oracle RMU behaves as though a tape device has a loader or stacker when actually it does not.

If you find that Oracle RMU is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that Oracle RMU expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

Owner_Uic=user-id
Synonymous with Owner qualifier. See the description of the Owner qualifier.

Owner=user-id
Specifies the owner of the tape volume set. The owner is the user who will be permitted to restore the database. The user-id parameter must be one of the following types of OpenVMS identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A UIC in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP

The Owner qualifier cannot be used with a backup operation to disk. When used with tapes, the Owner qualifier applies to all continuation volumes. Unless the Rewind qualifier is also specified, the Owner qualifier is not applied to the first volume. If the Rewind qualifier is not specified, the backup operation appends the file to a previously labeled tape, so the first volume can have a protection different from the continuation volumes.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Protection=file-protection
Specifies the system file protection for the backup file produced by the RMU Backup After_Journal command.

The default file protection varies, depending on whether you backup the file to disk or tape. This is because tapes do not allow delete or execute access and the SYSTEM account always has both read and write access to tapes. In addition, a more restrictive class accumulates the access rights of the less restrictive classes.

If you do not specify the Protection qualifier, the default protection is as follows:

- S:RWED,O:RE,G,W if the backup is to disk
- S:RW,O:R,G,W if the backup is to tape

If you specify the Protection qualifier explicitly, the differences in protection applied for backups to tape or disk as noted in the preceding paragraph are applied. Thus, if you specify Protection=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

Quiet_Point
Noquiet_Point
Specifies whether the quiet-point lock will be acquired when an .aij backup operation is performed. The default is the Quiet_Point qualifier if the Sequence qualifier is not specified. If the Sequence qualifier is specified, the default is NoQuiet_Point. Use of the Quiet_Point qualifier is meaningful only for a full backup operation; that is, a backup operation that makes a complete pass through all .aij files ready for backup as opposed to one which is done by-sequence (specified with the Sequence qualifier). A full .aij backup operation

can be performed regardless of whether an extensible or a fixed-size .aij journaling mechanism is being employed.

Each .aij backup operation is assigned an .aij sequence number. This labeling distinguishes each .aij backup file from previous .aij backup files. During a recovery operation, it is important to apply the .aij backup files in the proper sequence. The RMU Recover command checks the database root file structure and displays a message telling you the .aij sequence number with which to begin the recovery operation.

The quiet point is a state where all write transactions have either been committed or rolled back and no read/write transactions are in progress. This ensures that the recording of transactions do not extend into a subsequent .aij backup file. This backup file can then be used to produce a recovered database that is in the same state as when the quiet point was reached.

When fixed-size journaling is employed, the Quiet_Point qualifier is only relevant when the active .aij file is being backed up. In this case, a quiet point is acquired only once, regardless of the number of .aij files being backed up.

There is no natural quiet point if someone is writing or waiting to write to the database at any given time. (A natural quiet point is one that is not instigated by the use of the QP (quiet point) Lock.) The .aij backup operation may never be able to capture a state that does not have uncommitted data in the database. As a result, the Noquiet_Point qualifier creates .aij backup files that are not independent of one another. If you apply one .aij backup file to the database without applying the next .aij backup file in sequence, the recovery operation will not be applied completely.

See the Usage Notes section for recommendations on using the Quiet_Point and Noquiet_Point qualifiers.

The following combination of qualifiers on the same command line are invalid:

- Quiet_Point and Sequence
- Quiet_Point and Wait
- Noquiet_Point and Format=New

Rename
Norename

The Rename qualifier creates and initializes a new .aij file and creates the backup file by renaming the original .aij file. The effect is that the original .aij file has a new name and the new .aij file has the same name as the original .aij file.

The Rename qualifier sets the protection on the renamed backup file so that you can work with it as you would any backup file. You can specify the new name by using the Edit_Filename qualifier.

When the Rename qualifier is used, the backup operation is faster (than when Norename, the default, is specified) because the duration of the backup operation is the total time required to rename and initialize the .aij file; the data copy portion of the backup (reading and writing) is eliminated. However, the disk containing the .aij file must have sufficient space for both the new and original .aij files. Note also that the .aij backup file name must not include a device specification.

Note

If there is insufficient space for both the new and original .aij files when the Rename qualifier is specified, after-image journaling shutdown is invoked, resulting in a complete database shutdown.

The Rename qualifier can be used with both fixed-size and extensible journaling files.

The Norename qualifier copies the contents of the .aij file on tape or disk and initializes the original .aij file for reuse. The Norename qualifier results in a slower backup operation (than when Rename is specified), but it does not require space on the journal disk for both new and original .aij files.

The default is Norename.

Rewind
Norewind

Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The tape is initialized according to the Label and Density qualifiers. The Norewind qualifier is the default and causes the backup file to be created starting at the current logical end-of-tape (EOT).

These qualifiers are applicable only to tape devices.

Sequence=(n,m)
Nosequence

Specifies that the journals with sequence numbers from *n* to *m* inclusive are to be backed up. The values *n* and *m* are interpreted or interpolated as follows:

- If Sequence = (33, 35) is specified, then the .aij files with sequence numbers 33, 34, and 35 are backed up.

- If Sequence = (53, 53) is specified, then the .aij file with sequence number 53 is backed up.
- If Sequence = (53) is specified, then the .aij files with sequence numbers 53 and lower are backed up, if they have not been backed up already. For example, if .aij files with sequence numbers 51, 52, and 53 have not been backed up, then Sequence = (53) results in these three .aij files being backed up.
- If Sequence = (55, 53) is specified, then .aij files with sequence numbers 53, 54, and 55 are backed up.
- If the Sequence qualifier is specified without a value list, both *n* and *m* are set to the sequence number of the next journal that needs to be backed up.

The default is the Nosequence qualifier. When the default is accepted, the backup operation starts with the next journal that needs to be backed up and stops when the termination condition you have specified is reached.

The following qualifiers cannot be used or have no effect when used with the Sequence qualifier:

Continuous
Format=New_Tape
Interval
Quiet_Point
Threshold
Until

Furthermore, fixed-size after-image journals must be in use when this qualifier is specified.

Space_Check
NoSpace_Check
Specifies that a disk is checked for space based on the number of blocks which will be needed for the AIJ file to be backed up or renamed, or the number of blocks needed to allocate the temporary work file which will be created. The number of required blocks is compared to a snapshot of the current free blocks on the disk device where a backup file will be created, or the disk device where a new version of the backup file will be created (see the documentation for the RENAME qualifier), or the disk device where a switchover file will be created for backing up the currently active AIJ file. This check ensures that an AIJ Backup will not have to be aborted due to a lack of disk space.

The default if this qualifier is not specified is /SPACE_CHECK.

The disk space check is the default and will only not be done if /NOSPACE_CHECK is specified, if /COMPRESSION is specified, or if the AIJ is not being backed up to a disk device. Note that this disk space check is based on a snapshot of the current disk device free space which can change immediately before or after the snapshot is taken. The %RMU-E-DISKNOSPACE message that is output if insufficient disk free space is detected specifies the free blocks available on the disk device at the time the space check is made.

If there is insufficient disk space to continue the backup, the following message will be output to the operator console:

```
%RMU-I-OPERNOTIFY, system operator notification: error backing up AIJ
AFTER1 - no disk space on device DISK1:
```

The following error messages will be output:

```
%RMU-E-DISKNOSPACE, Insufficient space on device "DISK1:", needed
blocks 512, free blocks 400, total blocks 500

%RMU-F-AIJBCKNOSPACE, After journal "AFTER1" could not be backed up
because of insufficient space on device "DISK1:"
```

In the above example messages, "AFTER1" is the AIJ name in the database root, displayed as "AIJ_NAME =" by the RMU/DUMP/HEADER command, and "DISK1:" is the name of the disk device. "Needed blocks" are the required disk blocks, "free blocks" are the available free disk blocks and "total blocks" is the total of the allocated and free blocks on the disk device.

Tape_Expiration=date-time

Specifies the expiration date of the .aij backup file. Note that when Oracle RMU reads a tape, it looks at the expiration date in the file header of the first file on the tape and assumes the date it finds in that file header is the expiration date for the entire tape. Therefore, if you are backing up an .aij file to tape, specifying the Tape_Expiration qualifier only has meaning if the .aij file is the first file on the tape. You can guarantee that the .aij file will be the first file on the tape by specifying the Rewind qualifier and overwriting any existing files on the tape.

When the first file on the tape contains an expiration date in the file header, you cannot overwrite the tape before the expiration date unless you have the OpenVMS SYSPRV or BYPASS privilege.

Similarly, when you attempt to perform a recover operation with an .aij file on tape, you cannot perform the recover operation after the expiration date recorded in the first file on the tape unless you have the OpenVMS SYSPRV or BYPASS privilege

By default, no expiration date is written to the .aij file header. In this case, if the .aij file is the first file on the tape, the tape can be overwritten immediately. If the .aij file is not the first file on the tape, the ability to overwrite the tape is determined by the expiration date in the file header of the first file on the tape.

You cannot explicitly set a tape expiration date for an entire volume. The volume expiration date is always determined by the expiration date of the first file on the tape. The Tape_Expiration qualifier cannot be used with a backup operation to disk.

See the *Oracle Rdb Guide to Database Maintenance* for information on tape label processing.

Threshold=disk-blocks
Nothreshold
This qualifier can be used only when extensible journaling is enabled. It cannot be used with fixed-size journaling.

The Threshold qualifier sets an approximate limit on the size of the active .aij file. When the size of the active .aij file exceeds the threshold, you cannot initiate new transactions until the backup process finishes backing up and truncating (resetting) the active .aij file. During the backup operation, existing transactions can continue to write to the .aij file. Before new transactions can start, all activity issuing from existing transactions (including activity occurring after the threshold is exceeded) must be moved from the active .aij disk file to the .aij backup file. At that time, the active .aij file will be completely truncated.

If you use the default, the Nothreshold qualifier, each backup cycle will completely back up the active .aij file. Oracle Corporation recommends using the Nothreshold qualifier.

An appropriate value for the Threshold qualifier depends on the activity of your database, how much disk space you want to use, whether backup operations will be continuous, and how long you are willing to wait for a backup operation to complete.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on setting SPAM thresholds.

Until=time
Specifies the approximate future time and date to stop the continuous backup process. There is no default.

Wait=n
Nowait
Specifies whether the backup operation should wait (the Wait qualifier) or terminate (the Nowait qualifier) when it encounters a journal that is not ready to be backed up. The value specified for the Wait qualifier is the time interval in seconds between attempts to back up the journal that was not ready.

The Wait or Nowait qualifier can only be specified if the Sequence qualifier is also specified. When the Wait qualifier is specified, the default value for the time interval is 60 seconds.

The default is the Nowait qualifier.

Usage Notes

- To use the RMU Backup After_Journal command for a database, you must have the RMU$BACKUP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- See the *Oracle Rdb7 Guide to Database Performance and Tuning* for information on how to enhance the performance of the RMU Backup After_ Journal command.

Note

When fast commit is enabled and an extensible .aij file configuration is used, the after-image journal backup process compresses and retains some fraction of the original .aij file (in a new version of the current .aij file). This fraction can approach 100% of the original size. Therefore, be sure to reserve enough space to duplicate the maximum size .aij file before backing it up.

Oracle Corporation recommends that you schedule .aij backup operations with sufficient frequency and check the free space and journal file size periodically; you need to know when you are approaching a critical situation in terms of free space. (This is good practice whether or not you have fast commit enabled.)

However, if you issue the RMU Backup After_Journal command with fast commit enabled and find that you have insufficient space for the .aij file, you have the following options:

- Delete unneeded files to create sufficient space on the disk where the .aij file is located.
- Temporarily disable fast commit and back up the .aij file.

- Close the database, disable after-image journaling, enable a new after-image journal file, and perform a backup operation. (The database can be opened either before or after the backup operation.)
- Close the database. Create a bound volume set or stripe set that is large enough for the .aij file and copy the .aij file there. Use the RMU Set After_Journal command to change the .aij file name (or redefine the logical name if one was used to locate the journal), and then open the database again.

- Note the following issues and problems you can encounter when you specify the Format=Old_File qualifier for an .aij backup operation to tape or the Format=New_Tape qualifier for an .aij backup operation to disk:
  - If you use the Format=Old_File qualifier for an .aij backup operation to tape and the tape is mounted as a FOREIGN volume, the result is an unlabeled tape that can be difficult to use for recovery operations.

    Therefore, if you use the Format=Old_File qualifier with an .aij backup operation to tape, you must mount the tape as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier with the DCL MOUNT command).
  - You must remember (or record) the format you use when you back up your .aij file and specify that same format when you issue an RMU Dump After_Journal, RMU Optimize After_Journal, or RMU Recover command for the .aij backup file.

    If you always follow the guidelines of specifying Format=New_Tape for tape backups and Format=Old_File for disk backups, you do not need to track the format you specified for the .aij backup operation for future use with the other Oracle RMU .aij commands.
  - If you specify Format=Old_File for a backup operation to tape and the .aij spans tape volumes, you might have problems recovering the .aij file.
- You can use the RMU Backup After_Journal command to save disk space by spooling the .aij file to tape.
- When you use extensible .aij files, note that although a new version of the .aij file might be created when the after-image backup operation begins, the old .aij file continues to be active and growing. Until the switch occurs (which could be several hours after the creation of the new version of the .aij file), the old .aij file is still being accessed. For this and other reasons,

you should never use the DCL DELETE or DCL PURGE on .aij files (or any database files).

- The following list provides usage information for the Quiet_Point and Noquiet_Point qualifiers:
  - If the backup operation stalls when you attempt a quiet-point Oracle RMU backup operation, it may be because another user is holding the quiet-point lock. In some cases, there is no way to avoid this stall. However, you may find the stall is caused by a user who has previously issued and completed a read-write transaction, and is currently running a read-only transaction. When this user started the read-write transaction his or her process acquired the quiet-point lock. Ordinarily, such a process retains this lock until it detaches from the database.

    You can set the RDM$BIND_SNAP_QUIET_POINT logical name to control whether or not such a process retains the quiet-point lock. Set the value of the logical name to "1" to allow such a process to hold the quiet-point lock until they detach from the database. Set the value of the logical name to "0", to ensure that the process releases the quiet-point lock prior to starting a read-only transaction.
  - When devising your backup strategy for both the database and the after-image journal files, keep in mind the trade-offs between performing quiet-point backup operations and noquiet-point backup operations. A noquiet-point backup operation is quicker than a quiet-point backup operation, but usually results in a longer recovery operation. Because transactions can span .aij files when you perform noquiet-point .aij backup operations, you might have to apply numerous .aij files to recover the database. In a worst-case scenario, this could extend back to your last quiet-point .aij or database backup operation. If you rarely perform quiet-point backup operations, recovery time could be excessive.

    One method you can use to balance these trade-offs is to perform regularly scheduled quiet-point .aij backup operations followed by noquiet-point database backup operations. (You could do the converse, but a quiet-point backup of the .aij file improves the performance of the recovery operation should such an operation become necessary.) Periodically performing a quiet-point .aij backup operation helps to ensure that your recovery time will not be excessive.

  - You cannot specify the Noquiet_Point qualifier with the Format=New_Tape qualifier because an .aij file created with the Noquiet_Point qualifier does not end on a quiet point. Some transactions can bridge several backup files. When you recover from these backup files you frequently must apply several backup files in the same RMU Recover command. However, the RMU Recover command with the Format=New_Tape qualifier can only process one backup file at a time, so it cannot support backup files created with the Noquiet_Point qualifier.

- Oracle RMU tape operations do not automatically allocate the tape drives used. In an environment where many users compete for a few tape drives, it is possible for another user to seize a drive while Oracle RMU is waiting for you to load the next tape volume.

  To prevent this, issue a DCL ALLOCATE command for the drives you will be using before you issue the Oracle RMU command, and then issue a DCL DEALLOCATE command after you complete the Oracle RMU command.

- The Label qualifier can be used with indirect file reference. See Section 1.3 for more information.

- If an .aij backup process fails or is terminated prematurely, the user might discard the resultant .aij backup file because the backup operation was not completed. However, all .aij backup files, including those produced by a failed backup process, are necessary to recover a database. If an .aij backup file of a failed backup process is discarded, the database is not recoverable from that point forward. This is especially important if you use magnetic tapes as the .aij backup media; in this case, preserve this magnetic tape and do *not* reuse it.

- When an .aij backup process, especially one running in continuous (Continuous) mode, writes to the .aij backup file, it is possible for the transferred data to be deleted from the database .aij file. If the backup process subsequently fails or is prematurely terminated (for example with Ctrl/Y or the DCL STOP command), it might not be possible to retransfer the data to the subsequent .aij backup file because the data was deleted from the active database .aij file.

  Therefore, it is extremely important that you preserve *all* .aij backup files, even those produced by failed or terminated backup processes. If the resultant .aij backup file is discarded, the next .aij backup file could contain a "gap" in transactions, so that no transactions would ever be rolled forward from that point on.

This problem is more severe when backing up directly to tape. Therefore, when backing up to tape, you should back up one journal at a time, rather than using an open-ended or long-duration backup operation.

Note

If this problem occurs, the database is *not* inconsistent or corrupt. Rather, the database cannot be rolled forward past the discarded .aij backup file.

The solution to this problem is to preserve all .aij backup files to ensure that a database can be completely recovered.

If you have discarded an .aij backup file, perform a full and complete database backup operation immediately to ensure that the database can be restored up to the current transaction.

- When an AIJ backup operation completes, the after-image journal files are initialized with a pattern of -1 (hex FF) bytes. This initialization is designed to be as fast as possible. It fully utilizes the I/O subsystem by performing many large asynchronous I/O operations at once. However, this speed can come at the cost of a high load on I/O components during the initialization. This load could slow down other I/O operations on the system.

  You can use two logical names to control the relative I/O load that the AIJ initialization operation places on the system. If you define these logical names in the system logical name table, they are translated each time an AIJ file is initialized.

  The RDM$BIND_AIJ_INITIALIZE_IO_COUNT logical name specifies the number of asynchronous I/O operations that are queued at once to the AIJ file. If the logical name is not defined, the default value is 2, the minimum value is 1, and the maximum value is 128.

  The RDM$BIND_AIJ_INITIALIZE_IO_SIZE logical name controls the number of 512-byte disk blocks to be written per I/O operation. If the logical name is not defined, the default value is 256, the minimum value is 4, and the maximum value is 256.

  Reducing the value of either logical will probably increase the amount of time needed to initialize the AIJ file after a backup. However, it may also reduce load on the I/O subsystem.

- If you want to use an unsupported density value, use the VMS INITIALIZE and MOUNT commands to set the tape density. Do not use the Density qualifier.

- When you use the RMU Backup After_Journal command with the Log qualifier, the DCL global symbol RDM$AIJ_LAST_OUTPUT_FILE is automatically created. The value of the symbol is the full output backup AIJ file specification.
- Because data stream names representing the database are generated based on the backup file name specified for the Oracle RMU backup command, you must either use a different backup file name to store the next backup of the database to the Librarian utility or first delete the existing data streams generated from the backup file name before the same backup file name can be reused.

  To delete the existing data streams stored in the Librarian utility, you can use a Librarian management utility or the Oracle RMU Librarian/Remove command.
- The system logical RDM$BIND_AIJBCK_CHECKPOINT_TIMEOUT can be configured to control the checkpoint stall duration independent of the AIJ shutdown parameter. This logical works for both the AIJ backup and Automatic Backup Server (ABS) utilities.

Examples

Example 1

Assuming that you have enabled after-image journaling for the MF_ PERSONNEL database, the following command causes extensible .aij entries to be backed up continuously until the time specified.

```
$ RMU/BACKUP/AFTER_JOURNAL/CONTINUOUS/THRESHOLD=500 -
_$          /INTERVAL=300/UNTIL="01-JUL-1996 16:15:00.00" -
_$          MF_PERSONNEL.RDB  DISK12:[PERS_AIJ]BU_PERSONNEL.AIJ
```

Every 300 seconds, the backup process tests to determine if the active .aij file on disk has reached the threshold size of 500 blocks. If not, transaction processing continues normally for one or more 300-second intervals until the threshold test indicates that the active .aij file has reached a size of at least 500 blocks. When the .aij file reaches that file size, Oracle RMU allows existing transactions to continue to write to the active .aij file but does not allow new transactions to start.

Assuming that the active .aij file contains 550 blocks, Oracle Rdb moves those 550 blocks to the backup journal file and deletes them from the active journal file. Then, the backup process determines if the transactions already in progress have written more journal records to the active journal file during the backup operation. If so, Oracle RMU moves those journal records to the backup file.

After Oracle Rdb completely moves the active journal file, it truncates the journal file to 0 blocks. Oracle Rdb then allows new transactions to start and the backup process resumes threshold testing at 300-second intervals. The backup process continues until the time and date specified by the Until qualifier.

Example 2

The following examples show backing up .aij files in sequence. Note that a number of transactions were committed to the database between backup operations.

```
$ RMU/BACKUP/AFTER_JOURNAL/LOG MF_PERSONNEL MFPERS_BKUP_AIJ1.AIJ
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification:
 Oracle Rdb V7.2 Database DISK1:[DB]MF_PERSONNEL.RDB;1
 Event Notification AIJ backup operation started

%RMU-I-AIJBCKSEQ, backing up after-image journal
 sequence number 0
%RMU-I-LOGBCKAIJ, backing up after-image journal
 AIJ1 at 16:35:53.41
%RMU-I-LOGCREBCK, created backup file
 DISK1:[DB]MFPERS_BKUP_AIJ1.AIJ;1
%RMU-I-AIJBCKSEQ, backing up after-image journal
 sequence number 1
%RMU-I-LOGBCKAIJ, backing up after-image journal
 AIJ2 at 16:35:54.58
%RMU-I-QUIETPT, waiting for database quiet point
%RMU-I-OPERNOTIFY, system operator notification:
 Oracle Rdb V7.2 Database DISK1:[DB]MF_PERSONNEL.RDB;1
 Event Notification AIJ backup operation completed

%RMU-I-AIJBCKEND, after-image journal backup operation
 completed successfully
%RMU-I-LOGAIJJRN, backed up 2 after-image journals
 at 16:35:56.40
%RMU-I-LOGAIJBLK, backed up 508 after-image journal blocks
 at 16:35:56.41
   .
   .
   .
$ More transactions committed to the database
   .
   .
   .
$ RMU/BACKUP/AFTER_JOURNAL/LOG MF_PERSONNEL MFPERS_BKUP_AIJ2.AIJ
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification:
 Oracle Rdb V7.2 Database
 DISK1:[DB]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation started
```

1.11 RMU Backup After_Journal Command

```
%RMU-I-AIJBCKSEQ, backing up after-image journal sequence number 2
%RMU-I-LOGBCKAIJ, backing up after-image journal AIJ1 at 16:47:44.66
%RMU-I-LOGCREBCK, created backup file
 DISK2:[AIJ]MFPERS_BKUP_AIJ2.AIJ;1
%RMU-I-OPERNOTIFY, system operator notification:
 Oracle Rdb V7.2 Database
 DISK1:[DB]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation completed

%RMU-I-AIJBCKEND, after-image journal backup operation completed
 successfully
%RMU-I-LOGAIJJRN, backed up 1 after-image journal at 16:47:46.57
%RMU-I-LOGAIJBLK, backed up 254 after-image journal blocks at
 16:47:46.57
```

Example 3

The following example uses the Edit_Filename qualifier to give the .aij backup file a meaningful file name. The Rename qualifier specifies that Oracle RMU should create the backup file by renaming the current .aij file and by creating a new .aij file with the same name as the original .aij file.

```
$ RMU/BACKUP/AFTER_JOURNAL MF_PERSONNEL -
_$ /EDIT_FILENAME=(SEQUENCE,"_",HOUR,"_",MINUTE,"_",MONTH,"_", -
_$ DAY_OF_MONTH) AIJ2/RENAME

$ DIR DISK1:[DB.AIJ2]*.AIJ

Directory DISK1:[DB.AIJ_TWO]

AIJ23_15_46_07_09.AIJ;1
```

Example 4

The following example shows the syntax to use when you want the .aij backup file name to default to that previously specified with the RMU Set After_ Journal command. Note that the .aij backup file name used is that which corresponds to the first .aij file included in the backup operation.

```
$ RMU/SET AFTER_JOURNAL MF_PERSONNEL /ENABLE/RESERVE=5 -
_$ /ADD=(NAME=AIJ1, FILE=DISK1:[AIJ]AIJ_ONE, -
_$ BACKUP_FILE=DISK4:[AIJBCK]AIJ1BCK) -
_$ /ADD=(NAME=AIJ2, FILE=DISK2:[AIJ]AIJ_TWO, -
_$ BACKUP_FILE=DISK4:[AIJBCK]AIJ2BCK) -
_$ /ADD=(NAME=AIJ3, FILE=DISK3:[AIJ]AIJ_THREE, -
_$ BACKUP_FILE=DISK4:[AIJBCK]AIJ3BCK)
%RMU-W-DOFULLBCK, full database backup should be done to
 ensure future recovery
$ !
$ !Assume backup operation was performed and other database
   activity occurs.
$ !Then back up the .aij files:
$ !
$ RMU/BACKUP/AFTER_JOURNAL MF_PERSONNEL.RDB ""
$ !
$ DIR DISK4:[AIJBCK]

Directory DISK4:[AIJBCK]

AIJ1BCK.AIJ;1
```

Example 5

The following example uses a density value with compression:

```
RMU/BACKUP/AFTER_JOURNAL /DENSITY=(TK89,COMPACTION)/REWIND -
/LABEL=(LABEL1,LABEL2) MF_PERSONNEL TAPE1:MFP.AIJ, TAPE2:
```

Example 6

1.11 RMU Backup After_Journal Command

The following example shows the /SPACE_CHECK qualifier specified with the RMU/BACKUP/AFTER command. When backing up after-image journal "AFTER1" to disk device "DISK1:", 512 blocks will be needed but only 400 blocks are free and "DISK1:" only has a maximum capacity of 500 blocks. Therefore the backup is aborted. Note that /SPACE_CHECK is the default and /NOSPACE_CHECK must be specified to bypass the space check on disk devices.

```
$ rmu/backup/after/SPACE_CHECK/continuous/until=13:53:24.33 -
  /log mf_personnel DISK:[DIRECTORY]backup_aij.out
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification: AIJ backup operation started
%RMU-I-AIJBCKSEQ, backing up after-image journal sequence number 0
%RMU-I-LOGBCKAIJ, backing up after-image journal AFTER1 at 13:49:24.40
%RMU-E-DISKNOSPACE, Insufficient space on device "DISK1:", needed blocks 512,
 free blocks 400, total blocks 500
%RMU-I-OPERNOTIFY, system operator notification: error backing up AIJ AFTER1 -
 no disk space on device DISK1:
%RMU-I-AIJBCKSTOP, backup of after-image journal AFTER1 did not complete
%RMU-I-OPERNOTIFY, system operator notification: AIJ manual backup operation
failed
%RMU-F-AIJBCKNOSPACE, After journal "AFTER1" could not be backed up because
 of insufficient space on device "DISK1:"
%RMU-F-FTL_BCK, Fatal error for BACKUP operation at  2-JUN-2016 13:49:24.43
```

1.12 RMU Backup Plan Command

Executes a backup plan file previously created with the RMU Backup command (or created manually by the user).

Format

RMU/Backup/Plan plan-file

Command Qualifiers	Defaults
/[No]Database_Verification	Database_Verification
/[No]Execute	Execute
/List_Plan=output-file	None
/Progress_Report[=n]	See description

Description

A backup plan file is created when you execute an RMU Backup command with the Parallel and List_Plan qualifiers. See Section 1.10 for details on creating a plan file and the format of a plan file.

Command Parameters

plan-file-spec
The file specification for the backup plan file. The default file extension is .plan.

Command Qualifiers

Database_Verification
NoDatabase_Verification
This qualifier is used to improve performance of the RMU/BACKUP.

By default, RMU/BACKUP verifies the database root at the start of the backup before backing up an Oracle Rdb database. If the database root is invalid, the error diagnostics from the verification are output and the backup is terminated. This is to prevent backing up a corrupt database. To not verify the database root, the user must specify /NODATABASE_VERIFICATION. As part of this verification, all live and snapshot database storage areas are opened to verify the area prologue block and the area maximum page number and then closed. Later, the storage areas are again opened and closed a second time to do the actual backup of the data in each storage area.

1.12 RMU Backup Plan Command

To improve the performance of RMU/BACKUP, the extra open and close of the live and snapshot database storage areas just to verify the database root can be eliminated by specifying /NODATABASE_VERIFICATION. The same verification is still done but the storage areas will only be opened and closed once during the backup. Note that /DATABASE_VERIFICATION remains the default for RMU/BACKUP.

The qualifier "/[NO]DATABASE_VERIFICATION" is used to set the "[No]Database_Verification" option in the initial *.PLAN file created from the RMU/BACKUP command line. The *.PLAN file can then be edited to change this option if desired. The default is "/DATABASE_VERIFICATION".

Execute
Noexecute
The Execute qualifier specifies that Oracle RMU is to execute the plan file. The Noexecute qualifier specifies that Oracle RMU should not execute the plan file, but instead perform a validity check on the contents of the plan file.

The validity check determines such things as whether the storage areas names assigned to each worker executor exist.

By default, Oracle RMU executes the backup plan file when the RMU Backup Plan command is issued.

List_Plan=output-file
Specifies that Oracle RMU should generate a new plan file and write it to the specified output file. This new plan file is identical to the plan file you specified on the command line (the "original" plan file) with the following exceptions:

- Any user-added comments in the original plan file do not appear in the new plan file.
- The new plan file is formatted to match the standard format for RMU Backup plan files.

Progress_Report=n
Displays the performance and progress of the backup operation. This feature can be activated by typing Control-T after the RMU Backup operation has been started from the command line or by adding /PROGRESS_REPORT to the RMU command line.

The use of Control-T has to be enabled at the DCL level using:

```
$ SET CONTROL=T
```

While a Control-T just displays the information once, the /PROGRESS_ REPORT qualifier can be used to periodically print the information to SYS$OUTPUT in a batch job.

The /PROGRESS_REPORT qualifier will default to 60 seconds.

Usage Notes

- To use the RMU Backup Plan command for a database, you must have the RMU$BACKUP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- To execute the RMU Backup Plan command, Oracle SQL/Services must be installed on your system.

Examples

Example 1

The following example first creates a plan file by issuing an RMU Backup command with the Parallel and List_Plan qualifiers. Oracle RMU does not execute the plan file because the Noexecute qualifier is specified. The second command issues the RMU Backup Plan command to execute the plan file created with the RMU Backup command.

```
$ ! Create the Backup plan file:
$ !
$ RMU/BACKUP/PARALLEL=(EXEC=4, NODE=(NODE1, NODE2)) -
_$ /LIST_PLAN=(PARTIAL.PLAN)/NOEXECUTE/INCLUDE=(RDB$SYSTEM, -
_$ EMPIDS_LOW, EMPIDS_MID, EMPIDS_OVER, SALARY_HISTORY, EMP_INFO) -
_$ /LABEL=(001, 002, 003, 004, 005, 006, 007, 008, 009) -
_$ /CHECKSUM_VERIFICATION -
_$ MF_PERSONNEL TAPE1:MF_PARTIAL.RBF, TAPE2:, TAPE3, TAPE4
$ !
$ ! Execute the plan file created with the previous command:
$ !
$ RMU/BACKUP/PLAN partial.plan

$ TYPE partial.plan
! Plan created on 3-JUL-1996 by RMU/BACKUP.

Plan Name = PARTIAL
Plan Type = BACKUP
```

1.12 RMU Backup Plan Command

```
Plan Parameters:
    Database Root File = DISK1:[DB]MF_PERSONNEL.RDB;1
    Backup File = MF_PARTIAL.RBF
    ! Journal = specification for journal file
    ! Tape_Expiration = dd-mmm-yyyy
    ! Active_IO = number of buffers for each tape
    ! Protection = file system protection for backup file
    ! Block_Size = bytes per tape block
    ! Density = tape density
    ![No]Group_Size = number of blocks between XOR blocks
    ! Lock_Timeout = number of second to wait for locks
    ! Owner = identifier of owner of the backup file
    ! Page_Buffers = number of buffers to use for each storage area
    Checksum_Verification
    CRC=AUTODIN_II
    NoIncremental
    ! Accept_labels preserves all tape labels
    Log
    ! Loader_synchronization labels tapes in order across drives
    ! Media_loader forces support of a tape media loader
    NoOnline
    Quiet_Point
    NoRewind
    Statistics
    ACL
    ![No]Scan_Optimization
    Labels = (-
              001          -
              002          -
              003          -
              004          -
              005          -
              006          -
              007          -
              008          -
              009         )
End Plan Parameters
Executor Parameters :
    Executor Name = COORDINATOR
    Executor Type = Coordinator
End Executor Parameters
```

```
Executor Parameters :
    Executor Name = WORKER_001
    Executor Type = Worker
    Executor Node = NODE1
    Start Storage Area List
        EMP_INFO,
        SALARY_HISTORY
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE1:
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_002
    Executor Type = Worker
    Executor Node = NODE2
    Start Storage Area List
        EMPIDS_LOW,
        RDB$SYSTEM
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE2:
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_003
    Executor Type = Worker
    Executor Node = NODE1
    Start Storage Area List
        EMPIDS_MID
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE3
        MASTER
    End Tape Drive List
End Executor Parameters
Executor Parameters :
    Executor Name = WORKER_004
    Executor Type = Worker
    Executor Node = NODE2
    Start Storage Area List
        EMPIDS_OVER
    End Storage Area List
    Tape Drive List
        Tape Drive = TAPE4
        MASTER
    End Tape Drive List
End Executor Parameters
```

Example 2

1.12 RMU Backup Plan Command

The following example shows the use of the /Progress_Report qualifier to display backup performance every 10 seconds.

```
$ RMU/BACKUP $1$DGA10:[DB]SAMPLE $1$DGA20:[BCK]SAMPLE /DISK /PROGRESS_REPORT=10
Read    18 MB ( 0%) at  18 MB/s, estimated completion time 14:10:41.15
   .
   .
   .
Read  3934 MB (99%) at  28 MB/s, estimated completion time 14:10:39.86
```

Read n MB = raw blocks read so far
(n%) = percent of total blocks read
n MB/s = transfer rate since last display
estimated completion time = recalculated using the current transfer rate

Restores from other than disk files do not display the percentage completed nor the estimated completion time:

```
$ RMU/RESTORE/NOCDD $1$MGA500:SAMPLE,$1$MGA600: /REWIND /PROG=10
Read    72 MB at   14 MB/s
Read   135 MB at   15 MB/s
   .
   .
   .
Read  1441 MB at   12 MB/s
```

Example 3

The following example shows some of the verification diagnostics that can be output when the database root is verified by RMU/BACKUP.

```
$ rmu/backup/nolog mf_personnel mfp.rbf
%RMU-E-BADMAXPNO, unable to read last page (1052) in file
DEVICE:[DIRECTORY]DEPARTMENTS.RDA;1
%RMU-F-ABORTVER,  fatal error encountered; aborting verification
%RMU-F-FATALERR, fatal error on BACKUP
%RMU-F-FTL_BCK, Fatal error for BACKUP operation at 11-MAY-2009
11:16:59.46
$ rmu/backup/noquiet/online/nolog mf_personnel mfp
%RMU-W-BADPROID, DEVICE:[DIRECTORY]EMPIDS_MID.SNP;1
 file contains a bad identifier
 Expected "RDMSDATA", found "NOTRDBDB"
%RMU-W-INVALFILE, inconsistent database file
 DEVICE:[DIRECTORY]EMPIDS_MID.SNP;1
%RMU-F-ABORTVER,  fatal error encountered; aborting verification
%RMU-F-FATALERR, fatal error on BACKUP
%RMU-F-FTL_BCK, Fatal error for BACKUP operation at 11-MAY-2009
11:11:25.68
```

Example 4

The following example shows that the RMU/BACKUP *.PLAN file "[No]Database_Verification" option is set based on the "/DATABASE_ VERIFICATION" qualifier on the RMU/BACKUP command line. The default is "Database_Verification" so unless "/NODATABASE_VERIFICATION" is specified the "Database_Verification" option will be set in the *.PLAN file.

```
$ RMU/BACKUP/PARALLEL=EXEC=1/DISK=WRITER=2/LIST=TMP.PLAN/ONLINE-
  MF_PERSONNEL DISK:[DIRECTORY1]MFP,DISK:[DIRECTORY2]
$ SEAR TMP.PLAN DATABASE_VERIFICATION
    Database_Verification
$ RMU/BACKUP/PARALLEL=EXEC=1/DISK=WRITER=2/LIST=TMP.PLAN/ONLINE-
  /DATABASE_VERIFICATION MF_PERSONNEL DISK:[DIRECTORY1]MFP,-
  DISK:[DIRECTORY2]
$ SEAR TMP.PLAN DATABASE_VERIFICATION
    Database_Verification
$ RMU/BACKUP/PARALLEL=EXEC=1/DISK=WRITER=2/LIST=TMP.PLAN/ONLINE-
  /NODATABASE_VERIFICATION MF_PERSONNEL -
  DISK:[DIRECTORY1]MFP,DISK:[DIRECTORY2]
$ SEAR TMP.PLAN DATABASE_VERIFICATION
    NoDatabase_Verification
$ RMU/BACKUP/PLAN TMP.PLAN
```

1.13 RMU Checkpoint Command

When fast commit is enabled, requests that each active database process on each node flush updated database pages from its buffer pool to disk.

Format

RMU/Checkpoint root-file-spec

Command Qualifiers	Default
/[No]Wait[/Until=date-and-time]	/Wait

Description

Usually, each process performs a checkpoint operation after a certain set of thresholds has been exceeded. The RMU Checkpoint command allows you to spontaneously force each process to perform a checkpoint operation.

Performing a checkpoint operation is useful for several purposes. A checkpoint operation with the Wait qualifier causes all updated database pages to be flushed to disk. A checkpoint operation also improves the redo performance of the database recovery (DBR) process (although the per-process parameters should have already been properly initialized with this goal in mind).

When the Checkpoint command with the Wait qualifier completes (the system prompt is returned), all active processes have successfully performed a checkpoint operation.

When the system prompt is returned after you issue the Checkpoint command with the Nowait qualifier, there is no guarantee that all active processes have successfully performed a checkpoint operation.

Command Parameters

root-file-spec
The root file specification for the database you want to checkpoint. You can use either a full or partial file specification, or a logical name.

If you specify only a file name, Oracle Rdb looks for the database in the current default directory. If you do not specify a file extension, Oracle Rdb assumes a file extension of .rdb.

Command Qualifiers

Wait[/Until=date-and-time]
Nowait
Specifies whether or not the system prompt is to be returned before the checkpoint operation completes.

When you specify the Wait qualifier without the Until qualifier, the system prompt is not returned to you until all processes have flushed updated database pages to disk. The Wait qualifier is the default.

Used with the Wait qualifier, the Until qualifier specifies the time at which the RMU Checkpoint/Wait command stops waiting for the checkpoint and returns an error message. If you do not specify the Until qualifier, the wait is indefinite.

When you specify the Nowait qualifier, the system prompt is returned immediately, before all processes have flushed database pages to disk. In addition, when you specify the Nowait qualifier, there is no guarantee that all processes will flush their database pages to disk.

The Nowait qualifier is useful when it is more essential that the system prompt be returned immediately than it is to be certain that all processes have checkpointed.

Usage Notes

- To use the RMU Checkpoint command for a database, you must have the RMU$BACKUP or RMU$OPEN privilege in the root file access control list (ACL) for the database or you must have the OpenVMS WORLD privilege.
- The RMU Checkpoint command is useful only if the database fast commit feature has been enabled. If the fast commit feature is disabled, this command does nothing.

  For more information on the fast commit feature, see the FAST COMMIT IS ENABLED section of the SQL ALTER DATABASE statement in the *Oracle Rdb SQL Reference Manual*.

Examples

Example 1

The following command causes all the active database processes on all nodes to immediately perform a checkpoint operation:

```
$ RMU/CHECKPOINT MF_PERSONNEL.RDB
```

Example 2

The following command requests that all the active database processes on all nodes perform a checkpoint operation and that the system prompt be returned to you immediately. In this case, there is no guarantee that all processes will actually perform a checkpoint operation.

```
$ RMU/CHECKPOINT/NOWAIT MF_PERSONNEL.RDB
```

1.14 RMU Close Command

Closes an open database.

You should always specify the Wait qualifier, unless you are attempting to recover from some failure. When you specify the Wait qualifier, Oracle RMU performs all the auxiliary actions required to close and recover the database clusterwide, and it does not return the system prompt until those actions have been completed.

If you use the RMU Close command with the Nowait qualifier, the database must be open on the node where you issue the command. Otherwise, you will receive an error message stating that the database is not known. The system prompt is returned immediately, but it is only an indication that the database will be closed as soon as all other users have finished accessing the database. Therefore, the Wait qualifier is used almost exclusively.

Format

RMU/Close root-file-spec [,...]

Command Qualifiers	Defaults
/[No]Abort=option	/Abort=Forcex
/[No]Cluster	See description
/Path	None
/[No]Statistics=Export	/Nostatistics
/[No]Wait	/Nowait

Description

The RMU Close command closes an open database. A database root file is considered open if it has been specified in a previous RMU Open command or has active users attached to it.

You can close the database immediately by specifying the Abort qualifier, or you can allow current users to finish their session by specifying the Noabort qualifier.

If you have specified manual opening for your database (with the OPEN IS MANUAL clause of the SQL ALTER DATABASE statement), you must use the RMU Open command to manually open the database before any users can invoke it and the RMU Close command to manually close the database.

1.14 RMU Close Command

If you have specified automatic opening for your database (with the OPEN IS AUTOMATIC clause of the SQL ALTER DATABASE statement), the RMU Close command affects current database users only. Current processes are detached from the database but they and new processes can immediately reattach to the database.

Use the RMU Show Users command to display information about databases currently in use on your node. Use the RMU Dump Users command to display information about databases currently in use on your cluster.

Command Parameters

root-file-spec[,...]
An open database root file. The default file extension is .rdb.

Command Qualifiers

Abort=option
Noabort
Specifies whether to close the database immediately or allow processes to complete.

The Abort qualifier has two options. Both refer to OpenVMS system services. The options are as follows:

- Forcex

  When you use the Forcex (forced exit) option, recovery units are recovered and no recovery-unit journal (.ruj) files are left in the directories. Therefore, the RMU Backup command works. The option cannot force an exit of a database process with a spawned subprocess or a suspended or swapped out process. It aborts batch jobs that are using the database. Forcex is the default.

- Delprc

  When you use the Delprc (delete process) option, recovery units are not recovered. The .ruj files are left in the directories to be recovered on the next invocation of the database. The processes and any subprocesses of all database users are deleted, thereby deleting the processes from the database. Therefore, if you attempt to issue an RMU Backup command, you might receive the following error message:

  ```
  %RMU-F-MUSTRECDB, database must be closed or recovered
  ```

The Delprc and Forcex options are based on OpenVMS system services $DELPRC and $FORCEX. Refer to the OpenVMS documentation set for more information.

With the Noabort option, users already attached to the database can continue, and the root file global sections remain mapped in the virtual address file contents until all users exit the database. No new users will be allowed to attach to the database. When all current images terminate, Oracle RMU closes the database.

Cluster
Nocluster

Specifying the Cluster qualifier causes Oracle RMU to attempt to close a database on all nodes in a clustered environment that currently have the database open. Specifying the Cluster qualifier is similar to issuing the RMU Close command on every node in the cluster. Specifying the Nocluster qualifier causes Oracle RMU to close a database only on the cluster node from which you issue the RMU Close command.

The default is the Cluster qualifier if you specify the Wait qualifier. The default is the Nocluster qualifier if you specify the Nowait qualifier.

The following list describes the behavior of the command when you use various combinations of the [No]Cluster and [No]Wait qualifiers together in the same command line:

- Cluster and Wait

  When you specify the Cluster and Wait qualifiers, the RMU Close command closes the database on every node in the cluster, even if the database is not opened on the node from which the command is issued.

  Because you specified the Cluster and Wait qualifiers, the RMU Close command closes and recovers the database on every node in the cluster before the DCL prompt is returned to you.

- Cluster and Nowait

  When you specify the Cluster and Nowait qualifiers, the RMU Close command attempts to close the database on every node in the cluster. If the database is not opened on the node from which the Oracle RMU command is issued, the command cannot close the database on any node, and you receive the following error message:

```
%RDMS-F-CANTCLOSEDB, database could not be closed as requested
-RDMS-F-DBNOTACTIVE, database is not being used
%RMU-W-FATALERR, fatal error on DISK1:[USER1]DATABASE.RDB;1
```

  Because you used the Nowait qualifier, the database might not yet be closed on one or more nodes when the DCL prompt is returned to you. When you specify the Nowait qualifier, you can receive SYS-F-ACCONFLICT errors when you attempt to access a database after you have issued the RMU Close command with the Cluster and Nowait qualifiers and the DCL

prompt has been returned, but the monitor has not yet closed the database on all nodes in the cluster.

- Nocluster and Wait

  This combination provides the ability to have database shutdown complete on the local node before Oracle RMU returns to the DCL prompt.

- Nocluster and Nowait

  When you specify the Nocluster and Nowait qualifiers, Oracle RMU closes the database only on the node from which you issue the command, regardless of whether or not the database is open on other nodes.

  Because you used the Nowait qualifier, the database might not yet be closed on the node from which you issued the command when the DCL prompt is returned to you. With the Nowait qualifier, you can receive SYS-F-ACCONFLICT errors when you attempt to access a database after you have issued the RMU Close command with the Cluster and Nowait qualifiers and the DCL prompt has been returned, but the monitor has not yet closed the database on all nodes in the cluster.

Path

Specifies the full or relative data dictionary path name in which the definitions reside for the database you want to close.

The Path qualifier is a positional qualifier. Positional qualifiers operate on specific parameters based on the placement of the qualifiers in the command line. The path name cannot include wildcard characters.

Statistics=Export
Nostatistics

Specifies that statistic information is to be saved when the database is closed. The default is Nostatistics, which indicates that statistic information is not preserved when the database is closed.

Clusterwide statistic information is not stored in the statistic file, which allows you to decide on which nodes the statistic information should be initially loaded when the database is opened.

The statistic information is stored in a node-specific database file located in the same directory as the database root file. The file has the same name as the root-file-spec, with a default file extension of .rds. Because the statistic files contain node-specific information, they cannot be renamed or copied. They can be deleted if they are no longer needed.

The Statistics=Export qualifier cannot be specified in conjunction with the Cluster qualifier. To preserve the statistics information for a database open on a cluster, you must specifically close the individual nodes.

The RMU Backup command does not save the statistics files. They are considered temporary files and not part of the database.

Wait
Nowait
Specify the Wait qualifier to cause Oracle RMU to close and recover the database before the system prompt is returned to you.

The default is the Nowait qualifier. With the Nowait qualifier, the database might not be closed when the system prompt is returned to you. You can receive errors when you attempt to access a database after you issued the RMU Close command and the system prompt is returned, but before the monitor has closed the database.

See the Usage Notes for restrictions on using the Wait qualifier.

Usage Notes

- To use the RMU Close command for a database, you must have the RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- To use the Wait qualifier, Oracle RMU requires that the database be recoverable for correct operation. It must be possible to attach to the database on a node where it is opened. There are database recovery failures that preclude further attaches to the database. When such a failure occurs, any attempt to attach to the database (for example, with an SQL ATTACH statement) causes the process to be deleted from the system. In other words, you are logged out.

  In this situation, the RMU Close command with the Wait qualifier has the same effect as the RMU Close command with the Cluster and Nowait qualifiers. The operation does not wait, and it does not close the database unless it is opened on the node from which you issue the Oracle RMU command.

  If you encounter this situation, enter the following command from a node on which the database is open to close the database:

```
$ RMU/CLOSE/CLUSTER/NOWAIT/ABORT=DELPRC
```

Examples

Example 1

When you issue the following command from a node in a cluster, the Cluster qualifier shuts down the database for the entire cluster, even if no users are on the node from which you issued the command. The Wait qualifier causes Oracle RMU to close the database before the system prompt is returned.

```
$ RMU/CLOSE/CLUSTER/WAIT MF_PERSONNEL.RDB
```

Example 2

The following command closes the mf_personnel database in the [.WORK] directory, all the databases in the [.TEST] directory, and the databases specified by the path names CDD$TOP.FINANCE and SAMPLE_DB:

```
$ RMU/CLOSE DISK1:[WORK]MF_PERSONNEL, CDD$TOP.FINANCE/PATH, -
_$ DISK1:[TEST]*, SAMPLE_DB/PATH
```

1.15 RMU Collect Optimizer_Statistics Command

Collects cardinality and storage statistics for the Oracle Rdb optimizer. Also collects workload statistics if a workload profile has been generated.

Format

RMU/Collect Optimizer_Statistics root-file

Command Qualifiers	Defaults
/Exclude_Tables=(table-list)	None
/[No]Indexes[=(index-list)]	/Indexes
/[No]Log[=file-name]	Current DCL verify value
/Row_Count=n	/Row_Count=500
/Statistics[=(options)]	/Statistics
/[No]System_Relations	/Nosystem_Relations
/[No]Tables[=(table-list)]	/Tables
/Transaction_Type=option	/Transaction_Type=Automatic

Description

The purpose of collecting optimizer statistics is to maintain up-to-date statistics that the Oracle Rdb optimizer uses to determine solution costs and cardinalities during query optimization.

You can collect cardinality and storage statistics by issuing the RMU Collect Optimizer_Statistics command. You can direct Oracle RMU to collect these statistics for particular tables or indexes by using the Tables, System_ Relations, or Indexes qualifiers.

Before you can collect *workload* statistics, you must first generate a workload profile with SQL. The following list describes the general procedure for generating a workload profile and collecting workload statistics:

1. Enable workload profiling with the WORKLOAD COLLECTION IS ENABLED clause of the SQL ALTER DATABASE or SQL CREATE DATABASE statement.

   SQL creates a new system table called RDB$WORKLOAD.

2. Execute the queries for which you want the Oracle Rdb optimizer to have the best possible statistics.

When you execute the queries, the optimizer determines which groups of columns are important for optimal processing of the query. These groups of columns are referred to as **workload column groups**. Note that a workload column group may actually contain only one column.

Each set of workload column groups is entered as a row in the RDB$WORKLOAD system table.

At this point, the only data in the RDB$WORKLOAD system table are the workload column groups, the tables with which the column group is associated, and the date they were entered into the table. No statistics are currently recorded in the RDB$WORKLOAD system table.

3. In most cases, now you disable workload profiling with the SQL ALTER DATABASE WORKLOAD COLLECTION IS DISABLED clause.

   Queries executed after you disable workload profiling are not scanned by the Oracle Rdb optimizer for workload column groups. You can leave the workload profiling enabled if the same queries are always executed. In such a case, no new rows are entered into the RDB$WORKLOAD system table. However, if you anticipate that queries will be executed for which you do not want workload profiling to be enabled, you need to disable workload collection.

4. Execute an RMU Collect Optimizer_Statistics command with the Statistics=(Workload) qualifier.

   Oracle RMU reads the RDB$WORKLOAD system table to determine for which column groups it should collect statistics, and then collects the statistics.

5. Execute the queries previously profiled again.

   The optimizer uses the statistics gathered by Oracle RMU to make a best effort at optimizing the profiled queries.

The following list provides some guidelines on when to issue the RMU Collect Optimizer_Statistics command and which Statistics qualifier options you should use:

- You should enable workload profiling and execute the RMU Collect Optimizer_Statistics command with the Statistics=(Workload) qualifier when you introduce new, complex, frequently used queries as part of your regular work.
- You should execute the RMU Collect Optimizer_Statistics command with the Statistics=(Storage) qualifier after you add metadata, such as new tables or indexes, to the database. In this case, you do not need to reenable workload profiling.

- You should execute the RMU Collect Optimizer_Statistics command with the Statistics=(Storage, Workload) qualifier when the data in the database has significantly increased, decreased, or changed. In this case, you do not need to reenable workload profiling.

The statistics you can gather with the RMU Collect Optimizer_Statistics command and a description of how the optimizer uses these statistics are summarized in Table 1–6.

Table 1–6 Statistics Gathered by the RMU Collect Optimizer_Statistics Command

Cardinality Statistics		
Statistic Gathered:	**Definition:**	**Used by Optimizer to:**
Table Cardinality	Number of rows in a table.	Determine solution cardinality.
Index Cardinality	Number of distinct key values in an index.	Estimate the number of index keys returned.
Index Prefix Cardinality	Number of distinct key values in leading parts of a multi-segmented B-tree index.	Estimate the number of index keys returned based on a sorted index range.
Workload Statistics		
Statistic Gathered:	**Definition:**	**Used by Optimizer to:**
Column Group Duplicity Factor	Average number of duplicates per distinct value in a column group. This is an estimated value.	Determine strategies for equiselections (selections with the IS NULL predicate or selection predicates with the equals (=) operator), equijoins, grouped aggregation (for example, the SQL GROUP BY clause), or projection operations (for example, the SQL DISTINCT clause).
Column Group Null Factor	Number of table rows with a NULL value in at least one column of a column group. This is an estimated value.	Estimate the effects of NULL data on equijoins and equiselections (because they imply the removal of rows with NULL values). Also used for estimating the cardinality of an outer join result.

(continued on next page)

Table 1–6 (Cont.) Statistics Gathered by the RMU Collect Optimizer_Statistics Command

Storage Statistics		
Statistic Gathered:	**Definition:**	**Used by Optimizer to:**
Average Index Depth (sorted indexes only)	Average number of levels to traverse on a B-tree descent.	Estimate the cost of descending the B-tree. (A cross join with an inner table that is accessed by a sorted index involves repetitive B-tree descents.)
Index Key Clustering Factor	Average number of I/Os required to read one index key and all associated dbkeys during a hashed key lookup or a B-tree index scan, excluding the B-tree descent.	Improve the cost estimate of performing an index-only retrieval for hashed and sorted indexes.
Index Data Clustering Factor	Average number of I/Os required to fetch data rows using dbkeys associated with an index key.	Estimate the cost for fetching data rows from a sorted index scan or from a hash bucket.
Table Row Clustering Factor	The average number of I/Os required to read one row during a sequential scan of a table.	Estimate the cost of performing a sequential scan of a table.

Command Parameters

root-file-spec
Specifies the database for which statistics are to be collected. The default file type is .rdb.

Command Qualifiers

Exclude_Tables
Exclude_Tables=(table-list)
Specifies a list of database tables to be excluded from statistics collection and update for statistics used by the Rdb query optimizer. You must specify at least one list. You can specify an options file in place of a list of tables.

If the Exclude_Tables qualifier is used with the Tables qualifier in the same RMU Collect Optimizer command, the Exclude_Tables qualifier takes precedence. If the same table is specified in the table list for both qualifiers, that table is excluded from the statistics collection and update.

Indexes
Indexes=(index-list)
Noindex
Specifies the index or indexes for which statistics are to be collected. If you do not specify an index-list, statistics for all indexes defined for the tables specified with the Tables qualifier are collected. If you specify an index-list, statistics are collected only for the named indexes. If you specify the Noindex qualifier, statistics for the index cardinality, average index depth, index key clustering factor, and index data clustering factor are not collected.

Specify the Notable qualifier if you do not want statistics collected for tables. (Remember, the Tables qualifier without a table-list is the default.)

The default is the Indexes qualifier without an index-list.

Log
Log=file-name
Nolog
Specifies how the values calculated for the statistics are to be logged. Specify the Log qualifier to have the information displayed to SYS$OUTPUT. Specify the Log=file-spec qualifier to have the information written to a file. Specify the Nolog qualifier to prevent display of the information. If you do not specify any of variation of the Log qualifier, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Row_Count=n
Specifies the number of rows that are sent in a single I/O request when Workload Statistics are collected. You can experiment to find the value for *n* that provides the best performance and memory usage for your database and environment.

As you increase the value of *n*, you see an increase in performance at the expense of additional memory and overhead.

The minimum value you can specify for *n* is 1. The default value for *n* is 500.

Statistics[=(options)]
Specifies the type of statistics you want to collect for the items specified with the Tables, System_Relations, and Indexes qualifiers. If you specify the Statistics qualifier without an options list, all statistics are collected for the items specified.

1.15 RMU Collect Optimizer_Statistics Command

If you specify the Statistics qualifier with an options list, Oracle RMU collects types of statistics described in the following list. If you specify more than one option, separate the options with commas and enclose the options within parenthesis.

The Statistics qualifier options are:

- Cardinality

  Collects the table cardinality for the tables specified with the Tables and System_Relations qualifiers and the index and index prefix cardinalities for the indexes specified with the Indexes qualifier. Because cardinalities are automatically maintained by Oracle Rdb, it is usually not necessary to collect cardinality statistics using the RMU Collect Optimizer_Statistics command unless you have previously explicitly disabled cardinality updates.

- Workload

  Collects the Column Group, Duplicity Factor, and Null Factor workload statistics for the tables specified with the Tables and System_Relations qualifiers.

- Storage

  Collects the following statistics:

  – Table Row Clustering Factor for the tables specified with the Tables qualifier

  – Index Key Clustering Factor, the Index Data Clustering Factor, and the Average Index Depth for the indexes specified with the Indexes qualifier

  See Table 1–7 for information on the columns and tables used in the system relations to store these statistics.

System_Relations
Nosystem_Relations

Specifies that optimizer statistics are to be collected for system tables (relations) and their associated indexes.

If you do not specify the System_Relations qualifier, or if you specify the Nosystem_Relations qualifier, optimizer statistics are not collected for system tables or their associated indexes.

Specify the Noindex qualifier if you do not want statistics collected for indexes defined on the system tables.

The default is the Nosystem_Relations qualifier.

Tables
Tables[=(table-list)]
Notables
Specifies the table or tables for which statistics are to be collected. If you specify a table-list, statistics for those tables and their associated indexes are collected. If you do not specify a table-list, statistics for all tables and their associated indexes in the database are collected. If you do not specify the Table qualifier, statistics for all tables are collected. If you specify the Notables qualifier, statistics for for the table cardinality, table row clustering factor, column group duplicity factor, and column group null factor are not collected.

Specify the Noindex qualifier if you do not want statistics collected for indexes.

The Tables qualifier without a table-list is the default.

Transaction_Type=option
Allows you to specify the transaction mode for the transactions used to collect statistics. Valid options are:

- Automatic
- Read_Only
- Noread_Only

You must specify an option if you use this qualifier.

If you do not use any form of this qualifier, the Transaction_Type=Automatic qualifier is the default. This qualifier specifies that Oracle RMU is to determine the transaction mode used to collect statistics. If any storage area in the database (including those not accessed for collecting statistics) has snapshots disabled, the transactions used to collect data are set to read/write mode. Otherwise, the transactions to collect data are set to read-only mode.

The Transaction_Type=Read_Only qualifier specifies the transactions used to collect statistics be set to read-only mode. When you explicitly set the transaction type to read-only, snapshots need not be enabled for all storage areas in the database, but must be enabled for those storage areas from which statistics are collected. Otherwise, you receive an error and the collect optimizer statistics operation fails.

You might select this option if not all storage areas have snapshots enabled and you are collecting statistics on objects that are stored only in storage areas with snapshots enabled. In this case, using the Transaction_Type=Read_Only qualifier allows you to collect statistics and impose minimal locking on other users of the database.

The Transaction_Type=Noread_Only qualifier specifies that the transactions used to collect statistics be set to read/write mode. You might select this option if you want to eradicate the growth of snapshot files that occurs during a read-only transaction and are willing to incur the cost of increased locking that occurs during a read/write transaction.

Usage Notes

- To use the RMU Collect Optimizer_Statistics command for a database, you must have the RMU$ANALYZE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

  You must also have the SQL SELECT privilege on the database and other objects for which the collect is being performed.

- When you use the SQL ALTER DATABASE statement to set the RDB$SYSTEM storage area (or DEFAULT storage area) to read-only access for your database, the Oracle Rdb system tables are also set to read-only access. When the Oracle Rdb system tables are set to read-only access:

  - Automatic updates to table and index cardinality are disabled.
  - Manual changes made to the cardinalities to influence the optimizer are not allowed.
  - The I/O associated with the cardinality update is eliminated.
  - When you run RMU Collect Optimizer_Statistics on such databases, the RDB$SYSTEM storage area and, if necessary, the DEFAULT storage area will be changed to READ WRITE to allow updates to the system tables and then will be restored to READ ONLY.

- For indexes, the cardinality value is the number of unique entries for an index that allows duplicates. If the index is unique, Oracle Rdb stores zero for the cardinality, and uses the table cardinality instead. For tables, the cardinality value is the number of rows in the table. Oracle Rdb uses the cardinality values of indexes and tables to influence decisions made by the optimizer. If the actual cardinality values of tables and indexes are different from the stored cardinality values, the optimizer's performance can be adversely affected.

- As Oracle RMU performs the collect operation, it displays the maximum memory required to perform the operation. If the maximum amount required is not available, Oracle RMU makes adjustments to try to make use of the memory that is available. However, if after making these adjustments, memory is still insufficient, the collect operation skips the updates for the table causing the problem and continues with the operation. The skipped table is noted in the log file with the message, "Unable to allocate memory for <table-name>; default statistics values used."

  To avoid this problem, use the OpenVMS System Generation Utility (SYSGEN) to increase the VIRTUALPAGECNT parameter.

- If you prefer not to update optimizer statistics all at once, you can divide the work into separate commands. Oracle Corporation recommends that you collect Cardinality and Storage statistics in one RMU Collect Optimizer_Statistics command; and collect Workload statistics in a second command.

- You must decide if the improved performance provided by enabling and maintaining the workload profile is worth the cost. Generally speaking, it is worth the cost of maintaining this table for a stable set of queries that are run on a regular basis; it is not worth the cost of maintaining this table when the majority of your queries are ad hoc queries, each of which uses different access strategies.

  For example, if the majority of queries that access the EMPLOYEES table use the EMPLOYEE_ID as the selection criteria and the queries are using the same access strategy, you might want to maintain a workload profile for the EMPLOYEES table. However, if some queries access the EMPLOYEES table through the EMPLOYEE_ID, some through the LAST_NAME, and others through the STATE, in an unpredictable manner, the queries are using different access strategies for which you probably do not want to maintain a workload profile.

- Index prefix cardinalities are cumulative values. For example, suppose an index contains three segments and the first segment has a cardinality of A; the second has a cardinality of B; and the third has a cardinality of C. Then the index prefix cardinality for the first segment is A; the index prefix cardinality for the second segment is A concatenated with B (A | B); and the index prefix cardinality for the third segment is A concatenated with B concatenated with C (A | B | C). Therefore, the prefix cardinality for last segment in an index is always equal to the total cardinality for the index. Likewise, if the index only contains one segment, the index prefix cardinality is equal to the total cardinality for the index. In these cases, because the index prefix cardinality is the same as the total index

cardinality, Oracle RMU does not calculate an index prefix cardinality. Instead, Oracle RMU stores a value of "0" for the index prefix cardinality and the optimizer uses the value stored for the total index cardinality.

- Cardinality statistics are automatically maintained by Oracle Rdb. Physical storage and workload statistics are only collected when you issue an RMU Collect Optimizer_Statistics command. To get information about the usage of physical storage and workload statistics for a given query, define the RDMS$SET_FLAGS logical name to be "ESTIMATES". For example:

  ```
  $ DEFINE RDMS$SET_FLAGS "ESTIMATES"
  ```

  When you execute a query, if workload and physical statistics have been used in optimizing the query, you see a line such as the following in the command output:

  ```
  ~O: Workload and Physical statistics used
  ```

- Detected asynchronous prefetch should be enabled to achieve the best performance of this command. You can determine the setting for your database by issuing the RMU Dump command with the Header qualifier.

  If detected asynchronous prefetch is disabled, and you do not want to enable it for the database, you can enable it for your Oracle RMU operations by defining the following logicals at the process level:

  ```
  $ DEFINE RDM$BIND_DAPF_ENABLED 1
  $ DEFINE RDM$BIND_DAPF_DEPTH_BUF_CNT P1
  ```

  P1 is a value between 10 and 20 percent of the user buffer count.

- You can delete entries from the workload profile with the RMU Delete Optimizer_Statistics command. See Section 1.18 for details.
- You can display entries from the workload profile with the RMU Show Optimizer_Statistics command. See Section 1.67.7 for details.
- Table 1–7 provides a summary of the system tables in which statistics gathered by the RMU Collect Optimizer_Statistics command are stored.

Table 1–7 System Tables Used to Store Optimizer Statistics

Statistic	System Table Name	Column Name
Table Cardinality	RDB$RELATIONS	RDB$CARDINALITY
Table Row Clustering Factor	RDB$RELATIONS	RDB$ROW_CLUSTER_FACTOR
Column Group Duplicity Factor	RDB$WORKLOAD	RDB$DUPLICITY_FACTOR
Column Group Null Factor	RDB$WORKLOAD	RDB$NULL_FACTOR
Index Cardinality	RDB$INDICES	RDB$CARDINALITY
Index Prefix Cardinality	RDB$INDEX_SEGMENTS	RDB$CARDINALITY
Average Index Depth (B-Trees only)	RDB$INDICES	RDB$INDEX_DEPTH
Index Key Clustering Factor	RDB$INDICES	RDB$KEY_CLUSTER_FACTOR
Index Data Clustering Factor	RDB$INDICES	RDB$DATA_CLUSTER_FACTOR

Examples

Example 1

The following example collects cardinality statistics for the EMPLOYEES and JOB_HISTORY tables and their associated indexes. See the Usage Notes for an explanation for the value "0" for the index prefix cardinality.

```
$ RMU/COLLECT OPTIMIZER_STATISTICS mf_personnel.rdb -
_$ /STATISTICS=(CARDINALITY)/TABLES=(EMPLOYEES, JOB_HISTORY) -
_$ /INDEXES=(EMP_LAST_NAME,EMP_EMPLOYEE_ID, EMPLOYEES_HASH, -
_$  JH_EMPLOYEE_ID, JOB_HISTORY_HASH)/LOG
Start loading tables... at  3-JUL-1996 09:35:25.19
Done loading tables.... at  3-JUL-1996 09:35:25.91
Start loading indexes... at  3-JUL-1996 09:35:25.92
Done loading indexes.... at  3-JUL-1996 09:35:26.49
Start collecting btree index stats... at  3-JUL-1996 09:35:28.17
Done collecting btree index stats.... at  3-JUL-1996 09:35:28.23
Start collecting table & hash index stats... at  3-JUL-1996 09:35:28.23
Done collecting table & hash index stats.... at  3-JUL-1996 09:35:28.52
Start calculating stats... at  3-JUL-1996 09:35:28.76
Done calculating stats.... at  3-JUL-1996 09:35:28.76
Start writing stats... at  3-JUL-1996 09:35:30.16
------------------------------------------------------------------------
```

1.15 RMU Collect Optimizer_Statistics Command

```
Optimizer Statistics collected for table : EMPLOYEES

  Cardinality           : 100

Index name : EMP_LAST_NAME
  Index Cardinality     : 83
  Segment Column                Prefix cardinality
    LAST_NAME                      0

Index name : EMP_EMPLOYEE_ID
  Index Cardinality     : 100
  Segment Column                Prefix cardinality
    EMPLOYEE_ID                    0

Index name : EMPLOYEES_HASH
  Index Cardinality     : 100

------------------------------------------------------------------------

Optimizer Statistics collected for table : JOB_HISTORY

  Cardinality           : 274

Index name : JH_EMPLOYEE_ID
  Index Cardinality     : 100
  Segment Column                Prefix cardinality
    EMPLOYEE_ID                    0

Index name : JOB_HISTORY_HASH
  Index Cardinality     : 100
Done writing stats.... at  3-JUL-1996 09:35:30.83
```

Example 2

The following example collects storage statistics for the EMPLOYEES and JOB_HISTORY TABLES and their associated indexes:

```
$ RMU/COLLECT OPTIMIZER_STATISTICS mf_personnel -
_$ /STATISTICS=(STORAGE)/TABLES=(EMPLOYEES, JOB_HISTORY) -
_$ /INDEXES=(EMP_LAST_NAME,EMP_EMPLOYEE_ID, EMPLOYEES_HASH, -
_$ JH_EMPLOYEE_ID, JOB_HISTORY_HASH)/LOG
Start loading tables... at  3-JUL-1996 10:28:49.39
Done loading tables.... at  3-JUL-1996 10:28:50.30
Start loading indexes... at  3-JUL-1996 10:28:50.30
Done loading indexes.... at  3-JUL-1996 10:28:51.03
Start collecting btree index stats... at  3-JUL-1996 10:28:53.27
Done collecting btree index stats.... at  3-JUL-1996 10:28:53.37
Start collecting table & hash index stats... at  3-JUL-1996 10:28:53.38
Done collecting table & hash index stats.... at  3-JUL-1996 10:28:53.80
Start calculating stats... at  3-JUL-1996 10:28:54.07
Done calculating stats.... at  3-JUL-1996 10:28:54.07
Start writing stats... at  3-JUL-1996 10:28:55.61
------------------------------------------------------------------------

Optimizer Statistics collected for table : EMPLOYEES
```

```
  Row clustering factor  : 0.2550000

Index name : EMP_LAST_NAME
  Average Depth          : 2.0000000
  Key clustering factor  : 0.0481928
  Data clustering factor : 1.1686747

Index name : EMP_EMPLOYEE_ID
  Average Depth          : 2.0000000
  Key clustering factor  : 0.0100000
  Data clustering factor : 0.9500000

Index name : EMPLOYEES_HASH
  Key clustering factor  : 1.0000000
  Data clustering factor : 1.0000000

------------------------------------------------------------------

Optimizer Statistics collected for table : JOB_HISTORY

  Row clustering factor  : 0.0930657

Index name : JH_EMPLOYEE_ID
  Average Depth          : 2.0000000
  Key clustering factor  : 0.0500000
  Data clustering factor : 0.9500000

Index name : JOB_HISTORY_HASH
  Key clustering factor  : 1.0000000
  Data clustering factor : 1.0000000
Done writing stats.... at  3-JUL-1996 10:28:56.41
```

Example 3

The following example enables workload collection with an SQL ALTER DATABASE statement, executes frequently run queries to generate a workload profile, collects workload statistics for the EMPLOYEES and JOB_HISTORY tables (along with their associated indexes), and then displays the statistics gathered.

The SQL natural left outer join causes the first and third workload column groups to be created. The SQL DISTINCT clause causes the second and fourth workload column groups to be created.

1.15 RMU Collect Optimizer_Statistics Command

```
$ ! Enable workload collection:
$ SQL
SQL> ALTER DATABASE FILENAME mf_personnel.rdb
cont> WORKLOAD COLLECTION IS ENABLED;
SQL> --
SQL> -- Execute frequently run SQL queries.
SQL> --
SQL> ATTACH 'FILENAME mf_personnel.rdb';
SQL> SELECT DISTINCT *
cont> FROM JOB_HISTORY NATURAL LEFT OUTER JOIN EMPLOYEES;
   .
   .
   .
SQL> DISCONNECT DEFAULT;
SQL> -- Disable workload collection:
SQL> ALTER DATABASE FILENAME mf_personnel.rdb
cont> WORKLOAD COLLECTION IS DISABLED;
SQL> EXIT;
$
$ ! Direct Oracle RMU to collect statistics for the EMPLOYEES and
$ ! JOB_HISTORY tables.
$ !
$ RMU/COLLECT OPTIMIZER_STATISTICS mf_personnel.rdb -
_$ /TABLE=(EMPLOYEES, JOB_HISTORY)/STATISTICS=(WORKLOAD)/LOG
Start loading tables... at  3-JUL-1996 10:40:00.22
Done loading tables.... at  3-JUL-1996 10:40:00.90
Start collecting workload stats... at  3-JUL-1996 10:40:03.43
Maximum memory required (bytes) = 6810
Done collecting workload stats.... at  3-JUL-1996 10:40:05.03
Start calculating stats... at  3-JUL-1996 10:40:05.32
Done calculating stats.... at  3-JUL-1996 10:40:05.32
Start writing stats... at  3-JUL-1996 10:40:06.91

-------------------------------------------------------------------------

Optimizer Statistics collected for table : EMPLOYEES

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 1.0000000
  Null factor            : 0.0000000

  Workload Column group  : LAST_NAME,   FIRST_NAME,     MIDDLE_INITIAL,
ADDRESS_DATA_1, ADDRESS_DATA_2, CITY,   STATE,  POSTAL_CODE,    SEX,
BIRTHDAY,       STATUS_CODE
  Duplicity factor       : 1.5625000
  Null factor            : 0.3600000

-------------------------------------------------------------------------
Optimizer Statistics collected for table : JOB_HISTORY

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 2.7040000
  Null factor            : 0.0000000
```

```
  Workload Column group  : EMPLOYEE_ID,    JOB_CODE,       JOB_START,
JOB_END,        DEPARTMENT_CODE,        SUPERVISOR_ID
  Duplicity factor       : 1.5420582
  Null factor            : 0.3649635
Done writing stats.... at  3-JUL-1996 10:40:07.46
```

Example 4

The following example collects all statistics (cardinality, workload, and storage) for all tables and indexes in the database except system relations. Output is written to the file stats_nosys.log.

```
$ RMU/COLLECT OPTIMIZER_STATISTICS mf_personnel.rdb -
_$ /LOG=stats_nosys.log
```

Example 5

The following example collects all statistics (cardinality, workload, and storage) for all tables, indexes, and system relations. Output is written to the file stats_all.log.

```
$ RMU/COLLECT OPTIMIZER_STATISTICS mf_personnel.rdb/SYSTEM_RELATIONS -
_$ /LOG=stats_all.log
```

Example 6

In the following example the Employees and Departments tables are excluded from statistics collection.

```
$ RMU/COLLECT OPTIMIZER_STATISTICS MF_PERSONNEL /LOG -
_$ /EXCLUDE_TABLES=(EMPLOYEES,DEPARTMENTS)
```

1.16 RMU Convert Command

Converts any of the following versions of Oracle Rdb databases to an Oracle Rdb release 7.3 database:

- Version 7.0
- Version 7.1
- Version 7.2

See the *Oracle Rdb Installation and Configuration Guide* for the proper backup procedure prior to installing a new release of Oracle Rdb and converting databases.

Note

The following are important issues to consider when you convert a database:

- A database *must* be backed up immediately following an Oracle RMU convert operation.

  A database converted using the RMU Convert command may not be recoverable if a full database backup is not made immediately after the convert operation completes. If you attempt to restore a database using a backup file created prior to the conversion, the database may be left in an unrecoverable state.

- If after-image journaling is enabled when you issue the Convert command, Oracle RMU disables after-image journaling during the convert operation and then does one of the following, depending on the type of .aij file or files being employed when the Convert command was issued:

  – If an extensible .aij file was being used, Oracle RMU creates a new journal for the converted database and enables after-image journaling.

  – If fixed-size .aij files were being used, Oracle RMU activates the next available fixed-size journal and enables after-image journaling. If another fixed-size journal is not available, journaling remains disabled.

  Use only the .aij file (or files) created or activated during or after the convert operation together with the backup file you created immediately after the convert operation to restore and recover

your database. Any .aij files created prior to the Convert operation cannot be used to recover the converted database.

If you issue an RMU Convert command with the Rollback qualifier, Oracle RMU disables after-image journaling and returns the message: RMU-I-CANTENAAIJ. Oracle Corporation recommends that you back up the database and enable after-image journaling when the convert operation completes.

- Growth of the RDB$SYSTEM storage area is normal during a convert operation. You must be sure that there is sufficient disk space for the new metadata and the converted metadata.

  During a convert operation Oracle RMU makes an upgraded copy of the metadata. If the convert operation fails, the old metadata is available for rolling back. If you specify the Nocommit qualifier, both copies of the metadata exist at the same time (to allow a manual rollback operation). If you specify the Commit qualifier, the old metadata is deleted once the convert operation completes successfully.

Read the Description section carefully for important information on converting single-file and multifile databases.

Format

RMU/Convert database-list

Command Qualifiers	Defaults
/[No]Commit	/Commit
/[No]Confirm	See description
/Path	None
/Reserve = (Area=n, Aij=n)	See description
/[No]Rollback	/Norollback

Description

The RMU Convert command operates by creating a converted copy of the system tables and indexes. This implies that the RDB$SYSTEM storage area might grow during the conversion, but it is unlikely that the system tables will be fragmented by the conversion process.

1.16 RMU Convert Command

Because a copy of the system tables is made and new system indices created, the time taken by the conversion is proportional to the number of rows in the system tables. This is typically a few minutes per database.

All system indices are recreated as SORTED RANKED to allow a more compact representation of duplicates and to allow Rdb to better manage page space for system metadata.

After the conversion, both copies of the system tables are stored in the database. The Commit qualifier selects the converted copy and deletes the original copy. The Rollback qualifier selects the original copy and deletes the converted copy. You can specify either the Commit or the Rollback qualifier at a later time if you selected the Nocommit qualifier when the database was converted. Be aware that as long as Commit or Rollback are not selected after a Nocommit conversion, extra space will be taken up in the database to store both versions of the metadata. It is important to issue the Convert/Commit command after you have verified that the conversion was successful. (RMU will not let you convert to a newer version if the previous Convert was never committed, even if it was years ago.)

While both copies of the system tables exist, the database is usable under Oracle Rdb release 7.3, but not under the earlier version. Also, DDL (data definition language) operations to the database are prohibited to ensure that both copies of the system tables remain consistent. After you specify either the Commit or the Rollback qualifier, you can again perform DDL operations on the database.

Command Parameters

database-list
The database-list parameter is a list of databases to be converted. A list item can be either the file specification of a database root file or a data dictionary path name.

You can use wildcards in the file specification of a database root file.

You cannot use wildcards in a data dictionary path name.

Command Qualifiers

Commit
Nocommit
Makes the database conversion permanent. When you specify the Commit qualifier, the database is converted to an Oracle Rdb release 7.3 database and cannot be returned to the previous version. The default is Commit.

When you specify the Nocommit qualifier, you can convert the database to Oracle Rdb release 7.3 and roll it back to the previous version at a later time.

Using the Nocommit qualifier is helpful when you want to test your applications against a new version of Oracle Rdb. In the event that you find problems, you can roll back to the previous version. Once you feel confident that your applications work well with the new version, you should commit the converted database, otherwise unnecessary space is taken up in the database to store the obsolete alternate version of the metadata.

Confirm
Noconfirm
Requests user input during the conversion procedure. When you specify the Confirm qualifier, Oracle RMU asks if you are satisfied with your database and aij backup files. If the database being converted has after-image journaling enabled, Oracle RMU asks if you want to continue and states that after-image journaling will be temporarily disabled.

The default is the Confirm qualifier when the conversion is executed interactively. The default is the Noconfirm qualifier when the conversion is executed from a batch job.

If you specify the Noconfirm qualifier, you are not prompted during the conversion procedure.

If you specify the Confirm qualifier when the conversion is executed from a batch job, you must supply the responses to the questions the Confirm qualifier generates in the line immediately after that in which the Convert qualifier appears. See Example 6.

Path
Identifies that the database is being specified by its data dictionary path name instead of its file specification. The Path qualifier is a positional qualifier.

Reserve=(Area=n, Aij=n)
Reserves space in the database root file for storage areas or .aij files, or both. Replace the character *n* with the number of storage areas or .aij files for which you want to reserve space.

Note that you cannot reserve areas for a single-file database. You can reserve .aij files for a single-file database, but once the database is converted, you cannot alter that reservation unless you backup and restore the database.

This qualifier is useful if, when you are converting your database, you anticipate the need for additional storage areas or .aij files. Because the addition of new storage areas or .aij files requires that users not be attached to the database, adding them while the database is being converted minimizes the time that the database is inaccessible to users.

By default, one .aij file and no storage area files are reserved.

Rollback
Norollback
Returns a database that has been converted to an Oracle Rdb release 7.3 database (but not committed) to the previous version. You might decide to return to the previous version of the database for technical, performance, or business reasons.

The Norollback qualifier prevents you from returning your converted database to the previous version. The default is the Norollback qualifier.

If you specify both the Nocommit qualifier and the Rollback qualifier in the same RMU Convert command, your database is converted to Oracle Rdb release 7.3 and immediately rolled back to the previous version when the RMU Convert command is executed.

This qualifier is valid only if you are converting from one of the following releases: 7.0, 7.1 or 7.2.

Usage Notes

- To use the RMU Convert command for a database, you must have the RMU$CONVERT or RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- The RMU Convert command requires read/write access to the database root file, the RDB$SYSTEM area, and the directory in which the .ruj file will be entered.
- Oracle Corporation recommends that you update multisegment index cardinalities as part of, or soon after, the convert operation completes.

  Stored cardinality values can differ from the actual cardinality values if the RDB$SYSTEM storage area has been set to read-only access.

- If the system storage area attribute is set to read-only access, the RMU Convert command proceeds to reset the attribute to read/write, convert the database and then reset the attribute to read-only when the conversion is complete. If the system storage areas are located on a device that cannot be written to, the database conversion fails and returns an error message. The system storage areas include RDB$SYSTEM, any area defined as the LIST storage area, and any area defined as the DEFAULT storage area.
- RMU Convert NoCommit retains two copies of the metadata in the Rdb system areas. The old system tables and indices from the prior version and the newly created system tables and indices. Only the new system tables are updated by DDL commands and this could lead to inconsistencies between the two sets of system tables, therefore Rdb avoids this problem by restricting most updates to the metadata to allow a future RMU Convert Rollback to be executed.

  Here are some features that will not function in NoCommit mode. This is not an exhaustive list but is designed to give guidence.

  - No new storage area can be added, nor can storage maps or indices be modified to use them.

    ```
    SQL> alter database filename personnel
    cont> add storage area EXTRA_SPACE;
    %RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
    ```

  - WORKLOAD COLLECTION cannot be enabled or disabled, although RMU Collect Optimizer_Statistics can be used if a workload was previously created.

    ```
    SQL> alter database filename personnel workload collection is enabled;
    %RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
    ```

  - SYNONYMS may not be enabled. This will prevent the use of the RENAME statement and RENAME AS clauses of ALTER. Enabling synonyms creates a new optional system table to the database.

    ```
    SQL> alter database filename personnel synonyms are enabled;
    %RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
    ```

  - CREATE INDEX, ALTER INDEX, DROP INDEX cannot be performed, even for remedial needs (such as setting MAINTENANCE IS DISABLED)

```
SQL> create index LAST_NAME_NDX on EMPLOYEES (last_name);
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
-RDMS-F-IDXNOTDEF, index LAST_NAME_NDX has not been defined

SQL> alter index EMP_EMPLOYEE_ID maintenance is disabled;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
```

- You cannot alter a table to add a new column, drop a column, or change a data type or length of a column. This would create a new row version and, if rollback occurs, the older Rdb version would not be able to decode that row version.

```
SQL> alter table employees add column bonus float;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
```

- GRANT and REVOKE commands modify the access control lists in the metadata and are prohibited.

```
SQL> grant all privileges on database alias rdb$dbhandle to J_JONES;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
```

Examples

Example 1

The first command in the following example converts an Oracle Rdb release 7.0 database with an extensible .aij file to an Oracle Rdb release 7.3 database. Because the Nocommit qualifier is specified in the first command, you can roll back the converted database (the Oracle Rdb release 7.3 database) to the original Oracle Rdb release 7.0 database.

After-image journaling is disabled while the database is being converted. After the database is converted, a new extensible .aij file is created and after-image journaling is enabled again. Note that .aij files are version-specific. You should perform a full backup operation after a conversion because the old version and the new version of the .aij file are incompatible.

In the second command, the converted database is rolled back to the original database.

```
$RMU/CONVERT/CONFIRM/NOCOMMIT MF_PERSONNEL.RDB
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-00
Are you satisfied with your backup of
 DISK1:[TESTS]MF_PERSONNEL.RDB;1
 and your backup of any associated .aij files [N]? Y
%RMU-I-AIJ_DISABLED, after-image journaling is being disabled
 temporarily for the Convert operation
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-S-CVTDBSUC, database DISK1:[TESTS]MF_PERSONNEL.RDB;1 successfully
 converted from version V7.0 to V7.3
%RMU-I-LOGCREAIJ, created after-image journal file
 DISK1:[TESTS]BACKUP_AFTER1.AIJ;2
%RMU-I-LOGMODSTR,     activated after-image journal "AFTER1"
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery

$RMU/CONVERT/ROLLBACK MF_PERSONNEL.RDB
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-00
Are you satisfied with your backup of
 DISK1:[TESTS]MF_PERSONNEL.RDB;1 and your backup of
 any associated .aij files [N]? Y
%RMU-I-AIJ_DISABLED, after-image journaling is being disabled
 temporarily for the Convert operation
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-I-CVTROLSUC, CONVERT rolled-back for DISK1:[TESTS]MF_PERSONNEL.RDB;1
 to version V7.0
%RMU-I-CANTENAAIJ, The JOURNAL is now DISABLED. RMU CONVERT can not enable
 the JOURNAL for previous versions. You must do this manually.
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

1.16 RMU Convert Command

Example 2

This example is the same as Example 1, except fixed-size .aij journals are being employed at the time of the conversion. The first command in this example converts an Oracle Rdb release 7.1 database with fixed-size .aij files to an Oracle Rdb release 7.3 database. Because the Nocommit qualifier is specified in the first command, you can roll back the converted database (the Oracle Rdb release 7.3 database) to the original Oracle Rdb V7.1 database.

After-image journaling is disabled while the database is being converted. After the database is converted, Oracle RMU activates the next fixed-size .aij file and enables after-image journaling. Note that .aij files are version specific. You should perform a full backup operation after a conversion because the old .aij files are incompatible with the newly converted database.

In the second command, the converted database is rolled back to the original database.

```
$RMU/CONVERT/CONFIRM/NOCOMMIT MF_PERSONNEL.RDB
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-00
Are you satisfied with your backup of DISK1:[TESTS]MF_PERSONNEL.RDB;1
 and your backup of any associated .aij files [N]? Y
%RMU-I-AIJ_DISABLED, after-image journaling is being disabled
 temporarily for the Convert operation
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-S-CVTDBSUC, database DISK1:[TESTS]MF_PERSONNEL.RDB;1 successfully
 converted from version V7.1 to V7.3
%RMU-I-LOGMODSTR,     activated after-image journal "AFTER2"
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery

$RMU/CONVERT/ROLLBACK MF_PERSONNEL.RDB
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-00
Are you satisfied with your backup of
 DISK1:[TESTS]MF_PERSONNEL.RDB;1 and your backup of
 any associated .aij files [N]? Y
%RMU-I-AIJ_DISABLED, after-image journaling is being disabled
 temporarily for the Convert operation
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-I-CVTROLSUC, CONVERT rolled-back for
 DISK1:[TESTS]MF_PERSONNEL.RDB;1 to version V7.1
%RMU-I-CANTENAAIJ, The JOURNAL is now DISABLED. RMU CONVERT can not
 enable the JOURNAL for previous versions. You must do this manually.
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 3

The following command converts all the databases in DISK1:[RICK] and its subdirectories and also the SPECIAL_DB database that is identified by its data dictionary path name. The Noconfirm qualifier is specified, so Oracle RMU does not request user input. The Nocommit qualifier is not specified, so

the default qualifier, Commit, is used by default and the converted databases cannot be rolled back.

```
$ RMU/CONVERT/NOCONFIRM DISK1:[RICK...]*.RDB,CDD$TOP.RICK.SPECIAL_DB -
_$ /PATH
```

Example 4

The following command converts an Oracle Rdb release 7.2 database to release 7.3. In addition, it reserves space in the database root file of the converted database for four .aij files. After-image journaling is not enabled at the time the Convert command is issued.

```
RMU/CONVERT/CONFIRM/RESERVE=(AIJ=4)/COMMIT MF_PERSONNEL
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-20 on OpenVMS Alpha V8.4
Are you satisfied with the backup of DISK1:[TESTS]MF_PERSONNEL.RDB;1 and any
associated after image (.aij) files [N]? Y
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
%RMU-S-CVTDBSUC, database DISK1:[TESTS]MF_PERSONNEL.RDB;1 successfully
converted from version V7.2 to V7.3
%RMU-I-CVTCOMSUC, CONVERT committed for DISK1:[TESTS]MF_PERSONNEL.RDB;1 to
version V7.3
```

Example 5

The following example shows how the contents of a batch file might look if you were to issue the RMU Convert command with the Confirm qualifier from a batch job.

```
$ RMU/CONVERT/COMMIT/CONFIRM USER1:[COLLECT.V71DB]MF_PERSONNEL
Y
```

Example 6 Handling the AIJSUPRESSED warning

The following example shows the RMU-W-AIJSUPPRESSED issued by RMU Convert. A journal is suppressed when it contains data from a prior release of Oracle Rdb. In this example, the /NOCONFIRM qualifier is used to ignore the normal enquiry about being satisfied with database and after image journal backups.

1.16 RMU Convert Command

```
$ rmu/convert/noconfirm sample
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.3-120 on OpenVMS IA64 V8.4
%RMU-I-AIJ_DISABLED, after-image journaling is being disabled temporarily for
the Convert operation
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-S-CVTDBSUC, database USER2:[TESTING]SAMPLE.RDB;1 successfully converted
from version V7.2 to V7.3
%RMU-I-CVTCOMSUC, CONVERT committed for USER2:[TESTING]SAMPLE.RDB;1 to version
V7.3
%RMU-I-LOGMODSTR,     activated after-image journal "SAMPLE_JOURNAL_01"
%RMU-W-AIJSUPPRESSED, After journal "SAMPLE_JOURNAL_02" has been
suppressed and not converted since it contains data from the previous version
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$
```

While in this state, the suppressed journal will not be used (it will be skipped) until it is corrected.

An RMU Set After_Image statement can be used to modify the after image named in the warning to the current Oracle Rdb version. Oracle recommends that the contents of the after image file be backed up prior to the RMU/SET AFTER_IMAGE command since it will modify and truncate the prior contents.

```
$ rmu/set after/alter=(Name=SAMPLE_JOURNAL_02) SAMPLE
%RMU-I-LOGMODSTR,     unsuppressed after-image journal "SAMPLE_JOURNAL_02"
$
```

Additionally, RMU Backup will similarly warn that some of the after image journal files (AIJ) are incompatible with the current version. The subsequent database backup is unaffected by the state of the after image journals.

```
$ rmu/backup sample sample_bck73
%RMU-F-BADAIJVER, after-image journal version is incompatible with
the runtime system
%RMU-W-ROOERRORS,        1 error encountered in root verification
%RMU-I-BCKTXT_00, Backed up root file USER2:[TESTING]SAMPLE.RDB;3
%RMU-I-BCKTXT_02, Starting full backup of storage area (RDB$SYSTEM)
USER2:[TESTING]SAMPLE.RDA;3 at  4-SEP-2017 17:22:08.80
%RMU-I-BCKTXT_12, Completed full backup of storage area (RDB$SYSTEM)
USER2:[TESTING]SAMPLE.RDA;3 at  4-SEP-2017 17:22:08.81
%RMU-I-BCKTXT_02, Starting full backup of storage area (SAMPLE_AREA1)
USER2:[TESTING]SAMPLE_AREA1.RDA;3 at  4-SEP-2017 17:22:08.81
%RMU-I-BCKTXT_12, Completed full backup of storage area (SAMPLE_AREA1)
USER2:[TESTING]SAMPLE_AREA1.RDA;3 at  4-SEP-2017 17:22:08.82
%RMU-I-COMPLETED, BACKUP operation completed at  4-SEP-2017 17:22:08.82
$
```

1.17 RMU Copy_Database Command

Permits you to copy a database.

Format

RMU/Copy_Database root-file-spec [storage-area-list]

Command Qualifiers	Defaults
/[No]After_Journal[=file-spec]	See description
/[No]Aij_Options=journal-opts-file	See description
/Attributes=options	See description
/[No]Cdd_Integrate	Nocdd_Integrate
/[No]Checksum_Verification	/Checksum_Verification
/Close_Wait=n	See description
/Directory=directory-spec	None
/[No]Duplicate	/Noduplicate
/Global_Buffers=global-buffer-options	Current value
/Local_Buffers=local-buffer-options	Current value
/Lock_Timeout[=seconds]	See description
/[No]Log	Current DCL verify value
/Nodes_Max=n	Current value
/[No]Online	/Noonline
/Open_Mode={Automatic|Manual}	Current value
/Option=file-spec	None
/Page_Buffers=n	n=33
/Path=cdd-path	Existing value
/[No]Quiet_Point	/Quiet_Point
/Root=file-spec	None
/Row_Cache_Options=file-spec	None
/Transaction_Mode=(mode-list)	/Transaction_Mode=Current
/Threads=n	/Threads=10
/Users_Max=n	Current value

File or Area Qualifier	Defaults
/Blocks_Per_Page=n	None
/Extension={Disable | Enable }	Current value
/File=file-spec	None
/Read_Only	Current value
/Read_Write	Current value
/Snapshots=(Allocation=n,File=file-spec)	None
/[No]Spams	Current value
/Thresholds=(n,n,n)	None

1.17 RMU Copy_Database Command

Description

The RMU Copy_Database command allows you to modify certain area parameters when the copy operation is performed. All the files are processed simultaneously during the copy operation. The copy operation's performance is similar to that of the RMU Backup command. The RMU Copy_Database command eliminates the need for intermediate storage media.

Note

You must perform a full and complete Oracle RMU backup operation immediately after the Copy_Database operation completes to ensure that the database can be properly restored after a database failure or corruption.

Also note that if you do not specify either the After_Journal qualifier or the Aij_Options qualifier when you issue the RMU Copy_Database command, after-image journaling is disabled for the database copy and no .aij files are associated with the database copy.

Command Parameters

root-file-spec
The name of the database root file for the database you want to copy.

storage-area-list
The name of one or more storage areas whose parameters you are changing. The storage-area-list parameter is optional. Unless you are using the RMU Copy_Database command to modify the parameters of one or more storage areas, you should not specify any storage area names.

Command Qualifiers

After_Journal[=file-spec]
Noafter_Journal

Note

This qualifier is maintained for compatibility with versions of Oracle Rdb prior to Version 6.0. You might find it more useful to specify the Aij_Options qualifier, unless you are interested in creating an extensible .aij file only.

Specifies how Oracle RMU is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the After_Journal qualifier and provide a file specification, Oracle RMU enables journaling and creates a new extensible after-image journal (.aij) file for the database copy.
- If you specify the After_Journal qualifier but you do not provide a file specification, Oracle RMU enables after-image journaling and creates a new extensible .aij file for the database copy with the same name as, but a different version number from, the .aij file for the database being copied.
- If you specify the Noafter_Journal qualifier, Oracle RMU disables after-image journaling and does not create a new .aij file.
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, Oracle RMU disables after-image journaling and does not create a new .aij file.

You can specify only one, or none, of the following after-image journal qualifiers in a single RMU Copy_Database command: After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options.

You cannot use the After_Journal qualifier to create fixed-size .aij files; use the Aij_Options qualifier.

Aij_Options=journal-opts-file
Noaij_Options

Specifies how Oracle RMU is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the Aij_Options qualifier and provide a journal-opts-file, Oracle RMU enables journaling and creates the .aij file or files you specify for the database copy. If only one .aij file is created for the database copy, it will be an extensible .aij file. If two or more .aij files are created for the database copy, they will be fixed-size .aij files (as long as at least two .aij files are always available).
- If you specify the Aij_Options qualifier, but do not provide a journal-opts-file, Oracle RMU disables journaling and does not create any new .aij files.
- If you specify the Noaij_Options qualifier, Oracle RMU disables journaling and does not create any new .aij files.

- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, Oracle RMU disables after-image journaling and does not create a new .aij file.

You can only specify one, or none, of the following after-image journal qualifiers in a single Oracle RMU command: After_Journal, Noafter_Journal, Aij_Options, Noaij_Options.

See Section 1.67.1 for information on the format of a journal-opts-file.

Attributes=options
Used to specify a number of qualifiers and use an abbreviation for the qualifier name. Normally, /THRESHOLDS, /READ_ONLY and /READ_WRITE cannot be abbreviated or an error will be returned because the abbreviation is not unique for the first four characters (an OpenVMS DCL requirement). By using the /ATTRIBUTES qualifier, THRESHOLDS can be abbreviated to the first four characters and either the option READ_ONLY or the option READ_WRITE can be abbreviated to the first six characters. Examples of the syntax follow.

```
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_ONLY)
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_WRITE)
/ATTRIBUTES=THRESHOLDS=(50,50,50)
/ATTRIBUTES=READ_ONLY
/ATTRIBUTES=READ_WRITE
```

Examples of using abbreviation with the /ATTRIBUTES qualifier options follow.

```
/ATTR=(THRE=(50,50,50),READ_O)
/ATTR=(THRE=(50,50,50),READ_W)
/ATTR=THRE=(50,50,50)
/ATTR=READ_O
/ATTR=READ_W
```

Cdd_Integrate
Nocdd_Integrate
Integrates the metadata from the root (.rdb) file of the database copy into the data dictionary (assuming the data dictionary is installed on your system).

If you specify the Nocdd_Integrate qualifier, no integration occurs during the copy operation.

You might want to delay integration of the database metadata with the data dictionary until after the copy operation finishes successfully.

You can use the Nocdd_Integrate qualifier even if the DICTIONARY IS REQUIRED clause was used when the database being copied was defined.

The Cdd_Integrate qualifier integrates definitions *in one direction only*—from the database file to the dictionary. The Cdd_Integrate qualifier does *not* integrate definitions from the dictionary to the database file.

The Nocdd_Integrate qualifier is the default.

Checksum_Verification
Nochecksum_Verification
Requests that the page checksum be verified for each page copied. The default is to perform this verification.

The Checksum_Verification qualifier uses significant CPU resources but can provide an extra measure of confidence in the quality of the data being copied. For offline copy operations, the additional CPU cost of using the Checksum_ Verification qualifier might not be justified unless you are experiencing or have experienced disk, HSC, or CI port hardware problems. One symptom of these problems is pages being logged to the corrupt page table (CPT).

For online copy operations, use of the Checksum_Verification qualifier offers an additional level of data security when the database employs disk striping or RAID (redundant arrays of inexpensive disks) technology. These technologies fragment data over several disk drives, and use of the Checksum_Verification qualifier permits Oracle RMU to detect the possibility that the data it is reading from these disks has been only partially updated. If you use either of these technologies, you should use the Checksum_Verification qualifier.

Note, however, that if you specify the Nochecksum qualifier, and undetected corruptions exist in your database, the corruptions are included in the copied file. Such a corruption might be difficult to recover from, especially if it is not detected until weeks or months after the copy operation is performed.

Overall, Oracle Corporation recommends that you use the Checksum_ Verification qualifier with all copy operations where integrity of the data is essential.

Close_Wait=n
Specifies a wait time of *n* minutes before Oracle RMU automatically closes the database. You must supply a value for *n*.

In order to use this qualifier, the Open_Mode qualifier on the RMU Copy_Database command line must be set to Automatic.

Directory=directory-spec
Specifies the default destination for the copied database files. Note that if you specify a file name or file extension, all copied files are given that file name or file extension. There is no default directory specification for this qualifier.

See the Usage Notes for information on how this qualifier interacts with the Root, File, and Snapshot qualifiers and for warnings regarding copying database files into a directory owned by a resource identifier.

If you do not specify this qualifier, Oracle RMU attempts to copy all the database files (unless they are qualified with the Root, File, or Snapshot qualifier) to their current location.

Duplicate
Noduplicate

Causes the RMU Copy_Database command to generate a new database with the same content, but with a different identity from that of the original database. For this reason, .aij files cannot be interchanged between the original and the duplicate database. This qualifier creates copies of your databases that are expected to evolve independently in time. In this case, being able to exchange .aij files might be a security breach, and a likely source of corruption.

A duplicate database has the same contents as the original database, but not the same identity. A database copied with the Noduplicate qualifier is an exact replica of the original database in every way and, therefore, .aij files can be interchanged between the original and duplicate database.

The default is the Noduplicate qualifier.

Global_Buffers=global-buffer-options

Allows you to change the default global buffer parameters when you copy a database. The following options are available:

- Disabled

  Use this option to disable global buffering for the copy of the original database.

- Enabled

  Use this option to enable global buffering for the copy of the original database. You cannot specify both the Disabled and Enabled option in the same RMU Copy_Database command with the Global_Buffers qualifier.

- Total=total-buffers

  Use this option to specify the number of buffers available for all users.

- User_Limit=buffers-per-user

  Use this option to specify the maximum number of buffers available to each user.

If you do not specify a global buffers option, the database is copied with the values that are in effect for the database you are copying.

When you specify two or more options with the Global_Buffers qualifier, use a comma to separate each option and enclose the list of options in parentheses.

Local_Buffers=local-buffer-options
Allows you to change the default local buffer parameters when you copy a database. The following options are available:

- Number=number-buffers

  Use this option to specify the number of local buffers that will be available for all users. You must specify a number between 2 and 32,767 for the number-buffers parameter.

- Size=buffer-blocks

  Use this option to specify the size (specified in blocks) for each buffer. You must specify a number between 2 and 64 for the buffer-blocks parameter.

  If you specify a value smaller than the size of the largest page defined, Oracle RMU automatically adjusts the size of the buffer to hold the largest page defined. For example, if you specify the Local_Buffers=Size=8 qualifier and the largest page size for the storage areas in your database is 64 blocks, Oracle RMU automatically interprets the Local_Buffers=Size=8 qualifier as though it were a Local_Buffers=Size=64 qualifier.

  Take great care when selecting a buffer size; a poor choice causes performance to suffer greatly.

  The value specified for the buffer-blocks parameter determines the number of blocks for each buffer, regardless of whether local buffering or global buffering is enabled for the database.

If you do not specify a Local_Buffers option, the database is copied with the values that are in effect for the database you are copying.

Lock_Timeout[=seconds]
Specifies a timeout interval or maximum time in seconds to wait for the quiet-point lock and any other locks needed when the operation is performed online.

The /LOCK_TIMEOUT qualifier can also be specified without a value. In this case, the default lock timeout value specified for the database will be used. Specifically, the default lock timeout value used will be the value of the logical name RDM$BIND_LOCK_TIMEOUT_INTERVAL if it has been specified, or, if RDM$BIND_LOCK_TIMEOUT_INTERVAL has not been specified, the "LOCK TIMEOUT INTERVAL" specified by the SQL CREATE DATABASE or ALTER

DATABASE command will be used. If neither value has been specified, the value used will be the maximum possible lock timeout value which can be specified for an Oracle Rdb database.

Note that /LOCK_TIMEOUT is not the default and must be specified. The default if /LOCK_TIMEOUT is not specified is to wait indefinitely to acquire the QUIET POINT or other database locks needed during an online copy operation.

The Lock_Timeout=seconds qualifier is ignored for offline copy operations.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Nodes_Max=n
Specifies a new value for the database maximum node count parameter for the database copy. The default is to leave the value unchanged.

Online
Noonline
Specifies that the copy database operation be performed while other users are attached to the database. The areas to be copied are locked for read-only access, so the operation is compatible with all but exclusive access.

The default is the Noonline qualifier.

Open_Mode=Automatic
Open_Mode=Manual
Allows you to change the mode for opening a database when you copy a database. When you specify the Open_Mode=Automatic qualifier, users can invoke the database copy immediately after it is copied. If you specify the Open_Mode=Manual qualifier, an RMU Open command must be used to open the database before users can invoke the database copy.

The Open_Mode qualifier also specifies the mode for closing a database. If you specify Open_Mode=Automatic, you can also use the Close_Wait qualifier to specify a time in minutes before the database is automatically closed.

If you do not specify the Open_Mode qualifier, the database is copied with the open mode that is in effect for the database being copied.

Option=file-spec
Specifies an options file containing storage area names, followed by the storage area qualifiers that you want applied to that storage area. Do *not* separate the storage area names with commas. Instead, put each storage area name on a separate line in the file. The storage area qualifiers that you can include in the options file are: Blocks_Per_Page, File, Snapshots, and Thresholds.

You can use the DCL line continuation character, a hyphen (-), or the comment character (!) in the options file. There is no default for this qualifier. Example 6 in the Examples section shows an options file and how to specify it on the Oracle RMU command line.

If the Option qualifier is specified, the storage-area-list parameter is ignored.

Page_Buffers=n
Specifies the number of buffers to be allocated for each database file to be copied. The number of buffers used is twice the number specified; half are used for reading the file and half for writing the copy. Values specified with the Page_Buffers qualifier can range from 1 to 5. The default value is 3. Larger values might improve performance, but they increase memory use.

Path=cdd-path
Specifies a data dictionary path into which the definitions of the database copy will be integrated. If you do not specify the Path qualifier, Oracle RMU uses the CDD$DEFAULT logical name value of the user who enters the RMU Copy_Database command.

If you specify a relative path name, Oracle Rdb appends the *relative* path name you enter to the CDD$DEFAULT value. If the cdd-path parameter contains nonalphanumeric characters, you must enclose it within quotation marks ("").

Oracle Rdb ignores the Path qualifier if you use the Nocdd_Integrate qualifier or if the data dictionary is not installed on your system.

Quiet_Point
Noquiet_Point
Allows you to specify that a database copy operation is to occur either immediately or when a quiet point for database activity occurs. A quiet point is defined as a point where no active update transactions are in progress in the database.

When you specify the Noquiet_Point qualifier, Oracle RMU proceeds with the copy operation as soon as the RMU Copy_Database command is issued, regardless of any update transaction activity in progress in the database. Because Oracle RMU must acquire concurrent-read locks on all physical and logical areas, the copy operation fails if there are any active transactions

with exclusive locks on a storage area. However, once Oracle RMU has successfully acquired all concurrent-read storage area locks, it should not encounter any further lock conflicts. If a transaction that causes Oracle Rdb to request exclusive locks is started while the copy operation is proceeding, that transaction either waits or gets a lock conflict error, but the copy operation continues unaffected.

If you intend to use the Noquiet_Point qualifier with a copy procedure that previously specified the Quiet_Point qualifier (or did not specify either the Quiet_Point or Noquiet_Point qualifier), you should examine any applications that execute concurrently with the copy operation. You might need to modify your applications or your copy procedure to handle the lock conflicts that can occur when you specify the Noquiet_Point qualifier.

When you specify the Quiet_Point qualifier, the copy operation begins when a quiet point is reached. Other update transactions issued after the database copy operation begins are prevented from executing until after the root file for the database has been copied (copying of the database storage areas begins after the root file is copied).

The default is the Quiet_Point qualifier.

Root=file-spec

Requests that the database root file be copied to the specified location.

See the Usage Notes for information on how this qualifier interacts with the Directory, File, and Snapshot qualifiers.

Row_Cache_Options=file-spec

Specifies a row cache backing store options file which contains per database and/or per cache row cache backing store directory specifications.

The file-spec must be a valid file specification of an existing options file. This qualifier cannot be negated. The following syntax must be used in the row cache backing store options file.

```
$ TYPE FILENAME.OPT
/BACKING_STORE = device:[directory]
row_cache_name /BACKING_STORE = device:[directory]
row_cache_name /NOBACKING_STORE
```

To modify or remove the current per database default backing store location, either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified on the first line without being preceded by a row cache name.

To modify or remove a current per cache backing store location, an existing row cache name currently defined in the database root file must be specified followed by either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified. The wild card characters "%" and/or "*" can be specified as part of the row cache name, where "*" can be used in the place of one or more contiguous characters and "%" can be used in the place of a single character. In this case all matching per cache backing store entries will be modified with the following /BACKING_STORE = or /NOBACKING_STORE specification. The RMU/DUMP/HEADER command can be used to display the current per database default row cache backing store directory as well as the current per cache row cache backing store directories.

The /ROW_CACHE_OPTIONS qualifier cannot be used with the /ONLINE, /QUIET_POINT or /DUPLICATE qualifiers.

Transaction_Mode=(mode-list)
Sets the allowable transaction modes for the database root file created by the copy operation. The mode-list can include one or more of the following transaction modes:

- All - Enables all transaction modes
- Current - Enables all transaction modes that are set for the source database. This is the default transaction mode.
- None - Disables all transaction modes
- [No]Batch_Update
- [No]Exclusive
- [No]Exclusive_Read
- [No]Exclusive_Write
- [No]Protected
- [No]Protected_Read
- [No]Protected_Write
- [No]Read_Only
- [No]Read_Write
- [No]Shared
- [No]Shared_Read
- [No]Shared_Write

Your copy operation must include the database root file. Otherwise, RMU returns the CONFLSWIT error when you issue an RMU Copy_Database command with the Transaction_Mode qualifier.

If you specify more than one transaction mode in the mode-list, enclose the list in parenthesis and separate the transaction modes from one another with a comma. Note the following:

- When you specify a negated transaction mode such as Noexclusive_Write, it indicates that exclusive write is not an allowable access mode for the copied database.
- If you specify the Shared, Exclusive, or Protected transaction mode, Oracle RMU assumes you are referring to both reading and writing in that transaction mode.
- No mode is enabled unless you add that mode to the list, or you use the All option to enable all transaction modes.
- You can list one transaction mode that enables or disables a particular mode followed by another that does the opposite. For example, Transaction_Mode=(Noshared_Write, Shared) is ambiguous because the first value disables Shared_Write access and the second value enables Shared_Write access. Oracle RMU resolves the ambiguity by first enabling the modes as specified in the modes-list and then disabling the modes as specified in the modes-list. The order of items in the list is irrelevant. In the example presented previously, Shared_Read is enabled and Shared_ Write is disabled.

Threads=number

Specifies the number of reader threads to be used by the copy process.

RMU creates so called internal 'threads' of execution to read data from one specific storage area. Threads run quasi-parallel within the process executing the RMU image. Each thread generates its own I/O load and consumes resources like virtual address space and process quotas (e.g. FILLM, BYTLM). The more threads, the more I/Os can be generated at one point in time and the more resources are needed to accomplish the same task.

Performance increases with more threads due to parallel activities which keeps disk drives busier. However, at a certain number of threads, performance suffers because the disk I/O subsystem is saturated and I/O queues build up for the disk drives. Also the extra CPU time for additional thread scheduling overhead reduces the overall performance. Typically 2-5 threads per input disk drive are sufficient to drive the disk I/O susbsystem at its optimum. However, some controllers may be able to handle the I/O load of more threads, for example disk controllers with RAID sets and extra cache memory.

In a copy operation, one thread moves the data of one storage area at a time. If there are more storage areas to be copied than there are threads, then the next idle thread takes on the next storage area. Storage areas are copied in order of the area size - largest areas first. This optimizes the overall elapsed time by allowing other threads to copy smaller areas while an earlier thread is still working on a large area. If no threads qualifier is specified, then 10 threads are created by default. The minimum is 1 thread and the maximum is the number of storage areas to be copied. If the user specifies a value larger than the number of storage areas, then RMU silently limits the number of threads to the number of storage areas.

For a copy operation, you can specify a threads number as low as 1. Using a threads number of 1 generates the smallest system load in terms of working set usage and disk I/O load. Disk I/O subsystems most likely can handle higher I/O loads. Using a slightly larger value than 1 typically results in faster execution time.

Users_Max=n
Specifies a new value for the database maximum user count parameter.

The default is to use the same value as is in effect for the database being copied.

File or Area Qualifiers

Blocks_Per_Page=n
Specifies a new page size for the storage area to which it is applied. You cannot decrease the page size of a storage area.

You might want to increase the page size in storage areas containing hash indexes that are close to full. By increasing the page size in such a situation, you prevent the storage area from extending and still preserve the existing target pages for the hash index.

The Blocks_Per_Page qualifier is a positional qualifier.

Extension=Disable
Extension=Enable
Allows you to change the automatic file extension attribute for a storage area when you copy a database.

Use the Extension=Disable qualifier to disable automatic file extensions for a storage area.

Use the Extension=Enable qualifier to enable automatic file extensions for a storage area.

If you do not specify the Extension=Disable or the Extension=Enable qualifier, the storage areas are copied with the automatic file extension attributes that are in effect for the database being copied.

The Extension qualifier is a positional qualifier.

File=file-spec
Requests that the storage area to which this qualifier is applied be copied to the specified location.

See the Usage Notes for information on how this qualifier interacts with the Root, Snapshot, and Directory qualifiers and for warnings regarding copying database files into a directory owned by a resource identifier.

The File qualifier is a positional qualifier. This qualifier is not valid for single-file databases.

Read_Only
Use the Read_Only qualifier to change a read/write storage area or a write-once storage area to a read-only storage area.

If you do not specify the Read_Only or Read_Write qualifier, the storage areas are copied with the read/write attributes that are currently in effect for the database being copied.

This is a positional qualifier.

Read_Write
Use the Read_Write qualifier to change a read-only storage area or a write-once storage area to a read/write storage area.

If you do not specify the Read_Only or Read_Write qualifier, the storage areas are copied with the read/write attributes that are currently in effect for the database being copied.

This is a positional qualifier.

Snapshots=(Allocation=n,File=file-spec)
If you specify the Allocation parameter, specifies the snapshot file allocation size in *n* pages for a copied area. If you specify the File parameter, specifies a new snapshot file location for the copied storage area to which it is applied.

You can specify the Allocation parameter only, the File parameter only, or both parameters; however, if you specify the Snapshots qualifier, you must specify at least one parameter.

The Snapshots qualifier is a positional qualifier.

See the Usage Notes for information on how this qualifier interacts with the Root, File, and Directory qualifiers.

Spams
Nospams
Specifies whether to enable the creation of space area management (SPAM) pages or disable the creation of SPAM pages (Nospams) for specified storage areas. This qualifier is not permitted with a storage area that has a uniform page format.

When SPAM pages are disabled in a read/write storage area, the SPAM pages are initialized but they are not updated.

The Spams qualifier is a positional qualifier.

Thresholds=(n,n,n)
Specifies new SPAM thresholds for the storage area to which it is applied (for a mixed page format storage area). The thresholds of a storage area with a uniform page format cannot be changed.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for information on setting SPAM thresholds.

The Thresholds qualifier is a positional qualifier.

Usage Notes

- To use the RMU Copy_Database command for a database, you must have the RMU$COPY privilege in the root file access control list (ACL) for the database to be copied or the OpenVMS SYSPRV or BYPASS privilege.
- When you copy a database into a directory owned by a resource identifier, the ACE for the directory is applied to the database root file ACL first, and then the Oracle RMU ACE is added. This method is employed to prevent database users from overriding OpenVMS file security. However, this can result in a database which you consider yours, but to which you have no Oracle RMU privileges to access. See the *Oracle Rdb Guide to Database Maintenance* for details.
- The RMU Copy_Database command provides four qualifiers, Directory, Root, File, and Snapshots, that allow you to specify the target for the copied files. The **target** can be just a directory, just a file name, or a directory and file name.

If you use all or some of these four qualifiers, apply them as follows:

- Use the Root qualifier to indicate the target for the copy of database root file.
- Use local application of the File qualifier to specify the target for the copy of one or more storage areas.
- Use local application of the Snapshots qualifier to specify the target for the copy of one or more snapshot files.
- Use the Directory qualifier to specify a default target directory. The default target directory is the directory to which all files not qualified with the Root, File, or Snapshot qualifier are copied. It is also the default directory for files qualified with the Root, File, or Snapshot qualifier if the target for these qualifiers does not include a directory specification.

Note the following when using these qualifiers:

- Global application of the File qualifier when the target specification includes a file name causes Oracle RMU to copy all of the storage areas to different versions of the same file name. This creates a database that is difficult to manage.
- Global application of the Snapshot qualifier when the target specification includes a file name causes Oracle RMU to copy all of the snapshot files to different versions of the same file name. This creates a database that is difficult to manage.
- Specifying a file name or extension with the Directory qualifier is permitted, but causes Oracle RMU to copy all of the files (except those specified with the File or Root qualifier) to different versions of the same file name. Again, this creates a database that is difficult to manage.

See Example 8.

- You cannot use the RMU Copy_Database command to copy a database to a remote system or to an NFS (Network File System) mounted file system. The RMU Copy_Database command allows you to create a copy of a database on the same node as the original database.
- You cannot disable extents of snapshot (.snp) files.
- The file and area qualifiers for the RMU Copy_Database command are positional qualifiers, and if placed randomly, could be ignored or produce unexpected results. See Section 1.2 for more information on positional qualifiers.

- There are no restrictions on the use of the Nospams qualifier with mixed page format storage areas, but the use of the Nospams qualifier typically causes severe performance degradation. The Nospams qualifier is only useful where updates are rare and batched, and access is primarily by database key (dbkey).

Examples

Example 1

The following command makes a duplicate copy of the mf_personnel database in the DISK1:[USER1] directory:

```
$ RMU/COPY_DATABASE MF_PERSONNEL /DIRECTORY=DISK1:[USER1]
```

Example 2

The following example shows a simple duplication of a database within a user's directory. In this instance, the duplicated database has the same content and identity as the original database. After-image journal files can be interchanged between the original and the duplicated database. Execute the RMU Dump command with the header qualifier to verify that the copied database is the same as the original database. Note that the creation date listed in the header for each database is the same.

```
$ RMU/COPY_DATABASE MF_PERSONNEL
```

Example 3

The following example shows a duplication of a database within a user's directory through the use of the Duplicate qualifier. In this instance, the duplicated database differs from the original database. It has the same content as the original database, but its identity is different. As a result, .aij files cannot be exchanged between the original database and the duplicate database. If you use the RMU Dump command with the header qualifier for each database, you see that the creation date for the copy and the original database is different.

```
$ RMU/COPY_DATABASE/DUPLICATE MF_PERSONNEL
```

Example 4

The following command copies the mf_personnel database from the DISK2:[USER2] directory to the DISK1:[USER1] directory. The Extension=Disable qualifier causes extents to be disabled for all the storage area (.rda) files in the DISK1:[USER1]mf_personnel database:

1.17 RMU Copy_Database Command

```
$ RMU/COPY_DATABASE/EXTENSION=DISABLE/DIRECTORY=DISK1:[USER1] -
_$ DISK2:[USER2]MF_PERSONNEL
```

Example 5

The following command copies the mf_personnel database from the DISK2:[USER2] directory to the DISK2:[USER1] directory. Because the Extension=Disable qualifier is specified for only the EMPIDS_LOW and EMPIDS_MID storage areas, extents are disabled only for those two storage area (.rda) files in the DISK2:[USER1]mf_personnel database:

```
$ RMU/COPY_DATABASE/DIRECTORY=DISK2:[USER1] DISK2:[USER2]MF_PERSONNEL -
_$ EMPIDS_LOW/EXTENSION=DISABLE,EMPIDS_MID/EXTENSION=DISABLE
```

Example 6

The following command uses an options file to specify that the storage area files and snapshot (.snp) files be copied to different disks. Note that storage area .snp files are located on different disks from one another and from their associated storage area (.rda) files; this is recommended for optimal performance. (This example assumes that the disks specified for each storage area file in options_file.opt are different from those where the storage area files currently reside.)

```
$ RMU/COPY_DATABASE/OPTIONS=OPTIONS_FILE.OPT MF_PERSONNEL
```

The options file appears as:

```
$ TYPE OPTIONS_FILE.OPT

EMPIDS_LOW /FILE=DISK1:[CORPORATE.PERSONNEL]EMPIDS_LOW.RDA -
     /SNAPSHOT=(FILE=DISK2:[CORPORATE.PERSONNEL]EMPIDS_LOW.SNP)

EMPIDS_MID /FILE=DISK3:[CORPORATE.PERSONNEL]EMPIDS_MID.RDA -
     /SNAPSHOT=(FILE=DISK4:[CORPORATE.PERSONNEL]EMPIDS_MID.SNP)

EMPIDS_OVER /FILE=DISK5:[CORPORATE.PERSONNEL]EMPIDS_OVER.RDA -
     /SNAPSHOT=(FILE=DISK6:[CORPORATE.PERSONNEL]EMPIDS_OVER.SNP)

DEPARTMENTS /FILE=DISK7:[CORPORATE.PERSONNEL]DEPARTMENTS.RDA -
     /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]DEPARTMENTS.SNP)

SALARY_HISTORY /FILE=DISK9:[CORPORATE.PERSONNEL]SALARY_HISTORY.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]SALARY_HISTORY.SNP)

JOBS /FILE=DISK7:[CORPORATE.PERSONNEL]JOBS.RDA -
     /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]JOBS.SNP)

EMP_INFO /FILE=DISK9:[CORPORATE.PERSONNEL]EMP_INFO.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]EMP_INFO.SNP)

RESUME_LISTS /FILE=DISK11:[CORPORATE.PERSONNEL]RESUME_LISTS.RDA -
     /SNAPSHOT=(FILE=DISK12:[CORPORATE.PERSONNEL]RESUME_LISTS.SNP)
```

```
RESUMES /FILE=DISK9:[CORPORATE.PERSONNEL]RESUMES.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]RESUMES.SNP)
```

1.17 RMU Copy_Database Command

Example 7

The following example copies the mf_personnel database from one directory to another. In addition, by specifying the Aij_Options qualifier to add after-image journal files, it enables fixed-size journaling in the database copy and sets several of the journal options as shown in the aij_journal_options.opt file.

```
$ RMU/COPY_DATABASE MF_PERSONNEL/DIRECTORY=DB1:[ROOT] -
/AIJ_OPTIONS=AIJ_JOURNAL_OPTIONS.OPT

$ TYPE AIJ_JOURNAL_OPTIONS.OPT

JOURNAL IS ENABLED      -
RESERVE 2               -
ALLOCATION IS 1024      -
BACKUPS ARE MANUAL      -
OVERWRITE IS DISABLED   -
SHUTDOWN_TIMEOUT IS 120 -
CACHE IS DISABLED

ADD MF_PERS1 FILE DB2:[AIJONE]MF_PERS1.AIJ
ADD MF_PERS2 FILE DB3:[AIJTWO]MF_PERS2.AIJ
```

Example 8

The following example demonstrates the use of the Directory, File, and Root qualifiers. In this example:

- The default directory is specified as DISK2:[DIR].
- The target directory and file name for the database root file is specified with the Root qualifier. The target directory specified with the Root qualifier overrides the default directory specified with the Directory qualifier. Thus, Oracle RMU copies the database root to DISK3:[ROOT] and names it COPYRDB.RDB.
- The target directory for the EMPIDS_MID storage area is DISK4:[FILE]. Oracle RMU copies EMPIDS_MID to DISK4:[FILE].
- The target file name for the EMPIDS_LOW storage area is EMPIDS. Thus, Oracle RMU copies the EMPIDS_LOW storage area to the DISK2 default directory (specified with the Directory qualifier), and names the file EMPIDS.RDA.
- The target for the EMPIDS_LOW snapshot file is DISK5:[SNAP]EMPIDS.SNP Thus, Oracle RMU copies the EMPIDS_LOW snapshot file to DISK5:[SNAP]EMPIDS.SNP.

- All the other storage area files and snapshot files in the mf_personnel database are copied to DISK2:[DIR]; the file names for these storage areas remain unchanged.

```
$ RMU/COPY_DATABASE DISK1:[DB]MF_PERSONNEL.RDB -
_$ /DIRECTORY=DISK2:[DIR] -
_$ /ROOT=DISK3:[ROOT]COPYRDB.RDB -
_$ EMPIDS_MID/FILE=DISK4:[FILE], -
_$ EMPIDS_LOW/FILE=EMPIDS -
_$ /SNAPSHOT=(FILE=DISK5:[SNAP]EMPIDS.SNP)
```

Example 9

The following example demonstrates how to disallow exclusive mode for a copied database. It then shows the error messages returned when a user attempts to access the copied database using the disallowed mode:

```
$ RMU/COPY_DATABASE/TRANSACTION_MODE=NOEXCLUSIVE/DIRECTORY=[.COPY] -
_$ MF_PERSONNEL.RDB
%RMU-W-DOFULLBCK, full database backup should be done to ensure future
recovery
$ SQL
SQL> ATTACH 'FILENAME mf_personnel.rdb';
SQL> SET TRANSACTION READ WRITE RESERVING EMPLOYEES FOR EXCLUSIVE WRITE;
%RDB-E-BAD_TPB_CONTENT, invalid transaction parameters in the
 transaction parameter block (TPB)
-RDMS-E-INVTRANOPT, the transaction option "EXCLUSIVE WRITE" is not
 allowed
SQL>
```

Example 10

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ROW_CACHE_OPTIONS qualifier to read backing store directory options files to modify or remove the current per database and per cache Row Cache backing store directories in the database root file. The contents of the options file read is displayed when the command is executed and the RMU/DUMP/HEADER command is used to check the modified backing store directories in the database root file.

```
$ SET VERIFY
$ RMU/RESTORE/NOCDD/NOLOG/DIR=TEST$DIRECTORY-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:BSTORE.OPT -
 TEST$DIRECTORY:RSA.RBF
%RMU-I-RESTXT_18, Processing options file BSTORE.OPT
  /BACKING_STORE=DISK:[DIRECTORY]
  SAL_CACHE /BACKING_STORE=DISK:[DIRECTORY]
  JOB_CACHE/BACKING_STORE=DISK:[DIRECTORY]
  DROP_CACHE /BACKING_STORE=DISK:[DIRECTORY]
```

1.17 RMU Copy_Database Command

```
$ RMU/DUMP/HEADER/OUT=HDR.LIS TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/MOVE_AREA/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROOT=DISK:[DIRECTORY]FILENAME.EXT
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:WBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file WBSTORE.OPT
  *CACHE /BACKING_STORE=DISK:[DIRECTORY]

$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/COPY_DATABASE/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:NOBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file NOBSTORE.OPT
  /NOBACKING_STORE
  SAL_CACHE /NOBACKING_STORE
  JOB_CACHE/NOBACKING_STORE
  DROP_CACHE /NOBACKING_STORE

$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is database directory
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
```

Example 11

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ATTRIBUTES qualifier.

```
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTRIBUTES=(THRE=(50,50,50),READ_O)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTRIBUTES=READ_O
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTR=READ_W
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/RESTORE/NOLOG/NOCDD/DIR=DISK:[DIRECTORY] DISK:[DIRECTORY]MFP.RBF
JOBS/ATTR=READ_O
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not available.
```

1.18 RMU Delete Optimizer_Statistics Command

Deletes records from the RDB$WORKLOAD system table.

Format

RMU/Delete Optimizer_Statistics root-file-spec

Command Qualifiers	Defaults
/Column_Group=(column-list)	See description
/[No]Log[=file-name]	See description
/Tables=(table-list)	None - Required Qualifier

Description

When you enable and collect workload statistics, the system table, RDB$WORKLOAD, is created and populated. (See Section 1.15 for details.) If you are knowledgeable about the data in your database, or if workload statistics were gathered for queries that are no longer in use, you might decide that you no longer want Oracle RMU to collect statistics for particular column groups. The RMU Delete Optimizer_Statistics gives you the ability to selectively delete records for column groups in the RDB$WORKLOAD system table.

When you use the RMU Delete Optimizer_Statistics command, both the optimizer statistics themselves and the reference to the column duplicity factor and the null factor are deleted from the RDB$WORKLOAD system table.

If you issue an RMU Collect Optimizer_Statistics command after having issued an RMU Delete Optimizer_Statistics command, statistics for the specified column group are not collected.

Command Parameters

root-file-spec
Specifies the database from which optimizer statistics are to be deleted. The default file type is .rdb.

Command Qualifiers

Column_Group=(column-list)
Specifies a list of columns that comprise a single column group. The columns specified must be a valid column group for a table specified with the Tables=(table-list) qualifier. (Use the RMU Show Optimizer_Statistics command to display a valid column groups.) When you specify the Column_ Group qualifier, the entire record in the RDB$WORKLOAD system table that holds data for the specified column group is deleted. Therefore, the next time you issue the RMU Collect Optimizer_Statistics command, statistics for the specified column-group are not collected.

Log
Nolog
Log=file-name
Specifies whether the statistics deleted from the RDB$WORKLOAD system table are to be logged. Specify the Log qualifier to have the information displayed to SYS$OUTPUT. Specify the Log=file-spec qualifier to have the information written to a file. Specify the Nolog qualifier to prevent display of the information. If you do not specify any variation of the Log qualifier, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Tables=(table-list)
Specifies the table or tables for which column group entries are to be deleted, as follows:

- If you specify the Tables=(table-list) qualifier, but do not specify the Column_Group qualifier, then all column group entries for the listed tables are deleted from the RDB$WORKLOAD system table.
- If you specify the Tables=(table-list) qualifier, and you specify the Column_ Group=(column-list) qualifier, then the workload statistics entries for the specified tables that have exactly the specified column group are deleted from the RDB$WORKLOAD system table.
- If you use an asterisk (*) with the Tables qualifier (Tables=*), all tables registered in the RDB$WORKLOAD table are deleted. This allows the RDB$WORKLOAD table to be purged.

If you issue an RMU Collect Optimizer_Statistics command after you have deleted a workload column group from the RDB$WORKLOAD system table, those statistics are no longer collected.

The Tables=(table-list) qualifier is a required qualifier; you cannot issue an RMU Delete Optimizer_Statistics command without the Tables=(table-list) qualifier.

Usage Notes

- To use the RMU Delete Optimizer_Statistics command for a database, you must have the RMU$ANALYZE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

  You must also have the SQL DBADM privilege on the database for which the delete is being performed.

- When you use the SQL ALTER DATABASE statement to set the RDB$SYSTEM storage area (or DEFAULT storage area) to read-only access for your database, the Oracle Rdb system tables are also set to read-only access.

  When you run RMU Delete Optimizer_Statistics on such databases, the RDB$SYSTEM, and if necessary, the DEFAULT storage area will be changed to READ WRITE to allow updates to the system tables and then restored to READ ONLY.

- Cardinality statistics are automatically maintained by Oracle Rdb. Physical storage and workload statistics are only collected when you issue an RMU Collect Optimizer_Statistics command. To get information about the usage of physical storage and workload statistics for a given query, define the RDMS$SET_FLAGS logical name to be "ESTIMATES". For example:

  ```
  $ DEFINE RDMS$SET_FLAGS "ESTIMATES"
  ```

  When you execute a query, if workload and physical statistics have been used in optimizing the query, you will see a line such as the following in the command output:

  ```
  ~O: Workload and Physical statistics used
  ```

- Oracle Corporation recommends that you execute an RMU Show Optimizer_Statistics command with the Output qualifier prior to executing an RMU Delete Optimizer_Statistics command. If you accidentally delete statistics, you can replace them by issuing an RMU Insert Optimizer_ Statistics command and specifying the statistical values contained in the output file.

Examples

Example 1

The following example issues commands to do the following:

1. Display optimizer statistics for the EMPLOYEES and JOB_HISTORY tables and their indexes
2. Delete the entries for the column group (EMPLOYEE_ID, JOB_CODE, JOB_START, JOB_END, DEPARTMENT_CODE, SUPERVISOR_ID) in JOB_HISTORY

```
$ RMU/SHOW OPTIMIZER_STATISTICS MF_PERSONNEL.RDB -
_$ /TABLES=(EMPLOYEES, JOB_HISTORY)/STATISTICS=(WORKLOAD)
------------------------------------------------------------------------

Optimizer Statistics for table : EMPLOYEES

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 1.0000000
  Null factor            : 0.0000000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:46:10.73

  Workload Column group  : LAST_NAME,    FIRST_NAME,    MIDDLE_INITIAL,
ADDRESS_DATA_1, ADDRESS_DATA_2, CITY,   STATE,  POSTAL_CODE,    SEX,
BIRTHDAY,       STATUS_CODE
  Duplicity factor       : 1.5625000
  Null factor            : 0.3600000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:46:10.74

------------------------------------------------------------------------

Optimizer Statistics for table : JOB_HISTORY

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 2.7400000
  Null factor            : 0.0000000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:54:09.62

  Workload Column group  : EMPLOYEE_ID,    JOB_CODE,       JOB_START,
JOB_END,        DEPARTMENT_CODE,        SUPERVISOR_ID
  Duplicity factor       : 1.5930233
  Null factor            : 0.3649635
  First created time     :  3-JUL-1996 10:57:47.65
  Last collected time    :  3-JUL-1996 10:57:47.65
```

1.18 RMU Delete Optimizer_Statistics Command

```
$ !
$ ! Delete one of the entries for JOB_HISTORY
$ !
$ RMU/DELETE OPTIMIZER_STATISTICS MF_PERSONNEL.RDB/TABLE=(JOB_HISTORY) -
_$ /COLUMN_GROUP=(EMPLOYEE_ID,JOB_CODE,JOB_START,JOB_END, -
_$ DEPARTMENT_CODE,SUPERVISOR_ID)/LOG
Changing RDB$SYSTEM area to READ_WRITE.
Workload column group deleted for JOB_HISTORY :         EMPLOYEE_ID,
JOB_CODE,       JOB_START,      JOB_END,        DEPARTMENT_CODE,
SUPERVISOR_ID
```

1.19 RMU Dump Command

Displays or writes to a specified output file the contents of database, storage area (.rda), and snapshot (.snp) files, including root information.

Format

RMU/Dump root-file-spec

File Qualifiers	Defaults
/ABMS_Only	See description
/[No]Areas [= storage-area-list]	/Noareas
/Directory_Options=({Areas=,After_Journal= ,Row_Cache=}{option_file_spec})	See description
/End=integer	See description
/[No]Header[=detail-opt, type-opts]	See description
/[No]Lareas [= logical-area-list]	/Nolareas
/Option={Normal | Full | Debug}	/Option=Normal
/Output = file-name	/Output=SYS$OUTPUT
/Page_Number=integer	None
/Restore_Options=file-name	None
/[No]Snapshots [= storage-area-list]	/Nosnapshots
/Spams_Only	See description
/Start=integer	See description
/State=Blocked	See description
/[No]Users	/Nousers

Note

The Start, End and Page_Number qualifiers apply only when the Areas, Lareas, Snapshots, Abms_Only or Spams_Only qualifier is specified.

Description

Use this command to examine the contents of your database root (.rdb), storage area (.rda), and snapshot (.snp) files, to display current settings for database definition options, and to display a list of active database users.

The list of database users is maintained clusterwide in a VMScluster environment.

1.19 RMU Dump Command

You can display the contents of all pages in any data storage area of the database or display the contents of just those pages in which rows and indexes for a specific table are stored.

See the chapter that explains the internal database page format in the *Oracle Rdb Guide to Database Maintenance* for tutorial information.

Depending on your selection of qualifiers, the RMU Dump command can list:

- A formatted display of any number of pages in the storage area of the database.
- A formatted display of any number of pages in a uniform logical area of the database.
- A formatted display of any number of pages in the snapshot area of the database.
- Header information. (This is listed by default if no qualifiers are specified.)
- Current users of the database.

Command Parameters

root-file-spec
A file specification for the database root file whose root file header information, user information, storage area file pages, or snapshot area file pages you want to display.

File Qualifiers

ABMS_Only
Specifies that the RMU/DUMP command will only dump ABM pages in uniform storage areas or in logical areas contained within uniform storage areas.

The ABM pages can be dumped within a limited page range specified by the START and END qualifiers.

If there are no ABM pages within the specified page range or the storage area is a mixed format area or the logical area is contained within a mixed storage area, no ABM pages will be dumped.

This qualifier cannot be specified in the same Dump command as the SPAMS_ Only qualifier. This qualifier cannot be specified in the same Dump command with the Snapshots qualifier.

Areas [=storage-area-list]
Noareas
Specifies a display that consists of storage area pages. You can specify storage areas by name or by the area's ID number.

If you specify more than one storage area, separate the storage area names or ID numbers in the storage area list with a comma, and enclose the list within parentheses.

You can also specify the Areas=* qualifier to display all storage areas. If you do not specify the Areas qualifier, none of the storage areas are displayed.

You can use the Start and End qualifiers to display a range of storage area pages.

The Areas qualifier can be used with indirect file references. See Section 1.3 for more information.

Directory_Options=({Areas=,After_Journal=,Row_Cache=}{option_file_spec})
The Directory_Option qualifier is used to create storage area, after image journal and row cache option files for use with the /OPTIONS qualifier in the RMU/RECOVER command. These option files will contain entries for all storage areas, after image journal files and row caches defined for the database. These option files can then be edited by the user to eliminate storage area, after image journal or row cache entries that will not be created during the RMU/RECOVER session, or they can be used without being edited since entries for storage areas, after image journal files and row caches not created during the recovery operation will be ignored.

The syntax for this qualifier for storage areas is:

```
/DIRECTORY_OPTIONS=AREAS=option_file_specification
```

The syntax for this qualifier for after image journal files is:

```
/DIRECTORY_OPTIONS=AFTER_JOURNAL=option_file_specification
```

The syntax for this qualifier for row caches is:

```
/DIRECTORY_OPTIONS=ROW_CACHE=option_file_specification
```

To specify two or more of these options with the /DIRECTORY_OPTIONS qualifier, use the following syntax.

```
/DIRECTORY_OPTIONS=(AREAS=option_file_specification, -
                    AFTER_JOURNAL=option_file_specification, -
                    ROW_CACHE=option_file_specification)
```

The following example shows all three options used with the /DIRECTORY_ OPTIONS qualifier in the RMU/DUMP command.

```
$ RMU/DUMP-
  /DIRECTORY_OPTIONS=(AREAS=DISK:[DIRECTORY]RECOVOPTAREAS.OPT, -
  AFTER_JOURNAL=DISK:[DIRECTORY]RECOVOPTAIJ.OPT, -
  ROW_CACHE=DISK:[DIRECTORY]RECOVOPTRCACHE.OPT) -
  DISK:[DIRECTORY]TEST
```

End=integer
Specifies the highest-numbered area or snapshot page to include in the display. The default is the last page.

If you also use the Lareas qualifier, note that the Start and End qualifiers specify a page range relative to the logical area, not a specific storage area page number.

Header[=(detail-opt, type-opts)]
Noheader
Indicates whether to include the database header in the output. Specify the Header qualifier to include all database header information in the output. Specify the Noheader qualifier to suppress the database header listing. Specify the Header=(detail-opt, type-opts) qualifier to limit the output from the header to specific items of interest. Use the detail-opt options (Brief or Detail) to limit the amount of output. Use the type-opt options to limit the output to specific types of information.

Table 1–8 summarizes the Header options and the effects of specifying each option.

Table 1–8 RMU Dump Command Header Options

Option	Effect
All	Generates the full output of all the header information. If you specify this option and other Header options, the other options are ignored. This is the default option.
Areas	Output displays information about active storage areas and snapshot areas.
Backup	Output displays information about backup and recovery.

(continued on next page)

Table 1–8 (Cont.) RMU Dump Command Header Options

Option	Effect
Brief	Generates a summary of the requested database root file information.
Buffers	Output displays information about database buffers.
Corrupt_Page	Output displays the Corrupt Page Table (CPT).
Detail	Generates a complete report of the requested database root file information. This is the default.
Fast_Commit	Output displays information about whether fast commit is enabled or disabled, whether commit to AIJ optimization is enabled or disabled, the AIJ checkpointing intervals, and the transaction interval.
Hot_Standby	Output displays information regarding hot standby databases.
Locking	Output displays information about database locking, such as whether or not adjustable record locking, carry-over lock optimization, and lock tree partitioning are enabled or disabled, and fanout factors.
Journaling	Output displays information about RUJ and AIJ journaling.
Nodes	Output displays names of nodes that are accessing the specified database.
Parameters	Output displays basic root file header information.

(continued on next page)

1.19 RMU Dump Command

Table 1–8 (Cont.) RMU Dump Command Header Options

Option	Effect
Root_Record	Output describes the Oracle Rdb specific section of the database root. This includes backup, restore, verify, and alter timestamps, as well as flags that indicate that no such operation has been performed. The bootstrap DBKEY is used to locate the RDB$DATABASE row for this database, and then the other system tables. If an alternate bootstrap DBKEY exists, then this database has been converted using RMU Convert Nocommit command. In this case, the current metadata version is displayed.
Row_Caches	Output displays information about row caches.
Security_Audit	Output displays information about security auditing.
Sequence_Numbers	Output displays database sequence numbers.
Users	Output displays information about active database users.

If you specify both the Detail option and the Brief option, Detail takes precedence. If you specify the All option and other detail-opt options, the All option takes precedence. If you specify the Brief option or the Detail option only, the default for the type-opt is All. If you specify type-opts options, but do not specify a detail-opt option, the default for the detail-opt is Detail.

If you specify more than one option, separate the options with commas and enclose the list within parentheses.

See the Usage Notes section for information on understanding the derived values found in the database header.

The Header=All and Header=Root_Record qualifiers output information on the use of the RMU Alter command on the specified database. For example, you see the following line in the output if you have never used the RMU Alter command on the database:

```
Database has never been altered
```

Do not confuse this with alterations made by SQL ALTER statements. Information about alterations made with the SQL ALTER statement is not included in the output from the RMU Dump command.

If you specify the Areas, Lareas, or Snapshots qualifier, the Noheader qualifier is the default. Otherwise, Header=(All, Detail) is the default.

It is invalid to specify the Header=Root_Record and the Option=Debug qualifiers in the same Oracle RMU command line.

See the *Oracle Rdb7 and Oracle CODASYL DBMS: Guide to Hot Standby Databases* manual for information about the "Hot Standby" references in the database header.

For complete information on the contents of the database header, see the *Oracle Rdb Guide to Database Maintenance*.

Lareas[=logical-area-list]
Nolareas
Specifies a display that consists of storage area pages allocated to a logical area or areas. In a single-file database, each table in the database is stored in its own logical area.

You cannot use the Lareas qualifier with logical areas that are stored in storage areas that have a mixed page format.

If you specify more than one logical area name, separate the storage area names in the logical area list with a comma, and enclose the list within parentheses.

You can also specify the Lareas=* qualifier to display all logical areas that have a uniform page format.

The default is the Nolareas qualifier.

The Lareas qualifier can be used with indirect file references. See Section 1.3 for more information.

Option=type
Specifies the type of information and level of detail the output will include. Three types of output are available:

- Normal

  The output includes summary information. This is the default.

- Full

  In addition to the Normal information, the output includes more detailed information.

- Debug

  In addition to Normal and Full information, the output includes internal information about the data. In general, use the Debug option for diagnostic support purposes.

Output=file-name
Specifies the name of the file where output is to be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Page_Number=integer
Specifies the number of the page to dump. The default is none.

If you also use the Lareas qualifier, note that Page_Number specifies a page number relative to the logical area, not a specific storage area page.

Restore_Options=file-name
Generates an options file designed to be used with the Options qualifier of the RMU Restore command.

The Restore_Options file is created by reading the database root file. Therefore, there is no guarantee that this options file will work with all backup files you attempt to restore with a Restore operation. For example, if areas have been added or deleted from the database since the backup file was created, there will be a mismatch between the Restore_Options file and the backup file. Similarly if the backup file was created by a backup by-area operation, the Restore_Options file may refer to areas that are not in the backup file.

By default a Restore_Options file is not created. If you specify the Restore_ Options qualifier and a file, but not a file extension, Oracle RMU uses an extension of .opt by default.

Snapshots[=storage-area-list]
Nosnapshots
Specifies a display that consists of snapshot file pages. The RMU Dump command does not display snapshot pages if you omit the Snapshots qualifier or if you specify the Nosnapshots qualifier.

In a single-file database, there is only one snapshot file. In a multifile database, each storage area has a corresponding snapshot file. Note that this parameter specifies the storage area name, not the snapshot file name. If you specify more than one storage area name, separate the storage area names with commas, and enclose the storage-area-list within parentheses. If you specify the Snapshots qualifier without a storage area name, information is displayed for all snapshot files.

You can use the Start and End qualifiers to display a range of snapshot file pages.

The default is the Nosnapshots qualifier.

The Snapshots qualifier can be used with indirect file references. See Section 1.3 for more information.

Spams_Only

Allows you to dump only the space area management (SPAM) pages in the selected areas and page range.

A common usage for the RMU Dump command is to track down problems with storage allocation and record placement. When this qualifier is used, the SPAM pages are dumped, allowing you to locate the individual data pages that you want to examine.

There is no negated form for this qualifier, and, if it is omitted, all the selected pages are dumped.

The Start and End qualifiers can be used with the Spams_Only qualifier.

Start=integer

Specifies the lowest-numbered area or snapshot page to include in the display. The default is the first page; that is, the Start=1 qualifier.

If you also use the Lareas qualifier, note that the Start and End qualifiers specify a page range relative to the logical area, not a specific storage area page number.

State=Blocked

Specifies a list of all unresolved distributed transactions in the blocked database. A **blocked database** is a database that is not committed or rolled back and is involved in an unresolved distributed transaction. The State=Blocked qualifier displays the following information about each transaction:

- Process identification (PID)
- Stream identification
- Monitor identification
- Transaction identification
- Name of the recovery journal
- Transaction sequence number (TSN)
- Distributed transaction identifier (TID)

- Name of the node on which the failure occurred
- Name of the node initiating the transaction (parent node)

You can use the State=Blocked qualifier only with the Users qualifier. For information on resolving unresolved transactions with the RMU Dump command, see the *Oracle Rdb7 Guide to Distributed Transactions*.

Users
Nousers
Lists information about the current users of the database, including all users in a VMScluster environment. Oracle RMU does not consider a process that is running the Performance Monitor (with the RMU Show Statistics command or through the Windowing interface) to be a database user.

The default is Nousers.

Usage Notes

- To use the RMU Dump command with the Areas qualifier or the Lareas qualifier or the Snapshots qualifier for a database, you must have the RMU$DUMP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

  To use the RMU Dump command with the Header qualifier for a database, you must have the RMU$DUMP, RMU$BACKUP, or RMU$OPEN privileges in the root file access control list (ACL) for the database, or the OpenVMS SYSPRV or BYPASS privilege.

  To use the RMU Dump command with the Users qualifier, you must have the RMU$DUMP, RMU$BACKUP, or RMU$OPEN privileges in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- The Spams_Only qualifier conflicts with the Lareas and Snapshots qualifiers; an error is generated if you specify the Spams_Only qualifier with either of the other qualifiers.
- The Header=All and Header=Buffers qualifiers provide two derived values to provide an *estimated* size of the global section. These appear in the dump file as:

```
Derived Data...
   - Global section size
       With global buffers disabled is 43451 bytes
       With global buffers enabled is 941901 bytes
```

The first value (With global buffers disabled) indicates the approximate size of the global section when local buffers are being used. The second value (With global buffers enabled) indicates the approximate size of the global section if you were to enable global buffers.

You can use these values to determine approximately how much bigger the global section becomes if you enable global buffers. This allows you to determine, without having to take the database off line, how much larger to make the VIRTUALPAGECNT and GBLPAGES SYSGEN parameters to accommodate the larger global section.

However, note that you must take the database off line if you decide to enable global buffers and you must shut down and reboot the system to change the SYSGEN parameters. It is recommended that you run AUTOGEN after you change SYSGEN parameters.

Also note that these changes may require you to change the MONITOR account quotas as well to ensure the paging file quota is adequate.

Examples

Example 1

The following example displays the header information for the mf_personnel database on the terminal screen:

```
$ RMU/DUMP MF_PERSONNEL
```

Example 2

The following example generates a list of unresolved transactions for the mf_personnel database:

```
$ RMU/DUMP/USERS/STATE=BLOCKED MF_PERSONNEL
```

Example 3

The following example shows the command you might use to view the SPAM pages associated with the area EMPIDS_LOW:

```
$ RMU/DUMP/NOHEADER/AREAS=(EMPIDS_LOW)/SPAMS_ONLY -
_$ MF_PERSONNEL/OUTPUT=DUMP.LIS
```

Example 4

The following example demonstrates the use of the Restore_Options qualifier. The first command performs a dump operation on the mf_personnel database and creates a Restore_Options file. The second command shows a portion of the contents of the options file. The last command demonstrates the use of the options file with the RMU Restore command.

```
$ RMU/DUMP MF_PERSONNEL.RDB /RESTORE_OPTIONS=MF_PERS.OPT -
_$ /OUTPUT=DUMP.LIS
$ TYPE MF_PERS.OPT
!  Options file for database USER1:[DB]MF_PERSONNEL.RDB;1
!  Created 19-JUL-1995 14:55:17.80
! Created by DUMP command

RDB$SYSTEM -
        /file=USER2:[STO]MF_PERS_DEFAULT.RDA;1 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /snapshot=(allocation=100, -
                  file=USER2:[SNP]MF_PERS_DEFAULT.SNP;1)

DEPARTMENTS -
        /file=USER3:[STO]DEPARTMENTS.RDA;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=100, -
                  file=USER3:[SNP]DEPARTMENTS.SNP;1)

   .
   .
   .
$ RMU/RESTORE MF_PERSONNEL.RBF/OPTIONS=MF_PERS.OPT
```

Example 5

The following command generates a detailed display of backup, recovery, RUJ, and AIJ information for the mf_personnel database.

```
$ RMU/DUMP/HEADER=(BACKUP,JOURNALING) MF_PERSONNEL.RDB
```

See the *Oracle Rdb Guide to Database Maintenance* and the *Oracle Rdb7 Guide to Distributed Transactions* for more examples showing the RMU Dump command and the output.

Example 6

The following example dumps all ABM pages contained in all uniform storage areas in the specified Rdb database.

```
$ RMU/DUMP/ABMS_ONLY/OUT=DMP.OUT MF_PERSONNEL
```

Example 7

In the following example, only the ABM pages contained in the named uniform storage area in the specified Rdb database are dumped.

```
$ RMU/DUMP/ABMS_ONLY/AREA=RDB$SYSTEM MF_PERSONNEL
```

Example 8

In the following example, only the ABM pages contained in the named logical area in a uniform storage area in the specified Rdb database are dumped.

```
$ RMU/DUMP/ABMS_ONLY/LAREA=RDB$RELATIONS MF_PERSONNEL
```

Example 9

In the following example, only the ABM pages contained within the specified page range in the named uniform storage area in the specified Rdb database are dumped.

```
$ RMU/DUMP/ABMS_ONLY/AREA=RDB$SYSTEM/START=1/END=5 MF_PERSONNEL
```

Example 10

The following example, created by the RMU/DUMP /DIRECTORY_OPTIONS qualifier, shows the format of the option file used with the RMU/RECOVER /OPTIONS qualifier for storage areas. The live data storage area name is followed by /DIRECTORY=DISK:[DIRECTORY] for the live storage area file to specify the directory specification for the storage area *.RDA file. This is followed by /SNAPSHOT_DIRECTORY=DISK:[DIRECTORY] for the storage area snaphot file to specify the directory specification for the storage area *.SNP file. If /SNAPSHOT_DIRECTORY is not specified, the directory specification specified by /DIRECTORY is used for both the live and snapshot storage area files.

```
!  Recover Areas Directory Options file for database
!  DISK:[DIRECTORY]FILENAME.EXT;VERSION
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

RDB$SYSTEM -
        /directory=DISK:[DIRECTORY] -
        /snapshot_directory=DISK:[DIRECTORY]
```

1.19 RMU Dump Command

```
TEST_A1 -
        /directory=DISK:[DIRECTORY] -
        /snapshot_directory=DISK:[DIRECTORY]

TEST_A2 -
        /directory=DISK:[DIRECTORY] -
        /snapshot_directory=DISK:[DIRECTORY]

TEST_A3 -
        /directory=DISK:[DIRECTORY] -
        /snapshot_directory=DISK:[DIRECTORY]

TEST_A4 -
        /directory=DISK:[DIRECTORY] -
        /snapshot_directory=DISK:[DIRECTORY]
```

Example 11

The following example, created by the /DIRECTORY_OPTIONS qualifier of the RMU/DUMP command, shows the format of the option file used with the RMU/RECOVER /OPTIONS qualifier for after image journal files. The after image journal file name is followed by /DIRECTORY=DISK:[DIRECTORY] to specify the after image journal file directory specification. This is followed by /BACKUP_DIRECTORY=DISK:[DIRECTORY] to specify the directory specification for the after image journal backup file. If /BACKUP_DIRECTORY is not specified, the directory specification specified by /DIRECTORY is used for both the after image journal file and the after image journal backup file. If no backup directory is defined for the after image journal entry in the database, the backup directory specification will be ignored.

```
!  Recover After Journal Options file for database
!  DISK:[DIRECTORY]FILENAME.EXT;VERSION
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

TEST_J1 -
        /directory=DISK:[DIRECTORY] -
        /backup_directory=DISK:[DIRECTORY]

TEST_J2 -
        /directory=DISK:[DIRECTORY] -
        /backup_directory=DISK:[DIRECTORY]

TEST_J3 -
        /directory=DISK:[DIRECTORY] -
        /backup_directory=DISK:[DIRECTORY]
```

```
TEST_J4 -
        /directory=DISK:[DIRECTORY] -
        /backup_directory=DISK:[DIRECTORY]
```

Example 12

The following example, created by the /DIRECTORY_OPTIONS qualifier of the RMU/DUMP command, shows the format of the option file used with the RMU/RECOVER /OPTIONS qualifier for row caches. The row cache name is followed by /DIRECTORY=DISK:[DIRECTORY] to specify the row cache directory specification. If no directory specifcation is defined for the row cache entry in the database, the directory specification will be ignored.

```
!  Recover Row Cache Directory Options file for database
!  DISK:[DIRECTORY]FILENAME.EXT;VERSION
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

TEST_A1 -
        /directory=DISK:[DIRECTORY]

TEST_A2 -
        /directory=DISK:[DIRECTORY]

TEST_A3 -
        /directory=DISK:[DIRECTORY]

TEST_A4 -
        /directory=DISK:[DIRECTORY]
```

Example 13

In the following example for database DISK:[HOME]TEST.RDB, the first RMU/RECOVER command uses the /DIRECTORY qualifier to change all directories from DISK:[HOME] to DISK:[NEW] for the storage areas TEST_A3 and TEST_A4, the after image journal files TEST_J3 and TEST_J4 and the row caches TEST_A3 and TEST_A4, whose creation was recorded in the journal file TEST_J1_BCK.AIJ. Then, the /DIRECTORY_OPTIONS qualifier in the RMU/DUMP command is used to create option files for all TEST database after image journal files, storage areas and row caches, including those with directories changed to DISK:[NEW]. The TEST database is then deleted and again restored with all directories again set to DISK:[HOME]. The second RMU/RECOVER command then uses the /OPTIONS qualifier to specify the option files created by RMU/DUMP, which include the after image journal file, storage area and row cache directories changed by the first RMU/RECOVER to DISK:[NEW], to change the directories back again from the journaled directory specification of DISK:[HOME] to DISK:[NEW].

1.19 RMU Dump Command

```
$ ! Create the TEST database
$
$ SQL
  CREATE DATABASE FILENAME DISK:[HOME]TEST
  NUMBER OF CLUSTER NODES 1
  RESERVE 6 STORAGE AREAS
  RESERVE 6 JOURNALS
  RESERVE 6 CACHE SLOTS
  ROW CACHE IS ENABLED
  CREATE STORAGE AREA RDB$SYSTEM FILENAME DISK:[HOME]TEST_SYS
  CREATE STORAGE AREA TEST_A1 FILENAME DISK:[HOME]TEST_A1
  CREATE STORAGE AREA TEST_A2 FILENAME DISK:[HOME]TEST_A2
  CREATE CACHE TEST_A1
    CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
  CREATE CACHE TEST_A2
    CACHE SIZE 200 ROWS ROW LENGTH 200 BYTES LOCATION IS 'DISK:[HOME]';
  DISCONNECT ALL;

  ALTER DATABASE FILENAME TEST
    ADD JOURNAL TEST_J1 FILENAME DISK:[HOME]TEST_J1.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J1_BCK.AIJ
    ADD JOURNAL TEST_J2 FILENAME DISK:[HOME]TEST_J2.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J2_BCK.AIJ
    JOURNAL IS ENABLED
      (FAST COMMIT ENABLED);
%RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
  EXIT;
$
$ ! Back up the original database configuration.
$
$ RMU/BACKUP/NOLOG DISK:[HOME]TEST DISK:[HOME]TEST
$
$ ! Add new journals, row caches and storage areas
$
$ SQL
  ALTER DATABASE FILENAME DISK:[HOME]TEST
    ADD JOURNAL TEST_J3 FILENAME DISK:[HOME]TEST_J3.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J3_BCK.AIJ
    ADD JOURNAL TEST_J4 FILENAME DISK:[HOME]TEST_J4.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J4_BCK.AIJ
    ADD CACHE TEST_A3
        CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
    ADD CACHE TEST_A4
        CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
    ADD STORAGE AREA TEST_A3 FILENAME DISK:[HOME]TEST_A3
    ADD STORAGE AREA TEST_A4 FILENAME DISK:[HOME]TEST_A4;
%RDMS-W-FULBCKREQ, The next database backup must be a full backup
  EXIT;
$
$ ! Backup TEST_J1.AIJ, where the creation of the new journals, row caches
$ ! and storage areas in DISK:[HOME] is recorded.
$
```

```
$ RMU/BACKUP/AFTER/NOLOG/NOQUIET DISK:[HOME]TEST DISK:[HOME]TEST_J1_BCK.AIJ
$
$ ! Delete the database.
$
$ SQL
  DROP DATABASE FILENAME DISK:[HOME]TEST;
  EXIT;
$
$ ! Restore the database with all storage areas, after image journals
$ ! and row caches in DISK:[HOME]
$
$ RMU/RESTORE/NOCDD/NOLOG/NORECOVER/DIR=DISK:[HOME] DISK:[HOME]TEST
%RMU-I-AIJRSTAVL, 2 after-image journals available for use
%RMU-I-AIJRSTMOD, 1 after-image journal marked as "modified"
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$
$ ! Recover the database and change the directories from DISK:[HOME] to
$ ! DISK:[NEW] for the new storage areas, after image journals and row
$ ! caches whose creation was recorded in the backed up journal file
$ ! TEST_J1_BCK.AIJ
$
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
 /DIRECTORY=(AREAS=DISK:[NEW], -
 AFTER_JOURNAL=DISK:[NEW], -
 ROW_CACHE=DISK:[NEW]) -
 DISK:[HOME]TEST_J1_BCK.AIJ
%RMU-I-LOGRECDB, recovering database file DISK:[HOME]TEST_J1_BCK.AIJ;1
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
needed will be 1
$
$ ! Create directory option files for all storage areas,
$ ! after image journals and row caches in DISK:[HOME]
$ ! or DISK:[NEW]
$
$ RMU/DUMP-
 /DIRECTORY_OPTIONS=(AREAS=DISK:[HOME]RECOVOPTAREAS.OPT, -
 AFTER_JOURNAL=DISK:HOME]RECOVOPTAIJ.OPT, -
 ROW_CACHE=DISK:[HOME]RECOVOPTRCACHE.OPT) -
 DISK:[HOME]TEST
$
$ ! Delete the database.
$
$ SQL$
  DROP DATABASE FILENAME DISK:[HOME]TEST;
  EXIT;
$
$ ! Restore the database with all storage areas, after image journals
$ ! and row caches again in DISK:[HOME]
```

1.19 RMU Dump Command

```
$
$ RMU/RESTORE/NOCDD/NOLOG/NORECOVER/DIR=DISK:[HOME] DISK:[HOME]TEST
%RMU-I-AIJRSTAVL, 2 after-image journals available for use
%RMU-I-AIJRSTMOD, 2 after-image journals marked as "modified"
%RMU-F-AIJENBOVR, enabling AIJ journaling would overwrite an existing journal
%RMU-I-AIJISOFF, after-image journaling has been disabled
$
$ ! Recover the database and again change the directories from DISK:[HOME]
$ ! to DISK:[NEW] for the new storage areas, after image journals and row
$ ! caches whose creation was recorded in the backed up journal file
$ ! TEST_J1_BCK.AIJ
$
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
 /OPTIONS=(AREAS=DISK:[HOME]RECOVOPTAREAS.OPT, -
 AFTER_JOURNAL=DISK:[HOME]RECOVOPTAIJ.OPT, -
 ROW_CACHE=DISK:[HOME]RECOVOPTRCACHE.OPT) -
 TEST$SCRATCH:TEST_J1_BCK.AIJ
  !  Recover Areas Directory Options file for database
  !  DISK:[HOME]TEST.RDB;1
  !  Created 22-JUL-2014 09:37:24.29
  !  Created by DUMP command

  RDB$SYSTEM -
          /directory=DISK:[HOME] -
          /snapshot_directory=DISK:[HOME]

  TEST_A1 -
          /directory=DISK:[HOME] -
          /snapshot_directory=DISK:[HOME]

  TEST_A2 -
          /directory=DISK:[HOME] -
          /snapshot_directory=DISK:[HOME]

  TEST_A3 -
          /directory=DISK:[NEW] -
          /snapshot_directory=DISK:[NEW]

  TEST_A4 -
          /directory=DISK:[NEW] -
          /snapshot_directory=DISK:[NEW]
  !  Recover After Journal Directory Options file for database
  !  DISK:[HOME]TEST.RDB;1
  !  Created 22-JUL-2014 09:37:24.29
  !  Created by DUMP command

  TEST_J1 -
          /directory=DISK:[HOME] -
          /backup_directory=DISK:[HOME]
```

```
  TEST_J2 -
          /directory=DISK:[HOME] -
          /backup_directory=DISK:[HOME]

  TEST_J3 -
          /directory=DISK:[NEW] -
          /backup_directory=DISK:[NEW]

  TEST_J4 -
          /directory=DISK:[NEW] -
          /backup_directory=DISK:[NEW]
  !  Recover Row Cache Directory Options file for database
  !  DISK:[HOME]TEST.RDB;1
  !  Created 22-JUL-2014 09:37:24.29
  !  Created by DUMP command

  TEST_A1 -
          /directory=DISK:[HOME]

  TEST_A2 -
          /directory=DISK:[HOME]

  TEST_A3 -
          /directory=DISK:[NEW]

  TEST_A4 -
          /directory=DISK:[NEW]
%RMU-I-LOGRECDB, recovering database file DISK:[HOME]TEST.RDB;1
%RMU-I-AEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
needed will be 1
%RMU-I-AIJNOENABLD, after-image journaling has not yet been enabled
```

1.20 RMU Dump After_Journal Command

Displays an after-image journal (.aij) file, a backed up .aij file (.aij if the backup is on disk, .aij_rbf if the .aij file was backed up to tape), or an optimized after-image journal (.oaij) file in ASCII format. Use this command to examine the contents of your .aij, .aij_rbf, or .oaij file. Whenever the term .aij file is used in this RMU Dump After_Journal command description, it refers to .oaij and .aij_rbf files, as well as .aij files.

An .aij file contains header information and data blocks. Header information describes the data blocks, which contain copies of data stored in the database file.

Format

RMU/Dump/After_Journal aij-file-name

File Qualifiers	Defaults
/Active_IO=max-reads	/Active_IO=3
/Area=integer	None
/[No]Assist	See description
/[No]Data	/Data
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/End=integer	See description
/First=(select-list)	See description
/Format={Old_File|New-Tape}	Format=Old_File
/Label=(label-name-list)	See description
/Larea=integer	None
/Last=(select-list)	See description
/Librarian[=options]	None
/Line=integer	None
/[No]Media_Loader	See description
/Only=(select-list)	See description
/Option={Statistics|Nostatistics}	Option=Statistics
/Output=file-name	/Output=SYS$OUTPUT
/Page=integer	None
/Prompt={Automatic|Operator|Client}	See description
/[No]Rewind	Norewind
/Start=integer	See description
/State=Prepared	See description

1.20 RMU Dump After_Journal Command

Description

The RMU Dump After_Journal command specifies an .aij file, not a database file, as its parameter, and is a separate command from the RMU Dump command used to display database areas and header information.

The .aij file is in binary format. This command translates the binary file into an ASCII display format.

The RMU Dump After_Journal command always includes the header of the .aij file in the display. You can use the Nodata qualifier to exclude data blocks from the display entirely, or you can use the Start and End qualifiers to restrict the data block display to a specific series of blocks. If you do not specify any of these qualifiers, Oracle RMU includes all data blocks.

Command Parameters

aij-file-name
The .aij file you want to display. The default file type is .aij. For .oaij files, you must specify the file type of .oaij.

File Qualifiers

Active_IO=max-reads
Specifies the maximum number of read operations from a backup device that the RMU Dump After_Journal command will attempt simultaneously. This is not the maximum number of read operations in progress; that value is the product of active system I/O operations.

The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

Area=integer
Identifies a physical database storage area by number. Dump output is limited to the specified area. The minimum value is 1.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

1.20 RMU Dump After_Journal Command

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Data
Nodata
Specifies whether you want to display data blocks of the .aij file, or just the .aij file header.

The Data qualifier is the default. It causes the display of the .aij file data blocks (in addition to the file header) in an ASCII display format.

The Nodata qualifier limits the display to the record headers of the .aij file.

Encrypt=({Value= | Name=}[,Algorithm=])
The Encrypt qualifier decrypts the file of an after-image journal backup.

Specify a key value as a string or the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters type HELP ENCRYPT at the OpenVMS prompt.

This feature requires the OpenVMS Encrypt product to be installed and licensed on your system.

This feature only works for a newer format backup file which has been created using the Format=New_Tape qualifier. You must specify the Format=New_ Tape qualifier with this command if you use the Encrypt qualifier.

End=integer
Specifies the number of the last data block that you want to display. The default integer is the number of the last data block in the file. If you do not use the End qualifier, Oracle RMU displays the entire .aij file.

First=(select-list)
Allows you to specify where you want the dump output to begin. (See the Last=(select-list) qualifier for the end of the range.) If you specify more than one keyword in the select-list, separate the keywords with commas and enclose the list in parentheses. If you specify multiple items in the select list, the first occurrence is the one that will activate Oracle RMU. For example, if you specify First=(Block=100,TSN=0:52),the dump will start when either block 100 or TSN 52 is encountered.

The First and Last qualifiers are optional. You can specify both, either, or neither of them. The keywords specified for the First qualifier can differ from the keywords specified for the Last qualifier.

The select-list of the First qualifier consists of a list of one or more of the following keywords:

- BLOCK=block-number

  Specifies the first block in the AIJ journal.

- RECORD=record-number

  Specifies the first record in the AIJ journal. This is the same as the existing Start qualifier, which is still supported but obsolete.

- TID=tid

  Specifies the first TID in the AIJ journal.

- TIME=date_time

  Specifies the first date and time in the AIJ journal, using absolute or delta date-time format.

- TSN=tsn

  Specifies the first TSN in the AIJ journal, using the standard [n:]m TSN format.

By default, the entire .aij file is dumped.

Format=Old_Rms
Format=New_Tape
Synonymous with Format=Old_File and Format=New_Tape qualifiers. See the description of those qualifiers.

Format=Old_File
Format=New_Tape
Specifies whether the backup or optimized .aij file was written in the old (disk-optimized) or the new (tape-optimized) format. If you enter the RMU Dump After_Journal command without the Format qualifier, the default is the Format=Old_Tape qualifier. You must specify the same Format qualifier as was used with the RMU Backup After_Journal command or the RMU Optimize After_Journal command. If your .aij file resides on disk, you should use the Format=Old_File qualifier.

If you specified the Format=Old_File qualifier when you optimized or backed up the .aij file to tape, you must mount the backup media by using the DCL MOUNT command before you issue the RMU Dump After_Journal command. Because the RMU Dump After_Journal command uses RMS to read the tape, the tape must be mounted as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier with the MOUNT command).

If you specify the Format=New_Tape qualifier, you must mount the backup media by using the DCL MOUNT /FOREIGN command before you issue the RMU Dump After_Journal command.

Similarly, if you specify OpenVMS access (you do not specify the /FOREIGN qualifier on the DCL MOUNT command) although your .aij backup was created using the Format=New_Tape qualifier, you receive an RMU-F-MOUNTFOR error.

The following tape qualifiers have meaning only when used in conjunction with the Format=New_Tape qualifier:

Active_IO
Label
Rewind

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file have been labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses.

In a normal dump after-journal operation, the Label qualifier you specify with the RMU Dump After_Journal command should be the same Label qualifier you specified with the RMU Backup After_Journal command to back up your after-image journal file.

The Label qualifier can be used with indirect file references. See Section 1.3 for more information.

Larea=integer
Identifies a logical database storage area by number. Dump output is limited to the specified area. The minimum value is 0.

Last=(select-list)
Allows you to specify where you want the dump output to end. (See the First=(select-list) qualifier for the beginning range.) If you specify more than one keyword in the select-list, separate the keywords with commas and enclose the list in parentheses. If you specify multiple items in the select list, the first occurrence is the one that will activate Oracle RMU.

The First and Last qualifiers are optional. You can specify both, either, or neither of them. The keywords specified for the First qualifier can differ from the keywords specified for the Last qualifier.

The select-list of the Last qualifier consists of a list of one or more of the following keywords:

- BLOCK=block-number

  Specifies the last block in the AIJ journal.

- RECORD=record-number

  Specifies the last record in the AIJ journal. This is the same as the existing End qualifier, which is still supported but obsolete.

- TID=tid

  Specifies the last TID in the AIJ journal.

- TIME=date_time

  Specifies the last date and time in the AIJ journal, using absolute or delta date-time format.

- TSN=tsn

  Specifies the last TSN in the AIJ journal, using the standard [n:]m TSN format.

By default, the entire .aij file is dumped.

Librarian[=options]

Use the Librarian qualifier to restore files from data archiving software applications that support the Oracle Media Management interface. The file name specified on the command line identifies the stream of data to be retrieved from the Librarian utility. If you supply a device specification or a version number it will be ignored.

Oracle RMU supports retrieval using the Librarian qualifier only for data that has been previously stored by Oracle RMU using the Librarian qualifer.

The Librarian qualifier accepts the following options:

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.

  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian utility.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

  This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed.

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Line=integer
Identifies a database line number. Dump output is limited to the specified line. The minimum value is 0. This qualifier is intended for use during analysis or debugging.

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device from which the file is being read has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, Oracle RMU should recognize this fact. However, occasionally Oracle RMU does not recognize that a tape device has a loader or stacker. Therefore, when the first tape has been read, Oracle RMU issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes Oracle RMU behaves as though a tape device has a loader or stacker when actually it does not.

If you find that Oracle RMU is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that Oracle RMU expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

Only=(select-list)
Allows you to specify one select list item to output. (See also the First=(select-list) and Last=(select-list) qualifiers for specifying a range.) If you specify more than one keyword in the select-list, separate the keywords with commas and enclose the list in parentheses. If you specify multiple items in the select list, the first occurrence is the one that will activate Oracle RMU.

The Only qualifier is optional.

The select-list of the Only qualifier consists of a list of one or more of the following keywords:

- TID=tid

  Specifies a TID in the AIJ journal.

- TSN=tsn

  Specifies a TSN in the AIJ journal, using the standard [n:]m TSN format.

- Type=type-list

  Specifies the types of records to be dumped. The type-list consists of a list of one or more of the following keywords:

  - Ace_header

    Type=A records

  - Checkpoint

    Type=B records

  - Close

    Type=K records

  - Commit

    Type=C records

- Data
  Type=D records
- Group
  Type=G records
- Information
  Type=N records
- Open
  Type=O records
- Optimize_information
  Type=I records
- Prepare
  Type=V records
- Rollback
  Type=R records

By default, the entire .aij file is dumped.

Option=Statistics
Option=Nostatistics

The Option=Statistics qualifier specifies that you want Oracle RMU to include statistics on how frequently database pages are referenced by the data records in the .aij file. In addition, if the database root file is available, the output created by the Options=Statistics qualifier includes the value to specify for the Aij_Buffers qualifier of the RMU Recover command. If several .aij files will be used in your recovery operation, perform an RMU Dump After_Journal on each .aij file and add the recommended Aij_Buffer values. Use the total as the value you specify with the Aij_Buffers qualifier. See Example 2 in the Examples section for an example using this qualifier.

Note that the value recommended for the RMU Recover command's Aij_Buffers qualifier is the *exact* number of buffers required by the data records in the specified .aij file. If you specify fewer buffers, you may see more I/O, but you will not necessarily see performance degrade. (Performance also depends on whether asynchronous batch-writes are enabled.)

Using more buffers than are recommended may result in your process doing more paging than required, and if so, performance degrades.

If you specify the recommended value, note that this does not mean that no buffers are replaced during the recovery operation. The Oracle RMU buffer replacement strategy is affected by whether asynchronous prefetches and asynchronous batch-writes are enabled, and on the contents of the buffers before the recovery operation begins.

If the database root file is not available, the Option=Statistics qualifier does not provide a value for the RMU Recover command's Aij_Buffers qualifier. However, it does provide the statistics on the frequency with which each page is accessed.

Specify the Option=Nostatistics qualifier to suppress .aij statistics generation.

The default for the RMU Dump After_Journal command is Option=Statistics.

Output=file-name
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default file type is .lis, if you specify a file name.

Page=integer
Identifies a database page number. Dump output is limited to the specified page. The minimum value is 1. This qualifier is intended for use during analysis or debugging.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Rewind
Norewind
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The tape is searched for the backup file starting at the beginning-of-tape (BOT). The Norewind qualifier is the default and causes a search for the backup file to be started at the current tape position.

The Rewind and Norewind qualifiers are applicable only to tape devices.

Start=integer
Specifies the number of the first data block that you want to display. If you do not use the Start qualifier, the display begins with the first record in the .aij file.

State=Prepared
Specifies a list of all records associated with unresolved transactions.

For more information on listing unresolved transactions with the RMU Dump After_Journal command, see the *Oracle Rdb7 Guide to Distributed Transactions*.

Usage Notes

- The First and Last qualifiers have been added to make dumping portions of the .aij file easier. The Start and End qualifiers were intended to provide similar functionality, but are difficult to use because you seldom know, nor can you determine, the AIJ record number prior to issuing the command.
- Be careful when searching for TSNs or TIDs as they are not ordered in the AIJ journal. For example, if you want to search for a specific TSN, use the Only qualifier and not the First and Last qualifiers. For example, assume the AIJ journal contains records for TSN 150, 170, and 160 (in that order). If you specify the First=TSN=160 and Last=TSN=160 qualifiers, nothing will be dumped because TSN 170 will match the Last=TSN=160 criteria.
- To use the RMU Dump After_Journal command for an .aij file, you must have the RMU$DUMP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- You receive a file access error message regarding the database's .aij file if you issue the RMU Dump After_Journal command with the active .aij file when there are active processes updating the database. To avoid the file access error message, use the RMU Close command to close the database (which stops entries to the .aij file), then issue the RMU Dump After_ Journal command.
- See the *Oracle Rdb Guide to Database Maintenance* for information on the steps Oracle RMU follows for tape label checking when you execute an RMU Dump After_Journal command using magnetic tapes.
- Use of the wrong value for the Format qualifier typically results in a failure, but sometimes may produce unintelligible results.
- The RMU Dump After_Journal command does not validate the file being dumped. If the file is not an .aij file or a backup of an .aij file, the RMU Dump After_Journal command produces unintelligible output.

Examples

Example 1

The following command generates a list of records associated with unresolved transactions in the .aij file:

```
$ RMU/DUMP/AFTER_JOURNAL/STATE=PREPARED PERSONNEL.AIJ
```

Example 2

The following example shows the value to specify with the Aij_Buffers qualifier along with information on how frequently each page is accessed. The output from this example shows that you should specify the Aij_Buffers=29 qualifier when you recover aij_one.aij. In addition, it shows that pages (1:623-625) were referenced 37 times which means that 8.9% of all data records in the dumped after-image journal file reference this page.

```
$ RMU/DUMP/AFTER_JOURNAL/OPTION=STATISTICS aij_one.aij
.
.
.
Use "/AIJ_BUFFERS=29" when recovering this AIJ journal
1 recovery buffer referenced 37 times (1:623-625): 8.9%
1 recovery buffer referenced 23 times (4:23-25): 5.5%
1 recovery buffer referenced 22 times (4:5-7): 5.3%
1 recovery buffer referenced 21 times (4:44-46): 5.0%
1 recovery buffer referenced 20 times (4:50-52): 4.8%
1 recovery buffer referenced 19 times (4:41-43): 4.6%
2 recovery buffers referenced 18 times (4:38-40): 8.7%
1 recovery buffer referenced 17 times (4:17-19): 4.1%
1 recovery buffer referenced 16 times (4:29-31): 3.8%
2 recovery buffers referenced 15 times (4:35-37): 7.2%
1 recovery buffer referenced 14 times (4:2-4): 3.3%
2 recovery buffers referenced 13 times (4:11-13): 6.3%
3 recovery buffers referenced 12 times (4:8-10): 8.7%
2 recovery buffers referenced 11 times (5:2-4): 5.3%
4 recovery buffers referenced 10 times (4:14-16): 9.7%
1 recovery buffer referenced 9 times (4:47-49): 2.1%
2 recovery buffers referenced 8 times (1:617-619): 3.8%
1 recovery buffer referenced 6 times (4:20-22): 1.4%
1 recovery buffer referenced 2 times (1:503-505): 0.4%
Journal effectiveness: 97.3%
175 data records
412 data modification records
423 total modification records
2 commit records
3 rollback records
```

1.20 RMU Dump After_Journal Command

See the *Oracle Rdb Guide to Database Maintenance* and the *Oracle Rdb7 Guide to Distributed Transactions* for more examples of the RMU Dump After_ Journal command.

Example 3

The following example shows how to start a dump from Block 100 or TSN 52, whichever occurs first.

```
$ RMU/DUMP/AFTER_JOURNAL /FIRST=(BLOCK=100,TSN=0:52) mf_personnel.aij
```

Example 4

This example shows how to dump committed records only.

```
$ RMU/DUMP/AFTER_JOURNAL /ONLY=(TYPE=COMMIT) mf_personnel.aij
```

Example 5

This example shows the dump output when you specify an area, a page, and a line.

```
RMU/DUMP/AFTER_JOURNAL/AREA=3/PAGE=560/LINE=1 mf_personnel.aij

*-------------------------------------------------------------------------------
* Oracle Rdb X7.1-00                                        3-NOV-2005
10:42:23.56
*
* Dump of After Image Journal
*     Filename: DEVICE:[DIRECTORY]MF_PERSONNEL.AIJ;1
*
*-------------------------------------------------------------------------------

2/4             TYPE=D, LENGTH=122, TAD= 3-NOV-2005 10:31:12.56, CSM=00
    TID=6, TSN=0:640, AIJBL_START_FLG=01, FLUSH=00, SEQUENCE=1
        MODIFY: PDBK=3:560:1, LDBID=0, PSN=0, FLAGS=00, LENGTH=84

                           0022  0000  line 1 (3:560:1) record type 34
                        00 0001  0002  Control information
                                 ....  79 bytes of static data

86726576696C6F54343631303000010D  0005   data '...00164Toliver.'
5020363431411120846E69766C410420  0015   data ' .Alvin. .A146 P'
009820876563616C50206C6C656E7261  0025   data 'arnell Place. ..'
3330484E12208B6175726F636F684307  0035   data '.Chocorua. .NH03'
  20F03100630F72B31C00004D373138  0045   data '817M...³r.c.1ð '

2/6             TYPE=D, LENGTH=224, TAD= 3-NOV-2005 10:31:12.56, CSM=00
    TID=6, TSN=0:641, AIJBL_START_FLG=01, FLUSH=00, SEQUENCE=3
        MODIFY: PDBK=3:560:1, LDBID=0, PSN=1, FLAGS=00, LENGTH=84
```

```
                            0022  0000  line 1 (3:560:1) record type 34
                          00 0001  0002  Control information
                                   ....  79 bytes of static data
86726576696C6F54343631303000010D  0005   data '...00164Toliver.'
5020363431411120846E69766C410420  0015   data ' .Alvin. .A146 P'
009820876563616C50206C6C656E7261  0025   data 'arnell Place. ..'
3330484E12208B6175726F636F684307  0035   data '.Chocorua. .NH03'
  20F03100630F72B31C00004D373138  0045   data '817M...³r.c.1ð '

3/9             TYPE=D, LENGTH=330, TAD= 3-NOV-2005 10:31:12.73, CSM=00
    TID=6, TSN=0:642, AIJBL_START_FLG=01, FLUSH=00, SEQUENCE=5
        MODIFY: PDBK=3:560:1, LDBID=0, PSN=2, FLAGS=00, LENGTH=84

                            0022  0000  line 1 (3:560:1) record type 34
                          00 0001  0002  Control information
                                   ....  79 bytes of static data
86726576696C6F54343631303000010D  0005   data '...00164Toliver.'
5020363431411120846E69766C410420  0015   data ' .Alvin. .A146 P'
009820876563616C50206C6C656E7261  0025   data 'arnell Place. ..'
3330484E12208B6175726F636F684307  0035   data '.Chocorua. .NH03'
  20F03100630F72B31C00004D373138  0045   data '817M...³r.c.1ð '

Use "/AIJ_BUFFERS=3" when recovering this AIJ journal.
Make sure you have enough working set and pagefile quota
for the recommended number of buffers.

1 recovery buffer referenced 3 times (3:559-561): 50.0%
1 recovery buffer referenced 2 times (3:436-438): 33.3%
1 recovery buffer referenced 1 time (3:134-136): 16.6%
Journal effectiveness: 54.5%

3 data records
6 data modification records
11 total modification records
3 commit records
```

1.21 RMU Dump Audit Command

This command accesses the named OpenVMS audit journal and extracts some or all of the Oracle Rdb specified audit rows. The output can be written in a list format or an archive format (XML).

Format

RMU/Dump/Audit root-file-spec input-file-name

Command Qualifiers	Defaults
/Before=date-time	None
/Format=formatting-options	List
/Log	/NoLog
/Output[=file-spec]	/Output=SYS$OUTPUT
/Since=date-time	None
/Type=type-list	/Type=All

Description

When RMU/SET AUDIT is used to enable auditing for a database, Oracle Rdb writes records to the OpenVMS audit journal (for example SYS$MANAGER:SECURITY.AUDIT$JOURNAL). This command can be used to dump selected records from an OpenVMS AUDIT journal for a specific database for review.

This command is closely related to the RMU/LOAD/AUDIT command in that it reads and processes the rows from the audit journal.

Command Parameters

root-file-spec
The file specification for the database root file that was audited. The default file extension is .rdb. Only audit records for this database will be dumped.

input-file-name
The input-file-name parameter is the name of the journal containing the audit record data to be dumped. The default file extension is .AUDIT$JOURNAL. You can determine the name of the security audit journal by using the DCL SHOW AUDIT/JOURNAL command.

Command Qualifiers

Before=date-time
Specifies the ending date and time for records extracted from the audit journal. The value is a standard OpenVMS date and time. Enclose the date in quotes if it also includes a space between the date and time fields. If omitted, then all records to the end of the journal will be dumped.

Format=formatting-options
This qualifier allows the database administrator to change the default format (LIST) to XML. The XML output is more useful for archiving and historical analysis.

The following formatting options are accepted:

* LIST - the default output displays the attribute and value on a single line. Long values will be split across multiple lines if necessary.
* XML - formats the audit details as an XML record that can be archived.
* CHARACTER_ENCODING_XML - adjusts the character encoding to that appropriate to the data being dumped in XML format. CHARACTER_ ENCODING_XML is not compatible with the LIST keyword.

Log
If specified, RMU will display a summary line reporting the number of records read from the audit journal and the count of those displayed by RMU Dump.

Output=file-spec
Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Since=date-time
Specifies the starting date and time for records extracted from the audit journal. The value is a standard OpenVMS date and time. Enclose the date in quotes if it also includes a space between the date and time fields. If omitted, then all records from the start of the journal will be dumped.

Type=type-list
Select different types of audit records. Values include: ALL, NONE, AUDIT, DACCESS, PROTECTION, and RMU. The list may contain negated values, such as /TYPE=(ALL,NORMU) so that some categories are removed. The default is to display all types of audit records.

Usage Notes

- To use the RMU Dump Audit command, you must have the RMU$SECURITY privilege in the root file ACL for the database whose security audit records are being loaded. If you do not have this privilege, you must have the OpenVMS SECURITY or BYPASS privilege.
- The OpenVMS audit journal may contain data from multiple facilities in addition to Oracle Rdb and may also contain audit records for other databases. Therefore, only a subset may be read and formatted by RMU Dump.
- Each audit record is divided into packets. Each packet contains a piece of audit information. The output from RMU Dump Audit displays the type for the packet (for instance NSA$C_PKT_FINAL_STATUS), and the formatted value. If necessary, long text packets will be wrapped across multiple lines.
- Oracle Rdb uses a combination of OpenVMS types (those starting with NSA$C) and Oracle Rdb tags (those starting with RDBNSA$K). These tags are described below. Please refer to the relevant OpenVMS documentation for descriptions of the NSA$C types).
- The width of the terminal session is used to limit the lines for the output to SYS$OUTPUT.

Table 1–9 RDBNSA$K types

Type	Description
RDBNSA$K_PKT_DBNAME	File specification for the database root file
RDBNSA$K_PKT_TSN	TSN (transaction sequence number) for the user process
RDBNSA$K_PKT_DACCESS	Discretionary access privileges for the user; based on object ACL, and OpenVMS override privileges
RDBNSA$K_PKT_NEW_ACE	Result of a GRANT or REVOKE statement
RDBNSA$K_PKT_OBJ_TYPE	Type of object being altered

(continued on next page)

Table 1–9 (Cont.) RDBNSA$K types

Type	Description
RDBNSA$K_PKT_OLD_ACE	Prior value before a GRANT or REVOKE statement
RDBNSA$K_PKT_OPERATION_CODE	Description of the operation
RDBNSA$K_PKT_RDB_PRIV_USED	Oracle Rdb privilege used for the operation
RDBNSA$K_PKT_RMU_ARGS	The RMU command line for the operation
RDBNSA$K_PKT_STATUS_CODE	OpenVMS condition following the operation attempt

Examples

Example 1

The following example extracts just the DACCESS records since a known time.

```
$ RMU/DUMP/AUDIT-
    MF_PERSONNEL-
    SYS$MANAGER:SECURITY.AUDIT$JOURNAL-
    /TYPE=(NONE,DACCESS)-
    /LOG-
    /SINCE="14-MAR-2007 14:36:16.70"
   .
   .
   .
-------------------------------:
REC_SUBTYPE                    : DACCESS
RDBNSA$K_PKT_DBNAME            : _$1$DGA174:[SMITH.WORK.DB]MF_PE
                               : RSONNEL.RDB;1
NSA$C_PKT_AUDIT_NAME           : SECURITY
NSA$C_PKT_SYSTEM_ID            : 44262
NSA$C_PKT_IMAGE_NAME           : $1$DGA2:[SYS1.SYSCOMMON.][SYSEX
                               : E]SQL$73.EXE
NSA$C_PKT_PROCESS_ID           : 2B60A272
NSA$C_PKT_PROCESS_NAME         : Ian Smith
NSA$C_PKT_SYSTEM_NAME          : MALIBU
NSA$C_PKT_TIME_STAMP           : 14-MAR-2007 14:36:16.7079869
NSA$C_PKT_TERMINAL             : TNA38:
RDBNSA$K_PKT_TSN               : 0:9985
NSA$C_PKT_SUBJECT_OWNER        : [PROD,SMITH]
NSA$C_PKT_USERNAME             : SMITH
NSA$C_PKT_MESSAGE_TYPE_STR     : Attempted table access
NSA$C_PKT_OBJECT_NAME          : JOB_HISTORY
RDBNSA$K_PKT_OBJ_TYPE          : TABLE
```

1.21 RMU Dump Audit Command

```
RDBNSA$K_PKT_OPERATION_CODE    : Protection Change
RDBNSA$K_PKT_DACCESS           : SELECT,INSERT,UPDATE,DELETE,CRE
                               : ATE,ALTER,DROP,OPERATOR,DBADM,R
                               : EFERENCES
RDBNSA$K_PKT_PRIV_DESIRED      : DBCTRL
RDBNSA$K_PKT_STATUS_CODE       : Oracle Rdb privilege override
NSA$C_PKT_FINAL_STATUS         : %SYSTEM-S-NORMAL
RDBNSA$K_PKT_RDB_PRIV_USED     : DBADM
-------------------------------:
%RMU-I-DATRECREAD,  6454 data records read from input file.
%RMU-I-DATRECUNL,   4 data records unloaded.
```

Example 2

The following example extracts details for a time range in XML format.

```
$ RMU/DUMP/AUDIT -
    EVENTS_DB -
    SYS$MANAGER:SECURITY.AUDIT$JOURNAL -
    /FORMAT=XML -
    /SINCE="12-MAR-2013 16:02:01.40" -
    /BEFORE="12-MAR-2013 16:02:01.97" -
    /LOG
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- RMU Dump Audit for Oracle Rdb V7.3-10 -->
<!-- Generated: 12-MAR-2013 16:10:16.15 -->
<!-- Database: _$1$DGA174:[PRODUCTION.DATABASES]EVENTS_DB.RDB;1 -->
<!-- Since: 2013-03-12T16:02:01.4000000 -->
<!-- Before: 2013-03-12T16:02:01.9700000 -->
```

```
<audit>
  <audit_record type="AUDIT">
    <database_name>_$1$DGA174:[PRODUCTION.DATABASES]EVENTS_DB.RDB;1</database_name>
    <audit_name>SECURITY</audit_name>
    <system_id>44390</system_id>
    <image_name>DISK$VMSSYS:<SYS0.SYSCOMMON.SYSEXE>RMU73.EXE</image_name>
    <process_id>15928887</process_id>
    <process_name>DB_Admin</process_name>
    <system_name>PROD03</system_name>
    <time_stamp>2013-03-12T16:02:01.5802476</time_stamp>
    <terminal>TNA465:</terminal>
    <tsn>0</tsn>
    <subject_owner>[DBA,SMITH]</subject_owner>
    <username>SMITH       </username>
    <message_type>Auditing change</message_type>
    <rmu_command>RMU/SET AUDIT/TYPE=ALARM/START EVENTS_DB</rmu_command>
    <privilege_desired>RMU$SECURITY</privilege_desired>
    <status_code>RMU required privilege</status_code>
    <final_status>%SYSTEM-S-NORMAL</final_status>
    <rdb_privilege_used>RMU$SECURITY</rdb_privilege_used>
  </audit_record>
</audit>
<!-- 1 row unloaded -->
%RMU-I-DATRECREAD,  207 data records read from input file.
%RMU-I-DATRECUNL,   1 data records unloaded 12-MAR-2013 16:10:16.16.
$ set noverify
```

Example 3

The following example extracts details for a time range in LIST format.

1.21 RMU Dump Audit Command

```
$ RMU/DUMP/AUDIT -
    EVENTS_DB -
    SYS$MANAGER:SECURITY.AUDIT$JOURNAL -
    /TYPE=(AUDIT) -
    /SINCE="12-MAR-2013 16:02:01.40" -
    /BEFORE="12-MAR-2013 16:02:01.97" -
    /LOG
REC_SUBTYPE                    : AUDIT
RDBNSA$K_PKT_DBNAME            : _$1$DGA174:[PRODUCTION.DATABASES]EVENTS_DB.RDB;1
NSA$C_PKT_AUDIT_NAME           : SECURITY
NSA$C_PKT_SYSTEM_ID            : 44390
NSA$C_PKT_IMAGE_NAME           : DISK$VMSSYS:<SYS0.SYSCOMMON.SYSEXE>RMU73.EXE
NSA$C_PKT_PROCESS_ID           : 215ECCE5
NSA$C_PKT_PROCESS_NAME         : DB_Admin
NSA$C_PKT_SYSTEM_NAME          : PROD03
NSA$C_PKT_TIME_STAMP           : 2013-03-12 16:02:01.5802476
NSA$C_PKT_TERMINAL             : TNA465:
RDBNSA$K_PKT_TSN               : 0:0
NSA$C_PKT_SUBJECT_OWNER        : [DBA,SMITH]
NSA$C_PKT_USERNAME             : SMITH
NSA$C_PKT_MESSAGE_TYPE_STR     : Auditing change
RDBNSA$K_PKT_RMU_ARGS          : RMU/SET AUDIT/TYPE=ALARM/START EVENTS_DB
RDBNSA$K_PKT_PRIV_DESIRED      : RMU$SECURITY
RDBNSA$K_PKT_STATUS_CODE       : RMU required privilege
NSA$C_PKT_FINAL_STATUS         : %SYSTEM-S-NORMAL
RDBNSA$K_PKT_RDB_PRIV_USED     : RMU$SECURITY
------------------------------:
%RMU-I-DATRECREAD,  207 data records read from input file.
%RMU-I-DATRECUNL,   1 data records unloaded 12-MAR-2013 16:19:22.31.
$
```

1.22 RMU Dump Backup_File Command

Displays or writes to a specified output file the contents of a backup file. Use this command to examine the contents of a backup (.rbf) file created by the RMU Backup command.

Format

RMU/Dump/Backup_File [backup-file-name,...]

Command Qualifiers	Defaults
/Active_IO=max-reads	/Active_IO=3
/Area=identity	None
/[No]Assist	See description
/Disk_File=[(Reader_Threads=n)]	/Disk_file=(Reader_Threads=1)
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/End=integer	See description
/[No]Exit_Error	/Noexit_error
/Header_Only	See description
/Journal=file-name	See description
/Label=(label-name-list)	See description
/Librarian[=options]	None
/[No]Media_Loader	See description
/Options=options-list	See description
/Output=file-name	/Output=SYS$OUTPUT
/Page_Number=integer	None
/Process=process-list	See description
/Prompt={Automatic|Operator|Client}	See description
/Restore_Options=file-name	None
/[No]Rewind	/Norewind
/Skip=skip-list	See description
/Start=integer	See description

Description

The RMU Dump Backup_File command reads an .rbf file and displays the contents. It uses an .rbf file, not a database file, as its parameter, and is a separate command from the RMU Dump command. The output captures unrecoverable media errors and indicates if there are unknown backup blocks on tape. This command can can be used to confirm that a backup file is formatted correctly and that the media is readable for the RMU Restore command.

1.22 RMU Dump Backup_File Command

Note

Successful completion of this command does not guarantee that data in a backup file is uncorrupt, nor that the backup file is complete, nor that a restore operation will succeed.

Use the Root, Full, or Debug option to the Option qualifier to dump the database backup header information. The database backup header information includes the name of the backup file and the "Backup file database version". The "Backup file database version" is the version of Oracle Rdb that was executing at the time the backup file was created. The "Oracle Rdb structure level" listed in the section entitled "Database Parameters" is the currently executing version of Oracle Rdb.

The backup header information is contained on the first volume of a database backup file on tape.

If no backup-file-name is provided, then the command must include the /JOURNAL qualifier. In this case, the /OPTION=JOURNAL will direct RMU to dump only the contents of the backup journal (that contains volume information for tape and disk backups).

Command Parameters

backup-file-spec
A file specification for the backup file. The default file type is .rbf.

If you use multiple tape drives, the backup-file-spec parameter must include the tape device specifications. Separate the device specifications with commas. For example:

```
$ RMU/DUMP/BACKUP_FILE $111$MUA0:PERS_FULL.rbf,$112$MUA1: -
_$ /LABEL=BACK01
```

When multiple volume tape files are processed, Oracle RMU dismounts and unloads all but the last volume containing the file, which is the customary practice for multiple volume tape files. See the *Oracle Rdb Guide to Database Maintenance* for more information on using multiple tape drives.

If no backup-file-name is provided, then the command must include the /JOURNAL qualifier. In this case, the /OPTION=JOURNAL will direct RMU to dump only the contents of the backup journal (that contains volume information for tape and disk backups).

Command Qualifiers

Active_IO=max-reads
Specifies the maximum number of read operations from the backup file that the RMU Dump Backup_File command will attempt simultaneously. The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 might improve performance with multiple tape drives.

Area=identity
Only dump the storage area identified by the specified name or ID number. The area name must be the name of a storage area in the database root file and the area ID number must be a storage area ID number in the database root file. This information is contained in the "Database Parameters:" section of the backup file which is output at the start of the dump. Snapshot areas are not contained in the backup file and cannot be specified. If this qualifier is used without the /START and /END qualifiers, all page records in the specified storage area will be output.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Disk_File=[(Reader_Threads=integer)]
Specifies that you want to dump a multiple disk backup file. This is a backup file that was created by the RMU Backup command with the Disk_File qualifier.

The Reader_Threads keyword specifies the number of threads that Oracle RMU should use when performing a multithreaded read operation from disk files. You can specify no more than one reader thread per device specified on the command line (or in the command parameter options file). By default, one reader thread is used.

This qualifier and all qualifiers that control tape operations (Label, Media_ Loader, and Rewind) are mutually exclusive.

1.22 RMU Dump Backup_File Command

Encrypt=({Value= | Name=}[,Algorithm=])
Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

End=integer
Only dump pages ending with the specified page number in the specified storage area. This qualifier cannot be used unless the /AREA qualifier is also specified. If no pages are dumped, either the specified page or range of pages does not exist in the specified area in the backup file, or this qualifier has been used in the same RMU/DUMP/BACKUP command as an /OPTIONS, /SKIP or /PROCESS qualifier option that has excluded the specified page or range of pages from the dump. If this qualifier is not used with the /START qualifier, all page records in the specified storage area ending with the specified page number will be output.

If both the /START and /END qualifiers are specified, the starting page number must be less than or equal to the ending page number. If the starting page number equals the ending page number only the page records for the specified page number are dumped. The block header for each block which contains at least one of the requested pages is dumped followed by the requested page records in that block. The START AREA record is dumped at the start of requested page records and the END AREA record is dumped at the end of the requested page records. By default, the database root parameters are dumped at the very start following the dump header.

Exit_Error
NoExit_Error
This qualifier is used to optionally abort the dump operation as soon as an error is detected reading the backup file. The default is NoExit_Error. With NoExit_Error, the RMU/DUMP/BACKUP operation will only be aborted if a fatal error is detected which prevents RMU/DUMP/BACKUP from continuing to dump the database backup file. With Exit_Error, when the first error is detected while reading a backup file (even if it is non-fatal), the dump operation is aborted and the status of the error which caused the dump to be aborted is saved in the $STATUS symbol when the RMU/DUMP/BACKUP command exits.

Header_Only

This qualifier can only be used when the /Options=Root qualifier is used. This qualifier will insure that only the header information is processed. (Without this qualifier, the user has to wait until the entire backup file (.RBF) is processed to get any dump information. This means that if the backup file is stored on tape and spans multiple tapes, then all tapes have to be mounted and processed.)

```
$ RMU/DUMP /BACKUP/OPTIONS=ROOT /HEADER_ONLY DBNODE$LMA200:GLORY.RBF
*------------------------------------------------------------
* Oracle Rdb V7.2-4                  26-FEB-2009 07:21:58.69
*
* Dump of Database Backup Header
*     Backup filename: GLORY.RBF
*     Backup file database version: 7.2
*
*------------------------------------------------------------

Database Parameters:
    Root filename is "USER1:[BUG.8235615.FIX]GLORY.RDB;1"
    Created at 25-APR-2007 16:43:05.52
    Oracle Rdb structure level is 72.1
    Maximum user count is 23
    Maximum node count is 3
    Database open mode is AUTOMATIC
    Database close mode is AUTOMATIC
    Database will be mapped in process space
    All transaction modes are allowed
    Prestarted transactions are enabled
    Snapshot mode is NON-DEFERRED
    Statistics are enabled
    Operator notification is disabled
    Logical area count is 512
    Storage Areas...
.
.
.
```

Journal=file-name

Allows you improve tape performance by the dump backup file operation by specifying the journal file created by the RMU Backup command with the Journal qualifier.

The RMU Backup command with the Journal qualifier creates the journal file and writes to it a description of the backup operation, including identification of the tape volumes, their contents, and the tape drive name.

The RMU Dump Backup File with the Journal qualifier directs the RMU Dump Backup_File command to read the journal file and identify the tape volumes when the Label qualifier is not specified.

The journal file must be the one created at the time the backup operation was performed. If the wrong journal file is supplied, an informational message is generated, and the specified journal file is not used to identify the volumes to be processed.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file have been labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas, and enclose the list of names within parentheses.

In a normal dump backup operation, the Label qualifier you specify with the RMU Dump Backup_File command should be the same Label qualifier as you specified with the RMU Backup command that backed up your database.

If no label is specified, the system will internally generate one consisting of the first six characters in the backup-file-spec parameter.

See the *Oracle Rdb Guide to Database Maintenance* for information on tape label processing.

The Label qualifier can be used with indirect file references. See Section 1.3 for more information.

Librarian[=options]
Use the Librarian qualifier to restore files from data archiving software applications that support the Oracle Media Management interface. The file name specified on the command line identifies the stream of data to be retrieved from the Librarian utility. If you supply a device specification or a version number it will be ignored.

Oracle RMU supports retrieval using the Librarian qualifier only for data that has been previously stored by Oracle RMU using the Librarian qualifer.

The Librarian qualifier accepts the following options:

- Reader_Threads=n

  Use the Reader_Threads option to specify the number of backup data streams to read from the Librarian utility. The value of *n* can be from 1 to 99. The default is one reader thread. The streams are named BACKUP_FILENAME.EXT, BACKUP_FILENAME.EXT02, BACKUP_FILENAME.EXT03, up to BACKUP_FILENAME.EXT99. BACKUP_FILENAME.EXT is the backup file name specified in the RMU Backup command.

The number of reader threads specified for a database restore from the Librarian utility should be equal to or less than the number of writer threads specified for the database backup. If the number of reader threads exceeds the number of writer threads, the number of reader threads is set by Oracle RMU to be equal to the number of data streams actually stored in the Librarian utility by the backup. If the number of reader threads specified for the restore is less than the number of writer threads specified for the backup, Oracle RMU will partition the data streams among the specified reader threads so that all data streams representing the database are restored.

The Volumes qualifier cannot be used with the Librarian qualifer. Oracle RMU sets the volume number to be the actual number of data streams stored in the specified Librarian utility.

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  – Level 0 traces all error conditions. This is the default.

  – Level 1 traces the entry and exit from each Librarian function.

  – Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian utility.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian implementation documentation for the name and location of this image and how it should be installed. For a parallel RMU backup, define RMU$LIBRARIAN_PATH as a system-wide logical name so that the multiple processes created by a parallel backup can all translate the logical.

```
$ DEFINE /SYSTEM /EXECUTIVE_MODE -
_$ RMU$LIBRARIAN_PATH librarian_shareable_image.exe
```

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image. For a parallel RMU backup, the RMU$DEBUG_SBT logical should be defined as a system logical so that the multiple processes created by a parallel backup can all translate the logical.

The following lines are from a backup plan file created by the RMU Backup/Parallel/Librarian command:

```
Backup File = MF_PERSONNEL.RBF
Style = Librarian
Librarian_trace_level = #
Librarian_logical_names = (-
         logical_name_1=equivalence_value_1, -
         logical_name_2=equivalence_value_2)
Writer_threads = #
```

The "Style = Librarian" entry specifies that the backup is going to a Librarian utility. The "Librarian_logical_names" entry is a list of logical names and their equivalence values. This is an optional parameter provided so that any logical names used by a particular Librarian utility can be defined as process logical names before the backup or restore operation begins. For example, some Librarian utilities provide support for logical names for specifying catalogs or debugging.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device from which the backup file is being read has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, Oracle RMU should recognize this fact. However, occasionally Oracle RMU does not recognize that a tape device has a loader or stacker. Therefore, when the first tape has been read, Oracle RMU issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes Oracle RMU behaves as though a tape device has a loader or stacker when actually it does not.

If you find that Oracle RMU is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that Oracle RMU expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

Options=options-list
Specifies the type of information and level of detail the output will include. If you do not specify the Options qualifier or if you specify the Options=Normal qualifier, the backup file will be read, but dump output is not generated. This is useful for confirming that the backup file is structured correctly and the media is readable for the RMU Restore command. However, this command does not indicate if the data in a backup file is corrupted, nor does it guarantee that a restore operation will succeed.

If you specify more than one option, you must separate the options with a comma, and enclose the options-list parameter within parentheses. Eight types of output are available:

- Records

  Dumps the backup file record structure.

- Blocks

  Dumps the backup file block structure.

- Data

  The Data option can be used with either the Records option, the Blocks option, or both. When specified with the Records and Blocks options, the Data option dumps the contents of the backup file's records and blocks. When you do not specify the Data option, the Records and Blocks options dump the backup file's record structure and block structure only, not their contents.

1.22 RMU Dump Backup_File Command

- Journal

  Dumps the contents of the journal file.

  Use the Journal option of the RMU Dump Backup_File command to direct Oracle RMU to dump the journal file created with the RMU Backup command with the Journal qualifier. The RMU Backup command with the Journal qualifier creates a journal file to which it writes a description of the backup operation, including identification of the tape volumes and their contents. You can use the output of the RMU Dump Backup_File with the Journal qualifier to identify the contents of each of the tapes that comprises the backup file.

- Root

  Dumps the database root file contents as recorded in the backup file. This includes a dump of the database backup header information.

  When the /Options=Root qualifier is used, another qualifier, /HEADER_ ONLY, can be included on the command line. This qualifier will insure that only the header information is processed. (Without this qualifier, the user has to wait until the entire backup file (.RBF) is processed to get any dump information. This means that if the backup file is stored on tape and spans multiple tapes, then all tapes have to be mounted and processed.)

```
$ RMU/DUMP /BACKUP/OPTIONS=ROOT /HEADER_ONLY DBNODE$LMA200:GLORY.RBF
*-------------------------------------------------------------
* Oracle Rdb V7.2-4                  26-FEB-2009 07:21:58.69
*
* Dump of Database Backup Header
*     Backup filename: GLORY.RBF
*     Backup file database version: 7.2
*
*-------------------------------------------------------------
```

```
Database Parameters:
    Root filename is "USER1:[BUG.8235615.FIX]GLORY.RDB;1"
    Created at 25-APR-2007 16:43:05.52
    Oracle Rdb structure level is 72.1
    Maximum user count is 23
    Maximum node count is 3
    Database open mode is AUTOMATIC
    Database close mode is AUTOMATIC
    Database will be mapped in process space
    All transaction modes are allowed
    Prestarted transactions are enabled
    Snapshot mode is NON-DEFERRED
    Statistics are enabled
    Operator notification is disabled
    Logical area count is 512
    Storage Areas...
.
.
.
```

- Normal

  The backup file will be read, but no dump output is generated. This is useful to verify the integrity of the backup file format and to detect media errors.

- Full

  Specifying the Full option is the same as specifying the Root, Records, and Blocks options. Includes a dump of the database backup header information. The contents of the backup file's record structure and block structure are not dumped when the Full option is specified.

- Debug

  Specifying the Debug option is the same as specifying the Root, Records, Blocks, Full, and Data options. The contents of the backup file's header, record structure, and block structure are dumped when the Debug option is specified.

Output=file-name
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Page_Number=integer
Specifies the number of the page to dump. The default is none.

If you also use the Lareas qualifier, note that Page_Number specifies a page number relative to the logical area, not a specific storage area page.

Process=process-list
Specifies a list of keywords that determines how much of the backup file is to be dumped. If you specify more than one type of process-list option, separate the options with a comma, and enclose the process-list parameter within parentheses. You can specify the following three items in the process-list parameter:

- Volumes=integer

  The number of volumes to dump, starting at the position specified in the Skip qualifier for volumes. This option is ignored if the backup file does not reside on tape.

- Blocks=integer

  The number of blocks to dump, starting at the position specified in the Skip qualifier for blocks. This option is ignored if the backup file does not reside on tape.

- Records=integer

  The number of records to dump, starting at the position specified in the Skip qualifier for records. This option is valid regardless of whether the backup file resides on tape or disk.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Restore_Options=file-name
Generates an options file designed to be used with the Options qualifier of the RMU Restore command.

The Restore_Options file is created after the root information has been read from the backup file.

By default, a Restore_Options file is not created. If you specify the Restore_ Options qualifier and a file, but not a file extension, Oracle RMU uses an extension of .opt by default.

Rewind
Norewind
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The Norewind qualifier is the default.

The Rewind and Norewind qualifiers are applicable only to tape devices. You should use these qualifiers only when the target device is a tape device.

See the *Oracle Rdb Guide to Database Maintenance* for information on tape label processing.

Skip=skip-list
Specifies a list of keywords that determines where the output display begins. The keywords indicate the position in the backup file from which to start the dump. If you specify more than one type of Skip position, separate the options with a comma, and enclose the skip-list parameter in parentheses. You can specify the following three items in the skip-list parameter:

- Volumes=integer

  The number of volumes to ignore before starting. This option is ignored if the backup file does not reside on tape.

- Blocks=integer

  The number of blocks to ignore before starting. This option is ignored if the backup file does not reside on tape.

- Records=integer

  The number of records to ignore before starting. This option is valid regardless of whether the backup file resides on tape or disk.

Start=integer
Only dump pages starting with the specified page number in the specified storage area. This qualifier cannot be used unless the /AREA qualifier is also specified. If no pages are dumped, either the specified page or range of pages does not exist in the specified area in the backup file, or this qualifier has been used in the same RMU/DUMP/BACKUP command as an /OPTIONS, /SKIP or /PROCESS qualifier option that has excluded the specified page or range of pages from the dump. If this qualifier is not used with the /END qualifier, all page records in the specified storage area starting with the specified page number will be output.

If both the /START and /END qualifiers are specified, the starting page number must be less than or equal to the ending page number. If the starting page number equals the ending page number only the page records for the specified page number are dumped. The block header for each block which contains at least one of the requested pages is dumped followed by the requested page records in that block. The START AREA record is dumped at the start of requested page records and the END AREA record is dumped at the end of the requested page records. By default, the database root parameters are dumped at the very start following the dump header.

1.22 RMU Dump Backup_File Command

Usage Notes

- To use the RMU Dump Backup_File command for a database, you must have the RMU$DUMP, RMU$BACKUP, or RMU$RESTORE privileges in the root file access control list (ACL) for the database or the OpenVMS BYPASS privilege.

  You must also have read access to the .rbf file.
- If you do not specify the Options qualifier or if you specify the Options=Normal qualifier, the backup file will be read, but dump output will not be generated. This is useful to verify the backup file integrity and to detect media errors.
- See the *Oracle Rdb Guide to Database Maintenance* for examples that show the RMU Dump Backup_File command.

Examples

Example 1

The following commands show the use of the Journal qualifier with the RMU Backup command and the RMU Dump After_Journal command. The first command creates a binary journal file that identifies the tapes used in the backup operation. The second command directs Oracle RMU to read the backup file (using the tapes identified in the BACKUP_JOURNAL.JNL file) to confirm that the backup file is structured correctly and the media is readable for the RMU Restore command. No dump output is generated because the Option qualifier is not specified.

```
$ RMU/BACKUP MF_PERSONNEL.RDB -
_$ $222$DUA20:[BCK]MF_PERSONNEL.RBF/LOG/JOURNAL=BACKUP_JOURNAL.JNL

$ RMU/DUMP/BACKUP_FILE $222$DUA20:[BCK]MF_PERSONNEL.RBF -
_$ /JOURNAL=BACKUP_JOURNAL.JNL
```

Example 2

The following commands show the use of the Journal qualifier with the RMU Backup command and then with the RMU Dump Backup command. The first command creates a binary journal file that identifies the tapes used in the backup operation. The second command dumps the binary journal file created in the first command in ASCII format.

```
$ RMU/BACKUP MF_PERSONNEL.RDB -
_$ $222$DUA20:[BCK]MF_PERSONNEL.RBF/LOG/JOURNAL=BACKUP_JOURNAL.JNL

$ RMU/DUMP/BACKUP_FILE $222$DUA20:[BCK]MF_PERSONNEL.RBF -
_$ /JOURNAL=BACKUP_JOURNAL.JNL/OPTION=JOURNAL
```

Example 3

The following example demonstrates the use of the Restore_Options qualifier. The first command performs a dump operation on the backup file of the mf_personnel database and creates a Restore_Options file. The second command shows a portion of the contents of the options file. The last command demonstrates the use of the options file with the RMU Restore command.

```
$ RMU/DUMP/BACKUP MFP.RBF /RESTORE_OPTIONS=MFP.OPT -
_$ /OPTIONS=NORMAL/OUTPUT=DUMP.LIS
$ TYPE MFP.OPT
!  Options file for database DISK1:[DB]MF_PERSONNEL.RDB;1
!  Created 17-OCT-1995 13:09:57.56
!  Created by DUMP BACKUP command

RDB$SYSTEM -
        /file=DISK2:[RDA]MF_PERS_DEFAULT.RDA;1 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /snapshot=(allocation=248, -
                  file=DISK3:[SNAP]MF_PERS_DEFAULT.SNP;1)

EMPIDS_LOW -
        /file=DISK3:[RDA]EMPIDS_LOW.RDA;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /spams -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=10, -
                  file=DISK4:[SNAP]EMPIDS_LOW.SNP;1)
   .
   .
   .
$ RMU/RESTORE MFP.RBF/OPTIONS=MFP.OPT
```

Example 4

The following example shows the dump of the page records for page 10 in storage area 4 in the MFP.RBF backup file. Since the /START and /END qualifiers both specify page 10, only the page records for that page are dumped. At the start of the dump is the dump header, followed by the database root parameters which are not shown to save space, followed by the block header, which begins with the "HEADER_SIZE" field, for the block which contains the

records for page 10 in storage area 4, followed by the start area record for area 4 (REC_TYPE = 6), the data page header record (REC_TYPE = 7) for page 10, the data page data record (REC_TYPE = 8) for page 10, and ending with the end area record for area 4 (REC_TYPE = 11) which ends the dump.

```
$ RMU/DUMP/BACKUP/AREA=4/START=10/END=10/OPTION=FULL MFP.RBF
*-------------------------------------------------------------------------------
* Oracle Rdb V7.2-420                                    11-JAN-2011 15:50:09.25
*
* Dump of Database Backup Header
*     Backup filename: MFP.RBF
*     Backup file database version: 7.2
*
*-------------------------------------------------------------------------------

Database Parameters:
.
.
.
HEADER_SIZE = 80        OS_ID = 1024    UTILITY_ID = 722
APPLICATION_TYPE = 1    SEQUENCE_NUMBER = 22    MAJ_VER = 1    MIN_VER = 1
VOL_NUMBER = 1  BLOCK_SIZE = 32256      CRC = 0C5D3A78  NOCRC = 00
CRC_ALTERNATE = 00      BACKUP_NAME = MFP.RBF   AREA_ID = 4    HIGH_PNO = 259
LOW_PNO = 1     HDR_CHECKSUM = 9B3D

REC_SIZE = 2    REC_TYPE = 6    BADDATA = 00    ROOT = 00
AREA_ID = 4     LAREA_ID = 0    PNO = 0

REC_SIZE = 32   REC_TYPE = 7    BADDATA = 00    ROOT = 00
AREA_ID = 4     LAREA_ID = 0    PNO = 10

REC_SIZE = 28   REC_TYPE = 8    BADDATA = 00    ROOT = 00
AREA_ID = 4     LAREA_ID = 0    PNO = 10

REC_SIZE = 512  REC_TYPE = 11   BADDATA = 00    ROOT = 00
AREA_ID = 4     LAREA_ID = 0    PNO = 0
```

Example 5

The following example dumps the records for pages 10, 11 and 12 in the RDB$SYSTEM storage area in the MFP.RBF backup file. Following the block header containing the target records that starts with "HEADER_SIZE =", are the start area record for RDB$SYSTEM area 1 (REC_TYPE = 6), then the target ABM page records for pages 10, 11, and 12 (REC_TYPE = 10), and finally the end area record for area RDB$SYSTEM area 1 (REC_TYPE = 11) which ends the dump.

```
$ RMU/DUMP/BACKUP/AREA=RDB$SYSTEM/START=10/END=12/OPTION=FULL MFP.RBF
*-----------------------------------------------------------------------------
* Oracle Rdb V7.2-420                                  14-JAN-2011 14:40:46.88
*
* Dump of Database Backup Header
*     Backup filename: MFP.RBF
*     Backup file database version: 7.2
*
*-----------------------------------------------------------------------------

Database Parameters:
.
.
.

HEADER_SIZE = 80        OS_ID = 1024    UTILITY_ID = 722
APPLICATION_TYPE = 1    SEQUENCE_NUMBER = 1     MAJ_VER = 1     MIN_VER = 1
VOL_NUMBER = 1  BLOCK_SIZE = 32256      CRC = 8329C24B  NOCRC = 00
CRC_ALTERNATE = 00      BACKUP_NAME = MFP.RBF   AREA_ID = 1     HIGH_PNO = 178
LOW_PNO = 1     HDR_CHECKSUM = 40DE

REC_SIZE = 2    REC_TYPE = 6    BADDATA = 00    ROOT = 00       AREA_ID = 1
LAREA_ID = 0    PNO = 0

REC_SIZE = 10   REC_TYPE = 10   BADDATA = 00    ROOT = 00       AREA_ID = 1
LAREA_ID = 3    PNO = 10

REC_SIZE = 10   REC_TYPE = 10   BADDATA = 00    ROOT = 00       AREA_ID = 1
LAREA_ID = 4    PNO = 11

REC_SIZE = 10   REC_TYPE = 10   BADDATA = 00    ROOT = 00       AREA_ID = 1
LAREA_ID = 4    PNO = 12

REC_SIZE = 512  REC_TYPE = 11   BADDATA = 00    ROOT = 00       AREA_ID = 1
LAREA_ID = 0    PNO = 0
```

Example 6

In the following example, the /EXIT_ERROR qualifier is specified with the RMU/DUMP/BACKUP command. When the first error is detected while reading the backup file, MF_PERSONNEL.RBF, the dump operation is aborted and the status of the error which caused the dump to be aborted, RMU-E-BLOCKLOST, is saved in the $STATUS symbol when the RMU/DUMP/BACKUP command exits.

```
$ RMU/DUMP/BACKUP/EXIT_ERROR MF_PERSONNEL.RBF
%RMU-I-DMPTXT_163, No dump option selected. Performing read check.
%RMU-E-BLOCKLOST, block of DEVICE:[DIRECTORY]MF_PERSONNEL.RBF; lost due to
unrecoverable error
%RMU-F-FATALERR, fatal error on DUMP_BACKUP
%RMU-F-FTL_DUMP, Fatal error for DUMP operation at  1-MAY-2013 11:32:13.45
$ SHOW SYMBOL $STATUS
  $STATUS == "%X12C8821A"
```

Example 7

In the following example, the /NOEXIT_ERROR qualifier is first specified with the RMU/DUMP/BACKUP command. This is the default so the second RMU/DUMP/BACKUP command, which does not specify the /NOEXIT_ERROR qualifier, has the same results as the first RMU/DUMP/BACKUP command which does specify the /NOEXIT_ERROR qualifier. When non-fatal errors are detected reading the backup file, MF_PERSONNEL.RBF, the dump operation continues and is not aborted. The status of the last most severe error detected reading the backup file, RMU-E-BLOCKLOST, is saved in the $STATUS symbol when the RMU/DUMP/BACKUP command finishes reading the backup file, not a success status. A count is also given of any soft media errors which were not reported because they did not reoccur when retrying the media read operations.

```
$ RMU/DUMP/BACKUP/NOEXIT_ERROR MF_PERSONNEL.RBF
%RMU-I-DMPTXT_163, No dump option selected. Performing read check.
%RMU-E-BLOCKLOST, block of DEVICE:[DIRECTORY]MF_PERSONNEL.RBF; lost due to
unrecoverable error
%RMU-I-SOFTRERRS, 5 recoverable media errors occurred reading
DEVICE:[DIRECTORY]MF_PERSONNEL.RBF;
$ SHOW SYMBOL $STATUS
  $STATUS == "%X12C8821A"
$ RMU/DUMP/BACKUP MF_PERSONNEL.RBF
%RMU-I-DMPTXT_163, No dump option selected. Performing read check.
%RMU-E-BLOCKLOST, block of DEVICE:[DIRECTORY]MF_PERSONNEL.RBF; lost due to
unrecoverable error
%RMU-I-SOFTRERRS, 5 recoverable media errors occurred reading
DEVICE:[DIRECTORY]MF_PERSONNEL.RBF;
$ SHOW SYMBOL $STATUS
  $STATUS == "%X12C8821A"
```

Example 8

The following example shows how to dump just the backup journal (not to be confused with an after image journal) using the /OPTION=JOURNAL qualifier and omitting the database backup (.rbf) file.

```
$ RMU/DUMP-
    /BACKUP-
    /OPTION=JOURNAL-
    /JOURNAL=MFPERS.JNL

Dump of BACKUP JOURNAL file -- MFPERS.JNL
```

```
Backup File Summary Record, 271. bytes
BACKUP_NAME = RMU/BACKUP        USERNAME = JTESTR       UIC = [000050,000002]
Root creation date = 11-JUL-2013 17:37:31.62
Backup date =  9-MAR-2015 10:02:40.72
Last Full Backup date =  9-MAR-2015 09:59:26.34 OS type = 1024
OS Version = V8.3       System ID register = 2112345648
Utility = Oracle RMU                            Major version = 73
Minor version = 0       BLOCK_SIZE = 32256      GROUP_SIZE = 0  ACTIVE_IO = 3
FULL_BACKUP = 1 ONLINE_BACKUP = 0       AREAS_EXCLUDED = 0
USED_PAGE_MODIFIED_MAP = 00     SYSAREA_FIRST_THRESHOLD = 1
VOLUME_SET =        VOLUME_NUMBER = 1       LIVE_AREAS = 9
VOLUME_AREAS = 9

BACKUP FILE = MFPERS.RBF
COMMAND = RMU/BACKUP/NOLOG/JOURNAL=MFPERS.JNL MF_PERSONNEL MFPERS.RBF

ROOT_FILE = USER2:[TESTING.DUMP_JOURNAL]MF_PERSONNEL.RDB;1

DRIVE = _$1$DGA174
    Area RDB$SYSTEM with ID 1(10) contains 1012 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]MF_PERS_DEFAULT.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area DEPARTMENTS with ID 2(11) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]DEPARTMENTS.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area EMPIDS_LOW with ID 3(12) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_LOW.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area EMPIDS_MID with ID 4(13) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_MID.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area EMPIDS_OVER with ID 5(14) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_OVER.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area EMP_INFO with ID 6(15) contains 706 pages of 1024 bytes
     has the file name USER2:[TESTING.DUMP_JOURNAL]EMP_INFO.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area JOBS with ID 7(16) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]JOBS.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area MF_PERS_SEGSTR with ID 8(17) contains 703 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]MF_PERS_SEGSTR.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area SALARY_HISTORY with ID 9(18) contains 706 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]SALARY_HISTORY.RDA;1
      Full Backup at  9-MAR-2015 10:02:40.72, TSN 448. Recover from journal 0
    Area  with ID 10(1) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]MF_PERS_DEFAULT.SNP;1
    Area  with ID 11(2) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]DEPARTMENTS.SNP;1
    Area  with ID 12(3) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_LOW.SNP;1
    Area  with ID 13(4) contains 100 pages of 1024 bytes
```

1.22 RMU Dump Backup_File Command

```
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_MID.SNP;1
    Area  with ID 14(5) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMPIDS_OVER.SNP;1
    Area  with ID 15(6) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]EMP_INFO.SNP;1
    Area  with ID 16(7) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]JOBS.SNP;1
    Area  with ID 17(8) contains 100 pages of 1024 bytes
      has the file name USER2:[TESTING.DUMP_JOURNAL]MF_PERS_SEGSTR.SNP;1
    Area  with ID 18(9) contains 100 pages of 1024 bytes
     has the file name USER2:[TESTING.DUMP_JOURNAL]SALARY_HISTORY.SNP;1
    Started relative volume # 1 with label  on _$1$DGA174
    Volume 1 contains area 1 starting at 1
    Volume 1 contains area 9 starting at 1
    Volume 1 contains area 7 starting at 1
    Volume 1 contains area 6 starting at 1
    Volume 1 contains area 5 starting at 1

End of BACKUP JOURNAL file dump.
```

1.23 RMU Dump Export Command

Displays the contents of an export interchange (.rbr) file or a formatted .unl file created by the RMU Unload command. This is a useful debugging tool.

Format

RMU/Dump/Export export_file

Command Qualifiers	Defaults
/[No]Data	/Data
/[No]Options[=options-list]	/Nooptions
/Output=file-name	/Output=SYS$OUTPUT

Command Parameters

export-file
The .rbr file or formatted .unl file to be displayed.

Command Qualifiers

Data
Nodata
The Data qualifier specifies that the contents of segmented strings and tables are to be displayed in hexadecimal format along with the ASCII translation. Specifying the Nodata qualifier excludes the contents of segmented strings and tables from the display and generates much less output.

The default is the Data qualifier.

Options=option-list
The Options qualifier allows the user to modify the output from the RMU Dump Export command.

If you specify more than one option, you must separate the options with a comma and enclose the options-list parameter within parentheses.

– ALLOCATION

  When importing databases for testing, the full allocation recorded in the interchange file is often not required. The generated SQL clauses ALLOCATION and SNAPSHOT ALLOCATION are controlled by this option. The default is ALLOCATION. Use NOALLOCATION to omit these clauses from the generated SQL script. This option is ignored

if NOIMPORT_DATABASE is specified or defaulted for the OPTIONS qualifier.

– FILENAME_ONLY

When importing databases for testing, the full file specification for the database root, storage areas and snapshot areas recorded in the interchange file is often not required. The FILENAME clauses are controlled by this option which trims the specification to only the filename portion. The default is NOFILENAME_ONLY. Use FILENAME_ONLY to truncate the file specification in the generated SQL script. This option is ignored if NOIMPORT_DATABASE is specified or defaulted for the OPTIONS qualifier.

– HEADER_SECTION

This option allows the database administrator to display just the header portion of the interchange file and avoid dumping the data or metadata for every row in the table.

– IMPORT_DATABASE

This keyword requests that the output from RMU Dump Export be formatted as a SQL IMPORT DATABASE statement. It uses the database attributes present in the interchange file formatted as SQL clauses. Of particular interest are the CREATE STORAGE AREA clauses which are required to IMPORT the source interchange (.rbr) file.

The keyword HEADER_SECTION is implicitly selected when IMPORT_ DATABASE is used, limiting the I/O to the interchange file to the section containing the database attributes.

The default is NOIMPORT_DATABASE.

Output=file-name
Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Usage Notes

- You do not need Oracle RMU privileges to use the RMU Dump Export command. However, you must have OpenVMS read access to the .rbr or .unl file, or OpenVMS BYPASS privilege.

- If the source interchange file is created by RMU Unload, then it does not contain any IMPORT DATABASE information and the generated SQL script cannot be used to create a database from such an interchange file.

```
$ RMU/DUMP/EXPORT/OPTION=IMPORT_DATABASE EMPLOYEES.UNL/OUT=EMP.SQL
$ SQL$ @EMP.SQL
SQL> IMPORT DATABASE
cont>     from 'DISK1:[TESTING]EMPLOYEES.UNL;1'
cont>     -- ident ' Load/Unload utility'
cont>     -- backup file version 4
cont>     -- database ident 'Oracle Rdb V7.2-131'
cont>     filename 'DB$:MF_PERSONNEL'
cont> ;
%SQL-F-EXTRADATA, unexpected data at the end of the .RBR file
$
```

- The IMPORT_DATABASE option is intended to create a SQL script as an aid to the database administrator. Some editing of the generated script may be required under some circumstances.

  Only a subset of the database attributes are dumped by RMU for the IMPORT_DATABASE output. Continue to use the RMU Dump Export Option=NOIMPORT_DATABASE to see all attributes recorded in the interchange file.

Examples

Example 1

The following is an example of the RMU Dump Export command using the default qualifiers:

```
$ RMU/DUMP/EXPORT EMPLOYEES.UNL

BEGIN HEADER SECTION - (0)
    NONCORE_TEXT HDR_BRP_ID - (38) : Oracle Rdb V7.0 Load/Unload utility
    CORE_NUMERIC HDR_BRPFILE_VERSION - (1) : 4
    NONCORE_TEXT HDR_DBS_ID - (18) : Oracle Rdb V7.0
    NONCORE_TEXT HDR_DB_NAME - (16) : MF_PERSONNEL.RDB
    NONCORE_DATE HDR_DB_LOG_BACKUP_DATE - (8) :  18-JUN-1996 09:31:45.71
END HEADER SECTION - (0)

BEGIN RELATION SECTION - (0)
   .
   .
   .
```

Example 2

1.23 RMU Dump Export Command

The following is an example of how to use the HEADER_SECTION option to display just the header portion of the interchange file, and avoid dumping the data or metadata for every row in the table.

```
$ RMU/DUMP/EXPORT/OPTION=HEADER JOBS.UNL

BEGIN HEADER SECTION - (0)
    NONCORE_TEXT HDR_BRP_ID - (20) :  Load/Unload utility
    CORE_NUMERIC HDR_BRPFILE_VERSION - (1) : 4
    NONCORE_TEXT HDR_DBS_ID - (18) : Oracle Rdb V7.2-10
    NONCORE_TEXT HDR_DB_NAME - (16) : DB$:MF_PERSONNEL
    NONCORE_DATE HDR_DB_LOG_BACKUP_DATE - (8) :  3-JUL-2006 16:52:32.83
    CORE_NUMERIC HDR_DATA_COMPRESSION - (1) : 1
END HEADER SECTION - (0)
```

In this example, the output describes the creator of the interchange file (RMU/UNLOAD), the version of Rdb used to create the file, the file specification of the database used, the date and time the interchange file was created, and an indication that compression was used by RMU Unload.

1.24 RMU Dump Metadata_File Command

This command dumps the metadata file created by the RMU Unload After_ Journal command in a user readable format. The metadata file is created when using the /SAVE_METADATA qualifier of RMU Unload After_Journal command.

The metadata file can also be dumped if both the /RESTORE_METADATA and the /OPTIONS=DUMP qualifiers are specified with the RMU Unload After_ Journal command.

For complete information on the use and restrictions of the /SAVE_METADATA and /RESTORE_METADATA qualifiers, see the Oracle RMU Reference Manual section on the RMU Unload After_Journal command.

Format

RMU/Dump/Metadata_File metadata-file-name

Command Qualifiers	Defaults
/Header_Only	See Description
/Output = file-name	/Output=SYS$OUTPUT

Description

The RMU Dump Metadata_File command provides an independent command to dump the metadata file and optionally outputs just the metadata file header record which identifies the database root file and version and other general information about the metadata file dump without having to dump the entire metadata file.

The RMU Dump Metadata_File command specifies a .metadata file, not a database file, as its parameter, and is a separate command from the RMU Dump command used to display database areas and header information.

Note

Oracle Corporation reserves the right to change the format of the metadata file as required by future releases. The metadata file is intended to only be used internally by the RMU UNLOAD AFTER_ JOURNAL command. New metadata files should be generated to reflect changing metadata definitions in the database. Metadata files incompatible with the current database version or metadata file version

level will not be processed and will need to be recreated from the current database.

Command Parameters

metadata-file-name
The metadata file. The default file type is .metadata.

Command Qualifiers

Header_Only
If this qualifier is specified, only the first header record in the metadata file will be output. The default if this qualifier is not specified is to output all records in the metadata file.

Output=file-name
If this qualifier is specified, the metadata file will be dumped to the required named output file. The default file type for the output file is ".lis". The default if this qualifier is not specified is to output the metadata file records to the default output device (SYS$OUTPUT).

Usage Notes

- To use the RMU Dump Metadata_File command, you must have OpenVMS READ and READALL access to the metadata file to be dumped or be granted the OpenVMS SYSPRV or BYPASS privileges.

Examples

Example 1

The following example first creates the TEST_DATABASE.METADATA file from the database using the /SAVE_METADATA qualifier of the RMU/UNLOAD AFTER_JOURNAL command. The RMU/DUMP/METADATA_FILE/HEADER_ONLY command is used to dump only the metadata file header information to the default output device. The RMU/DUMP/METADATA_FILE/OUTPUT command is then used to save the entire dump of the TEST_DATABASE.METADATA file to the TEST_DATABASE.LIS file.

```
$ RMU/UNLOAD/AFTER_JOURNAL DEVICE:[DIRECTORY]TEST_DATABASE.RDB -
 /SAVE_METADATA=DEVICE:[DIRECTORY]TEST_DATABASE.METADATA/NOLOG
$ RMU/DUMP/METADATA_FILE/HEADER_ONLY DEVICE:[DIRECTORY]TEST_DATABASE.METADATA
*------------------------------------------------------------------------------
* Oracle Rdb V7.3-2                                    27-JUL-2015 16:02:56.29
*
* Dump of LogMiner Metadata File
*     Filename: DEVICE:[DIRECTORY]TEST_DATABASE.METADATA;1
*     Database: DEVICE:[DIRECTORY]TEST_DATABASE.RDB;1
*
*------------------------------------------------------------------------------

    Created  7-JUL-2015 15:33:05.91 by JONES
    Version 73.0 (Oracle Rdb V7.3-2) / Checksum 0043000D
    Metadata file version level is 101
    Database timestamp is 18-MAY-2015 12:01:05.75
    Logical Area Information Count is 280

$ RMU/DUMP/METADATA_FILE/OUTPUT=DEVICE:[DIRECTORY]TEST_DATABASE.LIS -
 DEVICE:[DIRECTORY]TEST_DATABASE.METADATA
$
```

1.25 RMU Dump Recovery_Journal Command

Displays a recovery-unit journal (.ruj) file in ASCII format. Use this command to examine the contents of an .ruj file. You might find .ruj files on your system following a system failure.

An .ruj file contains header information and data blocks. Header information describes the data blocks, which contain copies of data modified in the database file.

Format

RMU/Dump/Recovery_Journal ruj-file-name

Command Qualifiers	Defaults
/[No]Data	/Data
/Output = file-name	/Output=SYS$OUTPUT

Description

The RMU Dump Recovery_Journal command specifies an .ruj file, not a database file, as its parameter, and is a separate command from the RMU Dump command used to display database areas and header information.

The .ruj file is in binary format. This command translates the binary file into an ASCII display format.

Command Parameters

ruj-file-name
The .ruj file. The default file type is .ruj.

Command Qualifiers

Data
Nodata
Specifies whether you want to display data blocks of the .ruj file or just the .ruj file header.

The Data qualifier is the default. It causes the display of the .ruj file data blocks (in addition to the file header) in an ASCII display format.

The Nodata qualifier limits the display to the file header of the .ruj file.

Output=file-name
The name of the file where output will be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Usage Notes

- You do not need Oracle RMU privileges to use the RMU Dump Recovery_ Journal command. However, you must have OpenVMS READ access to the .ruj file or OpenVMS BYPASS privilege to use the RMU Dump Recovery_ Journal command.
- The RMU Dump Recovery_Journal command does not validate the file being dumped. If the file is not an .ruj file, the output from the RMU Dump Recovery_Journal command is unintelligible.
- See the *Oracle Rdb Guide to Database Maintenance* for examples showing the RMU Dump Recovery_Journal command.

1.26 RMU Dump Row_Cache Command

Allows you to display the in-memory contents of a row cache for an open database.

Format

RMU/Dump/Row_Cache root-file-spec

Command Qualifiers	Defaults
/Cache_Name=cachename	None
/[No]Data	/Data
/Output=file-name	/Output=SYS$OUTPUT

Description

The RMU Dump Row_Cache command is intended for use as a diagnostic aid that allows you to display the in-memory contents of a row cache for an open database. Use this command to display the following information for each row in the specified cache:

- GRIC - Address of the GRIC data structure for the cache slot
- GRIB - Address of the GRIB data structure for the cache slot
- SLOT - Slot number within the cache
- NXTGRIC - Slot number of the next slot within the hash chain
- LHMTE - Flag values indicating:
  - L - Row is latched
  - H - Row is marked Hot (modified since last checkpoint or sweep)
  - M - Row is marked Modified
  - T - Row is marked Too Big for (or removed from) the cache
  - E - End of on-disk checkpoint file; should never be seen with the RMU Dump Row_Cache command
- SNAPPNO - Snapshot pointer (either snapshot page number or snapshot slot number
- LEN - Length of the row in cache; 0 indicates row has been deleted

- ACTLEN - Actual length of allocated space on the database page for the row
- DBK - Database key for the row
- REFCNT - Reference count: number of processes with this row in a cache working set
- UPD_PID - Process ID of process currently updating the row in memory
- RVNO - In-memory row modification count
- TSN - Transaction sequence number of last transaction to modify the row

Command Parameters

root-file-spec
Specifies the database root file for which you want to dump the row_cache contents.

Command Qualifiers

Cache_Name=cachename
Specifies the name of the cache you want to dump. You must specify the cache name.

Data
Nodata
The Data qualifier specifies that the in-memory content of a row_cache is to be displayed in hexadecimal format along with the ASCII translation. The Data qualifier is the default.

Specify the Nodata qualifier to display only header information for each cache slot.

Output=filename
Specifies the name of the file where output is to be sent. The default is SYS$OUTPUT. If you specify a file name, the default output file type is .lis.

1.26 RMU Dump Row_Cache Command

Examples

Example 1

```
$ RMU/DUMP/ROW_CACHE /CACHE_NAME=test mf_personnel

*-------------------------------------------------------------------------------
* Oracle Rdb X7.1-00                                  21-OCT-2002  13:47:07.82
*
* Dump of Row Cache RDB$SYSTEM_CACHE
*     Database: DPA01:[V71]MF_PERSONNEL.RDB;2
*
*-------------------------------------------------------------------------------

GRIC     GRIB       SLOT  NXTGRIC LHMTE SNAPPNO   LEN ACTLN    DBK   REFCNT UPD_PID      RVNO   TSN

3F334580 3F36F400      0        0 _____       -1   430   430 38:2383:1     0 00000000        1  0:3968
3F334598 3F36F800      1        0 _____       -1   430   430 38:1671:0     0 00000000        1  0:3968
3F3345B0 3F36FC00      2       16 _____       -1   220   220 3:2023:2      0 00000000        1  0:3968
3F3345C8 3F370000      3        0 _____       -1   430   430 37:2359:1     0 00000000        1  0:3968
3F3345E0 3F370400      4        0 _____       -1   430   430 37:1665:0     0 00000000        1  0:3968
3F3345F8 3F370800      5        0 _____       -1   208   208 9:2007:0      0 00000000        1  0:3968
```

1.27 RMU Dump Symbols Command

Displays or writes to a specified output file the contents of database root information. The output is similar to that from RMU Dump Header except that the output is in the form of a DCL command procedure that defines global DCL symbols.

Format

RMU/Dump/Symbols root-file-spec

Command Qualifiers	Defaults
/Execute	NoExecute
/Output[=file-spec]	/Output=SYS$OUTPUT
/[No]Prefix[=text-string]	/Prefix=RMU$

Description

This command is designed for database administrators who wish to write DCL procedures that react to the current state of the Rdb database.

This command does not open the database and only provides access to the static on-disk database root file.

Command Parameters

root-file-spec
A file specification for the database root file whose root file header information you want to display.

Command Qualifiers

Execute
NoExecute
The file created by the /OUTPUT qualifier is executed before returning control to DCL. This allows immediate use of the defined global symbols. The default is NoExecute.

Output[=file-spec]
Specifies the name of the output command procedure created by RMU. The default is SYS$OUTPUT (which cannot be used by /EXECUTE).

Prefix=test-string
NoPrefix
By default, the generated symbol names start with RMU$. However, the database administrator can replace this with any text string. This would allow, for instance, two RMU Dump Symbol commands to be executed on different databases and the generated (unique) symbols compared.

If NoPrefix is used, then no prefix string is added to the created DCL symbols.

Examples

Example 1: Using RMU/DUMP/SYMBOLS

The following example shows a simple command procedure to get the CLIENT_FULL_BACKUP_TIMESTAMP for the named database and compute the delta time since it was last backed up.

```
$ v = 'f$verify(0)
$ set noon
$!
$! Check the last full backup date and see if the backup is past due
$!
$ temp_file = "temp" + f$getjpi(0,"PID") + ".tmp;"
$ RMU/DUMP/SYMBOL/EXECUTE/PREFIX=PERS_/OUTPUT=&TEMP_FILE SQL$DATABASE
$ delta_time = f$delta_time (PERS_CLIENT_FULL_BACKUP_TIMESTAMP,"TODAY")
$ days = f$integer(f$element(0," ",delta_time))
$ if days .gt. 7
$ then
$       alert_text = f$fao("Database PERS not backed up in !SL day!%S", days)
$       write sys$output alert_text
$       ! reply/username=DBADMIN "''alert_text'"
$ endif
$ delete &temp_file
$ exit ! 'f$verify(v)'
```

1.28 RMU Extract Command

Reads and decodes Oracle Rdb metadata and reconstructs equivalent statements in Relational Database Operator (RDO) or SQL (structured query language) code for the definition of that database. These statements can either be displayed or extracted. You can use these statements to create your database again if you no longer have the RDO or SQL code that defined your database.

In addition, you can direct the RMU Extract command to produce output for the following:

- An SQL or RDO IMPORT script (Items=Import)
- An RMU Unload command for each table (Items=Unload)
- An RMU Load command for each table (Items=Load)
- An RMU Set Audit command for the database (Items=Security)
- An RMU Verify command for the database (Items=Verify)

Format

RMU/Extract root-file-spec

Command Qualifiers	Defaults
/Defaults[=defaults-list]	/Defaults=(quoting_rules=SQL92)
/Items[=item-list]	/Items=All
/Language=lang-name	/Language=SQL
/[No]Log[=log-file]	/Nolog
/Options=options-list	/Option=Normal
/[No]Output[=out-file]	/Output=SYS$OUTPUT
/Transaction_Type[=(transaction_mode,options...])	See Description

Description

The RMU Extract command decodes information and reconstructs equivalent commands in the language you select with the Language qualifier for the definition of that database.

1.28 RMU Extract Command

You can extract the definitions to either a file or to SYS$OUTPUT. The RMU Extract command extracts the following character set information:

- For databases:
  - The database default character set
  - The national character set
- For domains:
  - The character set of each character data type domain
  - The length in characters of each character data type domain
- For tables:
  - The character set of each character data type column
  - The length in characters of each character data type column

The RMU Extract command may enclose object names in double quotation marks to preserve uppercase and lowercase characters, special character combinations, or reserved keywords.

The RMU Extract command is limited in many cases and output from the command may require manual editing. For example:

- The full SQL syntax is processed by SQL and an equivalent query or expression is stored in the database. In many cases, semantic information, not needed for correct execution, is discarded. Therefore, the regenerated expression may be slightly different than the original.
- When an original SQL definition is extracted with Language=RDO, it is probable that there is no matching RDO syntax for SQL features used by the definition. Extract will do its best to represent the view, constraint, trigger, computed by or default value.
- When an original RDO definition is extracted with Language=SQL, it is possible that there is no exact syntax in SQL for some features. For example, the REDUCE TO clause is far more general than the DISTINCT or GROUP BY clauses of SQL. Extract will do its best to represent the definition as a starting point for manual correction.

Command Parameters

root-file-spec

The file specification for the database root file from which you want to extract definitions. Note that you do not need to specify the file extension. If the database root file is not found, the command exits with a "file not found" error.

Command Qualifiers

Defaults[=defaults-list]
This qualifier is used to change the output of the RMU Extract command. The following defaults can be modified with the Defaults qualifier:

- Access_Mode
  NoAccess_Mode

  Using the ACCESS_MODE keyword for the DEFAULTS qualifier will output an ALTER DATABASE (CHANGE DATABASE for RDO) statement that modifies the access mode for all storage areas (except for RDB$SYSTEM). If the keyword is negated (NOACCESS_MODE), then the comment is not written to the SQL or RDO script. By default, when neither keyword is used, a comment is written to the script documenting the current access mode.

  Following is an example of using this keyword.

```
$       RMU/EXTRACT/ITEM=DATABASE -
            /DEFAULT=(ACCESS_MODE) -
            /OPTION=(NOHEADER,FILENAME_ONLY) -
            MF_PERSONNEL -
            /OUTPUT=DB.SQL
   .
   .
   .
```

- Allocation=integer
  Noallocation

  When you create a test database using the script generated by the RMU Extract command, the allocation from the source database may not be appropriate. You can use the Allocation keyword to specify an alternate value to be used by all storage areas, or you can use the Noallocation keyword to omit the clause from the CREATE STORAGE AREA clause. The default behavior, when neither keyword is used, is to use the allocation recorded in the database for each storage area. See also the Snapshot_Allocation keyword.

- Date_Format
  Nodate_Format

  By default, the RMU Extract process assumes that DATE types are SQL standard-compliant (that is DATE ANSI) and that the built-in function CURRENT_TIMESTAMP returns TIMESTAMP(2) values. If your environment uses DATE VMS exclusively, then you can modify the default by specifying the default DATE_FORMAT=VMS. The legal values are

described in the *Oracle Rdb SQL Reference Manual* in the SET DEFAULT DATE FORMAT section. The default is Date_Format=SQL92.

Use Nodate_Format to omit the setting of this session attribute from the script.

- Dialect
  Nodialect

  For some extracted SQL scripts the language dialect must be specified. You can use the Dialect keyword to supply a specified dialect for the script. You can find the legal values for this option in the Oracle Rdb SQL Reference Manual in the SET DIALECT section. The default is Nodialect.

- Language
  Nolanguage

  The RMU Extract commmand uses the process language, that is, the translated value of SYS$LANGUAGE, or ENGLISH, for the SET LANGUAGE command. However, if the script is used on a different system then this language might not be appropriate. You can use the Language keyword to supply a specified language for the script. Legal language names are defined by the OpenVMS system logical name table; examine the logical name SYS$LANGUAGES for a current set. Use the Nolanguage keyword to omit this command from the script.

- Quoting_Rules
  Noquoting_Rules

  You can use the Quoting_Rules keyword to supply a specified setting for the script. You can find the legal values for this option in the *Oracle Rdb SQL Reference Manual* in the SET QUOTING RULES section. The default is Quoting_Rules=SQL92. The RMU Extract command assumes that SQL keywords and names containing non-ASCII character set values are enclosed in quotation marks.

- Snapshot_Allocation=integer
  Nosnapshot_Allocation

  When you create a test database from the RMU Extract output, the snapshot file allocation from the source database may not be appropriate. You can use the Snapshot_Allocation keyword to specify an alternate value to be used by all snapshot areas, or you can use the Noallocation keyword to omit the "snapshot allocation is" clause. The default behavior, when neither keyword is used, is to use the snapshot allocation stored in the database for each snapshot area. See also the Allocation keyword.

Items[=item-list]
Allows you to extract and display selected definitions. Note that each of the item names can be combined to provide shorter command lines such as the following:

```
$ RMU/EXTRACT/NOLOG/ITEMS=(ALL,NODATABASE) MF_PERSONNEL
```

If you omit the Items qualifier from the command line or specify it without any options, the action defaults to Items=All.

The following options can be specified with the Items qualifier:

- All

  Indicates that all database items are to be extracted. This is the default and includes all items except Alter_Database, Forward_References, Import, Load, Protections, Revoke_Entry, Security, Synonyms, Unload, Verify, Volume, and Workload options. You can use either All or Noall in combination with other items to select specific output.

  In the following example, the Items=All option causes all the definitions except for Triggers to be extracted and displayed:

  ```
  $ RMU/EXTRACT/ITEMS=(ALL,NOTRIGGERS) MF_PERSONNEL
  ```

  The following example displays domain and table definitions. Note that the Noall option could have been omitted:

  ```
  $ RMU/EXTRACT/ITEMS=(NOALL, DOMAIN, TABLE) MF_PERSONNEL
  ```

- Alter_Database (or Change_Database)
  Noalter_Database

  Displays the physical database after-image journal object definition.

- Catalog
  Nocatalog

  Displays all contents of the catalog created for an SQL multischema database. This item is ignored if the interface is RDO.

- Collating_Sequences
  Nocollating_Sequences

  Displays all the collating sequences defined for the database that you select. Note that Oracle Rdb does not save the name of the source OpenVMS National Character Set (NCS) library and the name becomes the defined logical, NCS$LIBRARY, by default.

- Constraints
  Noconstraints

By default, table and column constraints are output by the Items=Table qualifier. If you specify Item=Noconstraints, constraint information is not extracted for any table. If you specify the Language=SQL qualifier, the default is to have Item=Constraints enabled when tables are extracted.

To extract all constraints as an ALTER TABLE statement, use the Item=Constraint and Option=Defer_Constraints qualifiers. To force all constraints to be defined after tables are defined, use the Item=Tables and Option=Defer_Constraints qualifiers.

- Database
  Nodatabase

  Displays the database attributes and characteristics. This includes information such as the database root file name, the number of buffers, the number of users, the repository path name, and the characteristics for each storage area.

  If you specify RMU Extract with the Option=Nodictionary_References qualifier, the data dictionary path name is ignored.

- Domains (or Fields)
  Nodomains

  Displays the domain definitions. If the domain was originally defined using the data dictionary path name, the output definition shows this. If the Option=Nodictionary_References qualifier is specified, the data dictionary path name is ignored and the column attributes are extracted from the system tables.

- Forward_References
  Noforward_References

  Queries the dependency information in the database (RDB$INTERRELATIONS) and extracts DECLARE FUNCTION and DECLARE PROCEDURE statements for only those routines that are referenced by other database objects. The default is Noforward_Reference.

  The Forward_References item is used in conjunction with other Item keywords, for example, `/Item=(All,Forward)`.

- Functions
  Nofunctions

  Displays external function definitions.

- Import
  Noimport

Generates an RDO or SQL IMPORT script that defines every storage area and row cache. The Language qualifier determines whether Oracle RMU generates an RDO or SQL IMPORT script (If you specify the Language=SQL or the Language=ANSI_SQL qualifier, the same SQL IMPORT script is generated.) Because the RDO interface does not accept many of the database options added to recent versions of Oracle Rdb, Oracle Corporation recommends that you specify the Language=SQL qualifier (or accept the default).

The Items=Import qualifier is useful when you want to re-create a database that is the same or similar to an existing database. Editing the file generated by Oracle RMU to change allocation parameters or add storage areas and so on is easier than writing your own IMPORT script from scratch.

When Oracle RMU generates the IMPORT script, it uses an interchange file name of rmuextract_rbr in the script. Therefore, you must either edit the IMPORT script generated by Oracle RMU to specify the interchange file that you want to import, or assign the logical name RMUEXTRACT_RBR to your interchange file name. (An interchange file is created by an SQL or RDO EXPORT statement.) See Example 14 in the Examples section.

- Indexes (or Indices)
  Noindexes

  Displays index definitions, including storage map information.

- Load
  Unload

  Generates a DCL command procedure containing an RMU Load or RMU Unload command for each table in the database. This item must be specified explicitly, and is not included by default when you use the Items=All qualifier.

  Oracle RMU generates the Fields qualifier for the Load and Unload scripts when you specify the Option=Full qualifier. If you do not specify the Option=Full qualifier, the scripts are generated without the Fields qualifier.

  If you specify the RMU Extract command with the Item=Unload qualifier, DCL commands are added to the script to create a file with type .COLUMNS. This file defines all the unloaded columns. The file name of the .COLUMNS file is derived from the name of the extracted table. You can reference the file by using the "@" operator within the Fields qualifer for the RMU Load and RMU Unload commands.

  Virtual columns, AUTOMATIC or COMPUTED BY table columns, and VIEW calculated columns appear in the .COLUMNS file as comments.

1.28 RMU Extract Command

- Module
  Nomodule

  Displays procedure and function definitions. This item is valid only when the Language specification is SQL; it is ignored if the Language specification is RDO or ANSI_SQL.

- Outlines
  Nooutlines

  Displays query outline definitions. This item is valid only when the Language specification is SQL; it is ignored if the Language specification is RDO or ANSI_SQL.

- Procedures
  Noprocedures

  Extracts external procedures.

- Profiles
  Noprofiles

  Displays profiles as defined by the CREATE PROFILE statement.

- Protections
  Noprotections

  Displays the access control list (ACL) definitions as a series of GRANT statements. You must explicitly specify the Protections option as it is not included by default when you use the Items=All qualifier.

- Recompile

  This keyword causes RMU to generate a series of ALTER MODULE commands to recompile any invalid routines. The default is to only include those modules that have at least one invalid routine. Use Option=FULL to generate a script that compiles all modules.

  The following example shows the generated SQL script for a database with routines that were marked invalid. RMU annotates the generated ALTER MODULE statement with the names of those invalid routines.

```
$ rmu/extract/option=(noheader,filename_only) abc /item=recompile
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename EXAMPLE';
```

```
alter module JACKET compile;
-- Invalid Procedure GU
-- Invalid Procedure GBS
-- Invalid Function F_GBS_DIRECT
commit work;
$
```

- Revoke_Entry
  Norevoke_Entry

  Extracts a SQL or RDO script that deletes the protections from all access control lists in the database: database, table, sequences, column, module, function, and procedure.

  The output script contains a series of SQL REVOKE ENTRY statements (or DELETE PROTECTION statements if the language selected is RDO) that remove the access control entry for the user and all objects.

- Role
  Norole

  Displays role definitions as defined by the SQL CREATE ROLE statement. In addition, any roles that have been granted are displayed as a GRANT statement. By default, roles are not extracted, nor are they included when you specify the Items=All qualifier.

- Schema
  Noschema

  Displays the schema definitions for an SQL multischema database. This option is ignored if the interface is RDO.

- Sequence
  Nosequence

  Displays the sequence definitions in the database that were originally defined with the SQL CREATE SEQUENCE statement.

- Security
  Nosecurity

  Displays RMU Audit commands based on information in the database. This item must be specified explicitly, and is not included by default when you use the Items=All qualifier.

- Storage_Maps
  Nostorage_Maps

  Displays storage map definitions, including the list (segmented string) storage map.

- Synonyms
  Nosynonyms

  Generates a report of all the synonyms defined for the database. All synonyms of a database object, including synonyms of those synonyms, are grouped together. The output is ordered by creation date and time as recorded by the RDB$CREATED column of the RDB$OBJECT_ SYNONYMS system table.

  This report is useful for viewing all synonyms or moving them to other databases. However, since synonyms refer to many different database objects, a single set of definitions is usually not adequate when defining a new database. Oracle Corporation recommends that you use the Option=Synonym qualifier in most cases.

- Tables (or Relations)
  Notables

  Displays table definitions in the same order in which they were created in the database.

  If the table was originally defined using the data dictionary path name, that path name is used for the definition.

  If you specify the Option=Nodictionary_References qualifier, the data dictionary path name is ignored and the table attributes are extracted from the system tables.

  If Item=Noconstraints is specified, constraint information is not extracted for any table.

  The Items=Tables qualifier handles domains in the following ways:

  – The output for this item reflects the original definitions. If a column is based on a domain of a different name, the BASED ON clause is used in RDO, and the domain name is referenced by SQL.

  – Any columns that are based on fields in a system table are processed but generate warning messages.

  – Certain domains created using RDO in a relation definition cannot be extracted for RDO because it is not possible to distinguish columns defined using a shorthand method as shown in the example that follows. In this case, the column FIELD_1 becomes or is defined as a domain.

```
DEFINE RELATION REL1.
    FIELD_1     DATATYPE IS TEXT SIZE 10.
END.
```

However, this type of definition in SQL causes special domains to be created with names starting with SQL$. In this case, the SQL domain is translated into the following data type:

```
CREATE TABLE TAB1
   (COLUMN_1   CHAR(10));
```

The output for this item also includes the table-level constraints that can be applied: PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, and CHECK. In the case of the CHECK constraint, the expression might not be translated to or from RDO and SQL due to interface differences.

- Triggers
  Notriggers

  Displays trigger definitions.

- User
  Nouser

  Displays user definitions as defined by the SQL CREATE USER statement. In addition, if you also specify Role with the Item qualifier, then any roles that have been granted to a user are displayed as GRANT statements. By default, Users are not displayed, nor are they displayed when you specify the Items=All qualifier.

- Verify
  Noverify

  Causes the generation of an optimal DCL command procedure containing multiple RMU Verify commands. Using this command procedure is equivalent to performing a full verification (RMU Verify with the All qualifier) for the database. This command procedure can be broken down further into partial command scripts to perform partial verify operations. These partial command scripts can then be submitted to different batch queues to do a full verify operation in parallel, or they can be used to spread out a full verify operation over several days by verifying a piece of the database at a time.

  A **partitioning algorithm** is a procedure to determine what portions of the database should be verified in the same command script. For example, areas with interrelations should be verified with the same partial command script. A partitioning algorithm considers the following when creating a partial command script from the equivalent RMU Verify command with the All qualifier:

  1. Each storage area is assigned to a partition.

2. For each table in the database, if the table is not partitioned, the table is put in the partial command script corresponding to that storage area; otherwise, if the table is partitioned across several storage areas, the partitions corresponding to all of the storage areas are merged into one partial command script and the table is added to this partial command script.
3. For each index in the database, the process shown in step 2 is followed.
4. For an index on a table, the index and table are merged into one partial command script.

The scripts of partial RMU Verify commands are written in the form of a command procedure. Each partial command script is preceded by a label of the form STREAM_n: where *n* is an integer greater than 1. For example, to execute the command at label STREAM_3:, invoke the command procedure by using the following syntax:

```
$ @<command-procedure-name> STREAM_3
```

The resultant command procedure is set up to accept up to four parameters, P1, P2, P3, and P4, as shown in Table 1–10.

Table 1–10 Parameters for Generated Command File

Parameter	Option	Description
P1	Stream_n	Specifies the command stream to be executed. The variable *n* is the "number" of the RMU Verify command stream to be executed. If omitted, all command streams are executed.
P2	[No]Log	Specifies whether to use the Log qualifier in the RMU Verify command line. If omitted, the DCL verify switch value is used.
P3	Read_Only | Protected | Exclusive	Provides the RMU Verify Transaction_Type value. If omitted, Transaction_Type = Protected is used.

(continued on next page)

Table 1–10 (Cont.) Parameters for Generated Command File

Parameter	Option	Description
P4		Specifies the name of the output file for the RMU Verify Output qualifier. If omitted, Output = SYS$OUTPUT is used.

- Views
  Noviews

  Displays view definitions. If the database was defined using SQL, it is possible that the view cannot be represented in RDO. In this case, the diagnostic message warns that the view definition is being ignored, and the user should use LANGUAGE=SQL to extract the view. Note the following transformations the RMU Extract command makes when it cannot precisely replicate the SQL source code:

  – The RMU Extract command cannot precisely replicate derived table column names or correlation names for any select expression.

    The RMU Extract command generates new names for correlation names (C followed by a number) and derived table column names (F followed by a number).

    For example, suppose you create a view, as follows:

```
SQL> ATTACH 'FILENAME mf_personnel';
SQL> CREATE VIEW DERIVED_1
cont> (F1) AS
cont> SELECT CAST(AVG(JOB_COUNT) AS INTEGER(2))
cont> FROM (SELECT EMPLOYEE_ID, COUNT (*)
cont>   FROM JOB_HISTORY
cont>   GROUP BY EMPLOYEE_ID) AS EMP_JOBS (EMPLOYEE_ID, JOB_COUNT);
SQL> COMMIT;
```

    If you issue the following RMU Extract command, you receive the output shown:

1.28 RMU Extract Command

```
$ RMU/EXTRACT/ITEM=VIEW/OPTION=(MATCH:DERIVED_1%,NOHEADER,FILENAME_ONLY) -
  MF_PERSONNEL
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename MF_PERSONNEL';
create view DERIVED_1
    (F1) as
    (select
        CAST(avg(C2.F2) AS INTEGER(2))
    from
        (select C4.EMPLOYEE_ID, count(*)
            from JOB_HISTORY C4
            group by C4.EMPLOYEE_ID)
        as C2 (F1, F2));

commit work;
```

– The RMU Extract command cannot generate the original SQL source code for the user-supplied names of AS clauses. This is particularly apparent when the renamed select expression is referenced in an ORDER BY clause.

```
SQL> create database filename XYZ;
SQL> create table DOCUMENT
cont>     (report CHAR(10));
SQL> create table REPORTING
cont>     (name CHAR(5));
SQL> create table TABLES
cont>     (codtab CHAR(5));
SQL> create view VIEW_TEST
cont>     (credit,
cont>     codtab,
cont>     codmon) as
cont> select
cont>     rp.name,
cont>     tb.codtab,
cont>     (select report from document) as com
cont> from reporting rp, Tables tb
cont> order by rp.name asc, tb.codtab asc, com asc;
SQL> commit;
```

As can be seen in the generated output, the following equivalent view definition does not use the same SQL syntax.

```
$ RMU/EXTRACT/ITEM=VIEW/OPTION=(NOHEADER,FILENAME_ONLY) XYZ
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename XYZ';
create view VIEW_TEST
    (CREDIT,
     CODTAB,
     CODMON) as
    (select
        C1."NAME",
        C2.CODTAB,
            (select C4.REPORT from DOCUMENT C4)
    from REPORTING C1, "TABLES" C2
    order by 1 asc, 2 asc, (select C3.REPORT from DOCUMENT C3) asc);

commit work;
```

- Volume
  Novolume

  Displays cardinality information in a PDL-formatted file for use by Oracle Expert for Rdb. This item must be specified explicitly, and is not included by default when the Items=All qualifier is used.

- Workload
  Noworkload

  Generates a DCL command language script. The script is used with the RMU Insert Optimizer_Statistics command to extract the work load and statistics stored in the RDB$WORKLOAD table. The unloaded information can be applied after a new database is created using the SQL EXPORT and IMPORT statements, or it can be applied to a similar database for use by the RMU Collect Optimizer_Statistics/Statistic=Workload command.

  This item must be specified explicitly, and is not included by default when the Items=All qualifier is used. The default is Noworkload.

  You can modify the output of the Item=Workload qualifier by specifying the following keywords with the Option qualifier:

  - Audit_Comment

    Each RMU Insert Optimizer_Statistics statement is preceded by the created and altered date for the workload entry. The default is Noaudit_comment.

  - Filename_Only

The database file specification output for the RMU Insert Optimizer_ Statistics statement is abbreviated to just the filename.

- Match

  A subset of the workload entries based on the wildcard file name is selected.

Language=lang-name

Allows you to select one of the following interfaces:

- SQL

  When you specify the Language=SQL qualifier, Oracle RMU generates the Oracle Rdb SQL dialect. The Oracle Rdb SQL dialect is a superset of SQL92 Entry level, with language elements from Intermediate and Full SQL92 levels. It also contains language elements from SQL:1999 and extensions specific to Oracle Rdb.

- ANSI_SQL

  When you specify the Language=ANSI_SQL qualifier and specify the Option=Normal qualifier, Oracle RMU tries to generate ANSI SQL statements that conform to the ANSI X3.135-1989 SQL standard.

  When you specify the Language=ANSI_SQL qualifier and the Option=Full qualifier, Oracle RMU tries to generate SQL statements that conform to the current ANSI and ISO SQL database language standards. Refer to the *Oracle Rdb SQL Reference Manual* for more information.

  Regardless of the Option parameter you specify, any Oracle Rdb specific features (such as DATATRIEVE support clauses and storage maps) are omitted.

- RDO

  When you specify the RDO language option, Oracle RMU generates RDO statements.

- CC

  For the C language, you must specify CC to avoid ambiguity with an abbreviated COBOL language.

  The output for C is a **typedef** with the columns of the table and named with the table name. This definition is designed to be used by the SQL Precompiler and so includes pseudo data types and CHARACTER SET clauses to establish the correct semantics for the precompiler.

The following example shows the output for the WORK_STATUS table. Note that, by default, RMU Extract assumes null terminated strings and adds one to the length. This can be disabled by specifying NONULL_ TERMINATED for the Option qualifier.

Applications would then declare a variable using this typedef definition as shown in the following code fragment.

```
$ rmu/extract/item=table /lang=cc sql$database/option=(noheader,
match:work_status,audit)

/* Table: WORK_STATUS (null terminated)
// Created on 11-JUL-2018 17:39:59.20
// Last altered on 11-JUL-2018 17:39:59.21
// Created by HR_SERVICES
//
*/

#ifndef _WORK_STATUS_
#define _WORK_STATUS_
typedef struct {
    char STATUS_CODE[2];
    char STATUS_NAME[9];
    char STATUS_TYPE[15];
    } WORK_STATUS;
#endif  // WORK_STATUS
 . . .
exec sql
  include 'work_status.h';

WORK_STATUS work_status_rec;

exec sql
    fetch work_status_cursor into :work_status_rec;
```

Note that the generated text file must be included by the EXEC SQL INCLUDE or EXEC SQL INCLUDE MODULE ... FROM LIBRARY statements so that the definition is visible to the SQL Precompiler.

- COBOL

  The output for COBOL is a 01 data division definition with the columns of the table and named with the table name. VARCHAR and VARBINARY columns are represented by level 49 field with two subfields to represent the length (LEN) and body (VAL) of the string.

  This definition is designed to be used by the SQL Precompiler and so includes pseudo data types and CHARACTER SET clauses to establish the correct semantics for the precompiler.

The following example shows the representation of a VARCHAR column.

```
$ rmu/extract -
    /item=table -
    /language=cobol -
    PERSONNEL -
    /option=(noheader,audit,match:candidates)
 . . .
** Table: CANDIDATES
** Created on 11-JUL-2018 17:39:59.03
** Never altered
** Created by HR_SERVICES
**

01 CANDIDATES.
    05 LAST_NAME                      picture X(14).
    05 FIRST_NAME                     picture X(10).
    05 MIDDLE_INITIAL                 picture X(1).
    05 CANDIDATE_STATUS               .
        49 LEN                        picture S9(4) comp.
        49 VAL                        picture X(255).
```

Note that the generated text file must be included by the EXEC SQL INCLUDE or EXEC SQL INCLUDE MODULE ... FROM LIBRARY statements so that the definition is visible to the SQL Precompiler.

- Pascal

  The output for Pascal is a **type** with the columns of the table and named with the table name. This definition is designed to be used by the SQL Precompiler and so includes pseudo data types and CHARACTER SET clauses to establish the correct semantics for the precompiler.

  Applications would then declare a variable (VAR) using this type definition and shown in the following code fragment.

```
$ rmu/extract/item=table -
    /lang=Pascal -
    /output=work_status.p -
    sql$database -
    /option=(noheader,match:work_status,audit)
   .
   .
   .
exec sql
    include 'work_status.p';

var
    work_status_rec : work_status := ZERO;
   .
   .
   .
```

Note that the generated text file must be included by the EXEC SQL INCLUDE or EXEC SQL INCLUDE MODULE ... FROM LIBRARY statements so that the definition is visible to the SQL Precompiler.

The default is Language=SQL.

The Language qualifier has no effect on the output generated by the Items=Load, Items=Unload, and Items=Verify qualifiers. This is because these qualifiers generate scripts that contain Oracle RMU commands only.

Log[=log-file]
Nolog
Enable or disables log output during execution of the RMU Extract command. The log includes the current version number of Oracle Rdb, and the values of the parameter and qualifiers. The default is Nolog. The default file extension is .log. If you specify Log without specifying a file name, output is sent to SYS$OUTPUT.

Options=options-list
This qualifier is used to change the output of the RMU Extract command. This qualifier is not applied to output created by the Items=Unload, Items=Load, Items=Security, or the Items=Verify qualifier.

The following options can be specified with the Options qualifier:

- Audit_Comment
  Noaudit_Comment

  Annotates the extracted objects with the creation and last altered timestamps as well as the username of the creator. The date and time values are displayed using the current settings of SYS$LANGUAGE and LIB$DT_FORMAT. Noaudit_Comment is the default.

- Cdd_Constraints
  Nocdd_Constraints

  Specifies that tables extracted by pathname include all constraints. The Option=Nocdd_Constraints qualifier is equivalent to the Option=Defer_Constraints qualifier for tables with a pathname. This option is ignored if Item=Noconstraints is specified.

  When you specify the Cdd_Constraints option and the Dictionary_References option, the RMU Extract command does not generate ALTER TABLE statements to add constraints, but instead assumes they will be inherited from the data dictionary.

When you use the Nocdd_Constraints option and the Dictionary_References option, the RMU Extract command generates ALTER TABLE statements to add FOREIGN KEY and CHECK constraints after all base tables have been created.

- Cdd_References
  Nocdd_References

  This option is an alias for Dictionary_References.

- Column_Volume
  Nocolumn_Volume

  Directs the RMU Extract command to output the table, column, and column segmented string cardinalities based on sorted indexes. Note that this qualifier must be used in combination with the Items=Volume qualifier. If the Items=Volume qualifier is omitted, cardinalities are not displayed.

  RMU Extract generates data of the following type:

```
Volume for schema MF_PERSONNEL
    Default volatility is 5;
    Table WORK_STATUS all is 3;
    Table EMPLOYEES all is 100;
        Column EMPLOYEE_ID all is 100;
        Column LAST_NAME all is 83;
   .
   .
   .
    Table RESUMES all is 3;
        List RESUME
            Cardinality IS 3
                Number of segments is 3
                Average length of segments is 24;
```

- Debug
  Nodebug

  Dumps the internal representation for SQL clauses such as AUTOMATIC AS, VALID IF, COMPUTED BY, MISSING_VALUE, DEFAULT_VALUE, CONSTRAINTS, SQL DEFAULT, TRIGGERS, VIEWS, and STORAGE MAPS during processing. The keyword Debug cannot be specified with the keywords Normal or Full in the same Options qualifier list.

- Defer_Constraints
  Nodefer_Constraints

  Forces all constraints to be defined (using an ALTER TABLE statement) after all tables have been extracted. This option is ignored if Item=Noconstraints is specified.

If Option=Nodefer_Constraints is specified, all constraints are generated with the CREATE TABLE statement. If neither Option=Defer_Constraints nor Option=Nodefer_Constraints is specified, the default behavior is to generate NOT NULL, UNIQUE, and PRIMARY KEY constraints with the CREATE TABLE statement, and generate CHECK and FOREIGN KEY constraints in a subsequent ALTER TABLE statement.

- Dictionary_References
  Nodictionary_References

  Directs the RMU Extract command to output definitions for domains (fields) and tables (relations) that reference data dictionary path names rather than using the information contained in the Oracle Rdb system tables. In addition to the database statements, this option also displays the data dictionary path name stored in the database. Refer to Example 8 in the Examples section for an example of using this option.

  If neither the Option=Dictionary_References qualifier nor the Option=Nodictionary_References qualifier is specified, then Oracle RMU examines the RDB$RELATIONS and RDB$FIELDS system tables to determine whether or not any domains or tables refer to the data dictionary. If references are made to the data dictionary, then the Option=Dictionary_References qualifier is the default. Otherwise, it is assumed that the data dictionary is not used, and the default is the Option=Nodictionary_References qualifier.

  The Nodictionary_References keyword causes all references to the data dictionary to be omitted from the output. This is desirable if the database definition is to be used on a system without the data dictionary or in a testing environment.

  If the Items=Database and Option=Nodictionary_References qualifiers are selected, the data dictionary path name stored in the system table is ignored. For SQL, the NO PATHNAME clause is generated, and for RDO, the clause DICTIONARY IS NOT USED is generated.

  If the Items qualifier specifies Domain or Table, and the Option qualifier specifies Nodictionary_References, the output definition includes all attributes stored in the system tables.

- Disable_Objects
  Nodisable_Objects

  Requests that all disabled objects be written to the RMU Extract output file as disabled (see the description for the Omit_Disabled option). Disable_Objects is the default.

  The Nodisable_Objects option displays the objects but omits the disabling syntax.

- Domains
  Nodomains

  The Nodomains option is used to eliminate the domain name from within metadata objects. The domain name is replaced by the underlying data type. This option is designed for use with tools that do not recognize this SQL:1999 SQL language feature.

  Effect on /Language=SQL output:

  The default is Option=Domains.

  A SQL script generated when Option=Nodomains was specified does not include the domain name in the CREATE TABLE column definition, CREATE FUNCTION or CREATE PROCEDURE parameter definitions, or any value expression which uses the CAST function to convert an expression to a domain data type (such as the CREATE VIEW and CREATE TRIGGER statements).

  The output generated by the RMU Extract command for functions and procedures in the CREATE MODULE statement is not affected by the Option=Nodomains option because it is based on the original source SQL for the routine body which is not edited by the RMU Extract command.

  Effect on /Language=ANSI_SQL output:

  The default is Option=Nodomains when the Option=Normal qualifier is specified or is the default. The RMU Extract command does not generate a list of domain definitions even if the Items=Domains or Items=All qualifier is used. If you want the generated script to include a list of domain definitions, use the Options=Domains qualifier:

  ```
  $RMU/EXTRACT/LANGUAGE=ANSI_SQL/OPTION=DOMAINS databasename
  ```

  Use the Option=Full qualifier to have the use of domains included in the syntax generated for SQL:1999.

- Filename_Only
  Nofilename_Only

  Causes all file specifications extracted from the database to be truncated to only the file name. The use of this qualifier allows for easier relocation of the new database when you execute the created procedure.

- Full
  Nofull

Specifies that if metadata that cannot be translated from the language that defined the database to the equivalent construct in the language specified with the Language qualifier (for example, DEFAULT for SQL and the language selected was RDO) then the metadata is displayed in comments, or Oracle RMU attempts to create a translation that most closely approximates the original construct.

Nofull is identical to the Normal option.

- Group_Table
  Nogroup_Table

  Specifies that indexes and storage maps are to be extracted and grouped by table. The table is extracted first, than any PLACEMENT VIA index, then any storage map, and finally all other indexes.

  When the Group_Table qualifier is combined with the Option=Match qualifier, you can select a table and its related storage map and indexes.

  The default behavior is Nogroup_Table, which means that items are extracted and grouped by type.

- Header
  Noheader

  Specifies that the script header and section headers are included in the extract. This is the default. Because the header has an included date, specifying Noheader to suppress the header may allow easier comparison with other database extractions when you use the OpenVMS DIFFERENCES command.

- Hidden_Objects
  NoHidden_Objects

  The RMU Extract command will include special objects created by OCI Services for Rdb. These data dictionary objects (domains, functions, modules, procedures, tables and views) are marked as hidden by Rdb and are not usually visible. However, RMU Extract includes them in the output SQL script, often making the output confusing.

  The option Hidden_Objects can be used to display these definitions if desired. The option NoHidden_Objects can be used to hide these objects. The default is NoHidden_Objects.

- Journals
  NoJournals

  By default, /ITEM=ALTER_DATABASE will extract the after image journal settings and definitions for the database. Using /OPTION=NOJOURNALS will prevent that output.

The journal name is filtered based on the /OPTION=MATCH qualifier, otherwise all journals are used.

- Limit_Volume=nn
  Nolimit_Volume

  Specifies the maximum amount of data to be scanned for segmented fields. The RMU Extract command stops scanning when the limit *nn* is reached. The number of segments and average length of segments are calculated from the data that was scanned. Limit_Volume=1000 is the default.

  Nolimit_Volume specifies that a full scan for segmented strings should be done.

- Match:match-string

  The Match option allows selection of wildcard object names from the database. The match string can contain the standard SQL wildcard characters: the percent sign (%) to match any number of characters, and the underscore (_) to match a single character. In addition, the backslash (\) can be used to prefix these wildcards to prevent them from being used in matching. If you are matching a literal backslash, use the backslash twice, as shown in the following example:

```
Option=Match:"A1\\A2%"
```

  The match string defaults to the percent sign (%) so that all objects are selected. To select those objects that start with JOB, use the qualifier Option=Match:"JOB%".

  From the mf_personnel database, this command displays the definitions for the domains JOB_CODE_DOM and JOB_TITLE_DOM, the tables JOBS and JOB_HISTORY, the index JOB_HISTORY_HASH, and the storage maps JOBS_MAP and JOB_HISTORY_MAP.

  The match string can be quoted as shown if the string contains spaces or other punctuation characters used by DCL.

  The Match option can be used in conjunction with the Item qualifier to extract specific tables, indexes, and so on, based on their name and type.

  If Group_Table is specified, the match name is assumed to match a table name; all indexes for that table will be extracted when the Items=Index qualifier is specified.

- Multischema
  Nomultischema

  Displays the SQL multischema names of database objects. It is ignored by the Relational Database Operator (RDO).

The Nomultischema option displays only the SQL single-schema names of database objects.

- Normal
  Nonormal

  Includes only the specific source language code used to define the database. This is the default.

  In addition, this option propagates RDO VALID IF clauses as column CHECK constraints with the attribute NOT DEFERRABLE when the Language specification is SQL or ANSI_SQL. When an RDO VALID IF clause is converted, Oracle RMU generates error messages similar to the following in your log file:

```
%RMU-W-UNSVALIDIF, VALID IF clause not supported in SQL - ignored
 for DEGREE.
%RMU-I-COLVALIDIF, changed VALID IF clause on domain DEGREE to
 column check constraint for DEGREES.DEGREE
```

  The first message is a warning that the VALID IF clause could not be added to the domain definition because the VALID IF clause is not supported by SQL. The second message is an informational message that tells you the VALID IF clause was changed to a column check constraint.

- Omit_Disabled
  Noomit_Disabled

  Causes all disabled objects to be omitted from the output of the RMU Extract command. This includes indexes that have MAINTENANCE IS DISABLED, USERS with ACCOUNT LOCK, and disabled triggers and constraints.

  The Noomit_Disabled option causes all disabled objects to be included in the output from the RMU Extract command. Noomit_Disabled is the default.

- Order_By_Name
  Noorder_By_Name

  Order_by_Name displays the storage area, cache, and journal names for the items Database, Alter_Database (also known as Change_Database), and Import in alphabetic order by the ASCII collating sequence.

  Noorder_By_Name displays the storage area, cache, and journal names for the items Database, Alter_Database, and Import in approximate definition order. The default ordering is approximate because a DROP STORAGE AREA, DROP CACHE, or DROP JOURNAL statement frees a slot that can be reused, thus changing the order. Noorder_By_Name is the default.

You can use the logical name RDMS$BIND_SORT_WORKFILES to allocate work files, if needed.

- Position_Column
  NoPosition_Column

  This option will order columns by position instead of the default field identification. This means that the affects of AFTER COLUMN and BEFORE COLUMN clauses will be applied. However, such definitions may not represent a legal table definition if columns are referenced before they are defined.

  The default option is NOPOSITION_COLUMN which will reconstruct the table using, if necessary, an ALTER TABLE statement to apply the affects of AFTER COLUMN and BEFORE COLUMN clauses.

- Row_Caches
  NoRow_Caches

  By default, /ITEM=DATABASE or /ITEM=IMPORT will include the row cache definitions for the database. Using /OPTION=NOROW_CACHES will prevent that output.

  Using /OPTION=ROW_CACHES with /ITEM=ALTER_DATABASE will include the cache definitions as ADD CACHE clauses.

  The row cache name is filtered based on the /OPTION=MATCH qualifier, otherwise all row caches are used.

- Storage_Areas
  NoStorage_Areas

  By default, /ITEM=DATABASE or /ITEM=IMPORT will include the storage area definitions for the database. Using /OPTION=NOSTORAGE_AREAS will prevent that output.

  Using /OPTION=STORAGE_AREAS with /ITEM=ALTER_DATABASE will include the storage area definitions as ADD STORAGE AREA clauses.

  The storage area name is filtered based on the /OPTION=MATCH qualifier, otherwise all storage areas are used.

- Synonyms
  Nosynonyms

  Causes the synonyms to be extracted immediately after the referenced object, as shown in the following excerpt from an output file created using the Item=Table qualifier:

```
create table HISTORICAL_JOB_INFORMATION (
       EMPLOYEE_ID
           INTEGER,
       USER_ID
           CHAR (15),
       JOB_TITLE TITLE,
       START_DATE
           DATE,
       CURRENT_SALARY MONEY_IN_DOLLARS
           default NULL);
   create synonym JOBHIST
       for table HISTORICAL_JOB_INFORMATION;
```

Because synonyms can be referenced from almost any database object, if you keep the definitions close to the target object you can eliminate occurrences of undefined symbols during script execution. The default is Option=Synonyms.

Use the Option=Nosynonyms qualifier to disable the display of CREATE SYNONYM statements. The synonyms referenced in database objects such as module, procedure, trigger, and table definitions are still extracted.

- Volume_Scan
  Novolume_scan

  Directs the RMU Extract command to perform queries to calculate the cardinality of each table, if both the Items=Volume and Options=Volume_Scan qualifiers are specified. The default is Options=Novolume_Scan, in which case the approximate cardinalities are read from the RDB$RELATIONS system table. The Options=Volume_Scan option is ignored if the Items=Volume qualifier is not selected.

- Width=n

  Specifies the width of the output files. You can select values from 60 to 512 characters. The default of 80 characters is appropriate for most applications.

Output=[out-file]
Nooutput

Names the file to which the RMU Extract command writes the data definition language (DDL) statements. The file extension defaults to .rdo, if you specify the Language=RDO qualifier; .sql, if you specify either the Language=SQL or the Language=ANSI_SQL qualifier. If you specify the Volume option only, the output file type defaults to .pdl. If you specify Load, Security, Verify, or Unload only, the output file type defaults to .com. The default is SYS$OUTPUT. If you disable the output by using the Nooutput qualifier, command scripts are not written to an output file. The Log output can be used to determine which features used by the database cannot be converted to SQL.

1.28 RMU Extract Command

Table 1–11 shows the effects of the various combinations of the Language and Options qualifiers.

Table 1–11 Using Qualifiers to Determine Output Selection

Language	Option	Effect on Output
RDO	Normal	Generates RDO syntax.
	Full	Generates RDO syntax.
	Dictionary_ References	Outputs path name references to the repository.
	Nodictionary_ References	Converts path name references to the repository to RDO syntax.
	Multischema	Ignored by RDO.
SQL	Normal	Generates SQL syntax.
	Full	Tries to convert RDO specific features to SQL (for example, the VALID IF clause).
	Dictionary_ References	Outputs path name references to the data dictionary.
	Nodictionary_ References	Converts path name references to the data dictionary to SQL syntax.
	Multischema	Selects SQL multischema naming of objects.
ANSI_SQL	Normal	Generates ANSI/ISO syntax.
	Full	Generates ANSI/ISO SQL92 syntax supported by SQL.
	Dictionary_ References	Ignored for ANSI_SQL.
	Nodictionary_ References	Converts path name references to the data dictionary to SQL syntax. This is the default for ANSI_SQL.

(continued on next page)

Table 1–11 (Cont.) Using Qualifiers to Determine Output Selection

Language	Option	Effect on Output
	Multischema	Selects SQL multischema naming of objects.
Any	Audit_Comment	Adds a comment before each definition.
	Debug	Annotates output where possible.
	Domains	Replaces domain names for CAST expression, column and parameter definitions, and returns clauses with SQL data type.
	Filename_Only	Truncates all file specifications extracted from the database to only the file name.
	Volume_Scan	Forces a true count of Tables. Only valid for Items=Volume.

Transaction_Type[=(transaction_mode,options,...)]
Allows you to specify the transaction mode, isolation level, and wait behavior for transactions.

Use one of the following keywords to control the transaction mode:

- Automatic

  When Transaction_Type=Automatic is specified, the transaction type depends on the current database settings for snapshots (enabled, deferred, or disabled), transaction modes available to the process, and the standby status of the database. Automatic mode is the default.

- Read_Only

  Starts a READ ONLY transaction.

- Write

  Starts a READ WRITE transaction.

Use one of the following options with the keyword Isolation_Level=[level] to specify the transaction isolation level:

- Read_Committed

- Repeatable_Read
- Serializable. Serializable is the default setting.

Refer to the SET TRANSACTION statement in the Oracle Rdb SQL Reference Manual for a complete description of the transaction isolation levels.

Specify the wait setting by using one of the following keywords:

- Wait

  Waits indefinitely for a locked resource to become available. Wait is the default behavior.

- Wait=n

  The value you supply for *n* is the transaction lock timeout interval. When you supply this value, Oracle Rdb waits *n* seconds before aborting the wait and the RMU Extract session. Specifying a wait timeout interval of zero is equivalent to specifying Nowait.

- Nowait

  Will not wait for a locked resource to become available.

Usage Notes

- To use the RMU Extract command, you must have privileges to ATTACH to the source database; either SELECT privilege on the database or OpenVMS SYSPRV, READALL or BYPASS privileges.

  When the command includes the Items=VOLUME, access to individual tables is required to prepare the report when either Option=VOLUME_SCAN or Option=COLUMN_VOLUME is specified. In that case, SELECT access will be required to the tables selected by the Option=MATCH qualifier, or all tables.

- For tutorial information on using output from the RMU Extract command to load or unload a database, refer to the *Oracle Rdb Guide to Database Design and Definition*.

- The following list contains a description of what the RMU Extract command generates when it encounters certain RDO statements:

  – RDO and the data dictionary have the concept of validation clauses at the domain level. The ANSI/ISO SQL92 standard allows CHECK constraints defined on domains. While the actions of the ANSI/ISO CHECK constraint do differ from VALID IF in some respects, the RMU Extract command extracts the VALID IF clauses as domain

CHECK constraints if you specify the Language=SQL and Option=Full qualifiers.

– RDO multiline descriptions

  Because the RDO interface removes blank lines in multiline descriptions, the description saved in the metadata is not identical to that entered by the database definition. The RMU Extract command therefore cannot completely reconstruct the original description.

– Some RDO trigger definitions

  RDO trigger definitions that contain a trigger action within a join of two or more tables generates invalid SQL syntax. For example, the following RDO trigger definition includes a join with an embedded ERASE statement. When the RMU Extract command encounters this statement, Oracle RMU generates the invalid SQL trigger definition shown.

```
DEFINE TRIGGER EXAMPLE
    AFTER ERASE
    FOR C1 IN EMPLOYEES
    EXECUTE
        FOR C2 IN JOB_HISTORY
            CROSS C3 IN EMPLOYEES
            WITH (((C2.EMPLOYEE_ID = C3.EMPLOYEE_ID)
                 AND (C2.JOB_END MISSING))
                 AND (C3.EMPLOYEE_ID = C2.EMPLOYEE_ID))
            ERASE C2
        END_FOR
    FOR EACH RECORD.

CREATE TRIGGER EXAMPLE
    AFTER DELETE ON EMPLOYEES
        (DELETE FROM JOB_HISTORY C2, EMPLOYEES C3
            WHERE (((C2.EMPLOYEE_ID = C3.EMPLOYEE_ID)
                AND (C2.JOB_END IS NULL))
                AND (C3.EMPLOYEE_ID = C2.EMPLOYEE_ID))
        ) FOR EACH ROW;
```

  Note that in Oracle Rdb Version 4.1 and higher, including a trigger action within a join of two or more tables is invalid RDO syntax. For more information on this RDO restriction, see the ERASE and MODIFY entries in RDO HELP.

• Oracle CDD/Repository Version 5.3 and higher support table and column constraint definition and maintenance through CDO. The RMU Extract command, by default, assumes all constraint maintenance is with SQL and so follows each CREATE TABLE with an ALTER TABLE FROM pathname to add the constraints. However, this is no longer necessary if you are

using the later versions of Oracle CDD/Repository. To disable the output of the SQL ALTER TABLE statements which add constraints use the Option=Cdd_Constraint qualifier.

- If the Transaction_Type qualifier is omitted from the RMU Extract command line, a READ ONLY transaction is started against the database. This behavior is provided for backward compatibility with prior Oracle Rdb releases. If the Transaction_Type qualifier is specified without a transaction mode, the default value Automatic is used.
- If the database has snapshots disabled and the Transaction_Type qualifier was omitted, the transaction is restarted as READ WRITE ISOLATION LEVEL READ COMMITTED to reduce the number of rows locked by operations performed with the Option=Volume_Scan qualifier enabled.
- When Transaction_Type=Write is specified, the RMU Extract process does not attempt to write to the database tables.
- The following list shows the equivalent SQL expressions matched by the RMU Extract process:
  - NULLIF (a, b) is eqivalent to

```
CASE
  WHEN a = b THEN NULL
  ELSE a
END
```

  - NVL (a, ..., b) or COALESCE (a, ..., b) is equivalent to

```
CASE
  WHEN a IS NOT NULL THEN a
  ...
  ELSE b
END
```

  - The simple CASE expression

```
CASE a
  WHEN b THEN v1
  WHEN NULL THEN v2
  ...
  ELSE v3
END
```

is equivalent to

```
CASE
  WHEN a = b THEN v1
  WHEN a IS NULL THEN v2
  ...
  ELSE v3
END
```

The RMU Extract procedure tries to decode the internal representation to as compact a SQL expression as possible.

- The RMU Extract procedure decodes case expressions into ABS (Absolute) functions:

  ABS(a) is equivalent to:

  ```
  CASE
    WHEN a < 0 THEN -a
    ELSE a
  END
  ```

  In addition, similar forms of CASE expression are also converted to ABS:

  ```
  CASE
    WHEN a <= 0 THEN -a
    ELSE a
  END

  CASE
    WHEN a > 0 THEN a
    ELSE -a
  END

  CASE
    WHEN a >= 0 THEN a
    ELSE -a
  END
  ```

  It is possible that the RMU Extract process will change existing CASE expressions into this more compact syntax, even if they were not originally coded as an ABS function call.

- If the Group_Table option is used and the Item qualifier omits one or more of the Table, Index, or Storage_Map keywords, only the included items are displayed. For example, to extract just the indexes for the EMPLOYEES table:

  ```
  $ RMU/EXTRACT/ITEM=INDEX/OPTION=(GROUP_TABLE,MATCH=EMPLOYEES%)
  ```

To extract only the storage map and indexes for a table, use the following command:

```
$ RMU/EXTRACT/ITEM=(STORAGE_MAP,INDEX)/OPTION=(GROUP_TABLE, -
_$ MATCH=EMPLOYEES%)
```

- If the name of the LIST storage map is not known, it can be extracted using the following generic command:

```
$ RMU/EXTRACT/ITEM=STORAGE_MAP/OPTION=(GROUP_TABLE, -
_$ MATCH=RDB$SEGMENTED_STRING%)
```

- If the access control lists are defined using SQL ANSI semantics, they cannot be represented in RDO. In this case, the diagnostic message warns you that the protections must be extracted using the Language=SQL qualifier.
- If you specify Language=ANSI_SQL, a diagnostic message warns you that the ACL-style protections cannot be extracted in ANSI format.
- An access control entry (ACE) may exist that denies all access (aka NONE) for a user, however, there is no GRANT statement which can create such an entry. RMU Extract emulates this action by first granting the harmless SHOW privileges to the user, hence creating the ACE, and then modifies it by using REVOKE ALL PRIVILEGES.

Examples

Example 1

The following command extracts these database items: COLLATING_SEQUENCES, DOMAINS, TABLES, INDEXES, STORAGE_MAPS, VIEWS, SEQUENCES, and TRIGGERS.

The All option is the default. The All or Noall option can be used in conjunction with other items to select specific output. For example, the Items=(All,Nodatabase) qualifier selects all metadata items except the physical database characteristics.

```
$ RMU/EXTRACT/ITEM=(ALL, NODATABASE) MF_PERSONNEL
```

Example 2

The following command generates a DCL command procedure containing an RMU Load command for each table in the database:

```
$ RMU/EXTRACT/ITEMS=LOAD MF_PERSONNEL
```

Example 3

The following command displays the protection access control list (ACL) definitions in the mf_personnel.rdb database:

```
$ RMU/EXTRACT/ITEMS=PROTECTIONS MF_PERSONNEL.RDB
```

Example 4

The following command generates a DCL command procedure containing an RMU Unload command for each table in the database:

```
$ RMU/EXTRACT/ITEMS=UNLOAD MF_PERSONNEL.RDB
```

Example 5

The following example displays index definitions:

```
$ RMU/EXTRACT/ITEMS=INDEXES MF_PERSONNEL
```

Example 6

The following example displays domain and table definitions. Note that the Noall option could have been omitted.

```
$ RMU/EXTRACT/ITEMS=(NOALL,DOMAINS,TABLES) MF_PERSONNEL
```

Example 7

The following example displays definitions for domains (fields) and tables (relations) that reference data dictionary path names rather than using the information contained in the Oracle Rdb system tables. In addition to the database statements, it also references the data dictionary path name stored in the database, as shown in the following example:

```
$ RMU/EXTRACT/LANG=SQL/ITEM=ALL/OPTION=DIC/OUTPUT=CDD_MODEL.LOG/LOG= -
_$ CDD_EXTRACT.LOG CDD_SQL_DB
```

Example 8

The following example creates a command procedure containing a script of partial RMU Verify commands or verify command partitions for the mf_personnel database. This command procedure was created with the following RMU Extract command:

```
$ RMU/EXTRACT/ITEM=VERIFY MF_PERSONNEL
```

Example 9

The following command displays a query outline definition that was previously added to the mf_personnel database:

```
$ RMU/EXTRACT/ITEMS=(OUTLINES) MF_PERSONNEL
```

Example 10

1.28 RMU Extract Command

The following command displays the after-image journal (.aij) file configuration for mf_personnel:

```
$ RMU/EXTRACT/ITEMS=(ALTER_DATABASE) MF_PERSONNEL
```

Example 11

The following command displays the function definitions in mf_personnel for functions previously created using SQL:

```
$ RMU/EXTRACT/ITEM=FUNCTION MF_PERSONNEL
```

Example 12

The following command displays the table and column cardinalities based on sorted indexes:

```
$ RMU/EXTRACT/OPTION=COLUMN_VOLUME/ITEM=VOLUME MF_PERSONNEL
```

Example 13

The following example:

- Executes an SQL EXPORT statement to create an interchange file.
- Executes an RMU Extract command with the Item=Import qualifier to generate an Import script. In addition, the Option=Filename_Only qualifier is specified to prevent full file specifications from appearing in the SQL IMPORT script. (If full file specifications are used, you cannot test the script without replacing the database that was exported.)
- Defines a logical to define the interchange file name used in the Import script file.
- Executes the Import script file.

```
SQL> -- Create interchange file, SAVED_PERS.RBR.
SQL> --
SQL> EXPORT DATABASE FILENAME MF_PERSONNEL.RDB INTO SAVED_PERS.RBR;
SQL> EXIT;
$ !
$ RMU/EXTRACT/ITEM=IMPORT/OPTION=FILENAME_ONLY/OUTPUT=IMPORT_PERS.SQL -
_$ MF_PERSONNEL
$ DEFINE/USER RMUEXTRACT_RBR SAVED_PERS.RBR
$ !
```

```
$ SQL$
SQL> @IMPORT_PERS.SQL
SQL> set language ENGLISH;
SQL> set default date format 'SQL92';
SQL> set quoting rules 'SQL92';
SQL> set date format DATE 001, TIME 001;
SQL>
SQL> -- RMU/EXTRACT for Oracle Rdb V7.2-00     2-JAN-2006 15:34:38.63
SQL> --
SQL> --                          Physical Database Definition
SQL> --
SQL> ----------------------------------------------------------------------
SQL> import database from rmuextract_rbr
cont>     filename 'MF_PERSONNEL'
   .
   .
   .
```

Example 14

The following example shows an extract from the generated script when the SYS$LANGUAGE and LIB$DT_FORMAT symbols are defined. The language and format will default to ENGLISH and the standard OpenVMS format if these logical names are not defined.

```
$ DEFINE LIB$DT_FORMAT LIB$DATE_FORMAT_002,LIB$TIME_FORMAT_001
$ DEFINE SYS$LANGUAGE french
$ RMU/EXTRACT/OUT=SYS$OUTPUT/ITEM=DOMAIN MF_PERSONNEL/OPT=AUDIT_COMMENT
  .
  .
  .
-- Created on  8 janvier 2006 13:01:31.20
-- Never altered
-- Created by RDB_EXECUTE
--
SQL> CREATE DOMAIN ADDRESS_DATA_1
cont> CHAR (25)
cont> comment on domain ADDRESS_DATA_1 is
cont>   ' Street name';
  .
  .
  .
```

Example 15

If a database has snapshots set to ENABLED DEFERRED, it may be preferable to start a read/write transaction. In this environment, using the Transaction_type=(Read_only) qualifier causes a switch to a temporary snapshots ENABLED IMMEDIATE state. This transition forces the READ ONLY transaction to wait while all READ WRITE transactions complete, and then all new READ WRITE transactions performing updates will start

writing rows to the snapshot files for use by possible read only transactions. To avoid this problem use an RMU Extract command specifying a READ WRITE ISOLATION LEVEL READ COMMITTED transaction.

```
$ RMU/EXTRACT/ITEM=TABLE/OUT=TABLES.SQL-
    /TRANSACTION_TYPE=(WRITE,ISOLATION=READ)-
    SAMPLE.RDB
```

Example 16

This example specifies the options which were the default transaction style in prior releases.

```
$ RMU/EXTRACT/ITEM=TABLE/OUT=TABLES.SQL-
    /TRANSACTION_TYPE=(READ_ONLY)-
    SAMPLE.RDB
```

Example 17

If the database currently has snapshots deferred, it may be more efficient to start a read-write transaction with isolation level read committed. This allows the transaction to start immediately (a read-only transaction may stall), and the selected isolation level keeps row locking to a minimum. This could be explicitly stated by using the following command:

```
$ RMU/EXTRACT-
    /TRANSACTION_TYPE=(WRITE,ISOLATION=READ_COMMITTED)-
    SAMPLE.RDB
```

Using a transaction type of automatic adapts to different database settings:

```
$ RMU/EXTRACT-
    /TRANSACTION_TYPE=(AUTOMATIC)-
    SAMPLE.RDB
```

Example 18

This example shows the use of the Item=Workload qualifier to create a DCL command language script.

```
$ RMU/EXTRACT/ITEM=WORKLOAD -
    SCRATCH/LOG/OPTION=(FILENAME,AUDIT)
$! RMU/EXTRACT for Oracle Rdb V7.2-00            7-JAN-2006 22:00:42.72
$!
$!                               WORKLOAD Procedure
$!
$!------------------------------------------------------------------------
$ SET VERIFY
$ SET NOON
$
$! Created on  7-JAN-2006 10:12:26.36
$! Last collected on  7-JAN-2006 22:00:34.47
$!
$ RMU/INSERT OPTIMIZER_STATISTICS -
  SCRATCH -
  /TABLE=(CUSTOMERS) -
  /COLUMN_GROUP=(CUSTOMER_NAME) -
  /DUPLICITY_FACTOR=(4.0000000) -
  /NULL_FACTOR=(0.0000000) /LOG
$
$! Created on  7-JAN-2006 10:12:26.36
$! Last collected on  7-JAN-2006 22:00:34.58
$!
$ RMU/INSERT OPTIMIZER_STATISTICS -
  SCRATCH -
  /TABLE=(RDB$FIELDS) -
  /COLUMN_GROUP=(RDB$FIELD_NAME) -
  /DUPLICITY_FACTOR=(1.7794118) -
  /NULL_FACTOR=(0.0000000) /LOG
$
   .
   .
   .
$ SET NOVERIFY
$ EXIT
```

Example 19

The following example shows the use of the Match option to select a subset of the workload entries based on the wildcard file name.

1.28 RMU Extract Command

```
$ RMU/EXTRACT/ITEM=WORKLOAD -
    SCRATCH/LOG/OPTION=(FILENAME,AUDIT,MATCH:RDB$FIELDS%)
$! RMU/EXTRACT for Oracle Rdb V7.2-00                    8-JAN-2006 10:53
$!
$!                              WORKLOAD Procedure
$!
$!------------------------------------------------------------------------
$ SET VERIFY
$ SET NOON
$
! Created on  7-JAN-2006 15:18:02.30
$ SET NOON
$
$! Created on  7-JAN-2006 15:18:02.30
$! Last collected on  7-JAN-2006 18:25:04.27
$!
$ RMU/INSERT OPTIMIZER_STATISTICS -
  SCRATCH -
  /TABLE=(RDB$FIELDS) -
  /COLUMN_GROUP=(RDB$FIELD_NAME) -
  /DUPLICITY_FACTOR=(1.0000000) -
  /NULL_FACTOR=(0.0000000) /LOG
$ SET NOVERIFY
$ EXIT
```

Example 20

The following example shows the use of Item options Defer_Constraints, Constraints, and Match to extract a table and its constraints.

```
$ RMU/EXTRACT/ITEM=(TABLE,CONSTRAINT)-
_$ /OPTION=(FILENAME_ONLY,NOHEADER,-
_$          DEFER_CONSTRAINT,MATCH:EMPLOYEES%) -
_$ MF_PERSONNEL
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename MF_PERSONNEL';
create table EMPLOYEES (
    EMPLOYEE_ID ID_DOM,
    LAST_NAME LAST_NAME_DOM,
    FIRST_NAME FIRST_NAME_DOM,
    MIDDLE_INITIAL MIDDLE_INITIAL_DOM,
    ADDRESS_DATA_1 ADDRESS_DATA_1_DOM,
    ADDRESS_DATA_2 ADDRESS_DATA_2_DOM,
    CITY CITY_DOM,
    STATE STATE_DOM,
    POSTAL_CODE POSTAL_CODE_DOM,
    SEX SEX_DOM,
    BIRTHDAY DATE_DOM,
    STATUS_CODE STATUS_CODE_DOM);
    comment on table EMPLOYEES is
      'personal information about each employee';

alter table EMPLOYEES
    add constraint EMP_SEX_VALUES
        check(EMPLOYEES.SEX in ('M', 'F', '?'))
        deferrable
    add constraint EMP_STATUS_CODE_VALUES
        check(EMPLOYEES.STATUS_CODE in ('0', '1', '2', 'N'))
        deferrable
    alter column EMPLOYEE_ID
        constraint EMPLOYEES_PRIMARY_EMPLOYEE_ID
            primary key
            deferrable;

commit work;
```

Example 21

The following example shows the use of the option Group_Table to extract a table and its indexes:

1.28 RMU Extract Command

```
$ rmu/extract/item=(table,index)-
_$ /option=(group_table,match=employees%,-
_$          filename_only,noheader) db$:mf_personnel
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename MF_PERSONNEL';
create table EMPLOYEES (
    EMPLOYEE_ID ID_DOM
        constraint EMPLOYEES_PRIMARY_EMPLOYEE_ID
            primary key
            deferrable,
    LAST_NAME LAST_NAME_DOM,
    FIRST_NAME FIRST_NAME_DOM,
    MIDDLE_INITIAL MIDDLE_INITIAL_DOM,
    ADDRESS_DATA_1 ADDRESS_DATA_1_DOM,
    ADDRESS_DATA_2 ADDRESS_DATA_2_DOM,
    CITY CITY_DOM,
    STATE STATE_DOM,
    POSTAL_CODE POSTAL_CODE_DOM,
    SEX SEX_DOM,
    BIRTHDAY DATE_DOM,
    STATUS_CODE STATUS_CODE_DOM);
    comment on table EMPLOYEES is
      'personal information about each employee';

    create unique index EMPLOYEES_HASH
        on EMPLOYEES (
        EMPLOYEE_ID)
        type is HASHED SCATTERED
        store
            using (EMPLOYEE_ID)
                in EMPIDS_LOW
                    with limit of ('00200')
                in EMPIDS_MID
                    with limit of ('00400')
                otherwise in EMPIDS_OVER;

    create unique index EMP_EMPLOYEE_ID
        on EMPLOYEES (
        EMPLOYEE_ID
            asc)
        type is SORTED
        node size 430
        disable compression;

    create index EMP_LAST_NAME
        on EMPLOYEES (
        LAST_NAME
            asc)
        type is SORTED;
```

```
commit work;

alter table EMPLOYEES
    add constraint EMP_SEX_VALUES
        check(EMPLOYEES.SEX in ('M', 'F', '?'))
        deferrable
    add constraint EMP_STATUS_CODE_VALUES
        check(EMPLOYEES.STATUS_CODE in ('0', '1', '2', 'N'))
        deferrable;

commit work;
```

Example 22

The following example shows the output when you use the Item=Revoke_Entry qualifier:

```
$ RMU/EXTRACT/ITEM=REVOKE_ENTRY ACCOUNTING_DB
...
--                              Protection Deletions
--
-------------------------------------------------------------------------------

revoke entry
    on database alias RDB$DBHANDLE
    from [RDB,JAIN];

revoke entry
    on database alias RDB$DBHANDLE
    from [RDB,JONES];

revoke entry
    on database alias RDB$DBHANDLE
    from PUBLIC;

revoke entry
    on table ACCOUNT
    from [RDB,JONES];

revoke entry
    on table ACCOUNT
    from PUBLIC;

revoke entry
    on table ACCOUNT_BATCH_PROCESSING
    from [RDB,JONES];

revoke entry
    on table ACCOUNT_BATCH_PROCESSING
    from PUBLIC;
revoke entry
    on table BILL
    from [RDB,JONES];
```

```
revoke entry
    on table BILL
    from PUBLIC;
...
```

Example 23

The following example shows sample output for the WORK_STATUS table of MF_PERSONNEL. The uppercase DCL commands are generated by RMU Extract.

```
$ RMU/EXTRACT/ITEM=UNLOAD-
_$ /OPTION=(NOHEADER,FULL,MATCH:WORK_STATUS%) sql$database
$ CREATE WORK_STATUS.COLUMNS
! Columns list for table WORK_STATUS
! in DISK1:[DATABASES]MF_PERSONNEL.RDB
! Created by RMU Extract for Oracle Rdb V7.2-00 on  1-JAN-2006 20:50:25.33
STATUS_CODE
STATUS_NAME
STATUS_TYPE
$ RMU/UNLOAD -
        DISK1:[DATABASES]MF_PERSONNEL.RDB -
        /FIELDS="@WORK_STATUS.COLUMNS" -
        WORK_STATUS -
        WORK_STATUS.UNL
$
$ EXIT
```

```
$ RMU/EXTRACT/ITEM=LOAD-
_$ /OPTION=(NOHEADER,FULL,MATCH:WORK_STATUS%) sql$database
$ RMU/LOAD -
        /TRANSACTION_TYPE=EXCLUSIVE -
        /FIELDS="@WORK_STATUS.COLUMNS" -
        DISK1:[DATABASES]MF_PERSONNEL.RDB -
        WORK_STATUS -
        WORK_STATUS.UNL
$
$ EXIT
```

Example 24

The following example shows how to extract all constraints as an ALTER TABLE statement.

```
$ rmu/extract/item=(notab,constr) db$:sql_personnel/opt=(nohead,mat=empl%,defer)
set verify;
set language ENGLISH;
set default date format 'SQL92';
set quoting rules 'SQL92';
set date format DATE 001, TIME 001;
attach 'filename $DISK1:[JONES]SQL_PERSONNEL.RDB';
alter table EMPLOYEES
    add constraint EMP_SEX_VALUES
        check((EMPLOYEES.SEX in ('M', 'F')
            or (EMPLOYEES.SEX is null)))
        initially deferred deferrable
    add constraint EMP_STATUS_CODE_VALUES
        check((EMPLOYEES.STATUS_CODE in ('0', '1', '2')
            or (EMPLOYEES.STATUS_CODE is null)))
        initially deferred deferrable
    alter column EMPLOYEE_ID
        constraint EMP_EMPLOYEE_ID_NOT_NULL
            not null
            initially deferred deferrable;
```

Example 25

Using the STORAGE_AREAS and JOURNALS options to control output

The following example shows how to generate an ALTER DATABASE command that defines the three storage areas used for the EMPLOYEES table in the sample MF_PERSONNEL database. Here we disable the output of the journals, but request the storage areas with a matching wildcard string.

```
$ RMU/EXTRACT -
    /ITEM=ALTER_DATABASE -
    /DEFAULT=(NOALLOCATION,NOSNAPSHOT_ALLOCATION) -
    /OPTION=(NOHEADER,FILENAME_ONLY,ORDER_BY_NAME, -
             MATCH:EMPIDS%,STORAGE_AREAS,NOJOURNALS) -
    MF_PERSONNEL
```

Example 26

Using MATCH option to extract just one row cache definition

This example shows the MATCH option being used to extract a row cache definition which is then used to recreate the row cache, possibly after database maintenance.

1.28 RMU Extract Command

```
$! use MATCH to just get the definition for the EMPLOYEES cache
$       RMU/EXTRACT -
            /ITEM=ALTER_DATABASE -
            RMU_EXTRACT_ALTER_DATABASE_EX -
            /OUTPUT=BEFORE.SQL -
            /DEFAULTS=(NOALLOCATION,NOSNAPSHOT_ALLOCATION) -
            /OPTION=(NOHEADER,FILENAME_ONLY,NOJOURNALS,ROW_CACHES,-
                     MATCH=EMPLOYEES%)
$       SQL$
alter database
    filename RMU_EXTRACT_ALTER_DATABASE_EX

    drop cache EMPLOYEES
;
@BEFORE.SQL
SQL> set verify;
SQL> set language ENGLISH;
SQL> set default date format 'SQL92';
SQL> set quoting rules 'SQL92';
SQL> set date format DATE 001, TIME 001;
SQL> alter database
cont>     filename 'RMU_EXTRACT_ALTER_DATABASE_EX'
cont>
cont>     add cache EMPLOYEES
cont>         cache size is 300 rows
cont>         row length is 256 bytes
cont>         row replacement is DISABLED
cont>         shared memory is PROCESS
cont>         large memory is ENABLED
cont>         window count is 100
cont>         working set count is 10
cont>         number of reserved rows is 20
cont>         allocation is 100 blocks
cont>         extent is 100 blocks
cont>         row snapshot is ENABLED
cont>             (cache size is 900 rows)
cont> ; -- end alter database
$
```

1.29 RMU Insert Optimizer_Statistics Command

Inserts workload records into the RDB$WORKLOAD system relation.

Format

RMU Insert Optimizer_Statistics root-file-spec

Command Qualifiers	Defaults
/Column_Group=(Column-list)	None - Required Qualifier
/Duplicity_Factor=(floating-number)	/Duplicity_Factor=(1.0)
/[No]Log[=file-spec]	See description
/Null_Factor=(floating-number)	/Null_Factor=(0.0)
/Tables=(table-list)	None - Required Qualifier

Description

When you enable and collect workload statistics, the system table RDB$WORKLOAD is created and populated. (See Section 1.15 for details.) You can update or delete these statistics using the RMU Collect Optimizer_ Statistics command or the RMU Delete Optimizer_Statistics command, respectively.

You might delete entries in the RDB$WORKLOAD table by accident or you might delete them to test how effective it is to maintain those particular workload statistics. If you decide that you want to maintain those deleted statistics, you can insert them with the RMU Insert Optimizer_Statistics command. To ensure that you insert accurate values, always issue an RMU Show Optimizer_Statistics command with the Log qualifier before you issue an RMU Delete Optimizer_Statistics command. Refer to your generated log file for the values you should specify with the RMU Insert Optimizer_Statistics command.

In addition you can use the RMU Insert Optimizer_Statistics command to create workload statistics in a copy of your master database.

If you issue an RMU Collect Optimizer_Statistics command after having issued an RMU Insert Optimizer_Statistics command, statistics for the specified column groups are updated.

1.29 RMU Insert Optimizer_Statistics Command

Command Parameters

root-file-spec
Specifies the database into which optimizer statistics are to be inserted. The default file type is .rdb.

Command Qualifiers

Column_Group=(column-list)
Specifies a list of columns that comprise a column group. You must use the Tables qualifier to specify the table or tables with which the columns are associated.

The Column_Group=(column-list) qualifier is a required qualifier.

Duplicity_Factor=(floating_number)
Specifies the value to be inserted in the RDB$DUPLICITY_FACTOR column in the RDB$WORKLOAD table for the specified column group and table (or tables). The minimum value is 1.0 and the maximum value is the cardinality of the specified table. The default is the Duplicity_Factor=(1.0) qualifier.

Log
Log=file-spec
Nolog
Specifies how the statistics inserted into the RDB$WORKLOAD system table are to be logged. Specify the Log qualifier to have the information displayed to SYS$OUTPUT. Specify the Log=file-spec qualifier to have the information written to a file. Specify the Nolog qualifier to prevent display of the information. If you do not specify any of variation of the Log qualifier, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Null_Factor=floating-number
Specifies the value to be inserted in the RDB$NULL_FACTOR column in the RDB$WORKLOAD table for the specified column group and table (or tables). The minimum value is 0.0 and the maximum value is 1.0. The default is the Null_Factor=(0.0) qualifier.

Tables=(table-list)
Tables
Specifies the table or tables for which column group entries are to be inserted.

If you issue an RMU Collect Optimizer_Statistics command after you have inserted a workload column group into the RDB$WORKLOAD system table, those statistics are collected.

The Tables=(table-list) qualifier is a required qualifier.

Usage Notes

- To use the RMU Insert Optimizer_Statistics command for a database, you must have the RMU$ANALYZE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

  You must also have the SQL DBADM privilege on the database for which the insert is being performed.

- When you use the SQL ALTER DATABASE statement to set the RDB$SYSTEM storage area (or DEFAULT storage area) to read-only access for your database, the Oracle Rdb system tables are also set to read-only access.

  When you run RMU Insert Optimizer_Statistics on such databases, the RDB$SYSTEM, and if necessary, the DEFAULT storage area will be changed to READ WRITE to allow updates to the system tables and then restored to READ ONLY.

- Cardinality statistics are automatically maintained by Oracle Rdb. Physical storage and workload statistics are only collected when you issue an RMU Collect Optimizer_Statistics command. To get information about the usage of physical storage and workload statistics for a given query, define the RDMS$SET_FLAGS logical name to be "ESTIMATES". For example:

  ```
  $ DEFINE RDMS$SET_FLAGS "ESTIMATES"
  ```

  When you execute a query, if workload and physical statistics have been used in optimizing the query, you will see a line such as the following in the command output:

  ```
  ~O: Workload and Physical statistics used
  ```

- The Insert Optimizer_Statistics command modifies the RDB$LAST_ALTERED date of the RDB$WORKLOAD row so that it is activated for use by the optimizer.

Examples

Example 1

The following example:

1. Collects workload statistics for the JOB_HISTORY table using the RMU Collect Optimizer_Statistics command
2. Deletes the statistics for one of the JOB_HISTORY workload column groups
3. Inserts the statistics that were just deleted into the RDB$WORKLOAD system table using the RMU Insert Optimizer_Statistics command
4. Displays the current data stored in the RDB$WORKLOAD table for the JOB_HISTORY table using the RMU Show Optimizer_Statistics command

```
$ RMU/COLLECT OPTIMIZER_STATISTICS MF_PERSONNEL.RDB -
_$ /TABLE=(JOB_HISTORY)/STATISTICS=(WORKLOAD)/LOG
Start loading tables... at  3-JUL-1996 10:54:04.16
Done loading tables.... at  3-JUL-1996 10:54:04.69
Start collecting workload stats... at  3-JUL-1996 10:54:06.76
Maximum memory required (bytes) = 6810
Done collecting workload stats.... at  3-JUL-1996 10:54:07.64
Start calculating stats... at  3-JUL-1996 10:54:07.84
Done calculating stats.... at  3-JUL-1996 10:54:07.86
Start writing stats... at  3-JUL-1996 10:54:09.34

------------------------------------------------------------------------

Optimizer Statistics collected for table : JOB_HISTORY

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 2.7400000
  Null factor            : 0.0000000

  Workload Column group  :    EMPLOYEE_ID,  JOB_CODE,     JOB_START,
JOB_END,        DEPARTMENT_CODE,        SUPERVISOR_ID
  Duplicity factor       : 1.5930233
  Null factor            : 0.3649635
Done writing stats.... at  3-JUL-1996 10:54:09.90
$ RMU/DELETE OPTIMIZER_STATISTICS MF_PERSONNEL.RDB  -
_$ /TABLE=(JOB_HISTORY)/COLUMN_GROUP=(EMPLOYEE_ID,JOB_CODE, -
_$ JOB_START,JOB_END,DEPARTMENT_CODE,SUPERVISOR_ID)/LOG
Changing RDB$SYSTEM area to READ_WRITE.
Workload column group deleted for JOB_HISTORY :         EMPLOYEE_ID,
JOB_CODE,       JOB_START,      JOB_END,        DEPARTMENT_CODE,
SUPERVISOR_ID
```

1.29 RMU Insert Optimizer_Statistics Command

```
$ !
$ RMU/INSERT OPTIMIZER_STATISTICS MF_PERSONNEL.RDB -
_$ /TABLE=(JOB_HISTORY) /COLUMN_GROUP=(EMPLOYEE_ID,JOB_CODE, -
_$ JOB_START,JOB_END,DEPARTMENT_CODE,SUPERVISOR_ID) -
_$ /DUPLICITY_FACTOR=(1.5930233)/NULL_FACTOR=(0.3649635)/LOG
Changing RDB$SYSTEM area to READ_WRITE.
Workload column group inserted for JOB_HISTORY :        EMPLOYEE_ID,
JOB_CODE,       JOB_START,      JOB_END,        DEPARTMENT_CODE,
SUPERVISOR_ID
$ !
$ RMU/SHOW OPTIMIZER_STATISTICS MF_PERSONNEL.RDB -
_$ /TABLE=(JOB_HISTORY)/STATISTICS=(WORKLOAD)/LOG

------------------------------------------------------------------------

Optimizer Statistics for table : JOB_HISTORY

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 2.7400000
  Null factor            : 0.0000000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:54:09.62

  Workload Column group  :    EMPLOYEE_ID,  JOB_CODE,     JOB_START,
JOB_END,        DEPARTMENT_CODE,        SUPERVISOR_ID
  Duplicity factor       : 1.5930233
  Null factor            : 0.3649635
  First created time     :  3-JUL-1996 10:57:47.65
  Last collected time    :  3-JUL-1996 10:57:47.65
```

1.30 RMU Librarian Command

Allows you to list or remove backed up Oracle Rdb databases from a Librarian utility.

Format

RMU/Librarian backup-file-spec

Command Qualifiers	Defaults
/List[=(Output=file-name),(options,...)]	See Description
/Remove[=[No]Confirm,(options,...)]	See Description

Description

The RMU Librarian command allows you to list or remove backed up Oracle Rdb databases from a Librarian utility that conforms to the Oracle Media Manager interface.

You cannot perform both the list and remove operations within one command.

Command Parameters

backup-file-spec
Identifies the backed up Oracle Rdb database previously stored in the Librarian utility. Use the same backup file name that was used in the Oracle RMU Backup command. A default file type of .RBF is assumed if none is specified. Any device, directory, or version number specified with the backup file name will be ignored.

If the Librarian utility supports wild card characters, you can use them for the backup file name when you use the List qualifier. Wild card characters cannot be used with the Remove qualifier.

Command Qualifiers

List[=(Output=file-name),(options,...)]
Allows you to display a backed up Oracle Rdb database stored in a Librarian utility. If you use the List qualifier without the Output option, the output is sent to the default output device. If you use the Output option, the output is sent to the specified file. All data streams existing in the Librarian that were generated for the specified backup name will be listed. The information listed for each data stream can include:

- The backup stream name based on the backup file.
- Any comment associated with the backup stream name.
- The creation method associated with the backup stream name. This will always be STREAM.
- The creation data and time when the stream was backed up to the Librarian.
- Any expiration date and time specified for deletion of the stream by the Librarian.
- The media sharing mode that indicates if the media can be accessed concurrently or not.
- The file ordering mode that indicates if files on the media can be accessed in random order or sequential order.
- Any volume labels for the media that contain the backup stream.

Not all of these items will be listed depending on the particular Librarian utility.

The List qualifier can accept the following options:

- Trace_File=file-specification

  The Librarian application writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian application. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian application. The pre-determined values are:

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical-name=equivalence-value,...)

  Use this option to specify a list of process logical names that the Librarian application can use to specify catalogs or archives for listing or removing backup files, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of the logical names. The list of

process logical names is defined by Oracle RMU prior to the start of the list or remove operation.

Remove[=[No]Confirm,(options...)]
Allows you to delete all data streams existing in the Librarian that were generated for the specified backup file. This command should be used with caution. You should be sure that a more recent backup for the database exists in the Librarian under another name before you use this command. The Confirm option is the default. It prompts you to confirm that you want to delete the backup from the Librarian. If you do not want to be prompted, use the Noconfirm option. The deletion will be performed with no confirmation prompt.

The Remove qualifier can accept the following options:

- Trace_File=file-specification

  The Librarian application writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian application. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian application. The pre-determined values are:

  - Level 0, trace all error conditions, is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical-name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian application can use to specify catalogs or archives for listing or removing backup files, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of the list or remove operation.

1.31 RMU Load Command

Loads data into the tables of the database.

You can use the RMU Load command to:

- Perform the initial load of an Oracle Rdb database.
- Reload a table after performing a restructuring operation.
- Load an archival database.
- Move data from one database to another.
- Load security audit records from an OpenVMS security audit table into the database being audited, or into a different database than the one being audited.
- Load additional rows into an existing table. (However, note that it cannot be used to modify existing rows.)
- Import data into a database from an application that generates RMS files.

You can load data using either of the following two methods:

- A single-process method

  This was the only method available prior to Oracle Rdb V7.0. The single process method uses one process to both read the input file and load the target table.

- A multiprocess method, also called a parallel load

  The **parallel load** method, which you specify with the Parallel qualifier, enables Oracle RMU to use your process to read the input file and use one or more executors (subprocesses or detached slave process, depending on additional factors) to load the data into the target table. This results in concurrent read and write operations, and in many cases, substantially improves the performance of the load operation.

By default, Oracle RMU sets up a parallel load operation as follows:

- Your process serves as the load operation execution manager.
- Each storage area (partition) in the table being loaded is assigned an executor.
- Each executor is assigned four communications buffers.

  (You can override this default with the Buffer_Count option to the Parallel qualifier.)

- Each communications buffer holds the number of rows defined by the Row_Count qualifier.

Once the executors and communications buffers are set up, the parallel load operation processes the input file as follows:

1. Your process begins reading the input file and determines the target storage area for each row in the input file.
2. Your process places each row in the communications buffer for the executor assigned to the data's target storage area.
3. When an executor's first communications buffer becomes full, it begins loading the data into the target storage area.
4. If your process has another portion of data ready for a given executor before that executor has completed loading its first buffer of data, your process places the next portion of data in the second communications buffer for that executor.
5. Each executor, concurrent with each of the other executors, loads the data from its buffers.
6. Your process continues reading, sorting, and assigning data to each executor (by placing it in that executor's communication buffer) until all the data from the input file has been sorted, assigned, and loaded.

The Row_Count qualifier and Parallel qualifier (which provides the Executor_Count and Buffer_Count options) give you the ability to fine tune the Parallel load operation.

See the *Oracle Rdb Guide to Database Design and Definition* for tips on optimizing the performance of the load operation.

Format

1.31 RMU Load Command

RMU/Load root-file-spec table-name input-file-name

Command Qualifiers	Defaults
/Audit[=Database_File=db-name]	No audit table loaded
/Before=date-time	None
/Buffers=n	See description
/Commit_Every=n	See description
/[No]Constraints[=Deferred]	/Constraints
/Corresponding	See description
/[No]Defer_Index_Updates	/Nodefer_Index_Updates
/[No]Dialect=(dialect-opts)	/Dialect=SQL99
/[No]Execute	/Execute
/Fields=(column-name-list)	See description
/[No]Limit_To=n	/NoLimit_To
/List_Plan=output-file	See description
/[No]Log_Commits	/Nolog_Commits
/[No]Match_Name=table-name	/Nomatch_Name
/Parallel[=(options)]	See description
/[No]Place	/Noplace
/Record_Definition= ({File|Path}=name[,options])	See description
/Replace_Rows	None
/[No]Restricted_Access	/Norestricted_Access
/Row_Count=n	See description
/Since=date-time	None
/[No]Skip=n	/Noskip
/Statistics=(stat-opts)	See description
/Transaction_Type=Share-mode	Protected
/[No]Trigger_Relations[=(table_name_list)]	/Trigger_Relations
/[No]Virtual_Fields[=[No]Automatic]	/Novirtual_Fields

Description

The RMU Load command accepts the following six types of data files:

- Binary data file (default extension .unl)
- Delimited text data file (default extension .unl)
- Text data file (default extension .unl)
- Specially structured file (default extension .unl)
- OpenVMS security audit journal file (default extension .audit$journal)
- Text data file with comma separated values (default extension .csv)

1.31 RMU Load Command

With the exception of the specially structured file and the security audit journal file, you must provide a record definition file (.rrd) on the RMU Load command line to load these data files. The record definition file provides Oracle RMU with a description of (metadata for) the data you are loading.

The following list describes the additional requirements for loading each of these types of files:

- Text data file

  To load a text data file (.unl), you must specify the Record_Definition qualifier with the Format=Text option.

  The following command loads text data (employees.unl) into the EMPLOYEES table of the mf_personnel database. The employees.rrd file provides the record definition for the data in employees.unl

```
$ RMU/LOAD/RECORD_DEFINITION=(FILE=employees.rrd, FORMAT=TEXT) -
_$ mf_personnel EMPLOYEES employees.unl
```

  You can generate an appropriate .rrd file for the preceding example by issuing the following command:

```
$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=employees.rrd, FORMAT=TEXT) -
_$ mf_personnel EMPLOYEES unload.unl
```

- Delimited text data files

  To load delimited text data files (.unl) you must specify the Record_ Definition qualifier with the Format=Delimited_Text option.

  The following command loads delimited text data (employees.unl) into the EMPLOYEES table of the mf_personnel database. The employees.rrd file describes the format of employees.unl.

```
$ RMU/LOAD/RECORD_DEFINITION=(FILE=employees.rrd, -
_$ FORMAT=DELIMITED_TEXT, TERMINATOR="#") -
_$ mf_personnel EMPLOYEES employees.unl
```

  You can generate an appropriate .rrd file for the preceding example by issuing the following command:

```
$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=employees.rrd, -
_$ FORMAT=DELIMITED_TEXT) mf_personnel EMPLOYEES unload.unl
```

- Text data file with comma separated values

  Files containing comma separated values look very similar to delimited text files. However, a CSV file, by convention, includes an initial row of data that contains the names of the fields. To load comma separated values data files (.csv), you must specify the Record_Definition qualifier with the Format=CSV option. The following command loads comma

separated values data (employees.csv) into the EMPLOYEES table of the mf_personnel database. The employees.rrd file describes the format of employees.unl.

```
$ RMU/LOAD/RECORD_DEFINITION=(FILE=employees.rrd, FORMAT=CSV) -
_$ mf_personnel EMPLOYEES employees.unl
```

You can generate an appropriate .rrd file for the preceding example by issuing the following command:

```
$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=employees.rrd, FORMAT=CSV) -
_$ mf_personnel EMPLOYEES unload.unl
```

- Binary data files

  To load binary data files, you must ensure that the records you load match the record definition in both size and data type. The records must all have the same length and the data in each record must fill the entire record. If the last field is character data and the information is shorter than the field length, the remainder of the field must be filled with spaces. You cannot load a field that contains data stored in packed decimal format.

  The following command loads binary data (employees.unl) into the EMPLOYEES table of the mf_personnel database. The employees.rrd file describes the format of employees.unl.

  ```
  $ RMU/LOAD/RECORD_DEFINITION=(FILE=employees.rrd) mf_personnel -
  _$ EMPLOYEES employees.unl
  ```

  You can generate an appropriate .rrd file for the preceding example by issuing the following command:

  ```
  $ RMU/UNLOAD/RECORD_DEFINITION=(FILE=employees.rrd) mf_personnel -
  _$ EMPLOYEES unload.unl
  ```

- Specially structured binary files that include both data and metadata.

  To load the specially structured binary files (created by the RMU Unload command without the Record_Definition qualifier) you must specify the file (.unl) created by the RMU Unload command.

  The following command loads the binary data contained in the employees.unl file into the EMPLOYEES table of the mf_personnel database. The record definition information is contained within the binary .unl file.

  ```
  $ RMU/LOAD MF_PERSONNEL EMPLOYEES employees.unl
  ```

This specially structured employees.unl file is created with the following RMU Unload command:

```
$ RMU/UNLOAD MF_PERSONNEL EMPLOYEES employees.unl
```

- Security audit journal files

  To load the records from a security audit journal file maintained by the OpenVMS operating system, you must decide whether to load records into the same database for which security audit journal records are being recorded or to load them into a separate database. In either case you do not need to specify a record definition file; use of the Audit qualifier indicates to Oracle RMU that the record definition is that of the security audit journal file.

  The following command loads the records from the security audit journal file (with a logical name of SECURITY_AUDIT) for the mf_personnel database into the AUDIT_TABLE table of the mf_personnel database:

```
$ RMU/LOAD/AUDIT MF_PERSONNEL.RDB AUDIT_TABLE -
_$ SECURITY_AUDIT
```

  This example loads the records from the security audit journal file (with a logical name of SECURITY_AUDIT) for the mf_personnel database into the AUDIT_TABLE table of the audit database:

```
$ RMU/LOAD/AUDIT=DATABASE_FILE=MF_PERSONNEL.RDB AUDIT.RDB -
_$ AUDIT_TABLE SECURITY_AUDIT
```

  See the Usage Notes for more detailed information on loading security audit journal records and the file name of the security audit journal.

In all cases where you specify a record definition file (.rrd), the record definition file and the database definition of the table being loaded must match in the number of specified fields and the data type of each field. If the data you want to load has more fields than the database table definition specifies, you can still load the data, but you must use the FILLER keyword with the field definition in your .rrd file to represent the additional field. See Example 15 in the Examples section.

By default, the table specified in the RMU Load command is reserved for PROTECTED WRITE.

Table 1–12 shows the data type conversions that can occur while you are performing a load or unload operation.

Table 1–12 Data Type Conversions Performed by Oracle Rdb

Original Data Type	New Data Type
TINYINT	INTEGER, QUADWORD, SMALLINT, FLOAT, DOUBLE PRECISION, VARCHAR, CHAR
SMALLINT	INTEGER, QUADWORD, FLOAT, DOUBLE PRECISION, VARCHAR, CHAR
INTEGER	SMALLINT, QUADWORD, FLOAT, DOUBLE PRECISION, VARCHAR, CHAR
QUADWORD	SMALLINT, INTEGER, FLOAT, DOUBLE PRECISION, VARCHAR, CHAR
FLOAT	DOUBLE PRECISION, CHAR, and VARCHAR
DOUBLE PRECISION	FLOAT, CHAR, and VARCHAR
DATE	CHAR or VARCHAR
TIME	CHAR or VARCHAR
TIMESTAMP	CHAR or VARCHAR
INTERVAL	CHAR or VARCHAR
CHAR	FLOAT, DOUBLE PRECISION, DATE, TIME, TIMESTAMP, INTERVAL, VARCHAR, SMALLINT, INTEGER, or QUADWORD

See the *Oracle Rdb SQL Reference Manual* for a description of these data types.

Command Parameters

root-file-spec
The file specification for the database root file into which the table will be loaded. The default file extension is .rdb.

table-name
The name of the table to be loaded, or its synonym.

When the Audit qualifier is specified, the table-name parameter is the name of the table in which you want the security audit journal records to be loaded. If the table does not exist, the RMU Load command with the Audit qualifier creates the table and loads it. If the table does exist, the RMU Load command with the Audit qualifier loads the table.

input-file-name
The name of the file containing the data to be loaded. The default file extension is .unl.

When the Audit qualifier is specified, the input-file-name parameter is the name of the journal containing the audit record data to be loaded. The default file extension is .AUDIT$JOURNAL. You can determine the name of the security audit journal by using the DCL SHOW AUDIT/JOURNAL command.

Command Qualifiers

Audit[=Database_File=db-name]
Allows you to load a database's security audit records from an OpenVMS security audit journal into one of the following:

- A table in the database being audited

  Specify the Audit qualifier without the Database_File option to indicate that you want the security audit records to be loaded into the database specified with the root-file-spec parameter.

- A table in a different database than the one being audited

  Specify the Audit=Database_File=db-name qualifier to indicate that you want to security audit records for the database specified with the root-file-spec command parameter to be loaded into the database specified with the db-name option parameter.

If you specify the Audit qualifier, you cannot specify the Fields or Trigger_ Relations qualifiers.

In addition you cannot specify the Audit qualifier with a parallel load operation. If you attempt to do so, Oracle RMU issues a warning and performs a single-executor load operation.

Before=date-time
The /Before qualifier is used in conjunction with the RMU/LOAD/AUDIT command. This qualifier, along with the /Since qualifier, allow audit records to be filtered by timestamp. The qualifiers BEFORE and SINCE can specify the date/time range which will be extracted from the OpenVMS audit journal and saved in the target auditing table.

These qualifiers accept the standard OpenVMS date/time specification that includes special keywords such as YESTERDAY, TODAY and TOMORROW. These values can be very effective when used with the List_Plan qualifier and used later when using RMU Load Plan.

If these qualifiers are omitted, then their values default to minimum and maximum possible date/time values.

This example shows the use of DCL symbols to be used at runtime to provide the date/time range.

```
$  RMU/LOAD/AUDIT -
    /SINCE=&start_ts -
    /BEFORE=&end_ts -
   TESTDB AUDIT_RECORDS -
   SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
%RMU-I-DATRECREAD,  91 data records read from input file.
%RMU-I-DATRECSTO,   63 data records stored 27-NOV-2013 00:59:33.18.
$
```

This example uses explicit date and time values to load a specific range of audit record.

```
$  RMU/LOAD/AUDIT -
    /SINCE=1-jan-2011 -
    /BEFORE="1-nov-2013 13:00" -
   TESTDB AUDIT_RECORDS -
   SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
%RMU-I-DATRECREAD,  91 data records read from input file.
%RMU-I-DATRECSTO,   0 data records stored 27-NOV-2013 00:59:33.75.
$
```

Buffers=n

Specifies the number of database buffers used for storing data during the load operation. If no value is specified, the default value for the database is used. (The default value for the database is defined by the logical name RDM$BIND_BUFFERS, or if the logical is not defined, can be determined by using the RMU Dump command with the Header qualifier. The RDM$BIND_BUFFERS logical name, if defined, overrides the value displayed with the RMU Dump command.) Fewer I/O operations are required if you can store as much data as possible in memory when many indexes or constraints are defined on the target table. Therefore, specify more buffers than allowed by the default value to increase the speed of the load operation.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for detailed recommendations on setting the number of database buffers.

Commit_Every=n

Specifies the frequency with which Oracle Rdb commits the data being loaded. For a single-executor load operation, Oracle Rdb commits the data after every *n* records that are stored. The default is to commit only after all records have been stored.

For a parallel load operation, the Commit_Every qualifier applies separately to each of the executors (processes) used. For example, if five parallel processes are running, and the Commit_Every=2 qualifier is specified, Oracle RMU commits data for each process after it has stored 2 records. This means that if the Commit_Every=1000 qualifier is specified when you load one million records with 10 parallel processes, the .ruj files will store up to 10,000 rows of before-image data.

If you specify the Defer_Index_Updates qualifier and a high value for the Commit_Every qualifier, memory requirements are high. See the description of the Defer_Index_Updates qualifier for details. Commit operations may occur more frequently than you specify under certain conditions. See the description of the Defer_Index_Updates qualifier for details.

To determine how frequently you should commit data, decide how many records you are willing to reload if the original load operation fails. If you use the Statistics=On_Commit qualifier, you receive a message indicating the number of records loaded at each commit operation. Then, if a failure occurs, you know where to resume loading.

If you specify the Place qualifier and a failure occurs, resume loading at the point of the previous commit, instead of the record number of the last successful commit. The Place qualifier restructures the .unl file prior to loading, so the record number on which the load operation failed does not correspond to the same number in the original .unl file.

Constraints[=Deferred]
Noconstraints

Specifies when or if constraints are evaluated for data being loaded. If you specify the Constraints qualifier, constraints are evaluated as each record is loaded. If you specify the Noconstraints qualifier, constraints are not evaluated at all during the load operation. If you specify the Constraints=Deferred qualifier, constraints are evaluated after all data from the input file has been loaded.

The default is the Constraints qualifier.

Oracle Corporation recommends that you accept the default for most load operations. The Noconstraints and Constraints=Deferred qualifiers are useful when load performance is your highest priority, you fully understand the constraints defined for your database, and you are familiar enough with the input data to be fairly certain that loading that data will not violate constraints; then you might use these qualifiers as follows:

- Constraints=Deferred

This qualifier is particularly useful for improving performance when you are loading data into a new table. Oracle Corporation strongly recommends that you issue an RMU Verify command with the Constraints qualifier when the load operation has completed. Note, however, that issuing the RMU Verify command after the load operation has completed takes about the same amount of time that would have been spent had you specified the RMU Load command with the Constraints qualifier. In other words, by specifying the Constraints=Deferred qualifier, you are only delaying when the constraint verification will take place.

- Noconstraints

  This qualifier is particularly useful when you are performing a parallel load operation with the Defer_Index_Updates qualifier. Oracle Corporation strongly recommends that you issue an RMU Verify command with the Constraints qualifier when the load operation has completed. Note, however, that when you issue the RMU Verify command with the Constraints qualifier, *all* rows in the table are checked for constraint violations, not just the rows that are loaded.

Consider the following before issuing an RMU Load command with the Noconstraints or Constraints=Deferred qualifier:

- If a table is populated with data prior to a load operation, it is less expensive to check constraints on each record as it is being loaded, than to verify constraints on the entire table after the set of new records has been loaded. For example, assume you load 200 new records into a table that currently holds 2,000 records and one constraint is defined on the table. If you verify constraints as the records are being loaded, constraint validation is performed 200 times. If you wait and verify constraints after the load operation completes, constraint verification must be performed for 2,200 records
- If an RMU Verify command reveals that constraint violations occurred during the load operation, you must track down those records and either remove them or make other modifications to the database to restore the data integrity. This can be a time-consuming process.

Also consider a situation where all of the following are true:

- You perform a parallel load operation
- You specify the Constraints qualifier
- The table into which you are loading data has a constraint defined on it
- The constraint defined on the table was defined as deferred

- Constraint evaluation fails during the load operation

In a case such as the preceding, you can not easily determine which rows were loaded and which were not. Therefore Oracle Corporation recommends that if deferred constraints are defined on a table, then you should also specify the Constraints=Deferred qualifier in your parallel load command. When you follow this recommendation, the records that violate the constraint are stored in the database. When the load operation completes, you can remove from the database those records that violate the constraint.

See Example 6 in Section 1.70 for an example of the steps to take if an RMU Verify command reveals that an RMU Load command has stored data that violates constraints into your database.

Corresponding
Loads fields into a table from the .unl file by matching the field names in the .rrd file to the column names in the table. The Corresponding qualifier makes it more convenient to unload, restructure, and reload a table.

For example, if the columns in the table appear in the order: EMPLOYEE_ID, LAST_NAME, FIRST_NAME, but the data in your .unl file appears in the order: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, and your .rrd file lists the fields in the order: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, you can use the Corresponding qualifier to load the data in your .unl file correctly. (You could also use the Fields qualifier to accomplish the same task, but this can get tedious if there are numerous fields.)

The .unl file must contain data for each field in the database into which it is being loaded; if it does not, you should use the Fields qualifier.

If the Corresponding qualifier is omitted, the RMU Load command loads the data into database fields by the ordinal position in which they appear in the .unl, not by the column name described in the .rrd file.

The Corresponding qualifier cannot be used with either the Fields or Audit qualifiers.

Defer_Index_Updates
Nodefer_Index_Updates
The Defer_Index_Updates qualifier specifies that non-unique indexes (other than those that define the placement information for data in a storage area) will not be rebuilt until commit time.

Use of this qualifier results in less I/O and fewer lock conflicts than when index builds are not deferred, but results in a total failure of a load operation if *any* lock conflicts are encountered. In such a case, the entire load operation is rolled back to the previous commit and you must repeat the load operation. (Record insertion recommences at the beginning of the input file). For this reason, you should only use the Defer_Index_Updates qualifier when all of the following are true:

- You specify the Noconstraints qualifier (or you have dropped constraints, or no constraints are defined on the table).
- You have dropped triggers from the table (or triggers are not defined for the table).
- No other users are accessing the table being loaded.

Also be aware that required virtual memory can be quite large when you defer index updates. Required virtual memory is directly proportional to the following:

- The length of the Ikeys in the indexes being deferred
- The number of indexes being deferred
- The value for n specified with the Commit_Every qualifier

You can *estimate* the amount of virtual memory required for each deferred index using the following formula, where:

- `n` = the value specified with the Commit_Every qualifier
- `I` = (length of the Ikey + 50)

```
n * (I * number_defered_ikeys)
```

The Nodefer_Index_Updates qualifier is the default. When you specify the Nodefer_Index_Updates qualifier (or accept the default), both the indexes that define the placement information for data in a storage area and any other indexes defined on the table being loaded are rebuilt at verb time.

This can result in a managed deadlock situation when the Parallel qualifier is specified. The following describes such a scenario:

- Executor_1 locks index node A in exclusive mode
- Executor_2 locks index node B in exclusive mode
- Executor_1 requests a lock on index node B
- Executor_2 requests a lock on index node A

In such a situation, Oracle Rdb resolves the deadlock by directing one of the executors to commit the data it has already stored. This resolves the deadlock situation and the load operation continues.

Dialect
Nodialect
The Dialect qualifier is used to control whether truncation of string data during the loading of data is reported or not. This loss of data might be significant. RMU Load defaults to SQL dialect SQL99 which implicitly checks for and reports truncations during INSERT operations.

- /NODIALECT, /DIALECT=SQL89 or /DIALECT=NONE will not report any truncation errors.
- /DIALECT=SQL99 (the default) enables reporting of truncation errors. Note that truncation occurs if non-space characters are discarded during the insert.

Execute
Noexecute
The Execute and Noexecute qualifiers are used with the List_Plan qualifier to specify whether or not the generated plan file is to be executed. The Noexecute qualifier specifies that the plan file should be created but should not be executed. Regardless of whether you use the Noexecute or Execute qualifier (or accept the default), Oracle RMU performs a validity check on the RMU Load command you specify.

The validity check determines such things as whether the specified table is in the specified database, the .rrd file (if specified) matches the table, and that the number of columns specified with the Fields qualifier matches the number of columns in the .unl file. The validity check does not determine such things as whether your process and global page quotas are sufficient.

By default, the plan file is executed when an RMU Load command with the List_Plan qualifier is issued.

Fields=(column-name-list)
Specifies the column or columns of the table to be loaded into the database. If you list multiple columns, separate the column names with a comma, and enclose the list of column names within parentheses. Also, this qualifier specifies the order of the columns to be loaded if that order differs from the order defined for the table. The number and data type of the columns specified must agree with the number and data type of the columns in the .unl file. The default is all columns defined for the table in the order defined.

If you specify an options file in place of a list of columns, and the options file is empty, the RMU Load command loads all fields.

Limit_To=n
NoLimit_To
The LIMIT_TO qualifier defines the maximum number of rows to read from the source data file. Depending upon the value specified for the SKIP qualifier, this also controls the number of rows written to the database.

The value of LIMIT_TO may not be zero, and the value of SKIP may not exceed this limit.

The default is NOLIMIT_TO which indicates that all rows read from the unload data file can be inserted into the database table, depending on other factors such as the value of the SKIP qualifier.

List_Plan[=output-file]
Specifies that Oracle RMU should generate a plan file and write it to the specified output file. A plan file is a text file that contains all the qualifiers specified on the RMU Load command line. In addition, it specifies the executor names (if you are performing a parallel load operation), the directory for the .ruj files, the exception files, and the file created by the Place_Only qualifier (if specified).

Oracle RMU validates the Oracle RMU command prior to generating the plan file to ensure that an invalid plan file is not created. (This is true regardless of whether or not you specify the Noexecute qualifier.) For example, the following command is invalid and returns an error message because it specifies conflicting qualifiers (Corresponding and Fields):

```
$ RMU/LOAD/RECORD_DEF=FILE=NAMES.RRD/CORRESPONDING -
_$ /FIELDS=(LAST_NAME, FIRST_NAME)/LIST_PLAN=my_plan.plan -
_$ MF_PERSONNEL.RDB EMPLOYEES NAMES.UNL
%RMU-F-CONFLSWIT, conflicting options CORRESPONDING and FIELDS...
```

See the description of the Execute qualifier for a description of what items are included when Oracle RMU validates the RMU Load command. See the Examples section for a complete example and description of a plan file.

You can use the generated plan as a starting point for building a load operation that is tuned for your particular configuration. The output file can be customized and then used with subsequent load operations as the parameter to the RMU Load Plan command. See Section 1.32 for details.

1.31 RMU Load Command

If you want to create only a load plan file and do not want to execute the load plan when the RMU Load command is issued, specify the Noexecute qualifier. When you specify the Noexecute qualifier, you must specify a valid Oracle RMU command.

One way to prototype a plan file prior to creating a potentially very large .unl file is to specify the List_Plan qualifier and the Noexecute qualifier along with a valid record definition (.rrd) file and an *empty* .unl file on the RMU Load command line. The .rrd file contains the information Oracle RMU needs to perform the validation of the plan file; however, because data is not loaded when you specify the Noexecute qualifier, Oracle RMU does not attempt to load the .unl file. Note, however, that you cannot specify the Fields qualifier when using this strategy. (When you specify the Fields qualifier, Oracle RMU checks to make sure the number of columns specified with the Fields qualifier match the number of columns specified in the .unl file.)

If you do not specify a file extension, the default file extension for the plan file is .plan.

Log_Commits
Nolog_Commits
Causes a message to be printed after each commit operation. In the case of a parallel load, a message is printed after each executor commits.

The default is the Nolog_Commits qualifier, where no message is printed after individual commit operations. The Nolog_Commits qualifier does, however, cause a commit operation total to be printed after the operation completes or generates an error.

Match_Name=table-name
Nomatch_Name
Specifies the table name to be read. Tables exported by SQL into an interchange file can be individually loaded into a database.

The default behavior of the RMU Load command is to locate and load the first set of table data in the unload file. If this is not the table you want, you can use the Match_Name qualifier to specify a different table name. If the Match_Name qualifier is used without a table-name, Oracle RMU assumes the name of the table being loaded is also the name of the table in the source data file. The default is the Nomatch_Name qualifier.

Parallel[=(options)]
Specifies a parallel load operation. A parallel load operation is especially effective when you have large partitioned tables that do not contain segmented strings and for which no constraints or triggers are defined.

If you specify the Parallel qualifier without any options, your load operation is assigned one executor and four communications buffers for that executor. A communications buffer is used for communications between your process and the executors.

If you want to assign additional executors or communications buffers, or both, use one or both of the following options:

- Buffer_Count=n

  Allows you to specify the number of *communications* buffers assigned to each executor in a parallel load operation.

  Do not confuse this with the Buffers=n qualifier. The Buffers=n qualifier specifies the number of *database* buffers to use during the load operation.

- Executor_Count=n

  Allows you to specify the number of worker processes to be assigned to the load operation. Ideally, the number of executors should be equal to the number of table partitions. You should not assign a greater number of executors than the number of table partitions. If a table is randomly or vertically partitioned, Oracle RMU creates only one executor, regardless of the number you specify.

  If the user account's MAXDETACH UAF value is greater than 0, then executors are created as detached processes. If there is no MAXDETACH value set, then executors are created as subprocesses. (A MAXDETACH value = 0 equates to unlimited detached processes.)

At the end of each load operation, Oracle RMU displays summary statistics for each executor in the load operation and the main process. Look at the "Idle time" listed in the statistics at the end of the job to detect data skew and look at "Early commits" to detect locking contention.

If some executors have a large amount of idle time, you likely have data that is skewed. Ideally, data loaded with the Parallel qualifier should appear in random order within the .unl file. Data that is already in partition order when you attempt to perform a parallel load operation results in high idle time for each executor and thus defeats the advantages of a parallel load operation.

The summary statistics also list the number of records read from the input file, the number of data records stored, and the number of data records rejected. In most cases, the number of data records rejected plus the number of data records stored equals the number of data read from the input file. However, under the following circumstances this equation does not hold:

- The parallel load operation aborts due to a duplicate record that is not allowed.

- You did not specify an exception file.

Similarly if a load operation aborts due to a record in the input file being improperly delimited for a delimited text load, the records rejected plus the records stored do not equal the number of records read from the input file.

You cannot use a parallel load operation to load list data (segmented string) records or security audit records. If you specify a parallel load operation and attempt to load list data or security audit records, Oracle RMU returns a warning and performs a single-process (non-parallel) load operation.

Place
Noplace
Sorts records by target page number before they are stored.

The Place qualifier automatically builds an ordered set of database keys (dbkeys) when loading data and automatically stores the records in dbkey order, sequentially, page by page. During a parallel load operation, each worker executor builds its own ordered set of dbkeys.

The number of work files used by the RMU Load command is controlled by the RDMS$BIND_SORT_WORKFILES logical name. The allowable values are 1 through 10 inclusive, with a default value of 2. The location of these work files can be specified with device specifications, using the SORTWORKn logical name (where *n* is a number from 0 to 9). See the OpenVMS documentation set for more information on using SORT/MERGE. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on using these Oracle Rdb logical names.

A significant performance improvement occurs when the records are stored by means of a hashed index. By using the Commit_Every qualifier with the Place qualifier, you can specify how many records to load between COMMIT statements. Performance may actually decrease when records are stored by means of a sorted index.

The default is the Noplace qualifier.

Record_Definition=({File | Path}=name[,options])
Specifies the RMS record definition or the data dictionary record definition to be used when data is loaded into the database. Use the File=name parameter to specify an RMS record definition file; use the Path=name parameter to specify that the record definition be extracted from the data dictionary. (If the record definition in the data dictionary contains variants, Oracle RMU will not be able to extract it.)

The default file extension for the File=name parameter is .rrd. The syntax for the .rrd file is similar to that used by the Common Dictionary Operator (CDO) interface for the data dictionary. You must define columns before you can define rows. You can place only one column on a line. You can create a sample .rrd file by using the RMU Unload command with the Record_ Definition qualifier. You must ensure that the record definition in the .rrd file and the actual data are consistent.

RMU Load does not check to see that data types in the record definition and the data match. The database administrator can add transformation routines to convert data from the data file prior to being inserted in the table. See Appendix A and the *Oracle Rdb Guide to Database Design and Definition* for more information about the format of the .rrd file.

You must specify either the File=name or Path=name parameter.

The options available are:

- Exception_File=exception-file

  Allows you to write unloadable records to a single exception file for a single-process load operation and into multiple exception files for a parallel load operation. If you generate a load plan for a parallel load operation, each executor is assigned its own exception file. In this case, the exception-file name you specify is given a different file extension for each executor.

  While Oracle RMU is loading data from an RMS file, if an exception file is specified, then under certain circumstances an invalid record in the input file does not cause the RMU Load command to abort. Instead, Oracle RMU creates the exception file (or files), writes the unloadable record into this exception file (or files), and continues loading the remaining records. This process occurs only if the data is invalid on the actual insert, due to index, constraint, or trigger errors. If the record has an invalid format in the RMS file (for example, a missing delimiter), the exception file is not used, and the load process aborts.

  At the end of the load operation, you can process the exception file (or files) to correct any problems, and then reload directly from the exception file or files. The load operation gives an informational message for each of the unloadable records and also gives a summary of the number of records stored and the number of records rejected.

  All records that could not be loaded will be written into the file or files as specified with the argument to the Exception_File option. The default file extension for the exception file is .unl for single-process loads; for parallel loads the default extension is EXC_n, where n corresponds to the executor number assigned by Oracle RMU. The exception file or files are created only if there are unloadable records. If the Exception_File option is not

specified, no exception files are created, and the load operation aborts at the first occurrence of an exception.

However, note that if the Defer_Index_Updates qualifier is specified, and a constraint violation or lock conflict occurs, the load operation aborts when it attempts to commit the transaction.

If the Defer_Index_Updates qualifier is not specified, records that cause a constraint violation are written to the exception file or files and the load operation continues loading the remaining records.

- Format=Text

  If you specify the Format=Text option, Oracle RMU converts all data to printable text before loading it.

- If you do not specify the Format option, then Oracle RMU expects to load a fixed-length binary flat file. The data type of the fields must be specified in the .rrd file.

- Format=(CSV[,delimiter-options])

  CSV (comma separated values) format is a variation of Delimited_Text format. Please refer to that Format option for options and restrictions.

- Format=(Delimited_Text [,delimiter-options])

  If you specify the Format=Delimited_Text option, the .rrd file contains only text fields and specifies the maximum length of the columns in the file containing delimited ASCII text. The column values that are longer than those specified in the .rrd file are truncated.

  Note that DATE VMS types must be specified in the collatable time format, which is *yyyymmddhhmmsscc*. For example, March 20, 1993 must be specified as: 1993032000000000.

  Unless you specify the Format=Delimited_Text option, delimiters are regarded as part of the data by Oracle RMU. Example 13 in the Examples section demonstrates the Format=Delimited_Text option. Delimiter options (and their default values if you do not specify delimiter options) are as follows. Note that with the exception of the Prefix and Suffix delimiter options, the values specified must be unique. The Prefix and Suffix values can be the same value as each other, but not the same as other delimiter options. The Null string must also be unique.

  – Prefix=string
    NoPrefix

Specifies a prefix string that begins any column value in the input file. If you omit this option, the column prefix is assumed to consist of a quotation mark ("). If you specify NoPrefix or an empty string, then no prefix will be used.

- Separator=string
  NoSeparator

  Specifies a string that separates column values of a row. If you omit this option, the column separator is assumed to consist of a single comma (,). If you specify NoSeparator or an empty string, then no separator will be used.

- Suffix=string
  NoSuffix

  Specifies a suffix string that ends any column value in the input file. If you omit this option, the column suffix is assumed to consist of a quotation mark ("). If you specify NoSuffix or an empty string, then no suffix will be used.

- Terminator=string

  Specifies the row terminator that completes all the column values corresponding to a row. If you omit this option, the row terminator is assumed to be the end of the line.

- Null=string
  NoNull

  Specifies a string that, when found in the database column, is unloaded as "NULL" in the output file.

  The Null option can be specified *on the command line* as one of the following:

  * A quoted string with an alternate replacement string
  * An empty string ("") or NoNull to request no replacement string

  If provided, the string that represents the null character must be quoted on the Oracle RMU command line, however, it *must not* be quoted in the input file. You cannot specify a blank space or spaces as the null character.

  If the final column or columns of a record are to be set to NULL, you only have to specify data for the column up to the last non-null column.

See the Examples section for an example of each of these methods of storing the NULL value.

Note

The values of each of the strings specified in the delimiter options must be enclosed by quotation marks. Oracle RMU strips these quotation marks while interpreting the values. If you want to specify a quotation mark (") as a delimiter, specify a string of four quotation marks. Oracle RMU interprets four quotation marks as your request to use one quotation mark as a delimiter. For example, Suffix = """".

Oracle RMU reads the quotation marks as follows:

- The first quotation mark is stripped from the string.
- The second and third quotation marks are interpreted as your request for one quotation mark (") as a delimiter.
- The fourth quotation mark is stripped. This results in one quotation mark being used as a delimiter.

Furthermore, if you want to specify a quotation mark as part of the delimiter string, you must use two quotation marks for each quotation mark that you want to appear in the string. For example, Suffix = "**""** causes Oracle RMU to use a delimiter of **"**.

A delimiter of blank spaces enclosed in quotes is not valid.

- Place_Only=sorted-placement-file

  Allows you to sort the input file and create an output file sorted in Placement order.

  The input file can first be sorted into Placement order by using the Place_Only option. The resultant file can then be loaded with the Commit_Every qualifier to gain the required efficiency. Do not use this option with a parallel load operation; parallel load operations perform best when the input file is not sorted.

  The Place_Only option cannot be used with either the Commit_Every qualifier or the Exception_File option (data is not being stored in the database). However, the Place_Only option requires the Place qualifier be specified (to sort the data).

  The placement-sorted output file has the default file extension of .unl.

Unless you specify the Null option (with the Format=Delimited_Text parameter of the Record_Definition qualifier), any null values stored in the rows of the tables being loaded are not preserved. Therefore, use the Null option if you want to preserve null values stored in tables and you are moving data within the database or between databases.

See the examples in Section 1.68 for more information.

Rms_Record_Def=({File=name | Path=name}[,options])
Synonymous with the Record_Definition qualifier. See the description of the Record_Definition qualifier.

Replace_Rows
If a PRIMARY KEY exists for the table, then the Replace_Rows qualifier will instruct Oracle Rdb to delete the row matching the primary key columns prior to replacing the whole row with new values from the source data file. If no PRIMARY KEY exists for the table, then the Replace_Rows qualifier is ignored (i.e. no delete will be performed before inserting new data).

See Example 28 below.

Please refer to the SQL REPLACE Statement for a complete description of the affects of the /Replace_Rows qualifier.

Restricted_Access
NoRestricted_Access
Allows a single process to load data and enables some optimizations available only when restricted access is in use. The default is Norestricted_Access.

If you are loading a table from an RMU Unload file which contains LIST OF BYTE VARYING data, the Restricted_Access qualifier reserves the LIST areas for EXCLUSIVE access. This reduces the virtual memory used by long transactions during a load operation and also eliminates I/O to the snapshot files for the LIST storage areas.

The Restricted_Access and Parallel qualifiers are mutually exclusive and cannot be specified together on the same RMU Load command line or within a plan file. While RMU Load is running with the Restricted_Access qualifier specified, no other user can attach to the database.

Row_Count=n
Specifies that Oracle Rdb buffer multiple rows between the Oracle Rdb server and the RMU Load process. The default for *n* is 500 rows; however, this value should be adjusted based on working set size and length of loaded data. Increasing the row count may reduce the CPU cost of the load operation. For remote databases, this may significantly reduce network traffic for large

volumes of data because the buffered data can be packaged into larger network packets.

The minimum value you can specify for *n* is 1. The default row size is the value specified for the Commit_Every qualifier or 500, whichever is smaller.

Since=date-time

The /Since qualifier is used in conjunction with the RMU/LOAD/AUDIT command. This qualifier, along with the /Before qualifier, allow audit records to be filtered by timestamp. The qualifiers BEFORE and SINCE can specify the date/time range which will be extracted from the OpenVMS audit journal and saved in the target auditing table.

These qualifiers accept the standard OpenVMS date/time specification that includes special keywords such as YESTERDAY, TODAY and TOMORROW. These values can be very effective when used with the List_Plan qualifier and used later when using RMU Load Plan.

If these qualifiers are omitted, then their values default to minimum and maximum possible date/time values.

This example shows the use of DCL symbols to be used at runtime to provide the date/time range.

```
$  RMU/LOAD/AUDIT -
    /SINCE=&start_ts -
    /BEFORE=&end_ts -
   TESTDB AUDIT_RECORDS -
   SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
%RMU-I-DATRECREAD,  91 data records read from input file.
%RMU-I-DATRECSTO,   63 data records stored 27-NOV-2013 00:59:33.18.
$
```

This example uses explicit date and time values to load a specific range of audit record.

```
$  RMU/LOAD/AUDIT -
    /SINCE=1-jan-2011 -
    /BEFORE="1-nov-2013 13:00" -
   TESTDB AUDIT_RECORDS -
   SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
%RMU-I-DATRECREAD,  91 data records read from input file.
%RMU-I-DATRECSTO,   0 data records stored 27-NOV-2013 00:59:33.75.
$
```

Skip=n
Noskip
Ignores the first *n* data records in the input file. Use this qualifier in conjunction with the Commit_Every qualifier when restarting an aborted load operation. An aborted load operation displays a message indicating how many records have been committed. Use this value for *n*. If you specify a negative number, you receive an error message. If you specify a number greater than the number of records in the file, you receive an error message stating that no records have been stored. If you do not specify a value, you receive an error message stating that there is a missing keyword value.

Using the Skip qualifier to restart an aborted parallel load operation is rarely useful. Because records are sorted by the controller for each executor involved in the parallel load, there are usually multiple sections of loaded and unloaded records in the input file. Unless you are very familiar with the data you are loading and how it is sorted by the controller, you risk loading some records twice and not loading other records at all, if you use the Skip qualifier when restarting an aborted parallel load operation.

The default is the Noskip qualifier.

Statistics=(stat-opts)
Specifies that statistics are to be displayed at regular intervals or each time a transaction commits, or both, so that you can evaluate the progress of the load operation.

The stat-opts are the options you can specify with this qualifier, namely: Interval=n, On_Commit, or both. If the Statistics qualifier is specified, you must also specify at least one option.

When the Statistics=(Interval=n) qualifier is specified, Oracle RMU prints statistics every *n* seconds. The minimum value for *n* is 1.

When the Statistics=(On_Commit) qualifier is specified, Oracle RMU prints statistics each time a transaction is committed.

If you specify both options, Statistics=(Interval=n, On_Commit), statistics are displayed every *n* seconds and each time a transaction commits.

The displayed statistics include:

- Elapsed time
- CPU time
- Buffered I/O
- Direct I/O

- Page faults
- Number of records loaded when the last transaction was committed
- Number of records loaded so far in the current transaction
- If the Record_Definition=Exception_File option is also specified, the following statistics are displayed also:
  - Number of records rejected when the last transaction was committed
  - Number of records rejected so far in the current transaction
- If the Parallel qualifier is specified also, the following statistics are displayed also:
  - Number of extra commits performed by executors

    Extra commits are caused when the Oracle RMU directs your process or the executors to commit a transaction earlier than usual to avoid a hung load operation. For example, if one executor is holding, but no longer needs a lock that another executor requires, Oracle RMU directs the first executor to commit its current transaction. By directing an executor or executors to commit a transaction earlier than usual, the locks under contention are released and the load operation can proceed.
  - The total number of executors
  - The number of executors that are initializing, idle, terminated, sorting, storing, committing, or executing

At any time during the load operation, you can press Ctrl/T to display the current statistics.

Transaction_Type=share-mode

Specifies the share mode for the load operation. The following share modes are available:

Batch_Update
Exclusive
Protected
Shared

You must specify a value if you use the Transaction_Type qualifier. If you do not specify the Transaction_Type qualifier, the default share mode is Protected.

If you specify a parallel load operation (with the Parallel qualifier), and constraints are defined on the table you are loading, Oracle Corporation recommends that you specify the Shared share mode, or drop the constraints

prior to starting a parallel load operation, or specify the Noconstraints qualifier. See the Usage Notes for details.

Trigger_Relations[=(table-name-list)]
Notrigger_Relations
You can use the Trigger_Relations qualifier in three ways:

- Trigger_Relations=(table-name-list)

  Specifies the tables to be reserved for update. Using this qualifier, you can explicitly lock tables that are updated by triggers in store operations. If you list multiple tables, separate the table names with a comma, and enclose the list of table names within parentheses.

- Trigger_Relations

  If you omit the list of table names, the tables updated by triggers are locked automatically as required. This is the default.

- NoTrigger_Relations

  Disables triggers on the target table. This option requires DROP privilege on the table being loaded. You cannot specify a list of table names with this option.

If you specify a parallel load operation (with the Parallel qualifier), and triggers are defined on the table you are loading, Oracle Corporation recommends that you specify the Shared share mode or drop the triggers prior to starting a parallel load operation. See the Usage Notes for details.

The Trigger_Relations qualifier can be used with indirect file references. See Section 1.3 for more information.

Virtual_Fields=([No]Automatic)
Novirtual_Fields
The Virtual_Fields qualifier is required to reload any AUTOMATIC (or IDENTITY) fields with real data.

The Novirtual_Fields qualifier is the default, which is equivalent to the Virtual_Fields=(Noautomatic) qualifier.

If you specify the Virtual_Fields qualifier without a keyword, all fields are loaded except COMPUTED BY columns and calculated VIEW columns.

Use this qualifier when restructuring a table and when you do not wish the AUTOMATIC INSERT AS or IDENTITY column to recompute new values. Instead, RMU will reload the saved values from a file created by RMU/UNLOAD/VIRTUAL_FIELDS=AUTOMATIC.

1.31 RMU Load Command

Usage Notes

- To use the RMU Load command for a database, you must have the RMU$LOAD privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege. The appropriate Oracle Rdb privileges for accessing the database tables involved are also required.
- To use the RMU Load command with the Audit qualifier, you must have both of the following:
  - The RMU$SECURITY privilege in the root file ACL for the database whose security audit records are being loaded
  - The RMU$LOAD privilege in the root file ACL for the database into which these security audit records are being loaded

  If you do not have both of the privileges described in the preceding list, you must have the OpenVMS SYSPRV or BYPASS privilege.
- You can unload a table from a database structured under one version of Oracle Rdb and load it into the same table of a database structured under another version of Rdb. For example, if you unload the EMPLOYEES table from a mf_personnel database created under Oracle Rdb V6.0, you can load the generated .unl file into an Oracle Rdb V7.0 database. Likewise, if you unload the EMPLOYEES table from a mf_personnel database created under Oracle Rdb V7.0, you can load the generated .unl file into an Oracle Rdb V6.1 database. This is true even for specially formatted binary files (created with the RMU Unload command without the Record_Definition qualifier). The earliest version into which you can load a .unl file from another version is Oracle Rdb V6.0.
- The following list provides information on parallel load operations:
  - Specify no more executors (with the Executor_Count option to the Parallel qualifier) than storage areas defined for the table you are loading.
  - You cannot use a parallel load operation to load list data (segmented string) records or security audit records. If you specify a parallel load operation and attempt to load list data or security audit records, Oracle RMU returns a warning and performs a single-executor load operation.

- Oracle Corporation recommends that you specify a shared mode transaction type or specify the Noconstraints qualifier and drop triggers during a parallel load operation; otherwise, constraints and triggers defined on the table you are loading can cause lock conflicts among the parallel load executors.
- If you are using parallel load and hashed indexes, do not sort the data prior to loading it. Instead, use the Place qualifier to the RMU Load command to sort the data as it is loaded. (The Place qualifier is useful for hashed indexes, not sorted.)

• The following list provides information on loading security audit journals:

- Loading security audit journals into a database other than that which is being audited

  When you load the security audit journals recorded for one database into another database, you specify the database that is being audited as a parameter to the Audit=Database_File qualifier, and you specify the database into which these security audit records should be loaded with the root-file-spec parameter to the Oracle RMU command.

  For instance, the following example loads the security audit journal records for the mf_personnel database into the MFP_AUDIT table of the audit_db database. Note that SECURITY_AUDIT is a logical name that points to the actual security audit journal file.

```
$ RMU/LOAD/AUDIT=DATABASE_FILE=MF_PERSONNEL AUDIT_DB -
_$  MFP_AUDIT SECURITY_AUDIT
```

  When you issue the preceding RMU Load command, the audit_db database must exist. However, the RMU Load command creates the MFP_AUDIT table in the audit_db database and appropriately defines the columns for the MFP_AUDIT database.

  In other words, the following SQL statement satisfies the minimum requirements for the audit_db database to be used correctly by the preceding RMU Load command:

```
SQL> CREATE DATABASE FILENAME audit_db.rdb;
```

  Note that there is no field in the audit record loaded by Oracle RMU to indicate the source database for the records. Therefore, it is not wise to mix auditing records from different databases in the same table. Instead, auditing information for different databases should be loaded into separate tables.

- Security audit journal file name

The name of the current security audit journal file is SYS$MANAGER:SECURITY.AUDIT$JOURNAL.

- Loading security audit journals into the database being audited

  The Oracle Rdb table into which you load the security audit journal records should be defined with the columns shown in Table 1–13 under the column marked Oracle Rdb Column Name so that the audit journal records can be loaded successfully into the table. If the table does not exist, the RMU Load Audit command creates it with the columns shown in Table 1–13 under the column marked Oracle Rdb Column Name. You can give the table any valid name.

- Table 1–13 lists the column names created by the RMU Load command with the Audit qualifier.

Table 1–13 Columns in a Database Table for Storing Security Audit Journal Records

Oracle Rdb Column Name	SQL Data Type and Length
AUDIT$EVENT	CHAR 16
AUDIT$SYSTEM_NAME	CHAR 15
AUDIT$SYSTEM_ID	CHAR 12
AUDIT$TIME_STAMP	CHAR 48
AUDIT$PROCESS_ID	CHAR 12
AUDIT$USER_NAME	CHAR 12
AUDIT$TSN	CHAR 25[1]
AUDIT$OBJECT_NAME	CHAR 255
AUDIT$OBJECT_TYPE	CHAR 12
AUDIT$OPERATION	CHAR 32
AUDIT$DESIRED_ACCESS	CHAR 16
AUDIT$SUB_STATUS	CHAR 32
AUDIT$FINAL_STATUS	CHAR 32
AUDIT$RDB_PRIV	CHAR 16
AUDIT$VMS_PRIV	CHAR 16
AUDIT$GRANT_IDENT	CHAR 192

[1]Prior to Oracle Rdb V7.0, RDBVMS$TSN data type and length was CHAR 12.

(continued on next page)

Table 1–13 (Cont.) Columns in a Database Table for Storing Security Audit Journal Records

Oracle Rdb Column Name	SQL Data Type and Length
AUDIT$NEW_ACE	CHAR 192
AUDIT$OLD_ACE	CHAR 192
AUDIT$RMU_COMMAND	CHAR 512

- Dates stored in ASCII text format can be converted to the VMS DATE data type format by the RMU Load command. See Example 7 in the Examples section, which demonstrates this conversion.
- To preserve the NULL indicator in a load or unload operation, specify the Null option when you use the Record_Definition qualifier. Using the Record_Definition qualifier without the Null option causes the RMU Load command to replace all NULL values with zeros. This can cause unexpected results with computed-by columns.
- When the RMU Load command is issued for a closed database, the command executes without other users being able to attach to the database.
- The RMU Load command recognizes character set information. When you load a table, the RMU Load command recognizes that the correct size of a column is based on its character set. For example, the RMU Load command recognizes that a column defined as CHAR (10) CHARACTER SET KANJI occupies 20 octets.
- By default, the RMU Load command changes any table or column names that you specify to uppercase. To preserve lowercase characters, use delimited identifiers; that is, enclose the names in quotation marks ("").
- Oracle RMU does not support the multischema naming convention and returns an error if you specify one. For example:

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME)                    -
_$ /RECORD_DEFINITION=(FILE=TEXT_NAMES,EXCEPTION_FILE=FILE.UNL)   -
_$ corporate_data ADMINISTRATION.PERSONNEL.EMPLOYEES EMP.UNL
%RDB-E-BAD_DPB_CONTENT, invalid database parameters in the database
 parameter block (DPB)
%RMU-I-DATRECSTO, 0 data records stored
%RMU-I-DATRECREJ, 0 data records rejected.
```

1.31 RMU Load Command

When using a multischema database, you must specify the SQL stored name for the database object. For example, to find the stored name that corresponds to the ADMINISTRATION.PERSONNEL.EMPLOYEES table in the corporate_data database, issue an SQL SHOW TABLE command.

```
SQL> SHOW TABLE ADMINISTRATION.PERSONNEL.EMPLOYEES
Information for table ADMINISTRATION.PERSONNEL.EMPLOYEES
    Stored name is EMPLOYEES
   .
   .
   .
```

Then, to load the table, issue the following RMU Load command:

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME)                  -
_$ /RECORD_DEFINITION=(FILE=TEXT_NAMES,EXCEPTION_FILE=FILE.UNL) -
_$ CORPORATE_DATA EMPLOYEES MY_DATA.UNL
%RMU-I-DATRECSTO, 3 data records stored
%RMU-I-DATRECREJ, 0 data records rejected.
```

The Fields qualifier can be used with indirect file references. When you use an indirect file reference in the field list, the referenced file is written to SYS$OUTPUT if the DCL SET VERIFY comand has been used. See Section 1.3 for more information.

- The Transaction_Type=Batch_Update qualifier cannot be used with multiple executors (Executor_Count greater than 1).
- The RMU Load procedure supports the loading of tables that reference system domains.
- If you use a synonym to represent a table or a view, the RMU Load command translates the synonym to the base object and processes the data as though the base table or view had been named. This implies that the unload interchange files (.UNL) or record definition files (.RRD) that contain the table metadata will name the base table or view and not use the synonym name. If the metadata is used against a different database, you may need to use the Match_Name qualifier to override this name during the RMU load process.

Examples

Example 1

This command loads the data from the RMS file, names.unl, into the newly created RETIREES table of the mf_personnel database. The record structure of RETIREES is in the file names.rrd. The names.unl and names.rrd files were created by a previous RMU Unload command. The unload operation unloaded data from a view derived from a subset of columns in the EMPLOYEES table.

```
$ RMU/LOAD/RECORD_DEFINITION=FILE=NAMES.RRD -
_$ MF_PERSONNEL RETIREES NAMES.UNL
```

Example 2

This command restarts an aborted load operation that was loading the newly created RETIREES table of the mf_personnel database from the names.unl file. The columns being loaded are EMPLOYEE_ID, LAST_NAME, and FIRST_ NAME. The original load operation had committed 25 records. Beginning with the 26th record, the restarted load operation commits the transaction at every record until it reaches the original point of failure.

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME) -
_$ /COMMIT_EVERY=1/SKIP=25 MF_PERSONNEL RETIREES NAMES.UNL
```

Example 3

This example loads a new table, PENSIONS, into the mf_personnel database by using record definitions located in the data dictionary.

This example assumes that you have first defined a temporary view, TEMP_PENSIONS, combining appropriate columns of the EMPLOYEES and SALARY_HISTORY tables. You must also create a permanent table, PENSIONS, into which you will load the data.

Unload the TEMP_PENSIONS view by using the RMU Unload command with the Record_Definition=File=name qualifier to create both an .rrd file containing the column definitions and a data.unl file containing the data from the TEMP_ PENSIONS view. Load the new record definitions from the pensions.rrd file into the data dictionary by using the @ command at the CDO prompt. Then you can load the data into the PENSIONS table of the mf_personnel database by using the RMU Load command.

1.31 RMU Load Command

```
$ RMU/UNLOAD/RECORD_DEFINITION=FILE=PENSIONS.RRD MF_PERSONNEL -
_$ TEMP_PENSIONS DATA.UNL
$ DICTIONARY OPERATOR
Welcome to CDO V7.0
The CDD/Repository V7.0 User Interface
Type HELP for help
CDO> @PENSIONS.RRD
CDO> EXIT
$ RMU/LOAD/RECORD_DEFINITION=PATH=PENSIONS MF_PERSONNEL PENSIONS -
_$ DATA.UNL
```

Example 4

The following command loads the audit records for the mf_personnel database from the security audit journal file into the AUDIT_TABLE table in the mf_ personnel database. Note that if the AUDIT_TABLE table does not exist, the RMU Load command with the Audit qualifier creates it with the columns shown in Table 1–13.

```
$ RMU/LOAD/AUDIT MF_PERSONNEL AUDIT_TABLE -
_$ SYS$MANAGER:SECURITY.AUDIT$JOURNAL
%RMU-I-DATRECREAD,  12858 data records read from input file.
%RMU-I-DATRECSTO,   27 data records stored.
```

Example 5

The following command loads the audit records for the mf_personnel database from the security audit journal file into the AUDIT_TABLE table into the audit_db database. Note that the AUDIT_TABLE table is not created when the database is created. In this case, the RMU Load command with the Audit=Database_File qualifier creates it with the columns shown in Table 1–13.

```
$ RMU/LOAD/AUDIT=DATABASE_FILE=MF_PERSONNEL AUDIT_DB AUDIT_TABLE -
_$ SYS$MANAGER:SECURITY.AUDIT$JOURNAL
```

Example 6

This example loads a new table, COLLEGES, into the mf_personnel database by using record definitions located in the data dictionary. A commit operation occurs after every record is stored. The Log_Commits qualifier prints a message after each commit operation.

```
$ RMU/LOAD/RECORD_DEFINITION=FILE=COLLEGES.RRD /COMMIT_EVERY=1 -
_$ /LOG_COMMIT MF_PERSONNEL COLLEGES RMU.UNL
%RMU-I-DATRECSTO, 1 data records stored
%RMU-I-DATRECSTO, 2 data records stored
%RMU-I-DATRECSTO, 3 data records stored
%RMU-I-DATRECSTO, 4 data records stored
%RMU-I-DATRECSTO, 4 data records stored
$
```

Example 7

The following example shows how a date stored in the .unl file as 16-character collatable text can be converted to VMS DATE format when loaded into the database by using the RMU Load command. (The form of the .unl date is *yyyymmddhhmmsscc*, whereas the form of the VMS DATE is *dd-mmm-yyyy:hh:mm:ss.cc*. In both cases, *y* is the year, *m* is the month, *d* is the day, *h* is the hour, *m* is the minute, *s* is the second, and *c* is hundredths of a second. However in the .unl format, the month is expressed as an integer, whereas in the VMS DATE format the month is expressed as a 3-character string.)

The example assumes that the default SYS$LANGUAGE is ENGLISH.

```
SQL> --
SQL> -- Show the definition of the TEST table, in which the
SQL> -- COL1 column is the VMS DATE data type:
SQL> --
SQL> SHOW TABLE DATETEST;

Columns for table DATETEST:
Column Name                    Data Type        Domain
-----------                    ---------        ------
COL1                           DATE VMS

   .
   .
   .
$ !
$ ! Show the .unl file that will be loaded into the TEST table:
$ !
$ TYPE TEST.UNL
$ !
1991060712351212
$ !
$ ! Note that the .rrd file shows a data type of TEXT of 16
$ ! characters.  These 16 characters are the number of characters
$ ! specified for the date in the test.unl file:
$ !
```

```
$ TYPE TEST.RRD
DEFINE FIELD COL1 DATATYPE IS text size is 16.
DEFINE RECORD TEST.
   COL1 .
END TEST RECORD.
$ !
$ ! Load the data in test.unl into the DATETEST table:
$ !
$ RMU/LOAD/RMS=FILE=TEST.RRD TEST.RDB DATETEST TEST.UNL
%RMU-I-DATRECREAD,  1 data records read from input file.
%RMU-I-DATRECSTO, 1 data records stored.
$ !
$ SQL
SQL> ATTACH 'FILENAME TEST';
SQL> SELECT * FROM DATETEST;
 COL1
  7-JUN-1991 12:35:12.12
1 row selected
```

Example 8

The following example shows how a date stored in the .unl file as 22-character collatable text can be converted to TIMESTAMP format when loaded into the database by using the RMU Load command. The correct format for the .unl TIMESTAMP value is *yyyy-mm-dd:hh:mm:ss.cc*, where y,m,d,h,m,s,and c represent the same elements of the date and time format as described in Example 7.

This example also shows the use of an exception file to trap data that cannot be stored.

```
$ ! Create a column in the mf_personnel database with a
$ ! TIMESTAMP datatype:
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> CREATE TABLE NEWTABLE (COL1 TIMESTAMP);
```

```
SQL> SHOW TABLE (COLUMN) NEWTABLE;
Information for table NEWTABLE
Columns for table NEWTABLE:
Column Name                     Data Type        Domain
-----------                     ---------        ------
COL1                            TIMESTAMP(2)

SQL> COMMIT;
SQL> EXIT
$ !
$ ! Create a .unl file with the data you want to load.  Note that
$ ! the second value is a valid TIMESTAMP specification, the first
$ ! value is not.
$ !
$ CREATE TEST.UNL
06-14-1991:12:14:14.14
1991-06-14:12:14:14.14

$ !
$ ! Create an .rrd file that defines the TIMESTAMP field
$ ! as a TEXT field:
$ !
$ CREATE TEST.RRD
DEFINE FIELD COL1 DATATYPE IS TEXT SIZE 22.
DEFINE RECORD NEWTABLE.
    COL1.
END NEWTABLE RECORD.
$ !
$ ! Attempt to load the data in the .unl file. Oracle RMU returns an
$ ! error on the first data record because the date was incorrectly
$ ! specified.  The first record is written to the exception file,
$ ! BAD.DAT.
$ !

$ RMU/LOAD/RMS=(FILE=TEST.RRD,EXCEPT=BAD.DAT) MF_PERSONNEL.RDB -
_$ NEWTABLE TEST.UNL
%RMU-I-LOADERR, Error loading row 1.
%RDB-E-CONVERT_ERROR, invalid or unsupported data conversion
-COSI-F-IVTIME, invalid date or time
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO, 1 data records stored.
%RMU-I-DATRECREJ, 1 data records rejected.
$ !
$ ! Type BAD.DAT to view the incorrect data record
$ !
$ TYPE BAD.DAT
06-14-1991:12:14:14.14
$ !
$ ! Fetch the data record that stored successfully.
$ !
```

1.31 RMU Load Command

```
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SELECT * FROM NEWTABLE;
 COL1
 1991-06-14:12:14:14.14
1 rows selected
```

Example 9

Using the RMU Load command, you can load a table in a database by placing the fields in a different order in the database than they were in the input file.

The jobs.unl file contains the following:

```
000001000000000190001Rdb Demonstrator    DEMO
```

The jobs.rrd file contains the following:

```
DEFINE FIELD J_CODE   DATATYPE IS TEXT SIZE IS 4.
DEFINE FIELD WAGE_CL  DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD J_TITLE  DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD MIN_SAL  DATATYPE IS TEXT SIZE 10.
DEFINE FIELD MAX_SAL  DATATYPE IS TEXT SIZE 10.
DEFINE RECORD JOBS.
   MIN_SAL.
   MAX_SAL.
   WAGE_CL.
   J_TITLE.
   J_CODE.
END JOBS RECORD.
```

The JOBS table has the following structure:

```
Columns for table JOBS:
Column Name                    Data Type        Domain
-----------                    ---------        ------
JOB_CODE                       CHAR(4)          JOB_CODE_DOM
WAGE_CLASS                     CHAR(1)          WAGE_CLASS_DOM
JOB_TITLE                      CHAR(20)         JOB_TITLE_DOM
MINIMUM_SALARY                 INTEGER(2)       SALARY_DOM
MAXIMUM_SALARY                 INTEGER(2)       SALARY_DOM
```

Notice that:

- The ordering of the columns is different for the JOBS table in the database and in the input RMS file.
- The names in the .rrd file are also different from the names in the database.
- The data types of the salary fields are different (Oracle Rdb will do the conversion).

To load the RMS file correctly, you must use the following command:

```
$ RMU/LOAD MF_PERSONNEL JOBS JOBS/RMS=FILE=JOBS -
_$ /FIELDS=(MINIMUM_SALARY,MAXIMUM_SALARY,WAGE_CLASS,JOB_TITLE, -
_$ JOB_CODE)
```

Notice that the Fields qualifier uses the names of the columns in the JOBS table (not the field names in the .rrd file), but in the order of the RMS file.

The names in the .rrd file are immaterial. The purpose of the Fields qualifier is to load the first field in the RMS file into the MINIMUM_SALARY column of the JOBS table, load the second field in the RMS file into the MAXIMUM_ SALARY column of the JOBS table, and so forth.

The results:

```
SQL> SELECT * FROM JOBS WHERE JOB_CODE = 'DEMO';

JOB_CODE  WAGE_CLASS  JOB_TITLE          MINIMUM_SALARY   MAXIMUM_SALARY
DEMO      1           Rdb Demonstrator   $10,000.00       $19,000.00
```

Example 10

The following example shows the sequence of steps used to sort a file into placement order by using the Place qualifier and the Place_Only option and then to load the file by using the Commit_Every qualifier:

```
$ RMU/LOAD/PLACE -
_$ /RECORD_DEFINITION=(FILE=NAMES.RRD,PLACE_ONLY=PLACED_NAMES) -
_$ MF_PERSONNEL EMPLOYEES UNLOADED_NAMES.UNL

$ RMU/LOAD/RECORD_DEFINITION=(FILE=NAMES.RRD) -
_$ /COMMIT_EVERY=30 MF_PERSONNEL -
_$ EMPLOYEES PLACED_NAMES.UNL
%RMU-I-DATRECREAD,  100 data records read from input file.
%RMU-I-DATRECSTO,   100 data records stored.
```

Example 11

The following example requests that statistics be displayed at a regular interval of every minute. It loads the data from the RMS file, names.unl, into the EMPLOYEES table of the mf_personnel database. The record structure of EMPLOYEES is in the file names.rrd. The names.rrd file was created by a previous RMU Unload command that unloaded data from a subset of columns in the EMPLOYEES table.

```
$ RMU/LOAD/STATISTICS=(INTERVAL=60) -
_$ /RECORD_DEFINITION=(FILE=NAMES) -
_$ /FIELDS=(EMPLOYEE_ID, LAST_NAME) -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
```

Example 12

The following example uses the Exception_File option to the Record_Definition qualifier to tell Oracle RMU the name of the file to hold the exception records. Oracle RMU returns informational messages to alert you to any data records rejected.

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME) -
_$ /RECORD_DEFINITION=(FILE=TEXT_NAMES,EXCEPTION_FILE=FILE.UNL) -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-LOADERR, Error loading row 1.
%RDB-E-NO_DUP, index field value already exists; duplicates not
 allowed for EMPLOYEES_HASH
%RMU-I-LOADERR, Error loading row 17.
%RDB-E-NO_DUP, index field value already exists; duplicates not
 allowed for EMPLOYEES_HASH
%RMU-I-LOADERR, Error loading row 33.
%RDB-E-NO_DUP, index field value already exists; duplicates not
 allowed for EMPLOYEES_HASH
%RMU-I-LOADERR, Error loading row 155.
%RDB-E-NO_DUP, index field value already exists; duplicates not
 allowed for EMPLOYEES_HASH
%RMU-I-DATRECREAD,  200 data records read from input file.
%RMU-I-DATRECSTO,   196 data records stored.
%RMU-I-DATRECREJ,   4 data records rejected.
```

Example 13

The following is an example of the format in which you can provide input data to the RMU Load command when you use the Format=Delimited_Text option with the Record_Definition qualifier. This is followed by the RMU Load command you use to load this data.

```
"99997","ABUSHAKRA","CAROLINE","S","5 CIRCLE STREET","BOX 506",
"CHELMSFORD", "MA", "02184", "1960061400000000"#
"99996","BRADFORD","LEO","M","4 PLACE STREET","BOX 555",
"NASHUA","NH", "03060", "1949051800000000"#



$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= NAMES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#" ) -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO,   2 data records stored.
```

Example 14

The following is an example of the format in which you must provide input data to the RMU Load command when you specify the Format=Text option with the Record_Definition qualifier. This is followed by the RMU Load command you use to load this data.

```
09166Watts         Leora     F
09190Margolis      David     M
09187McDonald      Lois      F

$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, SEX) -
_$ /RECORD_DEFINITION=(FILE=TEXT_NAMES.RRD, FORMAT=TEXT) -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-DATRECREAD,  3 data records read from input file.
%RMU-I-DATRECSTO,   3 data records stored.
```

Example 15

The following example assumes you want to load a data file into the JOBS table that contains more fields than the table definition in the mf_personnel database. The example first attempts to do this by just excluding the extra field from the list associated with the Fields qualifier. However, this causes an error to be returned. The example then uses the FILLER keyword in the .rrd file to tell Oracle RMU not to attempt to load the additional field. The command executes successfully.

The table definition for the JOBS table is as follows:

```
Columns for table JOBS:
Column Name                    Data Type        Domain
-----------                    ---------        ------
JOB_CODE                       CHAR(4)          JOB_CODE_DOM
 Primary Key constraint JOBS_PRIMARY_JOB_CODE
WAGE_CLASS                     CHAR(1)          WAGE_CLASS_DOM
JOB_TITLE                      CHAR(20)         JOB_TITLE_DOM
MINIMUM_SALARY                 INTEGER(2)       SALARY_DOM
MAXIMUM_SALARY                 INTEGER(2)       SALARY_DOM
```

The .rrd file for the data you want to load appears as follows (note that there is no corresponding field to JOB_STATUS in the mf_personnel database definition for the JOBS table):

```
DEFINE FIELD JOB_CODE DATATYPE IS TEXT SIZE IS 4.
DEFINE FIELD WAGE_CLASS DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD JOB_TITLE DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD MINIMUM_SALARY DATATYPE IS TEXT SIZE IS 13.
DEFINE FIELD MAXIMUM_SALARY DATATYPE IS TEXT SIZE IS 13.
DEFINE FIELD JOB_STATUS DATATYPE IS TEXT SIZE IS 4.
DEFINE RECORD JOBS.
   JOB_CODE .
   WAGE_CLASS .
   JOB_TITLE .
   MINIMUM_SALARY .
   MAXIMUM_SALARY .
   JOB_STATUS .
END JOBS RECORD.
```

The data file you want to load, jobs.unl, appears as follows:

```
DBAD4Corp Db Administratr55000.00     95000.00     Old
```

You attempt to load the file in the mf_personnel database by listing only the fields in the RMU Load command that have corresponding fields defined in the database:

```
$ RMU/LOAD MF_PERSONNEL/RMS=(FILE=JOBS.RRD, FORMAT=TEXT) -
_$ /FIELDS=(JOB_CODE, WAGE_CLASS, JOB_TITLE, MINIMUM_SALARY, -
_$ MAXIMUM_SALARY) JOBS JOBS.UNL
%RMU-F-FLDMUSMAT, Specified fields must match in number and datatype
with the unloaded data
%RMU-I-DATRECSTO, 0 data records stored
```

The workaround for the problem of a mismatch between your data and .rrd file, and database definition for a table is to use the FILLER keyword in your .rrd file, as follows:

```
DEFINE FIELD JOB_CODE DATATYPE IS TEXT SIZE IS 4.
DEFINE FIELD WAGE_CLASS DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD JOB_TITLE DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD MINIMUM_SALARY DATATYPE IS TEXT SIZE IS 13.
DEFINE FIELD MAXIMUM_SALARY DATATYPE IS TEXT SIZE IS 13.
DEFINE FIELD JOB_STATUS DATATYPE IS TEXT SIZE IS 4 FILLER.  <------
DEFINE RECORD JOBS.
   JOB_CODE .
   WAGE_CLASS .
   JOB_TITLE .
   MINIMUM_SALARY .
   MAXIMUM_SALARY .
   JOB_STATUS .
END JOBS RECORD.
```

Now that the .rrd file has been modified, attempt to load the record again:

```
$ RMU/LOAD MF_PERSONNEL/RMS=(FILE=JOBS.RRD, FORMAT=TEXT) -
_$ /FIELDS=(JOB_CODE, WAGE_CLASS, JOB_TITLE, MINIMUM_SALARY, -
_$ MAXIMUM_SALARY) JOBS JOBS.UNL
%RMU-I-DATRECSTO, 1 data records stored.
```

Example 16

The following example demonstrates the use of the Null="*" option of the Record_Definition qualifier to signal to Oracle RMU that any data that appears as an unquoted asterisk in the .unl file should have the corresponding column in the database be flagged as NULL.

The example shows the contents of the .unl file, followed by the RMU Load command used to load this .unl file, and then the output from an SQL statement to display the data loaded.

```
"98888","ABUSHAKRA","CAROLINE",*,"5 CIRCLE STREET","BOX 506",
"CHELMSFORD", "MA", "02184", "1960061400000000"#
"98889","BRADFORD","LEO",*,"4 PLACE STREET","BOX 555", "NASHUA","NH",
"03060", "1949051800000000"#
```

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#", -
-$ NULL="*" ) -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO,   2 data records stored.
```

```
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SELECT * FROM EMPLOYEES WHERE EMPLOYEE_ID > '98000'
cont> AND MIDDLE_INITIAL IS NULL;
 EMPLOYEE_ID   LAST_NAME        FIRST_NAME   MIDDLE_INITIAL
   ADDRESS_DATA_1              ADDRESS_DATA_2         CITY
      STATE   POSTAL_CODE   SEX    BIRTHDAY      STATUS_CODE
 98888         ABUSHAKRA        CAROLINE     NULL
   5 CIRCLE STREET             BOX 506                CHELMSFORD
      MA      02184         ?      14-Jun-1960   N

 98889         BRADFORD         LEO          NULL
   4 PLACE STREET              BOX 555                NASHUA
      NH      03060         ?      18-May-1949   N

2 rows selected
```

Example 17

The following example demonstrates the use of the Null="" option of the Record_Definition qualifier to signal to Oracle RMU that any data that is an empty string in the .unl file (as represented by two commas with no space separating them) should have the corresponding column in the database be flagged as NULL.

The example shows the contents of the .unl file, followed by the RMU Load command used to load this .unl file, and then the output from an SQL statement to display the data loaded.

```
"90021","ABUSHAKRA","CAROLINE","A","5 CIRCLE STREET",,
"CHELMSFORD", "MA", "02184", "1960061400000000"#
"90015","BRADFORD","LEO","B","4 PLACE STREET",, "NASHUA","NH",
"03030", "1949051800000000"#

$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#", -
_$ NULL="") -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO,   2 data records stored.

$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SELECT * FROM EMPLOYEES WHERE ADDRESS_DATA_2 IS NULL;
 EMPLOYEE_ID   LAST_NAME        FIRST_NAME   MIDDLE_INITIAL
   ADDRESS_DATA_1              ADDRESS_DATA_2         CITY
      STATE   POSTAL_CODE   SEX    BIRTHDAY      STATUS_CODE
 90021         ABUSHAKRA        CAROLINE     A
   5 CIRCLE STREET             NULL                    CHELMSFORD
      MA      02184         ?      14-Jun-1960   N

 90015         BRADFORD         LEO          B
   4 PLACE STREET              NULL                    NASHUA
      NH      03030         ?      18-May-1949   N

2 rows selected
```

Example 18

The following example is the same as Example 17 except it shows the use of the default value for the Null option of the Record_Definition qualifier to signal to Oracle RMU that any data that is an empty string in the .unl file (as represented by two commas with no space separating them) should have the corresponding column in the database be flagged as NULL.

The example shows the contents of the .unl file, followed by the RMU Load command used to load this .unl file, and then the output from an SQL statement to display the data loaded.

```
"90022","ABUSHAKRA","CAROLINE","A","5 CIRCLE STREET",,
"CHELMSFORD", "MA", "02184", "1960061400000000"#
"90014","BRADFORD","LEO","B","4 PLACE STREET",, "NASHUA","NH",
"03030", "1949051800000000"#
```

```
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#", -
_$ NULL) -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO,   2 data records stored.
```

```
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SELECT * FROM EMPLOYEES WHERE EMPLOYEE_ID = '90022' OR
cont> EMPLOYEE_ID ='90014' AND ADDRESS_DATA_2 IS NULL;
 EMPLOYEE_ID   LAST_NAME        FIRST_NAME   MIDDLE_INITIAL
   ADDRESS_DATA_1              ADDRESS_DATA_2         CITY
      STATE   POSTAL_CODE   SEX    BIRTHDAY      STATUS_CODE
 90014         BRADFORD         LEO          B
   4 PLACE STREET              NULL                   NASHUA
      NH      03030         ?      18-May-1949   N

 90022         ABUSHAKRA        CAROLINE     A
   5 CIRCLE STREET             NULL                   CHELMSFORD
      MA      02184         ?      14-Jun-1960   N

2 rows selected
```

Example 19

The following example demonstrates the use of the Null option of the Record_ Definition qualifier to signal to Oracle RMU that any data that is an empty string in the .unl file (as represented by two commas with no space separating them) should have the corresponding column in the database be flagged as NULL. In addition, any column for which there is only data for the first column or columns has the remaining columns set to NULL.

The example shows the contents of the .unl file, followed by the RMU Load command used to load this .unl file, and then the output from an SQL statement to display the data loaded.

```
"90026","ABUSHAKRA","CAROLINE","A","5 CIRCLE STREET","BOX 783",
"CHELMSFORD","MA", "02184", "1960061400000000"
"90011","BRADFORD","LEO",,,, "NASHUA","NH","03030","1949051800000000"
"90010"
"90009",,,,,,,,,"1966061600000000"

$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ NULL) -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECREAD,  5 data records read from input file.
%RMU-I-DATRECSTO,   5 data records stored.
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SELECT * FROM EMPLOYEES WHERE EMPLOYEE_ID ='90026' OR
cont> EMPLOYEE_ID BETWEEN '90009' AND '90011';
 EMPLOYEE_ID   LAST_NAME        FIRST_NAME   MIDDLE_INITIAL
   ADDRESS_DATA_1              ADDRESS_DATA_2         CITY
      STATE   POSTAL_CODE   SEX    BIRTHDAY      STATUS_CODE
 90009         NULL             NULL         NULL
   NULL                        NULL                   NULL
      NULL    NULL          ?      16-Jun-1966   N

 90010         NULL             NULL         NULL
   NULL                        NULL                   NULL
      NULL    NULL          ?      NULL          N

 90011         BRADFORD         LEO          NULL
   NULL                        NULL                   NASHUA
      NH      03030         ?      18-May-1949   N

 90026         ABUSHAKRA        CAROLINE     A
   5 CIRCLE STREET             BOX 783                CHELMSFORD
      MA      NULL          ?      14-Jun-1960   N

4 rows selected
```

Example 20

The following example demonstrates a parallel load operation. In this example, three executors are specified because there are three storage areas in the JOB_HISTORY table of the mf_personnel database. The Defer_Index_Updates qualifier is used because there are no constraints or triggers defined on the JOB_HISTORY table, and it is known that no other database activity will occur when this command is executed.

In addition, a plan file is generated to capture the specification of this load operation. See the next example for a description of the plan file.

Note that the pid provided in the output from this command is the process ID.

```
$ RMU/LOAD/PARALLEL=(EXEC=3)/DEFER_INDEX_UPDATES mf_personnel.rdb -
_$ /RECORD_DEFINITION=(FILE=JOB_HIST,FORMAT=DELIMITED_TEXT, -
_$ EXCEPTION_FILE=DISK1:[ERRORS]JOB_HIST.EXC) -
_$ /STATISTICS=(INTERVAL=30)/LIST_PLAN=JOB_HISTORY.PLAN -
_$ JOB_HISTORY JOB_HIST.UNL
%RMU-I-EXECUTORMAP, Executor EXECUTOR_1 (pid: 2941941B) will load
 storage area EMPIDS_LOW.
%RMU-I-EXECUTORMAP, Executor EXECUTOR_2 (pid: 2941F01D) will load
 storage area EMPIDS_MID.
%RMU-I-EXECUTORMAP, Executor EXECUTOR_3 (pid: 2941C81F) will load
 storage area EMPIDS_OVER.

-------------------------------------------------------------------------
ELAPSED: 0 00:00:30.05 CPU: 0:00:01.64 BUFIO: 59 DIRIO: 219 FAULTS: 2670

 1640 data records read from input file.
  1330 records loaded before last commit.
  220 records loaded in current transaction.
  0 records rejected before last commit.
  0 records rejected in current transaction.
 26 early commits by executors.
  3 executors:  0 Initializing; 0 Idle; 0 Terminated
                0 Sorting; 2 Storing; 1 Committing; 0 Executing
-------------------------------------------------------------------------
   .
   .
   .
-------------------------------------------------------------------------
ELAPSED: 0 00:02:30.12 CPU: 0:00:02.94 BUFIO: 103 DIRIO: 227  FAULTS: 267
1
 8070 data records read from input file.
  7800 records loaded before last commit.
  210 records loaded in current transaction.
  0 records rejected before last commit.
  0 records rejected in current transaction.
139 early commits by executors.
  3 executors:  0 Initializing; 0 Idle; 0 Terminated
                0 Sorting; 1 Storing; 2 Committing; 0 Executing
-------------------------------------------------------------------------

%RMU-I-EXECSTAT0, Statistics for EXECUTOR_1:
%RMU-I-EXECSTAT1,   Elapsed time:  00:02:45.84    CPU time:       12.95
%RMU-I-EXECSTAT2,   Storing time:  00:00:45.99    Rows stored:    2440
%RMU-I-EXECSTAT3,   Commit time:   00:01:33.17    Direct I/O:     6623
%RMU-I-EXECSTAT4,   Idle time:     00:00:22.34    Early commits:  47

%RMU-I-EXECSTAT0, Statistics for EXECUTOR_2:
%RMU-I-EXECSTAT1,   Elapsed time:  00:02:48.42    CPU time:       18.10
%RMU-I-EXECSTAT2,   Storing time:  00:01:24.98    Rows stored:    4319
%RMU-I-EXECSTAT3,   Commit time:   00:01:18.13    Direct I/O:     9621
%RMU-I-EXECSTAT4,   Idle time:     00:00:01.03    Early commits:  29
```

```
%RMU-I-EXECSTAT0, Statistics for EXECUTOR_3:
%RMU-I-EXECSTAT1,   Elapsed time:  00:02:46.50    CPU time:       9.78
%RMU-I-EXECSTAT2,   Storing time:  00:00:11.12    Rows stored:    2293
%RMU-I-EXECSTAT3,   Commit time:   00:02:26.67    Direct I/O:     3101
%RMU-I-EXECSTAT4,   Idle time:     00:00:04.14    Early commits:  77

%RMU-I-EXECSTAT5,   Main process idle time:  00:02:41.06
%RMU-I-DATRECREAD,  9052 data records read from input file.
%RMU-I-DATRECSTO,   9052 data records stored.
%RMU-I-DATRECREJ,   0 data records rejected.
```

Example 21

The following command is the same as in the previous example, except the Noexecute qualifier is specified. Because this qualifier is specified, the load operation is not performed. However, the load plan file is created and verified.

```
$ RMU/LOAD/PARALLEL=(EXEC=3)/DEFER_INDEX_UPDATES/NOEXECUTE -
_$  mf_personnel.rdb -
_$ /RECORD_DEFINITION=(FILE=JOB_HIST,FORMAT=DELIMITED_TEXT, -
_$ EXCEPTION_FILE=DISK1:[ERRORS]JOB_HIST.EXC) -
_$ /STATISTICS=(INTERVAL=30)/LIST_PLAN=JOB_HISTORY.PLAN -
_$ JOB_HISTORY JOB_HIST.UNL
```

Example 22

The following display shows the contents of the plan file, JOB_HISTORY.PLAN, created in the preceding example. The following callouts are keyed to this display:

❶ The Plan Parameters include all the parameters specified on the RMU Load command line and all possible command qualifiers.

❷ Command qualifiers that are not specified on the command line are sometimes represented as comments in the plan file. This allows you to edit and adjust the plan file for future use.

❸ Command qualifiers that are not specified on the command line and for which there are defaults are sometimes represented with their default value in the plan file.

❹ Command qualifiers that are explicitly specified on the command line are represented in the plan file as specified.

❺ Executor Parameters are listed for each executor involved in the load operation. Like the command qualifiers, both the values you specify on the command line and those that are allowed but were not specified are included in this list of parameters.

❻ Note that the exception file extension is appended with the executor number. When you specify such files on the command line, Oracle RMU generates a separate file for each executor. If desired, you could edit this plan file to place each exception file on a different disk or directory.

```
! Plan created on 20-JUL-1995 by RMU/LOAD.

Plan Name = LOAD_PLAN
Plan Type = LOAD

Plan Parameters:❶
    Database Root File = MF_PERSONNEL.RDB;
    Table Name = JOB_HISTORY
    Input File = JOB_HIST.UNL

    ! Fields = <all> ❷
    Transaction_Type = PROTECTED
    ! Buffers = <default>

    Row_Count = 50 ❸
    ! Skip = <none>
    NoLog_Commits
    NoCorresponding
    Defer_Index_Updates
    Constraints
    Parallel
    NoPlace
    Statistics = INTERVAL = 30 ❹
    NoTrigger_Relations
    Record_Definition_File = JOB_HIST
        Format = Delimited_Text
            Prefix = """
            Suffix = """
            NoNull
            Separator = ","
            End Of Line Terminator
End Plan Parameters

Executor Parameters: ❺
    Executor Name = EXECUTOR_1
    ! Place_Only = <none>
    Exception_File = DISK1:[DATABASE]JOB_HIST.EXC_1; ❻
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters
```

```
Executor Parameters:
    Executor Name = EXECUTOR_2
    ! Place_Only = <none>
    Exception_File = DISK1:[DATABASE]JOB_HIST.EXC_2;
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters

Executor Parameters:
    Executor Name = EXECUTOR_3
    ! Place_Only = <none>
    Exception_File = DISK1:[DATABASE]JOB_HIST.EXC_3;
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters
```

Example 23

The following example demonstrates the structure of the record definition file (.rrd) for an RMU Load command for several different data types. The first part of the example displays the table definition, the second part shows the RMU Unload command you could use to get an appropriate .rrd file for these data types, and the last part shows the .rrd file definitions for these data types:

```
SQL> attach 'filename data_types.rdb';
SQL> show table many_types;
Information for table MANY_TYPES

Columns for table MANY_TYPES:
Column Name                     Data Type        Domain
-----------                     ---------        ------
F_ID                            TINYINT
F_CHAR_3                        CHAR(3)
F_TINYINT                       TINYINT
F_SMALLINT                      SMALLINT
F_INTEGER                       INTEGER
F_BIGINT                        BIGINT
F_NTINYINT                      TINYINT(1)
F_NSMALLINT                     SMALLINT(2)
F_NINTEGER                      INTEGER(7)
F_NBIGINT                       BIGINT(5)
F_REAL                          REAL
F_DOUBLE_PREC                   DOUBLE PRECISION
F_DATE_VMS                      DATE VMS
F_DATE_ANSI                     DATE ANSI
F_VARCHAR                       VARCHAR(20)
F_FLOAT                         REAL
F_DATE                          DATE VMS
F_TIME                          TIME
F_TIMESTAMP                     TIMESTAMP(2)
F_INTERVAL                      INTERVAL
                                 DAY (2)
```

```
$ RMU/UNLOAD DATA_TYPES.RDB/RECORD_DEF=(FILE=MANY_TYPES.RRD) -
_$ MANY_TYPES MANY_TYPES.UNL

$ TYPE MANY_TYPES.RRD
DEFINE FIELD F_ID DATATYPE IS SIGNED BYTE.
DEFINE FIELD F_CHAR_3 DATATYPE IS TEXT SIZE IS 3.
DEFINE FIELD F_TINYINT DATATYPE IS SIGNED BYTE.
DEFINE FIELD F_SMALLINT DATATYPE IS SIGNED WORD.
DEFINE FIELD F_INTEGER DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD F_BIGINT DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD F_NTINYINT DATATYPE IS SIGNED BYTE SCALE -1.
DEFINE FIELD F_NSMALLINT DATATYPE IS SIGNED WORD SCALE -2.
DEFINE FIELD F_NINTEGER DATATYPE IS SIGNED LONGWORD SCALE -7.
DEFINE FIELD F_NBIGINT DATATYPE IS SIGNED QUADWORD SCALE -5.
DEFINE FIELD F_REAL DATATYPE IS F_FLOATING.
DEFINE FIELD F_DOUBLE_PREC DATATYPE IS G_FLOATING.
DEFINE FIELD F_DATE_VMS DATATYPE IS DATE.
DEFINE FIELD F_DATE_ANSI DATATYPE IS DATE ANSI.
DEFINE FIELD F_VARCHAR DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD F_FLOAT DATATYPE IS F_FLOATING.
DEFINE FIELD F_DATE DATATYPE IS DATE.
DEFINE FIELD F_TIME DATATYPE IS TIME.
DEFINE FIELD F_TIMESTAMP DATATYPE IS TIMESTAMP SCALE -2.
DEFINE FIELD F_INTERVAL DATATYPE IS INTERVAL DAY SIZE IS 2 DIGITS.
DEFINE RECORD MANY_TYPES.
   F_ID .
   F_CHAR_1 .
 . . .
END MANY_TYPES RECORD.
```

Example 24

The following example shows part of a script for loading a copy of the PERSONNEL database using the output from SQL EXPORT.

1.31 RMU Load Command

```
$! Export the database definition and the data
$ sql$ export database filename personnel into pers.rbr;
$
$! Create an empty database (use RMU Load to add data)
$ sql$ import database from pers.rbr filename copy_pers no data;
$
$! Now use load to add the same table
$ rmu/load copy_pers /match_name=employees employees pers.rbr
%RMU-I-DATRECREAD,  100 data records read from input file.
%RMU-I-DATRECSTO,   100 data records stored.
$
$ rmu/load copy_pers /match_name job_history pers.rbr
%RMU-I-DATRECREAD,  274 data records read from input file.
%RMU-I-DATRECSTO,   274 data records stored.
$
$ rmu/load copy_pers /match_name salary_history pers.rbr
%RMU-I-DATRECREAD,  729 data records read from input file.
%RMU-I-DATRECSTO,   729 data records stored.
$
   .
   .
   .
$ rmu/load copy_pers /match_name work_status pers.rbr
%RMU-I-DATRECREAD,  3 data records read from input file.
%RMU-I-DATRECSTO,   3 data records stored.
```

Example 25

The following example shows that, by default, truncation errors during a Load are reported.

```
$ rmu/load abc f2 f1
%RMU-I-LOADERR, Error loading row 1.
%RDB-E-TRUN_STORE, string truncated during assignment to a column
%RMU-I-DATRECREAD,  1 data records read from input file.
%RMU-I-DATRECSTO,   0 data records stored.
%RMU-F-FTL_LOAD, Fatal error for LOAD operation at 13-FEB-2008 15:39:44.40
$
```

Example 26

The following example shows the use of the /VIRTUAL_FIELDS qualifier. The values of the INTEGER field A and the AUTOMATIC field B are first unloaded into the AA.UNL file from the RMU_LOAD_AUTOMATIC_4_DB database table AA using the /VIRTUAL_FIELDS qualifier. Then the values of the INTEGER field A and the AUTOMATIC field B in the AA.UNL file are loaded into the AA table in the RMU_LOAD_AUTOMATIC_4_DB2 database.

```
$ SQL
create database
    filename RMU_LOAD_AUTOMATIC_4_DB;

-- create a sequence and a table
create sequence S increment by -1;
create table AA
    (a integer
    ,b automatic as s.nextval);

-- load 10 rows
begin
declare :i integer;
for :i in 1 to 10
do
    insert into AA (a) values (:i);
end for;
end;
commit;
disconnect all;
$ exit
$       rmu/unload-
            /virtual=(automatic)-
            /record=(file=rr,format=delim)-
            RMU_LOAD_AUTOMATIC_4_DB aa aa.unl
%RMU-I-DATRECUNL,   10 data records unloaded.
$
$
$! Load using /VIRTUAL
$       rmu/load-
            /record=(file=rr,format=delim)-
            /virtual-
            RMU_LOAD_AUTOMATIC_4_DB2 aa aa.unl
%RMU-I-DATRECREAD,  10 data records read from input file.
%RMU-I-DATRECSTO,   10 data records stored.
$
```

Example 27

This example shows loading a sample of values from the EMPLOYEES table using the LIMIT_TO qualifier.

1.31 RMU Load Command

```
$       RMU/LOAD -
            /LIMIT_TO=80 -
            /RECORD=(FILE:EMP_TXT,FORMAT:DELIMIT) -
            PERSONNEL_SAMPLE -
            EMPLOYEES -
            EMP.TXT
  DEFINE FIELD EMPLOYEE_ID DATATYPE IS TEXT SIZE IS 5.
  DEFINE FIELD LAST_NAME DATATYPE IS TEXT SIZE IS 14.
  DEFINE FIELD FIRST_NAME DATATYPE IS TEXT SIZE IS 10.
  DEFINE FIELD MIDDLE_INITIAL DATATYPE IS TEXT SIZE IS 1.
  DEFINE FIELD ADDRESS_DATA_1 DATATYPE IS TEXT SIZE IS 25.
  DEFINE FIELD ADDRESS_DATA_2 DATATYPE IS TEXT SIZE IS 25.
  DEFINE FIELD CITY DATATYPE IS TEXT SIZE IS 20.
  DEFINE FIELD STATE DATATYPE IS TEXT SIZE IS 2.
  DEFINE FIELD POSTAL_CODE DATATYPE IS TEXT SIZE IS 5.
  DEFINE FIELD SEX DATATYPE IS TEXT SIZE IS 1.
  DEFINE FIELD BIRTHDAY DATATYPE IS TEXT SIZE IS 16.
  DEFINE FIELD STATUS_CODE DATATYPE IS TEXT SIZE IS 1.
  DEFINE RECORD EMPLOYEES.
     EMPLOYEE_ID .
     LAST_NAME .
     FIRST_NAME .
     MIDDLE_INITIAL .
     ADDRESS_DATA_1 .
     ADDRESS_DATA_2 .
     CITY .
     STATE .
     POSTAL_CODE .
     SEX .
     BIRTHDAY .
     STATUS_CODE .
  END EMPLOYEES RECORD.
%RMU-I-DATRECREAD,  80 data records read from input file.
%RMU-I-DATRECSTO,   80 data records stored 14-OCT-2013 07:14:03.52.
$
```

Example 28

This example shows the use of the /Replace_Rows qualifier. If a PRIMARY KEY exists for the table, then the Replace_Rows qualifier will instruct Oracle Rdb to delete the row matching the primary key columns prior to replacing the whole row with new values from the source data file.

```
$ rmu/unload sql$database employees employees
%RMU-I-DATRECUNL,   100 data records unloaded 18-AUG-2015 22:32:10.48.
$ rmu/load/replace_rows sql$database employees employees
%RMU-I-DATRECREAD,  100 data records read from input file.
%RMU-I-DATRECSTO,   100 data records stored 18-AUG-2015 22:32:17.36.
```

Example 29

The following example compares the empty string syntax with the negated keyword usage.

```
$ rmu/unload-
    /record=(nofile,format=delimited,prefix="",suffix="",null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.43.
$ rmu/unload-
    /record=(nofile,format=delimited,noprefix,nosuffix,null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.56.
$
```

1.32 RMU Load Plan Command

Executes a load plan file previously created with the RMU Load command (or created manually by the user).

Format

RMU/Load/Plan plan-file

Command Qualifiers	Defaults
/Audit	None
/Before=date-time	None
/[No]Execute	Execute
/List_Plan=output-file	None
/Since=date-time	None

Description

A load plan file is created when you execute an RMU Load command with the List_Plan qualifier. See Section 1.31 for details on creating a plan file, the format of a plan file, and understanding the informational messages returned by a Parallel Load operation.

Command Parameters

plan-file-spec
The file specification for the load plan file. The default file extension is .plan.

Command Qualifiers

Audit
Allows records to be extracted from the OpenVMS audit journal and saved in the target auditing table. This qualifier can be used in conjunction with the /Before and /Since qualifiers to specify a date/time range so the audit records can be filtered by timestamp.

The /Before and /Since qualifiers accept the standard OpenVMS date/time specification that includes special keywords such as YESTERDAY, TODAY and TOMORROW. These values can be very effective when used with the List_Plan qualifier and used later when using RMU Load Plan.

If these qualifiers are omitted, then their values default to minimum and maximum possible date/time values.

Here is an example of using /Audit with this command.

```
$  RMU/LOAD/AUDIT -
    /SINCE=YESTERDAY -
    /NOEXECUTE -
    /LIST_PLAN=SAMPLE.PLAN -
   TESTDB AUDIT_RECORDS -
   SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
$
```

Later the Plan can be executed and inherit the current value for YESTERDAY.

```
$  RMU/LOAD/PLAN SAMPLE.PLAN
%RMU-I-PROCPLNFIL, Processing plan file SAMPLE.PLAN.
  ! Plan created on 28-NOV-2013 by RMU/LOAD.

  Plan Name = LOAD_PLAN
  Plan Type = LOAD


  Plan Parameters:
      Database Root File = DISK1:[TESTING]TESTDB.RDB;
      Table Name = AUDIT_RECORDS
      Input File = SYS$COMMON:[SYSMGR]SECURITY.AUDIT$JOURNAL
      Audit = TESTDB
          Since = "YESTERDAY"

      ! Fields = <all>
      NoVirtual_Fields
      NoMatch_Name
      Dialect = SQL99
      Transaction_Type = PROTECTED
      ! Buffers = <default>
      ! Commit_Every = <never>
      Row_Count = 500
      ! Skip = <none>
      ! Limit_To = <none>
      NoReplace_Rows
      NoLog_Commits
      NoCorresponding
      NoDefer_Index_Updates
      Constraints
      NoParallel
      NoRestricted_Access
      NoPlace
      ! Statistics = <none>
      ! Trigger_Relations = <not specified>
  End Plan Parameters
```

1.32 RMU Load Plan Command

```
  Executor Parameters:
      Executor Name = EXECUTOR_1
      ! Place_Only = <none>
      ! Exception_File = <none>
      ! RUJ Directory = <default>
      Communication Buffers = 1
  End Executor Parameters
%RMU-I-DATRECREAD,  195 data records read from input file.
%RMU-I-DATRECSTO,   21 data records stored 28-NOV-2013 23:34:50.27.
$
```

Before=date-time

This qualifier, along with the /Since qualifier, is used in conjunction with the /Audit qualifier to allow audit records to be filtered by timestamp. The qualifiers BEFORE and SINCE can specify the date/time range which will be extracted from the OpenVMS audit journal and saved in the target auditing table.

These qualifiers accept the standard OpenVMS date/time specification that includes special keywords such as YESTERDAY, TODAY and TOMORROW. These values can be very effective when used with the List_Plan qualifier and used later when using RMU Load Plan.

If these qualifiers are omitted, then their values default to minimum and maximum possible date/time values.

Execute
Noexecute

The Execute qualifier specifies that the plan file is to be executed. The Noexecute qualifier specifies that the plan file should not be executed, but rather that a validity check be performed on the contents of the plan file.

The validity check determines such things as whether the specified table is in the specified database, the .rrd file (if specified) matches the table, and so on. The validity check does not determine such things as whether your process and global page quotas are sufficient.

By default, data is loaded when the RMU Load Plan command is issued.

List_Plan=output-file

Specifies that Oracle RMU should generate a new plan file and write it to the specified output file. This new plan file is identical to the plan file you specified on the command line (the "original" plan file) with the following exceptions:

- Any comments that appear in the original plan file will not appear in the new plan file.

- If the number of executors specified in the original plan file exceeds the number of storage areas that the table being loaded contains, the new plan file will reduce the number of executors to match the number of storage areas.

Since=date-time
This qualifier, along with the /Before qualifier, is used in conjunction with the /Audit qualifier to allow audit records to be filtered by timestamp. The qualifiers BEFORE and SINCE can specify the date/time range which will be extracted from the OpenVMS audit journal and saved in the target auditing table.

These qualifiers accept the standard OpenVMS date/time specification that includes special keywords such as YESTERDAY, TODAY and TOMORROW. These values can be very effective when used with the List_Plan qualifier and used later when using RMU Load Plan.

If these qualifiers are omitted, then their values default to minimum and maximum possible date/time values.

Usage Notes

- To use the RMU Load Plan command for a database, you must have the RMU$LOAD privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege. Privileges for accessing the database tables involved are also required.
- When the load plan is executed, executors are created as detached processes if you have the OpenVMS DETACH privilege. If you do not have the OpenVMS DETACH privilege, executors are created as subprocesses of your process.

Examples

Example 1

The following example demonstrates the following:

1. The first Oracle RMU command creates a parallel load plan file. The RMU Load command is not executed because the point of issuing the command is to create the plan file, not to load data. Notice that the created load plan has only three executors, even though four were specified on the command line. This is because EMPLOYEES has only three storage areas.

2. The load plan file generated by the first Oracle RMU command is displayed.
3. The load plan file is edited to change some parameters and to rename the executors with names that describe the storage area each executor is responsible for loading.
4. The edited version of the load plan file is executed.

```
$ RMU/LOAD/PARALLEL=(EXECUTOR_COUNT=4, BUFFER_COUNT=4)/NOEXECUTE -
_$ /RECORD_DEFINITION=(FILE=EMPLOYEES.RRD, FORMAT=DELIMITED) -
_$ /LIST_PLAN=EMPLOYEES.PLAN MF_PERSONNEL.RDB EMPLOYEES EMPLOYEES.UNL
%RMU-W-TOOMANYEXECS, 4 executors were requested, but only 3 executors
 will be used.
$ !
$ TYPE EMPLOYEES.PLAN
! Plan created on 20-JUL-1995 by RMU/LOAD.

Plan Name = LOAD_PLAN
Plan Type = LOAD

Plan Parameters:
    Database Root File = MF_PERSONNEL.RDB
    Table Name = EMPLOYEES
    Input File = EMPLOYEES.UNL

    ! Fields = <all>
    Transaction_Type = PROTECTED
    ! Buffers = <default>

    Row_Count = 50
    ! Skip = <none>
    NoLog_Commits
    NoCorresponding
    NoDefer_Index_Updates
    Constraints
    Parallel
    NoPlace
    ! Statistics = <none>
    NoTrigger_Relations
    Record_Definition_File = EMPLOYEES.RRD
        Format = Delimited_Text
            Prefix = """
            Suffix = """
            NoNull
            Separator = ","
            End Of Line Terminator
End Plan Parameters
```

```
Executor Parameters:
    Executor Name = EXECUTOR_1
    ! Place_Only = <none>
    ! Exception_File = <none>
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters

Executor Parameters:
    Executor Name = EXECUTOR_2
    ! Place_Only = <none>
    ! Exception_File = <none>
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters

Executor Parameters:
    Executor Name = EXECUTOR_3
    ! Place_Only = <none>
    ! Exception_File = <none>
    ! RUJ Directory = <default>
    Communication Buffers = 4
End Executor Parameters
```

The following is an edited version of the plan file presented in the previous example. The file has been edited as follows:

- Comments have been added to indicate that the file has been edited.
- The Row_Count value has been changed from 50 to 60.
- Each executor name has been changed to reflect the storage area the executor is responsible for loading.

  This makes it easier to determine the storage area from which a record was rejected if an error occurs during loading. In addition, it makes it easier to determine, when records are rejected, which executor was attempting to load it and which Rdb error corresponds to a particular executor.
- The directory and file name for each exception file has been changed and the comment character preceding "Exception_File" has been removed.
- Directories for the .ruj files have been added and the comment character preceding "RUJ Directory" has been removed.

```
! Plan created on 20-JUL-1995 by RMU/LOAD.
! Edited on 21-JUL-1995 by John Stuart

Plan Name = LOAD_PLAN
Plan Type = LOAD
```

1.32 RMU Load Plan Command

```
Plan Parameters:
    Database Root File = MF_PERSONNEL.RDB
    Table Name = EMPLOYEES
    Input File = EMPLOYEES.UNL

    ! Fields = <all>
    Transaction_Type = PROTECTED
    ! Buffers = <default>
    Row_Count = 20
    ! Skip = <none>
    NoLog_Commits
    NoCorresponding
    NoDefer_Index_Updates
    Constraints
    Parallel
    NoPlace
    ! Statistics = <none>
    NoTrigger_Relations
    Record_Definition_File = EMPLOYEES.RRD
        Format = Delimited_Text
            Prefix = """
            Suffix = """
            NoNull
            Separator = ","
            End Of Line Terminator
End Plan Parameters

Executor Parameters:
    Executor Name = EMPIDS_LOW_EXEC
    ! Place_Only = <none>
    Exception_File = DISK1:[EXCEPTIONS]EMPIDS_LOW.EXC
    RUJ Directory = DISK1:[RUJ]EMPIDS_LOW.RUJ
    Communication Buffers = 4
End Executor Parameters

Executor Parameters:
    Executor Name = EMPIDS_MID_EXEC
    ! Place_Only = <none>
    Exception_File = DISK2:[EXCEPTIONS]EMPIDS_MID.EXC
    RUJ Directory = DISK2:[RUJ]EMPIDS_MID.RUJ
    Communication Buffers = 4
End Executor Parameters

Executor Parameters:
    Executor Name = EMPIDS_OVER_EXEC
    ! Place_Only = <none>
    Exception_File = DISK3:[EXCEPTIONS]EMPIDS_LOW.EXC
    RUJ Directory = DISK3:[RUJ]EMPIDS_LOW.RUJ
    Communication Buffers = 4
End Executor Parameters
```

```
$ !
$ ! Execute the plan file.
$ ! Each executor is assigned the storage area or areas and
$ ! the pid (process ID) for each executor is displayed.
$ ! Notice that Oracle RMU notifies you if an error occurs when
$ ! an executor attempts to load a row, and then lists the Rdb error
$ ! message.  Sometimes you receive two or more Oracle RMU
$ ! messages in a row and then the associated Oracle Rdb message. You
$ ! can match the Oracle RMU message to the Oracle Rdb message by
$ ! matching the executor name prefixes to the messages.
$ !
$ RMU/LOAD/PLAN EMPLOYEES.PLAN
%RMU-I-EXECUTORMAP, Executor EMPIDS_LOW_EXEC (pid: 3140A4CC) will
 load storage area EMPIDS_LOW.
%RMU-I-EXECUTORMAP, Executor EMPIDS_MID_EXEC (pid: 314086CD) will
 load storage area EMPIDS_MID.
%RMU-I-EXECUTORMAP, Executor EMPIDS_OVER_EXEC (pid: 314098CE) will
 load storage area EMPIDS_OVER.
EMPIDS_MID_EXEC:  %RMU-I-LOADERR, Error loading row 4.
EMPIDS_LOW_EXEC:  %RMU-I-LOADERR, Error loading row 1.
EMPIDS_MID_EXEC:  %RDB-E-NO_DUP, index field value already exists;
 duplicates not allowed for EMPLOYEES_HASH
EMPIDS_LOW_EXEC:  %RDB-E-NO_DUP, index field value already exists;
 duplicates not allowed for EMPLOYEES_HASH
%RMU-I-EXECSTAT0, Statistics for EMPIDS_LOW_EXEC:
%RMU-I-EXECSTAT1,   Elapsed time:  00:00:51.69    CPU time:      4.51
%RMU-I-EXECSTAT2,   Storing time:  00:00:32.33    Rows stored:   161
%RMU-I-EXECSTAT3,   Commit time:   00:00:00.66    Direct I/O:    932
%RMU-I-EXECSTAT4,   Idle time:     00:01:44.99    Early commits: 1
%RMU-I-EXECSTAT0, Statistics for EMPIDS_MID_EXEC:
%RMU-I-EXECSTAT1,   Elapsed time:  00:01:06.47    CPU time:      4.32
%RMU-I-EXECSTAT2,   Storing time:  00:00:38.80    Rows stored:   142
%RMU-I-EXECSTAT3,   Commit time:   00:00:01.04    Direct I/O:    953
%RMU-I-EXECSTAT4,   Idle time:     00:00:18.18    Early commits: 2
%RMU-I-EXECSTAT0, Statistics for EMPIDS_OVER_EXEC:
%RMU-I-EXECSTAT1,   Elapsed time:  00:01:04.98    CPU time:      3.22
%RMU-I-EXECSTAT2,   Storing time:  00:00:30.89    Rows stored:   100
%RMU-I-EXECSTAT3,   Commit time:   00:00:00.90    Direct I/O:    510
%RMU-I-EXECSTAT4,   Idle time:     00:00:26.65    Early commits: 1
%RMU-I-EXECSTAT5,   Main process idle time:  00:00:58.11
%RMU-I-DATRECREAD,  495 data records read from input file.
%RMU-I-DATRECSTO,   403 data records stored.
%RMU-I-DATRECREJ,   92 data records rejected.
```

1.33 RMU Monitor Reopen_Log Command

Closes the current Oracle Rdb monitor log file and opens another one without stopping the monitor.

Format

RMU/Monitor Reopen_Log

Description

The RMU Monitor Reopen_Log command closes the current Oracle Rdb monitor log file and opens another log file without stopping the monitor. The new log has the same name as, but a new version number of, the monitor log file you opened with the RMU Monitor Start command. Use the RMU Show Users command to determine the current name and location of the monitor log file before issuing the RMU Monitor Reopen_Log command.

You should use the RMU Monitor Reopen_Log command if the monitor log file gets too large. For example, if you are running out of space on your disk or if database performance slows, you might want to open another log file.

If the disk that contains the Oracle Rdb monitor log file becomes full, you must acquire space on the disk. Once there is sufficient space on this disk, use the RMU Monitor Reopen_Log command and consider backing up (using the DCL COPY command or the OpenVMS Backup utility) the old monitor log file.

When the disk that contains the monitor log becomes full, Oracle Rdb stops writing to the log file, but the Oracle Rdb system does not stop operating. A message is sent to the cluster system operator when this occurs.

Usage Notes

- To use the RMU Monitor Reopen_Log command, either you must have the OpenVMS SETPRV privilege or the OpenVMS WORLD, CMKRNL, DETACH, PSWAPM, ALTPRI, SYSGBL, SYSNAM, SYSPRV, and BYPASS privileges.

Examples

Example 1

The following example closes the existing monitor log file and creates a new one without stopping the Oracle Rdb monitor:

```
$ RMU/MONITOR REOPEN_LOG
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples that show the RMU Monitor commands.

1.34 RMU Monitor Start Command

Activates the Oracle Rdb monitor process.

Format

RMU/Monitor Start

Command Qualifiers	Defaults
/Output = file-name	/Output=SYS$SYSTEM:RDMMON.LOG
/Priority = integer	/Priority = 15
/[No]Swap	/Noswap

Description

The RMU Monitor Start command activates the Oracle Rdb monitor process (RDMS_MONITOR*nn*, where *nn* represents the version Oracle Rdb), sets the priority of this process, and specifies a device, directory and file name in which to create the monitor log file. If the monitor process is active already, you receive the following error message:

```
%RMU-F-MONMBXOPN, monitor is already running
```

An Oracle Rdb monitor process must be running on a node for users logged in to that node to use any Oracle Rdb database. In a VMScluster environment, a monitor process must be running on each node in the cluster from which databases are accessed.

The Oracle Rdb monitor process controls all database access and initiates the automatic database recovery procedure following a system failure or other abnormal termination of a database user process.

See the *Oracle Rdb Installation and Configuration Guide* for information on support for multiple versions of Oracle Rdb.

Command Qualifiers

Output=file-name

Specifies the device, directory, and file name that receives the monitor log. You can use this qualifier to redirect the placement of your monitor log file. The default device and directory is the SYS$SYSTEM directory. The default log file name is RDMMON.LOG. The RMU Monitor Start command causes a new version of the log file to be created for each database session.

Priority=integer
Specifies the base priority of the monitor process. This priority should always be higher than the highest database user process priority.

By default, the monitor runs at the highest interactive priority possible, 15. You should not normally have to lower the monitor process priority. If you change this to a lower priority, an attach operation can cause a deadlock. Deadlock occurs when multiple processes with higher priority than the monitor attempt to attach at the same time. In this case, the monitor must contend for CPU time with multiple higher-priority processes and is perpetually locked out. As a result, no one can use the database.

Swap
Noswap
Enables or disables swapping of the monitor process. The default is Noswap. The Swap qualifier is not recommended for time-critical applications, because no one can use the database while the monitor process is being swapped.

Usage Notes

- To use the RMU Monitor Start command, you must have either the OpenVMS SETPRV privilege or the OpenVMS WORLD, CMKRNL, DETACH, PSWAPM, ALTPRI, PRMMBX, SYSGBL, SYSNAM, SYSPRV, and BYPASS privileges.
- If the monitor has not been started on the system previously, use the RMONSTART.COM command file (which, by default, is located in the SYS$STARTUP directory) instead of the RMU Monitor Start command.
- Start the monitor from the SYSTEM account, which has the SETPRV privilege. The process starting the monitor attempts to give RDMS_MONITOR all privileges. In particular, the privileges required are ALTPRI, CMKRNL, DETACH, PSWAPM, PRMMBX, SETPRV, SYSGBL, SYSNAM, and WORLD.
- The monitor process inherits some quotas, such as MAXDETACH, and the user name of the user who starts it. This can result in severe restrictions on user access. For example, if the user who starts the monitor has a MAXDETACH quota of two, then the monitor can only start two recovery processes at one time. However, the system defines most of the quotas needed by the monitor.

- If the LNM$PERMANENT_MAILBOX table is not defined in the LNM$SYSTEM_TABLE logical name table, either of the following might occur:
  - The RMU Start Monitor command hangs
  - You receive the error, "monitor is not running", when you know it is.

  By default, the LNM$PERMANENT_MAILBOX table is defined in the LNM$SYSTEM_TABLE logical name table. However, sometimes a user or third-party application redefines the LNM$PERMANENT_MAILBOX table in another logical name table (such as the LNM$GROUP table). To recover from this situation, follow these steps:

  1. Define the LNM$PERMANENT_MAILBOX table in the LNM$SYSTEM_TABLE:

     ```
     $ DEFINE/TABLE=LNM$PROCESS_DIRECTORY LNM$PERMANENT_MAILBOX -
     _$ LNM$SYSTEM
     ```

  2. Start the database monitor:

     ```
     RMU/MONITOR START
     ```

  3. Start the application

  Or, change the application that redefines the LNM$PERMANENT_ MAILBOX table so that LNM$PERMANENT_MAILBOX is defined as a search list that includes the LNM$SYSTEM_TABLE table, as shown in the following example:

  ```
  $ DEFINE/TABLE=LNM$PROCESS_DIRECTORY LNM$PERMANENT_MAILBOX -
  _$ LNM$GROUP, LNM$SYSTEM
  ```

- Use the RMU Show System command to determine the location of the monitor log file if it is not in the default location. The monitor log file may not be in the default location if someone has issued the RMU Monitor Start command and specified a location different from the default with the Output qualifier.

CAUTION

The monitor process should be started only by a user whose account has adequate quotas. Ideally, the monitor process should be started from the SYSTEM account.

- To view the contents of monitor log file online (even when disk-based logging is disabled because of disk space problems), use the Performance Monitor and select the Monitor Log screen from the Per-Process menu. See the *Oracle Rdb7 Guide to Database Performance and Tuning* or the Performance Monitor Help for information about using the Performance Monitor.

Examples

Example 1

The following command activates the Oracle Rdb monitor process:

```
$ RMU/MONITOR START
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples that show the RMU Monitor commands.

1.35 RMU Monitor Stop Command

Stops the Oracle Rdb monitor process.

Format

RMU/Monitor Stop

Command Qualifiers	Defaults
/[No]Abort[={Forcex | Delprc}]	/NOABORT
/[No]Wait	/NOWAIT

Description

The RMU Monitor Stop command stops the Oracle Rdb monitor process (RDMS_MONITOR*nn*, where *nn* represents the version Oracle Rdb) normally, either with a shutdown and rollback of the databases or an immediate abort. You can use the RMU Monitor Stop command to shut down all database activity on your node, optionally aborting user processes by forcing an image exit or deleting their processes.

The RMU Monitor Stop command closes the monitor log file also.

An Oracle Rdb monitor process must be running on a node for users logged in to that node to use any Oracle Rdb database. In a VMScluster environment, a monitor process must be running on each node in the cluster from which databases is accessed.

The Oracle Rdb monitor process controls all database access and initiates the automatic database recovery procedure following a system failure or other abnormal termination of a database user process. The monitor log file automatically tracks all access to the database.

Command Qualifiers

Abort=Delprc
Abort=Forcex
Noabort
The Abort=Forcex qualifier stops the monitor immediately without allowing current Oracle Rdb users to complete active transactions or detach from their databases. However, the user processes are not deleted. Active transactions are rolled back. If a process using a database is waiting for a subprocess to complete, the transaction is not rolled back until the subprocess completes.

Using the Abort qualifier with no option is equivalent to specifying the Abort=Forcex qualifier.

The Abort=Delprc qualifier stops the monitor immediately without allowing current Oracle Rdb users to complete active transactions or detach from their databases. Each user process that was attached to an Oracle Rdb database is deleted immediately.

The Noabort qualifier allows current user processes to continue and complete before stopping. New users on the node are not allowed to attach to any database, but existing database users can complete their sessions normally. Once existing database user processes terminate, the database monitor shuts down.

The Noabort qualifier is the default.

Wait
Nowait
Specifies whether the Oracle RMU operation completes when the monitor acknowledges the stop request (Nowait), or whether RMU waits until the monitor finishes shutting down (Wait).

The default is Nowait.

Usage Notes

- To use the RMU Monitor Stop command, you must have either the OpenVMS SETPRV privilege or the OpenVMS WORLD, CMKRNL, DETACH, PSWAPM, PRMMBX, ALTPRI, SYSGBL, SYSNAM, SYSPRV, and BYPASS privileges.

  Note

  If Oracle Trace is installed on your system, you stall the Oracle Rdb monitor process with the RMU Monitor Stop command unless you do one of the following:

  - Shut down Oracle Trace, then shut down the Oracle Rdb monitor (in that order).
  - Use the RMU Monitor Stop command with the Abort=Delprc qualifier to shut down Oracle Rdb and force the monitor out of the Oracle Trace database.

1.35 RMU Monitor Stop Command

Examples

Example 1

The following command causes the Oracle Rdb monitor process to shut down after existing database users end their access to the database. New users on this node are unable to attach to any Oracle Rdb database.

```
$ RMU/MONITOR STOP
```

Example 2

The following command causes the Oracle Rdb monitor to stop immediately without allowing current Oracle Rdb users to complete active transactions (they are rolled back) or detach (DISCONNECT) from their databases. However, the user processes are not deleted. Because the monitor is shut down, all Oracle Rdb activity on this node is terminated.

```
$ RMU/MONITOR STOP /ABORT=FORCEX
```

Example 3

The following command causes the Oracle Rdb monitor to stop immediately without allowing current Oracle Rdb users to complete active transactions (they are not rolled back) or detach (DISCONNECT) from their databases. Each user process that was attached to a Oracle Rdb database on this node is deleted immediately.

```
$ RMU/MONITOR STOP /ABORT=DELPRC
```

1.36 RMU Move_Area Command

Permits you to move one or more storage areas to different disks. You can also choose to move the database root file to a different disk.

Format

RMU/Move_Area root-file-spec storage-area-list

Command Qualifiers	Defaults
/[No]After_Journal[=file-spec]	See description
/[No]Aij_Options[=journal-opts-file]	See description
/All_Areas	See description
/[No]Area	See description
/Attributes=options	See decsription
/[No]Cdd_Integrate	Nocdd_Integrate
/[No]Checksum_Verification	/Checksum_Verification
/Directory=directory-spec	None
/[No]Log	Current DCL verify value
/Nodes_Max=n	Keep current value
/[No]Online	Noonline
/Option=file-spec	None
/Page_Buffers=n	n=3
/Path=cdd-path	Existing value
/[No]Quiet_Point	/Quiet_Point
/Root=file-spec	None
/Row_Cache_Options=file-spec	None
/Threads=n	/Threads=10

Command Qualifiers (cont)	Defaults
/Users_Max=n	Keep current value

File or Area Qualifiers	Defaults
/Blocks_Per_Page=n	None
/[No]Data_File	See description
/Extension={Disable | Enable }	Current value
/File=file-spec	None
/Read_Only	Current value
/Read_Write	Current value
/[No]Snapshots[=([File=file-spec],[Allocation=n])]	See description
/[No]Spams	Leave attribute unchanged
/Thresholds=(n,n,n)	None

1.36 RMU Move_Area Command

Description

The RMU Move_Area command lets you modify certain area parameters when the move operation is performed. All the files are processed simultaneously during the move operation. The performance of the RMU Move_Area command is similar to that of the RMU Backup command, and it eliminates the need for intermediate storage media.

Note that when a snapshot file is moved, Oracle RMU does not actually move the snapshot file; instead, Oracle RMU re-creates and initializes the snapshot file in the specified location. See the description of the Snapshot qualifier for more information about using this qualifier, including information on its proper usage.

Note

You must perform a full and complete Oracle RMU backup operation immediately after the Oracle RMU move area operation completes to ensure that the database can be properly restored after a database failure or corruption.

Command Parameters

root-file-spec
The name of the database root file for the database whose storage areas you want to move.

storage-area-list
The name of one or more storage areas that you want to move.

Command Qualifiers

After_Journal[=file-spec]
Noafter_Journal

Note

This qualifier is maintained for compatibility with versions of Oracle Rdb prior to Version 6.0. You might find it more useful to specify the Aij_Options qualifier, unless you are only interested in creating extensible after-image journal (.aij) files.

Specifies how Oracle RMU is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the After_Journal qualifier and provide a file specification, Oracle RMU enables after-image journaling and creates a new extensible after-image journal (.aij) file for the database.
- If you specify the After_Journal qualifier but do not provide a file specification, Oracle RMU enables after-image journaling and creates a new extensible .aij file for the database with the same name as, but a different version number from, the .aij file for the database root file being moved.
- If you specify the Noafter_Journal qualifier, Oracle RMU disables after-image journaling and does not create a new .aij file.
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, Oracle RMU retains the original journal setting (enabled or disabled) and the original .aij file state.

You can only specify one, or none, of the following after-image journal qualifiers in a single RMU Move_Area command: After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options.

You cannot use the After_Journal qualifier to create fixed-size .aij files; use the Aij_Options qualifier.

You can facilitate recovery by creating a new .aij file because a single .aij file cannot be applied across a move area operation that changes an area page size. A single .aij file cannot be applied across a move operation because the move operation is never recorded in the .aij file (and therefore the increase in page size is also not journaled). Therefore, when you attempt to recover the database, the original page size is used for recovery purposes. So, if the .aij file contains database insert transactions, these updates might have more free space associated with them than is available on the original page size. This results in an inability to recover the insert transaction, which in turn results in a bugcheck and a corrupted database.

This qualifier is valid only when no users are attached to the database and only when the root file is moved.

Aij_Options[=journal-opts-file]
Noaij_Options

Specifies how Oracle RMU is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the Aij_Options qualifier and provide a journal-opts-file, Oracle RMU enables journaling and creates the .aij file or files you specify

for the database. If only one .aij file exists for the database, it will be an extensible .aij file. If two or more .aij files are created for the database, they will be fixed-size .aij files (as long as at least two .aij files are always available).

- If you specify the Aij_Options qualifier but do not provide a journal-opts-file, Oracle RMU disables journaling and does not create any new .aij files.
- If you specify the Noaij_Options qualifier, Oracle RMU retains the original journal setting (enabled or disabled) and retains the original .aij file.
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, Oracle RMU retains the original journal setting (enabled or disabled) and the original .aij file state.

See Section 1.67.1 for information on the format of a journal-opts-file.

Note that you cannot use the RMU Move_Area command with the Aij_Options qualifier to alter the journal configuration. However, you can use it to define a new after-image journal configuration. When you use it to define a new after-image journal configuration, it does not delete the journals in the original configuration. Those can still be used for recovery. If you need to alter the after-image journal configuration, you should use the RMU Set After_Journal command.

The Aij_Options qualifier is valid only when no users are attached to the database and only when the root file is moved.

All_Areas
Specifies that all database storage areas are to be moved. If you specify the All_Areas qualifier, you do not need to specify a storage-area-list.

By default, only areas specified in the storage-area-list are moved.

Area
Noarea

Note

Due to the confusing semantics of the Area and Noarea qualifiers, the Area and Noarea qualifiers are deprecated. Oracle Corporation recommends that you use one of the following methods to specify areas to be moved:

- To move all the storage areas in the database use the All_Areas qualifier and do not specify a storage-area-list parameter

- To move only selected areas in the database, specify the storage-area-list parameter or use the Options qualifier and specify an options file.
- To move only the database root file for a multifile database, or to move an entire single-file database, specify the root qualifier and do not specify a storage-area-list parameter.

Controls whether specific storage areas are moved. If you specify the Area qualifier, only the storage areas specified in the option file or the storage-area-list are moved. If you specify Noarea, all the storage areas in the database are moved.

The default is the Area qualifier.

Attributes=options
Used to specify a number of qualifiers and use an abbreviation for the qualifier name. Normally, /THRESHOLDS, /READ_ONLY and /READ_WRITE cannot be abbreviated or an error will be returned because the abbreviation is not unique for the first four characters (an OpenVMS DCL requirement). By using the /ATTRIBUTES qualifier, THRESHOLDS can be abbreviated to the first four characters and either the option READ_ONLY or the option READ_WRITE can be abbreviated to the first six characters. Examples of the syntax follow.

```
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_ONLY)
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_WRITE)
/ATTRIBUTES=THRESHOLDS=(50,50,50)
/ATTRIBUTES=READ_ONLY
/ATTRIBUTES=READ_WRITE
```

Examples of using abbreviation with the /ATTRIBUTES qualifier options follow.

```
/ATTR=(THRE=(50,50,50),READ_O)
/ATTR=(THRE=(50,50,50),READ_W)
/ATTR=THRE=(50,50,50)
/ATTR=READ_O
/ATTR=READ_W
```

Cdd_Integrate
Nocdd_Integrate
Integrates the metadata from the root (.rdb) file of the moved database into the data dictionary (assuming the data dictionary is installed on your system).

If you specify the Nocdd_Integrate qualifier, no integration occurs during the move operation.

You can use the Nocdd_Integrate qualifier even if the DICTIONARY IS REQUIRED clause was used when the database being moved was defined.

The Cdd_Integrate qualifier integrates definitions *in one direction only*—from the database file to the dictionary. The Cdd_Integrate qualifier does *not* integrate definitions from the dictionary to the database file.

The Nocdd_Integrate qualifier is the default.

Checksum_Verification
Nochecksum_Verification
Requests that the page checksum be verified for each page moved. The default is to perform this verification.

The Checksum_Verification qualifier uses CPU resources but can provide an extra measure of confidence in the quality of the data being moved.

Use of the Checksum_Verification qualifier offers an additional level of data security when the database employs disk striping or RAID (redundant arrays of inexpensive disks) technology. These technologies fragment data over several disk drives, and use of the Checksum_Verification qualifier permits Oracle RMU to detect the possibility that the data it is reading from these disks has been only partially updated. If you use either of these technologies, you should use the Checksum_Verification qualifier.

Oracle Corporation recommends that you use the Checksum_Verification qualifier with all move operations where integrity of the data is essential.

Directory=directory-spec
Specifies the destination directory for the moved database files. Note that if you specify a file name or file extension, all moved files are given that file name or file extension. There is no default directory specification for this qualifier.

See the Usage Notes for information on how this qualifier interacts with the Root, File, and Snapshot qualifiers and for warnings regarding moving database files into a directory owned by a resource identifier.

If you do not specify this qualifier, Oracle RMU attempts to move all the database files (unless they are qualified with the Root, File, or Snapshot qualifier) to their current location.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Nodes_Max=n
Specifies a new value for the database maximum node count parameter. The default is to leave the value unchanged.

Use the Nodes_Max qualifier only if you move the database root file.

Online
Noonline
Allows the specified storage areas to be moved without taking the database off line. This qualifier can be used only when you specify the storage-area-list parameter, or when you specify the Options=file-spec qualifier. The default is Noonline. You cannot move a database root file when the database is on line. The Root qualifier cannot be specified with the Online qualifier in an RMU Move_Area command.

Option=file-spec
Specifies an options file containing storage area names, followed by the storage area qualifiers that you want applied to that storage area. Do *not* separate the storage area names with commas. Instead, put each storage area name on a separate line in the file. The storage area qualifiers that you can include in the options file are:

Blocks_Per_Page
File
Snapshot
Thresholds

If you specify the Snapshot qualifier, you must also move the corresponding data files at the same time. To move a snapshot file independently of its corresponding data file, use the RMU Repair command with the Initialize=Snapshots=Confirm qualifier.

You can use the DCL line continuation character, a hyphen (-), or the comment character (!) in the options file.

There is no default for this qualifier. Example 3 in the Examples section shows the use of an options file.

If the Option qualifier is specified, the storage-area-list parameter is ignored.

Page_Buffers=n
Specifies the number of buffers to be allocated for each file to be moved. The number of buffers used is twice the number specified; half are used for reading the file and half for writing the moved files. Values specified with the Page_ Buffers qualifier can range from 1 to 5. The default value is 3. Larger values might improve performance, but they increase memory usage.

Path=cdd-path
Specifies a data dictionary path into which the definitions of the moved database will be integrated. If you do not specify the Path qualifier, Oracle RMU uses the CDD$DEFAULT logical name value of the user who enters the RMU Move_Area command.

If you specify a relative path name, Oracle Rdb appends the *relative* path name you enter to the CDD$DEFAULT value. If the cdd-path parameter contains nonalphanumeric characters, you must enclose it within quotation marks ("").

Oracle Rdb ignores the Path qualifier if you use the Nocdd_Integrate qualifier or if the data dictionary is not installed on your system.

Quiet_Point
Noquiet_Point
Allows you to specify that a database move operation is to occur either immediately or when a quiet point for database activity occurs. A **quiet point** is defined as a point where no active update transactions are in progress in the database.

When you specify the Noquiet_Point qualifier, Oracle RMU proceeds with the move operation as soon as the RMU Move_Area command is issued, regardless of any update transaction activity in progress in the database. Because Oracle RMU must acquire exclusive locks on the physical and logical areas for the areas being moved, the move operation fails if there are any active transactions with exclusive locks on storage areas that are being moved. However, once Oracle RMU has successfully acquired all the needed concurrent-read storage area locks, it should not encounter any further lock conflicts. If a transaction is started that causes Oracle Rdb to request exclusive locks on the areas that are in the process of being moved, that transaction either waits or gets a lock conflict error, but the move area operation continues unaffected.

If you intend to use the Noquiet_Point qualifier with a move procedure that previously specified the Quiet_Point qualifier (or did not specify either the Quiet_Point or the Noquiet_Point qualifier), you should examine any applications that execute concurrently with the move operation. You might

need to modify your applications or your move procedure to handle the lock conflicts that can occur when you specify the Noquiet_Point qualifier.

When you specify the Quiet_Point qualifier, the move operation begins when a quiet point is reached.

The default is Quiet_Point.

Row_Cache_Options=file-spec
Specifies a row cache backing store options file which contains per database and/or per cache row cache backing store directory specifications.

The file-spec must be a valid file specification of an existing options file. This qualifier cannot be negated. The following syntax must be used in the row cache backing store options file.

```
$ TYPE FILENAME.OPT
/BACKING_STORE = device:[directory]
row_cache_name /BACKING_STORE = device:[directory]
row_cache_name /NOBACKING_STORE
```

To modify or remove the current per database default backing store location, either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified on the first line without being preceded by a row cache name.

To modify or remove a current per cache backing store location, an existing row cache name currently defined in the database root file must be specified followed by either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified. The wild card characters "%" and/or "*" can be specified as part of the row cache name, where "*" can be used in the place of one or more contiguous characters and "%" can be used in the place of a single character. In this case all matching per cache backing store entries will be modified with the following /BACKING_STORE = or /NOBACKING_STORE specification. The RMU/DUMP/HEADER command can be used to display the current per database default row cache backing store directory as well as the current per cache row cache backing store directories.

The /ROW_CACHE_OPTIONS qualifier cannot be used with the /ONLINE or /QUIET_POINT qualifiers. It can only be used if the /ROOT qualifier is specified.

Threads=number
Specifies the number of reader threads to be used by the move process.

1.36 RMU Move_Area Command

RMU creates so called internal 'threads' of execution to read data from one specific storage area. Threads run quasi-parallel within the process executing the RMU image. Each thread generates its own I/O load and consumes resources like virtual address space and process quotas (e.g. FILLM, BYTLM). The more threads, the more I/Os can be generated at one point in time and the more resources are needed to accomplish the same task.

Performance increases with more threads due to parallel activities which keeps disk drives busier. However, at a certain number of threads, performance suffers because the disk I/O subsystem is saturated and I/O queues build up for the disk drives. Also the extra CPU time for additional thread scheduling overhead reduces the overall performance. Typically 2-5 threads per input disk drive are sufficient to drive the disk I/O susbsystem at its optimum. However, some controllers may be able to handle the I/O load of more threads, for example disk controllers with RAID sets and extra cache memory.

In a move operation, one thread moves the data of one storage area at a time. If there are more storage areas to be moved than there are threads, then the next idle thread takes on the next storage area. Storage areas are moved in order of the area size - largest areas first. This optimizes the overall elapsed time by allowing other threads to move smaller areas while an earlier thread is still working on a large area. If no threads qualifier is specified then 10 threads are created by default. The minimum is 1 thread and the maximum is the number of storage areas to be moved. If the user specifies a value larger than the number of storage areas, then RMU silently limits the number of threads to the number of storage areas.

For a move operation, you can specify a threads number as low as 1. Using a threads number of 1 generates the smallest system load in terms of working set usage and disk I/O load. Disk I/O subsystems most likely can handle higher I/O loads. Using a slightly larger value than 1 typically results in faster execution time.

Users_Max=n
Specifies a new value for the database maximum user count parameter.

The default is to leave the value unchanged.

Use the Users_Max qualifier only if you move the database root file.

File or Area Qualifiers

Blocks_Per_Page=n
Specifies a new page size for the storage area to which it is applied. You cannot decrease the page size of a storage area.

If you attempt to change the page size during an online Move_Area operation, you might receive a PAGESIZETOOBIG "Database must be off-line to assign a page size of <num> to area <name>" error message. Changing the page size sometimes requires that Oracle Rdb change the buffer size for the database also (because buffers must be large enough to hold at least one page from each area). However, the buffer size cannot change if other users are accessing the database.

You might want to increase the page size in storage areas containing hash indexes that are close to full. By increasing the page size in such a situation, you prevent the storage area from extending.

The Blocks_Per_Page qualifier is a positional qualifier.

[No]Data_File
NODATA_FILE says not to move the storage area data file(s). It only moves the snapshot storage area file(s). DATA_FILE is the default. It does not accept any values. DATA_FILE as a local qualifier can override NODATA_FILE as a global qualifier. NODATA_FILE as a local qualifier can override DATA_FILE as a global qualifier.

If NODATA_FILE is specified, the storage area data file is not moved or modified. A new version of the file will not be created. Any existing RMU/MOVE_AREA qualifiers that would require an update/change to the data area file are disallowed if /NODATA_FILE is specified.

Therefore, the following existing /MOVE_AREA qualifiers cannot be specified with /NODATA_FILE.

- /FILE
- /ROOT
- /BLOCKS_PER_PAGE
- /NODES_MAX
- /USERS_MAX
- /SPAMS
- /THRESHOLDS
- /READ_ONLY
- /READ_WRITE
- /EXTENSION

1.36 RMU Move_Area Command

The Data_File qualifier is a positional qualifier that can be specified globally as a default and/or for one or more named storage areas. It can be specified on the command line or in an options file using the existing RMU/MOVE_AREA /OPTION=filespec qualifier.

Extension=Disable
Extension=Enable
Allows you to change the automatic file extension attribute when you move a storage area.

Use the Extension=Disable qualifier to disable automatic file extensions for one or more storage areas.

Use the Extension=Enable qualifier to enable automatic file extensions for one or more storage areas.

If you do not specify the Extension=Disable or the Extension=Enable qualifier, the storage areas is moved with the automatic file extension attributes that are currently in effect.

The Extension qualifier is a positional qualifier.

File=file-spec
Requests that the storage area to which this qualifier is applied be moved to the specified location.

The File qualifier is a positional qualifier. This qualifier is not valid for single-file databases.

See the Usage Notes for information on how this qualifier interacts with the Root, Snapshot, and Directory qualifiers.

Read_Only
Use the Read_Only qualifier to change a read/write storage area or a write-once storage area to a read-only storage area.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas are moved with the read/write attributes that are currently in effect for the database.

This is a positional qualifier.

Read_Write
Use the Read_Write qualifier to change a read-only storage area or a write-once storage area to a read/write storage area.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas are moved with the read/write attributes that are currently in effect for the database.

This is a positional qualifier.

[No]Snapshots[=([File=file-spec],[Allocation=n])]
Allows you to specify a new snapshot file allocation size, a new snapshot file location, or both, for the storage area to which the qualifier is applied. NOSNAPSHOTS says to not move the storage area snapshot file(s). It only moves the data storage area file(s). SNAPSHOTS is the default. [=([FILE=filespec],[ALLOCATION=n])] cannot be specified with NOSNAPSHOTS. SNAPSHOTS as a local qualifier can override NOSNAPSHOTS as a global qualifier. NOSNAPSHOTS as a local qualifier can override SNAPSHOTS as a global qualifier.

Use the Allocation=n option to specify the snapshot file allocation size in *n* pages; use the File=file-spec option to specify a new file location for the snapshot file associated with the area being moved.

Note that when you specify a new file location for the snapshot file, the snapshot file is not actually moved; instead, Oracle RMU creates and initializes a new snapshot file in the specified directory. However, if a snapshot file is accidentally deleted or becomes corrupt, using this qualifier is not the recommended or supported method for re-creating the snapshot file. Use the RMU Repair command instead. See the Section 1.43 for information on using the RMU Repair command to re-create and initialize a deleted or corrupted snapshot file.

If the keyword Allocation is omitted, the *original* allocation is used, not the storage area's current allocation size.

You cannot specify a snapshot file name for a single-file database. When you create a snapshot file, Oracle Rdb does not store the file specification of the snapshot file. Instead, it uses the file specification of the root file (.rdb) to determine the file specification of the snapshot file.

See the Usage Notes for information on placing a snapshot file on a different device or directory when your database is a single-file database and for information on how this qualifier interacts with the Root, File, and Directory qualifiers.

The following existing /MOVE_AREA qualifiers cannot be specified with /NOSNAPSHOTS.

- /ROOT
- /BLOCKS_PER_PAGE

- /NODES_MAX
- /USERS_MAX
- /SNAPSHOTS=(FILE=file-spec)
- /SNAPSHOTS=(ALLOCATION=n)

The Snapshot qualifier is a positional qualifier.

Spams
Nospams
Specifies whether to enable the creation of space area management (SPAM) pages or to disable the creation of SPAM pages (Nospams) for specified storage areas when converting read/write storage areas to write-once storage areas or vice versa. This qualifier is not permitted with a storage area that has a uniform page format.

When SPAM pages are disabled in a read/write storage area, the SPAM pages are initialized, but they are not updated.

The Spams qualifier is a positional qualifier.

Thresholds=(n,n,n)
Specifies new SPAM thresholds for the storage area to which it is applied (for a mixed page format storage area). The thresholds of a storage area with a uniform page format cannot be changed.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for information on setting SPAM thresholds.

The Thresholds qualifier is a positional qualifier.

Usage Notes

- To use the RMU Move_Area command for a database, you must have the RMU$MOVE privilege in the root file access control list (ACL) for the database or have the OpenVMS SYSPRV or BYPASS privilege.
- You cannot disable extensions of snapshot (.snp) files.
- The parameter (file and area) qualifiers for the RMU Move_Area command have positional semantics. See Section 1.2 for more information on parameter qualifiers.

- The RMU Move_Area command provides four qualifiers, Directory, Root, File, and Snapshots, that allow you to specify the target for the moved files. The **target** can be just a directory, just a file name, or a directory and file name.

  If you use all or some of these four qualifiers, apply them as follows:

  – If you want to move the database root, use the Root qualifier to indicate the target for the moved database root file.

  – Use local application of the File qualifier to specify the target for the moved storage area or areas.

  – Use local application of the Snapshots qualifier to specify the target for the moved snapshot file or files.

  – Use the Directory qualifier to specify a default target directory. The default target directory is the directory to which all storage area and snapshot files not qualified with the File or Snapshot qualifier are moved. It is also the default directory for files qualified with the Root, File, or Snapshot qualifier if the target for these qualifiers does not include a directory specification.

  Note the following when using these qualifiers:

  – Global application of the File qualifier when the target specification includes a file name causes Oracle RMU to move all of the specified storage areas to different versions of the same file name. This creates a database that is difficult to manage.

  – Global application of the Snapshot qualifier when the target specification includes a file name causes Oracle RMU to move all of the specified snapshot files to different versions of the same file name. This creates a database that is difficult to manage.

  – Specifying a file name or extension with the Directory qualifier is permitted, but causes Oracle RMU to move all of the specified files (except those specified with the File or Root qualifier) to different versions of the same file name. Again, this creates a database that is difficult to manage.

  See Example 6.

- You must specify the Root qualifier when you use the RMU Move_Area command on a single-file database. If you omit the Root qualifier, you receive an error message. If you want to place the snapshot file for a single-file database on a different device or directory from the root file, Oracle Corporation recommends that you create a multifile database. However, you can work around this restriction by defining a search list

for a concealed logical name. (However, do not use a nonconcealed rooted logical name to define database files; a database created with a non-concealed rooted logical name can be backed up, but may not restore correctly when you attempt to restore the files to a new directory.)

To create a single-file database with a snapshot file on a different device or directory from the root file, define a search list by using a concealed logical name. Specify the location of the root file as the first item in the search list. When you create the database, use the logical name for the directory specification. Then, copy the snapshot file to the second device. The following example demonstrates the workaround:

```
$ ! Define a concealed logical name.
$ DEFINE /TRANS=CONCEALED/SYSTEM TESTDB USER$DISK1:[DATABASE], -
_$ USER$DISK2:[SNAPSHOT]
$
$ SQL
SQL> ! Create the database.
SQL> !
SQL> CREATE DATABASE FILENAME  TESTDB:TEST;
SQL> EXIT
$ !
$ ! Copy the snapshot file to the second disk.
$ COPY USER$DISK1:[DATABASE]TEST.SNP USER$DISK2:[SNAPSHOT]TEST.SNP
$ !
$ ! Delete the snapshot file from the original disk.
$ DELETE USER$DISK1:[DATABASE]TEST.SNP;
```

- There are no restrictions on the use of the Nospams qualifier option with mixed page format storage areas, but the use of the Nospams qualifier typically causes severe performance degradation. The Nospams qualifier is useful only where updates are rare and batched, and access is primarily by database key (dbkey).

Examples

Example 1

If a storage area is on a disk that is logging error messages, you can move the storage area to another disk by using the RMU Move_Area command. The following command moves the DEPARTMENTS storage area (departments.rda) and the DEPARTMENTS snapshot file (departments.snp) of the mf_personnel database to the DDV21:[RICK.SQL] directory:

```
$ RMU/MOVE_AREA MF_PERSONNEL DEPARTMENTS /DIRECTORY=DDV21:[RICK.SQL]
```

Example 2

The following command moves the EMPIDS_LOW, EMPIDS_MID, and EMPIDS_OVER storage areas for the mf_personnel database to the DISK2:[USER2] directory. The Extension=Disable qualifier disables automatic file extensions for the EMPIDS_LOW, EMPIDS_MID, and EMPIDS_OVER storage area (.rda) files when they are moved to the DISK2:[USER2] directory:

```
$ RMU/MOVE_AREA/EXTENSION=DISABLE/DIRECTORY=DISK2:[USER2]  -
_$ mf_personnel EMPIDS_LOW,EMPIDS_MID,EMPIDS_OVER
```

Example 3

The following RMU Move_Area command uses an options file to specify that the storage area files and snapshot files be moved to different disks. Note that storage area snapshot (.snp) files are located on different disks from one another and from their associated storage area (.rda) files; this is recommended for optimal performance. (This example assumes that the disks specified for each storage area file in options_file.opt are different from those where the storage area files currently reside.)

```
$ RMU/MOVE_AREA/OPTIONS=OPTIONS_FILE.OPT MF_PERSONNEL
```

The following command displays the contents of the options file:

```
$ TYPE options_file.opt

EMPIDS_LOW /FILE=DISK1:[CORPORATE.PERSONNEL]EMPIDS_LOW.RDA -
     /SNAPSHOT=(FILE=DISK2:[CORPORATE.PERSONNEL]EMPIDS_LOW.SNP)

EMPIDS_MID /FILE=DISK3:[CORPORATE.PERSONNEL]EMPIDS_MID.RDA -
     /SNAPSHOT=(FILE=DISK4:[CORPORATE.PERSONNEL]EMPIDS_MID.SNP)

EMPIDS_OVER /FILE=DISK5:[CORPORATE.PERSONNEL]EMPIDS_OVER.RDA -
     /SNAPSHOT=(FILE=DISK6:[CORPORATE.PERSONNEL]EMPIDS_OVER.SNP)

DEPARTMENTS /FILE=DISK7:[CORPORATE.PERSONNEL]DEPARTMENTS.RDA -
     /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]DEPARTMENTS.SNP)

SALARY_HISTORY /FILE=DISK9:[CORPORATE.PERSONNEL]SALARY_HISTORY.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]SALARY_HISTORY.SNP)

JOBS /FILE=DISK7:[CORPORATE.PERSONNEL]JOBS.RDA -
     /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]JOBS.SNP)

EMP_INFO /FILE=DISK9:[CORPORATE.PERSONNEL]EMP_INFO.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]EMP_INFO.SNP)

RESUME_LISTS /FILE=DISK11:[CORPORATE.PERSONNEL]RESUME_LISTS.RDA -
     /SNAPSHOT=(FILE=DISK12:[CORPORATE.PERSONNEL]RESUME_LISTS.SNP)

RESUMES /FILE=DISK9:[CORPORATE.PERSONNEL]RESUMES.RDA -
     /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]RESUMES.SNP)
```

Example 4

1.36 RMU Move_Area Command

The following RMU Move_Area command moves the database root for the mf_personnel database and defines a new after-image journal configuration, using the Aij_Options qualifier:

```
$ RMU/MOVE_AREA/ROOT=DISK1:[DATABASE.PERSONNEL]MF_PERSONNEL -
_$ /AIJ_OPTIONS=aij_config.opt MF_PERSONNEL/NOONLINE
```

The aij_config.opt file contains the following clauses:

```
JOURNAL IS ENABLED -
    RESERVE 2 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 120 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED
ADD AIJ1 -
    FILE DISK2:[MFPERS_AIJ1]AIJ_ONE
ADD AIJ2 -
    FILE DISK3:[MFPERS_AIJ2]AIJ_TWO
```

Example 5

The following example moves all the mf_personnel database storage areas to the DISK3:[db] directory:

```
$ RMU/MOVE_AREA MF_PERSONNEL.RDB /ALL_AREAS/DIR=DISK3:[DB]
```

Example 6

The following example demonstrates the use of the Directory, File, and Root qualifiers. In this example:

- The default directory is specified as DISK2:[DIR].
- The target directory and file name for the database root file is specified with the Root qualifier. The target directory specified with the Root qualifier overrides the default directory specified with the Directory qualifier. Thus, Oracle RMU moves the database root to DISK3:[ROOT] and names it MOVEDRDB.RDB.
- The target directory for the EMPIDS_MID storage area is DISK4:[FILE]. Oracle RMU moves EMPIDS_MID to DISK4:[FILE].
- The target file name for the EMPIDS_LOW storage area is EMPIDS. Thus, Oracle RMU moves the EMPIDS_LOW storage area to the DISK2 default directory (specified with the Directory qualifier), and names the file EMPIDS.RDA.

- The target for the EMPIDS_LOW snapshot file is DISK5:[SNAP]EMPIDS.SNP Thus, Oracle RMU moves the EMPIDS_LOW snapshot file to DISK5:[SNAP]EMPIDS.SNP.
- All the other storage area files and snapshot files in the mf_personnel database are moved to DISK2:[DIR]; the file names for these storage areas remain unchanged.

```
$ RMU/MOVE_AREA DISK1:[DB]MF_PERSONNEL.RDB /ALL-
_$ /DIRECTORY=DISK2:[DIR] -
_$ /ROOT=DISK3:[ROOT]MOVEDRDB.RDB -
_$ EMPIDS_MID/FILE=DISK4:[FILE], -
_$ EMPIDS_LOW/FILE=EMPIDS -
_$ /SNAPSHOT=(FILE=DISK5:[SNAP]EMPIDS.SNP)
```

Example 7

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ROW_CACHE_OPTIONS qualifier to read backing store directory options files to modify or remove the current per database and per cache Row Cache backing store directories in the database root file. The contents of the options file read is displayed when the command is executed and the RMU/DUMP/HEADER command is used to check the modified backing store directories in the database root file.

```
$ SET VERIFY
$ RMU/RESTORE/NOCDD/NOLOG/DIR=TEST$DIRECTORY-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:BSTORE.OPT -
 TEST$DIRECTORY:RSA.RBF
%RMU-I-RESTXT_18, Processing options file BSTORE.OPT
  /BACKING_STORE=DISK:[DIRECTORY]
  SAL_CACHE /BACKING_STORE=DISK:[DIRECTORY]
  JOB_CACHE/BACKING_STORE=DISK:[DIRECTORY]
  DROP_CACHE /BACKING_STORE=DISK:[DIRECTORY]
```

1.36 RMU Move_Area Command

```
$ RMU/DUMP/HEADER/OUT=HDR.LIS TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/MOVE_AREA/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROOT=DISK:[DIRECTORY]FILENAME.EXT
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:WBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file WBSTORE.OPT
  *CACHE /BACKING_STORE=DISK:[DIRECTORY]
$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/COPY_DATABASE/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:NOBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file NOBSTORE.OPT
  /NOBACKING_STORE
  SAL_CACHE /NOBACKING_STORE
  JOB_CACHE/NOBACKING_STORE
  DROP_CACHE /NOBACKING_STORE

$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is database directory
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
```

Example 8

In the following example, only the storage area snapshot files are moved for all database storage areas.

```
$ RMU/MOVE_AREA/ALL/NODATA_FILE/NOLOG/DIR=[.MOVE] MF_PERSONNEL.RDB
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 9

In the following example, only the storage area data files are moved for all database storage areas.

```
$ RMU/MOVE_AREA/ALL/NOSNAPSHOTS/NOLOG/DIR=[.MOVE] MF_PERSONNEL.RDB
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 10

In the following example, only the snapshot storage area file is moved for the EMP_INFO storage area and only the data storage area file is moved for the JOBS storage area. Note that for the JOBS storage area, /DATA_FILE did not need to be specified since it is the default.

```
$ RMU/MOVE_AREA/NOLOG MF_PERSONNEL.RDB -
  EMP_INFO /nodata_file -
           /snapshots=(file=DISK:[DIRECTORY]test_emp_info.snp), -
  JOBS  /data_file -
        /file=DISK:[DIRECTORY]test_jobs -
        /nosnapshots
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 11

In the following example, an options file is used to specify the storage areas to be moved. Only the data storage area file is moved for EMP_INFO; only the snapshot storage area file is moved for JOBS; and both the snapshot and data storage area files are moved for DEPARTMENTS. NOTE that /DATA_FILE and /SNAPSHOT are the defaults.

```
$ RMU/MOVE_AREA/NOLOG/DIR=DISK:[DIRECTORY]/OPTION=TESTMORE.OPT -
 MF_PERSONNEL.RDB
  EMP_INFO -
           /file=DISK:[DIRECTORY]test_emp_info.rda -
           /nosnapshot -
  JOBS     /nodata_file -
           /snapshot = (file=DISK:[DIRECTORY]test_jobs.snp)
  DEPARTMENTS -
              /file=DISK:[DIRECTORY]test_departments -
              /snapshot = (file=DISK:[DIRECTORY]test_departments.snp)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 12

In the following example, the global default qualifiers designate that only the snapshot files should be moved for all storage areas. However, an options file is used to override the default for specific storage areas. Therefore, only the data storage area file is moved for EMP_INFO, only the snapshot storage area file is moved for JOBS, and both the snapshot and data storage area files are moved for DEPARTMENTS. Note that, in this case, /DATA_FILE needed to be specified in the options file to override the global specification of /NODATA_ FILE but /NODATA_FILE did not have to be specified in the options file. Also /NOSNAPSHOT had to be specified in the options file to override the assumed global default of /SNAPSHOT.

1.36 RMU Move_Area Command

```
$ RMU/MOVE_AREA/ALL/DIR=DISK:[DIRECTORY]/NOLOG/NODATA_FILE-
/OPTION=TESTMOVE.OPT  MF_PERSONNEL.RDB
  EMP_INFO /data_file -
           /file=DISK:[DIRECTORY]test_emp_info.rda -
           /nosnapshot
  JOBS     /nodata_file -
           /snapshot = (file=DISK:[DIRECTORY]test_jobs.snp)
  DEPARTMENTS -
              /data_file -
              /file=DISK:[DIRECTORY]test_departments -
              /snapshot = (file=DISK:[DIRECTORY]test_departments.snp)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

Example 13

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ATTRIBUTES qualifier.

```
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTRIBUTES=(THRE=(50,50,50),READ_O)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTRIBUTES=READ_O
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTR=READ_W
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/RESTORE/NOLOG/NOCDD/DIR=DISK:[DIRECTORY] DISK:[DIRECTORY]MFP.RBF
JOBS/ATTR=READ_O
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not available.
```

1.37 RMU Open Command

Opens a database root file and maps its global section to the contents of an OpenVMS virtual address file.

You can use the RMU Open command in conjunction with the SQL ALTER DATABASE statement to control access to the database. See the description of the OPEN IS {AUTOMATIC | MANUAL} clause of the SQL ALTER DATABASE statement in the *Oracle Rdb SQL Reference Manual* for details.

Format

RMU/Open root-file-spec[,...]

Command Qualifiers	Default
/Access=[Un]Restricted	See description
/Global_Buffers[=(Total=i,User_Limit=j)]	See description
/Path	None
/Row_Cache=Disable	See description
/[No]Statistics[=options]	/Nostatistics
/[No]Wait	/Nowait

Description

Once you use the RMU Open command to open a database, the database remains open and mapped until you close it explicitly with an RMU Close command and all users have exited the database with the SQL DISCONNECT or EXIT statements. If you do not issue the RMU Open command, the first user to attach to the database incurs the cost of implicitly opening it and the last user to detach from the database incurs the cost of implicitly closing it.

The effect of the RMU Open command depends on whether you have specified the OPEN IS AUTOMATIC or OPEN IS MANUAL clause to the SQL ALTER DATABASE statement, as follows:

- OPEN IS AUTOMATIC

  If you have specified automatic opening for your database, users can invoke the database at any time without first issuing an RMU Open command. (Although as mentioned above, it is more efficient to explicitly open the database with an RMU Open command and close it with an RMU Close command.)

- OPEN IS MANUAL

  If you have specified manual opening for your database, the RMU Open command must be issued before users can invoke the database.

If you modify the database attribute from OPEN IS AUTOMATIC to OPEN IS MANUAL, the modification takes effect only after all users have detached from the database. (You can issue the RMU/CLOSE/ABORT=FORCEX command to force all users to detach.) Then, you must issue the RMU Open command before users can invoke the database.

If you modify the database attribute from OPEN IS MANUAL to OPEN IS AUTOMATIC, users can invoke the database at their discretion. You do not have to issue the RMU Open command. However, if a user has already opened the database manually when you make this change to the database attribute, the modification takes effect only after you manually close the database by issuing the RMU Close command.

See the *Oracle Rdb Guide to Database Maintenance* for information to help you decide whether to set your database attribute to automatic or manual opening.

When you create a database, you have a choice of how to set up buffers for database pages. You can choose either local or global buffering. Global buffers can provide better system performance. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on setting the number of global buffers for your system.

Command Parameters

root-file-spec[,...]
Specifies the database to open. If the database root file is open, you receive an informational message. The default file extension is .rdb.

Command Qualifiers

Access=Restricted
Access=Unrestricted
Permits the database administrator to open the database and restrict access to it in order to perform maintenance operations or to restructure the database without interference from users who want to gain access. If access is restricted (Access=Restricted), the DBADM privilege is required for SQL access to the database. If the Access=Unrestricted qualifier is specified, users without the DBADM privilege can attach to the database.

Note

Do not confuse the Oracle RMU Access=Restricted qualifier with the SQL restricted access clause (available for use with the following SQL statements: ATTACH, CREATE, DECLARE ALIAS, and IMPORT). When you specify the restricted access clause in SQL, only one user can attach to the database; when you specify the Access=Restricted qualifier using Oracle RMU, any number of users with the DBADM privilege can access the database.

Furthermore, note that an SQL SHOW DATABASE command displays the phrase "No Restricted Access" or the phrase "Restricted Access" if access has been restricted using the SQL restricted access clause. However, SHOW DATABASE tells you nothing about whether Oracle RMU has opened a database with access restricted. Use the RMU Dump command to view the Oracle RMU access setting.

Refer to the *Oracle Rdb SQL Reference Manual* for more information on the SQL restricted access clause.

If you specify the RMU Open command without the Access qualifier, Oracle RMU opens the database in the same access mode as the last RMU Open command performed. If the database was last opened as restricted, issuing the RMU Dump command results in the following message being displayed:

```
Access restricted to privileged users
```

Use this form of the RMU Open command to open the database on other nodes without changing the access mode.

The access mode is clusterwide and the last mode set with the RMU Open command is used for the entire cluster.

For example, if you open the mf_personnel database on node A with the Access=Unrestricted qualifier, and open the same database on node B with the Access=Restricted qualifier, the database has restricted access on both node A and node B. However, the commands do not terminate any user processes that may have gained access while the database was unrestricted.

The access mode is stored in the database. Consequently, if the system fails while access is restricted, access remains restricted unless the unrestricted mode is explicitly requested. The RMU Backup, RMU Restore, and RMU Copy_Database commands also preserve the access mode.

1.37 RMU Open Command

The RMU Close command does not alter the access mode. You can change the mode by using the RMU Open command only. You can use the RMU Open command to restrict access to any database, whether it was opened as AUTOMATIC or MANUAL.

The Access qualifier is a positional qualifier.

Global_Buffers[=(Total=i,User_Limit=j)]
Allows you to set the basic global buffer parameters on each RMU Open command. If you specify the Global_Buffers qualifier, you can optionally specify values for the Total and User_Limit parameters:

- Total is the number of global buffers per node to allocate for this opened instance of the database (minimum = 5, and maximum = 500,000).
- User_Limit is the maximum number of global buffers to be allotted to any given user (minimum = 5, maximum = Total).

The default values for Total and User_Limit are set by:

- The RMU Open command explicitly
- Values determined at the time the database was created

If you do not specify a value for the Total or User_Limit options, the values are determined based on what they were when the database was created.

If a database does not have global buffers enabled, the Global_Buffers qualifier is ignored. Use the RMU Dump command to see if global buffering is enabled or disabled. The RMU Dump command also shows the global buffer count and the maximum global buffer count per user. For example:

```
$ RMU/DUMP MF_PERSONNEL
*-----------------------------------------------------------------
* Oracle Rdb V7.0-00                        22-SEP-1995 10:11:51.14
*
* Dump of Database header
*     Database: DISK1:[DATABASE]MF_PERSONNEL.RDB;1
*
*-----------------------------------------------------------------
```

```
Database Parameters:
    Root filename is "DISK1:[DATABASE]MF_PERSONNEL.RDB;1"
    Created at 7-APR-1994 16:50:09.01
    Oracle Rdb structure level is 70.0
    Maximum user count is 50
    Maximum node count is 16
    Database open mode is AUTOMATIC
    Database close mode is AUTOMATIC
    Database is available for READ WRITE access
    Snapshot mode is NON-DEFERRED
    Statistics are enabled
    Storage Areas...
      - Active storage area count is 10
      - Reserved storage area count is 0
    Buffers...
      - Default user buffer count is 20
      - Default recovery buffer count is 20
      - Global buffers are enabled   <--------
      - Global buffer count is 250   <--------
      - Maximum global buffer count per user is 5   <--------
      - Buffer size is 6 blocks
   .
   .
   .
    Derived Data...
      - Global section size
          With global buffers disabled is 70962 bytes
          With global buffers enabled is 975992 bytes
   .
   .
   .
```

The Global_Buffers qualifier is a positional qualifier.

Path

Specifies the full or relative data dictionary path name in which the definitions reside for the database you want to open.

The Path qualifier is a positional qualifier. The path name cannot include wildcard characters.

Row_Cache=Disable

Disables row caching. This qualifier is provided for use with hot standby databases. Row caching cannot be enabled on a hot standby database while replication is active. If it is enabled, the hot standby feature will not start.

1.37 RMU Open Command

Statistics=Import
Nostatistics
Specifies that statistic information previously saved by using the Statistics=Export qualifier on the RMU Close command is to be loaded when the database is opened. The default is Nostatistics, which indicates that statistic information is not loaded when the database is opened.

After the database is opened using the Statistics=Import qualifier, the saved statistics file is closed. The statistics file is not automatically deleted. It can be deleted if it is no longer needed.

When you use the Statistics=Import qualifier, statistics information is automatically preserved in the event of abnormal database closure. To ensure that the ondisk statistic information files are accurate in the case of a node or monitor failure, the statistic information files are checkpointed by the database monitor every half-hour. The RMU Show Users command identifies when the checkpoint for each database occurs.

The statistic files are not loaded if the physical schema of the database has changed since the statistic file was created. This means that the addition or deletion of storage aras, logical areas, and record caches invalidate the statistic files. This restriction prevents incorrect statistic information from being loaded when intervening physical changes occur to the database. Closing the database updates the statistic files and makes them valid. Use the RMU Show Users command to verify that the statistic information file was imported.

Statistics=[No]Process_Global
Nostatistics
The RMU Open command, with the Statistics qualifier, supports the optional keyword [No]Process_Global, which indicates whether or not you are able to collect per-process statistics. The Statistics=Process_Global qualifier indicates that all processes attached to the database are eligible for per-process monitoring. The default qualifier, Statistics=Noprocess_Global, indicates that attached processes are not automatically eligible; however, you can still activate these processes at run-time using one of the other activation methods described below.

When you use the global activation method, the RMU Show Statistics utility changes the screen header Mode attribute from Online to Global.

Oracle Corporation recommends this method only when there are a small number of active processes. The Statistics=Process_Global qualifier causes each process attached to the database on that node to create a sizable global section into which the process global statistic collection occurs. Oracle Corporation recommends that you activate process global statistic collection on

a per-process basis. That is, you should activate only those processes you are currently interested in, and deactivate them when you are finished with them.

Refer to the RMU Show Statistics Manual for further details on per-process monitoring with RMU Show Statistics.

This example shows the PROCESS_GLOBAL option as well as an indication from RMU/SHOW USER that it has been specified.

```
$ RMU/OPEN/STATISTICS=(IMPORT,PROCESS_GLOBAL) MF_PERSONNEL
$ RMU/SHOW USER MF_PERSONNEL
Oracle Rdb V7.3-13 on node GANDLF  4-MAR-2015 08:23:26.93
    - monitor started  3-MAR-2015 19:14:07.75 (uptime 0 13:09:19)
    - monitor log filename is "$1$DGA231:[LOGS]RDMMON731_GANDLF.LOG;2991"

database $1$DGA170:[DATABASES.V73]MF_PERSONNEL.RDB;1
    - opened  4-MAR-2015 08:22:52.63 (elapsed 0 00:00:34)
    * database is opened by an operator
    * statistic information import failed
    * global per-process statistic collection activated
    * next statistic information checkpoint at  4-MAR-2015 08:52:52.66
```

Wait
Nowait
Specifies whether the system prompt should be returned before the database is completely open and available. Specify the Wait qualifier if you want the system prompt returned when the database is completely open and available. Specify Nowait if you want the system prompt returned immediately, regardless of the state of the open operation.

The Nowait qualifier is the default.

Usage Notes

- To use the RMU Open command for a database, you must have the RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- The command RMU/OPEN/ACCESS=RESTRICTED can be used to prevent non-privileged users attaching to the database while database maintenance is being performed.

  This command sets a permanent attribute in the database root that must be cleared by a subsequent RMU/OPEN/ACCESS=UNRESTRICTED command. This is true even if the database is defined OPEN IS AUTOMATIC. An RMU/CLOSE might be required before the RMU/OPEN can be executed.

The following example shows the reported error when this attribute is active; NOPRIV with an explanation DBNOTOPEN.

```
SQL> attach 'filename sql$database user ''dbuser2'' using ''********'' ';
%SQL-F-ERRATTDEC, Error attaching to database sql$database
-RDB-F-NO_PRIV, privilege denied by database facility
-RDMS-F-DBNOTOPEN, database is not open for access
SQL>
```

The database administrator must be granted RMU$OPEN on the database root file, as well as having DBADM on the database (this might be inherited from OpenVMS override privileges) to execute this command.

The RMU/DUMP/HEADER command will show this setting as seen in the following output:

```
$ rmu/dump/header=ROOT_RECORD PERSONNEL
*-------------------------------------------------------------------------------
* Oracle Rdb V7.3-210                                    22-NOV-2016 14:27:23.59
*
* Dump of Database header
*     Database: USER2:[TESTING]PERSONNEL.RDB;1
*
*-------------------------------------------------------------------------------

Database Parameters:
    Root filename is "USER2:[TESTING]PERSONNEL.RDB;1"

Oracle Rdb specific root record

    Dbkey for Oracle Rdb bootstrap page is 8:462:0
    Access restricted to privileged users
    Current metadata version is 26
    Database has never been backed up
    Database has never been incrementally restored
    Database has never been fully restored
    Database has never been verified
    Database has never been altered
```

Examples

Example 1

The following command opens the mf_personnel database:

```
$ RMU/OPEN MF_PERSONNEL
```

Example 2

The following command opens the mf_personnel database in the WORK directory, all the databases in the .TEST directory, and the databases specified by the path names CDD$TOP.FINANCE and SAMPLE_DB:

```
$ RMU/OPEN DISK1:[WORK]MF_PERSONNEL, CDD$TOP.FINANCE/PATH, -
_$ DISK1:[TEST]*, SAMPLE_DB/PATH
```

Example 3

This command opens the mf_personnel database, sets the total global buffers for this opened instance of the database, and sets the maximum number of global buffers that can be given to any user. This example limits the number of users who can access this database at any given time to 2 (Total divided by User_Limit). You may want to increase the values of Total and User_Limit.

```
$ RMU/OPEN MF_PERSONNEL/GLOBAL_BUFFERS=(TOTAL=10,USER_LIMIT=5)
```

If you define a user limit value that is greater than the value you specify for Total, you receive an error message:

```
$ RMU/OPEN MF_PERSONNEL/GLOBAL=(TOTAL=5,USER_LIMIT=10)
%RMU-F-VALGTRMAX, value (10) is greater than maximum allowed
 value (5) for GLOBAL_BUFFERS.USER_LIMIT
```

Example 4

This command disables row caching.

```
$ RMU/OPEN MF_PERSONNEL.RDB /ROW_CACHE=DISABLE
```

1.38 RMU Optimize After_Journal Command

Optimizes a backed up after-image journal (.aij) file for database recovery (rollforward) operations by eliminating unneeded and duplicate journal records, and by ordering journal records. An optimized .aij (.oaij) file created by the RMU Optimize After_Journal command provides better recovery performance for your database than an .aij file. A benefit of this improved recovery performance is that the database is made available to users sooner.

The RMU Optimize After_Journal command is used to read a backed up .aij file on disk and write the .oaij file to tape or disk.

Format

RMU/Optimize/After_Journal aij-file optimized-aij-file

Command Qualifiers	Defaults
/[No]Accept_Label	/Noaccept_Label
/Active_IO=max-writes	/Active_IO=3
/[No]Assist	See description
/Block_Size=integer	See description
/Crc[=Autodin_II]	See description
/Crc=Checksum	See description
/Nocrc	See description
/Density=density-value[,[No]Compaction]	See description
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/Format={Old_File|New_Tape}	/Format=Old_File
/[No]Group_Size=interval	See description
/Label=(label-name-list)	See description
/Librarian[=options]	None
/[No]Log	Current DCL verify value
/[No]Media_Loader	See description
/Owner=user-id	See description
/Protection[=openvms-file-protection]	See description
/Recovery_Method[=(Sequential|Scatter)]	/Recovery_Method=Sequential
/[No]Rewind	/Norewind
/Tape_Expiration=date-time	The current time
/[No]Trace	/Notrace

1.38 RMU Optimize After_Journal Command

Description

The RMU Optimize After_Journal command performs the following optimizations to backed up .aij files:

- The .aij records from transactions that rolled back are eliminated.

  Because transactions that are rolled back in an .aij file are not needed in a recovery operation, they are not part of an optimized .aij file.

- Duplicate .aij records are eliminated.

  Duplicate .aij records are .aij records that update the same database record. During the rollforward of an .aij file, duplicate .aij records cause a database record to be updated multiple times. Each update supersedes the previous update, meaning only the last update is relevant. Therefore, all but the last update to a database record can be eliminated from an .aij file.

- The .aij records are ordered by physical database key (dbkey).

  Ordering .aij records by physical dbkey improves I/O performance at recovery time.

See the *Oracle Rdb Guide to Database Maintenance* for further description of optimizing .aij files.

The RMU Optimize After_Journal command has the following restrictions:

- You can only optimize quiet-point .aij backup files.
- You cannot optimize a current .aij file.
- You cannot optimize an .oaij file.

  Note

  Because an .oaij file is not functionally equivalent to the original .aij file, the original .aij file should not be discarded after it has been optimized.

- You cannot use .oaij files with the following types of recovery operations:
  - By-area recovery operations (recovery operations that use the RMU Recover command with the Areas qualifier).
  - By-page recovery operations (recovery operations that use the RMU Recover command with the Just_Corrupt qualifier).

- RMU Recover commands with the Until qualifier. The .oaij file does not retain enough of the information from the original .aij file for such an operation.
- Recovery operation where the database or any storage areas (or both) are inconsistent with the .oaij file. A database or storage area will be inconsistent with the .oaij file if the transaction sequence number (TSN) of the last committed transaction of the database or storage area is not equal to the TSN of the last committed transaction in the open record of the .aij file. The last committed TSN in the .oaij file represents the last transaction committed to the database at the time the original .aij file was created.

As a workaround for these restrictions against using .oaij files in these recovery operations, use the original, unoptimized .aij files in these recovery operations instead.

- Any .aij file that possibly contains incomplete transactions cannot be optimized. Incomplete transactions can occur in an .aij file under the following circumstances:
  - The .aij file is backed up with a no-quiet-point backup operation (because transactions can span .aij files)

    Note that transactions in a fixed-size journal configuration may span .aij files. Thus, if each journal in a fixed-size journal configuration has been backed up on a per-journal basis, the resulting files are equivalent to a no-quiet-point .aij backup operation. These .aij backup files cannot be optimized unless you perform a manual quiet-point backup operation first. A quiet-point backup operation forces a switch-over to another available .aij file which ensures that no transaction spans two journal files.
  - The previous .aij file was backed up with a no-quiet-point backup operation
  - The .aij file has unresolved distributed transactions

  There are no workarounds to these restrictions against optimizing .aij files with incomplete transactions.

Command Parameters

aij-file

The name of the .aij file that you want to optimize. It cannot be a current .aij file.

The default file extension is .aij.

optimized-aij-file
The name of the optimized .oaij file to be produced by the RMU Optimize After_Journal command.

The default file extension is .oaij.

Command Qualifiers

Accept_Label
Specifies that Oracle RMU should keep the current tape label it finds on a tape during an optimize-to-tape operation even if that label does not match the default label or that specified with the Label qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your optimize-to-tape operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes.

If you do not specify this qualifier, the default behavior of Oracle RMU is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the Label qualifier).

See the description of the Labels qualifier in this section for information on default labels. See Table 1–5 for a summary of which labels are applied under a variety of circumstances.

Active_IO=max-writes
Specifies the maximum number of write operations to the .oaij file device that the RMU Optimize After_Journal command will attempt simultaneously. This is not the maximum number of write operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 might improve performance with some tape drives.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Block_Size=integer
Specifies the maximum record size for the optimized .oaij file. The size can vary between 2048 and 65,024 bytes. The default value is device dependent. The appropriate block size is a compromise between tape capacity and error rate. The block size you specify must be larger than the largest page length in the database.

Crc[=Autodin_II]
Uses the AUTODIN-II polynomial for the 32-bit cyclic redundancy check (CRC) calculation and provides the most reliable end-to-end error detection. This is the default for NRZ/PE (800/1600 bits/inch) tape drives.

Typing the Crc qualifier is sufficient to select the Crc=Autodin_II option. It is not necessary to type the entire qualifier.

Crc=Checksum
Uses one's complement addition, which is the same computation used to do a checksum of the AIJ data on disk. This is the default for GCR (6250 bits/inch) tape drives and for TA78, TA79, and TA81 drives.

The Crc=Checksum qualifier allows detection of data errors.

Nocrc
Disables end-to-end error detection. This is the default for TA90 (IBM 3480 class) drives.

Note

The overall effect of the Crc=Autodin_II, Crc=Checksum, and Nocrc defaults is to make tape reliability equal to that of a disk. If you retain your tapes longer than 1 year, the Nocrc default might not be adequate. For tapes retained longer than 1 year, use the Crc=Checksum qualifier.

If you retain your tapes longer than 3 years, you should always use the Crc=Autodin_II qualifier.

Tapes retained longer than 5 years could be deteriorating and should be copied to fresh media.

See the *Oracle Rdb Guide to Database Maintenance* for details on using the Crc qualifiers to avoid underrun errors.

Density=density-value[,[No]Compaction]
Specifies the density at which the output volume is to be written. The default value is the format of the first volume (the first tape you mount). You do not need to specify this qualifier unless your tape drives support data compression or more than one recording density.

The Density qualifier is applicable only to tape drives. Oracle RMU returns an error message if this qualifier is used and the target device is not a tape drive.

The following table lists the tape device density and compaction values supported by Oracle RMU.

DEFAULT	800	833	1600
6250	3480	3490E	TK50
TK70	TK85	TK86	TK87
TK88	TK89	QIC	8200
8500	8900	DLT8000	
SDLT	SDLT320	SDLT600	
DDS1	DDS2	DDS3	DDS4
AIT1	AIT2	AIT3	AIT4
LTO2	LTO3	COMPACTION	NOCOMPACTION

For the OpenVMS tape device density values that accept tape compression, you can use the following syntax:

```
/DENSITY = (new_density_value,[No]Compaction)
```

In order to use the Compaction or NoCompaction keyword, you must use one of the following density values that accepts compression:

DEFAULT	3480	3490E	8200
8500	8900	TK87	TK88
TK89	DLT8000	SDLT	SDLT320
AIT1	AIT2	AIT3	AIT4
DDS1	DDS2	DDS3	DDS4
SDLT600	LTO2	LTO3	

Refer to the OpenVMS documentation for more information about density values.

Encrypt=({Value= | Name=}[,Algorithm=])
The Encrypt qualifier decrypts the backup file of the optimized after-image journal file.

1.38 RMU Optimize After_Journal Command

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

This feature only works for a newer format backup file which has been created using the Format=New_Tape qualifier. Therefore you have to specify the Format=New_Tape qualifier with this command if you also use the Encrypt qualifier.

Format=Old_Rms
Format=New_Tape
Synonymous with the Format=Old_File and Format=New_Tape qualifiers. See the description of those qualifiers.

Format=Old_File
Format=New_Tape
The Format qualifier allows you to specify the format of the files written by the RMU Optimize After_Journal command.

If you specify the default, Format=Old_File, the RMU Optimize After_Journal command writes files in RMS format. This format is provided for compatibility with prior versions of Oracle Rdb. If you specify Format=Old_File, you must mount the media by using the DCL MOUNT command before you issue the RMU Optimize After_Journal command. Because the RMU Optimize After_ Journal command will use RMS to write to the tape, the tape must be mounted as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier with the MOUNT command).

If you specify FOREIGN access although your backup file was created using the Format=Old_File qualifier, you will not receive an error message. The tape will be considered unlabeled, and thus the operation will process whatever data is at the current position of the tape (labels, data, or something else). A failure will occur, but what will fail and how it will fail is unpredictable because the type of information that will be read is unknown. The result is an unlabeled tape that can be difficult to use for recovery operations.

If you specify Format=New_Tape, the RMU Optimize After_Journal command writes .aij files in a format similar to that used by an RMU Backup command. If you specify Format=New_Tape, you must mount the media by using the DCL MOUNT command before you issue the RMU Optimize After_Journal command. The tape must be mounted as a FOREIGN volume.

The following tape qualifiers have meaning only when used in conjunction with the Format=New_Tape qualifier:

Active_IO
Block_Size
Crc
Group_Size
Density
Label
Owner_Uic
Protection
Rewind
Tape_Expiration

Follow these steps when you optimize an .aij file to tape:

1. Use the RMU Backup After_Journal command with the Format=Old_File qualifier to back up the .aij file to disk.
2. Use the RMU Optimize After_Journal command with the Format=New_Tape qualifier to optimize the backed up .aij file to tape.
3. Use the DCL BACKUP command to create a copy of the backed up .aij file as insurance.

If you enter the RMU Optimize After_Journal command with no Format qualifier, the default is Format=Old_File.

Group_Size=interval
Nogroup_Size
Specifies the frequency at which XOR recovery blocks are written to tape. The group size can vary from 0 to 100. Specifying a group size of zero or specifying the Nogroup_Size qualifier results in no XOR recovery blocks being written. The Group_Size qualifier is applicable only to tape, and its default value is device dependent. Oracle RMU returns an error message if this qualifier is used and the target device is not a tape device.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the .oaij file are to be labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas, and enclose the list of names within parentheses.

1.38 RMU Optimize After_Journal Command

Use the label that you specify for the RMU Optimize After_Journal command when you issue the RMU Recover command.

The Label qualifier can be used with indirect file references. See Section 1.3 for more information.

Librarian[=options]
Use the Librarian qualifier to back up files to data archiving software applications that support the Oracle Media Management interface. The backup file name specified on the command line identifies the stream of data to be stored in the Librarian utility. If you supply a device specification or a version number it will be ignored.

The Librarian qualifier accepts the following options:

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian utility.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed.

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Log
Nolog

Specifies that the optimization of the .aij file be reported to SYS$OUTPUT. When optimization activity is logged, the output from the Log qualifier provides the number of transactions committed and rolled back. You can specify the Trace qualifier with the Log qualifier. The default is the setting of the DCL VERIFY flag, which is controlled by the DCL SET VERIFY command.

Media_Loader
Nomedia_Loader

Use the Media_Loader qualifier to specify that the tape device receiving the backup file has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, Oracle RMU should recognize this fact. However, occasionally Oracle RMU does not recognize that a tape device has a loader or stacker. Therefore, when the first backup tape fills, Oracle RMU issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes Oracle RMU behaves as though a tape device has a loader or stacker when actually it does not.

If you find that Oracle RMU is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that Oracle RMU expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

Owner_Uic=user-id
Synonymous with the Owner qualifier. See the description of the Owner qualifier.

Owner=user-id
Specifies the owner of the tape volume set. The owner is the user who will be permitted to recover (roll forward) the database. The user-id parameter must be one of the following types of OpenVMS identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A UIC in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP

When used with tapes, the Owner qualifier applies to all continuation volumes. The Owner qualifier applies to the first volume only if the Rewind qualifier is also specified. If the Rewind qualifier is not specified, the optimization operation appends the file to a previously labeled tape, so the first volume can have a different protection than the continuation volumes.

Protection[=openvms-file-protection]
Specifies the system file protection for the .oaij file produced by the RMU Optimize After_Journal command.

The default file protection varies, depending on whether you write the .oaij file to disk or tape. This is because tapes do not allow delete or execute access and the SYSTEM account always has both read and write access to tapes. In addition, a more restrictive class accumulates the access rights of the less restrictive classes.

If you do not specify the Protection qualifier, the default protection is as follows:

- S:RWED,O:RE,G,W if the .oaij file is written to disk
- S:RW,O:R,G,W if the .oaij file is written to tape

If you specify the Protection qualifier explicitly, the differences in protection applied for backups to tape or disk as noted in the preceding paragraph are applied. Thus, if you specify Protection=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

Recovery_Method[=(Sequential | Scatter)]
Specifies how .aij records are to be ordered. You can specify one of two possible order types:

- Sequential—.aij records are ordered by physical dbkey in an area:page:line sequence. This order type is the default.
- Scatter—.aij records are ordered by a sort key of page:area:line (page number, area number, and line number). This order can allow the RMU Recover command to perform more effective I/O prefetching and writing to multiple storage areas simultaneously, typically where storage areas of the database are distributed among multiple disk devices.

Scatter ordering allows more disk devices to be active during the recovery process. This helps reduce idle CPU time and allows the recovery to complete in less time. However, because database configurations vary widely, Oracle recommends that you perform tests with both Scatter and Sequential ordering of the optimized after-image journals to determine which method produces the best results for your system.

Rewind
Norewind
Specifies that the tape that will contain the .oaij file be rewound before processing begins. The tape will be initialized according to the Label qualifier. The Norewind qualifier is the default and causes the optimized .oaij file to be written starting at the current logical end-of-tape (EOT).

The Norewind and Rewind qualifiers are applicable only to tape devices. Oracle RMU returns an error message if these qualifiers are used and the target device is not a tape device.

Tape_Expiration=date-time
Specifies the expiration date of the .oaij file on tape. Note that when Oracle RMU reads a tape, it looks at the expiration date in the file header of the first file on the tape and assumes the date it finds in that file header is the expiration date for the entire tape. Therefore, if you are writing an .oaij file to tape, specifying the Tape_Expiration qualifier only has meaning if the .oaij file is the first file on the tape. You can guarantee that the .oaij file will be the first file on the tape by specifying the Rewind qualifier and overwriting any existing files on the tape.

When the first file on the tape contains an expiration date in the file header, you cannot overwrite the tape before the expiration date unless you have the OpenVMS SYSPRV or BYPASS privilege.

Similarly, when you attempt to perform a recover operation with an .oaij file on tape, you cannot perform the recover operation after the expiration date recorded in the first file on the tape unless you have the OpenVMS SYSPRV or BYPASS privilege

By default, no expiration date is written to the .oaij file header. In this case, if the .oaij file is the first file on the tape, the tape can be overwritten immediately. If the .oaij file is not the first file on the tape, the ability to overwrite the tape is determined by the expiration date in the file header of the first file on the tape.

You cannot explicitly set a tape expiration date for an entire volume. The volume expiration date is always determined by the expiration date of the first file on the tape. The Tape_Expiration qualifier cannot be used with a backup operation to disk.

Trace
Notrace
Specifies that the optimization of the .aij file be traced. The default is the Notrace qualifier, where optimization is not traced. When optimization is traced, the output from the Trace qualifier identifies transactions in the .aij file by transaction sequence numbers (TSNs) and describes what Oracle RMU did with each transaction during the optimization process. You can specify the Log qualifier with the Trace qualifier.

Usage Notes

- To use the RMU Optimize After_Journal command for a database, you must have the RMU$BACKUP or RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- You cannot optimize an .aij file in the process of backing it up. You must first back up the .aij file, using the RMU Backup After_Journal command with the Format=Old_File qualifier, and then optimize it.
- As part of the optimization process, Oracle RMU sorts journal records by physical dbkey which improves I/O performance of the recovery. Because AIJ file optimization uses the OpenVMS Sort/Merge utility (SORT/MERGE) to sort journal records, you can improve the efficiency of the sort operation by changing the number and location of the work files used by SORT/MERGE. The number of work files is controlled by the RDMS$BIND_SORT_WORKFILES logical name. The allowable values are 1 through 10 inclusive, with a default value of 2. The location of these work

files can be specified with device specifications, using the SORTWORKn logical name (where *n* is a number from 0 to 9). See the OpenVMS documentation set for more information on using SORT/MERGE. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on using these Oracle Rdb logical names.

- Do not use the OpenVMS Alpha High Performance Sort/Merge utility (selected by defining the logical name SORTSHR to SYS$SHARE:HYPERSORT) when using the RMU Optimize After_Journal command. HYPERSORT does not support several of the interfaces the command uses. In addition, HYPERSORT does not report errors or warnings when it is used with the RMU Optimize After_Journal command.

  Make sure that the SORTSHR logical name is not defined to reference HYPERSORT.EXE.

- You can redirect the AIJ rollforward temporary work files to a different disk and directory location than the default (SYS$DISK: and current directory) by assigning a different directory to the RDM$BIND_AIJ_WORK_FILE logical in the LNM$FILE_DEV name table.

  This can be helpful in alleviating I/O bottlenecks that might be occurring in the default location.

- You can optimize an inactive .aij file that results, for example, from backing up and renaming an extensible .aij file. Backing up and renaming an extensible .aij file creates a new active, primary .aij file and makes the previous .aij file inactive. After optimizing the inactive .aij file, you can use the OpenVMS BACKUP command to back up the .oaij file. Note that you cannot optimize an active, primary .aij file.

- The RMU Optimize After_Journal command can read an .aij file on disk or a backed up .aij file on disk or on tape that is in the Old_File format, and it can write the .oaij file to disk or to tape in either Old_File or New_Tape format.

- If an RMU Optimize After_Journal command is issued from a batch job, tape requests and problems are reported to the tape operator. This occurs because tape requests and problems often require manual intervention, and if the RMU Optimize After_Journal command was issued from a batch job, the only available person might be the operator.

- When the RMU Optimize After_Journal command is issued interactively and a tape request or problem arises, Oracle RMU notifies the person who issued the command through the I/O channel assigned to the logical name SYS$COMMAND. After being notified of the problem, the user who issued the command can either fix the problem (if the user has access to the

tape drive) or contact the tape operator to ask the tape operator to fix the problem. The REQUEST command can be used to notify the tape operator, as follows:

```
$ REQUEST/REPLY/TO=TAPES -
_$ "Please Write Enable tape ATOZBG on drive $255$MUA6:"
```

- You should use the density values added in OpenVMS Version 7.2-1 for OpenVMS tape device drivers that accept them because previously supported values may not work as expected. If previously supported values are specified for drivers that support the OpenVMS Version 7.2-1 density values, the older values are translated to the Version 7.2-1 density values if possible. If the value cannot be translated, a warning message is generated, and the specified value is used.

  If you use density values added in OpenVMS Version 7.2-1 for tape device drivers that do not support them, the values are translated to acceptable values if possible. If the value cannot be translated, a warning message is generated and the density value is translated to the existing default internal density value (MT$K_DEFAULT).

  One of the following density-related errors is generated if there is a mismatch between the specified density value and the values that the tape device driver accepts:

  ```
  %DBO-E-DENSITY, TAPE_DEVICE:[000000]DATABASE.BCK; does not support
   specified density

  %DBO-E-POSITERR, error positioning TAPE_DEVICE:

  %DBO-E-BADDENSITY, The specified tape density is invalid for
   this device
  ```

- If you want to use an unsupported density value, use the VMS INITIALIZE and MOUNT commands to set the tape density. Do not use the Density qualifier.
- Because data stream names representing the database are generated based on the backup file name specified for the Oracle RMU backup command, you must either use a different backup file name to store the next backup of the database to the Librarian utility or first delete the existing data streams generated from the backup file name before the same backup file name can be reused.

  To delete the existing data streams stored in the Librarian utility, you can use a Librarian management utility or the Oracle RMU Librarian/Remove command.

Examples

Example 1

The following command creates an .oaij file named mf_personnel.oaij from the .aij file named mf_personnel.aij:

```
$ RMU/OPTIMIZE/AFTER_JOURNAL MF_PERSONNEL.AIJ MF_PERSONNEL.OAIJ
```

Example 2

The following example uses a density value with compression:

```
RMU/OPTIMIZE/AFTER_JOURNAL /DENSITY=(TK89,COMPACTION)/REWIND -
/LABEL=(LABEL1,LABEL2) MF_PERSONNEL.AIJ TAPE1:MF_PERSONNEL.OAIJ, TAPE2:
```

1.39 RMU Populate_Cache Command

Reads one or more tables and indexes from the database and stores the data rows or index nodes in caches if they exist.

Format

RMU/Populate_Cache root-file-spec

Command Qualifiers	Defaults
/Index=index-list	None
/[No]Log	Current DCL verify switch
/[No]Only_Cached	/Only_Cached
/Statistics_Interval=n	/Statistics_Interval=10
/Table=table-list	None
/Transaction_Type=transaction-mode	/Transaction-Type=Automatic

Description

The RMU Populate_Cache command allows one or more tables and indexes to be read from the database and stored in caches if they exist.

Sorted indexes are read top-down, one index level at a time. Hashed indexes are read by sequentially scanning the storage areas containing the hashed indexes and fetching all nodes and the system record from each database page. Data table rows are read by sequentially scanning the storage areas containing the table and fetching all rows of the relation.

Command Parameters

root-file-spec
Specifies the database root (.rdb) file. The default file type is .rdb.

Command Qualifiers

Index=Index-list
Specifies one or more indexes to fetch. All nodes are fetched from each index. If you list multiple indexes, separate the index names with a comma and enclose the list within parentheses. Wildcard characters asterisk (*) and percent sign (%) are allowed.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request that information about the operation be displayed, and the Nolog qualifier to prevent it. If you specify neither qualifer, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Only_Cached
Noonly_Cached
Specifies whether table or index content is to be read only if the table or index has an associated row cache. The default is to read data only from objects that have a cache. If the Noonly_Cached qualifier is specified, then all data from the specified tables or indexes is read.

Statistics_Interval=n
Specifies if statistics information is to be periodically displayed during the populate operation. The default for this qualifier is an interval of 10 seconds. If you do not use this qualifier no statistics are displayed.

Table=table-list
Specifies one or more tables to be processed. All rows are fetched from each table. If you list multiple tables, separate the table names with a comma, and enclose the list within parentheses. Wildcard characters asterisk (*) and percent sign (%) are allowed.

Transaction_Type=option
Allows you to specify the transaction mode for the transactions used to perform the analyze operation. Valid options are:

- Automatic
- Read_Only
- Noread_Only

You must specify an option if you use this qualifier.

If you do not specify any form of this qualifier, the Transaction_Type=Automatic qualifier is the default. This qualifier specifies that Oracle RMU is to determine the transaction mode.

The Transaction_Type=Read_Only qualifier specifies the transactions used to perform the analyze operation be set to read-only mode. When you explicitly set the transaction type to read-only, snapshots need not be enabled for all storage areas in the database, but must be enabled for those storage areas that are read. Otherwise, you receive an error and the analyze operation fails.

You might select this option if not all storage areas have snapshots enabled and you are analyzing objects that are stored only in storage areas with snapshots enabled. In this case, using the Transaction_Type=Read_Only qualifier allows you to perform the analyze operation and impose minimal locking on other users of the database.

The Transaction_Type=Noread_Only qualifier specifies that the transactions used to for the analyze operation be set to read/write mode. You might select this option if you want to eradicate the growth of snapshot files that occurs during a read-only transaction and are willing to incur the cost of increased locking that occurs during a read/write transaction.

Examples

Example 1: Using RMU /POPULATE_CACHE to cache tables and indices

This example uses wildcards to target specific tables and indices. The summary shows the nodes and rows fetched into the cache by RMU.

```
$ RMU /POPULATE_CACHE /LOG RMU_POPULATE1_4_DB /INDEX=I% /TABLE = T%
------------------------------------------------------------------------------
 ELAPSED:    0 00:00:00.06  CPU: 0:00:00.06  BUFIO: 37  DIRIO: 171  FAULTS: 692
Table "T1" : 2000 rows fetched
Table "T2" : 1000 rows fetched
Table "T3" : 2000 rows fetched
Table "T4" : 1000 rows fetched
Table "T5" : 2000 rows fetched
Index "I1" : 1035 nodes fetched
Index "I2" : 53 nodes fetched
Index "I3" : 107 nodes fetched
Index "I4" : 35 nodes fetched
Index "I5" : 1031 nodes fetched
Total : 10261 fetched
```

Usage Notes

- You must have the SQL SELECT privelege on the database for which the Populate_Cache command is being performed.

1.40 RMU Reclaim Command

Allows you to rapidly reclaim deleted dbkeys and locked space from database pages.

Format

RMU/Reclaim root-file-spec

Command Qualifiers	Defaults
/Area[=storage-area-list]	All storage areas
/[No]Log	/NoLog
/Page_Skip_Limit[=n]	25%
/Retry_Time[=n]	60 seconds
/Spam_Skip_Limit[=n]	25%

Description

Applications that specify the database attach attribute DBKEY SCOPE IS ATTACH can accumulate locked space and locked dbkeys within the database. If one user is connected to the database in DBKEY SCOPE IS ATTACH mode, all users are forced to operate in this mode, even if they are explicitly connected in TRANSACTION mode. No dbkeys are reused until the ATTACH session disconnects.

The RMU Reclaim command allows database keys of deleted rows to be rapidly reset in one or more storage areas. The RMU Reclaim command reads and updates all pages in a storage area, and, where possible, releases locked lines and locked free space so that they are available for later allocation.

Additional qualifiers can be used to reduce lock contention even further. These qualifiers will detect that other users are currently locking pages in the same storage area or SPAM interval that RMU/RECLAIM is processing and skip to the next storage area or SPAM interval based on a percent of the pages that RMU/RECLAIM is not able to process in a SPAM interval or storage area when reaching a specified limit.

1.40 RMU Reclaim Command

Command Parameters

root-file-spec
Specifies the database that contains locked areas or keys to be reclaimed. The default file extension is .rdb.

Command Qualifiers

Area[=storage-area-list]
Lists the storage areas to be reclaimed. The default is all storage areas.

Log
Nolog
Displays a log message as each storage area is reclaimed. The default is Nolog.

Page_Skip_Limit[=n]
If this qualifier is specified without the qualifier /SPAM_SKIP_LIMIT, the current storage area will be skipped by RMU/RECLAIM once this percent of pages in the area have been ignored by RMU/RECLAIM due to lock contention with other users. If this qualifier is specified with the qualifier /SPAM_SKIP_ LIMIT, the current SPAM interval will be abandoned and RMU will skip to the next SPAM interval in the current storage area once this percent of pages in the current SPAM interval have been ignored by RMU/RECLAIM due to lock contention with other users. The minimum value that can be specified is 1 percent. The maximum value that can be specified is 100 percent. The default value is 25 percent.

Retry_Time[=n]
Before completing, RMU/RECLAIM will perform one retry of the processing of any storage areas that have been skipped when /PAGE_SKIP_LIMIT or /SPAM_SKIP_LIMIT have been specified. The /RETRY_TIME qualifier specifies a wait time in seconds before RMU/RECLAIM reprocesses the storage areas where pages and SPAM intervals were previously skipped in order to allow more time for page contention to decrease. This qualifier can only be specified if the /PAGE_SKIP_LIMIT qualifier is also specified. The minimum value that can be specified is 0 seconds. The maximum value that can be specified is 3600 seconds (one hour). The default value is 60 seconds.

Spam_Skip_Limit[=n]
This qualifier specifies the percent of SPAM intervals that can be skipped in the current storage area due to lock contention with other users before the current storage area is skipped and the next storage area is processed. This qualifier can only be specified if the /PAGE_SKIP_LIMIT qualifier is also specified. The minimum value that can be specified is 1 percent. The

maximum value that can be specified is 100 percent. The default value is 25 percent.

Usage Notes

- The RMU Reclaim command runs on-line and does not require exclusive access. However, if there are any users connected to the database in DBKEY SCOPE IS ATTACH mode, the RMU/RECLAIM operation has greatly reduced effect. In order to release all possible locked space, there should be no users attached to the database in DBKEY SCOPE IS ATTACH mode.
- To allow database page locked space to be reclaimed, the database session that controls the locked space must be detached from the database. This can be accomplished by having each attached session disconnect and reconnect to the database.

Examples

Example 1

Shows the %RMU-I-RCLMPRCT message that will be output if the /LOG qualifier is specified, showing the percentage of pages that the RMU/RECLAIM command was able to process and did not have to skip because other users were accessing those pages in an incompatible locking mode.

```
$ RMU/RECLAIM/LOG/AREA=DEPARTMENTS MF_PERSONNEL
%RMU-I-RCLMAREA, Reclaiming area DEPARTMENTS
%RMU-I-RCLMPRCT, 704 pages processed of 706 total pages for area DEPARTMENTS,
approximately 99 %
 ELAPSED:    0 00:00:00.18  CPU: 0:00:00.02  BUFIO: 20  DIRIO: 257  FAULTS: 207
```

Example 2

Shows a /PAGE_SKIP_LIMIT of 1 percent being specified causing the DEPARTMENTS storage area to be skipped. The immediate reprocessing of the area has the same result.

1.40 RMU Reclaim Command

```
$ RMU/RECLAIM/LOG/AREA=SALARY_HISTORY/PAGE_SKIP_LIMIT=1 MF_PERSONNEL
%RMU-I-RCLMAREA, Reclaiming area SALARY_HISTORY
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area SALARY_HISTORY,
approximately 1 %
%RMU-I-RCLMRTRY, Retrying processing of area DEV:[DIR]SALARY_HISTORY.RDA;1
that did not complete
%RMU-I-RCLMAREA, Reclaiming area DEV:[DIR]SALARY_HISTORY.RDA;1
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area
DEV:[DIR]SALARY_HISTORY.RDA;1, approximately 1 %
%RMU-I-RCLMNOPRC, Retry of area DEV:[DIR]SALARY_HISTORY.RDA;1 did not complete
due to continued lock contention
 ELAPSED:    0 00:01:00.11  CPU: 0:00:00.04  BUFIO: 29  DIRIO: 23  FAULTS: 201
```

Example 3

Shows that the immediate reprocessing of the area (above) succeeds.

```
$ RMU/RECLAIM/LOG/AREA=SALARY_HISTORY/PAGE_SKIP_LIMIT=1 MF_PERSONNEL
%RMU-I-RCLMAREA, Reclaiming area SALARY_HISTORY
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area SALARY_HISTORY,
approximately 1 %
%RMU-I-RCLMRTRY, Retrying processing of area DEV:[DIR]SALARY_HISTORY.RDA;1
that did not complete
%RMU-I-RCLMAREA, Reclaiming area DEV:[DIR]SALARY_HISTORY.RDA;1
%RMU-I-RCLMPRCT, 706 pages processed of 706 total pages for area
DEV:[DIR]SALARY_HISTORY.RDA;1, approximately 100 %
%RMU-I-RCLMPROC, Retry of area DEV:[DIR]SALARY_HISTORY.RDA;1 succeeded
 ELAPSED:    0 00:01:00.11  CPU: 0:00:00.04  BUFIO: 29  DIRIO: 23  FAULTS: 201
$
```

Example 4

Shows a /RETRY_TIMEOUT wait of 20 seconds specified before the reprocessing of the area, which has the same results as the previous example.

```
$ RMU/RECLAIM/LOG/AREA=SALARY_HISTORY/PAGE_SKIP_LIMIT=1/RETRY_TIMEOUT=20
MF_PERSONNEL
%RMU-I-RCLMAREA, Reclaiming area SALARY_HISTORY
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area SALARY_HISTORY,
approximately 1 %
%RMU-I-RCLMWAIT, Reclaim waiting for 20 seconds before reprocessing areas that
did not complete
%RMU-I-RCLMRTRY, Retrying processing of area DEV:[DIR]SALARY_HISTORY.RDA;1
that did not complete
%RMU-I-RCLMAREA, Reclaiming area DEV:[DIR]SALARY_HISTORY.RDA;1
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area
DEV:[DIR]SALARY_HISTORY.RDA;1, approximately 1 %
%RMU-I-RCLMNOPRC, Retry of area DEV:[DIR]SALARY_HISTORY.RDA;1 did not complete
due to continued lock contention
 ELAPSED:    0 00:00:01.11  CPU: 0:00:00.02  BUFIO: 29  DIRIO: 27  FAULTS: 200
```

Example 5

Shows a /SPAM_SKIP_LIMIT of 1 percent is specified. A /RETRY_TIMEOUT wait of 60 seconds is specified and the reprocessing of the area succeeds.

```
$ RMU/RECLAIM/LOG/AREA=DEPARTMENTS/PAGE_SKIP_LIMIT=1/SPAM_SKIP_LIMIT=1
/RETRY_TIMEOUT=60 MF_PERSONNEL
%RMU-I-RCLMAREA, Reclaiming area DEPARTMENTS
%RMU-I-RCLMPRCT, 1 pages processed of 706 total pages for area DEPARTMENTS,
approximately 1 %
%RMU-I-RCLMWAIT, Reclaim waiting for 60 seconds before reprocessing areas that
did not complete
%RMU-I-RCLMRTRY, Retrying processing of area DEV:[DIR]DEPARTMENTS.RDA;1 that
did not complete
%RMU-I-RCLMAREA, Reclaiming area DEV:[DIR]DEPARTMENTS.RDA;1
%RMU-I-RCLMPRCT, 706 pages processed of 706 total pages for area
DEV:[DIR]DEPARTMENTS.RDA;1, approximately 100 %
%RMU-I-RCLMPROC, Retry of area DEV:[DIR]DEPARTMENTS.RDA;1 succeeded
 ELAPSED:    0 00:00:01.10  CPU: 0:00:00.02  BUFIO: 29  DIRIO: 23  FAULTS: 200
```

1.41 RMU Recover Command

Completes a database reconstruction by processing past transactions from the after-image journal (.aij) file or optimized after-image journal (.oaij) file against a database restored from a backup file.

Format

RMU/Recover aij-file-name-list

Command Qualifiers	Defaults
/Active_IO=max-reads	/Active_IO=3
/[No]After_Journal	/After_Journal
/Aij_Buffers=integer	/Aij_Buffers=20
/Areas [= storage-area[,...]]	All storage areas
/[No]Assist	See description
/[No]Automatic	/Automatic
/[No]Confirm[=options]	See description
/Directory=({Areas=,After_Journal= ,Row_Cache=}{directory_spec})	See description
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/Format={Old_File|New_Tape}	/Format=Old_File
/Just_Corrupt	See description
/Label=(label-name-list)	See description
/Librarian[=options]	None
/[No]Log	Current DCL verify value
/[No]Media_Loader	See description
/[No]Online	/Noonline
/Options=({Areas=,After_Journal= ,Row_Cache=}{option_file_spec})	See description
/Order_Aij_Files	See description
/Output=file-name	See description
/Prompt={Automatic|Operator|Client}	See description
/Resolve	See description
/[No]Rewind	/Norewind
/Root=root-file-name	See description
/[No]Trace	See Description
/Until=date-time	Current time

1.41 RMU Recover Command

Description

You can use the RMU Recover command to apply the contents of an .aij file to a restored copy of your database. Oracle RMU rolls forward the transactions in the .aij file into the restored copy of the database.

The RMU Recover command accepts a list of .aij or .oaij file names. Unless you specify the Noautomatic qualifier, the RMU Recover command attempts to automatically complete the recovery operation by applying the journals currently associated with the database in the current journal configuration if they are in the recovery sequence. For example, if you specify the following RMU Recover command, Oracle RMU not only recovers AIJ1, but also AIJ2, AIJ3, and so on, for all journals in the recovery sequence:

```
$ RMU/RECOVER AIJ1
```

However, note that this automatic recovery feature means that if you want to specify a termination condition, you must specify the Until qualifier. Example 1 demonstrates how to specify a termination condition with the Until qualifier.

If you are using extensible journals, you can also use the RMU Backup After_ Journal command to copy your database's .aij file to tape, and truncate the original .aij file without shutting down your database.

If you have backed up your .aij files (using the RMU Backup After_Journal command), these .aij files are no longer part of the current journal configuration and automatic recovery does not take place because Oracle RMU does not know where to find the .aij files. (There is one exception to this rule: if the only .aij file that has been backed up is the first .aij file in the recovery sequence, then automatic recovery occurs. You specify the backed up .aij file on the Oracle RMU command line and Oracle RMU can determine where the remaining on-disk .aij files reside.)

When automatic recover does not, or cannot occur, you must specify the complete list of .aij files on the RMU Recover command line to return your database to the desired state.

If your backup files were created using the Noquiet_Point qualifier, you must provide the names of all the .aij files in just one command. In addition, you must be careful to apply these .aij files to the database in the order in which they were created. Oracle RMU checks the validity of the journal file entries against your database and applies only appropriate transactions. If none of the transactions apply, you will receive a warning message.

You can access your database for retrieval of data between recovery steps, but you must not perform additional updates if you want to perform more recovery steps.

1.41 RMU Recover Command

If a system failure causes a recovery step to abort, you can simply issue the RMU Recover command again. Oracle RMU scans the .aij file until it finds the first transaction that has not yet been applied to your restored database. Oracle RMU begins recovery at that point.

Command Parameters

aij-file-name-list
A list of after-image journal (.aij) files to be applied to the database. You can supply this list using one of the following methods:

- List the .aij files on the command line in the order in which they were created. In other words, the oldest .aij file must be the first in the list.
- Use an asterisk (*) or percent sign (%) to represent the .aij files. The .aij files are processed in the order that they are presented by OpenVMS.
- Append all your .aij files into one file and supply a single .aij file name. You must be certain when you append the files that you append them in the order in which they were created.
- Use an indirect command file. Include an .aij file name on each line of the command file. If the number of .aij files needed for recovery is large, listing each one on the command line can exceed the maximum allowed command-line length. You can avoid this problem by using an indirect command file. See Section 1.3 for more information on indirect command files.

Command Qualifiers

Active_IO=max-reads
Specifies the maximum number of read operations from a backup device that the RMU Recover command attempts simultaneously. This is not the maximum number of read operations in progress; that value is a function of active system I/O operations.

The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

After_Journal
NoAfter_Journal
Oracle Rdb normally writes information to the after image journal (AIJ) file describing the creation of new after image journal files. There are cases where the user does not want RMU/RECOVER to process this information and recreate AIJ files, such as lack of disk space. If the /NoAfter_Journal qualifier is specified, no new Rdb AIJ files will be created by the current

RMU/RECOVER command and any AIJ file deletion records for those AIJ files which were not created will also be ignored.

The default if this qualifier is not specified is /AFTER_JOURNAL. Therefore, /NOAFTER_JOURNAL must be specified to ignore the creation of new AIJ files recorded in any AIJ file being recovered by the current RMU/RECOVER command.

Aij_Buffers=integer
Specifies the number of buffers to be used by the recovery process. The default is 20 buffers. The valid range is 2 to 1,048,576 buffers. If the database root file is available, you can use the RMU Dump After_Journal command with the Option=Statistics qualifier to find a recommended value for this qualifier. See Section 1.20 for details.

Areas[=storage-area[,...]]
Specifies the areas you want to recover. You can specify each storage area by name or by the area's ID number.

You should use the Areas qualifier only if you have inconsistent storage areas to recover. The default for the Areas qualifier is all storage areas that must be recovered to make the database consistent.

If the Areas qualifier is specified, a recovery operation by area recovers until the storage areas being rolled forward are current with the other storage areas, then recovery stops, regardless of the time specified by the Until qualifier.

When the Areas qualifier is not specified or the Areas=* qualifier is specified, Oracle RMU recovers all the storage areas in the database to the time specified by the Until qualifier or to the time of the last committed transaction in the .aij file that can be applied. When the Areas qualifier is specified without a value, Oracle RMU recovers to the earliest consistent state only those storage areas that are not current with the database root (.rdb) file of the database.

The Areas qualifier works in the following manner:

- If the Areas qualifier is specified without a value, Oracle RMU automatically determines what areas are inconsistent and recovers those areas. If an inconsistent area cannot be recovered because it is at a higher transaction sequence number (TSN) value than the database root file, the entire database is recovered even if the Areas qualifier was specified.

  See the *Oracle Rdb Guide to Database Maintenance* for information on TSNs.

- If the Areas qualifier is omitted or the Areas qualifier is specified as Areas=*, the entire database (all storage areas) is recovered.

- If the Areas qualifier is specified as Areas=(A1,A2,A3), only areas A1, A2, and A3 are recovered until they are consistent. If one of these areas is already consistent, or if an area is at a higher TSN value than the database root file, the entire database is recovered.
- If the Online qualifier is specified with the Areas qualifier (as in the first three list items) and the end result is that the entire database must be recovered, an error message is generated because you can recover only individual areas by using the Online qualifier, not the entire database.

You cannot use the Areas qualifier with the Just_Corrupt qualifier because the Areas qualifier implies recovery for all named areas and pages in those areas. (That is, use of the Just_Corrupt qualifier with the Areas qualifier is redundant.)

The Areas qualifier can be used with indirect file references. See Section 1.3 for more information.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Automatic
Noautomatic
Specifies whether or not Oracle RMU should attempt automatic recovery of .aij files. If you specify the Noautomatic qualifier, only the .aij file or files you list on the Oracle RMU command line are applied. If you specify the Automatic qualifier, Oracle RMU attempts to recover all the .aij files currently associated with the database.

The Automatic qualifier is the default; Oracle RMU attempts to recover all the .aij files currently associated with the database unless the .aij files have been backed up.

See the description section for more information on how automatic recovery works.

Confirm[=options]
Noconfirm
Specifies whether or not the RMU /RECOVER command causes the operator to be queried when an incorrect sequence of AIJ files is detected.

The default for interactive recoveries is /CONFIRM, which prompts the user to see if he wants to continue. The default for RMU/RECOVER/NOCONFIRM and RMU/RECOVER executed in batch jobs is to terminate the RMU/RECOVER at the point where the out of sequence AIJ file is detected (equivalent to RMU/RECOVER/CONFIRM=ABORT).

To override the default behavior, the user can continue to roll forward and ignore the missing AIJ file either by specifying the command syntax RMU/RECOVER/CONFIRM to get a prompt on whether to continue rolling forward if there is an AIJ sequence gap, or by specifying the syntax RMU/CONFIRM=CONTINUE if he does not want the prompt or is executing the RMU/RECOVER in a batch job.

Note

Oracle recommends that, in general, an incorrect journal sequence not be applied as a corrupt database may result.

The /Order_Aij_Files qualifier can be used to help ensure that the specified journals are applied in the correct order.

The Confirm qualifier accepts the following options:

- CONFIRM=CONTINUE

  Do not prompt the user if a sequence gap is detected on the next AIJ file to be rolled forward but ignore the missing AIJ file and continue rolling forward.

- CONFIRM=ABORT

  Do not prompt the user if a sequence gap is detected on the next AIJ roll forward but end the database recover at this point. This is the same as the default behavior for RMU/RECOVER/NOCONFIRM and RMU/RECOVER in batch.

Directory=({Areas=,After_Journal=,Row_Cache=}{directory_spec})
The Directory qualifier is used to alter the directory specifications for new storage areas, after image journal files and row caches created during the recovery of an Oracle Rdb database. This is only for the creation of new storage areas, after image journal files and row caches as recorded in the

after image journal files used for a database recovery by the RMU/RECOVER command, and only for modifying the directory specifications defined in the after image journal files for the new storage areas, after image journal files and row caches at the time RMU/RECOVER creates them.

This allows the user to control where newly created storage areas, after image journal files and row caches are located. Previously, they could only be put in the directory locations recorded in the after image journal files used for the recovery.

Logicals can be used for the directory specifications but must translate to valid directory specifications that exist when the RMU/RECOVER command is executed.

This qualifier can be used to specify one directory specification for all storage areas created, one directory specification for all after image journal files created and/or one directory specification for all row caches created.

The syntax for this qualifier for storage areas is:

```
/DIRECTORY=AREAS=directory_specification
```

The syntax for this qualifier for after image journal files is:

```
/DIRECTORY=AFTER_JOURNAL=directory_specification
```

The syntax for this qualifier for row caches is:

```
/DIRECTORY=ROW_CACHE=directory_specification
```

To specify two or more of these options with the /DIRECTORY qualifier, use the following syntax.

```
/DIRECTORY=(AREAS=directory_specification, -
            AFTER_JOURNAL=directory_specification, -
            ROW_CACHE=directory_specification)
```

The following example shows all three options used with the /DIRECTORY qualifier in the RMU/RECOVER command.

```
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
  /DIRECTORY=(AREAS=DISK:[DIRECTORY], -
 AFTER_JOURNAL=DISK:[DIRECTORY], -
  ROW_CACHE=DISK:[DIRECTORY]) -
  DISK:[DIRECTORY]:TEST_J1_BCK.AIJ
```

Encrypt=({Value=|Name=}[,Algorithm=])

The Encrypt qualifier is used to recover the database from an encrypted after image journal backup file.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

This feature only works for a newer format backup file which has been created using the Format=New_Tape qualifier. Therefore you have to specify the Format=New_Tape qualifier with this command if you also use the Encrypt qualifier.

Format=Old_Rms
Format=New_Tape
Synonymous with the Format=Old_File and Format=New_Tape qualifiers. See the description of those qualifiers.

Format=Old_File
Format=New_Tape
Specifies whether the backed up or optimized .aij file was written in the old (disk-optimized) or the new (tape-optimized) format. The Format=Old_File qualifier is the default. You must specify the same Format qualifier that was used with the RMU Backup After_Journal command or the RMU Optimize After_Journal command. If your .aij file resides on disk, you should use the Format=Old_File qualifier.

If you specified the Format=Old_File qualifier when you optimized or backed up the .aij file to tape, you must mount the backup media by using the DCL MOUNT command before you issue the RMU Recover command. Because the RMU Recover command will use RMS to read the tape, the tape must be mounted as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier with the MOUNT command).

If you specify the Format=New_Tape qualifier, you must mount the backup media by using the DCL MOUNT /FOREIGN command before you issue the RMU Recover command.

Similarly, if you specify OpenVMS access (you do not specify the /FOREIGN qualifier on the DCL MOUNT command) although your .aij backup was created using the Format=New_Tape qualifier, you will receive an RMU-F-MOUNTFOR error.

The following tape qualifiers have meaning only when used in conjunction with the Format=New_Tape qualifier:

Active_IO
Label

1.41 RMU Recover Command

Rewind

Just_Corrupt
Specifies that only inconsistent pages in the corrupt page table (CPT) and areas marked as inconsistent should be recovered. You can use this qualifier while users are attached to the database. The CPT is maintained in the .rdb file. (Note that if the corrupt page table becomes full, the area with the highest number of corrupt pages is marked corrupt and the individual pages for that area are removed from the CPT.)

You can use the Just_Corrupt qualifier with the Until qualifier to limit the recovery period to a particular point in time.

You cannot use the Areas qualifier with the Just_Corrupt qualifier because the Areas qualifier implies recovery for all named areas and pages in those areas. (That is, use of the Just_Corrupt qualifier with the Areas qualifier is redundant.)

If you do not specify the Just_Corrupt qualifier, all pages are recovered.

Just_Pages
This qualifier is replaced with the Just_Corrupt qualifier. See the description of the Just_Corrupt qualifier.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file have been labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas, and enclose the list of names within parentheses.

In a normal recovery operation, the Label qualifier you specify with the RMU Recover command should be the same Label qualifier you specified with the RMU Backup After_Journal command to back up your .aij files.

The Label qualifier can be used with indirect file references. See Section 1.3 for more information.

Librarian[=options]

Use the Librarian qualifier to restore files from data archiving software applications that support the Oracle Media Management interface. The file name specified on the command line identifies the stream of data to be retrieved from the Librarian utility. If you supply a device specification or a version number it will be ignored.

Oracle RMU supports retrieval using the Librarian qualifier only for data that has been previously stored by Oracle RMU using the Librarian qualifer.

The Librarian qualifier accepts the following options:

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian application.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

  This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed.

- RMU$DEBUG_SBT

This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Log
Nolog
Specifies that the recovery activity be logged. The default is the setting of the DCL VERIFY flag, which is controlled by the DCL SET VERIFY command.

When recovery activity is logged, the output from the Log qualifier provides the number of transactions committed, rolled back, and ignored during the recovery process. You can specify the Trace qualifier with the Log qualifier.

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device from which the .aij file is being read has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, Oracle RMU should recognize this fact. However, occasionally Oracle RMU does not recognize that a tape device has a loader or stacker. Therefore, when the first tape has been read, Oracle RMU issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes Oracle RMU behaves as though a tape device has a loader or stacker when actually it does not.

If you find that Oracle RMU is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that Oracle RMU expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

Online
Noonline
Specifies that the recover operation be performed while other users are attached to the database. The Online qualifier can only be used with the Area or Just_Corrupt qualifier. The areas or pages to be recovered are locked for exclusive access, so the operation is not compatible with other uses of the data in the areas or on the pages specified.

The default is the Noonline qualifier.

Options=({Areas=,After_Journal=,Row_Cache=}{option_file_spec})
The Option qualifier is used to specify option files that specify different directory specifications for specific storage areas, after journal files and row caches. One option file must be specifed to select one or more storage areas for directory modification, one option file must be specified to select one or more after image journal files for directory modification and one option file must be specified to select one or more row caches for directory modification.

The syntax for this qualifier for storage areas is:

```
/OPTIONS=AREAS=option_file_specification
```

The syntax for this qualifier for after image journal files is:

```
/OPTIONS=AFTER_JOURNAL=option_file_specification
```

The syntax for this qualifier for row caches is:

```
/OPTIONS=ROW_CACHE=option_file_specification
```

To specify two or more of these options with the /OPTIONS qualifier, use the following syntax.

```
/OPTIONS=(AREAS=option_file_specification, -
          AFTER_JOURNAL=option_file_specification, -
          ROW_CACHE=option_file_specification)
```

The following example shows all three options used with the /OPTIONS qualifier in the RMU/RECOVER command.

```
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
  /OPTIONS=(AREAS=DISK:[DIRECTORY]RECOVOPTAREAS.OPT, -
  AFTER_JOURNAL=DISK:[DIRECTORY]RECOVOPTAIJ.OPT, -
  ROW_CACHE=DISK:[DIRECTORY]RECOVOPTRCACHE.OPT) -
  DISK:[DIRECTORY]TEST_J1_BCK.AIJ
```

Order_Aij_Files
Specifies that the input after-image journal files are to be processed in ascending order by sequence number. The .aij files are each opened, the first block is read to determine the sequence number, and the files are closed prior to the sequence number sorting operation. The Order_Aij_Files can be especially useful if you use wildcards to specify .aij files.

The Order_Aij_Files qualifier can also eliminate some .aij files from processing if they are known to be prior to the database recovery sequence starting point.

Note that due to the fact that the .aij backup files might have more than one journal sequence in them, it is not always possible for RMU to eliminate every journal file that might otherwise appear to be unneeded. But for those journals where RMU is able to know for certain that the journal will not be needed

based on the database recovery restart information, journals can be avoided from having to be processed.

Output=file-name
Redirects the log and trace output (selected with the Log and Trace qualifiers) to the named file. If this qualifier is not specified, the output generated by the Log and Trace qualifiers, which can be voluminous, is displayed on your terminal.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Resolve
Recovers a corrupted database and resolves an unresolved transaction by completing the transaction.

See the RMU Recover Resolve command (Section 1.42) for a description of the options available with the Resolve qualifier.

Rewind
Norewind
Specifies that the tape that contains the backup file be rewound before processing begins. The tape is searched for the backup file starting at the beginning-of-tape (BOT). The Norewind qualifier is the default and causes the backup file to be searched starting at the current tape position.

The Rewind and Norewind qualifiers are applicable only to tape devices. Oracle RMU returns an error message if these qualifiers are used and the target device is not a tape device.

Root=root-file-name
Specifies the name of the database to which the journal should be applied. The Root qualifier allows you to specify a copy of a database instead of the original whose file specification is in the .aij file. Use the Root qualifier to specify the new location of your restored database root (.rdb) file.

Specifying this qualifier lets you roll forward a database copy (possibly residing on a different disk) by following these steps:

1. Use the RMU Backup command to make a backup copy of the database:

```
$ RMU/BACKUP MF_PERSONNEL.RDB MF_PERS_FULL_BU.RBF
```

This command writes a backup file of the database mf_personnel to the file mf_pers_full_bu.rbf.

2. Use the RMU Restore command with the Root and Directory qualifiers, stating the file specifications of the database root and storage area files in the database copy.

```
$ RMU/RESTORE/ROOT=DB3:[USER]MF_PERSONNEL/DIRECTORY=DB3:[USER] -
_$ MF_PERS_FULL_BU
```

This command restores the database on disk DB3: in the directory [USER]. Default file names and file extensions are used.

3. If the database uses after-image journaling, you can use the RMU Recover command to roll forward the copy.

```
$ RMU/RECOVER DBJNL.AIJ/ROOT=DB3:[USER]MF_PERSONNEL.RDB
```

Thus, transactions processed and journaled since the backup operation are recovered on the copy on the DB3: disk.

Correct operation of this procedure requires that there are no write transactions for the restored copy between the restore and recover steps.

If you do not specify the Root qualifier, Oracle RMU examines the .aij file to determine the exact name of the database root (.rdb) file to which the journaled transactions will be applied. This name, which was stored in the .aij file, is the full file specification that your .rdb file had when after-image journaling was enabled.

The journal file for a single-file database does not include the file name for the database; to recover a single-file database, you must specify the location of the database to be recovered by using the Root qualifier.

Trace
Notrace

Specifies that the recovery activity be logged. The default is the setting of the DCL VERIFY flag, which is controlled by the DCL SET VERIFY command. When recovery activity is logged, the output from the Trace qualifier identifies transactions in the .aij file by TSN and describes what Oracle RMU did with each transaction during the recovery process. You can specify the Log qualifier with the Trace qualifier.

Until=date-time
Use the Until qualifier to limit the recovery to those transactions in the journal file bearing a starting timestamp no later than the specified time. For example, suppose your database fails today, but you have reason to believe that something started to go wrong at noon yesterday. You might decide that you only want to restore the database to the state it was in as of noon yesterday. You could use the Until qualifier to specify that you only want to recover those transactions that have a timestamp of noon yesterday or earlier.

If you do not specify the Until qualifier, all committed transactions in the .aij file will be applied to your database. If you specify the Until qualifier, but do not specify a date-time, the current time is the default.

If the Until qualifier is specified with a recover-by-area operation, the operation terminates when either the specified time is reached in the transaction sequence or the specified storage areas become consistent with the other storage areas; whichever condition occurs first.

Usage Notes

- To use the RMU Recover command for a database, you must have the RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- You can use the RMU Backup After_Journal command to copy an extensible .aij file to tape and truncate the original .aij file without shutting down your database.
- Transactions are applied to the restored copy of your database in the order indicated by their commit sequence number and the commit record in the .aij file; timestamps are not used for this purpose. Therefore, you need not be concerned over time changes made to the system (for example, resetting the time for United States daylight saving time) or inconsistencies in the system time on different nodes in a cluster.

  The only occasion when timestamps are considered in the application of .aij files is when you specify the Until qualifier. In this case, the timestamp is used only as the point at which to stop the recovery, not as a means to serialize the order in which transactions are applied. See the description of the Until qualifier for more information.

- You can redirect the AIJ rollforward temporary work files and the database recovery (DBR) redo temporary work files to a different disk and directory location than the default (SYS$DISK) by assigning a different directory to the RDM$BIND_AIJ_WORK_FILE logical in the LNM$FILE_DEV name table and a different directory to the RDM$BIND_DBR_WORK_FILE logical in the LNM$SYSTEM_TABLE, respectively.

  This can be helpful in alleviating I/O bottlenecks that might be occurring in the default location.

- In a normal recovery operation, the Format and Label qualifiers you specify with the RMU Recover command should be the same Format and Label qualifiers you specified with the RMU Backup After_Journal command to back up your .aij files or with the RMU Optimize After_Journal command to optimize your .aij files.

  For more information on the type of access to specify when mounting tapes, see the description of the Format=Old_File and Format=New_Tape qualifiers in the Command Qualifiers section.

- The following restrictions apply to using optimized .aij files with recovery operations:

  - Optimized .aij files cannot be used as part of by-area recovery operations (recovery operations that use the RMU Recover command with the Area qualifier).
  - Optimized .aij files cannot be used as part of by-page recovery operations (recovery operations that use the RMU Recover command with the Just_Corrupt qualifier).
  - Optimized .aij files cannot be specified for an RMU Recover command that includes the Until qualifier. The optimized .aij file does not retain enough of the information from the original .aij file for such an operation.
  - Optimized .aij files cannot be used with a recovery operation if the database has been modified since the .aij file was optimized.

  The workaround for these restrictions against using optimized .aij files in recovery operations is to use the original, unoptimized .aij file in the recovery operation instead.

- You can read your database between recovery steps, but you must not perform additional updates if you want to perform more recovery steps.

1.41 RMU Recover Command

- If a system failure causes a recovery step to abort, you can simply issue the RMU Recover command again. Oracle RMU scans the .aij file until it finds the first transaction that has not yet been applied to your restored database. Oracle RMU begins recovery at that point.
- You can use the RMU Recover command to apply the contents of an .aij file to a restored copy of your database. Oracle RMU will roll forward the transactions in the .aij file into the restored copy of the database. You can use this feature to maintain an up-to-date copy of your database for fast recovery after a failure. To do this, use the RMU Recover command to periodically apply your .aij files to a separate copy of the database.

  When you employ this procedure for fast recovery, you must be absolutely certain that no one will execute an update transaction on the database copy. Should someone execute an update transaction, it might result in the inability to apply the .aij files correctly.
- See the *Oracle Rdb Guide to Database Maintenance* for information on the steps Oracle RMU follows in tape label checking.
- When you use an optimized after-image journal for recovery, the optimal number of buffers specified with the Aij_Buffers qualifier depends on the number of active storage areas being recovered. For those journals optimized with Recover_Method=Sequential, a buffer count of 250 to 500 is usually sufficient.

  When you use journals optimized with Recover_Method=Scatter, reasonable performance can usually be attained with a buffer count of about five times the number of active storage areas being recovered (with a minimum of about 250 to 500 buffers).
- The number of asynchronous prefetch (APF) buffers is also a performance factor during recovery. For recovery operations of optimized after-image journals, the RMU Recover command sets the number of APF buffers (also known as the APF depth) based on the values of the process quotas ASTLM, BYTLM, and the specified AIJ_Buffers value. The APF depth is set to the maximum of:
  - 50% of the ASTLM process quota
  - 50% of the DIOLM process quota
  - 25% of the specified AIJ_Buffers value

  The accounts and processes that perform RMU Recover operations should be reviewed to ensure that various quotas are set to ensure high levels of I/O performance. The following table lists suggested quota values for recovery performance.

Quota	Setting
DIOLM	Equal to or greater than half of the count of database buffers specified by the AIJ_Buffers qualifier. Miminum of 250.
BIOLM	Equal to or greater than the setting of DIOLM.
ASTLM	Equal to or greater than 50 more than the setting of DIOLM.
BYTLM	Equal to or greater than 512 times the database buffer size times one half the value of database buffers specified by the AIJ_Buffers qualifier. Based on a 12-block buffer size and the desire to have 100 asynchronous I/O requests outstanding (either reading or writing), the minimum suggested value is 614,400 for a buffer count of 200.
WSQUOTA WSEXTENT	Large enough to avoid excessive page faulting.
FILLM	50 more than the count of database storage areas and snapshot storage areas.

Examples

Example 1

In the following example, the RMU Recover command requests recovery from the .aij file personnel.aij located on PR$DISK in the SMITH directory. It specifies that recovery should continue until 1:30 P.M. on May 7, 1996. Because the Trace qualifier is specified, the RMU Recover command displays detailed information about the recovery operation to SYS$OUTPUT.

```
$ RMU/RECOVER/UNTIL="07-MAY-1996 13:30"/TRACE PR$DISK:[SMITH]PERSONNEL
%RMU-I-LOGRECDB, recovering database file DISK1:[DB.70]MF_PERSONNEL.RDB;1
%RMU-I-LOGRECSTAT, transaction with TSN 0:256 committed
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJAUTOREC, starting automatic after-image journal recovery
%RMU-I-AIJONEDONE, AIJ file sequence 1 roll-forward operations completed
%RMU-W-NOTRANAPP, no transactions in this journal were applied
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
 needed will be 1
```

1.41 RMU Recover Command

Example 2

The following example shows how to use .aij files to recover a database:

```
SQL> CREATE DATABASE FILENAME DISK1:[SAMPLE]TEST_DB
cont> RESERVE 5 JOURNALS;
SQL> --
SQL> -- Use the DISCONNECT ALL statement to detach from the database,
SQL> -- then issue the ALTER DATABASE statement that automatically
SQL> -- invokes the specified database.
SQL> --
SQL> DISCONNECT ALL;
SQL> --
SQL> -- Create after-image journaling. The .aij files are given the
SQL> -- names aij_one.aij and aij_two.aij (and are placed on a disk
SQL> -- other than the disk holding the .rdb and .snp files):
SQL> --
SQL> ALTER DATABASE FILENAME DISK1:[SAMPLE]TEST_DB
cont> JOURNAL IS ENABLED
cont>   ADD JOURNAL AIJ_ONE
cont>       FILENAME 'USER$DISK:[CORP]AIJ_ONE'
cont>       BACKUP FILENAME 'USER$DISK2:[CORP]AIJ_ONE'
cont>   ADD JOURNAL AIJ_TWO
cont>       FILENAME 'USER$DISK3:[CORP]AIJ_TWO'
cont>       BACKUP FILENAME 'USER$DISK4:[CORP]AIJ_TWO';
SQL> EXIT
$ !
$ ! Using the RMU Backup command, make a backup copy of the database.
$ ! This command ensures that you have a copy of the
$ ! database at a known time, in a known state.
$ !
$ RMU/BACKUP DISK1:[SAMPLE]TEST_DB USER2:[BACKUPS]TEST_BACKUP.RBF
$ !
$ ! Now you can use SQL with after-image journaling enabled.
$ !
$ SQL
SQL> --
SQL> -- Attach to the database and perform some data definition
SQL> -- and storage.
SQL> --
SQL> ATTACH 'FILENAME DISK1:[SAMPLE]TEST_DB';
SQL> CREATE TABLE TABLE1 (NEW_COLUMN CHAR(10));
SQL> INSERT INTO TABLE1 (NEW_COLUMN) VALUES ('data');
SQL> COMMIT;
SQL> EXIT
$ !
$ ! Imagine that a disk failure occurred here. In such a situation,
$ ! the current database is inaccessible. You need a prior copy
$ ! of the database to roll forward all the transactions in the
$ ! .aij file.
$ !
```

```
$ !
$ ! You know that the backup file of the database is
$ ! uncorrupted. Use the RMU Restore command to restore and recover
$ ! the database. You do not have to issue the RMU Recover command
$ ! because the RMU Restore command will automatically recover the
$ ! database.
$ !
$ RMU/RESTORE/NOCDD_INTEGRATE/DIR=DDV21:[TEST] -
_$ USER2:[BACKUPS]TEST_BACKUP.RBF
%RMU-I-AIJRSTAVL, 2 after-image journals available for use
%RMU-I-AIJRSTMOD, 1 after-image journal marked as "modified"
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure
 future recovery
%RMU-I-LOGRECDB, recovering database file DDV21:[TEST]TEST_DB.RDB;1
%RMU-I-AIJAUTOREC, starting automatic after-image journal recovery
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJONEDONE, AIJ file sequence 1 roll-forward operations completed
%RMU-W-NOTRANAPP, no transactions in this journal were applied
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
 number needed will be 1
```

Example 3

The following example demonstrates how the recovery operation works when there are .aij backup files to be applied. First you must restore the database by using the RMU Restore command with the Norecovery qualifier, then apply the backed up .aij file by using the RMU Recover command. Oracle RMU will complete the recovery with the .aij files that were current when the restore operation was invoked. This example assumes that three .aij files have been added to the mf_personnel database prior to the first shown backup operation and that journaling is enabled.

```
$ ! Create a backup file of the complete and full database.
$ !
$ RMU/BACKUP MF_PERSONNEL DISK1:[BACKUPS]MF_PERSONNEL_BCK.RBF
$ !
$ ! Updates are made to the SALARY_HISTORY and DEPARTMENTS tables.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> UPDATE SALARY_HISTORY
cont> SET SALARY_END='20-JUL-1993 00:00:00.00'
cont> WHERE SALARY_START='14-JAN-1983 00:00:00'
cont> AND EMPLOYEE_ID='00164';
```

1.41 RMU Recover Command

```
SQL> INSERT INTO DEPARTMENTS
cont> (DEPARTMENT_CODE, DEPARTMENT_NAME,
cont>  MANAGER_ID,BUDGET_PROJECTED, BUDGET_ACTUAL)
cont> VALUES ('WLNS', 'WELLNESS CENTER', '00188',0,0);
SQL> COMMIT;
SQL> DISCONNECT DEFAULT;
SQL> EXIT
$ !
$ ! Create a backup file of the .aij files.
$ !
$ RMU/BACKUP/AFTER_JOURNAL MF_PERSONNEL DISK2:[BACKUP]MF_PERS_AIJBCK
$ !
$ ! An additional update is made to the DEPARTMENTS table.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> INSERT INTO DEPARTMENTS
cont> (DEPARTMENT_CODE, DEPARTMENT_NAME, MANAGER_ID,BUDGET_PROJECTED,
cont> BUDGET_ACTUAL)
cont> VALUES ('facl', 'FACILITIES', '00190',0,0);
SQL> COMMIT;
SQL> DISCONNECT DEFAULT;
SQL> EXIT;
$
$ ! Assume the disk holding the SALARY_HISTORY and DEPARTMENTS
$ ! storage areas is lost. Restore only those areas.  Specify
$ ! the Norecovery qualifier since you will need to apply the
$ ! .aij backup file.
$
$ RMU/RESTORE/AREA DISK1:[BACKUPS]MF_PERSONNEL_BCK.RBF -
_$ SALARY_HISTORY, DEPARTMENTS/NORECOVER
$ !
$ ! Now recover the database.  Although you only specify the .aij
$ ! backup file, Oracle RMU will automatically continue the
$ ! recovery with the current journals in the recovery sequence after
$ ! the backed up .aij files have been applied.
$ !
$ RMU/RECOVER/LOG DISK2:[BACKUP]MF_PERS_AIJBCK
%RMU-I-AIJBADAREA, inconsistent storage area DISK3:[STO_AREA]
 DEPARTMENTS.RDA;1  needs AIJ sequence number 0
%RMU-I-AIJBADAREA, inconsistent storage area
 DISK3:[STO_AREA]SALARY_HISTORY.RDA;1 needs AIJ sequence number 0
%RMU-I-LOGRECDB, recovering database file
 DISK3:[DATABASE]MF_PERSONNEL.RDB;1
%RMU-I-LOGOPNAIJ, opened journal file
 DISK2:[BACKUP]MF_PERS_AIJBCK.AIJ;1
```

```
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations
 completed
%RMU-I-LOGRECOVR, 3 transactions committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 0 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
 number needed will be 1
%RMU-I-AIJAUTOREC, starting automatic after-image journal recovery
%RMU-I-LOGOPNAIJ, opened journal file DISK4:[CORP]AIJ_TWO.AIJ;1
%RMU-I-AIJONEDONE, AIJ file sequence 1 roll-forward operations
 completed
%RMU-I-LOGRECOVR, 2 transactions committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 0 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
 number needed will be 2
%RMU-I-AIJALLDONE, after-image journal roll-forward operations
 completed
%RMU-I-LOGSUMMARY, total 5 transactions committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 0 transactions ignored
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJGOODAREA, storage area DISK3:[STO_AREA]DEPARTMENTS.RDA;1
 is now consistent
%RMU-I-AIJGOODAREA, storage area DISK3:[STO_AREA]SALARY_HISTORY.RDA;1
 is now consistent
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
 number needed will be 2
$ !
$ ! Database is restored and recovered and ready to use.
$ !
```

Example 4

The following example demonstrates how to recover all the known inconsistent pages in a database. Assume the RMU Show Corrupt_Pages command reveals that page 60 in the EMPIDS_LOW storage area is inconsistent and pages 11 and 123 in the EMPIDS_MID storage area is inconsistent. The RMU Recover command is issued to recover on line all pages logged inconsistent in the corrupt page table (CPT). After the recovery operation, the CPT will be empty.

1.41 RMU Recover Command

```
$ RMU/RECOVER/JUST_CORRUPT/ONLINE/LOG MF_PERSONNEL.AIJ
%RMU-I-AIJBADPAGE, inconsistent page 11 from storage area
 DISK1:[TEST5]EMPIDS_OVER.RDA;1 needs AIJ sequence number 0
%RMU-I-AIJBADPAGE, inconsistent page 60 from storage area
 DISK1:[TEST5]EMPIDS_LOW.RDA;1 needs AIJ sequence number 0
%RMU-I-AIJBADPAGE, inconsistent page 123 from storage area
 DISK1:[TEST5]EMPIDS_OVER.RDA;1 needs AIJ sequence number 0
%RMU-I-LOGRECDB, recovering database file
 DISK2:[TEST5]MF_PERSONNEL.RDB;1
%RMU-I-LOGOPNAIJ, opened journal file DISK3:[TEST5]MF_PERSONNEL.AIJ;1
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations
 completed
%RMU-I-LOGRECOVR, 1 transaction committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 0 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJALLDONE, after-image journal roll-forward operations
 completed
%RMU-I-LOGSUMMARY, total 1 transaction committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 0 transactions ignored
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJGOODPAGE, page 11 from storage area
 DISK1:[TEST5]EMPIDS_OVER.RDA;1 is now consistent
%RMU-I-AIJGOODPAGE, page 60 from storage area
 DISK1:[TEST5]EMPIDS_LOW.RDA;1 is now consistent
%RMU-I-AIJGOODPAGE, page 123 from storage area
 DISK1:[TEST5]EMPIDS_OVER.RDA;1 is now consistent
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
 number needed will be 0
```

Example 5

In the following example, note that the backed up AIJ files are specified in the order B1, B3, B2, B4 representing sequence numbers 1, 3, 2, 4. The /ORDER_ AIJ_FILES sorts the journals to be applied into ascending sequence order and then is able to remove B1 from processing because the database recovery starts with AIJ file sequence 2 as shown in the RMU/RESTORE output.

```
$ RMU/RESTORE/NEW/NOCDD/NOAFTER FOO
%RMU-I-RESTXT_00, Restored root file DUA0:[DB]FOO.RDB;16
    .
    .
    .
%RMU-I-AIJRECFUL, Recovery of the entire database starts with
AIJ file sequence 2
%RMU-I-COMPLETED, RESTORE operation completed at 24-MAY-2007 12:23:32.99
$!
$ RMU/RECOVER/LOG/ORDER_AIJ_FILES B1,B3,B2,B4
    .
    .
    .
%RMU-I-LOGOPNAIJ, opened journal file DUA0:[DB]B2.AIJ;24
%RMU-I-LOGRECSTAT, transaction with TSN 0:256 ignored
%RMU-I-LOGRECSTAT, transaction with TSN 0:257 ignored
%RMU-I-RESTART, restarted recovery after ignoring 2 committed transactions
%RMU-I-AIJONEDONE, AIJ file sequence 2 roll-forward operations completed
%RMU-I-LOGRECOVR, 0 transactions committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 2 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJNXTSEQ, to continue this AIJ file recovery, the
sequence number needed will be 3
    .
    .
    .
```

Example 6

The following example shows the "/CONFIRM=ABORT" syntax used so that RMU/RECOVER will not continue rolling forward if a sequence gap is detected. This is the default behavior if /NOCONFIRM is specified or for batch jobs. Note that the exit status of RMU will be "%RMU-E-AIJRECESQ" if the recovery is aborted due to a sequence gap. It is always a good policy to check the exit status of RMU, especially when executing RMU in batch jobs.

```
RMU/RECOVER/CONFIRM=ABORT/LOG/ROOT=user$test:foo faijbck1,faijbck2,faijbck4
%RMU-I-LOGRECDB, recovering database file DEVICE:[DIRECTORY]FOO.RDB;1
```

1.41 RMU Recover Command

```
%RMU-I-LOGOPNAIJ, opened journal file DEVICE:[DIRECTORY]FAIJBCK4.AIJ;1
 at 25-FEB-2009 17:27:42.29
%RMU-W-AIJSEQAFT, incorrect AIJ file sequence 8 when 7 was expected
%RMU-E-AIJRECESQ, AIJ roll-forward operations terminated due to sequence error
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-LOGSUMMARY, total 2 transactions committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 0 transactions ignored
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
 needed will be 7
%RMU-I-AIJNOENABLED, after-image journaling has not yet been enabled
```

Example 7

The following example shows the "/CONFIRM=CONTINUE" syntax used so that RMU/RECOVER will continue rolling forward if a sequence gap is detected.

```
RMU/RECOVER/CONFIRM=CONTINUE/LOG/ROOT=user$test:foo faijbck1,faijbck2,faijbck4
%RMU-I-LOGRECDB, recovering database file DEVICE:[DIRECTORY]FOO.RDB;1

%RMU-I-LOGOPNAIJ, opened journal file DEVICE:[DIRECTORY]FAIJBCK4.AIJ;1
 at 25-FEB-2009 17:26:04.00
%RMU-W-AIJSEQAFT, incorrect AIJ file sequence 8 when 7 was expected
%RMU-I-AIJONEDONE, AIJ file sequence 8 roll-forward operations completed
%RMU-I-LOGRECOVR, 1 transaction committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 0 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence number
 needed will be 9
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-LOGSUMMARY, total 3 transactions committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 0 transactions ignored
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
 needed will be 9
%RMU-I-AIJNOENABLED, after-image journaling has not yet been enabled
```

Example 8

In the following example, the creation of the new after image journal file J2.AIJ for the MF_PERSONNEL database is journaled to the current after image journal file RMU_RECOVER_4.AIJ_1. When the MF_PERSONNEL database is deleted and then restored from the MF_PERSONNEL.RBF file and then recovered from RMU_RECOVER_4.AIJ_1 using the /NOAFTER_ JOURNAL qualifier, the J2.AIJ file does not get created.

```
$!
$! Change the database to enable after image journaling to the
$! RMU_RECOVER_4.AIJ_1 AIJ file
$!
$ SQL$
      alter database filename MF_PERSONNEL
      reserve 10 journals
      journal filename disk:[directory]RMU_RECOVER_4.AIJ_1;
%RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
      exit
$!
$! Backup the database
$!
$ RMU/BACKUP/NOLOG MF_PERSONNEL DISK:[DIRECTORY]MF_PERSONNEL.RBF
$!
$! Add a new AIJ file J2.AIJ to put a create AIJ file record in the current
$! RMU_RECOVER_4.AIJ_1 AIJ file
$!
$ SQL$
 alter database file mf_personnel
       add journal j2 filename disk:[directory]j2 allocation is 1000 blocks;
exit
$!
$! Drop the database and then restore the database from the backup RBF file
$!
$ SQL$
  drop database filename MF_PERSONNEL;
  exit
$!
$! Delete the added j2.aij file
$!
$ DELETE DISK:[DIRECTORY]J2.AIJ;*
$!
$! Restore the database
$!
$ RMU/RESTORE/NOCDD/NOLOG/NOAFTER_JOURNAL -
  /ROOT=DISK:[DIRECTORY]MF_PERSONNEL.RDB DISK:[DIRECTORY]MF_PERSONNEL.RBF
$!
$! Recover the database from RMU_RECOVER_4.AIJ_1 to show that the J2.AIJ
$! file does not get created if RMU/RECOVER/NOAFTER_JOURNAL is specified
$!
$ RMU /RECOVER /NOAFTER_JOURNAL /NOLOG /ROOT=DISK:[DIRECTORY]MF_PERSONNEL.RDB -
    DISK:[DIRECTORY]RMU_RECOVER_4.AIJ_1
%RMU-I-LOGRECDB, recovering database file DISK:[DIRECTORY]MF_PERSONNEL.RDB;1
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
needed will be 1
%RMU-I-AIJNOENABLED, after-image journaling has not yet been enabled
$ DIR DISK:[DIRECTORY]J2.AIJ
%DIRECT-W-NOFILES, no files found
```

Example 9

1.41 RMU Recover Command

In the following example for database DISK:[HOME]TEST.RDB, the first RMU/RECOVER command uses the /DIRECTORY qualifier to change all directories from DISK:[HOME] to DISK:[NEW] for the storage areas TEST_A3 and TEST_A4, the after image journal files TEST_J3 and TEST_J4 and the row caches TEST_A3 and TEST_A4, whose creation was recorded in the journal file TEST_J1_BCK.AIJ. Then, the /DIRECTORY_OPTIONS qualifier in the RMU/DUMP command is used to create option files for all TEST database after image journal files, storage areas and row caches, including those with directories changed to DISK:[NEW]. The TEST database is then deleted and again restored with all directories again set to DISK:[HOME]. The second RMU/RECOVER command then uses the /OPTIONS qualifier to specify the option files created by RMU/DUMP, which include the after image journal file, storage area and row cache directories changed by the first RMU/RECOVER to DISK:[NEW], to change the directories back again from the journaled directory specification of DISK:[HOME] to DISK:[NEW].

```
$ ! Create the TEST database
$
$ SQL
  CREATE DATABASE FILENAME DISK:[HOME]TEST
  NUMBER OF CLUSTER NODES 1
  RESERVE 6 STORAGE AREAS
  RESERVE 6 JOURNALS
  RESERVE 6 CACHE SLOTS
  ROW CACHE IS ENABLED
  CREATE STORAGE AREA RDB$SYSTEM FILENAME DISK:[HOME]TEST_SYS
  CREATE STORAGE AREA TEST_A1 FILENAME DISK:[HOME]TEST_A1
  CREATE STORAGE AREA TEST_A2 FILENAME DISK:[HOME]TEST_A2
  CREATE CACHE TEST_A1
    CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
  CREATE CACHE TEST_A2
    CACHE SIZE 200 ROWS ROW LENGTH 200 BYTES LOCATION IS 'DISK:[HOME]';
  DISCONNECT ALL;
```

```
  ALTER DATABASE FILENAME TEST
    ADD JOURNAL TEST_J1 FILENAME DISK:[HOME]TEST_J1.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J1_BCK.AIJ
    ADD JOURNAL TEST_J2 FILENAME DISK:[HOME]TEST_J2.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J2_BCK.AIJ
    JOURNAL IS ENABLED
      (FAST COMMIT ENABLED);
%RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
  EXIT;
$
$ ! Back up the original database configuration.
$
$ RMU/BACKUP/NOLOG DISK:[HOME]TEST DISK:[HOME]TEST
$
$ ! Add new journals, row caches and storage areas
$
$ SQL
  ALTER DATABASE FILENAME DISK:[HOME]TEST
    ADD JOURNAL TEST_J3 FILENAME DISK:[HOME]TEST_J3.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J3_BCK.AIJ
    ADD JOURNAL TEST_J4 FILENAME DISK:[HOME]TEST_J4.AIJ
    BACKUP FILENAME DISK:[HOME]TEST_J4_BCK.AIJ
    ADD CACHE TEST_A3
        CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
    ADD CACHE TEST_A4
        CACHE SIZE 100 ROWS ROW LENGTH 100 BYTES LOCATION IS 'DISK:[HOME]'
    ADD STORAGE AREA TEST_A3 FILENAME DISK:[HOME]TEST_A3
    ADD STORAGE AREA TEST_A4 FILENAME DISK:[HOME]TEST_A4;
%RDMS-W-FULBCKREQ, The next database backup must be a full backup
  EXIT;
$
$ ! Backup TEST_J1.AIJ, where the creation of the new journals, row caches
$ ! and storage areas in DISK:[HOME] is recorded.
$
$ RMU/BACKUP/AFTER/NOLOG/NOQUIET DISK:[HOME]TEST DISK:[HOME]TEST_J1_BCK.AIJ
$
$ ! Delete the database.
$
$ SQL
  DROP DATABASE FILENAME DISK:[HOME]TEST;
  EXIT;
$
$ ! Restore the database with all storage areas, after image journals
$ ! and row caches in DISK:[HOME]
$
$ RMU/RESTORE/NOCDD/NOLOG/NORECOVER/DIR=DISK:[HOME] DISK:[HOME]TEST
%RMU-I-AIJRSTAVL, 2 after-image journals available for use
%RMU-I-AIJRSTMOD, 1 after-image journal marked as "modified"
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$
$ ! Recover the database and change the directories from DISK:[HOME] to
```

1.41 RMU Recover Command

```
$ ! DISK:[NEW] for the new storage areas, after image journals and row
$ ! caches whose creation was recorded in the backed up journal file
$ ! TEST_J1_BCK.AIJ
$
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
 /DIRECTORY=(AREAS=DISK:[NEW], -
 AFTER_JOURNAL=DISK:[NEW], -
 ROW_CACHE=DISK:[NEW]) -
 DISK:[HOME]TEST_J1_BCK.AIJ
%RMU-I-LOGRECDB, recovering database file DISK:[HOME]TEST_J1_BCK.AIJ;1
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
needed will be 1
$
$ ! Create directory option files for all storage areas,
$ ! after image journals and row caches in DISK:[HOME]
$ ! or DISK:[NEW]
$
$ RMU/DUMP-
 /DIRECTORY_OPTIONS=(AREAS=DISK:[HOME]RECOVOPTAREAS.OPT, -
 AFTER_JOURNAL=DISK:HOME]RECOVOPTAIJ.OPT, -
 ROW_CACHE=DISK:[HOME]RECOVOPTRCACHE.OPT) -
 DISK:[HOME]TEST
$
$ ! Delete the database.
$
$ SQL$
  DROP DATABASE FILENAME DISK:[HOME]TEST;
  EXIT;
$
$ ! Restore the database with all storage areas, after image journals
$ ! and row caches again in DISK:[HOME]
$
$ RMU/RESTORE/NOCDD/NOLOG/NORECOVER/DIR=DISK:[HOME] DISK:[HOME]TEST
%RMU-I-AIJRSTAVL, 2 after-image journals available for use
%RMU-I-AIJRSTMOD, 2 after-image journals marked as "modified"
%RMU-F-AIJENBOVR, enabling AIJ journaling would overwrite an existing journal
%RMU-I-AIJISOFF, after-image journaling has been disabled
$
$ ! Recover the database and again change the directories from DISK:[HOME]
$ ! to DISK:[NEW] for the new storage areas, after image journals and row
$ ! caches whose creation was recorded in the backed up journal file
$ ! TEST_J1_BCK.AIJ
$
$ RMU/RECOVER/NOLOG/NOTRACE/NOCONFIRM/NOAUTOMATIC-
 /OPTIONS=(AREAS=DISK:[HOME]RECOVOPTAREAS.OPT, -
 AFTER_JOURNAL=DISK:[HOME]RECOVOPTAIJ.OPT, -
 ROW_CACHE=DISK:[HOME]RECOVOPTRCACHE.OPT) -
 TEST$SCRATCH:TEST_J1_BCK.AIJ
  !  Recover Areas Directory Options file for database
```

```
!  DISK:[HOME]TEST.RDB;1
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

RDB$SYSTEM -
        /directory=DISK:[HOME] -
        /snapshot_directory=DISK:[HOME]

TEST_A1 -
        /directory=DISK:[HOME] -
        /snapshot_directory=DISK:[HOME]

TEST_A2 -
        /directory=DISK:[HOME] -
        /snapshot_directory=DISK:[HOME]

TEST_A3 -
        /directory=DISK:[NEW] -
        /snapshot_directory=DISK:[NEW]

TEST_A4 -
        /directory=DISK:[NEW] -
        /snapshot_directory=DISK:[NEW]
!  Recover After Journal Directory Options file for database
!  DISK:[HOME]TEST.RDB;1
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

TEST_J1 -
        /directory=DISK:[HOME] -
        /backup_directory=DISK:[HOME]

TEST_J2 -
        /directory=DISK:[HOME] -
        /backup_directory=DISK:[HOME]

TEST_J3 -
        /directory=DISK:[NEW] -
        /backup_directory=DISK:[NEW]

TEST_J4 -
        /directory=DISK:[NEW] -
        /backup_directory=DISK:[NEW]
!  Recover Row Cache Directory Options file for database
!  DISK:[HOME]TEST.RDB;1
!  Created 22-JUL-2014 09:37:24.29
!  Created by DUMP command

TEST_A1 -
        /directory=DISK:[HOME]
```

1.41 RMU Recover Command

```
  TEST_A2 -
          /directory=DISK:[HOME]

  TEST_A3 -
          /directory=DISK:[NEW]

  TEST_A4 -
          /directory=DISK:[NEW]
%RMU-I-LOGRECDB, recovering database file DISK:[HOME]TEST.RDB;1
%RMU-I-AEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence number
needed will be 1
%RMU-I-AIJNOENABLD, after-image journaling has not yet been enabled
```

1.42 RMU Recover Resolve Command

Recovers a corrupted database and resolves an unresolved distributed transaction by completing the transaction.

See the *Oracle Rdb7 Guide to Distributed Transactions* for complete information on unresolved transactions and for information on the DECdtm transaction manager.

Format

RMU/Recover/Resolve aij-file-name

Command Qualifiers	Defaults
/Active_IO=max-reads	See the RMU/Recover command
/Aij_Buffers=integer	See the RMU/Recover command
/Areas[=storage-area[,...]]	See the RMU/Recover command
/[No]Confirm	See description
/Format={Old_File|New_Tape}	See the RMU/Recover command
/Label=(label-name-list)	See the RMU/Recover command
/[No]Log	See the RMU/Recover command
/[No]Media_Loader	See the RMU/Recover command
/[No]Online	See the RMU/Recover command
/[No]Rewind	See the RMU/Recover command
/Root=root-file-name	See the RMU/Recover command
/State=option	See description
/[No]Trace	See the RMU/Recover command
/Until=date-time	See the RMU/Recover command

Description

Use the RMU Recover Resolve command to commit or abort any unresolved distributed transactions in the after-image journal (.aij) file. You must complete the unresolved transactions to the same state (COMMIT or ABORT) in every .aij file affected by the unresolved transactions.

The RMU Recover Resolve command performs the following tasks:

- Displays identification information for an unresolved transaction.
- Prompts you for the state to which you want the unresolved transaction resolved (if you did not specify the State qualifier on the command line). If you are using DECdtm to manage the transaction, you can specify COMMIT, ABORT, or IGNORE.
- Prompts for confirmation of the state you specified

- Commits, aborts, or ignores the unresolved transaction
- Continues until it displays information for all unresolved transactions

Command Parameters

aij-file-name
The name of the file containing the after-image journal. This cannot be an optimized after-image journal (.oaij) file. The default file extension is .aij.

Command Qualifiers

Confirm
Noconfirm
Prompts you for confirmation of each transaction state you alter. The default for interactive processing is Confirm.

Specify the Noconfirm qualifier to suppress this prompt. The default for batch processing is Noconfirm.

State=option
Specifies the state to which all unresolved transactions will be resolved.

If you are using DECdtm to manage your distributed transaction, options for the State qualifier are:

- Commit—Commits all unresolved transactions.
- Abort— Aborts all unresolved transactions.
- Ignore—Does not resolve any transactions.

If you do not specify the State qualifier, Oracle RMU prompts you to enter an action, for *each* unresolved transaction in that .aij file. If DECdtm is managing your transaction and you enter Ignore, Oracle RMU—instead of resolving the transaction—attempts to contact the coordinator to resolve the transaction. The transaction remains unresolved until the coordinator becomes available again and instructs the transaction to complete or until you manually complete the transaction by using the RMU Recover Resolve command again. For more information about the activities of the coordinator, see the *Oracle Rdb7 Guide to Distributed Transactions*.

Usage Notes

- To use the RMU Recover Resolve command for a database, you must have the RMU$RESTORE privilege in the root file for the database or the OpenVMS SYSPRV or BYPASS privilege.
- If you have restored the database by using the New qualifier and have not deleted the corrupted database, use the Root qualifier to override the original file specification for the database root file.
- After it rolls forward from the .aij file specified on the command line, Oracle RMU prompts you for the name of the next .aij file. If there are more .aij files to roll forward, enter the file name, including the version number for that .aij file. If there are no other .aij files, press the Return key. For more information about rolling forward and determining transaction sequence numbers for .aij files, see the *Oracle Rdb Guide to Database Maintenance*.

Examples

Example 1

The following command recovers the mf_personnel database and rolls the database forward from the old .aij file to resolve the unresolved distributed transactions. Because the State qualifier is not specified, Oracle RMU will prompt the user for a state for each unresolved transaction.

```
$ RMU RECOVER/RESOLVE MF_PERSONNEL.AIJ;1
```

Example 2

This example specifies that all unresolved transactions in the mf_personnel.aij file be committed.

```
$ RMU/RECOVER/RESOLVE/STATE=COMMIT MF_PERSONNEL.AIJ
```

For more examples of the RMU Recover Resolve command, see the *Oracle Rdb7 Guide to Distributed Transactions*.

1.43 RMU Repair Command

Corrects several types of database problems. You can use the RMU Repair command to:

- Repair all types of space area management (SPAM) page corruptions by reconstructing the SPAM pages in one or more storage areas.
- Repair all area bit map (ABM) page format errors.
- Repair all page tail errors to the satisfaction of the RMU Verify operation by making sure that every database page is in a logical area and contains the appropriate information for that logical area.
- Correct some performance problems that might otherwise have to be corrected by exporting and importing the database.
- Set damaged or missing segmented string (LIST OF BYTE VARYING) areas that are stored in write-once areas to null.

The repair operation cannot correct corrupted user data, or corrupted indexes; use other commands such as the RMU Restore, the RMU Recover, the SQL IMPORT, or the RMU Load command and delete the affected structures to correct these problems.

Note

Use of the Abm or the Initialize=Tsns qualifier disables after-image journaling. After issuing an RMU Repair command with these qualifiers, back up the database and reenable journaling manually.

Format

RMU/Repair root-file-spec

Command Qualifiers	Defaults
/[No]Abm	/Noabm
/[No]All_Segments	All segments
/Areas [={storage-area-list or *}]	See description
/Checksum	See description
/[No]Initialize=initialize-options	/Noinitialize
/[No]Spams	See description
/Tables [=table-list]	All nonsystem tables

1.43 RMU Repair Command

Description

Because RMU Repair cannot correct every type of corruption, or guarantee improved performance, Oracle Corporation recommends that you do not use the RMU Repair command unless you have a backup copy or exported copy of your database. You can return to this backup copy of the database if your repair efforts are ineffective.

The RMU Repair command operates off line and not in the context of a transaction, so no records are written to the database's .aij file by RMU Repair, and the repaired database cannot be rolled forward with the RMU Recover command. Oracle Corporation recommends that you make a backup copy of the database after using the RMU Repair command; the repair operation issues a message to this effect. Oracle RMU also issues a warning when you use this command on a database with after-image journaling enabled.

Command Parameters

root-file-spec
A file specification for the database root file for which you want to repair corruption or improve performance.

Command Qualifiers

Abm
Noabm
Causes the reconstruction of the logical area bit map (ABM) pages for areas specified with the Areas qualifier. After-image journaling is disabled when you specify the Abm qualifier. You must explicitly enable after-image journaling after the RMU Repair command completes if you want journaling enabled.

The NoAbm qualifier specifies that ABM pages are not to be reconstructed; this is the default.

All_Segments
Noall_Segments
The All_Segments qualifier specifies that RMU Repair should retrieve all segments of a segmented string; the Noall_Segments qualifier specifies that RMU Repair should only retrieve the first segment of a segmented string.

Specify the Noall_Segments qualifier if you know that the list storage map for any segmented strings stored on the specified areas might have contained multiple areas. For example, if the storage map was created using the following SQL command, Oracle Rdb would store all the segmented strings on AREA1 until AREA1 became full. If AREA1 became full, Oracle Rdb would

continue to write the rest of the segments into AREA2. Suppose AREA2 becomes corrupt. In this case, retrieving the first segment from AREA1 is not sufficient; all segments must be retrieved to determine if part of the segmented string is missing.

```
CREATE STORAGE MAP FOR LIST STORE IN (AREA1, AREA2) FOR (TABLE1)
   IN RDB$SYSTEM;
```

Specifying the Areas qualifier and the All_Segments qualifier is unnecessary and redundant because specifying the All_Segments qualifier causes RMU Repair to check all storage areas regardless of where the segmented string was stored initially.

Areas[={storage-area-list or *}]
Specifies the storage areas in the database you want to repair. You can specify storage areas by name or by the area's ID number.

By default, all the storage areas in the database are repaired. If you specify more than one storage area, separate the storage area names or ID numbers in the storage-area-list with a comma, and enclose the list within parentheses.

Checksum
Reads every page in the database storage areas to verify that the checksum on each page is correct. If the checksum on the page is incorrect, it is replaced with the correct checksum.

Use the Areas qualifier to specify which storage areas RMU Repair should check. If you do not specify the Areas qualifier, all pages in all storage areas are checked and updated (if incorrect).

This qualifier can be used whether or not users are attached to the database.

This qualifier is not valid for use with any other qualifiers except the Areas qualifier.

Initialize=initialize-options
Noinitialize
Allows you to specify initialization options. If more than one option is specified, separate the options with a comma, and enclose the list of options within parentheses.

The following options are available for the Initialize qualifier:

- Free_Pages

The Initialize=Free_Pages qualifier initializes database pages that do not contain data in the selected storage areas (that have a uniform page format). You can use the Initialize=Free_Pages qualifier to correct BADPTLARE errors found by the RMU Verify command and also to free pages from a table that has many deleted rows. If you specify the default, the Noinitialize qualifier, no database pages are initialized.

Frequently, you will receive one or more RMU-W-ABMBITTERR error messages after you issue the RMU Repair command with the Initialize=Free_Pages qualifier. This occurs because the initialization of pages can create new ABM errors. Correct these errors by issuing the RMU Repair command with the Abm qualifier. (However, note that you cannot specify the Initialize=Free_Pages qualifier and the Abm qualifier on the same command line.) If you ignore the RMU-W-ABMBITTERR error messages, extra I/O operations will be performed (one for each RMU-W-ABMBITTERR error you received) when a database query causes a sequential scan of an entire table.

If a table residing in a storage area that has a uniform page format is frequently accessed sequentially, the cost of the sequential access is determined by the number of allocated pages. If the maximum size allocated for the table is much larger than the table's average size, the cost of the sequential access can be excessive. By using the RMU Repair command with the Initialize=Free_Pages qualifier, you can purge the allocated but unused database pages from the table. In some cases, there may be a decrease in performance when you insert new data into the table after using this option. As with all Repair options, you should test the performance of the database after executing the command and be prepared to restore the backup made before executing the Repair command if you find that the command results in decreased performance.

The initialization of free pages requires access to the Oracle Rdb system tables. You should not initialize free pages until you know that the RDB$SYSTEM storage area (where the system tables are stored) is not corrupted.

- Larea_Parameters=options-file

  This option specifies an options file (default file extension .opt) that contains a list of logical areas and parameter values that RMU Repair uses to update the area inventory page (AIP) before it builds the space area management (SPAM) pages.

  The Larea_Parameters options file contains lines in the following format:

  ```
  name [/Areas=name][/Delete][/[No]Thresholds=(n[,n[,n]])[/Length=n][/Type=option]
  ```

A comment can be appended to the line (an exclamation point (!) is the comment character), and a line can be continued (as in DCL) by ending it with a hyphen (-).

The logical area can be specified by name or identification number (ID). The logical area named must be present in the AIP, or an error is generated.

The Larea_Parameters options are further described as follows:

- Areas=name

  Restricts this line to the logical area that resides in the specified storage area. The storage area can be specified by name or ID. By default, all logical areas with a matching name are altered independently of the storage area in which they reside.

  You can specify storage area ID numbers with the Areas qualifier.

- Delete

  Specifies that the logical area should be marked as deleted.

CAUTION

You will corrupt your database if you delete a logical area that is referenced by Oracle Rdb metadata.

- Length=n

  The Initialize=Length option specifies the record length to store in the logical area inventory entry. RMU Repair uses this value to calculate SPAM thresholds.

  When columns are deleted from or added to a table, the record length stored in the logical area inventory entry is not updated. Therefore the search for space needed to store a new record may be inefficient, and the SPAM thresholds will not be set properly. You can solve this problem by first correcting the length in the logical area inventory entry, then generating corrected SPAM pages using the RMU Repair command. See Example 2 in the Examples section.

- Thresholds=(n [,n [,n]])
  NoThresholds

  This option specifies the logical area SPAM thresholds. This is useful only for logical areas that reside in a storage area with a uniform page format. If thresholds are set, they are ignored in a storage area with a mixed page format.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for information on setting SPAM thresholds.

The Nothresholds option specifies that logical area thresholds be disabled.

– Type=keyword

By specifying a Type, you can update the on-disk logical area type in the AIP. For databases created prior to Oracle Rdb release 7.0.1, the logical area type information in the AIP is unknown. However, the RMU Show Statistics utility depends on this information to display information on a per-logical-area basis. A logical area is a table, B-tree index, hash index, or any partition of one of these.

In order to update the on-disk logical area type in the AIP, specify the type as follows:

Type=Table
Specifies that the logical area is a data table, such as is created with the SQL CREATE TABLE statement.

Type=Btree
Specifies that the logical area is a B-tree index, such as is created with the SQL CREATE INDEX TYPE IS SORTED statement.

Type=Hash
Specifies that the logical area is a hash index, such as is created with the SQL CREATE INDEX TYPE IS HASHED statement.

Type=System
Specifies that the logical area is a system record that is used to identify hash buckets. Users cannot explicitly create this type of logical area. This type should not be used for the RDB$SYSTEM logical areas. It does not identify system relations.

Type=Blob
Specifies that the logical area is a BLOB (LIST OF BYTE VARYING) repository.

There is no error checking of the type specified for a logical area. The specified type does not affect the collection of statistics, nor does it affect the readying of the affected logical areas. However, an incorrect type will cause incorrect statistics to be reported by the RMU Show Statistics utility.

• Only_Larea_Type

The Initialize=Only_Larea_Type option specifies that only the logical area type field is to be updated in the area inventory page (AIP).

1.43 RMU Repair Command

- Snapshots

  The Snapshots option allows you to create and initialize new snapshot files. In addition, it removes corrupt snapshot area pages from the Corrupt Page Table (CPT). This is much faster than using the RMU Restore command to do the same thing, especially when just one snapshot file is lost and needs to be created again. The default is not to create new files.

  When you specify the Confirm option with the Initialize=Snapshots option (Initialize=Snapshots=Confirm), you can use the RMU Repair command not only to initialize, but also to optionally rename, move, or change the allocation of snapshot files.

  These operations might be necessary when a disk with a snapshot file has a hardware problem or is removed in a hardware upgrade, or when a snapshot file has grown too large and you want to truncate it.

  The Confirm option causes RMU Repair to prompt you for a name and allocation for one or more snapshot files. If you use the Areas qualifier, you can select the snapshot files in the database that you want to modify. If you omit the Areas qualifier, all the snapshot files for the database are initialized and RMU Repair prompts you interactively for an alternative file name and allocation for each snapshot file. By specifying a new file name for a snapshot file, you can change the location of the snapshot file. By specifying a new allocation for a snapshot file, you can truncate a snapshot file or make it larger.

- Snapshots=Options=file_specification

  This additional syntax for RMU/REPAIR allows the user to specify an options file (default file extension .opt) that can be used to initialize snapshot files. This allows the user to avoid prompts for each snapshot file to be initialized and makes it more convenient to initialize a large number of options files in one RMU/REPAIR command.

  This optional syntax for RMU/REPAIR is as follows:

  ```
  /INITIALIZE=SNAPSHOTS=OPTIONS=file_specification
  ```

  The "file_specification" must be the valid file specification of an existing options file. This qualifier cannot be negated. The following syntax must be used in the specified options file.

  ```
  $ type filename.opt
  storage_area [/SNAPSHOT=([FILE=file_specification][,][ALLOCATION=n])]
  ```

In the above, "storage_area" specifies the name of the storage area which uses the snapshot file. The "RMU/DUMP/HEADER" command can be used to display the storage area names contained in the database root file - see 'Snapshot area for storage area "storage_area"'. "FILE=file_specification" specifies an optional new file specification for the snapshot file. The "RMU/DUMP/HEADER" command can be used to display the snapshot file specifications contained in the database root file - see 'Filename is "device:[directory]name.SNP;1"'. The snapshot file specification cannot be changed for a single file database. "ALLOCATION=n" specifies an optional new page count allocation size for the snapshot file. By specifying a new allocation for a snapshot file, you can truncate a snapshot file or make it larger. The "RMU/DUMP/HEADER" command can be used to display the allocation size for the snapshot file - see "Current physical page count is n". Note that if the "/AREAS" qualifier is used in the same RMU/REPAIR command with "/INITIALIZE=SNAPSHOTS=OPTIONS=file_specification" it will be in effect for other RMU/REPAIR options but will be ignored for the "/INITIALIZE=SNAPSHOTS" option since the options file specifies the storage areas that use the snapshot files to be initialized.

In the following example, "SET VERIFY" is specified to display the contents of the JUST_AREA.OPT options file which specifies that the snapshot files for the DEPARTMENTS, JOBS and SALARY_HISTORY storage areas in the MF_PERSONNEL database are to be initialized. Since the JOBS.SNP snapshot file has been deleted, it will be created before it is initialized. The current snapshot file specifications and allocations are used. An RMU/VERIFY of MF_PERSONNEL is done after the repair to verify the integrity of the database.

```
$ SET VERIFY
$ DELETE JOBS.SNP;*
$ DIRECTORY JOBS.SNP
%DIRECT-W-NOFILES, no files found
$!
$ RMU/REPAIR/INITIALIZE=SNAPSHOTS=OPTIONS=JUST_AREA.OPT MF_PERSONNEL
%RMU-I-FULBACREQ, A full backup of this database should be
 performed after RMU REPAIR
  departments
  jobs
  salary_history
$!
$ DIRECTORY JOBS.SNP

Directory DEVICE:[DIRECTORY]

JOBS.SNP;1

Total of 1 file.
```

```
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
$
```

In the following example, "SET VERIFY" is specified to display the contents of the AREA_SNAP.OPT options file which specifies that the snapshot files for the DEPARTMENTS, JOBS and SALARY_HISTORY storage areas in the MF_PERSONNEL database are to be initialized with a new file specification and page count allocation. An RMU/VERIFY of MF_PERSONNEL is done after the repair to verify the integrity of the database.

```
$ SET VERIFY
$ RMU/REPAIR/INITIALIZE=SNAPSHOTS=OPTIONS=AREA_SNAP.OPT MF_PERSONNEL
%RMU-I-FULBACREQ, A full backup of this database should be
 performed after RMU REPAIR
  departments /snapshot=(file=new_dept,allocation=200)
  jobs /snapshot=(file=new_jobs,allocation=210)
  salary_history /snapshot=(file=new_sal_hist,allocation=220)
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
$
```

- Tsns

  The Initialize=Tsns option resets the database transaction state. The default is to not alter the transaction state.

  After-image journaling is disabled when you specify the Initialize=Tsns option. You must explicitly enable after-image journaling after the RMU Repair command completes if you want journaling enabled.

  This operation is useful when the database transaction sequence number (TSN) approaches the maximum allowable value and the TSN values must be initialized to zero. The TSN value is contained in a quadword. The high longword can hold a maximum user value of 32768 (2^{15}) and the low longword can hold a maximum user value of 4,294,967,295 (2^{32}). A portion of the high-longword is used by Oracle Rdb for overhead.

  Initialization of the TSN values requires reading and writing to each page of the database, so the Areas qualifier is not meaningful. It also requires initialization of the snapshot areas even if the Snapshots option has not been specified.

  The Tsns initialization option carries the following restrictions:

  – It cannot be performed if the Replication Option for Rdb is being used unless all transfers have been completed. RMU Repair will ask for confirmation if an RDB$TRANSFERS table is defined.

– Old journal files will not be applicable to this repaired database. After TSNs have been initialized, you must reenable after-image journaling if you want journaling enabled.

  After the RMU Repair command completes, a full and complete backup operation should be performed on the database as soon as is practical. This operation ensures that new journaled changes can be applied to the restored database in the event that a restore operation should become necessary.

Spams
Nospams

Reconstructs the SPAM pages for the areas you specify with the Areas qualifier. If you specify the Nospams qualifier, the SPAM pages are not reconstructed. The default is the Spam qualifier if you do not specify any of the following qualifiers for the RMU Repair command:

- ABM
- Initialize=Free_Pages
- Initialize=Snapshots
- Initialize=Snapshots=Confirm
- Initialize=Snapshots=Options

If you use any of these qualifiers, the NoSpam qualifier is the default.

When columns are deleted from or added to a table, the record length stored in the logical area inventory entry is not updated. Therefore the search for space needed to store a new record may be inefficient, and the SPAM thresholds will not be set properly. You can solve this problem by first correcting the length in the logical area inventory entry, then generating corrected SPAM pages using the RMU Repair command. See Example 2 in the Examples section.

Tables[=table-list]

Specifies the list of tables that you want RMU Repair to check for complete segmented strings.

If no tables are listed, then all nonsystem tables are examined. (System tables do not store their segmented strings in write-once areas.) Note that RMU Repair has no knowledge of which storage areas contain segmented strings from a particular table; thus, the default is to search all tables.

1.43 RMU Repair Command

Usage Notes

- To use the RMU Repair command for a database, you must have the RMU$ALTER privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- Enable detected asynchronous prefetch to achieve the best performance of this command. You can determine the setting for your database by issuing the RMU Dump command with the Header qualifier.

  If detected asynchronous prefetch is disabled, and you do not want to enable it for the database, you can enable it for your RMU Repair operations by defining the following logicals at the process level:

  ```
  $ DEFINE RDM$BIND_DAPF_ENABLED 1
  $ DEFINE RDM$BIND_DAPF_DEPTH_BUF_CNT P1
  ```

  P1 is a value between 10 and 20 percent of the user buffer count.
- The Areas qualifier can be used with indirect file references. See Section 1.3.
- Oracle Corporation recommends that you use the RMU Backup command to perform a full backup operation on your database before using the RMU Repair command on the database.
- Use the SQL SHOW STORAGE AREA statement to display the new location of a snapshot (.snp) file and the RMU Dump command with the Header qualifier to display the new allocation.
- Be careful when you specify names for new .snp files with the RMU Repair command. If you specify the name of a file that already exists and was created for the database, it will be initialized as you requested.

  If you mistakenly initialize a live database file in this way, do not use the database until the error is corrected. Use the RMU Restore command to restore the database to the condition it was in when you backed it up just prior to issuing the RMU Repair command. If you did not back up the database before issuing the RMU Repair command, you must restore the database from your most recent backup file and then recover from .aij files (if the database had after-image journaling enabled).

  If you specify the wrong .snp file (for example, if you specify jobs.snp for all the .snp file name requests in Example 3 in the Examples section), you can correct this by issuing the RMU Repair command again with the correct .snp file names.

After the RMU Repair command completes, delete old .snp files and use the RMU Backup command to perform a full backup operation on your database.

Examples

Example 1

The following command repairs SPAM page corruption for all the storage areas in the mf_personnel database. No area bit map (ABM) pages are reconstructed because the Abm qualifier is not specified.

```
$ RMU/REPAIR MF_PERSONNEL
```

Example 2

When columns are deleted from or added to a table, the record length stored in the logical area inventory entry is not updated. Therefore the search for space needed to store a new record may be inefficient, and the SPAM thresholds will not be set properly. You can solve this problem by first correcting the length in the logical area inventory entry, then generating corrected SPAM pages using the RMU Repair command.

For example, suppose the Departments table was stored in the departments.rda uniform page format storage area and the Budget_Projected column (integer data type = 4 bytes) was deleted. As a result of this deletion, the row length changed from 47 bytes to 43 bytes. You can specify a smaller record length (43 bytes) in the fix_departments.opt options file to more efficiently use space in the storage area.

```
$ CREATE FIX_DEPARTMENTS.OPT
DEPARTMENTS /LENGTH=43
```

Then, the following RMU Repair command specifies the record length to store in the logical area inventory entry for this logical area and rebuilds the SPAM pages:

```
$ RMU/REPAIR/SPAMS/INITIALIZE=LAREA_PARAMETERS=FIX_DEPARTMENTS.OPT -
_$ MF_PERSONNEL
```

Example 3

The following RMU Repair command initializes and renames departments.snp; initializes and moves salary_history.snp; and initializes, moves, and truncates jobs.snp:

1.43 RMU Repair Command

```
$ RMU/REPAIR/NOSPAMS/INITIALIZE=SNAPSHOTS=CONFIRM -
_$ /AREAS=(DEPARTMENTS,JOBS,SALARY_HISTORY) MF_PERSONNEL
%RMU-I-FULBACREQ, A full backup of this database should be
 performed after RMU Repair
Area DEPARTMENTS snapshot filename
 [SQL1:[TEST]DEPARTMENTS.SNP;1]: NEW_DEPT
Area DEPARTMENTS snapshot file allocation [10]?
Area SALARY_HISTORY snapshot filename
 [SQL1:[TEST]SALARY_HISTORY.SNP;1]: SQL2:[TEST]
Area SALARY_HISTORY snapshot file allocation [10]?
Area JOBS snapshot filename [SQL1:[TEST]JOBS.SNP;1]: SQL2:[TEST2]
Area JOBS snapshot file allocation [10]? 5
```

Example 4

The following RMU Repair command finds incorrect checksums in the EMPIDS_LOW storage area and updates them to reflect the correct checksum:

```
$ RMU/REPAIR MF_PERSONNEL.RDB/AREA=EMPIDS_LOW/CHECKSUM
```

Example 5

The following command updates an AIP type for a table:

```
$ RMU/REPAIR MF_PERSONNEL /INITIALIZE=LAREA_PARAMETERS=TABLE.OPT
```

Type the TABLE.OPT file to show the contents of the file.

```
$ TYPE TABLE.OPT
EMPLOYEES /TYPE=TABLE
```

Example 6

The following command updates an AIP type for a storage area:

```
$ RMU/REPAIR MF_PERSONNEL /INITIALIZE=LAREA_PARAMETERS=AREAS.OPT
```

Type the AREAS.OPT file to show the contents of the file.

```
$ TYPE AREAS.OPT
EMPLOYEES /AREA=EMPIDS_OVER /TYPE=TABLE
```

1.44 RMU Resolve Command

Resolves all unresolved distributed transactions for the specified database. For more information on unresolved transactions, see the *Oracle Rdb7 Guide to Distributed Transactions* and the *Oracle Rdb Release Notes*.

Format

RMU/Resolve root-file-spec

Command Qualifiers	Defaults
/[No]Confirm	See description
/[No]Log	Setting of DCL VERIFY flag
/Parent_Node=node-nam e	See description
/Process=process-id	See description
/State=options	None
/Tsn=tsn	See description

Description

Use the RMU Resolve command to commit or abort any unresolved distributed transactions in the database. You must resolve the unresolved transactions to the same state (Commit or Abort) in every database affected by the unresolved transactions.

RMU Resolve performs the following tasks:

- Displays identification information for an unresolved transaction.
- Prompts you for the state (Commit or Abort) to which you want the unresolved transaction resolved (if you did not specify the State qualifier on the command line).
- Prompts you for confirmation of the state you chose.
- Commits or aborts the unresolved transaction. If you commit or abort the unresolved transaction, it is resolved and cannot be resolved again.
- Continues to display and prompt for states for subsequent unresolved transactions until it has displayed information for all unresolved transactions.

1.44 RMU Resolve Command

Use the Parent_Node, Process, or Tsn qualifiers to limit the number of unresolved transactions that Oracle RMU displays. Use the Users and State=Blocked qualifiers with the RMU Dump command to determine values for the Parent_Node, Process, and Tsn qualifiers.

Command Parameters

root-file-spec
The database root file for which you want to resolve unresolved transactions.

Command Qualifiers

Confirm
Noconfirm
Prompts you for confirmation of each unresolved transaction. This is the default for interactive processing.

Specify the Noconfirm qualifier to suppress this prompt. This is the default for batch processing.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request that summary information about the resolve operation be reported to SYS$OUTPUT and the Nolog qualifier to prevent this reporting. If you specify neither, the default is the current setting of the DCL VERIFY flag. (The DCL SET VERIFY command controls the setting of the DCL VERIFY flag.)

Parent_Node=node-name
Specifies the node name to limit the selection of transactions to those originating from the specified node. If you omit the Parent_Node qualifier, RMU Resolve includes transactions originating from all nodes.

You cannot specify the Tsn or Process qualifier with the Parent_Node qualifier.

Process=process-id
Specifies the process identification to limit the selection of transactions to those associated with the specified process. If you omit this qualifier, RMU Resolve includes all processes with transactions attached to the specified database.

You cannot specify the Parent_Node or Tsn qualifier with the Process qualifier.

State=options
Specifies the state to which all unresolved transactions be resolved.

Options for the State qualifier are:

- Commit—Commits unresolved transactions.
- Abort—Aborts unresolved transactions.

If you do not specify the State qualifier, RMU Resolve prompts you to enter an action, Commit or Abort, for each unresolved transaction on that database.

Tsn=tsn
Specifies the transaction sequence number (TSN) of the unresolved transactions whose state you want to modify.

The TSN value is contained in a quadword with the following decimal format:

```
high longword : low longword
```

The high longword can hold a maximum user value of 32768 (2^{15}) and the low longword can hold a maximum user value of 4,294,967,295 (2^{32}). A portion of the high longword is used by Oracle Rdb for overhead.

When you specify a TSN, you can omit the high longword and the colon if the TSN fits in the low longword. For example 0:444 and 444 are both valid TSN input values.

If you omit the Tsn qualifier, RMU Resolve includes all the unresolved transactions. You cannot specify the Parent_Node or the Process qualifier with the Tsn qualifier.

Usage Notes

- To use the RMU Resolve command for a database, you must have the RMU$RESTORE privilege in the root file ACL for the database or the OpenVMS SYSPRV or BYPASS privilege.

Examples

Example 1

The following command specifies that the first displayed unresolved transaction in the MF_PERSONNEL database be changed to the Abort state and rolled back:

```
$ RMU/RESOLVE/LOG/STATE=ABORT MF_PERSONNEL
```

Example 2

1.44 RMU Resolve Command

The following command will display a list of all transactions coordinated by node GREEN and might be useful if node GREEN failed while running an application that used the DECdtm two-phase commit protocol:

```
$ RMU/RESOLVE/PARENT_NODE=GREEN MF_PERSONNEL
```

Example 3

The following command displays a list of all transactions initiated by process 41E0364A. The list might be useful for resolving transactions initiated by this process if the process were deleted.

```
$ RMU/RESOLVE/PROCESS=41E0364A MF_PERSONNEL
```

Example 4

The following command completes unresolved transactions for the MF_ PERSONNEL database, and confirms and logs the operation:

```
$ RMU/RESOLVE/LOG/CONFIRM MF_PERSONNEL
```

For more examples of the RMU Resolve command, see the *Oracle Rdb7 Guide to Distributed Transactions*.

1.45 RMU Restore Command

Restores a database to the condition it was in at the time a full or incremental backup operation was performed with an RMU Backup command. In addition, if after-image journal (.aij) files have been retained, RMU Restore attempts to apply any pre-existing .aij files to recover the database completely. See the Description section for details on the conditions under which RMU Restore attempts an automatic .aij file recovery as part of the restore operation.

When you use the RMU Restore command to restore the database to a system with a more recent version of Oracle Rdb software, an RMU Convert command with the Noconfirm and Commit qualifiers is automatically executed as part of RMU Restore. Therefore, by executing the RMU Restore command, you convert that database to the current version. See the *Oracle Rdb Installation and Configuration Guide* for the proper backup procedure prior to installing a new release of Oracle Rdb and restoring (or converting) databases.

When you use the RMU Restore command to restore a database that was recently RMU/Converted but with the /NoCommit qualifier, the behavior is different than that stated above. /Commit is the default for an RMU Restore of an uncommited database (a database that contains both current and previous versions of the metadata that was converted by specifying RMU/CONVERT/NOCOMMIT or RMU/RESTORE/NOCOMMIT) but ONLY if the noncommited database being restored is NOT of the current Rdb version. RMU/RESTORE/COMMIT and RMU/RESTORE/NOCOMMIT only take effect if RMU/RESTORE needs to call RMU/CONVERT because the database being restored is of a previous Rdb version.

If the /COMMIT is specified or defaulted for the Restore of a database of the current level, it is ignored. In this case, an RMU/CONVERT/COMMIT must be used to commit the previous uncommited restore or conversion.

Note

When you restore a database, default or propagated OpenVMS access control entries (ACEs) for the database root (.rdb) file take precedence over any Oracle RMU database access you might have.

Therefore, if default or propagated entries are in use, you must use the RMU Show Privilege and RMU Set Privilege commands after a restore operation completes to verify and correct the Oracle RMU access. (You can tell if default or propagated entries are in use because RMU Restore displays the warning message "RMU-W-PREVACL, Restoring

the root ACL over a pre-existing ACL". This is a normal condition if the RMU Restore command was invoked from the CDO utility.)

To use RMU Show Privilege and RMU Set Privilege commands, you must have the rights to edit the access control list (ACL) using RMU$SECURITY access (which is VMS BIT_15 access in the access control entry (ACE)) and also (READ+WRITE+CONTROL) access. (Note that you can grant yourself BIT_15 access by using the DCL SET ACL command if you have (READ+WRITE+CONTROL) access.

If you do not have the required access after a restore operation to make the needed changes, someone with the required access or OpenVMS BYPASS or SECURITY access must examine and correct the ACL.

This behavior exists in Oracle RMU to prevent someone from using Oracle RMU to override the existing OpenVMS security policy.

Format

RMU/Restore backup-file-spec [storage-area-name[,...]]

Command Qualifiers	Defaults
/[No]Acl	/Acl
/Active_IO=max-reads	/Active_IO=3
/[No]After_Journal=file-spec	See description
/[No]Aij_Options=journal-opts	See description
/Area	See description
/[No]Assist	See description
/Attributes=options	See description
/[No]Cdd_Integrate	/Cdd_Integrate
/Close_Wait=n	See description
/[No]Commit	/Commit
/[No]Confirm[=Quit|Retry=x|Override|Unload]	See description
/Directory=directory-spec	See description
/Disk_File[=(Reader_Threads=n)]	/Disk_File=(Reader_Threads=1)
/[No]Duplicate	/Noduplicate
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/Global_Buffers=global-buffer-options	Current value
/Incremental	Full restore
/Journal=file-name	See description

/Just_Corrupt	See description
/Label=(label-name-list)	See description
/Librarian[=options]	None
/Loader_Synchronization	See description
/Local_Buffers=local-buffer-options	Current value
/[No]Log[=Brief|Full]	Current DCL verify value
/Master	See description
/[No]Media_Loader	See Description
/[No]New_Version	/Nonew_Version
/Nodes_Max=number-cluster-nodes	See description
/[No]Online	/Noonline
/Open_Mode={Automatic|Manual}	Current value
/Options=file-spec	None
/Output=file-spec	See description
/Page_Buffers=number-buffers	/Page_Buffers=3
/Path=cdd-path	Existing value
/Progress_Report[=n]	See description
/Prompt={Automatic|Operator|Client}	See description
/[No]Recovery[=Aij_Buffers=n]	See description
/[No]Rewind	/Norewind
/Root=root-file-spec	Existing value
/Row_Cache_Options=file-spec	None

Command Qualifiers (cont)	Defaults
/Transaction_Mode=mode-list)	/Transaction_Mode=Current
/Users_max=number-users	Existing value
/Volumes=n	/Volumes=1

File or Area Qualifiers	Defaults
/Blocks_Per_Page=integer	See description
/Extension= {Disable|Enable}	Current value
/File=file-spec	See description
/Just_Corrupt	See description
/Read_Only	Current value
/Read_Write	Current value
/Snapshot=(Allocation=n,File=file-spec)	See description
/[No]Spams	Current value
/Thresholds=(val1[,val2[,val3]])	Current value

Description

RMU Restore rebuilds a database from a backup file, produced earlier by an RMU Backup command, to the condition the database was in when the backup operation was performed and attempts to automatically recover the .aij files to provide a fully restored and recovered database.

1.45 RMU Restore Command

You can specify only one backup file parameter in an RMU Restore command. If this parameter is a full backup file, you cannot use the Incremental qualifier. However, you must use the Incremental qualifier if the parameter names an incremental backup file.

RMU Restore attempts automatic .aij file recovery by default when you issue a database restore command if you are using fixed-size .aij files, if .aij files have been retained, and if a database conversion has not been performed. (The .aij files are not retained when you specify any of the following qualifiers: Aij_Options, After_Journal, or Duplicate.) RMU Restore does not attempt automatic .aij file recovery if you have backed up any of your .aij files (using the RMU Backup After_Journal command) because RMU Restore has no knowledge of those backup files.

In addition, success of the automatic .aij file recovery operation requires that the following criteria be met:

- Fixed-size after-image journaling is in effect.
- The .aij files must be on disk (not on tape).
- The .aij files must not have been marked as inaccessible at the time the database backup operation was performed.
- The .aij files must exist and have proper privileges for both read and write operations.
- The .aij files must be able to be accessed exclusively; failure indicates that an .aij file is in use by another database user.
- The .aij files must have a nonzero length.
- The .aij files must have valid header information that corresponds to the current Oracle Rdb product and version number.
- The sequence number in the .aij file header must not conflict with the restored definition in the database root information.
- The original .rdb file name must not exist.

Note

RMU Restore attempts automatic .aij file recovery when you restore a database from a full, incremental, by-area, or by-page backup file. However, in some cases, you will want to disable this feature by using the Norecovery qualifier. Specifically, you should specify the Norecovery qualifier if either of the following are true:

- You are restoring the database from a previous version of Oracle Rdb.
- You need to issue more than one RMU Restore command to completely restore the database.

  For example, if you intend to restore a database by first issuing a full RMU Restore command followed by the application of one or more RMU Restore commands with the Incremental or Area qualifiers, you *must* specify the Norecovery qualifier on all but the last RMU Restore command in the series you intend to issue. Allowing Oracle RMU to attempt automatic recovery with a full restore operation when you intend to apply additional incremental, by-area, or by-page backup files can result in a corrupt database.

RMU Restore does not attempt automatic .aij file recovery if any of the following conditions are true:

- The database has been converted since the time you created the backup file that you are attempting to restore.
- The first .aij file is not available (perhaps because it has been backed up).
- After-image journaling was disabled when the backup operation was performed.
- After-image journaling was disabled when the database (or portion of it) was lost.
- You specify the Aij_Options, After_Journal, or Duplicate qualifier with the RMU Restore command.

If RMU Restore attempts automatic .aij file recovery but fails, you can still recover your database by using the RMU Recover command if the restore operation was successful.

Note

Using the DCL COPY command with a multifile database (assuming the files are copied to a new location) will result in an unsupported, unusable database. This happens because the DCL COPY command cannot update the full file specification pointers (stored in the database root file) to the other database files (.rda, .snp, and optional .aij).

You can rename or move the files that comprise a multifile Oracle Rdb database by using one of the following commands:

- The RMU Backup and RMU Restore commands

- The SQL EXPORT and IMPORT statements
- The RMU Move_Area command
- The RMU Copy_Database command

By default, RMU Restore integrates the metadata stored in the database root (.rdb) file with the data dictionary copy of the metadata (assuming the data dictionary is installed on your system). However, you can prevent dictionary integration by specifying the Nocdd_Integrate qualifier.

When you specify the Incremental or Area qualifiers, do not specify the following additional qualifiers:

Directory
Nodes_Max
New_Version
Nonew_Version
Users_Max

You must specify the Root qualifier when you restore an incremental backup file to a new version of the database, renamed database, or a restored database in a new location.

See the Usage Notes section for information on restoring a database from tape.

Command Parameters

backup-file-spec
A file specification for the backup file produced by a previous RMU Backup command. Note that you cannot perform a remote restore operation on an .rbf file that has been backed up to tape and then copied to disk.

The default file extension is .rbf.

Depending on whether you are performing a restore operation from magnetic tape, disk, or multiple disks, the backup file specification should be specified as follows:

- To restore from magnetic tape:

  If you used multiple tape drives to create the backup file, the backup-file-spec parameter must be provided with (and only with) the first tape drive name. Additional tape drive names must be separated from the first and subsequent tape drive names with commas, as shown in the following example:

```
$ RMU/RESTORE /REWIND $111$MUA0:PERS_FULL_NOV30.RBF,$112$MUA1:
```

- To restore from single or multiple disk files:

  If you used multiple disk files to create the backup file, the backup-file-spec parameter must be provided with (and only with) the first disk device name. Additional disk device names must be separated from the first and subsequent disk device names with commas. You must also be sure to include the Disk_File qualifier. For example:

  ```
  $ RMU/RESTORE/DISK_FILE DISK1:[DIR1]MFP.RBF,DISK2:[DIR2],DISK3:[DIR3]
  ```

  As an alternative to listing the disk device names on the command line (which, if you use several devices, can exceed the line-limit length for a command line), you can specify an options file in place of the backup-file-spec. For example:

  ```
  $ RMU/RESTORE/DISK_FILE "@DEVICES.OPT"
  ```

  The contents of devices.opt might appear as follows:

  ```
  DISK1:[DIR1]MFP.RBF
  DISK2:[DIR2]
  DISK3:[DIR3]
  ```

  The backup files referenced from such an options file are:

  ```
  DISK1:[DIR1]MFP.RBF
  DISK2:[DIR2]MFP01.RBF
  DISK3:[DIR3]MFP02.RBF
  ```

storage-area-name[,...]

A storage area name from the database. This parameter is optional. Use it in the following situations:

- When you want to change the values for thresholds or blocks per page.
- When you want to change the names specified with the Snapshot or the File qualifier for the restored database.
- If you want to restore only selected storage areas from your backup file, you must use the Area qualifier and specify the names of the storage areas you want to restore in either the storage-area-name parameter in the RMU Restore command line, or in the file specified with the Options qualifier.

  To use this option, specify the storage area *name* rather than the file specification for the storage area.

1.45 RMU Restore Command

By using the RMU Backup and RMU Restore commands, you can back up and restore selected storage areas of your database. This Oracle RMU backup and restore by-area feature is designed to:

- Speed recovery when corruption occurs in some (not all) of the storage areas of your database.
- Reduce the time needed to perform backup operations because some data (data in read-only storage areas, for example) does not need to be backed up with every backup operation performed on the database.

If you plan to use the RMU Backup and RMU Restore commands to back up and restore only selected storage areas for a database, you *must* perform full and complete backup operations on the database at regular intervals. A full and complete backup is a full backup (*not* an incremental backup) operation on *all* the storage areas in the database. If the database root (.rdb) file is corrupted, you can only recover storage areas up to (but not past) the date of the last full and complete backup operation. Therefore, Oracle Corporation recommends that you perform full and complete backup operations regularly.

If you plan to back up and restore only selected storage areas for a database, Oracle Corporation strongly recommends that you enable after-image journaling for the database (in addition to performing the full and complete backup operation on the database as described earlier). That is, if you are not backing up and restoring *all* the storage areas in your database, you should have after-image journaling enabled. This ensures that you can recover all the storage areas in your database in the event of a system failure. If you *do not* have after-image journaling enabled and one or more of the areas restored by RMU Restore are not current with the storage areas not restored, Oracle Rdb will not allow any transactions to use the storage areas that are not current in the restored database. In this situation, you can return to a working database by restoring the database, using the backup file from the last full and complete backup operation on the database storage areas. However, any changes made to the database since the last full and complete backup operation was performed are not recoverable.

If you have after-image journaling enabled, use the RMU Recover command to apply transactions from the .aij file to storage areas that are not current after the RMU Restore command completes. When the RMU Recover command completes, your database will be consistent and usable.

Command Qualifiers

Acl
Noacl
Allows you to specify whether to restore the root file access control list (ACL) that was backed up.

If you specify the Acl qualifier, the root file ACL that was backed up is restored with the database. If the root file ACL was not backed up and you specify the Acl qualifier with the RMU Restore command, then RMU Restore restores the database without a root file ACL.

If you specify the Noacl qualifier, the root file ACL is not restored with the database.

The default is the Acl qualifier.

Active_IO=max-reads
Specifies the maximum number of read operations from the backup file that RMU Restore attempts simultaneously. The value of the Active_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 might improve performance with multiple tape drives.

After_Journal=file-spec
Noafter_Journal

Note

This qualifier is maintained for compatibility with versions of Oracle Rdb prior to Version 6.0. You might find it more useful to specify the Aij_Options qualifier, unless you are interested in creating an extensible .aij file only. (An **extensible .aij file** is one that is extended by a specified amount when it reaches a certain threshold of fullness—assuming there is sufficient space on the disk where it resides.)

Specifies how RMU Restore is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the After_Journal qualifier and provide a file specification, the RMU process creates a new extensible .aij file and enables journaling.
- If you specify the After_Journal qualifier but you do not provide a file specification, RMU Restore creates a new extensible .aij file with the same name as the journal that was active at the time of the backup operation.

- If you specify the Noafter_Journal qualifier, RMU Restore disables after-image journaling and does not create a new .aij file. Note that if you specify the Noafter_Journal qualifier there will be a gap in the sequence of the .aij files. For example, suppose your database has .aij file sequence number 1 when you back it up. If you issue an RMU Restore command with the Noafter_Journal qualifier, the .aij file sequence number will be changed to 2. This means that you cannot (and do not want to) apply the original .aij file to the restored database (doing so would result in a sequence mismatch).
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, RMU Restore recovers the journal state (enabled or disabled) and tries to reuse the .aij file or files. (See the Description section for details on when automatic .aij file recovery is not attempted.)

When you specify an .aij file name, you should specify a new device and directory for the .aij file. If you do not specify a device and directory, you receive a warning message. To protect yourself against media failures, put the .aij file on a different device from that of your database files.

If the original database is lost or corrupted but the journal files are unaffected, you would typically restore the database without the use of either the Aij_Options or the After_Journal qualifier.

The After_Journal qualifier conflicts with the Area and Incremental qualifiers; you cannot specify the After_Journal qualifier and either of these two other qualifiers in the same RMU Restore command line.

You cannot use the After_Journal qualifier to create fixed-size .aij files; use the Aij_Options qualifier.

Aij_Options=journal-opts
Noaij_Options

Specifies how RMU Restore is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the Aij_Options qualifier and provide a journal-opts file, RMU Restore creates the .aij file or files you specify for the restored database. If only one .aij file is created for the restored database, it will be an extensible .aij file. If two or more .aij files are created for the restored database, they will be fixed-size .aij files (as long as at least two .aij files are always available). Depending on what is specified in the options file, after-image journaling can either be disabled or enabled.
- If you specify the Aij_Options qualifier, but do not provide a journal-opts file, RMU Restore disables journaling and does not create any new .aij files.

- If you specify the Noaij_Options qualifier, RMU Restore reuses the original .aij file configuration and recovers the journaling state (enabled or disabled) from the backed-up .aij file.
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, RMU Restore recovers the journaling state (enabled or disabled) and tries to reuse the .aij file or files. (This is the same as specifying the Noaij_Options qualifier.)

  See the Description section for details on when automatic .aij file recovery is not attempted.

The Aij_Options qualifier conflicts with the Area and Incremental qualifiers; you cannot specify the Aij_Options qualifier and either of these two other qualifiers in the same RMU Restore command line.

If the original database is lost or corrupted but the journal files are unaffected, you would typically restore the database without the use of either the Aij_ Options or the After_Journal qualifier.

See Section 1.67.1 for information on the format of a journal-opts-file.

Area

Specifies that only the storage areas listed in the storage-area-name parameter on the command line or in the Options file are to be restored. You can use this qualifier to simplify physical restructuring of a large database.

By default, the Area qualifier is not specified. When the Area qualifier is not specified, all the storage areas and the database root (.rdb) file are restored. Therefore, if you want to restore all the storage areas, omit the Area qualifier. If you specify the Area qualifier, a valid database root must exist. (First issue the RMU Restore Only Root command with a full backup file to create a valid database if one does not exist.)

By using the RMU Backup and RMU Restore commands, you can back up and restore selected storage areas of your database. This Oracle RMU backup- and restore-by-area feature is designed to:

- Speed recovery when corruption occurs in some (not all) of the storage areas of your database.
- Reduce the time needed to perform backup operations because some data (data in read-only storage areas, for example) does not need to be backed up with every backup operation performed on the database.

1.45 RMU Restore Command

Note

When you perform a by-area restore operation, an area may be marked as inconsistent; that is, the area may not be at the same transaction state as the database root when the restore operation completes. This may happen, for example, when automatic aij recovery is disabled with the Norecovery qualifier, or if automatic recovery fails. You can check to see if an area is consistent by using the RMU Show Corrupt_Pages command. If you find that one or more areas are inconsistent, use the RMU Recover command to apply the .aij files. If the .aij files are not available, refer to the section on Clearing an Inconsistent Flag in the *Oracle Rdb Guide to Database Maintenance* for information on the implications of setting a corrupt area to consistent. Then refer to Section 1.57 for information on using the Set Corrupt_Pages command to clear the inconsistent flag.

If you attempt to restore a database area that is not in the backup file, you receive an error message and, typically, the database will be inconsistent or unusable until the affected area is properly restored.

In the following example, the DEPARTMENTS storage area is excluded from the backup operation; therefore, a warning message is displayed when the attempt is made to restore DEPARTMENTS, which is not in the backup file. Note that when this restore operation is attempted on a usable database, it completes, but the DEPARTMENTS storage area is now inconsistent.

```
$ RMU/BACKUP /EXCLUDE=DEPARTMENTS MF_PERSONNEL.RDB -
_$ PERS_BACKUP5JAN88.RBF
$ RMU/RESTORE /NEW_VERSION /AREA PERS_BACKUP5JAN88.RBF DEPARTMENTS
%RMU-W-AREAEXCL, The backup does not contain the storage
 area - DEPARTMENTS
```

If you create a backup file by using the RMU Backup command and the Exclude qualifier, it is your responsibility to ensure that all areas of a database are restored and recovered when you use the RMU Restore and RMU Recover commands to duplicate the database.

The Area qualifier conflicts with the After_Journal and Aij_Options qualifiers.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Attributes=options
Used to specify a number of qualifiers and use an abbreviation for the qualifier name. Normally, /THRESHOLDS, /READ_ONLY and /READ_WRITE cannot be abbreviated or an error will be returned because the abbreviation is not unique for the first four characters (an OpenVMS DCL requirement). By using the /ATTRIBUTES qualifier, THRESHOLDS can be abbreviated to the first four characters and either the option READ_ONLY or the option READ_ WRITE can be abbreviated to the first six characters. Examples of the syntax follow.

```
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_ONLY)
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_WRITE)
/ATTRIBUTES=THRESHOLDS=(50,50,50)
/ATTRIBUTES=READ_ONLY
/ATTRIBUTES=READ_WRITE
```

Examples of using abbreviation with the /ATTRIBUTES qualifier options follow.

Attributes=options
Used to specify a number of qualifiers and use an abbreviation for the qualifier name. Normally, /THRESHOLDS, /READ_ONLY and /READ_WRITE cannot be abbreviated or an error will be returned because the abbreviation is not unique for the first four characters (an OpenVMS DCL requirement). By using the /ATTRIBUTES qualifier, THRESHOLDS can be abbreviated to the first four characters and either the option READ_ONLY or the option READ_ WRITE can be abbreviated to the first six characters. Examples of the syntax follow.

```
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_ONLY)
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_WRITE)
/ATTRIBUTES=THRESHOLDS=(50,50,50)
/ATTRIBUTES=READ_ONLY
/ATTRIBUTES=READ_WRITE
```

Examples of using abbreviation with the /ATTRIBUTES qualifier options follow.

1.45 RMU Restore Command

Cdd_Integrate
Nocdd_Integrate
Integrates the metadata from the database root (.rdb) file into the data dictionary (assuming the data dictionary is installed on your system).

If you specify the Nocdd_Integrate qualifier, no integration occurs during the restore operation.

You might want to delay integration of the database metadata with the data dictionary until after the restore operation finishes successfully.

You can use the Nocdd_Integrate qualifier even if the DICTIONARY IS REQUIRED clause was used when the database was defined.

The Cdd_Integrate qualifier integrates definitions *in one direction only*—from the database file to the dictionary. The Cdd_Integrate qualifier does *not* integrate definitions from the dictionary to the database file.

Close_Wait=n
Specifies a wait time of *n* minutes before RMU Restore automatically closes the database. You must supply a value for *n*.

In order to use this qualifier, the Open_Mode qualifier on the RMU Restore command line must be set to Automatic.

Commit
NoCommit
Instructs Oracle RMU to commit the converted database to the current version of Oracle Rdb before completing the restore operation. Use this qualifier only when the backup file being restored is from a previous version of Oracle Rdb. The conversion is permanent and the database cannot be returned to the previous version. The NoCommit qualifier instructs Oracle RMU not to commit the converted database. In this case, you can rollback the database to its original version using the RMU Convert command with the Rollback qualifier, or you can permanently commit it to the current version by issuing the RMU Convert command with the Commit qualifier. It is important to either Commit or Rollback the conversion after you have verified that the conversion was successful otherwise unnecessary space is taken up in the database to store the obsolete alternate version of the metadata. (RMU will not let you convert to a newer version if the previous Convert was never committed, even if it was years ago.)

The Commit qualifier is the default.

Confirm[=Quit | Retry=x | Override | Unload]
Noconfirm
Specifies that RMU Restore notify you of the name of the database on which you are performing the incremental restore operation. You can thus be sure that you have specified the correct .rdb file name to which the incremental backup file will be applied. This is the default for interactive processing.

Confirmation is especially important on an incremental restore operation if you have changed the .rdb file name or created a new version of the database during a restore operation from the full backup file. (You must specify the Root qualifier also to create new version or change the .rdb file name.)

When doing a Restore from a tape device, the Confirm options can be used. These let you specify to RMU ahead of time what to do if there are problems reading the tape(s) that contain the database backup. If you do not specify one of these options, RMU will give you a prompt and wait for a reply. You can specify only one of these 4 options at a time.

- Confirm=Quit
- Confirm=Retry=x
- Confirm=Override
- Confirm=Unload

Specify the Noconfirm qualifier to have RMU Restore apply the incremental backup file to the database without prompting for confirmation. This is the default for batch processing.

Directory=directory-spec
Specifies the default destination for the restored database files. If you specify a file name or file extension, all restored files are given that file name or file extension. There is no default directory specification for this qualifier. If you do not specify the Directory qualifier, RMU Restore attempts to restore all the database files to the directories they were in at the time the backup file was created; if those directories no longer exist, the restore operation fails.

See the Usage Notes for information on how this qualifier interacts with the Root and File qualifiers and for warnings regarding restoring database files into a directory owned by a resource identifier.

Disk_File[=(Reader_Threads=integer)]
Specifies that you want to perform a multithreaded restore operation from disk files, floppy disks, or other disks external to the PC. This qualifier must have been specified on the RMU Backup command when the backup files from which you are restoring were created.

The Reader_Threads keyword specifies the number of threads that Oracle RMU should use when performing a multithreaded restore operation from disk files. You can specify no more than one reader thread per device specified on the command line (or in the command parameter options file). By default, one reader thread is used.

This qualifier and all qualifiers that control tape operations (Label, Loader_ Synchronization, Master, Media_Loader, and Rewind) are mutually exclusive.

Duplicate
Noduplicate
Specifies a new database with the same content but different identity from that of the original database. The default is the Noduplicate qualifier.

The Duplicate qualifier creates a copy of your database that is not expected to remain in sequence with the original database. Note that you cannot interchange after-image journal (.aij) files between the original and duplicate copy of the database because each database is unique.

You can create a duplicate database when you use the Duplicate qualifier or create the original database again when you use the Noduplicate qualifier.

The Duplicate qualifier conflicts with the Incremental, Area, and Online qualifiers.

Encrypt=({Value=|Name=}[,Algorithm=])
The Encrypt qualifier decrypts the save set file of a database backup.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Global_Buffers=global-buffer-options
Allows you to change the default global buffer parameters when you restore a database. The following options are available:

- Disabled

  Use this option to disable global buffering for the database being restored.

- Enabled

  Use this option to enable global buffering for the database being restored. You cannot specify both the Global_Buffers=Disabled and Global_ Buffers=Enabled qualifiers in the same RMU Restore command.

- Total=total-buffers

  Use this option to specify the number of buffers available for all users. The minimum value you can specify is 2; the maximum value you can specify is the global buffer count stored in the .rdb file.

- User_Limit=buffers-per-user

  Use this option to specify the maximum number of buffers available to each user.

If you do not specify a Global_Buffers qualifier, the database is restored with the values that were in effect when the database was backed up.

When you specify two or more options with the Global_Buffers qualifier, use a comma to separate each option and enclose the list of options within parentheses.

Incremental

The Incremental qualifier restores a database from an incremental backup file.

Use the Incremental qualifier only when you have first issued an RMU Restore command that names the full backup file that was the basis for this incremental backup file. Each incremental backup file is tied to a particular full backup file.

After restoring both the full and the incremental backup files, you have restored the database to the condition it was in when you performed the incremental database backup operation.

By default, RMU Restore performs a full restore operation on the backup file.

You cannot specify the After_Journal or Just_Corrupt qualifier with the Incremental qualifier.

Journal=file-name

Allows you to specify a journal file to be used to improve tape performance by a restore operation (including a by-area or just-corrupt restore operation).

The backup operation creates the journal file and writes to it a description of the backup operation. This description contains identification of the tape drives, the tape volumes and their contents. The Journal qualifier directs RMU Restore to read the journal file and select only the useful tape volumes.

The journal file must be the one created at the time the backup operation was performed. If the wrong journal file is supplied, RMU Restore returns an informational message and does not use the specified journal file to select the volumes to be processed.

If you omit the Label qualifier, the restore operation creates a list of volume labels from the contents of the journal file.

A by-area restore operation also constructs a list of useful tape volume labels from the journal file; only those volumes are mounted and processed.

Just_Corrupt
You can apply the Just_Corrupt qualifier as a global qualifier, a local qualifier, or both. See the description of the Just_Corrupt qualifier under the list of File and Area Qualifiers for a description of all three uses.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file have been labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas, and enclose the list of names within parentheses.

In a normal restore operation, the Label qualifier you specify with the RMU Restore command should be the same Label qualifier you specified with the RMU Backup command that backed up your database.

You can use the Label qualifier with indirect file references. See Section 1.3 for more information.

Librarian[=options]
Use the Librarian qualifier to restore files from data archiving software applications that support the Oracle Media Management interface. The file name specified on the command line identifies the stream of data to be retrieved from the Librarian utility. If you supply a device specification or a version number it will be ignored.

Oracle RMU supports retrieval using the Librarian qualifier only for data that has been previously stored by Oracle RMU using the Librarian qualifer.

The Librarian qualifier accepts the following options:

- Reader_Threads=n

  Use the Reader_Threads option to specify the number of backup data streams to read from the Librarian utility. The value of *n* can be from 1 to 99. The default is one reader thread. The streams are named BACKUP_FILENAME.EXT, BACKUP_FILENAME.EXT02, BACKUP_FILENAME.EXT03, up to BACKUP_FILENAME.EXT99. BACKUP_FILENAME.EXT is the backup file name specified in the RMU Backup command.

The number of reader threads specified for a database restore from the Librarian utility should be equal to or less than the number of writer threads specified for the database backup. If the number of reader threads exceeds the number of writer threads, the number of reader threads is set by Oracle RMU to be equal to the number of data streams actually stored in the Librarian utility by the backup. If the number of reader threads specified for the restore is less than the number of writer threads specified for the backup, Oracle RMU will partition the data streams among the specified reader threads so that all data streams representing the database are restored.

The Volumes qualifier cannot be used with the Librarian qualifer. Oracle RMU sets the volume number to be the actual number of data streams stored in the specified Librarian utility.

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  – Level 0 traces all error conditions. This is the default.

  – Level 1 traces the entry and exit from each Librarian function.

  – Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian application.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian utility documentation for the name and location of this image and how it should be installed. For a parallel RMU backup, define RMU$LIBRARIAN_PATH as a system-wide logical name so that the multiple processes created by a parallel backup can all translate the logical.

```
$ DEFINE /SYSTEM /EXECUTIVE_MODE -
_$ RMU$LIBRARIAN_PATH librarian_shareable_image.exe
```

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image. For a parallel RMU backup, the RMU$DEBUG_SBT logical should be defined as a system logical so that the multiple processes created by a parallel backup can all translate the logical.

The following lines are from a backup plan file created by the RMU Backup/Parallel/Librarian command:

```
Backup File = MF_PERSONNEL.RBF
Style = Librarian
Librarian_trace_level = #
Librarian_logical_names = (-
         logical_name_1=equivalence_value_1, -
         logical_name_2=equivalence_value_2)
Writer_threads = #
```

The "Style = Librarian" entry specifies that the backup is going to a Librarian utility. The "Librarian_logical_names" entry is a list of logical names and their equivalence values. This is an optional parameter provided so that any logical names used by a particular Librarian utility can be defined as process logical names before the backup or restore operation begins. For example, some Librarian utilities provide support for logical names for specifying catalogs or debugging.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Loader_Synchronization

Allows you to preload tapes in order to minimize the need for operator support. When you specify the Loader_Synchronization qualifier and specify multiple tape drives, the restore operation reads from the first set of tape volumes concurrently, then waits until all concurrent tape operations conclude before assigning the next set of tape volumes. This ensures that the tapes can be loaded into the loaders or stackers in the order required by the restore operation.

The Loader_Synchronization qualifier does result in reduced performance. For maximal performance, no drive should remain idle, and the next identified volume should be placed on the first drive that becomes idle. However, because the order in which the drives become idle depends on many uncontrollable factors and cannot be predetermined, the drives cannot be preloaded with tapes.

Because the cost of using the Loader_Synchronization qualifier is dependent on the hardware configuration and the system load, the cost is unpredictable. A 5% to 20% additional elapsed time for the operation is typical. You must determine whether the benefit of a lower level of operator support compensates for the loss of performance. The Loader_Synchronization qualifier is most useful for large restore operations.

The Loader_Synchronization qualifier has no effect unless you specify the Volumes qualifier also.

Local_Buffers=local-buffer-options

Allows you to change the default local buffer parameters when you restore a database. The following options are available:

- Number=number-buffers

  Use this option to specify the number of local buffers available for all users. You must specify a number between 2 and 32,767 for the number-buffers parameter.

- Size=buffer-blocks

  The size (in blocks) for each buffer. You must specify a number between 2 and 64 for the buffer-blocks parameter.

  If you specify a value smaller than the size of the largest page defined, RMU Restore automatically adjusts the size of the buffer to hold the largest page defined. For example, if you specify the Local_Buffers=Size=8 qualifier and the largest page size for the storage areas in your database is 64 blocks, RMU Restore automatically interprets the Local_Buffers=Size=8 qualifier as though it were a Local_Buffers=Size=64 qualifier.

The value you specify for the Size option determines the number of blocks for each buffer, regardless of whether local buffering or global buffering is enabled for the database.

If you do not specify a Local_Buffers qualifier, the database is restored with the values that were in effect when the database was backed up.

Log
Log=Brief
Log=Full
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request that the progress of the restore operation be written to SYS$OUTPUT, or the Nolog qualifier to suppress this report. If you specify the Log=Brief option, which is the default if you use the Log option without a qualifier, the log contains the start and completion time of each storage area. If you specify the Log=Full option, the log also contains thread assignment and storage area statistics messages.

If you do not specify the Log or the Nolog qualifier, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Master
Allows you to explicitly state how drives should be used when they are to be accessed concurrently. This is a positional qualifier that designates a tape drive as a master tape drive.

When the Master qualifier is used, it must be used on the first drive specified. All additional drives become slaves to that master until the end of the command line, or until the next Master qualifier, whichever comes first.

If the Master qualifier is used on a drive that does not have an independent I/O path (not a hardware master), performance decreases.

If the Master qualifier is not used, and concurrent tape access is requested (using the Volumes=n qualifier), RMU Restore uses the same automatic configuration procedure it employs with the backup operation to select the master drives.

Using the Master qualifier is an error if you do not specify concurrent tape access (you do not specify the Volumes=n qualifier). See the description of the Volumes qualifier for further information on specifying concurrent tape access.

Media_Loader
Nomedia_Loader
Use the Media_Loader qualifier to specify that the tape device from which RMU Restore is reading the backup file has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, RMU Restore should recognize this fact. However, occasionally RMU Restore does not recognize that a tape device has a loader or stacker. Therefore, after reading the first tape, RMU Restore issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes RMU Restore behaves as though a tape device has a loader or stacker when actually it does not.

If you find that RMU Restore is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that RMU Restore expects a loader or stacker when it should not, specify the Nomedia_ Loader qualifier.

New_Version
Nonew_Version
Specifies whether new versions of database files should be produced if the destination device and directory contain a previous version of the database files.

If you use the New_Version qualifier, the new database file versions are produced. The New_Version qualifier conflicts with the Incremental qualifier.

If you use the Nonew_Version qualifier, the default, an error occurs if an old copy exists of any of the database files being restored.

A restore operation that creates a new database root (.rdb) file must always either disable after-image journaling or create a new .aij file. Attempting to use a pre-existing .aij file with a restored database corrupts the journal and makes future recovery from .aij files impossible. The New_Version qualifier cannot and does not apply to the .aij file.

Nodes_Max=number-cluster-nodes
Specifies a new upper limit on the number of VMScluster nodes from which users can access the restored database. The Nodes_Max qualifier accepts values between 1 and 96 VMScluster nodes. The actual maximum is the highest number of VMScluster nodes possible in the current version of OpenVMS. The default value is the limit defined for the database before it was backed up.

You cannot specify the Nodes_Max qualifier if you use the Incremental or Area qualifier.

Online
Noonline
Specifies that the restore operation be performed while other users are attached to the database. You can specify the online qualifier only with the Area or Just_Corrupt qualifier. The pages to be restored are locked for exclusive access, so the operation is not compatible with any other use of the data in the specified pages.

The default is the Noonline qualifier.

Open_Mode=Automatic
Open_Mode=Manual
Allows you to change the mode for opening a database when you restore that database. When you specify Open_Mode=Automatic, users can invoke the database immediately after it is restored. If you specify Open_Mode=Manual, an RMU Open command must be used to open the database before users can invoke the database.

The Open_Mode qualifier also specifies the mode for closing a database. If you specify Open_Mode=Automatic, you can also use the Close_Wait qualifier to specify a time in minutes before the database is automatically closed.

If you do not specify the Open_Mode qualifier, the database is restored with the open mode of the database that was in effect when the database was backed up.

Options=file-spec
Specifies the options file that contains storage area names, followed by the storage area qualifiers that you want applied to that storage area.

You can direct RMU Restore to create an options file for use with this qualifier by specifying the Restore_Options qualifier with the RMU Backup, RMU Dump, and RMU Dump Backup commands. See Section 1.10, Section 1.19, and Section 1.22 for details.

If you create your own options file, *do not* separate the storage area names with commas. Instead, put each storage area name on a separate line in the file. You can include any or all of the area qualifiers in the options file. (See the format section for the list of Area qualifiers.) You can use the DCL line continuation character, a hyphen (-), or the comment character (!) in the options file. The default file extension is .opt.

Output=file-spec
This qualifier redirects log output from SYS$OUTPUT to the specified file. If a file extension is not specified in the output file specification, the output file extension will be ".LIS".

The default if this qualifier is not specified is to send output from the RESTORE command to SYS$OUTPUT. This qualifier must specify a valid file specification and cannot be negated. The /NOLOG qualifier cannot be specified on the same command line as the /OUTPUT qualifier. The qualifer /LOG=BRIEF or /LOG=FULL can be specified with the /OUTPUT=file_spec qualifier to control the amount of information written to the designated output file. The /LOG=BRIEF (the default) qualifier will display the start and completion time of the backup or restore of each database storage area. The /LOG=FULL qualifier will also display thread assignment information and statistical information for each storage area.

In this example, the /LOG=BRIEF qualifier is used first with the RMU/RESTORE command and then with the RMU/BACKUP command to direct output to the TEST.LIS file.

```
$ rmu/backup/log=brief/output=test mf_personnel mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, BACKUP operation completed at 27-JUN-2018 16:41:00.35
$ delete test.lis;*
$!
$ sql
drop database fi disk:[directory]mf_personnel;
$ rmu/restore/nocdd/dir=disk:[directory]/log=brief/output=test mfp.rbf
$ dir test.lis
Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, RESTORE operation completed at 27-JUN-2018 16:43:00.37
```

In this example, the /LOG=FULL qualifier is used first with the RMU/RESTORE command and then with the RMU/BACKUP command to direct output to the TEST.LIS file.

1.45 RMU Restore Command

```
$ rmu/backup/log=full/output=test mf_personnel mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, BACKUP operation completed at 27-JUN-2018 16:45:00.23
$ delete test.lis;*
$!
$ sql
drop database fi disk:[directory]mf_personnel;
$ rmu/restore/nocdd/dir=disk:[directory]/log=full/output=test mfp.rbf
$ dir test.lis

Directory DISK:[DIRECTORY]

TEST.LIS;1

Total of 1 file.
$ sear test.lis RMU-I-COMPLETED
%RMU-I-COMPLETED, RESTORE operation completed at 27-JUN-2018 16:48:00.22
```

Page_Buffers=number-buffers

Specifies the maximum number of buffers Oracle Rdb uses during the RMU Restore operation while the database files are being created. The value of the Page_Buffers qualifier can range from 1 to 5. The default is 3 buffers. Values larger than 3 might improve performance, especially during incremental restore operations.

When RMU Restore enters the stage of reconstructing internal structures at the end of the restore operation, a high value for the Page_Buffers qualifier can be useful for very large databases. However, the cost of using these extra buffers is that memory use is high. Thus, the trade-off during a restore operation is memory use against performance.

Path=cdd-path

Specifies a data dictionary path into which the database definitions be integrated. If you do not specify the Path qualifier, RMU Restore uses the CDD$DEFAULT logical name value of the user who entered the RMU Restore command.

If you specify a relative path name, Oracle Rdb appends the *relative* path name you enter to the CDD$DEFAULT value. If the cdd-path parameter contains nonalphanumeric characters, you must enclose it within quotation marks ("").

Oracle Rdb ignores the Path qualifier if you use the Nocdd_Integrate qualifier or if the data dictionary is not installed on your system.

Progress_Report=n
Displays the performance and progress of the restore operation. This feature can be activated by typing Control-T after the RMU Restore operation has been started from the command line or by adding /PROGRESS_REPORT to the RMU command line.

The use of Control-T has to be enabled at the DCL level using:

```
$ SET CONTROL=T
```

While a Control-T just displays the information once, the /PROGRESS_REPORT qualifier can be used to periodically print the information to SYS$OUTPUT in a batch job.

The /PROGRESS_REPORT qualifier will default to 60 seconds.

Prompt=Automatic
Prompt=Operator
Prompt=Client
Specifies where server prompts are to be sent. When you specify Prompt=Automatic, prompts are sent to the standard input device, and when you specify Prompt=Operator, prompts are sent to the server console. When you specify Prompt=Client, prompts are sent to the client system.

Recovery[=Aij_Buffers=n]
Norecovery
The Recovery=Aij_Buffers=n qualifier allows you to specify the number of recovery buffers to use during an automatic recovery. The default value of *n* is 100 recovery buffers.

The Recovery qualifier explicitly specifies that RMU Restore should attempt an automatic recovery of the .aij files during the restore operation.

Specify either the Recover=Aij_Buffers=n qualifier and the Recovery qualifier only if .aij files are being retained. If you specify either qualifier in a situation where .aij files are not retained (the Aij_Options, After_Journal, or Duplicate qualifier has been specified), a warning message is displayed and RMU Restore performs the restore operation without attempting to recover the .aij files.

The Norecovery qualifier specifies that RMU Restore should not attempt an automatic recovery of the .aij files during the restore operation. Specify this qualifier if you want to use the RMU Recover command with the Until qualifier or if you intend to perform an incremental restore operation.

1.45 RMU Restore Command

Rewind
Norewind
Specifies that the tape that contains the backup file will be rewound before processing begins. The Norewind qualifier, the default, causes the search for the backup file to begin at the current tape position.

The Rewind and Norewind qualifiers are applicable only to tape devices. RMU Restore returns an error message if you use these qualifiers and the target device is not a tape device.

Root=root-file-spec
Specifies the database root (.rdb) file specification of the restored database. See the Usage Notes for information on how this qualifier interacts with the Directory, File, and Snapshot qualifiers and for warnings regarding restoring database files into a directory owned by a resource identifier.

The Root qualifier is only meaningful when used with a multifile database.

Row_Cache_Options=file-spec
Specifies a row cache backing store options file which contains per database and/or per cache row cache backing store directory specifications.

The file-spec must be a valid file specification of an existing options file. This qualifier cannot be negated. The following syntax must be used in the row cache backing store options file.

```
$ TYPE FILENAME.OPT
/BACKING_STORE = device:[directory]
row_cache_name /BACKING_STORE = device:[directory]
row_cache_name /NOBACKING_STORE
```

To modify or remove the current per database default backing store location, either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified on the first line without being preceded by a row cache name.

To modify or remove a current per cache backing store location, an existing row cache name currently defined in the database root file must be specified followed by either /BACKING_STORE = followed by a valid directory specification or /NOBACKING_STORE must be specified. The wild card characters "%" and/or "*" can be specified as part of the row cache name, where "*" can be used in the place of one or more contiguous characters and "%" can be used in the place of a single character. In this case all matching per cache backing store entries will be modified with the following /BACKING_STORE = or /NOBACKING_STORE specification. The RMU/DUMP/HEADER command can be used to display the current per database default row cache backing store directory as well as the current per cache row cache backing store directories.

The /ROW_CACHE_OPTIONS qualifier cannot be used with the /AREA, /INCREMENTAL, /JUST_CORRUPT, /DUPLICATE or /ONLINE qualifiers. It can be used for a full or /ONLY_ROOT restore.

Transaction_Mode=(mode-list)
Sets the allowable transaction modes for the database root file restored by the restore operation. The primary use of this qualifier is when you restore a backup file (of a master database) to create a Hot Standby database. Because only read-only transactions are allowed on a standby database, you should use the Transaction_Mode=Read_Only qualifier setting. This setting prevents modifications to the standby database at all times, even when replication operations are not active. For more information on Hot Standby see the *Oracle Rdb7 and Oracle CODASYL DBMS: Guide to Hot Standby Databases*. The mode-list can include one or more of the following transaction modes:

- All - Enables all transaction modes
- Current - Enables all transaction modes that are set for the source database. This is the default transaction mode.
- None - Disables all transaction modes
- [No]Batch_Update
- [No]Read_Only
- [No]Exclusive
- [No]Exclusive_Read
- [No]Exclusive_Write
- [No]Protected
- [No]Protected_Read
- [No]Protected_Write
- [No]Read_Write
- [No]Shared
- [No]Shared_Read
- [No]Shared_Write

Your restore operation must include the database root file. Otherwise, RMU Restore returns the CONFLSWIT error when you issue an RMU Restore command with the Transaction_Mode qualifier.

If you specify more than one transaction mode in the mode-list, enclose the list in parenthesis and separate the transaction modes from one another with a comma. Note the following:

- When you specify a negated transaction mode, it indicates that a mode is not an allowable access mode. For example, if you specify the Noexclusive_ Write access mode, it indicates that exclusive write is not an allowable access mode for the restored database.
- If you specify the Shared, Exclusive, or Protected transaction mode, Oracle RMU assumes you are referring to both reading and writing in that transaction mode.
- No mode is enabled unless you add that mode to the list, or you use the All option to enable all transaction modes.
- You can list one transaction mode that enables or disables a particular mode followed by another that does the opposite.

  For example, Transaction_Mode=(Noshared_Write, Shared) is ambiguous because the first value disables Shared_Write access and the second value enables Shared_Write access. Oracle RMU resolves the ambiguity by first enabling the modes as specified in the modes-list and then disabling the modes as specified in the modes-list. The order of items in the list is irrelevant. In the example presented previously, Shared_Read is enabled and Shared_Write is disabled.

Users_Max=number-users
Specifies a new upper limit on the number of users that can simultaneously access the restored database. The valid range is between 1 and 2032 users. The default value is the value defined for the database before it was backed up.

You cannot specify the Users_Max qualifier if you use the Incremental qualifier or the Area qualifier.

Volumes=n
Allows you to specify that concurrent tape access is to be used to accelerate the restore operation.

The Volumes qualifier indicates concurrent tape access and specifies the number of tape volumes in the backup file. The number of volumes must be specified accurately for the restore operation to complete.

If you are restoring from a multidisk backup file, the value of "n" indicates the number of disk devices containing backup files needed for the restore operation.

If you do not specify the Volumes qualifier, the restore operation does not use concurrent tape access.

File or Area Qualifiers

Blocks_Per_Page=integer
Lets you restore a database with larger page sizes than existed in the original database. This creates new free space on each page in the storage area file and does not interfere with record clustering. RMU Restore ignores this qualifier when it specifies an integer less than or equal to the current page size of the area.

You might want to increase the page size in storage areas containing hash indexes that are close to full. By increasing the page size in such a situation, you prevent the storage area from extending and still preserve the existing target pages for the hash index.

Extension=Disable
Extension=Enable
Allows you to change the automatic file extension attribute when you restore a database. These qualifiers are positional qualifiers.

Use the Extension=Disable qualifier to disable automatic file extension for a storage area.

Use the Extension=Enable qualifier to enable automatic file extension for a storage area.

If you do not specify the Extension=Disable or Extension=Enable qualifier, the storage areas are restored with the automatic file extension attributes that were in effect when the database was backed up.

File=file-spec
Requests that the storage area to which this qualifier is applied be restored in the specified location.

This qualifier is not valid for single-file databases. This is a positional qualifier.

See the Usage Notes for information on how this qualifier interacts with the Root, Directory, and Snapshot qualifiers and for warnings regarding restoring database files into a directory owned by a resource identifier.

Just_Corrupt
This qualifier replaces the Just_Pages qualifier beginning in Oracle Rdb V7.0.

Allows you to restore the corrupt pages and areas in the database as recorded in the corrupt page table (CPT). The CPT is maintained in the .rdb file. (Note that if the corrupt page table becomes full, the area with the highest number of corrupt pages is marked corrupt and the individual pages for that area are removed from the CPT.)

1.45 RMU Restore Command

Often, only one or a few pages in the database are corrupted due to hardware or software faults. The Just_Corrupt qualifier allows you to recover that database in minimal time with minimal interference; availability of the uncorrupted data is unaffected. It allows you to restrict the restoration to the pages (or areas) logged as corrupt in the corrupt page table.

The Just_Corrupt qualifier is a positional qualifier. If you use it in the global position, RMU Restore restores all the corrupt pages and all the corrupt areas as logged in the corrupt page table. If you use it in the local position, RMU Restore restores only the corrupt pages (or the entire area) of the area name it modifies.

It is possible to mix restoration of complete areas and just corrupt pages in the same command. The following example restores all of AREA_1 (regardless of whether or not it is corrupt), but just the corrupt pages (logged to the CPT) in AREA_2.

```
$ RMU/RESTORE/AREA backup_file  AREA_1, AREA_2/JUST_CORRUPT
```

Note that when the Just_Corrupt qualifier is used globally, all the corrupt pages logged to the CPT for the areas specified are restored. For example, the following command restores all the corrupt pages logged to the CPT for AREA_1 and AREA_2. (However, if one of the areas specified contains no corruptions, an informational message is displayed and that area is not restored.)

```
$ RMU/RESTORE/JUST_CORRUPT backup_file /AREA AREA_1, AREA_2
```

Restoration of corrupt pages and area can be performed on line. Online operations lock only the corrupt pages or areas for the duration of the restore operation. The remainder of the storage area can be read or updated by an application. When an entire area is restored on line, applications are locked out of the entire area for the duration of the restore operation.

There are some restrictions on the use of the Just_Corrupt qualifier:

- The backup file must be a full backup file that contains the selected area.
- When space area management (SPAM) pages are restored, RMU Restore rebuilds the SPAM page using information from the range of data pages that the SPAM page manages.
- Area bit map (ABM) pages can be restored, but their content is not reconstructed. If ABM pages have been corrupted, regenerate them with the RMU Repair command.
- A by-page restore operation is like a by-area restore operation in that after-image journal (AIJ) recovery is required to make the restored data consistent with the rest of the database.

Once the pages are restored, access to these restored pages is prohibited until they are made consistent. Inconsistent pages are stored in the corrupt page table (CPT) and have their timestamp field flagged by Oracle Rdb.

- You can also use the Just_Corrupt qualifier in a restore options file. However, you cannot use any of the following qualifiers with the Just_Corrupt qualifier (neither within an options file nor on the command line):
  - Blocks_Per_Page
  - Extension
  - File
  - Incremental
  - Read_Only
  - Read_Write
  - Snapshot
  - Spams
  - Thresholds

  You can use the Just_Corrupt qualifier in conjunction with the Journal=file qualifier to greatly speed up processing of a large tape backup file. When you use the Journal qualifier, only those tapes containing corrupt pages, areas, or both, are mounted and processed.

Just_Pages[=(p1,p2,...)]
This qualifier is replaced with the Just_Corrupt qualifier beginning in Oracle Rdb V7.0. See the description of the Just_Corrupt qualifier.

Read_Only
Use the Read_Only qualifier to change a read/write storage area or a write-once storage area to a read-only storage area.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas are restored with the read/write attributes that were in effect when the database was backed up.

This is a positional qualifier.

Read_Write
Use the Read_Write qualifier to change a read-only storage area or a write-once storage area to a read/write storage area.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas are restored with the read/write attributes that were in effect when the database was backed up.

This is a positional qualifier.

Snapshot=(Allocation=n,File=file-spec)
If you specify the Allocation parameter, specifies the snapshot file allocation size in *n* pages for a restored area. If you specify the File parameter, specifies a new snapshot file location for the restored storage area to which it is applied. If you specify only a directory (and not a file specification), the directory specification must end with a backslash (\ ). You can specify the Allocation parameter only, the File parameter only, or both parameters; however, if you specify the Snapshots qualifier, you must specify at least one parameter.

This is one of the commands used to alter the parameters of the restored database from those defined at the time of the database backup. Others are /DIRECTORY, /ROOT and /FILE.

See the Usage Notes for information on how this qualifier interacts with the Root, File, and Directory qualifiers.

The Shapshot qualifier is a positional qualifier. It can be used locally or globally, depending on where the qualifier is placed on the command line. See Examples 22 and 23.

To save read/write disk space, you can specify that less space be allocated for the storage area's .snp file when it remains as a read/write file on a read/write disk. If the keyword Allocation is omitted, the original allocation is used. This qualifier is not valid for single-file databases.

You cannot specify an .snp file name for a single-file database. When you create an .snp file for a single-file database, Oracle Rdb does not store the file specification of the .snp file. Instead, it uses the file specification of the database root (.rdb) file to determine the file specification of the .snp file.

If you want to place the .snp file on a different device or directory, Oracle Corporation recommends that you create a multifile database. However, you can work around the restriction by defining a search list for a concealed logical name. (However, do not use a nonconcealed rooted logical name to define database files; a database created with a non-concealed rooted logical name can be backed up, but may not restore correctly when you attempt to restore the files to a new directory.)

To create a database with an .snp file on a different device or directory, define a search list by using a concealed logical name. Specify the location of the root file as the first item in the search list. When you create the database, use the logical name for the directory specification. Then, copy the .snp file to the second device. The following example demonstrates the workaround:

```
$ ! Define a concealed logical name.
$ DEFINE /TRANS=CONCEALED/SYSTEM TESTDB USER$DISK1:[DATABASE], -
_$ USER$DISK2:[SNAPSHOT]
$
$ SQL
SQL> -- Create the database.
SQL> --
SQL> CREATE DATABASE FILENAME  TESTDB:TEST;
SQL> EXIT
$ !
$ ! Copy the snapshot (.snp) file to the second disk.
$ COPY USER$DISK1:[DATABASE]TEST.SNP -
_$ USER$DISK2:[SNAPSHOT]TEST.SNP
$ !
$ ! Delete the snapshot (.snp) file from the original disk.
$ DELETE USER$DISK1:[DATABASE]TEST.SNP;
```

Spams
Nospams

Enables the space area management (SPAM) pages for the specified area. The Nospams qualifier disables the SPAM pages for the specified area. The default is to leave the attribute unchanged. The Spams and Nospams qualifiers are not allowed for a storage area that has a uniform page format. This is a positional qualifier.

Thresholds=(val1[,val2[,val3]])

Specifies a storage area's fullness percentage threshold. You can adjust SPAM thresholds to improve future space utilization in the storage area. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the space management entry for the data page is updated to contain that threshold value.

The Thresholds qualifier applies only to storage areas with a mixed page format.

If you do not use the Thresholds qualifier with the RMU Restore command, Oracle Rdb uses the storage area's original thresholds.

This is a positional qualifier.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on setting SPAM thresholds.

Usage Notes

- To use the RMU Restore command for a database, you must have the RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- The RMU Restore command provides four qualifiers, Directory, Root, File, and Snapshots, that allow you to specify the target for the restored files. The **target** can be just a directory, just a file name, or a directory and file name.

  If you use all or some of these four qualifiers, apply them as follows:

  – Use the Root qualifier to indicate the target for the restored database root file.

  – Use local application of the File qualifier to specify the target for the restored storage area or areas.

  – Use local application of the Snapshots qualifier to specify the target for the restored snapshot file or files.

  – Use the Directory qualifier to specify a default target directory. The default target directory is the directory to which all files not qualified with the Root, File, or Snapshot qualifier are restored. It is also the default directory for files qualified with the Root, File, or Snapshot qualifier if the target for these qualifiers does not include a directory specification.

  Note the following when using these qualifiers:

  – Global application of the File qualifier when the target specification includes a file name causes RMU Restore to restore all of the storage areas to different versions of the same file name. This creates a database that is difficult to manage.

  – Global application of the Snapshot qualifier when the target specification includes a file name causes RMU Restore to restore all of the snapshot files to different versions of the same file name. This creates a database that is difficult to manage.

- Specifying a file name or extension with the Directory qualifier is permitted, but causes RMU Restore to restore all of the files (except those specified with the File or Root qualifier) to different versions of the same file name. Again, this creates a database that is difficult to manage.

See Example 17.

- When you restore a database into a directory owned by a resource identifier, the ACE for the directory is applied to the database root file ACL first, and then the Oracle RMU ACE is added. This method is employed to prevent database users from overriding OpenVMS file security. However, this can result in a database which you consider yours, but to which you have no Oracle RMU privileges to access. See the *Oracle Rdb Guide to Database Maintenance* for details.
- If a backup file to tape is created using a single tape device, it must be restored using a single tape device; it cannot be restored using multiple tape devices.

Note

An incremental backup file created for a database running under one version of Oracle Rdb cannot be applied if that database has been restored under another version of Oracle Rdb. For example, if you do the following, step 6 fails with the error message, "XVERREST, Cross version RESTORE is not possible for by-area or incremental functions":

1. Apply a full backup operation to a Version 7.1 database.
2. Apply updates to the database.
3. Perform an incremental backup operation on the database.
4. Move backup files to a system running Oracle Rdb Version 7.2.
5. Restore the database by using the full backup file.
6. Attempt to apply the incremental backup file created in step 1.

- If you apply an incremental backup file, you must specify the Norecovery qualifier when you issue a full RMU Restore command for the corresponding full backup file.

1.45 RMU Restore Command

- If you mistakenly attempt to restore a backup file in a version of Oracle Rdb that is earlier than the version for which the backup file was created, you might receive INVRECTYP errors and your operation will probably terminate with an access violation (ACCVIO) exception. If you receive this error, check the version of the backup file and the version of Oracle Rdb you are running. Be sure the environment version matches, or is greater than, the version under which the backup file was created.
- RMU Restore might create an .rdb file and .rda files when it starts up. If you specify the Log qualifier, these files will be noted in the log file. These are *not* database files until the end of the operation when they have been populated with the backed-up contents. Therefore, if the restore operation aborts or is stopped using Ctrl/Y, you must delete these unpopulated files by using the DCL DELETE command. You know which files to delete by the contents of the backup file and the form of the command issued, or by examining the output in the log file if you specified the Log qualifier. Deleting the files usually requires OpenVMS privileges. Until they are restored, these files are not a database, and Oracle RMU or SQL operations do not function with them.
- RMU Restore preserves any area reservations and after-image journal (.aij) file reservations that exist in the backed-up database.
- If you restore a database without its root file ACL (using the Noacl qualifier with the RMU Restore command, for example), a user who wants to create ACL entries for the database must have the OpenVMS SECURITY or BYPASS privilege.
- The RMU Restore command with the Area and Online qualifiers requires exclusive access to the area files being restored. The RMU Restore command with the Area, Online, and Just_Corrupt qualifiers requires exclusive access to only the pages being restored.
- There are no restrictions on the use of the Nospams qualifier with storage areas that have a mixed page format, but the use of the Nospams qualifier typically causes severe performance degradation. The Nospams qualifier is useful only where updates are rare and batched, and access is primarily by database key (dbkey).
- The RMU Restore command automatically uses the RMU Convert command when restoring the database to a system with a more recent version of Oracle Rdb software. When this is done, the metadata in the Oracle Rdb database changes and invalidates incremental backup files from the previous version. By default, no areas are reserved and one .aij file is reserved. (You can override the after-image journal default reservation by

using the Aij_Options qualifier.) See Section 1.16 for information on the versions of Oracle Rdb that the Convert command supports.

- Always back up your Oracle Rdb databases as recommended in the *Oracle Rdb Installation and Configuration Guide* just prior to installing a newer version of Oracle Rdb software. The last backup file made prior to converting to a more recent version of Oracle Rdb should be a full and complete backup file.
- See the *Oracle Rdb Guide to Database Maintenance* for information on the steps RMU Restore follows in tape label checking when you restore a database from tape.
- RMU Restore might initialize the SPAM thresholds for some data pages of some storage areas that have a uniform page format to values that are not acceptable to the RMU Verify command. This occurs when some of the data pages in a logical area are restored before the logical area definition (Area Inventory). This is not a frequent occurrence, and when it does happen, the consequences are usually cosmetic (the RMU Verify command issues a warning message for each page affected). However, if many pages are affected, the volume of warnings can cause you to overlook a real problem. Moreover, in some cases, this can result in additional I/O operations when new data is stored in an affected table.

  As a workaround, you can use the RMU Repair command to reconstruct the SPAM pages in one or more storage areas. The RMU Repair command corrects the condition caused by the RMU Restore command as well as other SPAM page corruptions. See Section 1.43 for more information on the RMU Repair command.

Examples

Example 1

The following example restores the mf_personnel database from the backup file pers_bu.rbf and requests a new version of the database file. Because the After_Journal qualifier has been specified, automatic recovery will not be attempted.

```
$ RMU/RESTORE/NEW_VERSION/AFTER_JOURNAL=AIJ_DISK:[AIJS]PERSAIJ -
_$ /NOCDD_INTEGRATE/LOG  PERS_BU -
_$ EMP_INFO  /THRESHOLDS=(65,75,80)/BLOCKS_PER_PAGE=3
```

1.45 RMU Restore Command

The command changes the .aij file location and name to AIJ_DISK:[AIJS]PERSAIJ.AIJ, prevents integration with the data dictionary, and displays the progress of the restore operation. For the storage area, EMP_INFO, the command changes the SPAM threshold values to 65%, 75%, and 80%, and increases the number of blocks per page to 3 blocks.

Example 2

Assume that at 10 A.M., Wednesday, October 25, 2005, a disk device hardware failure corrupted all the files on the device, including the mf_personnel.rdb file. The following command restores the full database backup file (pers_full_oct22.rbf) created on the previous Sunday and then restores the incremental backup file made on Tuesday. Note that an incremental database backup file was created on Monday, but each new incremental backup file made since the latest full backup file replaces previous incremental backup files made since the last full backup operation.

```
$ RMU/RESTORE/LOG/NORECOVERY MUA1:PERS_FULL_OCT22.RBF
$ RMU/RESTORE/INCREMENTAL/CONFIRM/LOG/NORECOVERY -
_$ PERS_INCR_OCT24.RBF
```

At this point, the database is current up until 11:30 P.M., Tuesday, when the last incremental backup file was made of mf_personnel. Because after-image journaling is enabled for this database, automatic recovery of the .aij file could have been employed. However, if the recovery process should fail for some reason or, as in this case, the Norecovery qualifier is specified, you can still use the RMU Recover command to apply the .aij file that contains changes made to the database from 11:30 P.M., Tuesday, until just before the hardware failure to the restored mf_personnel.rdb file and its storage area files. For example:

```
$ RMU/RECOVER/UNTIL = "25-OCT-2005 09:55:00.00" -
_$ AIJ_DISK:[AIJS]PERSAIJ.AIJ;1
```

Example 3

If a storage area is on a disk that fails, you might want to move that storage area to another disk by using the RMU Restore command. The following RMU Restore command restores only the EMPIDS_OVER storage area from the full backup file of mf_personnel, and moves the EMPIDS_OVER storage area and snapshot (.snp) file to a new location on the 333$DUA11 disk. The recovery operation is only required if the required .aij file has been backed up and is no longer in the current aij state.

```
$ RMU/RESTORE/AREA 222$DUA20:[BACKUPS]MF_PERS_BU.RBF -
_$ EMPIDS_OVER /FILE=333$DUA11:[DBS]EMPIDS_OVER.RDA -
_$ /SNAPSHOT=(FILE=333$DUA11:[DBS]EMPIDS_OVER.SNP)
$ !
$ ! Recovery from the after-image journal is automatic.  If
$ ! automatic recovery is not possible, or if the Norecovery
$ ! qualifier had been specified, perform the following:
$ !
$ RMU/RECOVER/AREA AIJ_DISK:PERS.AIJ
```

Example 4

The following example demonstrates how you can use by-area backup and restore operations for a single storage area in the mf_personnel database. In addition, it demonstrates the use of the automatic recovery feature of the RMU Restore command.

```
$ !
$ ! Create an .aij file for the database. Because three
$ ! .aij files are created, fixed-size .aij
$ ! journaling will be used.
$ !
$ RMU/SET AFTER_JOURNAL/ENABLE/RESERVE=4     -
_$ /ADD=(name=AIJ1, FILE=DISK2:[CORP]AIJ_ONE)   -
_$ /ADD=(name=AIJ2, FILE=DISK2:[CORP]AIJ_TWO)   -
_$ /ADD=(NAME=AIJ3, FILE=DISK2:[CORP]AIJ_THREE) -
_$ MF_PERSONNEL.RDB
%RMU-W-DOFULLBCK, full database backup should be done to
 ensure future recovery
$ RMU/BACKUP MF_PERSONNEL DISK3:[BACKUP]MF_PERS.RBF

$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';

SQL> --
SQL> -- On Monday, define a new row in the DEPARTMENTS table. The
SQL> -- new row is stored in the DEPARTMENTS storage area.
SQL> --
SQL> INSERT INTO DEPARTMENTS
cont>   (DEPARTMENT_CODE, DEPARTMENT_NAME, MANAGER_ID,
cont>   BUDGET_PROJECTED, BUDGET_ACTUAL)
cont>   VALUES ('WLNS', 'Wellness Center', '00188', 0, 0);
1 row inserted
SQL>

SQL> COMMIT;
SQL> EXIT;
```

1.45 RMU Restore Command

```
$ !
$ ! Assume that you know that the only storage area ever updated in
$ ! the mf_personnel database on Tuesdays is the SALARY_HISTORY
$ ! storage area, and you decide that you will create an incremental
$ ! backup file of just the SALARY_HISTORY storage area on Tuesday.
$ ! Before you perform the by-area backup operation on the
$ ! SALARY_HISTORY storage area on Tuesday, you must perform a full
$ ! and complete backup operation on the mf_personnel database when
$ ! it is in a known and working state.
$ !

$ RMU/BACKUP MF_PERSONNEL.RDB -
_$ DISK3:[BACKUP]MF_MONDAY_FULL.RBF
$ !

SQL> --
SQL> -- On Tuesday, two rows are updated in
SQL> -- the SALARY_HISTORY storage area.
SQL> --
SQL> UPDATE SALARY_HISTORY
cont>    SET SALARY_END ='20-JUL-2003 00:00:00.00'
cont>    WHERE SALARY_START='14-JAN-1993 00:00:00.00'
cont>    AND EMPLOYEE_ID = '00164';
1 row updated
SQL> UPDATE SALARY_HISTORY
cont>    SET SALARY_START ='5-JUL-2000 00:00:00.00'
cont>    WHERE SALARY_START='5-JUL-1990 00:00:00.00'
cont>    AND EMPLOYEE_ID = '00164';
1 row updated

SQL> COMMIT;
SQL> EXIT;

$ !
$ ! On Tuesday, you create an incremental backup file of the
$ ! SALARY_HISTORY storage area only. Only the SALARY_HISTORY
$ ! storage area is included in the by-area backup file.
$ ! Oracle RMU provides an informational message telling
$ ! you that not all storage areas in the database are included
$ ! in the mf_tuesday_partial.rbf backup file.
```

```
$ RMU/BACKUP/INCLUDE=(SALARY_HISTORY) -
_$ /INCREMENTAL/LOG DISK1:[USER]MF_PERSONNEL.RDB -
_$ DISK3:[BACKUPS]MF_TUESDAY_PARTIAL.RBF
%RMU-I-NOTALLARE, Not all areas will be included in
 this backup file
%RMU-I-LOGLASCOM, Last full and complete backup was dated
 18-JAN-2006 11:19:46.31
%RMU-I-BCKTXT_00, Backed up root file
 DISK1:[DB]MF_PERSONNEL.RDB;1
%RMU-I-BCKTXT_03, Starting incremental backup of
 storage area DISK3:[SA}SALARY_HISTORY.RDA;1 at
 18-JAN-2006 11:20:49.29
%RMU-I-BCKTXT_13, Completed incremental backup of
 storage area DISK3:[SA]SALARY_HISTORY.RDA;1 at
 18-JAN-2006 11:20:49.40
%RMU-I-COMPLETED, BACKUP operation completed at
 18-JSN-2006 11:20:49.59
   .
   .
   .

$ !

SQL> -- Update another row in the SALARY_HISTORY table:
SQL>  UPDATE SALARY_HISTORY
cont>     SET SALARY_START ='23-SEP-1991 00:00:00.00'
cont>     WHERE SALARY_START='21-SEP-1981 00:00:00.00'
cont>     AND EMPLOYEE_ID = '00164';
1 row updated
SQL> COMMIT;
SQL> EXIT;

$ ! Assume that a disk device hardware error occurs here
$ ! and only the SALARY_HISTORY storage area and snapshot
$ ! file is lost. Also assume that the database root (.rdb)
$ ! file and other storage areas in the database are still
$ ! fine and do not need to be restored or recovered.
$ ! Therefore, you do not need to restore the .rdb file or
$ ! other storage areas from the full and complete backup
$ ! file. Because only the SALARY_HISTORY storage area was
$ ! lost, you must do the following:
$ ! 1) Restore the SALARY_HISTORY storage area and snapshot
$ !    file from the last full and complete backup file.  Note
$ !    this operation can be done on line.  Specify the Norecovery
$ !    qualifier because you still have an incremental restore
$ !    operation to perform.
$ ! 2) Restore the SALARY_HISTORY storage area from the last
$ !    incremental backup file.  Note this operation can be
$ !    done on line.  This time do not specify the Norecovery
$ !    qualifier so that the automatic recovery provided by
$ !    Oracle RMU will be implemented.
$ !
```

1.45 RMU Restore Command

```
$ RMU/RESTORE/NOCDD_INTEGRATE/ONLINE/LOG/NORECOVERY -
_$ /AREA DISK3:[BACKUP]MF_MONDAY_FULL.RBF SALARY_HISTORY
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 at 18-JAN-2006 11:25:13.17
%RMU-I-RESTXT_24, Completed full restore of storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 at 18-JAN-2006 11:25:13.86
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]SALARY_HISTORY.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page is 2
 blocks long
%RMU-I-AIJWASON, AIJ journaling was active when the database
 was backed up
%RMU-I-AIJRECFUL, Recovery of the entire database starts with
 AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area SALARY_HISTORY starts with
 AIJ file sequence 0
%RMU-I-COMPLETED, RESTORE operation completed at 18-JAN-2006 11:25:14.51

$ RMU/RESTORE/NOCDD_INTEGRATE/INCREMENTAL/ONLINE/LOG -
_$ /AREA DISK3:[BACKUPS]MF_TUESDAY_PARTIAL.RBF SALARY_HISTORY
DISK1:[USER]MF_PERSONNEL.RDB;1, restore incrementally? [N]:Y
%RMU-I-RESTXT_22, Starting incremental restore of storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 at 18-JAN-2006 11:29:35.54
%RMU-I-RESTXT_25, Completed incremental restore of storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 at 18-JAN-2006 11:29:35.64
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]SALARY_HISTORY.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page is 2
 blocks long
%RMU-I-AIJWASON, AIJ journaling was active when the database
 was backed up
%RMU-I-AIJRECFUL, Recovery of the entire database starts with
 AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area SALARY_HISTORY starts with
 AIJ file sequence 0
%RMU-I-AIJBADAREA, inconsistent storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 needs AIJ sequence number 0
%RMU-I-LOGRECDB, recovering database file
 DISK1:[USER]MF_PERSONNEL.RDB;1
%RMU-I-AIJAUTOREC, starting automatic after-image journal recovery
%RMU-I-LOGOPNAIJ, opened journal file DISK2:[CORP]AIJ_ONE.AIJ;17
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%RMU-I-LOGRECOVR, 1 transaction committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 3 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJALLDONE, after-image journal roll-forward operations completed
%RMU-I-LOGSUMMARY, total 1 transaction committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 3 transactions ignored
%RMU-I-AIJSUCCES, database recovery completed successfully
```

Example 5

In the following example, the options file specifies that the storage area (.rda) files are to be restored to different disks. Note that storage area snapshot (.snp) files are restored to different disks from one another and from their associated storage area (.rda) files; this is recommended for optimal performance. (This example assumes that the disks specified for each storage area file in options_file.opt are different from those where the storage area files currently reside.)

```
$ RMU/RESTORE/NOCDD_INTEGRATE/OPTIONS=OPTIONS_FILE.OPT -
_$ MF_PERS_BCK.RBF

$ TYPE OPTIONS_FILE.OPT

EMPIDS_LOW /FILE=DISK1:[CORPORATE.PERSONNEL]EMPIDS_LOW.RDA -
   /SNAPSHOT=(FILE=DISK2:[CORPORATE.PERSONNEL]EMPIDS_LOW.SNP )

EMPIDS_MID /FILE=DISK3:[CORPORATE.PERSONNEL]EMPIDS_MID.RDA -
   /SNAPSHOT=(FILE=DISK4:[CORPORATE.PERSONNEL]EMPIDS_MID.SNP )

EMPIDS_OVER /FILE=DISK5:[CORPORATE.PERSONNEL]EMPIDS_OVER.RDA -
   /SNAPSHOT=(FILE=DISK6:[CORPORATE.PERSONNEL]EMPIDS_OVER.SNP )

DEPARTMENTS /FILE=DISK7:[CORPORATE.PERSONNEL]DEPARTMENTS.RDA -
   /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]DEPARTMENTS.SNP )

SALARY_HISTORY /FILE=DISK9:[CORPORATE.PERSONNEL]SALARY_HISTORY.RDA -
   /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]SALARY_HISTORY.SNP )

JOBS /FILE=DISK7:[CORPORATE.PERSONNEL]JOBS.RDA -
   /SNAPSHOT=(FILE=DISK8:[CORPORATE.PERSONNEL]JOBS.SNP )

EMP_INFO /FILE=DISK9:[CORPORATE.PERSONNEL]EMP_INFO.RDA -
   /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]EMP_INFO.SNP )

RESUME_LISTS /FILE=DISK11:[CORPORATE.PERSONNEL]RESUME_LISTS.RDA -
   /SNAPSHOT=(FILE=DISK12:[CORPORATE.PERSONNEL]RESUME_LISTS.SNP )

RESUMES /FILE=DISK9:[CORPORATE.PERSONNEL]RESUMES.RDA -
   /SNAPSHOT=(FILE=DISK10:[CORPORATE.PERSONNEL]RESUMES.SNP )
```

Example 6

The following example shows what .aij file sequence to use following an RMU Restore command with the Area qualifier if automatic recovery fails:

```
$ RMU/RESTORE/AREA MFPERS_62691.RBF -
         DEPARTMENTS, JOBS
   .
   .
   .
%RMU-I-AIJWASON, AIJ journaling was active when the
 database was backed up
%RMU-I-AIJRECFUL, Recovery of the entire database
 starts with AIJ file  sequence 0
```

1.45 RMU Restore Command

Example 7

The following example shows how to move a single-file database to a new directory, using the RMU Backup and RMU Restore commands:

```
$ RMU/BACKUP PERSONNEL PERS
$!
$ RMU/RESTORE/NOCDD/NOAFTER_JOURNAL -
_$ /DIRECTORY=DISK4:[USER2] PERS
```

Example 8

The following example shows how to rename a single-file database when you move the database by using the RMU Backup and RMU Restore commands:

```
$ RMU/BACKUP PERSONNEL PERS
$!
$ RMU/RESTORE/NOCDD/NOAFTER_JOURNAL -
_$ /DIRECTORY=DISK4:[USER2]TEST_PERSONNEL PERS
```

Example 9

The following example causes the database being restored from the mf_pers_ bck.rbf backup file to have 60 global buffers, with a limit of 2 buffers for each database user. Because the Enabled option is used, global buffering is in effect for the database immediately after it is restored:

```
$ RMU/RESTORE/NOCDD/GLOBAL_BUFFERS=(ENABLED,TOTAL=60,USER_LIMIT=2) -
_$ MF_PERS_BCK.RBF
```

Example 10

The following command causes the SALARY_HISTORY storage area from the database being restored from the mf_pers_bu.rbf backup file to be restored as a read-only storage area. None of the other database storage areas are modified as part of this restore operation.

```
$ RMU/RESTORE/NOCDD MF_PERS_BU.RBF SALARY_HISTORY /READ_ONLY
```

Example 11

The following example assumes that you are using multiple tape drives to perform a large restore operation. By specifying the Loader_Synchronization and Volumes qualifiers, this command does not require you to load tapes as each completes. Instead, you can load tapes on a loader or stacker and the RMU restore process will wait until all concurrent tape operations have concluded for one set of tape volumes before assigning the next set of tape volumes. This example assumes that the backup operation used two tape output threads and each thread wrote four tapes.

This example uses Master qualifiers to indicate that you want the 111MUA0: and 444MUA2: drives to be master drives.

Using this example, you would:

1. Allocate each tape drive.
2. Manually place tapes BACK01 and BACK05 on the 111MUA0: master drive.
3. Manually place tapes BACK02 and BACK06 on the 333MUA2: master drive.
4. Manually place tapes BACK03 and BACK07 on the 222MUA1: slave drive.
5. Manually place tapes BACK04 and BACK08 on the 444MUA3: slave drive.
6. Mount the first volume (BACK01).
7. Perform the restore operation.
8. Dismount the last tape mounted.
9. Deallocate each tape drive.

```
$ ALLOCATE $111$MUA0:
$ ALLOCATE $222$MUA1:
$ ALLOCATE $333$MUA2:
$ ALLOCATE $444$MUA3:
$
$ MOUNT/FOREIGN $111$MUA0:
$
$ RMU/RESTORE/LOG/REWIND/LOADER_SYNCHRONIZATION             -
_$ /LABEL=(BACK01, BACK02, BACK03, BACK04, BACK05,          -
_$ BACK06, BACK07, BACK08)                                  -
_$ /VOLUMES=8                                               -
_$ $111$MUA0:PERS_FULL_MAR30.RBF/MASTER, $222$MUA1:         -
_$ $333$MUA2:/MASTER, $444$MUA3
$
$ DISMOUNT $222$MUA3:
$
$ DEALLOCATE $111$MUA0:
$ DEALLOCATE $222$MUA1:
$ DEALLOCATE $333$MUA2:
$ DEALLOCATE $444$MUA3:
```

1.45 RMU Restore Command

Example 12

The following example demonstrates the automatic .aij recovery mechanism in the RMU Restore command. The example does the following:

- Uses the RMU Set After_Journal command to reserve space for four .aij files, adds three .aij files, and enables after-image journaling
- Performs a backup operation on the database
- Performs database update activity, which will be written to an .aij file
- Determines the database root file is lost
- Restores and recovers the database in one RMU Restore command

```
$ SET DEFAULT DISK1:[USER]
$ !
$ RMU/SET AFTER_JOURNAL/ENABLE/RESERVE=4        -
_$ /ADD=(name=AIJ1, FILE=DISK2:[CORP]AIJ_ONE)   -
_$ /ADD=(name=AIJ2, FILE=DISK2:[CORP]AIJ_TWO)   -
_$ /ADD=(NAME=AIJ3, FILE=DISK2:[CORP]AIJ_THREE) -
_$ MF_PERSONNEL
%RMU-W-DOFULLBCK, full database backup should be done
 to ensure future recovery
$ !
$ ! Back up database, as instructed.
$ !
$ RMU/BACKUP MF_PERSONNEL DISK3:[BACKUPS]MF_PERS.RBF
$ !
$ ! Database update activity occurs.
$ !
```

```
$!
$! Database is lost.  Issue the RMU Restore command to
$! restore and recover the database.  Because the Norecovery
$! qualifier is not specified, Oracle RMU will
$! automatically attempt to recover the database.
$!
$ RMU/RESTORE DISK3:[BACKUPS]MF_PERS.RBF/NOCDD_INTEGRATE
%RMU-I-AIJRSTAVL, 3 after-image journals available for use
%RMU-I-AIJRSTMOD, 1 after-image journal marked as "modified"
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done
  to ensure future recovery
%RMU-I-LOGRECDB, recovering database file
 DISK1:[USER]MF_PERSONNEL.RDB;1
%RMU-I-AIJAUTOREC, starting automatic after-image
 journal recovery
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward
 operations completed
%RMU-I-AIJONEDONE, AIJ file sequence 1 roll-forward
 operations completed
%RMU-W-NOTRANAPP, no transactions in this journal
 were applied
%RMU-I-AIJALLDONE, after-image journal roll-forward
 operations completed
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery,
 the sequence number needed  will be 1
```

Example 13

The following example demonstrates how to restore and recover all the corrupt pages and areas in the mf_personnel database. Assume that the RMU Show Corrupt_Pages command shows that the JOBS storage area is corrupt and that only page 3 in the DEPARTMENTS storage area is corrupt. All the other storage areas are neither corrupt nor inconsistent. Because the Just_Corrupt qualifier is specified in the global position, and mf_personnel.rbf is a full backup file, the RMU restore process restores all of the JOBS storage area and just page 3 in the DEPARTMENTS storage area. If after-image journaling is enabled, automatic recovery will be attempted.

```
$ RMU/RESTORE/AREA/JUST_CORRUPT MF_PERSONNEL.RBF
```

Example 14

The following example demonstrates how to restore and recover specific corruptions in the mf_personnel database. Like example 12, assume that the RMU Show Corrupt_Pages command shows that the JOBS storage area is corrupt and that only page 3 in the DEPARTMENTS storage area is corrupt. All the other storage areas are neither corrupt nor inconsistent. The backup file, mf_partial.rbf, is a by-area backup file containing backups of the JOBS,

DEPARTMENTS, and SALARY_HISTORY storage areas. In this example, the JOBS, DEPARTMENTS, and SALARY_HISTORY areas are specified for restoring. Because the SALARY_HISTORY area contains no corruptions, an informational message is returned. The RMU restore process restores all of the JOBS storage area and just page 3 in the DEPARTMENTS storage area. If after-image journaling is enabled, automatic recovery will be attempted.

```
$ RMU/RESTORE/JUST_CORRUPT/AREA MF_PARTIAL.RBF JOBS, -
_$ DEPARTMENTS,SALARY_HISTORY
%RMU-I-RESTXT_20, Storage area DISK1:[AREA]SALARY_HISTORY.RDA;1 is not
  corrupt and will not be restored
```

Example 15

The following example demonstrates how to restore and recover specific corruptions in the mf_personnel database along with restoring an area that is not corrupt. Like example 13, assume that the RMU Show Corrupt_Pages command shows that the JOBS storage area is corrupt and that only page 3 in the DEPARTMENTS storage area is corrupt. All the other storage areas are neither corrupt nor inconsistent. The backup file, mf_personnel.rbf, is a full backup file. In this example, the Just_Corrupt qualifier is used locally with the DEPARTMENTS storage area.

The JOBS, DEPARTMENTS, and SALARY_HISTORY areas are specified for restoring. Although the SALARY_HISTORY area contains no corruptions, an informational message is not returned in this case because by specifying the Just_Corrupt qualifier locally with DEPARTMENTS, the Restore command is requesting that the RMU restore process restore the JOBS and SALARY_ HISTORY storage areas regardless of corruptions, and the DEPARTMENTS storage area be restored to fix corruptions. The RMU restore process restores all of the JOBS and SALARY_HISTORY storage areas and just page 3 in the DEPARTMENTS storage area. If after-image journaling is enabled, automatic recovery will be attempted.

```
$ RMU/RESTORE/AREA MF_PERSONNEL.RBF JOBS, SALARY_HISTORY, -
_$ DEPARTMENTS/JUST_CORRUPT
```

Example 16

The following example is the same as example 15, except the Just_Corrupt qualifier is specified locally with the SALARY_HISTORY storage area. Because the SALARY_HISTORY qualifier contains no corruptions, an error message is returned:

```
$ RMU/RESTORE/AREA MF_PERSONNEL.RBF JOBS,SALARY_HISTORY/JUST_CORRUPT, -
_$ DEPARTMENTS/JUST_CORRUPT
%RMU-I-RESTXT_20, Storage area DISK1:[AREA]SALARY_HISTORY.RDA;1 is
 not corrupt and will not be restored
```

Example 17

The following example demonstrates the behavior of the RMU Restore command when the Just_Corrupt qualifier is used both globally and locally. The global use of the Just_Corrupt qualifier overrides an local use of the qualifier. In this case, the RMU restore process restores the JOBS, SALARY_ HISTORY, and DEPARTMENTS storage areas only if they contain corruptions; otherwise an error is returned. Assume, like the previous examples, that only the JOBS and DEPARTMENTS storage areas contain corruptions:

```
$ RMU/RESTORE/JUST_CORRUPT/AREA MF_PERSONNEL.RBF  SALARY_HISTORY, -
_$ JOBS/JUST_CORRUPT, DEPARTMENTS/JUST_CORRUPT
%RMU-I-RESTXT_20, Storage area DISK1:[AREA]SALARY_HISTORY.RDA;1 is
 not corrupt and will not be restored
```

Example 18

The following example demonstrates the use of the Directory, File, and Root qualifiers. In this example:

- The default directory is specified as DISK2:[DIR].
- The target directory and file name for the database root file is specified with the Root qualifier. The target directory specified with the Root qualifier overrides the default directory specified with the Directory qualifier. Thus, the RMU restore process restores the database root in DISK3:[ROOT] and names it COPYRDB.RDB.
- The target directory for the EMPIDS_MID storage area is DISK4:[FILE]. The RMU restore process restores EMPIDS_MID in DISK4:[FILE].
- The target file name for the EMPIDS_LOW storage area is EMPIDS. Thus, the RMU restore process restores the EMPIDS_LOW storage area to the DISK2:[DIR] default directory (specified with the Directory qualifier), and names the file EMPIDS.RDA.
- The target for the EMPIDS_LOW snapshot file is DISK5:[SNAP]EMPIDS.SNP. Thus, the RMU restore process restores the EMPIDS_LOW snapshot file to DISK5:[SNAP]EMPIDS.SNP.
- All the other storage area files and snapshot files in the mf_personnel database are restored in DISK2:[DIR]; the file names for these storage areas and snapshot files remain unchanged.

1.45 RMU Restore Command

```
$ RMU/RESTORE MF_PERSONNEL.RBF -
_$ /DIRECTORY=DISK2:[DIR] -
_$ /ROOT=DISK3:[ROOT]MF_PERSONNEL.RDB -
_$ EMPIDS_MID/FILE=DISK4:[FILE], -
_$ EMPIDS_LOW/FILE=EMPIDS -
_$ /SNAPSHOT=(FILE=DISK5:[SNAP]EMPIDS.SNP)
```

Example 19

The following example demonstrates how to restore a database such that the newly restored database will allow read-only transactions only. After the RMU restore process executes the command, the database is ready for you to start Hot Standby replication operations. See the *Oracle Rdb7 and Oracle CODASYL DBMS: Guide to Hot Standby Databases* for details on starting Hot Standby replication operations.

```
$RMU/RESTORE/TRANSACTION_MODE=READ_ONLY MF_PERSONNEL.RBF
```

Example 20

The following example uses the Nocommit qualifier while restoring a backup file of a database that has a structure level of V7.1 in a V7.2 environment.

```
$ RMU/SHOW VERSION
Executing RMU for Oracle Rdb V7.2-00
$ RMU/RESTORE MFP71.RBF /NOCOMMIT/NOCDD/NORECOVER
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-S-CVTDBSUC, database USER1:[80]MF_PERSONNEL.RDB;1 successfully
converted from version V7.1 to V7.2
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not
available.
$ RMU/SHOW VERSION
Executing RMU for Oracle Rdb V7.2-00
$ RMU/CONVERT/ROLLBACK MF_PERSONNEL.RDB
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.2-00
Are you satisfied with your backup of RDBVMS_USER1:[V71]MF_PERSONNEL.RDB;1
and your backup of any associated .aij files [N]? Y
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-I-CVTROLSUC, CONVERT rolled-back for RDBVMS_USER1:[V71]MF_PERSONNEL.
RDB;1 to version V7.1
```

Example 21

The following example uses the Close_Wait qualifier to set the database close mode to TIMED AUTOMATIC, specifying that the database will be closed automatically in 10 minutes.

```
$ RMU/RESTORE/OPEN_MODE=AUTOMATIC/CLOSE_WAIT=10/DIR=DISK:[DIR] TEST_DB.RBF
$ RMU/DUMP/HEADER=PARAMETERS TEST_DB.RDB
```

Example 22

The following example demonstrates that /SNAPSHOT=(ALLOCATION=N) is a positional qualifier. The behavior is different (local or global) depending on the placement of the qualifier on the command line. In the following example, it is used both globally and locally.

```
MALIBU-> RMU/RESTORE/NOCDD -
         /DIR=SYS$DISK:[]/SNAP=ALLO=12345 [JONES.RDB]MF_PERSONNEL_V71.RDF -
         DEPARTMENTS/SNAP=ALLO=2
MALIBU-> DIR/SIZE *.SNP

Directory DBMS_USER3:[JONES.WORK]

DEPARTMENTS.SNP;1          6
EMPIDS_LOW.SNP;1       24692
EMPIDS_MID.SNP;1       24692
EMPIDS_OVER.SNP;1      24692
EMP_INFO.SNP;1         24692
JOBS.SNP;1             24692
MF_PERS_DEFAULT.SNP;1
                       24692
MF_PERS_SEGSTR.SNP;1
                       24692
SALARY_HISTORY.SNP;1
                       24692

Total of 9 files, 197542 blocks.
```

Example 23

The following example demonstrates how /SNAPSHOT=(ALLOCATION=N) can be used to alter the parameters of the restored database from those defined at the time of the database backup. /SNAPSHOT is ofter used with /FILE: /FILE for the storage area RDA file and /SNAPSHOT for the storage area snapshot file.

```
$ RMU/RESTORE MFP.RBF -
  /DIRECTORY=DISK1:[DIRECTORY] -
  /ROOT=DISK2:[DIRECTORY]MF_PERSONNEL.RDB -
 EMPIDS_MID /FILE=[DISK3:[DIRECTORY]  /SNAPSHOT=(ALLOCATION=2000),  -
 EMPIDS_LOW /FILE=[DISK3:[DIRECTORY]NEWNAME  -
      /SNAPSHOT=(FILE=DISK4:[DIR]NEWNAME, ALLOCATION=3000)
```

In this example, the root would go to one disk, EMPIDS_MID would go to another, EMPIDS_LOW to another disk and the snap to another disk and both snaps would be allocated the specified number of pages. All the other snaps and RDA files would go to where /DIRECTORY points (and the snaps would keep their original allocation).

Example 24

1.45 RMU Restore Command

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ROW_CACHE_OPTIONS qualifier to read backing store directory options files to modify or remove the current per database and per cache Row Cache backing store directories in the database root file. The contents of the options file read is displayed when the command is executed and the RMU/DUMP/HEADER command is used to check the modified backing store directories in the database root file.

```
$ SET VERIFY
$ RMU/RESTORE/NOCDD/NOLOG/DIR=TEST$DIRECTORY-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:BSTORE.OPT -
 TEST$DIRECTORY:RSA.RBF
%RMU-I-RESTXT_18, Processing options file BSTORE.OPT
  /BACKING_STORE=DISK:[DIRECTORY]
  SAL_CACHE /BACKING_STORE=DISK:[DIRECTORY]
  JOB_CACHE/BACKING_STORE=DISK:[DIRECTORY]
  DROP_CACHE /BACKING_STORE=DISK:[DIRECTORY]

$ RMU/DUMP/HEADER/OUT=HDR.LIS TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/MOVE_AREA/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROOT=DISK:[DIRECTORY]FILENAME.EXT
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:WBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file WBSTORE.OPT
  *CACHE /BACKING_STORE=DISK:[DIRECTORY]

$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is "DISK:[DIRECTORY]"
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
      - Cache file directory is "DISK:[DIRECTORY]"
$ RMU/COPY_DATABASE/NOLOG/DIRECTORY=DISK:[DIRECTORY]-
 /ROW_CACHE_OPTIONS=TEST$DIRECTORY:NOBSTORE.OPT -
 TEST$DIRECTORY:RMU_SET_RCACHE_ALTER_DB
%RMU-I-RESTXT_18, Processing options file NOBSTORE.OPT
  /NOBACKING_STORE
  SAL_CACHE /NOBACKING_STORE
  JOB_CACHE/NOBACKING_STORE
  DROP_CACHE /NOBACKING_STORE
```

```
$ RMU/DUMP/HEADER/OUT=HDR.LIS DISK:[DIRECTORY]FILENAME.EXT
$ SEARCH HDR.LIS "DEFAULT BACKING FILE DIRECTORY"
         Default backing file directory is database directory
$ SEARCH HDR.LIS "CACHE FILE DIRECTORY"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
      - Derived cache file directory is "DISK:[DIRECTORY]"
```

Example 25

The following are some examples of using the Confirm options during an RMU/RESTORE.

```
$RMU/RESTORE/NOCDD/REWIND LMA1001:VOL002.RBF MF_PERSONNEL /DIR=[] /LOG
/CONFIRM=QUIT
%MOUNT-I-MOUNTED, MF_PER mounted on LMA1001:
%RMU-I-TAPEDEF, Terminating restore operation as requested by user
%RMU-F-ABORT, operator requested abort on fatal error
%RMU-F-FATALERR, fatal error on RESTORE
%RMU-F-FTL_RSTR, Fatal error for RESTORE operation at 13-JUL-2009 11:43:31.35
```

```
$RMU/RESTORE/NOCDD/REWIND LMA1001:VOL002.RBF MF_PERSONNEL /DIR=[] /LOG
/CONFIRM=RETRY=2
%RMU-I-TAPEDEF, Retrying tape operation as requested by user
%MOUNT-I-MOUNTED, MF_PER mounted on LMA1001:
%RMU-I-TAPEDEF, Retrying tape operation as requested by user
%MOUNT-I-MOUNTED, MF_PER mounted on LMA1001:
%RMU-F-ABORT, operator requested abort on fatal error
%RMU-F-FATALERR, fatal error on RESTORE
%RMU-F-FTL_RSTR, Fatal error for RESTORE operation at 13-JUL-2009 11:43:42.97
```

```
$RMU/RESTORE/NOCDD/REWIND LMA1001:VOL002.RBF MF_PERSONNEL /DIR=[] /LOG
/CONFIRM=OVERRIDE
%RMU-I-TAPEDEF, Overriding tape label as requested by user
%RMU-F-FILACCERR, error opening input file LMA1001:[000000]VOL002.RBF;
-SYSTEM-W-NOSUCHFILE, no such file
%RMU-F-FATALERR, fatal error on RESTORE
%RMU-F-FTL_RSTR, Fatal error for RESTORE operation at 13-JUL-2009 11:43:58.38
```

Example 26

The following example shows the use of the /Progress_Report qualifier to display backup performance every 10 seconds.

```
$ RMU/BACKUP $1$DGA10:[DB]SAMPLE $1$DGA20:[BCK]SAMPLE /DISK /PROGRESS_REPORT=10
Read    18 MB ( 0%) at  18 MB/s, estimated completion time 14:10:41.15
   .
   .
   .
Read  3934 MB (99%) at  28 MB/s, estimated completion time 14:10:39.86
```

Read n MB = raw blocks read so far

(n%) = percent of total blocks read
n MB/s = transfer rate since last display
estimated completion time = recalculated using the current transfer rate

Restores from other than disk files do not display the percentage completed nor the estimated completion time:

```
$ RMU/RESTORE/NOCDD $1$MGA500:SAMPLE,$1$MGA600: /REWIND /PROG=10
Read    72 MB at   14 MB/s
Read   135 MB at   15 MB/s
   .
   .
   .
Read  1441 MB at   12 MB/s
```

Example 26

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ATTRIBUTES qualifier.

```
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTRIBUTES=(THRE=(50,50,50),READ_O)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTRIBUTES=READ_O
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTR=READ_W
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/RESTORE/NOLOG/NOCDD/DIR=DISK:[DIRECTORY] DISK:[DIRECTORY]MFP.RBF
JOBS/ATTR=READ_O
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not available.
```

1.46 RMU Restore Only_Root Command

Permits you to recover more quickly from the loss of a database root (.rdb) file by restoring only the root file. This command is not valid for single-file databases.

Format

RMU/Restore/Only_Root backup-file-spec [storage-area-list]

Command Qualifiers	Defaults
/Active_IO=max-reads	/Active IO=3
/[No]After_Journal=file-spec	See description
/[No]Aij_Options=journal-opts	See description
/[No]Assist	See description
/Attributes=options	See description
/Directory=directory-spec	See description
/Encrypt=({Value=|Name=}[,Algorithm=])	See description
/[No]Initialize_Tsns	/Noinitialize_Tsns
/Label=(label-name-list)	See description
/Librarian[=options]	None
/[No]Log	Current DCL verify value
/[No]Media_Loader	See description
/[No]New_Snapshots	/Nonew_Snapshots
/Nodes_Max=number-cluster-nodes	Existing value
/Options=file-spec	None
/[No]Rewind	/Norewind
/Root=root-file-spec	Existing value
/[No]Set_Tsn=(Tsn=n[,Csn=m])	See description
/Transaction_Mode=(modes-list)	/Transaction_Mode=Current
/[No]Update_Files	/Update_Files
/Users_Max=number-users	Existing value

File or Area Qualifiers	Defaults
/[No]Blocks_Per_Page=integer	/Noblocks_Per_Page
/File=file-spec	See description
/Read_Only	Current value
/Read_Write	Current value
/Snapshot=(Allocation=n,File=file-spec)	See description
/[No]Spams	Current value
/Thresholds=(val1[,val2[,val3]])	Existing area file value

1.46 RMU Restore Only_Root Command

Description

The RMU Restore Only_Root command rebuilds only the database root (.rdb) file from a backup file, produced earlier by an RMU Backup command, to the condition the .rdb file was in when the backup operation was performed. Use the command qualifiers to update the .rdb file. The area qualifiers alter only the .rdb file, not the storage areas themselves. Use the area qualifiers to correct the restored backup root file so that it contains storage area information that was updated since the last backup operation was performed on the database. This is useful when you need to match the root from an older backup file of your database with the area information in the more recent backup file of your database in order to have a usable database.

When the .rdb file is restored by itself, be sure that you correctly set the transaction state of the database with the Initialize_Tsns qualifier or the Set_Tsn qualifier. If the database transaction sequence number (TSN) and commit sequence number (CSN) are not set to the same values as those that were in the lost .rdb file, there will be an inconsistency in the journaling if after-image journaling is enabled. Therefore, you cannot recover the database by using journal files created before you used either the Initialize_Tsns qualifier or the Set_Tsn qualifier in a restore-only-root operation.

You should set the TSN to a value equal to or greater than the value that was in the lost .rdb file. If the TSN is set to a lower value than the value stored in the lost database root file, the database is corrupted, and it might return incorrect data or result in application failures. If the number you have selected is less than the Next CSN and Next TSN values, you will receive a fatal error message as follows:

```
%RMU-F-VALLSSMIN, value (0:40) is less than minimum allowed
 value (0:74) for Set_Tsn=tsn
```

After the set TSN and reinitialize TSN operations complete, and after you have verified the .rdb file, enabled after-image journaling, and the new .aij file is created, all .aij records are based on the new starting TSN and CSN numbers in the .rdb file.

Although Oracle Corporation recommends that your backup strategy ensures that you maintain a current full and complete database backup file, it is possible to restore the database from current full by-area backup files only. This is accomplished by restoring the root and specifying the Noupdate_Files and Noset_Tsn qualifiers. When you specify the Noset_Tsn qualifier, the TSN and CSN values on the restored database will be the same as those recorded in the backup file. When you specify the Noupdate_Files qualifier, the database root is restored but RMU Restore Only_Root will not link that restored root to any of the area files, nor will it create or update the snapshot (.snp) files.

1.46 RMU Restore Only_Root Command

By specifying the Noupdate_Files and Noset_Tsn qualifiers with the RMU Restore Only_Root command, you can use the following strategy to restore your database:

1. Restore the root from the most recent full by-area backup file.
2. Restore the storage areas by applying the by-area backup files in reverse order to their creation date.

   Apply the most recent by-area backup file first and the oldest by-area backup file last. (Be sure you do not restore any area more than once.)
3. Recover the database by applying the after-image journal (.aij) files.

   You can recover the .aij files manually by using the RMU Recover command. Or, if the state of your .aij files permits it, you can allow RMU Restore Only_Root to automatically recover the .aij files by *not* specifying the Norecovery qualifier with the last RMU Restore command you issue. For details on the automatic recovery feature of the RMU Restore command, see Section 1.45. (The automatic recovery feature is not available for the RMU Restore Only_Root command.)

When you use this strategy, be sure that the first RMU Restore command after the RMU Restore Only_Root command includes the most recent RDB$SYSTEM storage area. The RDB$SYSTEM storage area contains the structures needed to restore the other database storage areas. For this reason, Oracle Corporation suggests that you back up the RDB$SYSTEM storage area in every by-area backup operation you perform.

See Example 6 in the Examples section for a demonstration of this method.

Note that the database backup file must be recent—differences between the database and backup file must be known, and the number of storage areas must be unchanged since the backup file was created. If you have moved a storage area, use the File qualifier to show its new location and the Snapshot qualifier to indicate the current version of the area's .snp file.

Note

You must perform a full and complete backup operation on your database when the RMU Restore Only_Root command completes. Oracle Corporation recommends that you define a new after-image journal configuration with the RMU Restore Only_Root command by using either the After_Journal or the Aij_Options qualifier. This action ensures that the new .aij file can be rolled forward in the event that another database restore operation becomes necessary.

1.46 RMU Restore Only_Root Command

Command Parameters

backup-file-spec
A file specification for the backup file produced by a previous RMU Backup command. The default file extension is .rbf.

Note that you cannot perform a remote restore operation on an .rbf file that has been backed up to tape and then copied to disk. When copying .rbf files to disk from tape, be sure to copy them onto the system on which you will be restoring them.

Depending on whether you are performing a restore operation from magnetic tape, disk, or multiple disks, the backup file specification should be specified as follows:

- Restoring from magnetic tape

  If you used multiple tape drives to create the backup file, the backup-file-spec parameter must be provided with (and only with) the first tape drive name. Additional tape drive names must be separated from the first and subsequent tape drive names with commas, as shown in the following example:

  ```
  $ RMU/RESTORE /REWIND $111$MUA0:PERS_FULL_NOV30.RBF,$112$MUA1:
  ```

- Restoring from multiple or single disk files

  If you used multiple disk files to create the backup file, the backup-file-spec parameter must be provided with (and only with) the first disk device name. Additional disk device names must be separated from the first and subsequent disk device names with commas. You must include the Disk_file qualifier. For example:

  ```
  RMU/RESTORE/ONLY_ROOT/DISK_FILE DISK1:[DIR1]MFP.RBF,DISK2:[DIR2],
  DISK3:[DIR3]
  ```

  As an alternative to listing the disk device names on the command line (which can exceed the line-limit lenght for a command line if you use several devices), you can specify an options file in place of the backup-file-spec. For example:

  ```
  $ RMU/RESTORE/ONLY-ROOT/DISK_FILE" @DEVICES.OPT"
  ```

  The contents of devices.opt might appear as follows:

  ```
  DISK1:[DIR1]MFP.RBF
  DISK2:[DIR2]
  DIS3:[DIR3]
  ```

The backup files referenced from sjuch an options file are:

```
DISK1:[DIR1]MFP.RBF
DISK2:[DIR2]MFP01.RBF
DISK3:[DIR3]MFP02.RBF
```

storage-area-list
This option is a list of storage area names from the database. Use it in the following situations:

- When you need to change the values for thresholds with the Thresholds qualifier or blocks per page with the Blocks_Per_Page qualifier
- When you need to change the names or version numbers specified with the Snapshot or the File qualifier for the restored database

To use the storage-area-list option, specify the storage area *name*, not the system file name for the storage area. By restoring the database root only, you save the additional time normally needed to restore all the storage areas. Place commas between each storage area name in the list.

If the storage area parameters have changed since the file was last backed up, the storage-area-list option updates the .rdb file parameters so they agree with the current storage area parameters in terms of location and file version.

Command Qualifiers

Active_IO=max-reads
Specifies the maximum number of read operations to the backup file that the RMU Restore Only_Root command will attempt simultaneously. The value of the Active_IO qualifier can range from 1 to 5. The default value is 3.

After_Journal=file-spec
Noafter_Journal

Note

This qualifier is maintained for compatibility with versions of Oracle Rdb prior to Version 6.0. You might find it more useful to specify the Aij_Options qualifier, unless you are only interested in creating extensible .aij files.

1.46 RMU Restore Only_Root Command

Specifies how RMU Restore Only_Root is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the After_Journal qualifier and provide a file specification, RMU Restore Only_Root creates a new extensible .aij file and enables journaling.
- If you specify the After_Journal qualifier but you do not provide a file specification, RMU Restore Only_Root creates a new extensible .aij file with the same name as the journal that was active at the time of the backup operation.
- If you specify the Noafter_Journal qualifier, RMU Restore Only_Root disables after-image journaling and does not create a new .aij file. Note that if you specify the Noafter_Journal qualifier, there will be a gap in the sequence of .aij files. For example, suppose your database has .aij file sequence number 1 when you back it up. If you issue an RMU Restore Only_Root command with the Noafter qualifier, the .aij file sequence number will be changed to 2. This means that you cannot (and do not want to) apply the original .aij file to the restored database (doing so would result in a sequence mismatch).
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, RMU Restore Only_Root recovers the journal state (enabled or disabled) and tries to reuse the .aij file or files.

  If you choose this option, take great care to either set the database root TSN and CSN correctly, or create a full and complete backup file of the database. Failure to do so might make it impossible for you to recover your database from the .aij file should it become necessary.

  However, if the .aij file or files are not available (for example, they have been backed up), after-image journaling is disabled.

You cannot use the After_Journal qualifier to create fixed-size .aij files; use the Aij_Options qualifier.

Aij_Options=journal-opts
Noaij_Options

Specifies how RMU Restore Only_Root is to handle after-image journaling and .aij file creation, using the following rules:

- If you specify the Aij_Options qualifier and provide a journal-opts file, RMU Restore Only_Root enables journaling and creates the .aij file or files you specify for the restored database. If only one .aij file is created for the restored database, it will be an extensible .aij file. If two or more .aij files

are created for the database copy, they will be fixed-size .aij files (as long as at least two .aij files are always available).

- If you specify the Aij_Options qualifier, but do not provide a journal-opts file, RMU Restore Only_Root disables journaling and does not create any new .aij files.
- If you specify the Noaij_Options qualifier, RMU Restore Only_Root disables journaling and does not create any new .aij files.
- If you do not specify an After_Journal, Noafter_Journal, Aij_Options, or Noaij_Options qualifier, RMU Restore Only_Root recovers the journaling state (enabled or disabled) and tries to reuse the .aij file or files.

  If you choose this option, take great care to either set the database root TSN and CSN correctly, or create a full and complete backup file of the database. Failure to do so might make it impossible for you to recover your database from the .aij file should it become necessary.

  However, if the .aij file or files are not available (for example, they have been backed up), after-image journaling is disabled.

See Section 1.67.1 for information on the format of a journal-opts-file.

Assist
NoAssist
Specifies where tape handling requests are to be sent. With /NoAssist, these requests are sent to the current process' SYS$OUTPUT device and allows a command line user to respond to these requests interactively.

With /Assist, the requests are sent to an operator terminal and mount commands are issued with assistance enabled (see MOUNT/ASSIST).

The default for an interactive process (see F$MODE()) is /NoAssist and /Assist otherwise (e.g. batch job).

Attributes=options
Used to specify a number of qualifiers and use an abbreviation for the qualifier name. Normally, /THRESHOLDS, /READ_ONLY and /READ_WRITE cannot be abbreviated or an error will be returned because the abbreviation is not unique for the first four characters (an OpenVMS DCL requirement). By using the /ATTRIBUTES qualifier, THRESHOLDS can be abbreviated to the first four characters and either the option READ_ONLY or the option READ_ WRITE can be abbreviated to the first six characters. Examples of the syntax follow.

1.46 RMU Restore Only_Root Command

```
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_ONLY)
/ATTRIBUTES=(THRESHOLDS=(50,50,50),READ_WRITE)
/ATTRIBUTES=THRESHOLDS=(50,50,50)
/ATTRIBUTES=READ_ONLY
/ATTRIBUTES=READ_WRITE
```

Examples of using abbreviation with the /ATTRIBUTES qualifier options follow.

```
/ATTR=(THRE=(50,50,50),READ_O)
/ATTR=(THRE=(50,50,50),READ_W)
/ATTR=THRE=(50,50,50)
/ATTR=READ_O
/ATTR=READ_W
```

Directory=directory-spec
Specifies the default directory for the database root and the default directory for where the root can expect to find the database storage areas and snapshot files.

See the Usage Notes for information on how this qualifier interacts with the Root, File, and Snapshot qualifiers and for warnings regarding restoring database files into a directory owned by a resource identifier.

Encrypt=({Value=|Name=}[,Algorithm=])
The Encrypt qualifier decrypts the save set file of a database backup.

Specify a key value as a string or the name of a predefined key. If no algorithm name is specified, the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Initialize_Tsns
Noinitialize_Tsns
Initializes all transaction sequence number (TSN) values for the entire database by setting the values to zero. Each time a transaction is initiated against a database, a TSN is issued. The numbers are incremented sequentially over the life of the database.

TSN and CSN values are each contained in a quadword with the following decimal format:

```
high longword : low longword
```

The high longword can hold a maximum user value of 32768 (2^{15}) and the low longword can hold a maximum user value of 4,294,967,295 (2^{32}). A portion of the high longword is used by Oracle Rdb for overhead.

When you specify a TSN or CSN, you can omit the high longword and the colon if the TSN or CSN fits in the low longword. For example 0:444 and 444 are both valid input values.

As your next TSN value approaches the maximum value allowed, you should initialize the TSNs. You can determine the next TSN and next commit sequence number (CSN) values by dumping the database root file, using the RMU Dump command with the Header and Option=Debug qualifiers.

The Initialize_Tsns qualifier takes much more time to execute because all TSN values in the database are set to zero, which requires writing to every page in the database. When the database TSNs are reset, using the Initialize_ Tsns qualifier, you should use the After_Journal qualifier or the Aij_Options qualifier and immediately perform a full database backup operation and create a new .aij file. This ensures continuity of journaling and the ability to recover the database.

The default Noinitialize_Tsns qualifier does not initialize the database TSNs.

Note that you cannot use the Initialize_Tsns with the Set_Tsn or Noset_Tsn qualifier in the same command. This restriction is required because Initialize_ Tsns directs RMU Restore Only_Root to reset the TSN value to zero, while Set_Tsn directs RMU Restore Only_Root to reset the TSN to the value you have indicated, and Noset_Tsn leaves the TSN value unchanged.

CAUTION

Never use the Initialize_Tsns qualifier if the Replication Option for Rdb transfers have been defined for the database. The Initialize_Tsns qualifier does not reset the Replication Option for Rdb transfers.

Label=(label-name-list)
Specifies the 1- to 6-character string with which the volumes of the backup file have been labeled. The Label qualifier is applicable only to tape volumes. You must specify one or more label names when you use the Label qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas, and enclose the list of names within parentheses.

In a normal restore operation, the Label qualifier you specify with the RMU Restore Only_Root command should be the same Label qualifier you specified with the RMU Backup command you used to back up your database.

1.46 RMU Restore Only_Root Command

The Label qualifier can be used with indirect file references. See Section 1.3 for more information.

Librarian[=options]
Use the Librarian qualifier to restore files from data archiving software applications that support the Oracle Media Management interface. The file name specified on the command line identifies the stream of data to be retrieved from the Librarian utility. If you supply a device specification or a version number it will be ignored.

Oracle RMU supports retrieval using the Librarian qualifier only for data that has been previously stored by Oracle RMU using the Librarian qualifer.

The Librarian qualifier accepts the following options:

- Trace_file=file-specification

  The Librarian utility writes trace data to the specified file.

- Level_Trace=n

  Use this option as a debugging tool to specify the level of trace data written by the Librarian utility. You can use a pre-determined value of 0, 1, or 2, or a higher value defined by the Librarian utility. The pre-determined values are :

  - Level 0 traces all error conditions. This is the default.
  - Level 1 traces the entry and exit from each Librarian function.
  - Level 2 traces the entry and exit from each Librarian function, the value of all function parameters, and the first 32 bytes of each read/write buffer, in hexadecimal.

- Logical_Names=(logical_name=equivalence-value,...)

  You can use this option to specify a list of process logical names that the Librarian utility can use to specify catalogs or archives where Oracle Rdb backup files are stored, Librarian debug logical names, and so on. See the specific Librarian documentation for the definition of logical names. The list of process logical names is defined by Oracle RMU prior to the start of any Oracle RMU command that accesses the Librarian application.

The following OpenVMS logical names must be defined for use with a Librarian utility before you execute an Oracle RMU backup or restore operation. Do not use the Logical_Names option provided with the Librarian qualifier to define these logical names.

- RMU$LIBRARIAN_PATH

This logical name must be defined so that the shareable Librarian image can be loaded and called by Oracle RMU backup and restore operations. The translation must include the file type (for example, .exe), and must not include a version number. The shareable Librarian image must be an installed (known) image. See the Librarian implementation documentation for the name and location of this image and how it should be installed.

- RMU$DEBUG_SBT

  This logical name is not required. If it is defined, Oracle RMU will display debug tracing information messages from modules that make calls to the Librarian shareable image.

You cannot use device specific qualifiers such as Rewind, Density, or Label with the Librarian qualifier because the Librarian utility handles the storage meda, not Oracle RMU.

Log
Nolog

Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request that the progress of the restore operation be written to SYS$OUTPUT and the Nolog qualifier to suppress this report. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Media_Loader
Nomedia_Loader

Use the Media_Loader qualifier to specify that the tape device from which the backup file is being read has a loader or stacker. Use the Nomedia_Loader qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, RMU Restore Only_Root should recognize this fact. However, occasionally RMU Restore Only_Root does not recognize that a tape device has a loader or stacker. Therefore, when the first tape has been read, RMU Restore Only_Root issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes RMU Restore Only_Root behaves as though a tape device has a loader or stacker when actually it does not.

If you find that RMU Restore Only_Root is not recognizing that your tape device has a loader or stacker, specify the Media_Loader qualifier. If you find that RMU Restore Only_Root expects a loader or stacker when it should not, specify the Nomedia_Loader qualifier.

1.46 RMU Restore Only_Root Command

New_Snapshots
Nonew_Snapshots
Allows you to specify whether to create new snapshot (.snp) files as part of a Restore Only_Root operation.

The default is the Nonew_Snapshots qualifier, which causes the command to initialize the existing .snp files.

If you specify the New_Snapshots qualifier, the command creates and initializes new .snp files. When you specify the New_Snapshots qualifier, you should either delete the existing .snp files before the restore operation or purge the .snp files afterwards.

Nodes_Max=number-cluster-nodes
Specifies a new upper limit on the number of VMScluster nodes from which users can access the restored database. The Nodes_Max qualifier will accept values between 1 and 96 VMScluster nodes. The actual maximum is the highest number of VMScluster nodes possible in the current version of OpenVMS. The default value is the limit defined for the database before it was backed up.

Options=file-spec
Specifies the options file that contains storage area names, followed by the storage area qualifiers that you want applied to that storage area.

You can direct RMU Restore Only_Root to create an options file for use with this qualifier by specifying the Restore_Options qualifier with the RMU Backup, RMU Dump, and RMU Dump Backup commands. See Section 1.10, Section 1.19, and Section 1.22 for details.

If you create your own options file, *do not* separate the storage area names with commas. Instead, put each storage area name on a separate line in the file. The storage area qualifiers that you can include in the options file are: Blocks_Per_Page, File, Snapshot, and Thresholds. You can use the DCL line continuation character, a hyphen (-), or the comment character (!) in the options file. The default file extension is .opt. See Example 5 in the Examples section.

Rewind
Norewind
Specifies whether the tape that contains the backup file will be rewound before processing begins. The Norewind qualifier, the default, causes the search for the backup file to begin at the current tape position.

The Rewind and Norewind qualifiers are applicable only to tape devices. RMU Restore Only_Root returns an error message if you use these qualifiers and the device is not a tape device.

Root=root-file-spec
Requests that the database root (.rdb) be restored to the specified location.

See the Usage Notes for information on how this qualifier interacts with the Directory, File, and Snapshot qualifiers and for warnings regarding restoring database files into a directory owned by a resource identifier.

The Root qualifier is only meaningful when used with a multifile database.

Set_Tsn=(Tsn=n[,Csn=m])
Noset_Tsn
The Set_Tsn qualifier sets the database transaction sequence number (TSN) and commit sequence number (CSN) to the specified values. The correct value can be extracted from the original .rdb file if it is still accessible, or from the last .aij file if one is available. If that fails, you can use a TSN value larger than the maximum number of transactions applied to the database since it was created, or since TSNs were last initialized.

The TSN and CSN values do not have to be the same value. If the CSN option is omitted, it will default to the same value as provided to the transaction sequence number (TSN).

The TSN and CSN values need to be greater than the last values assigned to a transaction. Set_Tsn values are expected to be multiples of eight. If you specify a value that is not a multiple of eight, RMU Restore Only_Root assigns the next highest value that is a multiple of eight. (For example, if you specify Set_Tsn=(Tsn=90, Csn=90), RMU Restore Only_Root assigns the Next TSN a value of 96.)

The default value for the Set_Tsn qualifier is the TSN and CSN values stored in the backup file plus 1,000,000 when TSNs are not being initialized. The new TSN and CSN values for most database applications should be larger than the number of transactions committed since the database was last backed up. Set the TSN and CSN values higher than this default increment value plus the value in the backup file when needed. You can determine the next TSN and CSN values by dumping the .rdb file, using the Option=Debug qualifier.

The Noset_Tsn qualifier specifies that the root will be restored with the same TSN state as was recorded in the backup file.

1.46 RMU Restore Only_Root Command

When you use the Noset_Tsn qualifier in conjunction with the Noupdate_Files qualifier, you can use a backup strategy that uses recent by-area full backup files in place of a recent full and complete backup file of the entire database. See Example 6 in the Examples section.

Note that you cannot use the Initialize_Tsns with the Set_Tsn or Noset_Tsn qualifier in the same command. This restriction is required because Initialize_Tsns directs RMU Restore Only_Root to reset the TSN value to zero, while Set_Tsn directs RMU Restore Only_Root to reset the TSN to the value you have indicated, and Noset_Tsn leaves the TSN value unchanged.

Transaction_Mode=(mode-list)
Sets the allowable transaction modes for the database root file created by the restore operation. The mode-list can include one or more of the following transaction modes:

- All - Enables all transaction modes
- Current - Enables all transaction modes that are set for the source database. This is the default transaction mode.
- None - Disables all transaction modes
- [No]Batch_Update
- [No]Read_Only
- [No]Exclusive
- [No]Exclusive_Read
- [No]Exclusive_Write
- [No]Protected
- [No]Protected_Read
- [No]Protected_Write
- [No]Read_Write
- [No]Shared
- [No]Shared_Read
- [No]Shared_Write

If you specify more than one transaction mode in the mode-list, enclose the list in parenthesis and separate the transaction modes from one another with a comma. Note the following:

- When you specify a negated transaction mode, for example Noexclusive_ Write, it indicates that exclusive write is not an allowable access mode for the copied database.
- If you specify the Shared, Exclusive, or Protected transaction mode, Oracle RMU assumes you are referring to both reading and writing in that transaction mode.
- No mode is enabled unless you add that mode to the list, or you use the All option to enable all transaction modes.
- You can list one transaction mode that enables or disables a particular mode followed by another that does the opposite. For example, Transaction_Mode=(Noshared_Write, Shared) is ambiguous because the first value disables Shared_Write access and the second value enables Shared_Write access. Oracle RMU resolves the ambiguity by first enabling the modes as specified in the modes-list and then disabling the modes as specified in the modes-list. The order of items in the list is irrelevant. In the example presented previously, Shared_Read is enabled and Shared_ Write is disabled.

Update_Files
Noupdate_Files

The Update_Files qualifier specifies that the root will be restored, and RMU Restore Only_Root will attempt to link that restored root to the area files. In addition, the snapshot (.snp) file will be updated or created. This is the default.

The Noupdate_Files qualifier specifies that the restore operation will restore the root, but it will not link that restored root to any of the area files, nor will it create or update the .snp files.

When you use the Noupdate_Files qualifier in conjunction with the Noset_Tsn qualifier, you can use a backup strategy that uses recent by-area full backup files in place of a recent full and complete backup file of the entire database. See Example 6 in the Examples section.

Users_Max=number-users

Specifies a new upper limit on the number of users that can simultaneously access the restored database. The valid range is between 1 and 2032 users. The default value is the value defined for the database before it was backed up.

1.46 RMU Restore Only_Root Command

Note

Use these qualifiers to reconcile the information in the database root file with the storage area files on disk. These values can get out of synchronization when changes have been made to storage areas or snapshot files after the backup from which you are restoring the database root file was created.

Setting these parameters updates the data in the root file only; it does not change the attributes of the storage areas or snapshot files themselves.

File or Area Qualifiers

Blocks_Per_Page=integer
Noblocks_Per_Page
Updates the database root file with the number of blocks per page for the storage area. Use this qualifier to update the root when the blocks per page for a storage area has changed since the backup file from which you are restoring was created. This qualifier does not change the page size of a storage area itself; its purpose is to update the database root file with corrected information.

If you use the default, the Noblocks_Per_Page qualifier, RMU Restore Only_Root takes the page size for the storage area from the page size specified for the database you backed up. This is a positional qualifier. This qualifier conflicts with storage areas that have a uniform page format.

File=file-spec
Updates the database root file with the file specification for the storage-area-name parameter it qualifies. Use this qualifier to update the root when the file specification for a storage area has changed since the backup file from which you are restoring the root was created. (For example, if you have used the RMU Move_Area command since the backup file was created.) This qualifier does not change the file specification of the storage area it qualifies; its purpose is to update the database root file with corrected information. When you specify the File qualifier, you must supply a file name.

See the Usage Notes for information on how this qualifier interacts with the Root, Snapshot, and Directory qualifiers.

This qualifier is not valid for single-file databases. This is a positional qualifier.

Read_Only
Updates the database root file to reflect the read-only attribute for the storage area it qualifies. Use this qualifier to update the root when the read/write or read-only attribute has changed since the backup file from which you are restoring has changed. This qualifier does not change the attribute of the storage area it qualifies; its purpose is to update the database root file with corrected information.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas is restored with the read/write attributes that were in effect when the database was backed up.

Read_Write
Updates the database root file to reflect the read/write attribute for the storage area it qualifies. Use this qualifier to update the root when the read/write or read-only attribute has changed since the backup file from which you are restoring has changed. This qualifier does not change the attribute of the storage area it qualifies; its purpose is to update the database root file with corrected information.

If you do not specify the Read_Only or the Read_Write qualifier, the storage areas is restored with the read/write attributes that were in effect when the database was backed up.

Snapshot=(Allocation=n,File=file-spec)
Updates the database root file to reflect the snapshot allocation or snapshot file specification (or both) for the area it qualifies. Use this qualifier to update the root when the snapshot attributes have changed since the backup file from which you are restoring the database root has changed. This qualifier does not change the attributes of the snapshot file it qualifies; its purpose is to update the database root file with corrected information.

See the Usage Notes for information on how this qualifier interacts with the Root, Snapshot, and Directory qualifiers.

The Snapshot qualifier is a positional qualifier.

When you do not specify the Snapshot qualifier, RMU Restore Only_Root restores snapshot areas according to the information stored in the backup file.

Spams
Nospams
Updates the database root file to reflect the space area management (SPAM) information for the storage areas in the storage-area-list. Use this qualifier when the setting of SPAM pages (enabled or disabled) has changed since the backup file from which you are restoring the root was created. This qualifier

does not change the attributes of the storage area it qualifies; its purpose is to update the database root file with corrected information.

Use the Spams qualifier to update the root file information to indicate that SPAM pages are enabled for the storage areas qualified; use the Nospams qualifier to update the root file information to indicate that SPAM pages are disabled for the storage areas qualified. The default is to leave the attribute unchanged from the setting recorded in the backup file. This is a positional qualifier.

Thresholds=(val1[,val2[,val3]])
Updates the database root file to reflect the threshold information for the storage areas in the storage-area-list. Use this qualifier when the threshold values have changed since the backup file from which you are restoring the root was created. This qualifier does not change the attributes of the storage area it qualifies; its purpose is to update the database root file with corrected information.

This is a positional qualifier.

The Thresholds qualifier applies only to storage areas with a mixed page format.

If you do not use the Thresholds qualifier with the RMU Restore Only_Root command, Oracle Rdb uses the storage area's thresholds as recorded in the backup file.

See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on SPAM thresholds.

Usage Notes

- To use the RMU Restore Only_Root command for a database, you must have the RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- The RMU Restore Only_Root command provides two qualifiers, Directory, and Root, that allow you to specify the target for the restored database root file. In addition, the Directory, File, and Snapshot file qualifiers allow you to specify a target for updates to the database root for the storage area and snapshot file locations. The **target** can be just a directory, just a file name, or a directory and file name.

If you use all or some of these qualifiers, apply them as follows:

- Use the Root qualifier to indicate the target for the restored database root file.
- Use local application of the File qualifier to specify the current location of a storage area file if its location has changed since the database was backed up. The storage area is not affected by this qualifier. This qualifier updates the location of the storage area as recorded in the database root file.
- Use local application of the Snapshots qualifier to specify the current location of a snapshot file if its location has changed since the database was backed up. The snapshot file is not affected by this qualifier. This qualifier updates the location of the snapshot file as recorded in the database root file.
- Use the Directory qualifier to specify a default target directory for the root file and as a default directory for where the storage areas and snapshot files currently reside. The default target directory is where the database root file is restored if a directory specification is not specified with the Root qualifier. The default directory for the storage area and snapshot files is the directory specification with which the root file is updated if these files are not qualified with the Root, File, or Snapshot qualifier. It is also the default directory with which the Root file is updated for files qualified with the Root, File, or Snapshot qualifier if these qualifiers do not include a directory specification.

Note the following when using these qualifiers:

- Global application of the File qualifier when the target specification includes a file name causes RMU Restore Only_Root to update the file name recorded in the database root file for all storage areas to be the same file name.
- Global application of the Snapshot qualifier when the target specification includes a file name causes RMU Restore Only_Root to update the file name recorded in the database root file for all snapshot files to be the same file name.
- Specifying a file name or extension with the Directory qualifier is permitted, but causes RMU Restore Only_Root to restore the database root file to the named directory and file and update the file name recorded in the database root file for all the storage areas and snapshot files to be the same directory and file specification.

1.46 RMU Restore Only_Root Command

- When you restore a database root into a directory owned by a resource identifier, the ACE for the directory is applied to the database root file ACL first, and then the Oracle RMU ACE is added. This method is employed to prevent database users from overriding OpenVMS file security. However, this can result in a database which you consider yours, but to which you have no Oracle RMU privileges to access. See the *Oracle Rdb Guide to Database Maintenance* for details.
- Only the database parameter values and the storage area parameter values for which there are qualifiers can be updated in the database root (.rdb) file using the restore-only-root operation. All other database and storage area parameter values that have changed since the database was last backed up must be reapplied to the .rdb file using the SQL ALTER DATABASE statement.
- There are no restrictions on the use of the Nospams qualifier option with storage areas that have a mixed page format, but the use of the Nospams qualifier typically causes severe performance degradation. The Nospams qualifier is useful only where updates are rare and batched, and access is primarily by database key (dbkey).
- You must set both TSN and CSN values at the same time. You cannot set the TSN value lower than the CSN value; however, you can set a CSN value higher than the TSN value.
- The RMU Restore Only_Root command cannot be used if any storage area has been extended since the backup operation was done. You can use the RMU Dump Backup command with the Option=Root qualifier to determine if this is the case.
- If it is necessary to recover an Oracle Rdb database after restoring the database, Oracle Rdb recommends executing the recover after a full restore of the database files, not after restoring only the database root file using the RMU/RESTORE/ONLY_ROOT command.
- The RMU/RESTORE/ONLY_ROOT command is designed to recover more quickly from the loss of a database root (.rdb) file, not for routinely restoring the database and then immediately recovering the database. The RMU/RECOVER command should be executed after an RMU/RESTORE/ONLY_ROOT command only if it is not possible to do a full database restore before executing the RMU/RECOVER command.
- If it is necessary to recover a database following the RMU/RESTORE/ONLY_ROOT command, be sure that the transaction state of the database is correctly set in the database root by the RMU/RESTORE/ONLY_ROOT command. Otherwise transactions in journal files created before

the RMU/RESTORE/ONLY_ROOT command will be ignored by the RMU/RECOVER command which uses the transaction state of the database stored in the database root file to select which journaled transactions to recover from the After Image Journal (AIJ) files used by the RMU/RECOVER command. Important journaled updates to database parameters, such as the client sequence numbers maintained both in the database root (CLTSEQ) and the database system tables (Rdb$SEQUENCES), may be lost or made inconsistent.

The database should be verified both after the RMU/RESTORE/ONLY_ ROOT command completes and after the RMU/RECOVER command completes. Please consult the documentation in the Oracle Rdb RMU Reference Manual for more information and examples of the options related to AIJ files and setting the database root transaction TSN and CSN values when the Rdb database root file (.rdb) is restored using the RMU/RESTORE/ONLY_ROOT command.

- The optional /SET_TSN=(TSN=n,CSN=m) or /NOSET_TSN qualifiers can be used with the RMU/RESTORE/ONLY_ROOT command to set the database transaction sequence number (TSN) and the database commit sequence number (CSN) in the database root file created by the restore operation to values which will allow transactions journaled to AIJ files since the last database backup to be recovered by the RMU/RECOVER command. RMU/RECOVER determines which transactions to recover based on comparing transaction sequence numbers (TSNs) in AIJ files with the last committed transaction TSN numbers stored in the database root file.
- The values for setting the database transaction sequence number (TSN) and the database commit sequence number (CSN) in the database root file which must to be specified by the optional /SET_TSN=(TSN=n,CSN=m) qualifier can be extracted from an RMU/DUMP/HEADER of the original database root file before the only root restore, or an RMU/DUMP/AFTER_ JOURNAL of the last AIJ file used before the only root restore.
- The /NOSET_TSN qualifier can be used with the RMU/RESTORE/ONLY_ ROOT command to allow transactions journaled to AIJ files since the last database backup to be recovered by the RMU/RECOVER command. The /NOSET_TSN qualifier does not change either of the TSN or CSN values but sets the same TSN and CSN root values for the restored database that are in the backup file specified by the RMU/RESTORE/ONLY_ROOT command.

If neither the /SET_TSN nor /NOSET_TSN qualifiers are used with the RMU/RESTORE/ONLY_ROOT command, the default is to set the root TSN and CSN values of the restored database root to the TSN and CSN values in the backup file plus 1,000,000. This ensures that the TSN and CSN values will not be set low enough to corrupt the database. However, if an RMU/RECOVER command is then executed for the database, transactions in the recovered AIJ files with lower TSN values will be ignored by RMU/RECOVER.

- The TSN and CSN values set in the database root restored by the RMU/RESTORE/ONLY_ROOT command are displayed by an informational message if logging is enabled.

```
%RMU-I-SETRTTSNCSN, Setting Root Transaction Sequence TSN to 384,
 Commit Sequence CSN to 384
```

Examples

Example 1

To prevent corruption of your databases, check your CSN and TSN values and set them to zero based on when they approach the maximum. First, enter an RMU Dump command to display the next CSN and next TSN values:

```
$ RMU/DUMP/HEADER=(SEQUENCE_NUMBERS) MF_PERSONNEL
   .
   .
   .
    Sequence Numbers...
      - Transaction sequence number
        Next number is 0:256
        Group size is 0:32
      - Commit sequence number
        Next number is 0:256
        Group size is 0:32
```

If the next CSN and the next TSN values are approaching the maximum number allowed, you must perform the following operations to initialize all TSN and CSN values to the value zero in your database. The operation might take some time to execute as it writes to every page in the database.

First, create a backup file for the database. Then restore the database and initialize the CSN and TSN values with the Initialize_Tsns qualifier. Then, enter an RMU Dump command again to examine the next CSN and next TSN values. This example shows that both values have been set to zero. If you displayed the database pages, you would also notice that all TSN and CSN values are set to zero.

```
$ RMU/BACKUP MF_PERSONNEL MF_PER_124.RBF
$ RMU/RESTORE/ONLY_ROOT /INITIALIZE_TSNS MF_PER_124.RBF
$ RMU/DUMP/HEADER=(SEQUENCE_NUMBERS) MF_PERSONNEL
   .
   .
   .
    Sequence Numbers...
      - Transaction sequence number
        Next number is 0:0
        Group size is 0:32
      - Commit sequence number
        Next number is 0:0
        Group size is 0:32
```

Example 2

Perform the following to set the TSN and CSN values to a number that you select; a number that is greater than or equal to the next CSN and next TSN values. If the number you have selected is less than the next CSN and next TSN values recorded in the database header, you receive an error as follows:

```
$ RMU/RESTORE/ONLY_ROOT/SET_TSN=(TSN=40,CSN=40)
_$ MF_PERSONNEL.RBF
%RMU-F-TSNLSSMIN, value (0:40) is less than minimum
 allowed value (0:224) for /SET_TSN=TSN
%RMU-F-FTL_RSTR, Fatal error for RESTORE operation
 at 18-JUN-1997 16:59:19.32
```

Enter a number equal to or greater than the next CSN and next TSN values recorded in the database header:

```
$ RMU/RESTORE/ONLY_ROOT/SET_TSN=(TSN=274,CSN=274) -
_$ MF_PERSONNEL.RBF
```

Enter an RMU Dump command to see the next CSN and next TSN values:

```
$ RMU/DUMP/HEADER=(SEQUENCE_NUMBERS) MF_PERSONNEL
   .
   .
   .
    Sequence Numbers...
      - Transaction sequence number
        Next number is 0:288
        Group size is 0:32
      - Commit sequence number
        Next number is 0:288
        Group size is 0:32
      - Database bind sequence number
        Next number is 0:288
```

Example 3

1.46 RMU Restore Only_Root Command

The following RMU Restore Only_Root command restores the database root file from the database backup file (.rbf) to another device:

```
$ RMU/RESTORE/ONLY_ROOT/ROOT=DXXV9:[BIGLER.TESTING]MF_PERSONNEL -
_$ MF_PERSONNEL_BACKUP.RBF
```

The following DIRECTORY command confirms that the MF_PERSONNEL.RDB file was restored in the specified directory:

```
$ DIRECTORY DXXV9:[BIGLER.TESTING]MF_PERSONNEL.RDB

Directory DXXV9:[BIGLER.TESTING]

MF_PERSONNEL.RDB;1   21-JAN-1991 14:37:36.87

Total of 1 file.
```

Example 4

Use the File=file-spec qualifier to update the .rdb file with a storage area's new location. If you have moved a storage area to a new location, use the File qualifier to show its new location and the Snapshot qualifier to indicate the current version of the area's snapshot (.snp) file. Enter the following RMU commands to execute a series of operations that use the File and Snapshot qualifiers in a restore-only-root operation to update the .rdb file with new information since the database was last backed up.

Back up the database file:

```
$ RMU/BACKUP MF_PERSONNEL MFPERS_122.RBF.
```

Move the area to another directory:

```
$ RMU/MOVE_AREA MF_PERSONNEL JOBS -
_$ /FILE=[BIGLER.MFTEST.TEST1]JOBS.RDA
```

With the RMU Restore Only_Root command, give the area name, and specify both the storage area file specification and its new location. Also specify the snapshot (.snp) file with its correct version. Note that .snp file version numbers increment with the RMU Move_Area command.

```
$ RMU/RESTORE/ONLY_ROOT MFPERS_122.RBF JOBS -
_$ /FILE=[BIGLER.MFTEST.TEST1]JOBS.RDA -
_$ /SNAPSHOT=(FILE=[BIGLER.V41MFTEST]JOBS.SNP;2)
```

Display the .rdb file header and note that the file is correctly updated.

The dump of the database root file lists these file specifications:

```
$ RMU/DUMP/HEADER MF_PERSONNEL
DXXV9:[BIGLER.MFTEST.TEST1]JOBS.RDA;1
DXXV9:[BIGLER.MFTEST]JOBS.SNP;2
```

Verify the .rdb file to be certain that it has been properly and completely updated relative to the files and their version numbers that comprise the database.

```
$ RMU/VERIFY/ROOT MF_PERSONNEL
```

Example 5

The following command achieves the same results as the RMU Restore Only_ Root command in Example 4, but uses an options file to specify the current location of the JOBS storage area and the associated .snp file.

1.46 RMU Restore Only_Root Command

```
$ RMU/RESTORE/ONLY_ROOT MFPERS_122.RBF -
_$ JOBS/OPTIONS=OPTIONS_FILE.OPT
$ !
$ TYPE OPTIONS_FILE.OPT
JOBS /FILE=[BIGLER.V41MFTEST.TEST1]JOBS.RDA -
     /SNAPSHOT=(FILE=BIGLER.V41MFTEST]JOBS.SNP)
```

Example 6

The following example demonstrates the use of the Noset_Tsn qualifier and the Noupdate_Files qualifier to restore a database using by-area backup files. In addition, it demonstrates the automatic recovery feature of the RMU Restore command.

```
$ !
$ SET DEFAULT DISK1:[USER]
$ !
$ ! Create .aij files for the database. Because three .aij files are
$ ! created, fixed-size after-image journaling will be used.
$ !
$ RMU/SET AFTER_JOURNAL/ENABLE/RESERVE=4     -
_$ /ADD=(name=AIJ1, FILE=DISK2:[CORP]AIJ_ONE)   -
_$ /ADD=(name=AIJ2, FILE=DISK2:[CORP]AIJ_TWO)   -
_$ /ADD=(NAME=AIJ3, FILE=DISK2:[CORP]AIJ_THREE) -
_$ MF_PERSONNEL
%RMU-W-DOFULLBCK, full database backup should be done to
 ensure future recovery
$ !
$ !
$ ! For the purposes of this example, assume the backup operation
$ ! recommended in the preceding warning message is done, but
$ ! that the time between this backup operation and the following
$ ! operations is several months so that this backup file is too
$ ! old to use in an efficient restore operation.
$ !
$ ! Update the DEPARTMENTS table.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> --
SQL> -- On Monday, insert a new row in the DEPARTMENTS table. The
SQL> -- new row is stored in the DEPARTMENTS storage area.
SQL> --
SQL> INSERT INTO DEPARTMENTS
cont>   (DEPARTMENT_CODE, DEPARTMENT_NAME, MANAGER_ID,
cont>   BUDGET_PROJECTED, BUDGET_ACTUAL)
cont>   VALUES ('WLNS', 'Wellness Center', '00188', 0, 0);
1 row inserted
SQL>
```

```
SQL> COMMIT;
SQL> DISCONNECT DEFAULT;
SQL> EXIT
$ !
$ ! Perform a by-area backup operation, including half of the
$ ! storage areas from the mf_personnel database.
$ !
$ RMU/BACKUP/INCLUDE=(RDB$SYSTEM, EMPIDS_LOW, EMPIDS_MID, -
_$ EMPIDS_OVER, DEPARTMENTS) MF_PERSONNEL -
_$ DISK3:[BACKUP]MONDAY_FULL.RBF
%RMU-I-NOTALLARE, Not all areas will be included in
 this backup file
$ !
$ ! Update the SALARY_HISTORY table.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> --
SQL> -- On Tuesday, one row is updated in the
SQL> -- SALARY_HISTORY storage area.
SQL> --
SQL> UPDATE SALARY_HISTORY
cont>    SET SALARY_END ='20-JUL-1993 00:00:00.00'
cont>    WHERE SALARY_START='14-JAN-1983 00:00:00.00'
cont>    AND EMPLOYEE_ID = '00164';
1 row updated
SQL> COMMIT;
SQL> DISCONNECT DEFAULT;
SQL> EXIT
$ !
$ ! On Tuesday, back up the other half of the storage areas.
$ !
$ RMU/BACKUP/INCLUDE=(SALARY_HISTORY, JOBS, EMP_INFO, -
_$ MF_PERS_SEGSTR, RDB$SYSTEM) MF_PERSONNEL -
_$ DISK3:[BACKUP]TUESDAY_FULL.RBF
%RMU-I-NOTALLARE, Not all areas will be included in this
 backup file
$ !
$ ! On Wednesday, perform additional updates.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> --
SQL> -- Update another row in the SALARY_HISTORY table:
SQL>  UPDATE SALARY_HISTORY
cont>     SET SALARY_START ='23-SEP-1991 00:00:00.00'
cont>     WHERE SALARY_START='21-SEP-1981 00:00:00.00'
cont>     AND EMPLOYEE_ID = '00164';
1 row updated
SQL> COMMIT;
SQL> DISCONNECT DEFAULT;
SQL> EXIT
```

1.46 RMU Restore Only_Root Command

```
$ !
$ ! Assume the database is lost on Wednesday.
$ !
$ ! Restore the database root from the latest full-area backup file.
$ !
$ RMU/RESTORE/ONLY_ROOT/NOUPDATE_FILES/NOSET_TSN -
_$ DISK3:[BACKUP]TUESDAY_FULL.RBF/LOG
%RMU-I-AIJRSTBEG, restoring after-image journal "state" information
%RMU-I-AIJRSTJRN, restoring journal "AIJ1" information
%RMU-I-AIJRSTSEQ, journal sequence number is "0"
%RMU-I-AIJRSTSUC, journal "AIJ1" successfully restored from
 file "DISK2:[CORP]AIJ_ONE.AIJ;1"
%RMU-I-AIJRSTJRN, restoring journal "AIJ2" information
%RMU-I-AIJRSTNMD, journal has not yet been modified
%RMU-I-AIJRSTSUC, journal "AIJ2" successfully restored from
 file "DISK2:[CORP]AIJ_TWO.AIJ;1"
%RMU-I-AIJRSTJRN, restoring journal "AIJ3" information
%RMU-I-AIJRSTNMD, journal has not yet been modified
%RMU-I-AIJRSTSUC, journal "AIJ3" successfully restored from
 file "DISK2:[CORP]AIJ_THREE.AIJ;1"
%RMU-I-AIJRSTEND, after-image journal "state" restoration complete
%RMU-I-RESTXT_00, Restored root file
 DISK1:[USER]MF_PERSONNEL.RDB;1
%RMU-I-AIJRECBEG, recovering after-image journal "state" information
%RMU-I-AIJRSTAVL, 3 after-image journals available for use
%RMU-I-AIJRSTMOD, 1 after-image journal marked as "modified"
%RMU-I-LOGMODSTR,     activated after-image journal "AIJ2"
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to
 ensure future recovery
%RMU-I-AIJRECEND, after-image journal "state" recovery complete
```

```
$ !
$ ! Restore the database areas, starting with the most recent
$ ! full-area backup file. (If the RDB$SYSTEM area is not in the
$ ! most recent full-area backup file, however, it must be restored
$ ! first.) Do not restore any area more than once.
$ !
$ ! Specify the Norecovery qualifier since there are additional
$ ! backup files to apply.
$ !
$ RMU/RESTORE/AREA/NOCDD/NORECOVER -
_$ DISK3:[BACKUP]TUESDAY_FULL.RBF -
_$ RDB$SYSTEM, SALARY_HISTORY, JOBS, -
_$ EMP_INFO, MF_PERS_SEGSTR/LOG
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]MF_PERS_DEFAULT.RDA;1 at 18-JUN-1997 16:14:40.88
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]SALARY_HISTORY.RDA;1 at 18-JUN-1997 16:14:41.28
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]JOBS.RDA;1 at 18-JUN-1997 16:14:41.83
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]EMP_INFO.RDA;1 at 18-JUN-1997 16:14:42.06
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]MF_PERS_SEGSTR.RDA;1 at 18-JUN-1997 16:14:42.27
%RMU-I-RESTXT_24, Completed full restore of storage area
 DISK1:[USER]JOBS.RDA;1 at 18-JUN-1997 16:14:42.49
%RMU-I-RESTXT_24, Completed full restore of storage area
 DISK1:[USER]EMP_INFO.RDA;1 at 18-JUN-1997 16:14:42.74
   .
   .
   .
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]MF_PERS_DEFAULT.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page
 is 2 blocks long
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]EMP_INFO.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page
 is 2 blocks long
   .
   .
   .
%RMU-I-AIJWASON, AIJ journaling was active when
 the database was backed up
%RMU-I-AIJRECFUL, Recovery of the entire database
 starts with AIJ file sequence 0
%RMU-I-COMPLETED, RESTORE operation completed
 at 18-JUN-1997 16:14:46.82
```

1.46 RMU Restore Only_Root Command

```
$ !
$ ! Complete restoring database areas by applying the most
$ ! recent full-area backup file.  However, do not include
$ ! the RDB$SYSTEM table because that was already restored
$ ! in the previous restore operation.  This restore
$ ! operation will attempt an automatic recovery of the .aij files.
$ !
$ RMU/RESTORE/AREA/NOCDD DISK3:[BACKUP]MONDAY_FULL.RBF -
_$ EMPIDS_LOW, EMPIDS_MID, EMPIDS_OVER, DEPARTMENTS/LOG
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]EMPIDS_OVER.RDA;1 at 18-JUN-1997 16:20:05.08
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]EMPIDS_MID.RDA;1 at 18-JUN-1997 16:20:05.40
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]EMPIDS_LOW.RDA;1 at 18-JUN-1997 16:20:05.91
%RMU-I-RESTXT_21, Starting full restore of storage area
 DISK1:[USER]DEPARTMENTS.RDA;1 at 18-JUN-1997 16:20:06.01
%RMU-I-RESTXT_24, Completed full restore of storage area
 DISK1:[USER]EMPIDS_OVER.RDA;1 at 18-JUN-1997 16:20:06.24
   .
   .
   .
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]DEPARTMENTS.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page
 is 2 blocks long
%RMU-I-RESTXT_01, Initialized snapshot file
 DISK1:[USER]EMPIDS_LOW.SNP;1
%RMU-I-LOGINIFIL,     contains 100 pages, each page
 is 2 blocks long
   .
   .
   .
%RMU-I-AIJWASON, AIJ journaling was active when
 the database was backed up
%RMU-I-AIJRECFUL, Recovery of the entire database
 starts with AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area DEPARTMENTS starts
 with AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area EMPIDS_LOW starts
 with AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area EMPIDS_MID starts
 with AIJ file sequence 0
%RMU-I-AIJRECARE, Recovery of area EMPIDS_OVER starts
 with AIJ file sequence 0
%RMU-I-AIJBADAREA, inconsistent storage area
 DISK1:[USER]DEPARTMENTS.RDA;1 needs AIJ sequence number 0
%RMU-I-AIJBADAREA, inconsistent storage area
 DISK1:[USER]EMPIDS_LOW.RDA;1 needs AIJ sequence number 0
   .
   .
   .
```

```
%RMU-I-LOGRECDB, recovering database file
 DISK1:[USER]MF_PERSONNEL.RDB;1
%RMU-I-AIJAUTOREC, starting automatic after-image
 journal recovery
%RMU-I-LOGOPNAIJ, opened journal file DISK2:[CORP]AIJ_ONE.AIJ;1
%RMU-I-AIJONEDONE, AIJ file sequence 0 roll-forward
 operations completed
%RMU-I-LOGRECOVR, 1 transaction committed
%RMU-I-LOGRECOVR, 0 transactions rolled back
%RMU-I-LOGRECOVR, 2 transactions ignored
%RMU-I-AIJNOACTIVE, there are no active transactions
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJALLDONE, after-image journal roll-forward
 operations completed
%RMU-I-LOGSUMMARY, total 1 transaction committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
%RMU-I-LOGSUMMARY, total 2 transactions ignored
%RMU-I-AIJSUCCES, database recovery completed successfully
%RMU-I-AIJGOODAREA, storage area
 DISK1:[USER]DEPARTMENTS.RDA;1 is now consistent
%RMU-I-AIJGOODAREA, storage area
 DISK1:[USER]EMPIDS_LOW.RDA;1 is now consistent
%RMU-I-AIJGOODAREA, storage area
 DISK1:[USER]EMPIDS_MID.RDA;1 is now consistent
   .
   .
   .
%RMU-I-AIJFNLSEQ, to start another AIJ file recovery,
 the sequence number needed  will be 0
%RMU-I-COMPLETED, RESTORE operation completed at
 18-JUN-1997 16:20:11.45
$ !
$ ! The database is now restored and recovered.  However, if
$ ! for some reason the automatic .aij file recovery was not
$ ! possible (for example, if you had backed up the .aij files),
$ ! apply the .aij files in the same order in
$ ! which they were created.  That is, if .aij files were backed
$ ! up each night, apply aij_mon.aij first and aij_tues.aij second.
```

Example 7

The following example demonstrates the use of the Directory, File, and Root qualifiers. First, the database is backed up, then a couple storage area files and a snapshot file are moved. The restore-only-root operation does the following:

- The default directory is specified as DISK2:[DIR].
- The target directory and file name for the database root file is specified with the Root qualifier. The target directory specified with the Root qualifier overrides the default directory specified with the Directory

qualifier. Thus, the RMU Restore Only_Root process restores the database root in DISK3:[ROOT] and names it COPYRDB.RDB.

- The target directory for the EMPIDS_MID storage area is DISK4:[FILE]. The RMU Restore Only_Root process updates the database root file to indicate that EMPIDS_MID currently resides in DISK4:[FILE].
- The target for the EMPIDS_MID snapshot file is DISK5:[SNAP]EMPIDS_ MID.SNP Thus, the RMU Restore Only_Root process updates the database root file to indicate that the EMPIDS_MID snapshot file currently resides in DISK5:[SNAP]EMPIDS_MID.SNP.
- The target file name for the EMPIDS_LOW storage area is EMPIDS. Thus, the RMU Restore Only_Root process updates the database root file to indicate that the EMPIDS_LOW storage area currently resides in the DISK2 default directory (specified with the Directory qualifier), and the file is currently named EMPIDS.RDA.
- The target for the EMPIDS_LOW snapshot file is DISK5:[SNAP]EMPIDS.SNP. Thus, the RMU Restore Only_Root process updates the database root file to indicate that the EMPIDS_LOW snapshot file currently resides in DISK5:[SNAP]EMPIDS.SNP.
- Data for all the other storage area files and snapshot files remain unchanged in the database root file.

```
$ ! Back up the database:
$ !
$ RMU/BACKUP MF_PERSONNEL.RDB MF_PERSONNEL.RBF
$ !
$ ! Move a couple of storage areas and a snapshot file:
$ !
$ RMU/MOVE_AREA MF_PERSONNEL.RDB -
_$ /DIRECTORY=DISK2:[DIR] -
_$ EMPIDS_MID/FILE=DISK4:[FILE] -
_$ /SNAPSHOT=(FILE=DISK3:[SNAP]EMPIDS_MID.SNP), -
_$ EMPIDS_LOW/FILE=EMPIDS -
_$ /SNAPSHOT=(FILE=DISK5:[SNAP]EMPIDS.SNP)
$ !
$ ! Database root is lost.  Restore the root and update the
$ ! locations of the moved storage areas and snapshot file as
$ ! recorded in the database root file because the locations
$ ! recorded in the backup file from which the root is restored
$ ! are not up-to-date:
$ !
```

```
$ RMU/RESTORE/ONLY_ROOT MF_PERSONNEL.RBF -
_$ /ROOT=DISK3:[ROOT]MF_PERSONNEL.RDB -
_$ EMPIDS_MID/FILE=DISK4:[FILE] -
_$ /SNAPSHOT=(FILE=DISK2:[DIR]EMPIDS_MID.SNP), -
_$ EMPIDS_LOW/FILE=DISK2:[DIR]EMPIDS -
_$ /SNAPSHOT=(FILE=DISK5:[SNAP]EMPIDS.SNP)
```

Example 8

This example shows how to use the /ENCRYPT qualifier with the RMU/RESTORE/ONLY_ROOT command. If you forget to use the /ENCRYPT command on a database backup that has been encrypted, an error message will output.

```
$ RMU/BACKUP/ENCRYPT=(VALUE="My secret key",ALGORITHM=DESCBC)/NOLOG -
 MF_PERSONNEL.RDB MF_PERSONNEL_BCK.RBF
%RMU-I-ENCRYPTUSED, Encryption key required when future restore performed.
$ SQL
DROP DATABASE FILENAME MF_PERSONNEL;
EXIT;
$ RMU/RESTORE/ENCRYPT=(VALUE="My secret key",ALGORITHM=DESCBC)/NOCDD/NOLOG
MF_PERSONNEL_BCK
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not available.
$ RMU/VERIFY/NOLOG MF_PERSONNEL
$ DELETE MF_PERSONNEL.RDB;*
$ RMU/RESTORE/ONLY_ROOT/NOLOG MF_PERSONNEL_BCK
%RMU-F-ENCRYPTSAVSET, save set is encrypted, /ENCRYPT must be specified
%RMU-F-FATALERR, fatal error on RESTORE_ROOT_ONLY
%RMU-F-FTL_RSTR, Fatal error for RESTORE operation at 21-JUN-2012 09:45:52.41
$ RMU/RESTORE/ONLY_ROOT/ENCRYPT=(VALUE="My secret key",ALGORITHM=DESCBC)/NOLOG
MF_PERSONNEL_BCK.RBF
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
$ RMU/VERIFY/NOLOG MF_PERSONNEL
```

Example 9

The following example shows the RMU/RESTORE, RMU/MOVE_AREA, and RMU/COPY_DATABASE commands used with the /ATTRIBUTES qualifier (an RMU/RESTORE/ONLY_ROOT example is not shown but this qualifier can be used with this command).

1.46 RMU Restore Only_Root Command

```
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTRIBUTES=(THRE=(50,50,50),READ_O)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/MOVE/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTRIBUTES=READ_O
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG/THREADS=5 DISK:[DIRECTORY]MF_PERSONNEL
JOBS/ATTR=THRES=(50,50,50)
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/COPY/DIR=DISK:[DIRECTORY]/NOLOG DISK:[DIRECTORY]MF_PERSONNEL
JOBS/BLOCK=4/ATTR=READ_W
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
$ RMU/RESTORE/NOLOG/NOCDD/DIR=DISK:[DIRECTORY] DISK:[DIRECTORY]MFP.RBF
JOBS/ATTR=READ_O
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not available.
```

Example 10 Using RMU/RESTORE/ONLY_ROOT command with /SET_TSN

In the example below, the TSN and CSN database root values of the restored root file are specified by the /SET_TSN qualifier. As a result, the original journaled client sequence value of "21" is recovered by the RMU/RECOVER command executed after the RMU/RESTORE/ONLY_ROOT command.

```
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                     21
1 row selected
exit;
$ delete mf_personnel.rdb;*
$
$ rmu/restore/only_root/log/SET_TSN=(tsn=384,csn=384) mf_personnel
%RMU-I-AIJRSTBEG, restoring after-image journal "state" information
%RMU-I-AIJRSTJRN, restoring journal "AIJ1" information
%RMU-I-AIJRSTSEQ, journal sequence number is "0"
%RMU-I-AIJRSTSUC, journal "AIJ1" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_ONE.AIJ;1"
%RMU-I-AIJRSTJRN, restoring journal "AIJ2" information
%RMU-I-AIJRSTNMD, journal has not yet been modified
%RMU-I-AIJRSTSUC, journal "AIJ2" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_TWO.AIJ;1"
%RMU-I-AIJRSTEND, after-image journal "state" restoration complete
%RMU-I-SETRTTSNCSN, Setting Root Transaction Sequence TSN to 384,
 Commit Sequence CSN to 384
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure
 future recovery
%RMU-I-AIJRECEND, after-image journal "state" recovery complete
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                      1
1 row selected
exit;
$ rmu/recover/out=recov.sav AIJ_ONE.aij,AIJ_TWO.aij
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                     21
1 row selected
exit;
$
```

Example 11 Using RMU/RESTORE/ONLY_ROOT command with /NOSET_TSN

1.46 RMU Restore Only_Root Command

In this example, the /NOSET_TSN qualifier is used so that the TSN and CSN values of the restored root file are set to the values in the backup file used by the RMU/RESTORE/ONLY_ROOT command. As a result, the original journaled client sequence value of "21" is recovered by the RMU/RECOVER command executed after the RMU/RESTORE/ONLY_ROOT command.

```
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                     21
1 row selected
exit;
$ delete mf_personnel.rdb;*
$ rmu/restore/only_root/log/NOSET_TSN mf_personnel
%RMU-I-AIJRSTBEG, restoring after-image journal "state" information
%RMU-I-AIJRSTJRN, restoring journal "AIJ1" information
%RMU-I-AIJRSTSEQ, journal sequence number is "0"
%RMU-I-AIJRSTSUC, journal "AIJ1" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_ONE.AIJ;1"
%RMU-I-AIJRSTJRN, restoring journal "AIJ2" information
%RMU-I-AIJRSTNMD, journal has not yet been modified
%RMU-I-AIJRSTSUC, journal "AIJ2" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_TWO.AIJ;1"
%RMU-I-AIJRSTEND, after-image journal "state" restoration complete
%RMU-I-SETRTTSNCSN, Setting Root Transaction Sequence TSN to 384,
 Commit Sequence CSN to 384
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure
 future recovery
%RMU-I-AIJRECEND, after-image journal "state" recovery complete
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                      1
1 row selected
exit;
$ rmu/recover/out=recov.sav  AIJ_ONE.aij,AIJ_TWO.aij
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                     21
1 row selected
exit;
```

```
$
```

Example 12 Using RMU/RESTORE/ONLY_ROOT command with neither /SET_TSN nor /NOSET_TSN

In this example, where the default is taken, the TSN and CSN database root values of the restored root file are set to the values in the backup file used by the RMU/RESTORE/ONLY_ROOT command added to the value 1,000,000. As a result, the original journaled client sequences value of "21" is not recovered by the RMU/RECOVER command executed after the RMU/RESTORE/ONLY_ ROOT command since the transaction TSN number recorded in the journal file does not match the transaction TSN state set by the RMU/RESTORE/ONLY_ ROOT command.

```
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                     21
1 row selected
exit;
$ delete mf_personnel.rdb;*
$
$ rmu/restore/only_root/log mf_personnel
%RMU-I-AIJRSTBEG, restoring after-image journal "state" information
%RMU-I-AIJRSTJRN, restoring journal "AIJ1" information
%RMU-I-AIJRSTSEQ, journal sequence number is "0"
%RMU-I-AIJRSTSUC, journal "AIJ1" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_ONE.AIJ;1"
%RMU-I-AIJRSTJRN, restoring journal "AIJ2" information
%RMU-I-AIJRSTNMD, journal has not yet been modified
%RMU-I-AIJRSTSUC, journal "AIJ2" successfully restored from file
 "DEVICE:[DIRECTORY]AIJ_TWO.AIJ;1"
%RMU-I-AIJRSTEND, after-image journal "state" restoration complete
%RMU-I-SETRTTSNCSN, Setting Root Transaction Sequence TSN to 1000384,
 Commit Sequence CSN to 1000384
%RMU-I-AIJISON, after-image journaling has been enabled
%RMU-W-DOFULLBCK, full database backup should be done to ensure
 future recovery
%RMU-I-AIJRECEND, after-image journal "state" recovery complete
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                      1
1 row selected
exit;
$ rmu/recover/out=recov.sav  AIJ_ONE.aij,AIJ_TWO.aij
```

1.46 RMU Restore Only_Root Command

```
$
$ sql$
SQL> attach 'file mf_personnel';
SQL> select RDB$SEQUENCE_NAME, RDB$NEXT_SEQUENCE_VALUE
cont> from rdb$sequences;
 RDB$SEQUENCE_NAME                RDB$NEXT_SEQUENCE_VALUE
 S                                                      1
1 row selected
exit;
$
```

1.47 RMU Server After_Journal Reopen_Output Command

Allows you to close the current AIJ log server (ALS) output file for the specified database and open a new one. This allows you to see the current contents of the original ALS output file.

Format

RMU/Server After_Journal Reopen_Output root-file-spec

Description

The RMU Server After_Journal Reopen_Output command allows you to reopen an ALS output file that was previously created with an RMU Server After_ Journal Start command with the Output qualifier. (The ALS output file is opened for exclusive access by the ALS process.)

Reopening the output file results in the current output file being closed and a new output file being created. The new output file has the same file name as the original output file, but its version number is incremented by one.

The ALS is an optional process that flushes log data to the after-image journal (.aij) file. All database servers deposit transaction log data in a cache located in the database global section. If the ALS is active, it continuously flushes the log data to disk. Otherwise, server processes might block temporarily if the cache in the global section is full.

Parameters

root-file-spec
Specifies the database root file for which you want to reopen the ALS output file.

Usage Notes

- To use the RMU Server After_Journal Reopen_Output command for a database, you must have RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.

1.47 RMU Server After_Journal Reopen_Output Command

- To issue the RMU Server After_Journal Reopen_Output command successfully, the database must be opened. Other users can be attached to the database when this command is issued.
- To determine whether the ALS is running, use the RMU Show Users command.

Examples

Example 1

In the following example the first Oracle RMU command starts the log server and specifies an output file. The second Oracle RMU command reopens the ALS output file, so you can view the data that is contained in the ALS output file so far.

```
$ RMU/SERVER AFTER_JOURNAL START MF_PERSONNEL/OUT=ALS
$ ! Database updates occur
$ RMU/SERVER AFTER_JOURNAL REOPEN_OUTPUT MF_PERSONNEL
$ ! View the ALS.OUT;-1 file:
$ TYPE ALS.OUT;-1
------------------------------------------------------------------------

16-OCT-1995 13:02:05.21 - Oracle Rdb V7.0-00 database utility started

------------------------------------------------------------------------
   .
   .
   .
```

1.48 RMU Server After_Journal Start Command

Allows you to manually start the AIJ log server (ALS) for the specified database and specify a file for the AIJ log server output.

Format

RMU/Server After_Journal Start root-file-spec

Command Qualifier	Default
/Output=file-spec	See description

Description

The ALS is an optional process that flushes log data to the after-image journal (.aij) file. All database servers deposit transaction log data in a cache located in the database global section. If the ALS is active, it continuously flushes the log data to disk. Otherwise, server processes might block temporarily if the cache in the global section is full. The ALS should be started only when AIJ processing is a bottleneck. Typically, multiuser databases with medium to high update activity can benefit from using the ALS.

You can start the ALS either manually, using the RMU Server After_Journal Start command, or automatically when the database is opened (by specifying LOG SERVER IS AUTOMATIC in the SQL ALTER DATABASE command). By default, the ALS startup is set to manual.

Command Parameters

root-file-spec
Specifies the database root file for which you want to start the ALS.

Command Qualifiers

Output=file-spec
Specifies the file for the ALS output file. Use this qualifier in anticipation of issuing an RMU Server After_Journal Reopen_Output command. By specifying the output file, you will know the location of, and therefore can view, the ALS output file.

By default, the ALS output file is not available to the user.

1.48 RMU Server After_Journal Start Command

Usage Notes

- To use the RMU Server After_Journal Start command for a database, you must have RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- The ALS can be started only if the database is open and if after-image journaling is enabled.
- The RMU Server After_Journal Start command can be issued while users are attached to the database.
- If the ALS process stops abnormally, regardless of whether the current setting of the ALS is automatic or manual, the only way to restart it is to use the RMU Server After_Journal Start command.
- To determine whether the ALS is running, use the RMU Show Users command.
- Any errors encountered when you try to start the ALS are logged in the monitor log file. Use the RMU Show System command to find the location of the monitor log file.

Examples

Example 1

The following Oracle RMU command starts the log server.

```
$ RMU/SERVER AFTER_JOURNAL START MF_PERSONNEL
```

1.49 RMU Server After_Journal Stop Command

Allows you to manually stop the AIJ log server (ALS) for the specified database.

Format

RMU/Server After_Journal Stop root-file-spec

Command Qualifiers	Defaults
/Output=file-name	See description

Description

The ALS is an optional process that flushes log data to the after-image journal (.aij) file. All database servers deposit transaction log data in a cache located in the database global section. If the ALS is active, it continuously flushes the log data to disk. Otherwise, server processes might block temporarily if the cache in the global section is full.

Command Parameters

root-file-spec
Specifies the database root file for which you want to stop the ALS.

Command Qualifiers

Output=file-name
Allows you to specify the file where the operational log is to be created. The operational log records the transmission and receipt of network messages.

If you do not include a directory specification with the file name, the log file is created in the database root file directory. It is invalid to include a node name as part of the file name specification.

Note that all Hot Standby bugcheck dumps are written to the corresponding bugcheck dump file; bugcheck dumps are not written to the file you specify with the Output qualifier.

1.49 RMU Server After_Journal Stop Command

Usage Notes

- To use the RMU Server After_Journal Stop command for a database, you must have RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- To issue the RMU Server After_Journal Stop command sucessfully, the database must be open. Other users can be attached to the database.
- If the ALS process stops abnormally, regardless of whether the current setting of the ALS is automatic or manual, the only way to restart it is to use the RMU Server After_Journal Start command.
- To determine whether the ALS is running, use the RMU Show Users command.
- If database replication is active and you attempt to stop the database AIJ log server, Oracle Rdb returns an error. You must stop database replication before attempting to stop the server.

Examples

Example 1

The following example stops the log server.

```
$ RMU/SERVER AFTER_JOURNAL STOP MF_PERSONNEL
```

1.50 RMU Server Backup_Journal Resume Command

Allows you to reinstate the ability to perform AIJ backup operations after they have been manually suspended with the RMU Server Backup_Journal Suspend command.

Format

RMU/Server Backup_Journal Resume root-file-spec

Command Qualifiers	Defaults
/[No]Log	Current DCL verify value

Description

When you issue the RMU Server Backup_Journal Suspend command, after-image journal (AIJ) backup operations are temporarily suspended. Use the RMU Server Backup_Journal Resume command to reinstate the ability to backup .aij files.

The RMU Server Backup_Journal Resume command must be issued from the same node from which AIJ backup operations were originally suspended. If you attempt to resume AIJ backup operations from another database node, the following errors are returned:

```
%RDMS-F-CANTRESUMEABS, error resuming AIJ backup operations
-RDMS-F-ABSNSUSPENDED, AIJ backup operations not suspended
%RMU-F-FATALRDB, Fatal error while accessing Oracle Rdb.
```

Command Parameters

root-file-spec
Specifies the database root file for which you want to resume AIJ backup operations.

Command Qualifiers

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the

DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Usage Notes

- To use the RMU Server Backup_Journal Resume command for a database, you must have RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- To determine whether AIJ backup operations have been suspended, use the RMU Show Users command.

Examples

Example 1

The following example demonstrates how to reinstate the ability to perform backup operations.

```
$ RMU/SERVER BACKUP_JOURNAL RESUME MF_PERSONNEL.RDB
```

1.51 RMU Server Backup_Journal Suspend Command

Allows you to temporarily suspend .aij backup operations on all database nodes. While suspended, you cannot back up .aij files manually (with the RMU Backup After_Journal command) nor will the AIJ backup server (ABS) perform .aij backup operations.

Format

RMU/Server Backup_Journal Suspend root-file-spec

Command Qualifiers	Default
/[No]Log	Current DCL verify value

Description

When you issue the RMU Server Backup_Journal Suspend command, after-image journal (AIJ) backup operations are temporarily suspended. However, the suspended state is not stored in the database root file. Thus, if the node from which the AIJ backup operations were suspended fails, then AIJ backup operations by the AIJ Backup Server (ABS) are automatically resumed (assuming the ABS was running prior to the suspension).

The purpose of RMU Server Backup_Journal Suspend command is to temporarily suspend AIJ backup operations during a period of time when backing up .aij files would prevent subsequent commands from operating properly. For example, if you have a Hot Standby database, the time from when the master database is backed up to the time that database replication could commence might be long. During this period, .aij backup operations would prevent the replication from starting. (See the *Oracle Rdb7 and Oracle CODASYL DBMS: Guide to Hot Standby Databases* for information on Hot Standby databases.)

The solution to this problem is to use the RMU Server Backup_Journal Suspend command to suspend AIJ backups from the time just prior to beginning the database backup until after database replication commences.

AIJ backup operations are suspended until any of the following events occur:

- The database is closed on the node from which AIJ backup operations were suspended.
- The node fails from which AIJ backup operations were suspended.

- Database replication is started on the node from which AIJ backup operations were suspended, as a master database.
- AIJ backup operations are explicitly resumed on the node from which AIJ backup operations were suspended. (This occurs when you issue the RMU Server Backup_Journal Resume command. See Section 1.50 for details.)

Command Parameters

root-file-spec
Specifies the database root file for which you want to suspend AIJ backup operations.

Command Qualifiers

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Usage Notes

- To use the RMU Server Backup_Journal Suspend command for a database, you must have RMU$OPEN privilege in the root file access control list (ACL) for the database or the OpenVMS WORLD privilege.
- To determine whether AIJ backup operations have been suspended, use the RMU Show Users command.

Examples

Example 1

The following example first suspends .aij backup operations, then issues the RMU Show Users command to confirm that suspension has occurred. If you attempt an .aij backup operation, you receive the %RMU-F-LCKCNFLCT error message.

1.51 RMU Server Backup_Journal Suspend Command

```
$ RMU/SERVER BACKUP_JOURNAL SUSPEND MF_PERSONNEL.RDB
$ RMU/SHOW USERS MF_PERSONNEL.RDB
 . . .
    * After-image backup operations temporarily suspended
      from this node
 . . .
$ RMU/BACKUP/AFTER_JOURNAL MF_PERSONNEL.RDB AIJ_BACKUP.AIJ
%RMU-F-LCKCNFLCT, lock conflict on AIJ backup
```

1.52 RMU Server Record_Cache Checkpoint Command

Allows the database administrator to force the Record Cache Server (RCS) process to checkpoint all modified rows from cache back to the database.

Format

RMU/Server Record_Cache Checkpoint root-file-spec

Command Qualifier	Defaults
/[No]Log	Current DCL verify value
/[No]Wait	/NoWait

Description

When you use row caches, it is possible for a large number of database records to be modified in row cache areas. These modified records must be written to the physical database files on disk at various times, such as backing up or verifying the database, or when closing the database. The RMU Server Record_Cache Checkpoint command causes the RCS process to immediately write all modified records from all row cache areas back to the physical database files on disk.

If there are a large number of modified records to be written back to the database, this operation can take a long time.

Command Parameters

root-file-spec
Specifies the database root file for which you want to checkpoint all modified rows.

Command Qualifiers

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

1.52 RMU Server Record_Cache Checkpoint Command

Wait
Nowait
Specifies whether the Oracle RMU operation completes right away (Nowait) or whether RMU waits for the record cache server to complete the checkpoint before returning to the user. The default is Nowait.

1.53 RMU Set After_Journal Command

Allows you to do any of the following with respect to after-image journal (.aij) files:

- Enable or disable after-image journaling.
- Alter an .aij file (occurs only if .aij file is re-created).
- Add, drop, modify, or reserve .aij files.
- Suppress the use of an .aij file.
- Add AIJ caches.
- Set the initial .aij file allocation.
- Set the .aij file extent (for extensible journals).
- Enable or disable .aij file overwriting.
- Send OpenVMS operator communication manager (OPCOM) messages when specific after-image journal events occur.
- Set the shutdown timeout period.

Note

Prior to Oracle Rdb Version 6.0, the ability to alter an .aij file name was provided through the RdbALTER DEPOSIT ROOT command. Beginning with Oracle Rdb Version 6.0, the RdbALTER DEPOSIT ROOT command no longer provides this capability; use the Alter qualifier with the RMU Set After_Journal command instead.

1.53 RMU Set After_Journal Command

Format

RMU/Set After_Journal root-file-spec

Command Qualifiers	Defaults
/Add=(keyword[,...])	No journals added
/Aij_Options=OptionsFile	None
/Allocation=number-blocks	See description
/Alter=(keyword[,...])	No journals altered
/Backups=(keyword_list)	See description
/[No]Cache=file	See description
/Disable	None
/Drop=(Name=name)	No journals deleted
/Enable	None
/Extent=number-blocks	See description
/[No]Log	Current DCL verify value
/[No]Notify=(operator-class-list)	See description
/[No]Overwrite	None
/Reserve=number-journals	None
/Shutdown_Timeout=minutes	60 minutes
/Suppress=(Name=name)	No journals suppressed
/Switch_Journal	None

Description

Many of the RMU Set After_Journal functions are also available through the use of the following SQL ALTER DATABASE clauses:

ADD JOURNAL clause
DROP JOURNAL clause
ALTER JOURNAL clause

Command Parameters

root-file-spec
Specifies the database root file for which you want to enable journaling or set .aij file characteristics.

Command Qualifiers

Add=(keyword, ...)
Adds an .aij file to the after-image journal file configuration. You can add an .aij file while users are attached to the database. If you specify the Suppress, Drop, or Alter qualifiers in the same RMU Set After_Journal command, they

are processed before the Add qualifier. The Add qualifier can appear several times in the same command.

Specify an .aij file to add by using the following keywords:

- Name=name

  Specifies a unique name for the after-image journal object to be added. An after-image journal object is the .aij file specification plus all of its attributes, such as allocation, extent, and backup file name.

  This keyword is required.

- File=file

  Specifies the file for the journal to be added. This keyword is required. If you do not provide a full file specification, and only the file name, the file is placed in your current directory. If more than one journal resides in the same directory, each journal must have a unique file name. However, each fixed-size journal file should be located on a separate device. This minimizes risks associated with journal loss or unavailability should a device fail or be brought off line. For example, if two or more journal files reside on the same failed device, the loss of information or its unavailability is far greater than that of a single journal file.

- Backup_File=file

  Specifies the file to be used for automatic backup operations. This keyword is optional. If you specify a file name, but not a file extension, the .aij file extension is used by default. If you supply only a file name (not a complete file specification), the backed up .aij file is placed in the database root file directory.

- Edit_Filename=(option)

  Specifies an edit string to apply to the backup file when an .aij is backed up automatically. This keyword is optional. However, if it is specified, the Backup_File=file keyword must be specified also. When you specify the Edit_Filename=(options) keyword, the .aij backup file name is modified by appending the options you specify.

  See the description of the Edit_Filename keyword for the Backups qualifier for a list of the available keyword options.

  This keyword and the options you specify affect the backup file name of the .aij file specified with the associated Name keyword only. If you want the same edit string applied to all backed up .aij files, you might find it more efficient to use the Backups qualifier with the Edit_Filename keyword instead of the Add qualifier with the Edit_Filename keyword.

If you use a combination of the Edit_Filename keyword with the Add qualifier and the Edit_Filename keyword with the Backups qualifier, the Add qualifier keyword takes precedence over the Backups qualifier keyword for the named .aij file. In other words, the options you specify with Edit_ Filename keyword to the Backups qualifier are applied to all backed up .aij files except those for which you explicitly specify the Edit_Filename keyword with the Add qualifier. See Example 6.

This keyword is useful for creating meaningful file names for your backup files and makes file management easier.

- Allocation=number-blocks

  Sets the initial size, in disk blocks, of the .aij file. If this keyword is omitted, the default allocation is used. The minimum valid value is 512, the maximum value is eight million. The default is 512.

  See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the allocation size.

- Extent=number-blocks

  Specifies the maximum size to extend an .aij file if it is, or becomes, an extensible .aij file (in blocks). (If the number of available after-image journal files falls to one, extensible journaling is employed.)

  If there is insufficient free space on the .aij file device, the journal is extended using a smaller extension value than specified. However, the minimum, and default, extension size is 512 blocks.

  See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the extent size.

AIJ_Options=OptionsFile

Specifies an options file name. The default extension is .opt. The OptionsFile is the same as that generated by an RMU Show After_Journal command and is also used by the RMU Copy_Database, Move_Area, Restore, and Restore Only_Root commands. The AIJ_Options qualifier may be used alone or in combination with other RMU Set After_Journal command qualifiers.

Allocation=number-blocks

Sets the default .aij file allocation. You can change the allocation while users are attached to the database. If the Allocation qualifier is omitted, the default allocation is unchanged.

The minimum value you can specify is 512. The default is also 512.

See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the allocation size.

1.53 RMU Set After_Journal Command

Alter=(keyword,...)
Specifies that an after-image journal object be altered.

You can alter an after-image journal object while users are attached to the database. The Alter qualifier can be used several times within the same RMU Set After_Journal command. If you specify a previously suppressed .aij file with the Alter qualifier, that named .aij file is unsuppressed. Oracle RMU performs this unsuppress action as soon as the command is processed.

The changes specified by the Alter qualifier are stored in the database root file (and thus are visible in the dump file if you issue an RMU Dump command), but the changes are not applied to the .aij file until it is re-created (or backed up, in the case of the Backup_File= file keyword). A new extensible .aij file is re-created, for example, when the following are true:

- Fast commit is enabled.
- Extensible after-image journaling is being used.
- Users are actively updating the database.
- You issue an RMU Backup After_Journal command with the Noquiet_Point qualifier.

Backing up an extensible .aij file does not ensure that a new .aij file will be created. In most cases, the existing .aij file is truncated and reused.

Specify an after-image journal object to alter by using the following keywords:

- Name=name

  Specifies the name of the after-image journal object. This is a required keyword that must match the name of an existing after-image journal object.

- File=file

  This option only takes effect if a journal is, or becomes, an extensible .aij file and only when that journal is re-created. This option allows you to supply a new .aij file specification to be used for the extensible .aij file if and when it is re-created. This can be used to move the re-created .aij file to a new location. If you do not provide a full file specification, and only the file name, the file is placed in your current directory. See the general description of the Alter qualifier for an example of when an extensible .aij file is re-created.

  This option cannot be used to move a fixed-size .aij file. To move a fixed-size .aij file, you must first create a new .aij file and then drop the existing .aij file.

  This keyword is optional.

- Backup_File=file

  Specifies a new file to be used for automatic backup operations.

  This keyword is optional.

- Edit_Filename=(options)

  Specifies a new edit string to apply to the backup file name of the named .aij file when the .aij is backed up automatically. This keyword is optional. See the description of the Edit_Filename keyword for the Backups qualifier for a list of the available keyword options.

- Allocation=number-blocks

  Specifies the initial size of the .aij file that is re-created if that file is, or becomes, a fixed-size .aij file.

- Extent=number-blocks

  Specifies the extent size of the .aij file that is re-created if it is, or becomes, extensible.

  See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the extent size.

Backups=(keyword_list)

Specifies options to control the AIJ backup server. You can select one or more of the following keywords:

- Automatic

  Specifies that the AIJ backup server will run automatically, as required. You cannot specify both the Automatic and Manual keywords. If neither the Automatic nor the Manual keyword is specified, the backup server state is unchanged.

- Manual

  Specifies that the RMU Backup After_Journal command will be used to back up the .aij files. The AIJ backup server will not run automatically. You cannot specify both Automatic and Manual keywords. If neither the Automatic nor the Manual keyword is specified, the backup server state is unchanged.

- Backup_File=file

  Specifies a default file specification for the AIJ backup server to use as the backup file name if no backup file name is associated with the .aij file to be backed up.

- Nobackup_File

Specifies that there is no default backup file specification. Omission of this keyword retains the current default backup file specification.

- [No]Compression[=ZLIB[=level]]

  Establishes the default compression setting for both automatic and manual after image journal backups. The default is NoCompression. This information is saved in the database root and will be used during automatic after image backups (ABS) and when manual backups specify neither COMPRESSION nor NOCOMPRESSION on the RMU/BACKUP/AFTER_ JOURNAL command. The RMU Dump Header=Journal command can be used to display the current settings.

  ZLIB [=level]

  The ZLIB keyword specifies the currently supported compression algorithm. It can be followed by the compression level; the higher the level number the more time is spent compressing the data.

  This value can be specified between 1 and 9, with a default value of 6. The default is a fair trade off between reasonable compression and least CPU cost for compression.

  This example shows the selection of level 5 compression.

```
$ rmu/set after_journal mf_personnel/backup=compression=ZLIB:5/log
%RMU-I-LOGMODVAL,     modified AIJ backup compression level to 5
```

  See the RMU/BACKUP/After_Journal command for more information on compression.

- Edit_Filename=(options)

  The Edit_Filename keyword specifies an edit string to apply to .aij files when they are backed up automatically. When the Edit_Filename=(options) keyword is used, the .aij backup file names are edited by appending any or all of the values specified by the following options to the backup file name:

  – Day_Of_Year

    The current day of the year expressed as a 3-digit integer (001 to 366).

  – Day_Of_Month

    The current day of the month expressed as a 2-digit integer (01 to 31).

  – Hour

    The current hour of the day expressed as a 2-digit integer (00 to 23).

  – Julian_Date

    The number of days passed since 17-Nov-1858.

- Minute

  The current minute of the hour expressed as a 2-digit integer (00 to 59).

- Month

  The current month expressed as a 2-digit integer (01 to 12).

- Sequence

  The journal sequence number of the first journal in the backup operation.

- Vno

  Synonymous with the Sequence option. See the description of the Sequence option.

- Year

  The current year (A.D.) expressed as a 4-digit integer.

If you specify more than one option, place a comma between each option.

The edit is performed in the order specified. For example, the file backup.aij and the keyword EDIT_FILENAME=(HOUR, MINUTE, MONTH, DAY_OF_MONTH, SEQUENCE) creates a file with the name backup_160504233.aij when journal 3 is backed up at 4:05 P.M. on April 23rd.

You can make the name more readable by inserting quoted strings between each Edit_Filename option. For example, the option shown in the following code adds the string "$30_0155-2" to the .aij file name if the day of the month is the 30th, the time is 1:55 and the version number is 2:

```
/EDIT_FILENAME=("$",DAY_OF_MONTH,"_",HOUR,MINUTE,"-",SEQUENCE)
```

This keyword is useful for creating meaningful file names for your backup files and makes file management easier.

If you use a combination of the Edit_Filename keyword with the Add qualifier and the Edit_Filename keyword with the Backups qualifier, the Add qualifier keyword takes precedence over the Backups qualifier keyword for the named .aij file. In other words, the options you specify with Edit_ Filename keyword to the Backups qualifier are applied to all .aij back up files except those for which you explicitly specify the Edit_Filename keyword with the Add qualifier. See Example 6.

- Quiet_Point

Specifies that the after-image journal backup operation is to acquire the quiet-point lock prior to performing an .aij backup operation for the specified database. This option (as with all the other Backup options) affects only the database specified in the RMU Set After_Journal command line. For information on specifying that the quiet-point lock be acquired before any .aij backup operation is performed on a system, see the Usage Notes.

- Noquiet_Point

  Specifies that the after-image journal backup operation will not acquire the quiet-point lock prior to performing an .aij backup operation for the specified database. This option (as with all the other Backup options) affects only the database specified in the RMU Set After_Journal command line. For information on specifying that the quiet-point lock will not be acquired prior to any .aij backup operations performed on a system, see the Usage Notes.

Cache=file
Nocache

Specifies an after-image journal cache file specification on a solid-state disk. If the Cache qualifier is specified, after-image journal caches are enabled. If you specify a file name, but not a file extension, the file extension .aij is used by default.

If the Nocache qualifier is specified, AIJ caches are disabled. You can use this qualifier only when users are detached from the database.

This file must be written to a solid-state disk. If a solid-state disk is not available, after-image journal caching should not be used. Unless you are involved in a high performance, high-volume environment, you probably do not need the features provided by this qualifier.

You can determine whether the cache file is accessible by executing the RMU Dump command with the Header qualifier. If caching is enabled, but the cache file is unavailable, the cache file is marked inaccessible and after-image journaling continues as if caching was disabled. Once the cache file has been marked inaccessible, it will remain so marked until either the existing cache file is dropped from the database, or a new cache file is added to the database (even if this is the same cache file as was previously used).

If this qualifier is omitted, the AIJ cache state remains unchanged.

Disable
Disables after-image journaling if it has already been enabled. If after-image journaling has already been disabled, this qualifier has no effect. You can specify the Disable qualifier only when users are detached from the database.

When the Disable qualifier and other qualifiers are specified with the RMU Set After_Journal command, after-image journaling is disabled before other requested operations.

There is no default for the Disable qualifier. If you do not specify either the Disable or Enable qualifier, the after-image journaling state remains unchanged.

Drop=(Name=name)
Specifies that the named after-image journal object be deleted. You can drop an after-image journal object while users are attached to the database, but the named after-image journal object must not be the current .aij file or be waiting to be backed up. When the Drop qualifier is specified with the Alter or Add qualifiers on the RMU Set After_Journal command, the named after-image journal object is dropped before any after-image journal objects are altered or added.

Each after-image journal object to be deleted is specified by the required keyword, Name=name. This specifies the name of the after-image journal object to be dropped, which must match the name of an existing after-image journal object.

Enable
Enables after-image journaling if it has been disabled. You can specify the Enable qualifier only when users are detached from the database and at least one unmodified .aij file is available (unless you also specify the Overwrite qualifier). After-image journaling is enabled after other specified qualifiers have been processed.

Extent=number-blocks
Sets the size, in blocks, of the default .aij file extension. This qualifier has no effect on fixed-length .aij files. This qualifier can be used while users are attached to the database.

The minimum valid number-blocks value is 512. The default is also 512.

If the Extent qualifier is omitted, the default extension remains unchanged.

See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the extent size.

1.53 RMU Set After_Journal Command

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

Notify=(operator-class-list)
Nonotify
Sets the operator notification state for after-image journaling and selects the operators to be notified when the journaling state changes. Oracle RMU uses the OpenVMS operator communication manager (OPCOM). The following events evoke operator notification:

- An error writing to an .aij file.
- No .aij file is available for write operations.
- The .aij file has been overwritten.
- The RMU Backup After_Journal command fails.

You can use this qualifier while users are attached to the database. If you specify the Nonotify qualifier, operator notification is disabled. If the qualifier is omitted, the operator notification state is unchanged.

The operator classes follow:

- [No]All

  The All operator class broadcasts a message to all terminals that are attached to the system or cluster. These terminals must be turned on and have broadcast-message reception enabled. The Noall operator class inhibits the display of messages to the entire system or cluster.

- [No]Central

  The Central operator class broadcasts messages to the central system operator. The Nocentral operator class inhibits the display of messages to the central system operator.

- [No]Disks

  The Disks operator class broadcasts messages pertaining to mounting and dismounting disk volumes. The Nodisks operator class inhibits the display of messages pertaining to mounting and dismounting disk volumes.

- [No]Cluster

The Cluster operator class broadcasts messages from the connection manager pertaining to cluster state changes. The Nocluster operator class inhibits the display of messages from the connection manager pertaining to cluster state changes.

- [No]Security

  The Security operator class displays messages pertaining to security events. The Nosecurity operator class inhibits the display of messages pertaining to security events.

- [No]Oper1 through [No]Oper12

  The Oper1 through Oper12 operator classes display messages to operators identified as OPER1 through OPER12. The Nooper1 through Nooper12 operator classes inhibit messages from being sent to the specified operator.

Note

Use the Notify qualifier conservatively. Be sure that messages regarding a private database are not broadcast to an entire system or cluster of users who may not be interested in the broadcast information. Similarly, be conservative regarding even a clusterwide database. You do not want to overload the operators with insignificant messages.

Overwrite
Nooverwrite

The Overwrite qualifier specifies that .aij files can be overwritten without first being backed up. The Nooverwrite qualifier specifies that only an .aij file that has been backed up can be overwritten. You can specify the Nooverwrite qualifier only when users are detached from the database. If you do not specify either the Overwrite qualifier or the Nooverwrite qualifier, the Overwrite characteristic remains unchanged.

This qualifier is ignored if only one .aij file is available. When you specify the Overwrite qualifier, it is only activated when two or more .aij files are, or become, available.

Note that if you use the Overwrite qualifier, you will be unable to perform a rollforward from a restored backup file. Most users will not want to use the Overwrite qualifier; it is provided for layered applications that might want to take advantage of some performance features provided by Oracle Rdb that require after-image journaling, but where the use of after-image journaling is not required for the application to run reliably.

1.53 RMU Set After_Journal Command

Reserve=number-journals
Reserves additional space in the after-image journal configuration for the specified number of .aij files. You can specify the Reserve qualifier only when users are detached from the database. If you do not specify the Reserve qualifier, no space is reserved for additional .aij files.

Note that you cannot reserve space in a single-file database for .aij files by using this qualifier with the RMU Set After_Journal command. After-image journal file reservations for a single-file database can be made only when you use the RMU Convert, RMU Restore, or RMU Copy_Database commands.

Note that once you reserve space in the journal configuration (using the Reserve=n qualifier), the reservations are permanent. There is no way to unreserve this space unless you back up and restore the database. Specify fewer reservations with RMU Restore command After_Journal qualifier.

Each reservation uses two blocks of space in the root file and the run-time global sections.

When you reserve journals slots to create additional journals for your journal system, the reserve operation is not journaled. Therefore, you should perform a full database backup operation to ensure database consistency.

Shutdown_Timeout=minutes
Modifies the after-image journal shutdown time in the event that after-image journaling becomes unavailable. The after-image journaling shutdown time is the period, in minutes, between the point when after-image journaling becomes unavailable and the point when the database is shut down. During the after-image journaling shutdown period, all database update activity is stalled.

If operator notification has been enabled, operator messages are broadcast to all enabled operator classes and to the RMU Show Statistics screen at 1-minute intervals.

To recover from the after-image journaling shutdown state and to resume normal database operations, you must make an .aij file available for use. You can do this by backing up an existing modified journal, or, if you have a journal reservation available, by adding a new journal to the after-image journaling configuration. If you do not make a journal available before the after-image journal shutdown time expires, the database is shut down and all active database attaches are terminated.

The after-image journaling shutdown period is only in effect when fixed-size AIJ journaling is used. When a single extensible .aij file is used, the default action is to shut down all database operations when the .aij file becomes unavailable.

If you do not specify the Shutdown_Timeout qualifier, the database shuts down 60 minutes after the after-image journaling configuration becomes unavailable. The maximum value you can specify for the Shutdown_Timeout qualifier is 4320 minutes (3 days).

Suppress=(Name=name)
Prevents further use of the named after-image journal object. The named after-image journal object must be an existing after-image journal object.

This qualifier is useful when you want to temporarily disallow the use of an .aij file. For example, suppose the disk containing the next .aij file to use goes off line. You do not want the database to attempt to access that file until the disk is back on line. Use the Suppress qualifier so the database does not attempt to access the specified .aij file. When the disk is back on line, use the RMU Set After_Journal command with the Alter qualifier to unsuppress the after-image journal object that references this .aij file.

You can specify the Suppress qualifier while users are attached to the database, but the .aij file referenced by the after-image journal object must not be the current journal or be waiting to be backed up. You must back up the referenced .aij file before the after-image journal object that references it can be suppressed.

The Suppress qualifier is processed prior to any Drop, Add, or Alter qualifiers specified with the same command.

Switch_Journal
Changes the currently active .aij file to the next available .aij file in a fixed-size after-image journaling configuration.

In an extensible journal file configuration, the Switch_Journal qualifier has no effect and is ignored if specified.

The Switch_Journal qualifier is useful for forcing a switch to an .aij file on another disk when you want to perform maintenance on the disk containing the currently active journal file.

You cannot specify the Switch_Journal qualifier and the Enable or the Disable qualifier on the same command line. In addition, after-image journaling must be enabled when you issue the Switch_Journal qualifier.

It is seldom necessary to specify this option because normally a switch occurs automatically.

Usage Notes

- You must have the RMU$ALTER, RMU$BACKUP, or RMU$RESTORE privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege to use the RMU Set After_Journal command.
- Use the RMU Dump command with the Header qualifier to see if after-image journaling additions or changes you have made have been recorded as you expect. However, note that although the AIJ attributes change as you specify, the changed .aij file might be flagged as unmodified in the dump of the header. This occurs because the transaction containing your changes to the .aij file is captured in the current .aij file, not the .aij file for which you specified modifications.
- When you use RMU Set After_Journal to specify a fixed-size journal configuration, specify a different disk for each .aij file, if possible. Using this method, you can suppress a journal on a given disk if that disk should start to fail.
- If the disk fails on which the current .aij file resides, Oracle Rdb immediately starts using a new .aij file if your journal configuration contains more than one journal. For example, if AIJ_DISK1 contains AIJ_ONE, the current .aij file, and AIJ_DISK1 fails, Oracle Rdb will immediately start using AIJ_TWO, the .aij file on AIJ_DISK2.
- Execute a full database backup operation after issuing an RMU Set After_Journal command that displays the RMU-W-DOFULLBCK warning message (such as a command that includes the Reserve or the Enable qualifier).
- Use the Alter qualifier to unsuppress an .aij file that has been suppressed with the Suppress qualifier.
- Use the Backup=(Quiet_Point) qualifier to specify that the quiet-point lock must be acquired prior to performing an .aij backup operation for the *specified database*. (Use the Backup=(Noquiet_Point) qualifier to specify that the quiet-point lock will not be acquired prior to an .aij backup operation for the specified database.)
- Use the RDM$BIND_ABS_QUIET_POINT logical to specify whether or not the quiet-point lock must be acquired prior to performing any .aij backup operation on *any database* on a cluster.

Define the value for the logical to be 1 to specify that the quiet-point lock must be acquired prior to performing .aij backup operations; define the value to be 0 to specify that the quiet-point lock need not be acquired prior to .aij backup operations. You must define this logical in the system table on all nodes in the cluster as shown in the following example:

```
$ DEFINE/SYSTEM RDM$BIND_ABS_QUIET_POINT 1
```

- The selection of which journal in a set of fixed-size journal files is used by Oracle RMU is unpredictable and depends on availability. For example, while a journal is temporarily unavailable, it cannot be selected as the next journal file. Thus, a journal file might be reused before all journals in the set have been used once.
- When using RMU Backup After_Image and when neither the /COMPRESSION nor the /NOCOMPRESSION qualifier is used, then the database default journal setting will be used for compression.
- The RMU Dump Header=Journal command can be used to display the current database default journal setting.

Examples

Example 1

The following command reserves space for three .aij files, adds two .aij files to the mf_personnel database, and then enables after-image journaling:

```
$ RMU/SET AFTER_JOURNAL/ENABLE/RESERVE=3           -
_$ /ADD=(NAME=AIJ2, FILE=DISK1:[JOURNAL]AIJ_TWO)   -
_$ /ADD=(NAME=AIJ3, FILE=DISK2:[JOURNAL]AIJ_THREE) -
_$ MF_PERSONNEL
%RMU-W-DOFULLBCK, full database backup should be done to
 ensure future recovery
```

Example 2

The following example demonstrates how to switch the current .aij file from DISK1:[DB]AIJ1 to the next available journal file in a fixed-size journal configuration, and then suppress the original journal in anticipation of maintenance on the disk that contains it. The last Oracle RMU command moves AIJ1 to a new disk and implicitly unsuppresses it.

1.53 RMU Set After_Journal Command

```
$ RMU/DUMP/HEADER=(JOURNAL) MF_PERSONNEL
   .
   .
   .
    AIJ Journaling...
     - After-image journaling is enabled
      - Database is configured for 5 journals
      - Reserved journal count is 5
      - Available journal count is 3
      - Journal switches to next available when full
      - 1 journal has been modified with transaction data
      - 2 journals can be created while database is active
      - Journal "AIJ1" is current
      - All journals are accessible
   .
   .
   .
$ RMU/SET AFTER_JOURNAL/SWITCH_JOURNAL MF_PERSONNEL/LOG
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
 USER1:[DB]MF_PERSONNEL.RDB;1 Event Notification
After-image journal 0 switch-over in progress (to 1)

%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
 USER1:[DB]MF_PERSONNEL.RDB;1 Event Notification
After-image journal switch-over complete

%RMU-I-LOGMODSTR,     switching to after-image journal "AIJ2"
   .
   .
   .

$ RMU/BACKUP/AFTER_JOURNAL MF_PERSONNEL DISK1:[DB]AIJ1_BCK/LOG
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
 USER1:[DB]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation started
%RMU-I-AIJBCKSEQ, backing up after-image journal sequence number 2
%RMU-I-LOGBCKAIJ, backing up after-image journal AIJ1 at 10:59:58.83
%RMU-I-LOGCREBCK, created backup file DISK1:[DB]AIJ1_BCK.AIJ;1
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
 USER1:[DB]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation completed

%RMU-I-AIJBCKEND, after-image journal backup operation completed
 successfully
%RMU-I-LOGAIJJRN, backed up 1 after-image journal at 11:00:02.59
%RMU-I-LOGAIJBLK, backed up 254 after-image journal blocks
 at 11:00:02.59
$ RMU/SET AFTER_JOURNAL/SUPPRESS=(NAME=AIJ1) MF_PERSONNEL/LOG
%RMU-I-LOGMODSTR,     suppressed after-image journal "AIJ1"
$ RMU/SET AFTER_JOURNAL MF_PERSONNEL -
_$ /ALTER=(NAME=AIJ1,FILE=DISK2:[DB]AIJ1)/LOG
%RMU-I-LOGMODSTR,     unsuppressed after-image journal "AIJ1"
```

Example 3

The following example turns on the automatic backup server for .aij files and defines a default backup file name:

```
$ RMU/SET AFTER_JOURNAL /BACKUPS=(AUTOMATIC, -
_$ BACKUP_FILE=DISK:[AIJ_BACKUPS]AIJ_BACKUP.AIJ) -
_$ DB$DISK:[DIRECTORY]MF_PERSONNEL.RDB
```

Example 4

The following example turns off the automatic backup server for .aij files and removes the default backup file name:

```
$ RMU/SET AFTER_JOURNAL /BACKUPS=(MANUAL,NOBACKUP_FILE) -
_$ DB$DISK:[DIRECTORY]MF_PERSONNEL.RDB
```

Example 5

The following example changes the .aij backup file name without changing the setting of the AIJ backup server:

```
$ RMU/SET AFTER_JOURNAL /BACKUPS= -
_$ (BACKUP_FILE=NEW_DISK:[AIJ_BACKUPS]BETTER_BACKUP_NAME.AIJ) -
_$ DB$DISK:[DIRECTORY]MF_PERSONNEL.RDB
```

Example 6

The following example sets a local and a global edit string for .aij backup files. When AIJ_ONE is backed up, it is appended with the string _LOCAL. When AIJ_TWO or AIJ_THREE are backed up, they are appended with the string _GLOBAL. Although it is unlikely that you would select these edit strings, they demonstrate the behavior of the Edit_Filename keyword when it is used with the Backup qualifier (global effect) versus the behavior of the Edit_Filename keyword when it is used with the Add qualifier (local effect).

```
$ RMU/SET AFTER_JOURNAL/ENABLE/RESERVE=5 -
_$ /BACKUP=EDIT_FILENAME=("_GLOBAL")/ADD=(NAME=AIJ1, -
_$    FILE=DISK1:[AIJS]AIJ_ONE, -
_$    BACKUP_FILE=AIJ1BCK, -
_$    EDIT_FILENAME=("_LOCAL")) -
_$  /ADD=(NAME=AIJ2, -
_$    FILE=DISK1:[AIJS]AIJ_TWO,  -
_$    BACKUP_FILE=AIJ2BCK) -
_$  /ADD=(NAME=AIJ3, -
_$    FILE=DISK1:[AIJS]AIJ_THREE, -
_$    BACKUP_FILE=AIJ3BCK) -
_$ MF_PERSONNEL
```

1.53 RMU Set After_Journal Command

```
$ !
$ ! After these .aij files are backed up:
$ !
$ DIR .AIJ
AIJ1BCK_LOCAL.AIJ;1
AIJ2BCK_GLOBAL.AIJ;1
AIJ3BCK_GLOBAL.AIJ;1
AIJ_ONE.AIJ;1
AIJ_THREE.AIJ;1
AIJ_TWO.AIJ;1
```

1.54 RMU Set AIP Command

Allows the user to modify the contents of the AIP (Area Inventory Pages) structure. The AIP structure provides a mapping for logical areas to physical areas as well describing each of those logical areas. Information such as the logical area name, length of the stored record, and storage thresholds can now be modified using this simple command interface.

Format

RMU/Set AIP root-file-spec [larea-name]

Command Qualifiers	Defaults
/Larea=(n [,...])	See description
/Length[=n]	See description
/Log	See description
/Rebuild_Spams	See description
/Rename_To=new-name	See description
/Threshold=(p,q,r)	See description

Description

This RMU command is used to modify some attributes of an existing logical area. It cannot be used to add or delete a logical area. This command can be used to correct the record length, thresholds and name of a logical area described by an AIP entry. It can also be used to rebuild the SPAM pages for a logical area stored in UNIFORM page format areas so that threshold settings for a page correctly reflect the definition of the table.

See also the RMU Repair Spam command for information on rebuilding SPAM pages for MIXED areas.

Command Parameters

root-file-spec
The file specification for the database root file to be processed. The default file extension is .rdb.

larea-name
An optional parameter that allows the logical areas to be selected by name. Only those AIP entries are processed.

Any partitioned index or table will create multiple logical areas all sharing the same name. This string may contain standard OpenVMS wildcard characters (% and *) so that different names can be matched. Therefore, it is possible for many logical areas to match this name.

The value of **larea-name** may be delimited so that mixed case characters, punctuation and various character sets can be used.

Command Qualifiers

Larea=(n [,...])
Specifies a list of logical area identifiers. The LAREA qualifier and larea-name parameter are mutually exclusive.

Length [=value]
Sets the length of the logical area. If no value is provided on the RMU Set AIP command, then Oracle Rdb will find the matching table and calculate a revised AIP nominal record length and apply it to the AIP.

Log
Logs the names and identifiers of logical areas modified by this command.

Rebuild_Spams
Locate each logical area with the "rebuild-spam" flag set and rebuild the SPAM pages.

Rename_To=new-name
Used to change the logical area name. This qualifier should be used with caution as some RMU commands assume a strict mapping between table/index names and names of the logical area. This command can be used to repair names that were created in older versions of Oracle Rdb where the rename table command did not propagate the change to the AIP. The value of new-name may be delimited so that mixed case, punctuation and various character sets can be used.

Threshold = (t1 [,t2 [, t3]])
Changes the threshold on all logical areas specified using the Larea qualifier or the larea-name parameter. RMU accepts THRESHOLD=(0,0,0) as a valid setting to disable logical area thresholds. Values must be in the range 0 through 100. Any missing values default to 100.

Usage Notes

- The database administrator requires RMU$ALTER privilege to run the command and the Rdb server also requires SELECT and ALTER privilege on the database.
- This command supersedes the RMU Repair Initialize=Larea_Parameters command that can also change the Thresholds and Length for a logical area. This command can be executed online, whereas the RMU Repair command must be run offline.
- Wildcard names are not permitted with the following qualifiers to prevent accidental propagation of values to the wrong database objects.
  - LENGTH qualifier with a value specified,
  - RENAME_TO qualifier,
  - and THRESHOLDS qualifier.
- RMU Set AIP may be used on a master database configured for HOT STANDBY. All AIP changes and SPAM rebuild actions are written to the after image journal and will be applied to the standby database. This command cannot be applied to a STANDBY database.
- THRESHOLDS for MIXED format areas are physical area attributes and are not supported at the logical area (aka AIP) level. Therefore, THRESHOLDS can not be applied to MIXED areas and specifying logical areas will cause an exception to be raised.
- The REBUILD_SPAMS qualifier is only applied to logical areas stored in UNIFORM page format storage areas.
- This command will implicitly commit any changes with no opportunity to undo them using rollback. Access to the functionality is controlled by privileges at the RMU and Rdb database level. We suggest that RMU Show AIP be used prior to any change so that you can compare the results and repeat the RMU Set AIP command with corrections if necessary.

  Some wildcard operations are restricted to prevent accidental damage to the database. For instance, a wildcard matching many objects will be rejected if more than one type of object is being changed. If a wildcard selects both table and index types then this command will be rejected.
- This command is an online command. Each logical area will be processed within a single transaction and interact with other online users.

- When the AIP entry is changed online, any existing users of the table or index will start to use the new values if the logical areas are reloaded.
- Various SQL alter commands will register changes for the AIP and these are applied at COMMIT time. RMU Verify and RMU Show AIP Option=REBUILD_SPAMS will report any logical areas that require SPAM rebuilding. The database administrator can also examine the output from the RMU Dump Larea=RDB$AIP command.
- How long can the SPAM rebuild be delayed? The fullness of some page will have been calculated using the old AIP length or THRESHOLD values. Therefore, it might appear that a page is full when in fact the revised length will fit on the page, or the page may appear to have sufficient free space to store a row but once accessed the space is not available. By rebuilding SPAM pages, you may reduce I/O during insert operations. However, delaying the rebuild to a convenient time will not affect the integrity of the database.
- The amount of I/O required for Rebuild_Spams depends upon the number of pages allocated to the table or index involved. Assuming just one logical area is selected then Oracle Rdb will read the ABM (Area Bitmap) to locate all SPAM pages in that area that reference this logical area. Rdb will then read each page in the SPAM interval for that SPAM page and recalculate the fullness based on the rows stored on each page.

Examples

Example 1

RMU will call Rdb for each logical area that requires rebuilding.

```
$ RMU/SET AIP/REBUILD_SPAMS MF_PERSONNEL
%RMU-I-AIPSELMOD, Logical area id 86, name ACCOUNT_AUDIT selected for
modification
%RMU-I-AIPSELMOD, Logical area id 94, name DEPARTMENTS_INDEX selected for
modification
```

Example 2

RMU will request that the EMPLOYEES table length be updated in the AIP. Oracle Rdb will use the latest table layout to calculate the length in the AIP and write this back to the AIP. The EMPLOYEES table is partitioned across three storage areas and therefore the Log qualifier shows these three logical areas being updated.

```
$ RMU/SET AIP MF_PERSONNEL EMPLOYEES/LENGTH/LOG
%RMU-I-AIPSELMOD, Logical area id 80, name EMPLOYEES selected for modification
%RMU-I-AIPSELMOD, Logical area id 81, name EMPLOYEES selected for modification
%RMU-I-AIPSELMOD, Logical area id 82, name EMPLOYEES selected for modification
```

Example 3

RMU will request that the EMPLOYEES table length be updated in the AIP and then the SPAM pages will be rebuilt. This is an ONLINE operation. Note: there is an implied relationship between the logical area name and the name of the object. This example assumes that the EMPLOYEES object is mapped to a UNIFORM page format area.

```
$ RMU/SET AIP MF_PERSONNEL EMPLOYEES/LENGTH/REBUILD_SPAMS
```

Example 4

When Thresholds for an index are modified they will not be effective until the SPAM pages are updated (rebuilt) to use these new values. The following example shows that index maintenance performed by SQL. The SET FLAGS command is used to display information about the change. Note that the change is applied at COMMIT time and that the SPAM rebuild is deferred until a later time. RMU Set AIP is then used to rebuild the SPAM pages.

```
$ SQL$
SQL> set flags 'index_stats';
SQL> alter index candidates_sorted store in rdb$system (thresholds are (32,56,
77));
~Ai alter index "CANDIDATES_SORTED" (hashed=0, ordered=0)
~Ai larea length is 215
~As locking table "CANDIDATES" (PR -> PU)
~Ai: reads: async 0 synch 58, writes: async 8 synch 0
SQL> commit;
%RDMS-I-LOGMODVAL,     modified space management thresholds to (32%, 56%, 77%)
%RDMS-W-REBUILDSPAMS, SPAM pages should be rebuilt for logical area
CANDIDATES_SORTED
$
$ RMU/SET AIP MF_PERSONNEL CANDIDATES_SORTED/REBUILD_SPAMS/LOG
%RMU-I-AIPSELMOD, Logical area id 74, name CANDIDATES_SORTED selected for
modification
```

1.55 RMU Set Audit Command

Enables Oracle Rdb security auditing. When security auditing is enabled, Oracle Rdb sends security alarm messages to terminals that have been enabled as security operators and makes entries in the database's security audit journal whenever specified audit events are detected.

Format

RMU/Set Audit root-file-spec

Command Qualifiers	Defaults
/Disable=enable-disable-options	See description
/Enable=enable-disable-options	See description
/[No]Every	/Every
/First	Synonym for /Noevery
/[No]Flush	/Noflush
/Start	See description
/Stop	See description
/Type={Alarm|Audit}	Alarm and Audit

Description

The RMU Set Audit command is the Oracle Rdb equivalent to the DCL SET AUDIT command. Because Oracle Rdb security auditing uses many OpenVMS system-level auditing mechanisms, certain auditing characteristics (such as /FAILURE_MODE) can only be set and modified by using the DCL SET AUDIT command, which requires the OpenVMS SECURITY privilege.

Command Parameters

root-file-spec
The file specification of the database root for which auditing information will be modified.

Command Qualifiers

Disable=enable-disable-options
Disables security auditing for the specified audit event classes. To disable alarms and audits for all classes, specify the All option. You can also selectively disable alarms and audits for one or more classes that are currently enabled. You must specify at least one class when you specify the Disable qualifier. See

the Enable qualifier description for a list of the classes you can specify with the Disable qualifier.

When you specify audit classes with the Disable qualifier, the events you specify are immediately disabled. For other audit events that have not been explicitly disabled with the Disable qualifier, records continue to be recorded in the security audit journal and alarms continue to be sent to security-enabled terminals, as specified.

When processing the RMU Set Audit command, Oracle Rdb processes the Disable qualifier last. If you accidentally specify both Enable and Disable for the same event type in the same command, the Disable qualifier prevails.

Enable=enable-disable-options
Enables security auditing for the specified audit event classes. To enable alarms and audits for all events, specify the All option. You can also selectively enable alarms and audits for one or more classes that are currently disabled. You must specify at least one class when you specify the Enable qualifier.

When you specify audit classes with the Enable qualifier, the audit events you specify are immediately enabled, so that audit events of currently attached users are recorded in the security audit journal and alarms are sent to security-enabled terminals, as specified.

With the Enable and Disable qualifiers, you can specify one or more of the following six valid class options: All, Daccess, Daccess=object-type, Identifier=(identifier-list), Protection, and Rmu. If you specify more than one class, separate the classes with commas, and enclose the list of classes within parentheses. The following list provides a description of each option:

- All

  Enables or disables all possible audit event classes.

- Daccess

  Enables or disables DACCESS (discretionary access) audit events.

  A DACCESS audit event occurs whenever a user issues a command that causes a check to be made for the existence of the appropriate privilege in an access privilege set (APS). To monitor access to a particular database object or group of objects, use the Daccess=object-type option to specify that a DACCESS audit record be produced whenever an attempt is made to access the object.

Specifying the general Daccess option enables or disables the general DACCESS audit event type. If DACCESS event auditing is enabled and started for specific objects, auditing takes place immediately after you issue the RMU Set Audit command with the Enable=Daccess qualifier. Auditing starts for any users specified in the Identifier=(identifier-list) option who are attached to the database when the command is issued.

- Daccess=object-type[=(object name)]/Privileges=(privilege-list)

  Allows you to audit access to database objects by users in the Identifier=(identifier-list) option with the privileges you specify.

  A DACCESS type event record indicates the command issued, the privilege used by the process issuing the command, and whether the attempt to access the object was successful.

  The object-type option enables or disables DACCESS auditing for the specified object type. You can specify one or more object types in an RMU Set Audit command. The valid object types are:

  – DATABASE

    When you specify the DATABASE object type, you must use the Privileges qualifier to specify one or more privileges to be audited for the database. Do not specify an object name with the DATABASE object type.

  – TABLE

    Specify the TABLE option for both tables and views. When you specify the TABLE object type, you must specify one or more table names with the object name parameter. You must also use the Privileges qualifier to specify one or more privileges to be audited for the specified tables.

  – COLUMN

    When you specify the COLUMN object type, you must specify one or more column names with the object name parameter. Specify the table name that contains the column by using the following syntax:

    table-name.column-name

    If you specify more than one column, separate the list of table-name.column-names with commas, and enclose the list within parentheses. You must also use the Privileges qualifier to specify one or more privileges to be audited for the specified columns.

  – SEQUENCE

Specify the name of a sequence, either explicitly or using a wildcard. Only user defined sequences will be selected by this type of wildcard command. Those created by Oracle Rdb must be completely specified.

- ROUTINE

  Specify the name of a function or procedure, either explicitly or using a wildcard. Only user defined routines will be selected by this type of wildcard command. Those created by Oracle Rdb must be completely specified.

- MODULE

  Specify the name of a module, either explicitly or using a wildcard. Only user defined modules will be selected by this type of wildcard command. Those created by Oracle Rdb must be completely specified. Note also that routines defined with the clause USAGE IS LOCAL will not be selected by wildcards. Such routines can only be activated by routines in the same module.

  The MODULE keyword provides a time saving shortcut for auditing related routines. In a similar way that routine access control is inherited from the containing module, audit and alarm settings are inherited from the owning module when a routine is first referenced. When a subsequent ALTER MODULE ... ADD FUNCTION, or ALTER MODULE ... ADD PROCEDURE statement is used, these new routines will also inherite audit and alarm settings from the module.

- VIEW

  Specify the name of a view, either explicitly or using a wildcard. Only user defined views will be selected by this type of wildcard command. Views created by Oracle Rdb must be completely specified.

  In prior releases, views could be marked for audit using the TABLE keyword. The TABLE keyword remains a superset of VIEW and selects both table and view objects. However, to perform wildcard selection of views you must use the VIEW keyword.

The object name parameter enables or disables DACCESS auditing for the specified object or objects. If you specify more than one object name, separate the object names with commas, and enclose the list of object names within parentheses.

- Privileges=(privilege-list)

If you specify one or more object names, you must select one or more privileges to audit. Use the Privileges=privilege-list qualifier to select the privileges that are to be audited for each of the objects in the object name list when the selected objects are accessed. The privileges that can be specified with the Privileges qualifier are listed in Table 1–14 and Table 1–15.

Privilege names SUCCESS and FAILURE can be used as a convenient way to specify that all successful or failed accesses to that object for all privileges should be audited. The privilege name All can be used with the Enable or Disable qualifier to turn on or turn off auditing for all privileges applicable to the object.

If you specify a privilege that does not apply to an object, Oracle Rdb allows it but will not produce any auditing for that privilege. You can specify only SQL privileges with the Privileges=(privilege-list) qualifier. The privileges that can be specified for each Oracle Rdb object type are shown in Table 1–14 and Table 1–15.

Table 1–14 DACCESS Privileges for Database Objects (Part 1)

SQL Privilege	Database	Table View	Column
ALTER	Y	Y	
CREATE	Y	Y	
DBADM	Y		
DBCTRL	Y	Y	
DELETE	Y	Y	
DISTRIBTRAN	Y		
DROP	Y	Y	
INSERT	Y	Y	
OPERATOR	Y		
REFERENCES		Y	Y
SECURITY	Y		
SELECT	Y	Y	
SHOW	Y	Y	Y
UPDATE	Y	Y	Y

(continued on next page)

Table 1–14 (Cont.) DACCESS Privileges for Database Objects (Part 1)

SQL Privilege	Database	Table View	Column
SUCCESS	Y	Y	Y
FAILURE	Y	Y	Y
ALL	Y	Y	Y

Table 1–15 DACCESS Privileges for Database Objects (Part 2)

SQL Privilege	Module	Function Procedure	Sequence
ALTER	Y	Y	Y
CREATE			
DBADM			
DBCTRL	Y	Y	Y
DELETE			
DISTRIBTRAN			
DROP	Y	Y	Y
EXECUTE	Y	Y	
INSERT			
OPERATOR			
REFERENCES	Y	Y	Y
SECURITY			
SELECT			Y
SHOW	Y	Y	Y
UPDATE			
SUCCESS	Y	Y	Y
FAILURE	Y	Y	Y
ALL	Y	Y	Y

- Identifier=(identifier-list)

  Enables or disables auditing of user access to objects listed in the Enable=Daccess=object-type qualifier. If you do not specify this option, no users are audited for the DACCESS event. Any user whose identifier

you specify is audited for accessing the database objects with the privileges specified.

You can specify wildcard characters within the identifiers to identify groups of users. The [*,*] identifier indicates public, and causes all users to be audited. The * identifier matches all rights identifiers (roles) and user. If you specify a nonexistent identifier, you receive an error message.

The order of identifiers in the identifier list is not significant. A user is audited if he or she holds any of the identifiers specified in the identifier list.

You can specify user identification code (UIC) identifiers, general identifiers, and system-defined identifiers in the identifier list. For more information on identifiers, see the *Oracle Rdb Guide to Database Design and Definition*.

If you specify more than one identifier, separate the identifiers with commas, and enclose the identifier list within parentheses. UIC identifiers with commas such as [RDB,JONES] must be enclosed within quotation marks as follows:

```
IDENTIFIER=(INTERACTIVE,"[RDB,JONES]",SECRETARIES)
```

When you use Identifier=(identifier-list) to specify one or more identifiers to be audited, those identifiers are audited whenever they access any object for which auditing has been enabled.

- Protection

  Allows you to audit changes made to access privilege sets for database objects by means of the SQL GRANT and REVOKE statements.

- Rmu

  Audits the use of Oracle RMU commands by users with the privilege to use them.

Every
Noevery

Sets the granularity of DACCESS event auditing for the database. When you specify the Every qualifier, every access check for the specified objects using the specified privilege or privileges during a database attachment is audited. When you specify the Noevery qualifier, each user's first access check for the specified audit objects using the specified privilege or privileges during a database attachment is audited. The First qualifier is a synonym for the Noevery qualifier; the two qualifiers can be used interchangeably.

The default is the Every qualifier.

First
Specifies that when DACCESS event auditing is enabled, each user's first access check for the specified audit objects using the specified privilege or privileges during a database attachment is audited. The First qualifier is a synonym for the Noevery qualifier; the two qualifiers can be used interchangeably.

Flush
Noflush
Indicates whether forced writes of audit journal records are currently enabled for the database. Forced writes will cause Oracle Rdb to write (flush) the audit journal record immediately out to disk when the audit record is produced, rather than waiting for the audit server to flush the audit records at specified intervals of seconds.

The default is the Noflush qualifier, which flushes audit records every interval of seconds. To specify the interval, use the DCL command SET AUDIT/INTERVAL=JOURNAL_FLUSH=time.

Start
Starts Oracle Rdb security auditing for the database. The Start qualifier by itself starts both security alarms and security audit journal records. Also, you can supply the Type=Alarm qualifier or the Type=Audit qualifier to start security alarms only or security audit journaling only.

When you specify the Start qualifier, auditing starts immediately for all audit event classes that are currently enabled. Any subsequent audit events of currently attached users are recorded in the security audit journal, or alarms are sent to security-enabled terminals, or both, depending on what you have specified for your database.

Stop
Stops Oracle Rdb security auditing for the database. The Stop qualifier by itself stops both security alarms and security audit journal records. Also, you can supply the Type=Alarm qualifier or the Type=Audit qualifier to stop security alarms only or security audit journaling only.

When you specify the Stop qualifier, the alarms or audits (or both) of all audit event classes are immediately stopped (depending on whether you specified the Type=Alarm qualifier, the Type=Audit qualifier, or neither). The audit event classes previously specified with the Enable qualifier remain enabled, and you can start them again by using the Start qualifier.

Type=option
Specifies that security alarms or security audit journal records (or both) be enabled or disabled. The following options are available with the Type qualifier:

- Alarm

  Causes subsequent qualifiers in the command line (Start, Stop, Enable, and Disable) to generate or affect security alarm messages that are sent to all terminals enabled as security operator terminals.

- Audit

  Causes subsequent qualifiers in the command line (Start, Stop, Enable, and Disable) to generate or affect security audit journal records that are recorded in the security audit journal file.

  If you do not specify the Type qualifier with the RMU Set Audit command, Oracle RMU enables or disables both security alarms and security audit journal records.

Usage Notes

- To use the RMU Set Audit command for a database, you must have the RMU$SECURITY privilege in the root file ACL for the database or the OpenVMS SECURITY or BYPASS privilege.
- There is very little overhead associated with security auditing; no extra disk I/O is involved. Instead, the audit information is handled by the OpenVMS AUDIT server (a separate process) which writes them to the audit journal.
- Audit journal records collected on a database and stored in the OpenVMS audit journal can be loaded into a database using the RMU Load command with the Audit qualifier. The target database is determined by qualifiers on RMU Load Audit command.
- You can use the Daccess=object-type option to enable DACCESS checking for specific objects, but the general DACCESS class is not enabled until you explicitly enable it by using the Enable=Daccess qualifier with the RMU Set Audit command. Also, you need to use the Start qualifier with the RMU Set Audit command to start the auditing and alarms that have been enabled.
- Alarms are useful for real-time tracking of auditing information. At the moment an alarm occurs, text messages regarding the alarm are displayed on security-enabled terminals.

To enable a terminal to receive Oracle Rdb security alarms, enter the DCL REPLY/ENABLE=SECURITY command. You must have both the OpenVMS SECURITY and OpenVMS OPER privileges to use the REPLY/ENABLE=SECURITY command.

- Audit records are useful for periodic reviews of security events. Audit records are stored in a security audit journal file, and can be reviewed after they have been loaded into a database table with the RMU Load command with the Audit qualifier. Use the DCL SHOW AUDIT/JOURNAL command to determine the security audit journal file being used by your database.
- The AUDIT class is always enabled for both alarms and audit records, but does produce any alarms or audit records until auditing is started. The AUDIT class cannot be disabled.
- When you specify the Daccess=object-type option and one or more other options in an options list, the Privileges=(privilege-list) qualifier must begin after the closing parenthesis for the options list.
- TABLE=* and COLUMN=*.* are accepted to specify all tables or all columns of all tables.
- Table and column names may also contain OpenVMS style wildcards: "*" and "%". For instance, in the PERSONNEL database, JOB* will match both the JOBS and JOB_HISTORY table and *HISTORY.EMPLOYEE_ID will match all EMPLOYEE_ID columns in any table ending in HISTORY which includes both the JOB_HISTORY and SALARY_HISTORY tables.
- When using wildcard characters with the /ENABLE=DACCESS or /DISABLE=DACCESS option, only user defined objects will be selected.
  - TABLE - only user defined tables will be matched by the wildcard search. Views, system tables and special tables created by OCI Services for Rdb will not be returned. For excluded tables, you must specify their full name on the RMU command line.
  - SEQUENCE - only user defined sequences will be matched by the wildcard search. This includes sequences implicity created for IDENTITY columns. For excluded sequences, you must specify their full name on the RMU command line.
  - MODULE - only user defined modules will be matched by the wildcard search. For excluded modules, you must specify their full name on the RMU command line.

  - ROUTINE - only user defined routines will be matched by the wildcard search. Any routine defined in a module with USAGE IS LOCAL will be excluded from this matching. Such routines can only be called from within the module itself and so additional auditing is not required. For excluded routines, you must specify their full name on the RMU command line.
  - VIEW - only user defined views will be matched by the wildcard search. Base tables, system views, and special views created by OCI Services for Rdb will not be returned. For excluded views, you must specify their full name on the RMU command line.

- To display the results of an RMU Set Audit command, enter the RMU Show Audit command.
- You can use the Disable and Enable qualifiers with indirect file references. See Section 1.3 for more information.
- When the RMU Set Audit command is issued for a closed database, the command executes without other users being able to attach to the database.

Examples

Example 1

In the following example, the first command enables alarms for the RMU and PROTECTION classes. The second command shows that alarms for the RMU and PROTECTION classes are enabled but not yet started. The AUDIT class is always enabled and cannot be disabled. The third command starts alarms for the RMU and PROTECTION classes. The fourth command shows that alarms for the RMU and PROTECTION classes are enabled and started.

```
$ ! Enable alarms for RMU and PROTECTION classes:
$ RMU/SET AUDIT/TYPE=ALARM/ENABLE=(RMU,PROTECTION) MF_PERSONNEL
$ !
$ ! Show that alarms are enabled, but not yet started:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STOPPED for:
    PROTECTION (enabled)
    RMU (enabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access
```

```
Enabled identifiers:
    None

$ ! Start alarms for the enabled RMU and PROTECTION classes:
$ RMU/SET AUDIT/START/TYPE=ALARM MF_PERSONNEL
$ !
$ ! Show that alarms are started for the RMU and PROTECTION classes:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STARTED for:
    PROTECTION (enabled)
    RMU (enabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    None
```

Example 2

In this example, the first command shows that alarms are started and enabled for the RMU class. The second command disables alarms for the RMU class. The third command shows that alarms for RMU class are disabled.

```
$ ! Show that alarms are enabled and started for the RMU class:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STARTED for:
    PROTECTION (disabled)
    RMU (enabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    None
```

```
$ ! Disable alarms for the RMU class:
$ RMU/SET AUDIT/TYPE=ALARM/DISABLE=RMU MF_PERSONNEL
$ !
$ ! Show that alarms are disabled for the RMU class:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STARTED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    None
```

Example 3

In this example, the first command enables auditing for users with the [SQL,USER1] and [RDB,USER2] identifiers. The second command shows the enabled identifiers. The third command enables DACCESS checks requiring SELECT and INSERT privileges for the EMPLOYEES and COLLEGES tables. The fourth command displays the DACCESS checks that have been specified for the COLLEGES and EMPLOYEES tables. Note that because the general DACCESS type has not been enabled, DACCESS for the EMPLOYEES and COLLEGES tables is displayed as disabled.

```
$ ! Enable auditing for users with the [SQL,USER1] and
$ ! [RDB,USER2] identifiers:
$ RMU/SET AUDIT/ENABLE=IDENTIFIER=("[SQL,USER1]","[RDB,USER2]") -
_$ MF_PERSONNEL
$ !
$ ! Show that [SQL,USER1] and [RDB,USER2] are enabled identifiers:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access
```

```
Enabled identifiers:
    (IDENTIFIER=[SQL,USER1])
    (IDENTIFIER=[RDB,USER2])

$ ! Enable and start DACCESS checks for the SELECT and INSERT
$ ! privileges for the COLLEGES and EMPLOYEES tables:
$ RMU/SET AUDIT/ENABLE=DACCESS=TABLE=(COLLEGES,EMPLOYEES) -
_$ /PRIVILEGES=(SELECT,INSERT)/START MF_PERSONNEL
$ !
$ ! Display the DACCESS checks that are enabled and
$ ! started for the COLLEGES and EMPLOYEES tables:
$ RMU/SHOW AUDIT/DACCESS=TABLE MF_PERSONNEL
Security auditing STARTED for:
    DACCESS (disabled)
        TABLE : EMPLOYEES
            (SELECT,INSERT)
        TABLE : COLLEGES
            (SELECT,INSERT)

Security alarms STARTED for:
    DACCESS (disabled)
        TABLE : EMPLOYEES
            (SELECT,INSERT)
        TABLE : COLLEGES
            (SELECT,INSERT)
```

1.55 RMU Set Audit Command

Example 4

In this example, the first command enables auditing of the JOBS and EMPLOYEES tables for DACCESS checks for users with the [SQL,USER1] or BATCH identifier. The Privileges=All qualifier specifies that auditing will be produced for every privilege. The second command shows that auditing is enabled for users with the [SQL,USER1] or BATCH identifier. The third command shows that DACCESS checking for the JOBS and EMPLOYEES tables for all privileges is specified. The fourth command enables the general DACCESS class. The fifth command's output shows that the general DACCESS class is now enabled. The sixth command starts the auditing that is enabled, and the seventh command shows that the enabled auditing is started.

```
$ ! Enable DACCESS checks for users with the [SQL,USER1] or
$ ! BATCH identifier for the JOBS and EMPLOYEES tables:
$ RMU/SET AUDIT/TYPE=AUDIT -
_$ /ENABLE=(IDENTIFIER=("[SQL,USER1]",BATCH), -
_$ DACCESS=TABLE=(JOBS,EMPLOYEES)) /PRIVILEGES=ALL MF_PERSONNEL
$ !
$ ! Show that auditing is enabled for users with the [SQL,USER1]
$ ! or BATCH identifiers:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Security alarms STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    (IDENTIFIER=[SQL,USER1])
    (IDENTIFIER=BATCH)

$ ! Show that DACCESS checking for all privileges for the
$ ! JOBS and EMPLOYEES tables is enabled:
$ RMU/SHOW AUDIT/DACCESS=TABLE MF_PERSONNEL
Security auditing STOPPED for:
    DACCESS (disabled)
        TABLE : EMPLOYEES
            (ALL)
        TABLE : JOBS
            (ALL)

Security alarms STOPPED for:
    DACCESS (disabled)
```

```
$ ! Enable the general DACCESS class:
$ RMU/SET AUDIT/ENABLE=DACCESS MF_PERSONNEL
$ !
$ ! Show that the general DACCESS class is enabled:
$ RMU/SHOW AUDIT/DACCESS=TABLE MF_PERSONNEL
Security auditing STOPPED for:
    DACCESS (enabled)
        TABLE : EMPLOYEES
            (ALL)
        TABLE : JOBS
            (ALL)

Security alarms STOPPED for:
    DACCESS (enabled)


$ ! Start the auditing that is enabled:
$ RMU/SET AUDIT/START MF_PERSONNEL
$ !
$ ! Show that the enabled auditing is started:
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STARTED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (enabled)

Security alarms STARTED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    DACCESS (enabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    (IDENTIFIER=[SQL,USER1])
    (IDENTIFIER=BATCH)
```

Example 5

In this example, the first command enables DACCESS checks requiring the INSERT privilege for the mf_personnel database, for the EMPLOYEES table, and for the EMPLOYEE_ID column of the EMPLOYEES table. The second command shows that the DACCESS check for the INSERT privilege is enabled for the specified objects.

```
$ ! Enable a DACCESS check for the INSERT privilege for the
$ ! MF_PERSONNEL database, EMPLOYEES table, and EMPLOYEE_ID
$ ! column of the EMPLOYEES table:
$ RMU/SET AUDIT -
_$ /ENABLE=DACCESS=(DATABASE,TABLE=EMPLOYEES, -
_$ COLUMN=EMPLOYEES.EMPLOYEE_ID) -
_$ /PRIVILEGES=(INSERT) MF_PERSONNEL
```

```
$ !
$ ! Show that the DACCESS check for the INSERT privilege is
$ ! enabled for the specified objects. (The general DACCESS
$ ! class remains disabled until you issue an
$ ! RMU/SET AUDIT/ENABLE=Daccess command without specifying
$ ! any object-type parameter to the Daccess option.
$ ! See the fourth Oracle RMU command in Example 4.)
$ !
$ RMU/SHOW AUDIT/DACCESS=(DATABASE,TABLE,COLUMN) MF_PERSONNEL
Security auditing STOPPED for:
    DACCESS (disabled)
        DATABASE
            (INSERT)
        TABLE : EMPLOYEES
            (INSERT)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (INSERT)

Security alarms STOPPED for:
    DACCESS (disabled)
        DATABASE
            (INSERT)
        TABLE : EMPLOYEES
            (INSERT)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (INSERT)
```

Example 6

In this example, the first command enables a DACCESS check requiring the INSERT privilege for the EMPLOYEES and COLLEGES tables, as well as for the EMPLOYEE_ID and LAST_NAME columns of the EMPLOYEES table and the COLLEGE_CODE column of the COLLEGES table in the mf_personnel database. The second command shows that the DACCESS check for the INSERT privilege is enabled for the specified objects.

```
$ ! Enable a DACCESS check for the INSERT privilege for the
$ ! EMPLOYEES and COLLEGES table, the LAST_NAME and EMPLOYEE_ID
$ ! column of the EMPLOYEES table, and the COLLEGE_CODE column
$ ! of the COLLEGES table:
$ RMU/SET AUDIT -
_$ /ENABLE=DACCESS=(TABLE=(EMPLOYEES,COLLEGES), -
_$                  COLUMN=(EMPLOYEES.EMPLOYEE_ID, -
_$                          EMPLOYEES.LAST_NAME, -
_$                          COLLEGES.COLLEGE_CODE)) -
_$ /PRIVILEGES=(INSERT) MF_PERSONNEL
$ !
$ ! Show that the DACCESS check for the INSERT privilege is
$ ! enabled for the specified objects. (The general DACCESS
$ ! class remains disabled until you issue an
$ ! RMU/SET AUDIT/ENABLE=Daccess command without specifying
$ ! any object-type parameter to the Daccess option.
$ ! See the fourth Oracle RMU command in Example 4.)
$ !
```

```
$ RMU/SHOW AUDIT/DACCESS=(DATABASE,TABLE,COLUMN) MF_PERSONNEL
Security auditing STOPPED for:
    DACCESS (disabled)
        DATABASE
            (NONE)
        TABLE : COLLEGES
            (INSERT)
        TABLE : EMPLOYEES
            (INSERT)
        COLUMN : COLLEGES.COLLEGE_CODE
            (INSERT)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (INSERT)
        COLUMN : EMPLOYEES.LAST_NAME
            (INSERT)

Security alarms STOPPED for:
    DACCESS (disabled)
        DATABASE
            (NONE)
        TABLE : COLLEGES
            (INSERT)
        TABLE : EMPLOYEES
            (INSERT)
        COLUMN : COLLEGES.COLLEGE_CODE
            (INSERT)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (INSERT)
        COLUMN : EMPLOYEES.LAST_NAME
            (INSERT)
```

Example 7

The following example shows the simplified commands for enabling auditing for important fields in the PERSONNEL database by using wildcards.

1.55 RMU Set Audit Command

```
$ rmu/set audit-
    /enable=daccess=column=(-
        *.employee_id,-
        *_history.*end,-
        *_history.*start)-
    /privileges=(insert,select,update,delete) personnel
$
$ rmu/show audit/daccess=(DATABASE,TABLE,COLUMN) personnel
Security auditing STOPPED for:
    DACCESS (disabled)
        DATABASE
            (NONE)
        COLUMN : DEGREES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.JOB_START
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.JOB_END
           (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : RESUMES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.SALARY_START
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.SALARY_END
            (SELECT,INSERT,UPDATE,DELETE)
```

```
Security alarms STOPPED for:
    DACCESS (disabled)
        DATABASE
            (NONE)
        COLUMN : DEGREES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : EMPLOYEES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.JOB_START
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : JOB_HISTORY.JOB_END
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : RESUMES.EMPLOYEE_ID
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.EMPLOYEE_ID
           (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.SALARY_START
            (SELECT,INSERT,UPDATE,DELETE)
        COLUMN : SALARY_HISTORY.SALARY_END
            (SELECT,INSERT,UPDATE,DELETE)
```

1.56 RMU Set Buffer_Object Command

On a database basis, controls which database objects use the OpenVMS Fast I/O and Buffer Objects features.

Format

RMU/Set Buffer_Object root-file-spec

Command Qualifiers	Defaults
/Disable=enable-disable-options	See description
/Enable=enable-disable-options	See description
/[No]Log	Current DCL verify value

Description

Use the RMU Set Buffer_Object command to control, on a database basis, which database objects use the OpenVMS Fast I/O and Buffer Objects features.

Command Parameters

root-file-spec
The root file specification of the database. The default file extension is .rdb.

Command Qualifiers

Disable=enable-disable-options
Disables buffer objects for the specified Oracle Rdb buffers. You can specify one or more of the following buffer objects: Page, AIJ, RUJ, and Root. Refer to Table 1–16 for more information about these keywords. If you specify more than one object, separate the objects with commas, and enclose the list of objects within parentheses.

Enable=enable-disable-options
Enables buffer objects for the specified Oracle Rdb buffers. You can specify one or more of the following buffer objects: Page, AIJ, RUJ, and Root. Refer to Table 1–16 for more information about these keywords. If you specify more than one object, separate the objects with commas, and enclose the list of objects within parentheses.

If you specify the Enable and Disable qualifiers for the same buffer object, the Enable option prevails and the buffer object state is enabled for the specified object type.

Table 1–16 Buffer Object Control

Object	Keyword	Logical Name
Data pages	PAGE	RDM$BIND_PAGE_BUFOBJ_ENABLED
AIJ output	AIJ	RDM$BIND_AIJ_BUFOBJ_ENABLED
RUJ	RUJ	RDM$BIND_RUJ_BUFOBJ_ENABLED
Root file	ROOT	RDM$BIND_ROOT_BUFOBJ_ENABLED

Note

If a logical is defined as "1" then the corresponding buffer will be created as an OpenVMS buffer object.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch.

Usage Notes

- The Enable and Disable qualifiers are mutually exclusive.
- The RMU Set Buffer_Object command requires exclusive database access; that is, the database cannot be open or be accessed by other users.
- Buffer objects are memory resident and thus reduce the amount of physical memory available to OpenVMS for other uses. Buffer object use requires that the user be granted the VMS$BUFFER_OBJECT_USER rights identifier. The system parameter MAXBOBMEM needs to be large enough to allow all buffer objects for all users to be created. For further information regarding Fast I/O, consult the OpenVMS documentation.

Example

The following example demonstrates enabling ROOT buffer objects and disabling PAGE buffer objects. The RMU /DUMP /HEADER command is used to validate the change.

```
$RMU /SET BUFFER_OBJECT /ENABLE=(ROOT) /DISABLE=(PAGE) MF_PERSONNEL
%RMU-I-MODIFIED, Buffer objects state modified
%RMU-W-DOFULLBCK, full database backup should be done to ensure futur
$ RMU/DUMP/HEAD MF_PERSONNEL
    .
    .
    .
      - OpenVMS Alpha Buffer Objects are enabled for
          Root I/O Buffers
```

1.57 RMU Set Corrupt_Pages Command

Allows you to set pages, storage areas, and snapshot files as either corrupt or consistent in the corrupt page table (CPT). A corrupt page is one that contains meaningless data; an inconsistent page is one that contains old data (data that is not at the same transaction level as the database root file). Corrupt pages are logged to the CPT, which is maintained in the database root file. When the CPT becomes full (due to a large number of pages being logged), the area containing the most corrupt pages is marked as corrupt and the individual corrupt pages for that area are removed from the corrupt page table. The Oracle RMU Set Corrupt_Pages operation is an offline operation.

If you reset a page or storage area in the CPT to consistent it does not remove any true corruption or inconsistencies. However, if you reset a snapshot file in the CPT to consistent, Oracle RMU initializes the snapshot file and thus removes any true corruption or inconsistency.

Caution

Use the RMU Set Corrupt_Pages command only after you fully understand the internal data structure and know the information the database should contain. Setting a page in a storage area that is truly corrupt or inconsistent to consistent does not remove the corruption or inconsistency. Setting truly corrupt or inconsistent pages in a storage area to consistent and continuing to access those pages can result in unrecoverable corruptions to the database.

The RMU Restore and RMU Recover commands should be used first and should be part of your normal operating procedure.

Note

This command replaces two RdbALTER statements: MAKE CONSISTENT and UNCORRUPT. Both the RdbAlter statements, MAKE CONSISTENT and UNCORRUPT, are deprecated commands that may be removed in future versions.

1.57 RMU Set Corrupt_Pages Command

When a storage area is restored from backup files on a by-area basis, it does not reflect data that has been updated since the backup operation. The transaction level of the restored area reflects the transaction level of the backup file, not the transaction level of the database. Therefore, the transaction level of the restored area differs from that of the database. Oracle Rdb marks the area by setting a flag in the storage area file to inconsistent.

You can perform a recovery by area to upgrade the transaction level of the restored area to that of the database. (After-image journaling must be enabled in order to restore by area.) If you are certain that no updates have been made to the database since the backup operation, you can use the RMU Set Corrupt_Pages command to change the setting of the flag from inconsistent to consistent.

In addition, storage areas are corrupted by attempting an SQL rollback with one or more storage areas opened in batch-update transaction mode.

The RMU Set Corrupt_Pages command allows you to access a database that is in an uncertain condition. Accordingly, the following message and question are displayed when you enter it to correct a corrupt or inconsistent storage area or storage area page. (This message is not displayed if you enter it to correct a corrupt or inconsistent snapshot file.)

```
             ***** WARNING! *****

Marking a storage area or page consistent does not
remove the inconsistencies. Remove any inconsistencies
or corruptions before you proceed with this action.

Do you wish to continue? [N]
```

Format

RMU/Set Corrupt_Pages root-file-spec

Command Qualifiers	Defaults
/Area=identity	None
/Consistent	None
/Corrupt	None
/Disk=device	None
/Page=(n,...)	None

1.57 RMU Set Corrupt_Pages Command

Description

The RMU Set Corrupt_Pages command allows you to override the required RMU Recover command after a by-area restore operation. Although Oracle RMU cannot determine when the recover operation is superfluous, you might have that knowledge. If you are certain of this knowledge, you can abridge the requirement for the recover operation by using the RMU Set Corrupt_Pages command to set corrupt pages to consistent.

Similarly, sometimes you might know of a problem that Oracle RMU does not recognize. For example, you might find that a page contains an index node that causes a bugcheck dump each time it is accessed. You can use the RMU Set Corrupt_Pages command to mark this page as corrupt and then follow your usual procedure for recovering from database corruption.

Note that the RMU Set Corrupt_Pages command with the Consistent qualifier does not make truly corrupt storage area pages usable. Corrupt storage area pages detected during normal operation are logged in the CPT, and likely have an invalid checksum value. The RMU Set Corrupt_Pages command with the Consistent qualifier removes the specified pages from the CPT, but the next time a user tries to touch that storage area page, it is logged in the CPT again because it is still physically corrupt. To correct a storage area page that is truly corrupt, you must restore it from a backup file.

The RMU Set Corrupt_Pages command with the Consistent qualifier does make truly corrupt or inconsistent pages in a snapshot file usable. When you use this command and specify a snapshot file with the areas qualifier, Oracle RMU initializes the specified snapshot file.

Command Parameters

root-file-spec
The file specification of the database root file for which you want to set pages or areas to corrupt or consistent.

Command Qualifiers

Area=identity
Specifies a particular storage area file or snapshot file. The identity for a storage area can be either the area name (for example, EMPIDS_OVER), or a storage area ID number (for example, 5). The identity for a snapshot file must be the snapshot file ID number. Use the RMU Dump command with the Header qualifier to display the ID numbers associated with a storage area file or a snapshot file.

When you use the Area qualifier with the Page=(n,...) qualifier, the command specifies the named pages in the named storage area or snapshot file. When you specify the Area qualifier without the Page qualifier, the command specifies all pages of the specified storage area or snapshot file.

The Area qualifier cannot be used with the Disk qualifier.

Consistent
Specifies that the pages, areas, or snapshot files specified with the Page, Area, or Disk qualifier are to be considered consistent with the remainder of the database.

If you make a storage area or page in a storage area consistent while it is marked in the database as not corrupt, but inconsistent, you receive a warning and are required to confirm your request to carry out this operation before the operation will complete.

You cannot use the Consistent qualifier with the Corrupt qualifier.

Corrupt
Specifies that the pages, areas, or snapshot files specified with the Page, Area, or Disk qualifier are to be considered corrupt.

You cannot use the Corrupt qualifier with the Consistent qualifier.

Disk=device
Specifies all the pages, all the storage areas, and all the snapshot files on the named device be set as you indicate with the Corrupt or the Consistent qualifier.

You cannot use the Disk qualifier with the Page or the Area qualifier.

Page=(n,...)
Specifies the listed page numbers.

You must specify the Area qualifier when you use the Page qualifier.

You cannot use the Page qualifier with the Disk qualifier.

Usage Notes

- You must have the RMU$ALTER, RMU$BACKUP, or RMU$RESTORE privilege in the root file access control list (ACL) for a database or the OpenVMS SYSPRV or BYPASS privilege to use the RMU Set Corrupt_ Pages command for the database.

- You can issue the RMU Set Corrupt_Pages command while users are attached to the database.
- You must specify either the Corrupt or the Consistent qualifier (but not both) when you use the RMU Set Corrupt_Pages command.
- When you use the RMU Set Corrupt_Pages command to mark a page as corrupt or consistent, the database is marked as having been altered.

Examples

Example 1

The following command sets storage area EMPIDS_MID in the mf_personnel database as corrupt:

```
$ RMU/SET CORRUPT_PAGES/AREA=EMPIDS_MID/CORRUPT MF_PERSONNEL
%RMU-I-AREAMARKED, Area 4 was marked corrupt.
```

Example 2

The following command marks EMPIDS_MID as consistent. This is the area that was marked as corrupt in Example 1. However, in this case, instead of using the storage area name in the Oracle RMU command, the storage area identifier is used.

```
$ RMU/SET CORRUPT_PAGES/AREA=4/CONSISTENT MF_PERSONNEL

              ***** WARNING! *****

Marking a storage area or page consistent does not
remove the inconsistencies. Remove any inconsistencies
or corruptions before you proceed with this action.

Do you wish to continue? [N] Y
%RMU-I-AREAMARKED, Area 4 was marked consistent .
```

Example 3

The following command marks page 1 in area 3 in the mf_personnel database as corrupt. Using the RMU Show Corrupt_Pages command confirms that the page has been marked as expected.

```
$ RMU/SET CORRUPT_PAGES/AREA=3/PAGE=1/CORRUPT MF_PERSONNEL
%RMU-I-PAGEMARKED, Page 1 in area 3 was marked corrupt.
```

1.57 RMU Set Corrupt_Pages Command

```
$ RMU/SHOW CORRUPT_PAGES MF_PERSONNEL.RDB
*-------------------------------------------------------------------------
* Oracle Rdb V7.0-00                           3-JUL-1996 17:01:20.62
*
* Dump of Corrupt Page Table
*     Database: USER1:[DB]MF_PERSONNEL.RDB;1
*
*-------------------------------------------------------------------------

Entries for storage area EMPIDS_LOW
-----------------------------------

    Page 1
        - AIJ recovery sequence number is -1
        - Live area ID number is 3
        - Consistency transaction sequence number is 0:0
        - State of page is: corrupt

*-------------------------------------------------------------------------
* Oracle Rdb V7.0-00                           3-JUL-1996 17:01:20.82
*
* Dump of Storage Area State Information
*     Database: USER1:[DB]MF_PERSONNEL.RDB;1
*
*-------------------------------------------------------------------------

All storage areas are consistent.
```

Example 4

The following example sets page 4 of the snapshot file for EMPIDS_OVER to consistent. Because Oracle RMU initializes snapshot files specified with the Set Corrupt_Pages command, the snapshot file is removed from the corrupt page table and is now usable.

```
$ RMU/SET CORRUPT_PAGES MF_PERSONNEL.RDB/AREA=14/PAGE=3/CONSISTENT
%RMU-I-PAGEMARKED, Page 3 in area 14 was marked consistent.
```

1.58 RMU Set Database Command

Allows you to alter the database-allowed transaction modes without marking the database as modified.

Format

RMU/Set Database root-file-spec

Command Qualifiers	Defaults
/Transaction_Mode=[mode,...]	See description

Description

The RMU /SET command "DATABASE /TRANSACTION_MODE=(...)" allows altering of the database-allowed transaction modes without marking the database as modified. This command is intended to be used to set the transaction modes allowed on a standby database. This command requires exclusive database access (the database cannot be open or be accessed by other users).

Because only read-only transactions are allowed on a standby database, you may wish to use the TRANSACTION_MODE=READ_ONLY qualifier setting on a standby database. This setting prevents modifications to the standby database at all times, even when replication operations are not active.

This command requires the RMU$ALTER privilege.

Command Parameters

root-file-spec
The file specification of the database root file for which you want to specify the database transaction mode.

Command Qualifiers

Transaction_Mode=[mode,...]
The transaction mode to set the standby database to. This command requires exclusive database access (the database cannot be open or be accessed by other users).

1.58 RMU Set Database Command

Valid keywords for the RMU /SET DATABASE /TRANSACTION_MODE=(...) qualifier are:

- ALL - Enables all transaction modes
- CURRENT - Enables all transaction modes that are set in the database
- NONE - Disables all transaction modes
- [NO]BATCH_UPDATE
- [NO]READ_ONLY
- [NO]EXCLUSIVE
- [NO]EXCLUSIVE_READ
- [NO]EXCLUSIVE_WRITE
- [NO]PROTECTED
- [NO]PROTECTED_READ
- [NO]PROTECTED_WRITE
- [NO]READ_WRITE
- [NO]SHARED
- [NO]SHARED_READ
- [NO]SHARED_WRITE

If you specify more than one transaction mode in the mode-list, enclose the list in parenthesis and separate the transaction modes from one another with a comma. Note the following:

- When you specify a negated transaction mode, it indicates that a mode is not an allowable access mode. For example, if you specify the Noexclusive_ Write access mode, it indicates that exclusive write is not an allowable access mode for the restored database.
- If you specify the Shared, Exclusive, or Protected transaction mode, Oracle RMU assumes you are referring to both reading and writing in that transaction mode.
- No mode is enabled unless you add that mode to the list or you use the All option to enable all transaction modes.
- You can list one transaction mode that enables or disables a particular mode followed by another that does the opposite.

For example, /TRANSACTION_MODE=(NOSHARED_WRITE, SHARED) is ambiguous because the first value disables Shared_Write access and the second value enables Shared_Write access. Oracle RMU resolves the ambiguity by first enabling the modes as specified in the modes-list and then disabling the modes as specified in the modes-list. The order of items in the list is irrelevant. In the example presented previously, Shared_Read is enabled and Shared_Write is disabled.

1.59 RMU Set Galaxy Command

Allows you to enable or disable the database utilization of an OpenVMS Galaxy configuration without requiring that the database be open.

Format

RMU/Set Galaxy root-file-spec

Command Qualifiers	Defaults
/Disable	See description
/Enable	See description
/[No]Log	Current DCL verify value

Description

Use this command to enable or disable Galaxy features on an Oracle Rdb database. Databases opened on multiple copies of the OpenVMS operating system within a Galaxy system can share, in memory, database structures including global buffers, row caches, and root file objects.

Command Parameters

root-file-spec
The root file specification of the database. The default file extension is .rdb.

Command Qualifiers

Disable
Specifies that Galaxy features are to be disabled for the database.

Enable
Specifies that Galaxy features are to be enabled for the database.

Log
Nolog
Displays a log message at the completion of the RMU Set Galaxy operation.

Usage Notes

- The Enable and Disable qualifiers are mutually exclusive.
- The RMU Set Galaxy command requires exclusive database access; that is, the database cannot be open or be accessed by other users.

Example

The following example enables the Galaxy features for the specified database.

```
$ RMU /SET GALAXY /ENABLE root-file-spec
```

1.60 RMU Set Global_Buffers Command

Allows you to control the database global buffers feature without requiring that the database be open.

Format

RMU/Set Global_Buffers root-file-spec

Command Qualifiers	Defaults
/Disabled	None
/Enabled	None
/Large_Memory={Enabled|Disabled}	None
/Log	None
/Number=n	None
/User_Limit=n	None

Description

If you move a database from one system to another, or when memory usage or system parameters change, you may have to modify the global buffer parameters for a database when it is not possible to open the database. This situation could occur, for example, if you have insufficient available physical or virtual memory.

The RMU Set Global_Buffers command allows you to alter some of the global buffer-related parameters without opening the database. This allows you to reconfigure the database so that it can be opened and accessed on the system.

Command Parameters

root-file-spec
Specifies the database root file for which you want to modify the global buffers feature.

Command Qualifiers

Disabled
Disables global buffers for the specified database.

Enabled
Enables global buffers for the specified database.

Large_Memory={Enabled | Disabled}
Large_Memory=Enabled enables global buffers in large memory (VLM).

Large_Memory=Disabled disables global buffers in large memory (VLM).

Log
Displays a log message at the completion of the RMU Set Global_Buffers command.

Number=n
Sets the number of global buffers.

User_Limit=n
Sets the global buffers user limit value.

Usage Notes

- This command requires exclusive database access (the database cannot be open or accessed by other users).
- The Enabled and Disabled qualifiers are mutually exclusive.
- The Large_Memory=Enabled and Large_Memory=Disabled qualifiers are mutually exclusive.
- Changes made by the RMU Set Global_Buffers command are not journaled. You should make a subsequent full database backup to ensure recovery.
- When global buffers are set to reside in large memory (Large_Memory=Enabled), the process that opens the database must be granted the VMS$MEM_RESIDENT_USER rights identifier. Oracle Corporation recommends that you use the RMU Open command when you utilize this feature.

1.61 RMU Set Logminer Command

Allows you to change the LogMiner state of a database.

Format

RMU/Set Logminer root-file-spec

Command Qualifiers	Defaults
/Continuous	/NoContinuous
/Disable	See description
/Enable	See description
/[No]Log	Current DCL verify value

Description

Use this command to enable or disable LogMiner operations on an Oracle Rdb database. When LogMiner is enabled, the Oracle Rdb database software writes additional information to the after-image journal file when records are added, modified, and deleted from the database. This information is used during the unload operation.

Command Parameters

root-file-spec
The root file specification of the database. The default file extension is .rdb.

Command Qualifiers

Continuous
NoContinuous
Enables the database for the Continuous LogMiner feature when used in conjunction with the Enable qualifier. Use the NoContinuous qualifier with the Enable qualifier to disable use of the Continuous LogMiner feature.

The RMU Set Logminer /Disable command explicitly disables the Continuous LogMiner feature as well as the base LogMiner functionality. To enable the Continuous LogMiner feature again, the entire RMU Set Logminer /Enable /Continuous command must be used.

Disable
Specifies that LogMiner operations are to be disabled for the database. When LogMiner is disabled, the Oracle Rdb software does not journal information required for LogMiner operations. When LogMiner is disabled for a database, the RMU Unload After_Journal command is not functional on that database.

Enable
Specifies that LogMiner operations are to be enabled for the database. When LogMiner is enabled, the Oracle Rdb database software logs additional information to the after-image journal file. This information allows LogMiner to extract records. The database must already have after-image journaling enabled.

Log
Nolog
Specifies that the setting of the LogMiner state for the database be reported to SYS$OUTPUT. The default is the setting of the DCL VERIFY flag, which is controlled by the DCL SET VERIFY command.

Usage Notes

- To use the RMU Set Logminer command, you must have the RMU$BACKUP, RMU$RESTORE, or RMU$ALTER privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- The RMU Set Logminer command requires offline access to the database. The database must be closed and no other users may be accessing the database.
- Execute a full database backup operation after issuing an RMU Set Logminer command that displays the RMU-W-DOFULLBCK warning message. Immediately after enabling LogMiner, you should perform a database after-image journal backup using the RMU Backup After_Journal command.

Examples

Example 1

The following example enables a database for LogMiner for Rdb operation.

```
$ RMU /SET LOGMINER /ENABLE OLTPDB.RDB
```

1.62 RMU Set Privilege Command

Allows you to modify the root file access control list (ACL) for a database.

A database's root file ACL determines the Oracle RMU commands that users can execute for the associated database.

Format

RMU/Set Privilege root-file-spec

Command Qualifiers	Defaults
/Acl[=(ace[,...])]	See description
/Acl_File=filename	See description
/After=ace	See description
/Delete[=All]	See description
/Edit	No editor invoked
/[No]Journal[=file-spec]	/Journal
/Keep[=(Recovery_Journal)]	See description
/Like=source-root-file-spec	None
/[No]Log	/Nolog
/Mode=[No]Prompt	/Mode=Prompt
/New	None
/[No]Recover[=file-spec]	/Norecover
/Replace=(ace[,...])	None

Description

The RMU Set Privilege command allows you to manipulate an entire root file ACL, or to create, modify, or delete access control entries (ACEs) in a root file ACL. See the *Oracle Rdb Guide to Database Design and Definition* for introductory information on ACEs and ACLs.

Use the RMU Set Privilege command to add ACEs to a root file ACL by specifying the ACEs with the Acl qualifier.

Table 1–1 shows the privileges a user must have to access each Oracle RMU command.

If the database root file you specify with RMU Set Privilege command does not have an ACL, Oracle RMU creates one.

1.62 RMU Set Privilege Command

The RMU Set Privilege command provides the following qualifiers to manipulate ACEs and ACLs in various ways:

- After
- Delete
- Like
- New
- Replace

By default, any ACEs you add to a root file ACL are placed at the top of the ACL. Whenever Oracle RMU receives a request for Oracle RMU access for a database that has a root file ACL, it searches each entry in the ACL from the first to the last for the first match it can find, and then stops searching. If another match occurs further down in the root file ACL, it has no effect. Because the position of an ACE in a root file ACL is so important, you can use the After qualifier to correctly position an ACE. When you use the After qualifier, any additional ACEs are added after the specified ACE.

You can delete ACEs from an ACL by including the Delete qualifier and specifying the ACEs with the Acl qualifier. To delete all the ACEs, include the Delete qualifier and specify the Acl qualifier without specifying any ACEs.

You can copy an ACL from one root file to another by using the Like qualifier. The ACL of the root file specified with the Like qualifier replaces the ACL of the root file specified with the root-file-spec parameter.

Use the New qualifier to delete all ACEs before adding any ACEs specified by the Acl, Like, or Replace qualifiers.

You can replace existing ACEs in a root file ACL by using the Replace qualifier. Any ACEs specified with the Acl qualifier are deleted and replaced by those specified with the Replace qualifier.

The existing ACE can be abbreviated when you use the Delete, Replace, or After qualifiers.

Use the RMU Set Privilege command with the Edit qualifier to invoke the ACL editor. You can specify the following qualifiers only when you specify the Edit qualifier also:

- Journal
- Keep
- Mode
- Recover

For more information on the ACL editor, see the OpenVMS documentation set.

1.62 RMU Set Privilege Command

Command Parameters

root-file-spec
The root file for the database whose root file ACL you are modifying.

Command Qualifiers

Acl[=(ace[,...])]
Specifies one or more ACEs to be modified. When no ACE is specified, the entire ACL is affected. Separate multiple ACEs with commas. When you include the Acl qualifier, the specified ACEs are inserted at the top of the ACL unless you also specify the After qualifier. You cannot specify the Acl qualifier and the Acl_File qualifier on the same RMU command line.

The format of an ACE is as follows:

(Identifier=user-id, Access=access_mask)

The user-id must be one of the following types of identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A user identification code (UIC) in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP
- Wildcard characters in [*,*] format

Names are not case sensitive. In addition, the Identifier and Access keywords can be abbreviated to one character. For example, the following ACE is valid:

(I=isteward, A=RMU$ALL)

The access_mask can be any of the following:

- One or more of the Oracle RMU privileges listed in Table 1–1

  If more than one privilege is specified, a plus sign (+) must be placed between the privileges.
- The keyword RMU$ALL

  These keywords indicate that you want the user to have all of the RMU privileges. (This keyword has no effect on system file privileges.)

- The keyword None

  This keyword indicates that you do not want the user to have any RMU or OpenVMS privileges. If you specify Acl=(id=username, access=READ+NONE), the specified user will have no RMU privileges and no READ privileges for the database files.

Acl_File=filename
Specifies a file containing a list of ACEs, with one ACE specified per line. You can use continuation characters to continue an ACE on the next line, and you can include commented lines within the file. Within this file, use the dash (-) as a continuation character and the exclamation point (!) to indicate a comment.

You cannot specify the Acl_File qualifier and the Acl qualifier on the same RMU command line.

After=ace
Indicates that all ACEs specified with the Acl qualifier are to be added after the ACE specified with the After qualifier. By default, any ACEs added to the ACL are always placed at the top of the list.

You cannot use this qualifier with the Edit qualifier.

Delete[=All]
Indicates that the ACEs specified with the Acl qualifier are to be deleted. If no ACEs are specified with the Acl qualifier, the entire ACL is deleted. If you specify an ACE that was not specified with the Acl qualifier, you are notified that the ACE does not exist, and the delete operation continues.

You cannot use this qualifier with the Edit qualifier.

Edit
Invokes the ACL editor and allows you to use the Journal, Keep, Mode, or Recover qualifiers. Oracle RMU ignores any other qualifiers you specify with the Edit qualifier.

The RMU Set Privilege command with the Edit qualifier only functions off line. If you attempt it on line, an error message is generated. This restriction is necessary because the ACL editor requests exclusive write access to the database.

To use the Edit qualifier, the SYS$SHARE:ACLEDTSHR.EXE image must be installed at system startup time, or, be installed by RMONSTART.COM. Contact your system manager if this image is not installed as needed.

For more information on the ACL editor, see the OpenVMS documentation set.

1.62 RMU Set Privilege Command

Journal[=file-spec]
Nojournal
Controls whether a journal file is created from the editing session. By default, a journal file is created if the editing session ends abnormally.

If you omit the file specification, the journal file has the same name as the root file and a file type of .tjl. You can use the Journal qualifier to specify a journal file name that is different from the default. No wildcard characters are allowed in the Journal qualifier file-spec parameter.

You must specify the Edit qualifier to use this qualifier.

Keep[=(Recovery,Journal)]
Determines whether the journal file, the recovery file, or both, are deleted when the editing session ends. The options are:

- Recovery–Saves the journal file used for restoring the ACL.
- Journal–Saves the journal file for the current editing session.

You can shorten the Journal and Recover options to J and R, respectively. If you specify only one option, you can omit the parentheses.

You must specify the Edit qualifier to use this qualifier. If you specify the Edit qualifier but do not specify the Keep qualifier, both the journal file for the current editing session and the journal file used for restoring the ACL are deleted when the editing session ends.

Like=source-root-file-spec
Indicates that the ACL of the root file specified with the Like qualifier is to replace the ACL of the root file specified with the root-file-spec parameter of the RMU Set Privilege command. Any existing ACEs are deleted before the root file ACL specified by the Like qualifier is copied.

You cannot use this qualifier with the Edit qualifier.

Log
Nolog
Directs the RMU Set Privilege command to return both the name of the root file that has been modified by the command and the ACL associated with the database. The default of Nolog suppresses this output.

You cannot use this qualifier with the Edit qualifier.

Mode=[No]Prompt
Determines whether the ACL editor prompts for field values. By default, the ACL editor selects prompt mode.

You must specify the Edit qualifier to use this qualifier.

New
Indicates that any existing ACE in the ACL of the root file specified with RMU Set Privilege is to be deleted. To use the New qualifier, you must specify a new ACL or ACE with the Acl, Like, or Replace qualifiers.

You cannot use this qualifier with the Edit qualifier.

Recover[=file-spec]
Norecover
Specifies the name of the journal file to be used in a recovery operation. If the file specification is omitted with the Recover qualifier, the journal is assumed to have the same name as the root file and a file type of .tjl. No wildcard characters are allowed with the Recover qualifier file-spec parameter.

The default is the Norecover qualifier, where no recovery is attempted when you invoke the ACL editor to edit a root file ACL.

You must specify Edit to use this qualifier.

Replace=(ace[,...])
Deletes the ACEs specified with the Acl qualifier and replaces them with those specified with the Replace qualifier. Any ACEs specified with the Acl qualifier must exist and must be specified in the order in which they appear in the ACL.

This qualifier cannot be used with the Edit qualifier.

Usage Notes

- You must have the RMU$SECURITY privilege in the root file ACL for a database or the OpenVMS SECURITY or BYPASS privilege to use the RMU Set Privilege command for the database. The RMU$SECURITY access is VMS BIT_15 access in the ACE. You can grant yourself BIT_15 access by using the DCL SET ACL command if you have (READ+WRITE+CONTROL) access.
- By default, a root file ACL is created for every Oracle Rdb database. In some cases, the root file ACL may not allow the appropriate Oracle RMU access for the database to all Oracle RMU users. In these situations, you must use the RMU Set Privilege command to modify the root file ACL to give the appropriate Oracle RMU access to Oracle RMU users. Table 1–1 shows the privileges required to access each Oracle RMU command.

1.62 RMU Set Privilege Command

- The root file ACL created by default on each Oracle Rdb database controls only a user's Oracle RMU access to the database (by specifying privileges that will allow a user or group of users access to specific Oracle RMU commands). Root file ACLs do not control a user's access to the database with SQL statements.

  A user's access to a database with SQL statements is governed by the privileges granted to the user in the database ACL (the ACL that is displayed using the SQL SHOW PROTECTION ON DATABASE command).

- If you find that the root file ACL has changed, or is not set as expected, it may be because a layered product has manipulated the OpenVMS directory or file ACLs. This can result in the unintentional alteration of an Oracle RMU access right.

  For example, Oracle CDD/Repository may use the following ACE:

  ```
  (IDENTIFIER=[*,*],OPTIONS=DEFAULT+PROPAGATE,ACCESS=NONE)
  ```

  If this ACE is propagated to an Oracle Rdb database, such as CDD$DATABASE or CDD$TEMPLATE, OpenVMS privileges may be required to manage that database. Or, you can use the RMU Set Privilege command to change the ACL on the affected database.

- If you need to move a database from one system to another, you should be aware that the identifiers used in the database's root file ACL on the source system are not likely to be valid identifiers on the destination system. Thus, if the database root file ACL from the source system is moved to the destination system without modification, only those users with the same identifiers on both systems have the same Oracle RMU access to the database on the destination system as they had to the database on the source system.

  For example, suppose that the mf_personnel database with the following root file ACL is moved from its current system to another node. If the database root file ACL is moved without modification to the destination node, the users USER, USER2, USER3, USER4, and USER5 will not have any Oracle RMU access to the database on the destination node unless they have the same identities on the destination node.

  ```
  $ RMU/SHOW PRIVILEGE MF_PERSONNEL.RDB
  Object type: file, Object name:SQL_USER:[USER]MF_PERSONNEL.RDB;1,
  on 31-MAR-1992 15:48:36.24
  ```

```
(IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
RMU$VERIFY)
(IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
(IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
(IDENTIFIER=[RDB,USER4],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
RMU$RESTORE)
(IDENTIFIER=[RDB,USER5],ACCESS=RMU$LOAD+RMU$SHOW)
(IDENTIFIER=[*,*],ACCESS=NONE)
```

- The following list describes some ways to move a database from one node to another and explains what happens to the original root file ACL in each scenario:
  - RMU Restore command

    First, use the RMU Backup command to back up the database on the source node and to create an .rbf file. Then, copy the .rbf file from the source node to the destination node. When you use the RMU Restore command to re-create the database from the source node on the destination node, the database on the destination node will have the same root file ACL as the database on the source node. If a user with the RMU$SECURITY privilege in the root file ACL on the source node has the same identifier on the destination node, that user can modify the root file ACL on the destination node to grant users the privileges they need for Oracle RMU access to the database. Otherwise, a user with one of the OpenVMS override privileges (SECURITY or BYPASS) needs to modify the root file ACL.
  - RMU Restore command with the Noacl qualifier

    First, use the RMU Backup command to back up the database on the source node and to create an .rbf file. Then, copy the .rbf file from the source node to the destination node. When you use the RMU Restore command with the Noacl qualifier to re-create the database from the source node on the destination node, the database on the destination node is created with an empty root file ACL. A user with one of the OpenVMS override privileges (SECURITY or BYPASS) needs to modify the root file ACL to grant users the privileges they need for Oracle RMU access to the database.
  - SQL IMPORT statement

    First, use the SQL EXPORT statement on the source node to create an .rbr file. Then, copy the .rbr file from the source node to the destination node. When you use the SQL IMPORT statement on the destination node, the imported database is created with the same root file ACL as existed on the database on the source node. If a user with the

RMU$SECURITY privilege in the root file ACL on the source node has the same identifier on the destination node, that user can modify the root file ACL on the destination node to grant users the privileges they need for Oracle RMU access to the database. Otherwise, a user with one of the OpenVMS override privileges (SECURITY or BYPASS) needs to modify the root file ACL to grant users the privileges they need for Oracle RMU access to the database.

- SQL IMPORT NO ACL statement

  First, use the SQL EXPORT statement on the source node to create an .rbr file. Then, copy the .rbr file from the source node to the destination node. When you use the SQL IMPORT NO ACL statement on the destination node, the imported database is created with a root file ACL that contains one ACE. The single ACE will grant the OpenVMS READ, WRITE, and CONTROL privileges plus all the Oracle RMU privileges to the user who performed the IMPORT operation. The user who performed the IMPORT operation can modify the root file ACL to grant users the privileges they need for Oracle RMU access to the database.

Examples

Example 1

The following example assumes that the user with a user identification code (UIC) of [SQL,USER] has created the mf_test_db database and is therefore the owner of the database. After creating the mf_test_db database, the owner displays the root file ACL for the database. Then the owner grants Oracle RMU privileges to database users. The Oracle RMU privileges granted to each type of user depend on the type of Oracle RMU access the user needs to the database.

```
$! Note that by default the owner (the user with a UIC of [SQL,USER])
$! is granted all the Oracle RMU privileges in the root file
$! ACL and no other users are granted any Oracle RMU privileges.

$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:51:55.79
```

```
    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
$!
$! The owner uses the RMU Set Privilege command and the After
$! qualifier to grant the RMU$ANALYZE, RMU$OPEN, and
$! RMU$VERIFY privileges to a user with a UIC of [SQL,USER2].
$! This user will serve as the database administrator for the
$! mf_test_db database.

$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE -
_$ +RMU$OPEN+RMU$VERIFY) -
_$ /AFTER=(IDENTIFIER=[SQL,USER])/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified

$!
$! Next, the owner grants the RMU$SECURITY privilege to a user with a
$! UIC of [SQL,USER3].  This gives the user USER3 the ability
$! to grant other users the appropriate privileges they need for
$! accessing the database with Oracle RMU commands.  Because both
$! the database creator and user USER3 have the RMU$SECURITY
$! privilege, both of them can modify the root file ACL for the
$! database.

$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY) -
_$ /AFTER=(IDENTIFIER=[SQL,USER2])/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$! The user with a UIC of [RDB,USER4], who will serve as the database
$! operator, is granted the RMU$BACKUP, RMU$CONVERT, RMU$DUMP, and
$! RMU$RESTORE privileges:
$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[RDB,USER4],ACCESS=RMU$BACKUP -
_$ +RMU$CONVERT+RMU$DUMP+RMU$RESTORE) -
_$ /AFTER=(IDENTIFIER=[SQL,USER3])/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$! The RMU$LOAD and RMU$SHOW privileges are granted to the user
$! with a UIC of [RDB,USER5]. This user will be writing programs
$! that load data into the database.

$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[RDB,USER5],ACCESS=RMU$LOAD -
_$ +RMU$SHOW) /AFTER=(IDENTIFIER=[RDB,USER4]) MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$! No privileges are granted to all other users.

$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[*,*],ACCESS=NONE) -
_$ /AFTER=(IDENTIFIER=[RDB,USER5])/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$! The RMU/SHOW PRIVILEGE command displays the root file ACL for the
$! mf_test_db database.

$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:52:17.03
```

1.62 RMU Set Privilege Command

```
    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[RDB,USER4],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[RDB,USER5],ACCESS=RMU$LOAD+RMU$SHOW)
    (IDENTIFIER=[*,*],ACCESS=NONE)
```

Example 2

The following command adds an ACE for the user with a UIC of [RDB,USER1] to the root file ACL for the personnel database. This ACE grants [RDB,USER1] the RMU$BACKUP privilege for the personnel database. The RMU$BACKUP privilege allows user [RDB,USER1] to access the RMU Backup, RMU Backup After_Journal, and RMU Checkpoint commands for the personnel database.

```
$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[RDB,USER1],ACCESS=RMU$BACKUP) -
_$ PERSONNEL.RDB
```

Example 3

The Replace qualifier in the following example causes the ACE in the root file ACL for the user with a UIC of [RDB,USER4] to be replaced by the ACE specified for the user with a UIC of [SQL,USER6]:

```
$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[RDB,USER4]) -
_$ /REPLACE=(IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT -
_$ +RMU$DUMP+RMU$RESTORE)/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:52:23.92

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[RDB,USER5],ACCESS=RMU$LOAD+RMU$SHOW)
    (IDENTIFIER=[*,*],ACCESS=NONE)
```

Example 4

The Delete qualifier in the following example causes the ACE for the user with a UIC of [RDB,USER5] to be deleted from the root file ACL for the mf_test_db database:

```
$ RMU/SET PRIVILEGE/ACL=(IDENTIFIER=[RDB,USER5]) -
_$ /DELETE/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
$!
$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:52:29.07

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[*,*],ACCESS=NONE)
```

Example 5

In the following example, the Like qualifier copies the root file ACL from the mf_test_db database to the test_db database. As part of this operation, the original root file ACL for the test_db database is deleted.

```
$ RMU/SHOW PRIVILEGE TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]TEST_DB.RDB;1,  on
30-MAR-1996 15:52:31.48

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
$ !
$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:52:33.86

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[*,*],ACCESS=NONE)
$!
$ RMU/SET PRIVILEGE/LIKE=MF_TEST_DB.RDB/LOG TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]TEST_DB.RDB;1 modified
```

1.62 RMU Set Privilege Command

```
$!
$ RMU/SHOW PRIVILEGE TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]TEST_DB.RDB;1,  on
30-MAR-1996 15:52:41.36

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[*,*],ACCESS=NONE)
```

Example 6

The New qualifier in the following example deletes all the existing ACEs and the Acl qualifier specifies a new ACE for the root file ACL for the mf_test_db database. Note that after the RMU Set Privilege command in this example is issued, only the user with a UIC of [SQL,USER2] or a user with an OpenVMS override privilege would be able to display the root file ACL for the mf_test_db database.

```
$ RMU/SHOW PRIVILEGE MF_TEST_DB.RDB
Object type: file,  Object name: SQL_USER:[USER]MF_TEST_DB.RDB;1,
on 30-MAR-1996 15:52:44.50

    (IDENTIFIER=[SQL,USER],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
    RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
    RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
    RMU$VERIFY)
    (IDENTIFIER=[SQL,USER2],ACCESS=RMU$ANALYZE+RMU$OPEN+RMU$VERIFY)
    (IDENTIFIER=[SQL,USER3],ACCESS=RMU$SECURITY)
    (IDENTIFIER=[SQL,USER6],ACCESS=RMU$BACKUP+RMU$CONVERT+RMU$DUMP+
    RMU$RESTORE)
    (IDENTIFIER=[*,*],ACCESS=NONE)
$!
$ RMU/SET PRIVILEGE/NEW -
_$ /ACL=(IDENTIFIER=[SQL,USER2],ACCESS=READ+WRITE+CONTROL+ -
_$ RMU$ALTER+RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+ -
_$ RMU$DUMP+RMU$LOAD+RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SHOW+ -
_$ RMU$UNLOAD+RMU$VERIFY)/LOG MF_TEST_DB.RDB
%RMU-I-MODIFIED, SQL_USER:[USER]MF_TEST_DB.RDB;1 modified
```

1.63 RMU Set Row_Cache Command

Allows you to enable or disable the database Row Cache feature and to modify certain parameters on a per cache basis.

Format

RMU/Set Row_Cache root-file-spec

Command Qualifiers	Defaults
/Alter=(Name=cache-name,option(,...))	See Description
/Backing_Store_Location=devdir	See Description
/NoBacking_Store_Location	See Description
/Disable	None
/Enable	None
/[No]Log	Current DCL verify value
/Sweep_Interval=n	See Description
/[No]Sweep_Interval	See Description

Description

You can use the RMU Set Row_Cache command to allow the database Row Cache feature to be enabled or disabled without requiring that the database be opened.

You can also use the Alter parameter to make modifications to one cache at a time.

Command Parameters

root-file-spec
Specifies the database root file for which you want to modify the Row Cache feature.

Command Qualifiers

Alter=(Name=cachename,option(,...))
Specifies the action to take on the named cache. You must specify the cache name and at least one other option. The /Alter qualifier may be specified multiple times on the command line. Each /Alter qualifier specified operates on one unique cache if no wildcard character (% or *) is specified. Otherwise, each /Alter operates on all matching cache names.

- Name=cachename

Name of the cache to be modified. The cache must already be defined in the database. You must specify the cache name if you use the Alter qualifier. This parameter accepts the wildcard characters asterisk (*) and percent sign (%).

- Backing_Store_Location=devdir

  Specifies the name of the cache-specific default directory to which row cache backing file information is written for the specified cache. The database system generates a file name (row-cache-name.rdc) automatically for each row cache backing file it creates when the RCS process starts. Specify a device name and directory name; do not include a file specification. By default, the location is the directory of the database root file unless a database-specific default directory or a cache-specific default directory has been set.

- NoBacking_Store_Location

  Specifies that there is no cache-specific default directory to which row cache backing file information is written for the specified cache.

- Drop

  Specifies that the indicated row cache is to be dropped (deleted) from the database.

- Shared_Memory=keyword

  Specifies the shared memory type and parameters for the cache. Valid keywords are:

  – Type=option

    Specify one of the following options:

    * Process

      Specifies traditional shared memory global section, which means that the database global section is located in process (P0) address space and may be paged from the process working set as needed.

    * Resident

      Specifies that the database global section is memory resident in process (P0) address space using shared page tables. This means that the global section is fully resident, or pinned, in memory, and uses less physical and virtual memory (for process page tables) than a traditional shared memory global section.

  – Rad_Hint=n
    NoRad_Hint

Indicates a request that memory should be allocated from the specified OpenVMS Resource Affinity Domain (RAD). This keyword specifies a hint to Oracle Rdb and OpenVMS about where memory should be physically allocated. It is possible that if the requested memory is not available, it will be allocated from other RADs in the system. For systems that do not support RADs, no Rad_Hint specification is valid.

The Rad_Hint keyword is valid only when the shared memory type is set to Resident. If you set the shared memory type to System or Process, you disable any previously defined RAD hint.

Use Norad_Hint to disable the RAD hint.

- Slot_Count=n

  Specifies the number of slots in the cache.

- Slot_Size=n

  Specifies the size (in bytes) of each slot in the cache.

- Snapshot_Slot_Count=n

  Specifies the number of snapshot slots in the cache. A value of zero disables the snapshot portion for the specified cache.

- Sweep_Interval=n

  Specifies the periodic cache sweep timer interval in seconds. Valid values are from 1 to 3600.

- NoSweep_Interval

  Disables the periodic cache sweep timer interval.

- Working_Set_Count=n

  Specifies the number of working set entries for the cache. Valid values are from 1 to 100.

Backing_Store_Location=devdir

Specifies the name of the database-specific default directory to which row cache backing file information is written. The database system generates a file name (row-cache-name.rdc) automatically for each row cache backing file it creates when the RCS process starts up. Specify a device name and directory name; do not include a file specification. The file name is the row-cache-name specified when creating the row cache. By default, the location is the directory of the database root file unless a database-specific default directory or a cache-specific default directory has been set.

Disable

Disables row caching. Do not use with the Enable qualifier.

1.63 RMU Set Row_Cache Command

Enable
Enables row caching. Do not use with the Disable qualifier.

Log
Nolog

Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch.

NoBacking_Store_Location
Specifies that there is no database-specific default directory to which row cache backing file information is written.

Usage Notes

- This command requires exclusive database access (the database cannot be open or accessed by other users).
- The Alter qualifier can be specified multiple times on the command line. Each use of the qualifier operates on a unique cache.
- Only one value can be supplied with the Rad_Hint keyword. The indicated RAD must contain memory.
- When shared memory is set to System (with Galaxy enabled) or to Resident, then the process that opens the database must be granted the VMS$MEM_RESIDENT_USER identifier.
- For applications that can be partitioned into one or more RADs, the Rad_Hint qualifier allows additional control over exactly where memory for caches and global sections is allocated. This control can permit increased performance if all application processes run in the same RAD, and the database and row cache global sections also reside in that same RAD.
- When Resident shared memory is specified, the global demand-zero pages are always resident in memory and are not backed up by any file on any disk. The pages are not placed into the process's working set list when the process maps to the global section and the virtual memory is referenced by the process. The pages are also not charged against the process's working set quota or against any page-file quota.
- To save physical memory, Oracle Rdb generally attempts to create and use shared page tables when creating large resident global sections.

- The total number of rows for any individual cache (the combination of live rows and snapshot rows) is limited to 2,147,483,647.
- Previously, the Record Cache Server (RCS) process would perform modified row cache "sweep" operations only when a cache was full (also known as "clogged") with modified rows. Now a database may be configured to perform timed cache sweeps. This feature is intended to help perform "lazy" updates of modified rows to the database from caches without performing a full cache checkpoint operation.

  The timer for the periodic cache sweeps is specified with the "SWEEP INTERVAL is numeric-literal seconds" clause of the ALTER DATABASE ... ROW CACHE IS ENABLED statement, as in the following example:

```
ALTER DATABASE FILENAME MF_PERSONNEL
  ROW CACHE IS ENABLED (SWEEP INTERVAL IS 300 SECONDS);
```

  The number of slots per cache to sweep is specified with the ALTER CACHE statement. Legal values for "SWEEP INTERVAL" are from 0 seconds (to disable periodic timed sweeps) to 3600 seconds (1 hour).

  The RMU /SET ROW_CACHE command accepts a /[NO]SWEEP_ INTERVAL=n qualifier as an alternate method to specify the periodic cache sweep timer. /NOSWEEP_INTERVAL disables periodic timed sweeps and /SWEEP_INTERVAL=n can be used to set the timer for the periodic cache sweeps. Legal values for /SWEEP_INTERVAL=n are from 1 second to 3600 seconds (1 hour).

  The Record Cache Server (RCS) process log file contains information about periodic row cache "sweep" operations and can be a useful analysis tool.

Examples

Example 1

The following example sets the slot count on cache "mycache".

```
$ RMU/SET ROW_CACHE/ALTER=(NAME=mycache, SLOT_COUNT=8888)
```

Example 2

This command disables all caches.

```
$ RMU/SET ROW_CACHE/DISABLE
```

Example 3

The following sample specifies that cache "cache2" should use RAD 2.

```
$ RMU/SET ROW_CACHE/ALTER=(NAME=cache2, SHARED_MEM=(TYPE=RESIDENT, -
_$ RAD_HINT=2)
```

Example 4

This example drops cache “seacache”.

```
$ RMU/SET ROW_CACHE/ALTER=(NAME=seacache, DROP)
```

Example 5

This example shows multiple uses of the Alter qualifier.

```
$ RMU /SET ROW_CACHE MF_PERSONNEL/ALTER=(NAME = RDB$SYS_CACHE, -
_$ SLOT_COUNT = 800, WINDOW_COUNT = 25) -
_$ /ALTER= (NAME = RESUMES,SLOT_SIZE=500,WORKING_SET_COUNT = 15)
```

Example 6

The following example modifies the database MYDB to set the snapshot slot count for the cache EMPL_IDX to 25000 slots and disables snapshots in cache for the SALES cache.

```
$ RMU /SET ROW_CACHE DGA0:[DB]MYDB.RDB -
_$ /ALTER=(NAME=EMPL_IDX, SNAPSHOT_SLOT_COUNT=25000) -
_$ /ALTER=(NAME=SALES, SNAPSHOT_SLOT_COUNT=0)
```

Example 7

The following example alters two caches:

```
$ RMU /SET ROW_CACHE MF_PERSONNEL -
        /ALTER= (   NAME = RDB$SYS_CACHE,
                    SLOT_COUNT = 800) -
        /ALTER= (   NAME = RESUMES, -
                    SLOT_SIZE=500, -
                    WORKING_SET_COUNT = 15)
```

Example 8

The following command alters caches named FOOD and FOOT (and any other cache with a 4 character name with the first three characters of “FOO” defined in the database):

```
$ RMU /SET ROW_CACHE MF_PERSONNEL -
        /ALTER= (   NAME = FOO%,
                    BACKING_STORE_LOCATION=DISK$RDC:[RDC])
```

1.64 RMU Set Server Command

Allows you to identify output files for several database server processes.

Format

RMU/Set Server server-type root-file-spec

Command Qualifiers	Defaults
/Log	None
/[No]Output[=file-name]	/NoOutput

Description

You can use the Set Server/Output command to identify output log file names and locations for various database server processes. The following table shows valid values for the server-type parameter and the corresponding logical name.

1.64 RMU Set Server Command

Table 1–17 Server Types and Logical Names

Server	Server-type	Logical Name
AIJ Backup Server	ABS	RDM$BIND_ABS_LOG_FILE
AIJ Log Server	ALS	RDM$BIND_ALS_OUTPUT_FILE
AIJ Log Roll-Forward Server	LRS	RDM$BIND_LRS_OUTPUT_FILE
AIJ Log Catch-Up Server	LCS	RDM$BIND_LCS_OUTPUT_FILE
Database Recovery Server	DBR	RDM$BIND_DBR_LOG_FILE
Row Cache Server	RCS	RDM$BIND_RCS_LOG_FILE

If the output file specification is empty (Output=""), the log file information for that server will be deleted from the database. RMU /SET SERVER /NOOUTPUT can also be specified as a way to disable the output file entry. Note that /NOOUTPUT is the default and if /OUTPUT is not specified, the output file server logging entry will be disabled.

If an existing logical name specifies an output file name for the specified server process, it takes precedence over the file name designated in the Set Server/Output command.

Command Parameters

server-type
Identifies the server process for which you want to log output. Refer to Table 1–17 for a list of valid server types and their corresponding logical names.

root-file-spec
Specifies the database root file for which you want to specify the server process output file.

Command Qualifiers

Log
Displays a log message at the completion of the RMU Set command.

Output[=file-spec]
NoOutput
Identifies the output log file for several database server processes.

If the output file specification is empty (Output=""), the log file information for that server will be deleted from the database. RMU /SET SERVER /NOOUTPUT can also be specified as a way to disable the output file entry. Note that /NOOUTPUT is the default and if /OUTPUT is not specified, the output file server logging entry will be disabled.

If an existing logical name specifies an output file name for the specified server process, it takes precedence over the file name designated in the Set Server/Output command.

Usage Notes

- This output name supports substitution of the process id (PID) within the filename of the opened log file. When this log file name is used, the first occurrence of the string _PID is replaced by the current OpenVMS process id. This string can appear in the directory, filename and file type portions of the file specification. The file specification can be in upper or lower case.

  This identifies each version of the log file by the executing user.

Examples

Example 1

This example specifies the output file for the row cache server and displays a log message when the procedure finishes.

```
$ RMU /SET SERVER RCS /OUTPUT=RCS_PID.LOG /LOG DUA0:[DB]MYDB.RDB
```

Example 2

This example specifies the output file for the AIJ log server.

```
$ RMU /SET SERVER ALS /OUTPUT=ALS$LOGS:ALS_DB1.LOG DUA0:[DB1]MFP.RDB
```

Example 3

This example deletes the log file information in the database for the AIJ log roll-forward server.

```
$ RMU /SET SERVER LRS /OUTPUT="" DUA0:[ZDB]ZDB.RDB
```

Example 4

This example does the same thing as Example 3 but uses the /NoOutput qualifier to accomplish the task.

```
$ RMU /SET SERVER LRS /NOOUTPUT DUA0:[ZDB]ZDB.RDB
```

1.64 RMU Set Server Command

Example 5

This example specifies the output file for the database recovery server.

```
$ RMU /SET SERVER DBR /OUTPUT=DBR$LOGS:DBR.LOG DUA0:[ADB]ADB.RDB
```

1.65 RMU Set Shared_Memory Command

Allows you to alter the database shared memory configuration without requiring that the database be open.

Format

RMU/Set Shared_Memory root-file-spec

Command Qualifiers	Defaults
/[No]Log	Current DCL verify value
/[No]Rad_Hint=n	None
/[No]Section_Name=secname	No Section_Name
/Type={Process|Resident|System}	None

Description

You can use the RMU Set Shared_Memory command to alter the database shared memory configuration without requiring that the database be open.

Command Parameters

root-file-spec
Specifies the database root file for which you want to modify the shared memory configuration.

Command Qualifiers

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the Log qualifier to request log output and the Nolog qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify switch.

Rad_Hint=n
Norad_Hint
Indicates a request that memory should be allocated from the specified OpenVMS Alpha Resource Affinity Domain (RAD). This qualifier specifies a hint to Oracle Rdb and OpenVMS about where memory should be physically allocated. It is possible that if the requested memory is not available, it will

be allocated from other RADs in the system. For systems that do not support RADs, a Rad_Hint value of zero is valid.

The Rad_Hint qualifier is only valid when the shared memory type is set to Resident. If you set the shared memory type to System or Process, you disable any previously defined RAD hint.

Use the Norad_Hint qualifier to disable the RAD hint.

Note

OpenVMS support for RADs is available only on the AlphaServer GS series systems. For more information about using RADs, refer to the *OpenVMS Alpha Partitioning and Galaxy Guide*.

Section_Name=secname
NoSection_Name

Allows the DBA to reserve physical memory in a global section and set a global section name for it that will persist regardless of where the root file resides. This name is stored in the database root file until it is overwritten by another name or negated.

"Secname" is a string that will be used as the common name portion for all memory resident global sections associated with this database. At runtime, Rdb will add a prefix to the common portion in order to distinguish the different global section types. Currently, there are three Rdb global section types that can be made memory resident:

- The database global section (TROOT/NODGBL)
- Row Cache global sections
- Row Cache RUJ Buffers

The prefix is composed using: "RDM" + (RDB major-minor version) + (a single character identifying the global section type). For example, in Rdb V7.3, the prefixes would be:

- RDM73N for the database global section (TROOT/NODGBL)
- RDM73R for the Row Cache global sections
- RDM73J for the Row Cache RUJ Buffers

After using the above syntax to set the section name, you can see the actual complete name for each affected global section using the RMU/DUMP/HEADER/OPT=DEBUG command (see Examples section).

Type=option
If you use the Type qualifier, you must specify one of the following options:

- Process

  Specifies traditional shared memory global section, which means that the database global section is located in process (P0) address space and may be paged from the process working set as needed.

- Resident

  Specifies that the database global section is memory resident in process (P0) address space using OpenVMS Alpha shared page tables. This means that the global section is fully resident, or pinned, in memory, and uses less physical and virtual memory (for process page tables) than a traditional shared memory global section.

- System

  Specifies that the database global section is located in OpenVMS Alpha system space, which means that the section is fully resident, or pinned, in memory, does not use process (P0) address space, and does not affect the quotas of the working set of a process.

Usage Notes

- This command requires exclusive database access (the database cannot be open or accessed by other users).
- Only one value can be supplied to the Rad_Hint qualifier. The indicated RAD must contain memory.
- When shared memory is set to System (with Galaxy enabled) or to Resident, then the process that opens the database must be granted the VMS$MEM_RESIDENT_USER identifier.
- For applications that can be partitioned into one or more RADs, the Rad_Hint qualifier allows additional control over exactly where memory for caches and global sections is allocated. This control can permit increased performance if all application processes run in the same RAD, and the database and row cache global sections also reside in that same RAD.
- When Resident shared memory is specified, the global demand-zero pages are always resident in memory and are not backed up by any file on any disk. The pages are not placed into the process's working set list when the process maps to the global section and the virtual memory is referenced by

the process. The pages are also not charged against the process's working set quota or against any page-file quota.

- To save physical memory, Oracle Rdb generally attempts to create and use shared page tables when creating large resident global sections.

Examples

Example 1

The following example sets the memory type to Resident and requests that it be put in RAD 4.

```
$ RMU/SET SHARED_MEMORY/TYPE=RESIDENT/RAD_HINT=4
```

Example 2

This example specifies that system space buffers are to be used.

```
$ RMU/SET SHARED_MEMORY/TYPE=SYSTEM
```

Example 3

The following example specifies that process address space shared memory is to be used.

```
$ RMU/SET SHARED_MEMORY/TYPE=PROCESS/LOG
```

Example 4

The following example shows setting a global section name and then using RMU/DUMP to show the full name. It then shows removing a specific section name.

```
$ RMU/SET SHARED_MEMORY/SECTION_NAME=$$MFP MF_PERSONNEL
$ RMU/DUMP/HEADER/OPT=DEBUG/OUT=MFP.DMP MF_PERSONNEL
$ SEARCH MFP.DMP "GLOBAL SECTION NAME"
      - Global Section Name is "RDM73N$$MFP00000000"
      - Global Section Name is "RDM73J$$MFP00000000"
      - Global Section Name is "RDM73R$$MFP00000001"
      - Global Section Name is "RDM73R$$MFP00000002"
      - Global Section Name is "RDM73R$$MFP00000003"
      - Global Section Name is "RDM73R$$MFP00000004"
      - Global Section Name is "RDM73R$$MFP00000005"
      - Global Section Name is "RDM73R$$MFP00000006"
      - Global Section Name is "RDM73R$$MFP00000007"
      - Global Section Name is "RDM73R$$MFP00000008"
      - Global Section Name is "RDM73R$$MFP00000009"
      - Global Section Name is "RDM73R$$MFP0000000A"
      - Global Section Name is "RDM73R$$MFP0000000B"
      - Global Section Name is "RDM73R$$MFP0000000C"
      - Global Section Name is "RDM73R$$MFP0000000D"
      - Global Section Name is "RDM73R$$MFP0000000E"
      - Global Section Name is "RDM73R$$MFP0000000F"
$
$ RMU/SET SHARED_MEMORY/NOSECTION_NAME MF_PERSONNEL
$ RMU/DUMP/HEADER/OPT=DEBUG/OUT=MFP.DMP MF_PERSONNEL
$ SEARCH MFP.DMP "GLOBAL SECTION NAME"
      - Global Section Name is "RDM73N$1$DGA21235C714FC000000000000"
      - Global Section Name is "RDM73J$1$DGA21235C714FC000000000000"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000001"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000002"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000003"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000004"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000005"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000006"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000007"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000008"
      - Global Section Name is "RDM73R$1$DGA21235C714FC000000000009"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000A"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000B"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000C"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000D"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000E"
      - Global Section Name is "RDM73R$1$DGA21235C714FC00000000000F"
```

1.66 RMU Set Statistics Command

Allows the user to manage the saving and restoring of database statistics.

Format

RMU/Set Statistics root-file_spec

Command Qualifiers	Defaults
/[No]Checkpoint=[n]	See description
/Export=[Close|NoClose]	See description
/Import={Open|NoOpen}	See description
/[No]Log	NoLog

Description

The Oracle Rdb RMU Set Statistics command allows the user to manage the saving and restoring of database statistics.

RMU Set Statistics allows saving database statistics by writing them to a node-specific database file located in the same directory as the database root file, and initializing database statistics by reading them from the same node-specific database file. This file has the same name as the root file, with a default file extension of .rds.

The statistics file contains node-specific information, and it cannot be renamed or copied. The exported database statistics file (.rds) can be deleted if it is no longer needed. The RMU Backup command does not save the statistics files. They are considered to be temporary files and not part of the database.

The RMU Set Statistics command can be used with databases defined with a Manual open mode and databases defined with an Automatic open mode. The RMU Close command Statistics=Export qualifier can also be used to save statistics to the same node-specific database file and the RMU Open command Statistics=Import qualifier can be used to set database statistics by reading them from the same node-specific database file, but only for databases defined with a Manual open mode. For more information, see the documentation for the RMU Close and RMU Open commands.

You must have RMU$ALTER privilege in the root file access control list (ACL) for a database or the OpenVMS SYSPRV or BYPASS privilege to use the RMU Set Statistics command.

Database statistics must be enabled in the database root. The RMU Dump Header command will display the following message if statistics are enabled for the database.

```
- Statistics are enabled
```

Command Parameters

root-file-spec
Specifies the database root file for which you want to manage the saving and restoring of database statistics.

Command Qualifiers

Checkpoint
Checkpoint=n
NoCheckpoint
When statistics values are imported from the node-specific database statistics file by the database monitor at the time the database is opened, automatic periodic checkpoints are started by default to export the statistics values to the node-specific statistics file to keep the statistics file as current as possible in case a system crash occurs before the database is closed.

If NOCHECKPOINT is specified, these default periodic export checkpoints are disabled by modifying a parameter in the database root. If only CHECKPOINT (without a value) is specified, a default periodic export checkpoint interval of 30 minutes is used.

The monitor log and the RMU Show System command will indicate if these checkpoints are occurring.

All options of the checkpoint qualifier require that the database must currently be closed since the root parameters of the database are being modified.

If statistic export checkpoints are enabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics export checkpoints are enabled
```

If statistic export checkpoints are disabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics export checkpoints are disabled
```

By default, these periodic export checkpoints occur every 30 minutes. If CHECKPOINT = n is specified, the export checkpoints interval parameter can be modified in the database root for checkpoints to occur every n minutes, where n can be a minimum export checkpoint interval of 30 minutes, a maximum checkpoint interval of 1440 minutes (which is one checkpoint every 24 hours), or any number of minutes between these minimum and maximum values. The RMU Dump Header command will display the setting of the export checkpoint value in the root as follows:

```
- Statistics export interval is n minutes
```

If CHECKPOINT = n is specified, only the statistics export checkpoint interval for the database is modified. NOCHECKPOINT or CHECKPOINT without the checkpoint interval in minutes parameter, must be specified to disable or enable periodic export checkpoints.

If statistics are disabled for the database, the RMU Dump Header command will display the following:

```
Statistics are disabled
```

If statistics are enabled for the database but the import of database statistics on database open is disabled, the RMU Dump Header command will not show any statistics checkpoint information since statistics checkpoints are only executed if statistics are imported when the database is opened.

```
Statistics...
  - Statistics are enabled
  - Statistics import on database open is disabled
  - Statistics export on database close is disabled
```

If statistics are enabled for the database and the import of database statistics on database open is enabled, the RMU Dump Header command will show the statistics export checkpoint parameters for the database.

```
Statistics...
   - Statistics are enabled
   - Statistics import on database open is enabled
   - Statistics export on database close is enabled
   - Statistics export checkpoints are enabled
   - Statistics export interval is 30 minutes
```

Export
Export=Close
Export=NoClose

If just EXPORT is specified, the database monitor will immediately write the database statistics to the node-specific database statistics file. The monitor must be running and the database must currently be open. The default if

EXPORT is omitted is not to write the database statistics to the node-specific database statistics file. The monitor log will record the export of the database statistics. If the node-specific database statistics file does not exist, it will be created. The existing node-specific database statistics file contents will be replaced and the version number of this file will not be incremented.

If EXPORT = CLOSE is specified, a parameter will be set in the database root to cause the monitor to automatically save the database statistics to the node-specific database statistics file whenever the database is closed. No immediate statistics export is executed. The RMU Dump Header command will show the open and close modes for the database as either:

```
Database open mode is MANUAL
Database close mode is MANUAL
```

or:

```
Database open mode is AUTOMATIC
Database close mode is AUTOMATIC
```

If the open and close modes are MANUAL, the database must be opened by an RMU Open command and closed by an RMU Close command. For more information, see the documentation for the RMU Close and RMU Open commands. If the open and close modes are AUTOMATIC, the database is opened when the first user attaches to the database and closed when no users are attached to the database.

If EXPORT = NOCLOSE is specified, a parameter will be set in the database root to cause the monitor to not save the database statistics to the node-specific database statistics file whenever the database is closed.

If EXPORT = CLOSE or EXPORT = NOCLOSE is specified, the database must currently be closed since the root parameters of the database are being modified.

If automatic exporting of the database statistics to the node-specific database statistics file is enabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics export on database close is enabled
```

If automatic exporting of the database statistics to the node-specific database statistics file is disabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics export on database close is disabled
```

1.66 RMU Set Statistics Command

Import=Open
Import=NoOpen
If IMPORT = OPEN is specified, a parameter is set in the database root to cause the monitor to automatically initialize the database statistics by reading the node-specific database statistics file whenever the database is opened. Statistics can only be imported when the database is opened. No immediate import of database statistics is allowed once the database has been opened to prevent currently active database statistics values from being overwritten.

If IMPORT = NOOPEN is specified, a parameter will be set in the database root to cause the monitor to not initialize the database statistics by reading the node-specific database statistics file when the database is opened. The RMU Dump Header command will show the open and close modes for the database as either of the following:

```
Database open mode is MANUAL
Database close mode is MANUAL
```

or:

```
Database open mode is AUTOMATIC
Database close mode is AUTOMATIC
```

If the open and close modes are MANUAL, the database must be opened by an RMU Open command and closed by an RMU Close command. For more information, see the documentation for the RMU Close and RMU Open commands. If the open and close modes are AUTOMATIC, the database is opened when the first user attaches to the database and closed when no users are attached to the database.

If IMPORT = OPEN or IMPORT = NOOPEN is specified, the database must currently be closed since the root parameters of the database are being modified.

If automatic importing of the database statistics from the node-specific database statistics file is enabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics import on database open is enabled
```

If automatic importing of the database statistics from the node-specific database statistics file is disabled for the database, the RMU Dump Header command will display the following message:

```
- Statistics import on database open is disabled
```

Log
Nolog
Displays informational messages during the execution of the RMU Set Statistics command. The default for this qualifier is NOLOG.

Examples

Example 1 RMU Set Statistics Export

The following example shows the use of the RMU Set Statistics command to do an immediate export to the database node-specific statistics file to save the current database statistics.

An RMU Show System command shows that the database monitor is running and that the database is open, which are requirements for doing immediate exports and imports of statistics. No *.rds statistics file currently exists for the database so the RMU Set Statistics command Export qualifier creates the file using the node name and the database name and writes the current database statistics values to the statistics file.

```
$ rmu/show user
Oracle Rdb V7.3-300 on node TSTNOD 11-AUG-2016 16:16:33.77
    - monitor started 10-AUG-2016 19:14:09.13 (uptime 0 21:02:24)
    - monitor log filename is "DEVICE:[DIRECTORY]RDMMON73_TSTNOD.LOG;3487"

database DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1
    - opened 11-AUG-2016 16:13:09.96 (elapsed 0 00:03:23)
    - 2 active database users on this node

$ dir *.rds
%DIRECT-W-NOFILES, no files found
$ rmu/set statistics/export/log mf_personnel.
%RMU-I-MODIFIED, Database Statistics Import/Export file state modified
$ dir/date *.rds

Directory DEVICE:[DIRECTORY]

MF_PERSONNEL_TSTNOD.RDS;1
                     11-AUG-2016 16:17:05.54

Total of 1 file.
```

Example 2 RMU Set Statistics Checkpoint

The following example shows the use of the RMU Set Statistics command to modify the statistics export and import parameters in the database root used by the database monitor for the export of statistics to the database node-specific statistics file when the database is closed and the import of statistics from the database node-specific statistics file when the database is opened, as

well as for periodic checkpoints to export the statistics to keep them as up to date as possible in case a system crash occurs before the database is closed.

The first RMU Show System command shows that the database is closed and not being accessed by database users, which is a requirement for setting statistics import and export parameters in the database root.

The RMU Dump Header command shows that statistics collection is enabled for the database but the export of statistics when the database is closed and the import of statistics when the database is opened are both disabled. The statistics export checkpoint parameters are not shown because statistic checkpoints are only executed if statistics are imported when the database is opened.

The RMU Set Statistics command is then used to enable statistic imports when the database is opened, statistic exports when the database is closed and to set the interval between periodic statistic exports to 60 minutes (the default is 30 minutes). Periodic export checkpoints are the default if statistics are imported when the database is opened.

The RMU Dump Header command is used to display the new statistics parameters that have been set in the database root. Later, the RMU Set Statistics command is used to disable statistic imports when the database is opened, statistic exports when the database is closed, and periodic checkpoints to export statistics.

The RMU Dump Header command shows that statistics collection is enabled for the database but the export of statistics when the database is closed and the import of statistics when the database is opened are both disabled. The statistics export checkpoint parameters are not shown because statistic checkpoints are only executed if statistics are imported when the database is opened.

```
$ rmu/show system
Oracle Rdb V7.3-300 on node TSTNOD 12-AUG-2016 09:54:08.45
    - monitor started 11-AUG-2016 19:14:09.20 (uptime 0 14:39:59)
    - monitor log filename is "DEVICE:[DIRECTORY]RDMMON73_MALIBU.LOG;3488"
    - no databases are accessed by this node

$ rmu/dump/header/out=mfp.hdr mf_personnel
$ search mfp.hdr statistics
    Statistics...
      - Statistics are enabled
      - Statistics import on database open is disabled
      - Statistics export on database close is disabled
```

```
$ rmu/set statistics/import=open/export=close/checkpoint=60/log mf_personnel
%RMU-I-LOGMODFLG,     enabled database open statistics import
%RMU-I-LOGMODFLG,     enabled database close statistics export
%RMU-I-LOGMODFLG,     modified statistics export interval to 60 minutes
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
%RMU-I-MODIFIED, Database Statistics Import/Export file state modified

$ search mfp.hdr statistics
    Statistics...
      - Statistics are enabled
      - Statistics import on database open is enabled
      - Statistics export on database close is enabled
      - Statistics export checkpoints are enabled
      - Statistics export interval is 60 minutes

$ rmu/set statistics/import=noopen/export=noclose/nocheckpoint/log mf_personnel
%RMU-I-LOGMODFLG,     disabled database open statistics import
%RMU-I-LOGMODFLG,     disabled database close statistics export
%RMU-I-LOGMODFLG,     disabled database statistics export checkpoints
%RMU-W-DOFULLBCK, full database backup should be done to ensure future recovery
%RMU-I-MODIFIED, Database Statistics Import/Export file state modified

$ search mfp.hdr statistics
    Statistics...
      - Statistics are enabled
      - Statistics import on database open is disabled
      - Statistics export on database close is disabled
```

1.67 RMU Show Command

Displays current information about security audit characteristics, version numbers, active databases, active users, active recovery-unit files, after-image journal files, area inventory pages, corrupt areas and pages, optimizer statistics, or database statistics related to database activity on your node. Note that, with the exception of the RMU Show Locks and RMU Show Users commands, the RMU Show commands display information for your current node only in a clustered environment.

Oracle RMU provides the following Show commands:

After_Journal
AIP
Audit
Corrupt_Pages
Locks
Logical_Names
Optimizer_Statistics
Privilege
Statistics
System
Users
Version

Each show command is described in a separate section.

1.67.1 RMU Show After_Journal Command

Displays the after-image journal configuration in the form required for the Aij_Options qualifier. You can use the Aij_Options qualifier with the RMU Copy_Database, RMU Move_Area, RMU Restore, RMU Restore Only_Root, and RMU Set After_Journal commands.

Optionally, this command initializes the RDMAIJ_BACKUP_SEQNO, RDMAIJ_COUNT, RDM$AIJ_CURRENT_SEQNO, RDM$AIJ_ENDOFFILE, RDM$AIJ_FULLNESS, RDM$AIJ_LAST_SEQNO, RDM$AIJ_NEXT_SEQNO, and RDM$AIJ_SEQNO global process symbols.

Format

RMU/Show After_Journal root-file-spec

Command Qualifiers	Defaults
/[No]Backup_Context	/Nobackup_Context
/[No]Checkpoint	/Nocheckpoint
/Output[=file-name]	SYS$OUTPUT

Description

The output of the RMU Show After_Journal command appears in the form shown in Figure 1–1. This is the form required by the Aij_Options qualifier for the RMU Copy_Database, Move_Area, and Restore commands. When you issue the RMU Show After_Journal command, you may see fewer items than shown in Figure 1–1; some options do not appear unless you specified them when you created your after image journal file configuration (for example, with the RMU Set After_Journal command).

When you use the output from the Show After_Journal command as a template for the Aij_Options qualifier of the RMU Copy_Database, Move_Area, and Restore commands, note the following regarding the syntax:

- As shown in Figure 1–1, you can use the DCL continuation character (-) at the end of each line in the Journal and Add clauses. Although continuation characters are not required if you can fit each clause (Journal or Add clause) on a single line, using them might improve readability.
- The Journal Is clause must precede the Add clause.

Figure 1–1 Output from the RMU Show After_Journal Command

```
Journal [Is] {Enabled | Disabled} -
[Reserve n] -
[Allocation [Is] n] -
[Extent [Is] n] -
[Overwrite [Is] {Enabled|Disabled}] -
[Shutdown_Timeout [Is] n] -
[Notify [Is] {Enabled|Disabled}] -
[Backups [Are] {Manual|Automatic} -
[[No]Quiet_Point] [File filename]] -
[Cache [Is] {Enabled File filename|Disabled}]
[Logminer [Is] {Enabled|Disabled} [Continuous]] -
Add [Journal] journal-name -
! File file-specification
File filename -
[Allocation [Is] n] -
[Backup_File filename] -
[Edit_String [Is] (edit-string-options)]
```

- Because the Journal clause and the Add clause are two separate clauses, a continuation character should not be used between the last option in the Journal clause and the Add clause (or clauses).
- The journal options file can contain one Journal clause only, but it can contain several Add clauses. However, the number of Add clauses cannot exceed the number of reservations made for .aij files. In addition, if you are enabling journaling, you must add at least one journal.
- You can specify only one of each option (for example, one Extent clause, one Cache clause, and so on) for the Journal Is clause.

The clauses and options have the following meaning:

- Journal Is Enabled

  Enables after-image journaling. At least one Add clause must follow. If this option is omitted, the current journaling state is maintained.

- Journal Is Disabled

  Disables after-image journaling. You can specify other options or Add clauses but they do not take effect until journaling is enabled. The Add clause is optional. If this option is omitted, the current journaling state is maintained.

- Reserve n

Allocates space for an .aij file name for a maximum of *n* .aij files. By default, no reservations are made. Note that you cannot reserve space in a single-file database for .aij files by using this option with the RMU Move_Area command with the Aij_Options qualifier. After-image journal file reservations for a single-file database can be made only when you use the RMU Convert, RMU Restore, or RMU Copy_Database commands.

- Allocation Is n

  Specifies the size (in blocks) of each .aij file. If this option is omitted, the default allocation size is 512 blocks. The maximum allocation size you can specify is eight million blocks.

  See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the allocation size.

- Extent Is n

  Specifies the maximum size to extend an .aij journal if it is, or becomes, an extensible .aij journal (in blocks). (If the number of available after-image journal files falls to one, extensible journaling is employed.)

  If there is insufficient free space on the .aij journal device, the journal is extended using a smaller extension value than specified. However, the minimum, and default, extension size is 512 blocks.

  See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the extent size.

- Overwrite Is Enabled

  Enables overwriting of journals before they have been backed up. If this option is omitted, overwriting is disabled.

  This option is ignored if only one .aij file is available. When you specify the Overwrite Is Enabled option it is activated only when two or more .aij files are, or become, available.

- Overwrite Is Disabled

  Disables overwriting of journals before they have been backed up. If this option is omitted, overwriting is disabled.

- Shutdown_Timeout Is n

  Sets the delay from the time a journal failure is detected until the time the database aborts all access and shuts itself down. The value *n* is in minutes.

  If this option is omitted, the shutdown timeout is 60 minutes. The maximum value you can specify is 4320 minutes.

- Notify Is Enabled

Enables operator notification when the journal state changes. If this option is omitted, operator notification is disabled.

- Notify Is Disabled

  Disables operator notification when the journal state changes. If this option is omitted, operator notification is disabled.

- Backups Are Manual

  Automatic backup operations are not enabled. This is the default behavior.

- Backups Are Automatic [File filename]

  Automatic backup operations are triggered by the filling of a journal. The backup file will have the specified file name unless a different file name or an edit string is specified in the Add clause. If this option is omitted, backup operations are manual.

- Edit String Is (edit-string-options)

  Specifies a default edit string to apply to the backup file when an .aij is backed up automatically. See the description of the Edit_Filename keyword in Section 1.53 for a description of the available options. An Edit_String that appears with the definition of an added journal takes precedence over this edit string.

- Quiet_Point

  Specifies that the after-image journal backup operation is to acquire the quiet-point lock prior to performing an .aij backup operation for the specified database.

- Noquiet_Point

  Specifies that the after-image journal backup operation will not acquire the quiet-point lock prior to performing an .aij backup operation for the specified database.

- Cache Is Enabled File filename

  Specifies that a journal cache file should be used. The cache file must reside on a nonvolatile solid-state disk. If it does not, caching is ineffectual. See Section 1.53 for information on what happens if the cache file becomes inaccessible.

  By default, caching is disabled.

- Cache Is Disabled

  Specifies that a journal cache file should not be used. This is the default behavior.

- LogMiner Is Disabled

  Specifies that both LogMiner and the Continuous LogMiner feature are disabled. The Continuous LogMiner feature cannot be enabled unless LogMiner is also enabled.

- LogMiner Is Enabled

  Specifies that LogMiner is enabled but the Continuous LogMiner feature is disabled.

- LogMiner Is Enabled Continuous

  Specifies that both LogMiner and the Continuous LogMiner feature are enabled. The Continuous LogMiner feature cannot be enabled unless LogMiner is also enabled.

- The Add clause or clauses specify the name and location of the journal file and the backup file generated by automatic backup operations as follows:

  - Add [Journal] journal-name

    Specifies the name for the after-image journal file described in the Journal clause. The journal-name is the name of the journal object. A journal object is the journal file specification plus all the attributes (allocation, extent, and so on) given to it in the journal clause.

  - ! File file-specification

    Provides the full file specification and version number of the .aij file named in the Add clause. This line of output is provided because the next line (File filename) provides the string that the user entered when he or she created the .aij file. For example, if the user entered a file name only, and this line of output was not provided, you would have to issue the RMU Dump command to determine in which directory the file resides.

  - File filename

    Specifies the file name for the .aij file being added. This option is required.

  - Allocation Is n

    Specifies the size of the .aij file (in blocks). If this option is omitted, the default allocation size is 512 blocks.

    See the *Oracle Rdb Guide to Database Maintenance* for guidance on setting the allocation size.

  - Backup_File filename

Specifies the backup file name for automatic backup operations. Note that it is not valid to specify a Backup_File clause in the Add clause if you have specified Backups Are Manual in the Journal clause; Oracle RMU returns an error if you attempt to do so.

– Edit String Is (edit-string-options)

  Specifies an edit string to apply to the backup file when the .aij is backed up automatically. See the description of the Edit_Filename keyword in Section 1.53 for a description of the available keywords.

Command Parameters

root-file-spec
The root file specification of the database for which you want the after-image journal configuration to be displayed.

Command Qualifiers

Backup_Context
Nobackup_Context
The Backup_Context qualifier specifies that the following symbols be initialized, unless you have issued a DCL SET SYMBOL/SCOPE=(NOLOCAL, NOGLOBAL) command:

- RDM$AIJ_SEQNO

  Contains the sequence number of the last .aij backup file written to tape. This symbol has a value identical to RDM$AIJ_BACKUP_SEQNO. RDM$AIJ_SEQNO was created prior to Oracle Rdb Version 6.0 and is maintained for compatibility with previous versions of Oracle Rdb.

- RDM$AIJ_CURRENT_SEQNO

  Contains the sequence number of the currently active .aij file. A value of –1 indicates that after-image journaling is disabled.

- RDM$AIJ_NEXT_SEQNO

  Contains the sequence number of the next .aij file that needs to be backed up. This symbol always contains a positive integer value (which may be 0).

- RDM$AIJ_LAST_SEQNO

  Contains the sequence number of the last .aij file available for a backup operation, which is different from the current sequence number if fixed-size journaling is being used. A value of –1 indicates that no journal has ever been backed up.

If the value of the RDM$AIJ_NEXT_SEQNO symbol is greater than the value of the RDM$AIJ_LAST_SEQNO symbol, then no more .aij files are currently available for the backup operation.

- RDM$AIJ_BACKUP_SEQNO

  Contains the sequence number of the last .aij file backed up (completed) by the backup operation. This symbol is set at the completion of an .aij backup operation. A value of –1 indicates that this process has not yet backed up an .aij file.

- RDM$AIJ_COUNT

  Contains the number of available .aij files.

- RDM$AIJ_ENDOFFILE

  Contains the end of file block number for the current AIJ journal.

- RDM$AIJ_FULLNESS

  Contains the percent fullness of the current AIJ journal.

- RDM$HOT_STANDBY_STATE - Contains the current replication state. Possible state strings and the description of each state are listed below:

  - "Inactive" - Inactive
  - "DB_Bind" - Binding to database
  - "Net_Bind" - Binding to network
  - "Restart" - Replication restart activity
  - "Connecting" - Waiting for LCS to connect
  - "DB_Synch" - Database synchronization
  - "Activating" - LSS server activation
  - "SyncCmpltn" - LRS synchronization redo completion
  - "Active" - Database replication
  - "Completion" - Replication completion
  - "Shutdown" - Replication cleanup
  - "Net_Unbind" - Unbinding from network
  - "Recovery" - Unbinding from database
  - "Unknown" - Unknown state or unable to determine state

- RDM$HOT_STANDBY_SYNC_MODE - Contains the current replication synchronization mode when replication is active. Possible synchronization mode strings are listed below:
  - "Cold"
  - "Warm"
  - "Hot"
  - "Commit"
  - "Unknown"
- RDM$LOGMINER_STATE - Contains the current LogMiner state. Possible state strings and the description of each state are listed below:
  - "Disabled"

    Both LogMiner and the Continuous LogMiner feature are disabled.
  - "Enabled"

    LogMiner is enabled but the Continuous LogMiner feature is disabled.
  - "Enabled Continuous"

    Both LogMiner and the Continuous LogMiner feature are enabled. The Continuous LogMiner feature cannot be enabled unless LogMiner is also enabled.

The Nobackup_Context qualifier specifies that the preceding symbols will not be initialized.

The Nobackup_Context qualifier is the default.

Note that these are string symbols, not integer symbols, even though their equivalence values are numbers. Therefore performing arithmetic operations with them produces unexpected results.

If you need to perform arithmetic operations with these symbols, first convert the string symbol values to numeric symbol values using the OpenVMS F$INTEGER lexical function. For example:

```
$ SEQNO_RANGE = F$INTEGER(RDB$AIJ_LAST_SEQNO) - F$INTEGER(RDB$AIJ_NEXT_SEQNO)
```

Checkpoint
Nocheckpoint

The RMU/SHOW AFTER_JOURNAL [NO]CHECKPOINT qualifier can be used to request that all active database processes on all nodes perform a checkpoint when the /BACKUP_CONTEXT qualifier is also specified to set After Image Journal database symbols based on the current After Image

Journal configuration defined in the database root file and the database fast commit to journal feature is enabled for the database. For more information on these After Image Journal database symbols that begin with "RDM$AIJ_", see the documentation for the /BACKUP_CONTEXT qualifier.

The checkpoint occurs immediately before the AIJ global symbols are defined or modified. The checkpoint will only be executed if the RMU/SHOW AFTER_ JOURNAL command /BACKUP_CONTEXT qualifier is also specified and the database fast commit to journal feature is currently enabled. This checkpoint ensures that the work done by committed transactions is flushed to the AIJ file just before /BACKUP_CONTEXT sets the global RDM$* AIJ database logicals so the AIJ sequence numbers and other data in the root used to set the logicals will be up to date with the work done by committed transactions.

The default if this qualifier is not specified is /NOCHECKPOINT.

Output[=file-name]
Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify only a file name.

Usage Notes

- To use the RMU Show After_Journal command for a database, you must have the RMU$BACKUP, RMU$RESTORE, or RMU$VERIFY privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or OpenVMS BYPASS privilege.

Examples

Example 1

The following example shows the output from the RMU Show After_Journal command when one journal is available, which means extensible journaling will be used. The commented line is generated by the RMU Show After_ Journal command to display the full file specification for the added .aij file. The next line shows the actual file specification entered by the user when he or she created the .aij file configuration. In this example, the user did not enter a full specification, therefore only the file name appears in the uncommented portion of the code.

```
$ RMU/SHOW AFTER_JOURNAL MF_PERSONNEL

JOURNAL IS ENABLED -
    RESERVE 1 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED -
    LOGMINER IS DISABLED
ADD JOURNAL AIJ_ONE -
! FILE USER2:[JOURNALONE]AIJ1.AIJ;1
    FILE AIJ1.AIJ -
    BACKUP DISK1:[BACKUP_AIJ]AIJ1BCK.AIJ; -
    EDIT_STRING IS (SEQUENCE)
    ALLOCATION IS 512
```

Example 2

The following example shows the output from the RMU Show After_Journal command when two journal files are enabled, which means fixed-size journaling will be used. In this example, the user entered a full file specification for the .aij file when the .aij file configuration was created. Thus, the commented line and the one appearing below it are identical with the exception of the file version:

```
$ RMU/SHOW AFTER_JOURNAL MF_PERSONNEL

JOURNAL IS ENABLED -
    RESERVE 2 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED -
    LOGMINER IS DISABLED
ADD JOURNAL AIJ_ONE.AIJ -
! FILE DISK2:[AIJ]AIJ1.AIJ;1
    FILE DISK2:[AIJ]AIJ1.AIJ -
    BACKUP DISK1:[BACKUP_AIJ]AIJ1BCK.AIJ; -
    EDIT_STRING IS (SEQUENCE)
    ALLOCATION IS 512
```

```
ADD JOURNAL AIJ_TWO.AIJ -
! FILE DISK3:[AIJTWO]AIJ2.AIJ;1
    FILE DISK3:[AIJTWO]AIJ2.AIJ -
    BACKUP DISK1:[BACKUP_AIJ]AIJ2BCK.AIJ; -
    EDIT_STRING IS (SEQUENCE)
     ALLOCATION IS 512
```

Example 3

The following example uses the RMU Show After_Journal command to show the settings of the symbolic names for the .aij sequence numbers before and after the RMU Backup command is executed:

```
$ RMU/SHOW AFTER_JOURNAL/BACKUP_CONTEXT MF_PERSONNEL

JOURNAL IS ENABLED -
    RESERVE 4 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED -
    LOGMINER IS DISABLED
ADD JOURNAL AIJ2 -
! FILE DISK2:[DB]AIJ_TWO;1
    FILE DISK2:[DB]AIJ_TWO -
    ALLOCATION IS 512
ADD JOURNAL AIJ3 -
! FILE DISK3:[DB]AIJ_THREE;1
    FILE DISK3:[DB]AIJ_THREE -
    ALLOCATION IS 512
$ SHOW SYMBOL RDM$AIJ*
  RDM$AIJ_COUNT == "2"
  RDM$AIJ_CURRENT_SEQNO == "0"
  RDM$AIJ_ENDOFFILE == "1"
  RDM$AIJ_FULLNESS == "0"
  RDM$AIJ_LAST_SEQNO = "-1"
  RDM$AIJ_NEXT_SEQNO = "0"
$ RMU/BACKUP/AFTER MF_PERSONNEL AIJ_TWO, AIJ_THREE

%RMU-I-LOGBCKAIJ, backing up after-image journal RDM$JOURNAL
%RMU-I-AIJBCKSEQ, backing up current after-image journal sequence
 number 0
$ RMU/SHOW AFTER_JOURNAL/BACKUP_CONTEXT MF_PERSONNEL
   .
   .
   .
```

```
$ SHOW SYMBOL RDM$AIJ*
  RDM$AIJ_BACKUP_SEQNO == "-1"
  RDM$AIJ_COUNT == "2"
  RDM$AIJ_CURRENT_SEQNO = "1"
  RDM$AIJ_ENDOFFILE == "1"
  RDM$AIJ_FULLNESS == "0"
  RDM$AIJ_LAST_SEQNO = "0"
  RDM$AIJ_NEXT_SEQNO = "1"
  RDM$AIJ_SEQNO == "-1"
```

Example 4

The following example shows a database defined with circular After Image Journal files and fast commit to the journal files enabled. Then an RMU/SHOW AFTER_JOURNAL command is executed with the /BACKUP_CONTEXT qualifier and the /CHECKPOINT qualifier specified. In this case, a global checkpoint will be executed before the global symbols that start with 'RDM$AIJ_' are defined or modified. The second RMU/SHOW AFTER_JOURNAL command is executed with the /BACKUP_CONTEXT qualifier and the /NOCHECKPOINT qualifier specified. Therefore, a global checkpoint will not be executed before the global symbols that start with 'RDM$AIJ_' are defined or modified. The /NOCHECKPOINT qualifier did not have to be specified since it is the default.

```
$ SQL
create database filename foo
    reserve 10 journals
    create storage area RDB$SYSTEM
        filename foo
        alloc 3000
        snap alloc 1;
create table tab (a int, b char(500)) ;
commit;
disc all ;
alter database filename foo
      journal is enabled (fast commit enabled)
      add journal foo_aij_0
          filename test$scratch:foo_aij_0.aij
      add journal foo_aij_1
          filename test$scratch:foo_aij_1.aij
      add journal foo_aij_2
          filename test$scratch:foo_aij_2.aij
      add journal foo_aij_3
          filename test$scratch:foo_aij_3.aij
      add journal foo_aij_4
          filename test$scratch:foo_aij_4.aij
      add journal foo_aij_5
          filename test$scratch:foo_aij_5.aij
      add journal foo_aij_6
          filename test$scratch:foo_aij_6.aij
```

```
      add journal foo_aij_7
          filename test$scratch:foo_aij_7.aij
   ;
%RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
exit
$ rmu/dump/header/out=foo.hdr foo
$ sear foo.hdr fast
    Fast Commit...
      - Fast commit is enabled
      - Fast incremental backup is enabled
$ rmu/show after_journal/backup_context/checkpoint foo
JOURNAL IS ENABLED -
    RESERVE 10 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED -
    LOGMINER IS DISABLED
ADD JOURNAL FOO_AIJ_0 -
! FILE DISK:[DIRECTORY]FOO_AIJ_0.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_0.AIJ
ADD JOURNAL FOO_AIJ_1 -
! FILE DISK:[DIRECTORY]FOO_AIJ_1.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_1.AIJ
ADD JOURNAL FOO_AIJ_2 -
! FILE DISK:[DIRECTORY]FOO_AIJ_2.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_2.AIJ
ADD JOURNAL FOO_AIJ_3 -
! FILE DISK:[DIRECTORY]FOO_AIJ_3.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_3.AIJ
ADD JOURNAL FOO_AIJ_4 -
! FILE DISK:[DIRECTORY]FOO_AIJ_4.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_4.AIJ
ADD JOURNAL FOO_AIJ_5 -
! FILE DISK:[DIRECTORY]FOO_AIJ_5.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_5.AIJ
ADD JOURNAL FOO_AIJ_6 -
! FILE DISK:[DIRECTORY]FOO_AIJ_6.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_6.AIJ
ADD JOURNAL FOO_AIJ_7 -
! FILE DISK:[DIRECTORY]FOO_AIJ_7.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_7.AIJ
$ show symbol rdm$aij*
  RDM$AIJ_BACKUP_SEQNO == "-1"
  RDM$AIJ_COUNT == "8"
  RDM$AIJ_CURRENT_SEQNO == "0"
  RDM$AIJ_ENDOFFILE == "2"
  RDM$AIJ_FULLNESS == "0"
  RDM$AIJ_LAST_SEQNO == "-1"
```

```
  RDM$AIJ_NEXT_SEQNO == "0"
  RDM$AIJ_SEQNO == "-1"
$!
$ rmu/backup/nolog foo foo
$!
$ rmu/show after_journal/backup_context/nocheckpoint foo
JOURNAL IS ENABLED -
    RESERVE 10 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED -
    LOGMINER IS DISABLED
ADD JOURNAL FOO_AIJ_0 -
! FILE DISK:[DIRECTORY]FOO_AIJ_0.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_0.AIJ
ADD JOURNAL FOO_AIJ_1 -
! FILE DISK:[DIRECTORY]FOO_AIJ_1.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_1.AIJ
ADD JOURNAL FOO_AIJ_2 -
! FILE DISK:[DIRECTORY]FOO_AIJ_2.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_2.AIJ
ADD JOURNAL FOO_AIJ_3 -
! FILE DISK:[DIRECTORY]FOO_AIJ_3.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_3.AIJ
ADD JOURNAL FOO_AIJ_4 -
! FILE DISK:[DIRECTORY]FOO_AIJ_4.AIJ;1
ADD JOURNAL FOO_AIJ_5 -
! FILE DISK:[DIRECTORY]FOO_AIJ_5.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_5.AIJ
ADD JOURNAL FOO_AIJ_6 -
! FILE DISK:[DIRECTORY]FOO_AIJ_6.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_6.AIJ
ADD JOURNAL FOO_AIJ_7 -
! FILE DISK:[DIRECTORY]FOO_AIJ_7.AIJ;1
    FILE TEST$SCRATCH:FOO_AIJ_7.AIJ
$ show symbol rdm$aij*
  RDM$AIJ_BACKUP_SEQNO == "-1"
  RDM$AIJ_COUNT == "8"
  RDM$AIJ_CURRENT_SEQNO == "0"
  RDM$AIJ_ENDOFFILE == "2"
  RDM$AIJ_FULLNESS == "0"
  RDM$AIJ_LAST_SEQNO == "-1"
  RDM$AIJ_NEXT_SEQNO == "0"
  RDM$AIJ_SEQNO == "-1"
```

1.67.2 RMU Show AIP Command

Displays the contents of the AIP (Area Inventory Pages) structure. The AIP structure provides a mapping for logical areas to physical areas as well as describing each of those logical areas. Information such as the logical area name, length of the stored record, storage thresholds and other information can be displayed using this simple command interface.

Format

RMU/Show AIP root-file-spec [larea-name]

Command Qualifiers	Defaults
/Brief	See description
/Larea=(n [,...])	See description
/Parea=(n [,...])	See description
/Option=Rebuild_Spams	See description
/Output=output-filename	/Output=SYS$OUTPUT
/Type=type-name	See description

Description

The RMU Show AIP command allows the database administrator to display details of selected logical areas or all logical areas in the database.

Command Parameters

root-file-spec
The file specification for the database root file to be processed. The default file extension is .rdb.

larea-name
An optional parameter that allows the logical areas to be selected by name. Only those AIP entries are displayed. This parameter is optional and will default to all logical areas being displayed.

Any partitioned index or table will create multiple logical areas all sharing the same name. This string may contain standard OpenVMS wildcard characters (% and *) so that different names can be matched. Therefore, it is possible for many logical areas to match this name.

The value of **larea-name** may be delimited so that mixed case characters, punctuation and various character sets can be used.

Command Qualifiers

Brief
Displays AIP information in a condensed, tabular form (see example below).

Larea=(n [,...])
Specifies a list of logical area identifiers. The LAREA qualifier and larea-name parameter are mutually exclusive. The default if neither the LAREA or PAREA qualifiers nor the larea-name parameter is specified is to display all AIP entries.

Parea=(n [,...])
Specifies a list of physical area identifiers. The PAREA qualifier and larea-name parameter are mutually exclusive. The default if neither the PAREA or LAREA qualifiers nor the larea-name parameter is specified is to display all AIP entries.

Option=REBUILD_SPAMS
Display only those logical areas which have the REBUILD_SPAMS flag set.

Output [= outout-filename]
This qualifier is used to capture the output in a named file. If used, a standard RMU header is added to identify the command and database being processed. If omitted, the output is written to SYS$OUTPUT and no header is displayed.

Type = type-name
Legal values for type-name are TABLE, SORTED_INDEX, HASH_INDEX, LARGE_OBJECT, and SYSTEM_RECORD.

This qualifier is used in conjunction with **larea-name** to select a subset of the AIP entries that may match a name. For instance, it is legal in Rdb to create a table and an index with the name EMPLOYEES. So using EMPLOYEES/TYPE=TABLE will make the selection unambiguous. It also allows simpler wildcarding. Commands using *EMPLOYEE*/TYPE=TABLE will process only those tables that match and not the associated index logical areas.

Usage Notes

- The database administrator requires RMU$DUMP privilege as this command is closely related to the RMU DUMP LAREA=RDB$AIP command.

- Only AIP entries that are in use are displayed. In contrast, the RMU Dump command also displays deleted and unused AIP entries.

Examples

Example 1

This example uses the name of a known database table to display details for this single logical area.

```
$ RMU/SHOW AIP SQL$DATABASE JOBS

Logical area name JOBS
Type: TABLE
Logical area 85 in mixed physical area 7
Physical area name JOBS
Record length 41
Thesholds are (0, 0, 0)
AIP page number: 151
ABM page number: 0
Snapshot Enabled TSN: 64
```

Example 2

The wildcard string "*EMPLOYEE* matches both indices and table logical areas, so here we use /TYPE to limit the display to just table logical areas. The table EMPLOYEES in the MF_PERSONNEL database is partitioned across three storage areas and hence there exists three logical areas.

```
$ RMU/SHOW AIP SQL$DATABASE *EMPLOYEE*/TYPE=TABLE

Logical area name EMPLOYEES
Type: TABLE
Logical area 80 in mixed physical area 3
Physical area name EMPIDS_LOW
Record length 126
Thesholds are (0, 0, 0)
AIP page number: 150
ABM page number: 0
Snapshot Enabled TSN: 4800

Logical area name EMPLOYEES
Type: TABLE
Logical area 81 in mixed physical area 4
Physical area name EMPIDS_MID
Record length 126
Thesholds are (0, 0, 0)
AIP page number: 151
ABM page number: 0
Snapshot Enabled TSN: 1504
```

```
Logical area name EMPLOYEES
Type: TABLE
Logical area 82 in mixed physical area 5
Physical area name EMPIDS_OVER
Record length 126
Thesholds are (0, 0, 0)
AIP page number: 151
ABM page number: 0
Snapshot Enabled TSN: 1504
```

Example 3

This example shows the REBUILD_SPAMS option used to locate logical areas that require SPAM rebuilds. This may occur because the stored row length changed size or THRESHOLDS were modified for the index or storage map.

```
$ RMU/SHOW AIP/OPTION=REBUILD_SPAMS
_Root: SQL$DATABASE
_Logical area name:

Logical area name ACCOUNT_AUDIT
Type: TABLE
Logical area 86 in uniform physical area 1
Physical area name RDB$SYSTEM
Record length 12
Thesholds are (10, 100, 100)
Flags:
    SPAM pages should be rebuilt
AIP page number: 151
ABM page number: 1004
Snapshot Enabled TSN: 5824

Logical area name DEPARTMENTS_INDEX
Type: SORTED INDEX
Logical area 94 in uniform physical area 10
Physical area name DEPARTMENT_INFO
Record length 430
Thesholds are (30, 65, 72)
Flags:
    SPAM pages should be rebuilt
AIP page number: 151
ABM page number: 2
Snapshot Enabled TSN: 7585
```

Example 4

The /BRIEF qualifier specifies that a condensed tabular output format be used. The /PAREA qualifier is used here to specify that only logical areas stored in physical areas 4 and 5 are to be displayed.

```
$ RMU /SHOW AIP /BRIEF MF_PERSONNEL /PAREA=(4,5)
*-------------------------------------------------------------------------------
* Logical Area Name             LArea PArea   Len Type
*-------------------------------------------------------------------------------
RDB$SYSTEM_RECORD                 60     4   215 SYSTEM RECORD
RDB$SYSTEM_RECORD                 61     5   215 SYSTEM RECORD
EMPLOYEES_HASH                    79     4   215 HASH INDEX
EMPLOYEES                         82     4   121 TABLE
JOB_HISTORY_HASH                  85     4   215 HASH INDEX
JOB_HISTORY                       88     4    42 TABLE
DEPARTMENTS_INDEX                 89     5   430 SORTED INDEX
DEPARTMENTS                       90     5    55 TABLE
```

The columns displayed include:

- Logical Area Name - Name of the logical area stored in the AIP entry
- LArea - Logical area number stored in the AIP entry
- PArea - Physical area number stored in the AIP entry
- Len - Object length stored in the AIP entry
- Type - Object type stored in the AIP entry. The following object types may be displayed:
  - UNKNOWN - The logical area type is unknown or has not been set
  - TABLE - A data table type
  - SORTED INDEX - A sorted index type
  - HASH INDEX - A hashed index type
  - SYSTEM RECORD - A system record type
  - LARGE OBJECT - A large object (BLOB) type

1.67.3 RMU Show Audit Command

Displays the set of security auditing characteristics established by the RMU Set command with Audit qualifier.

Format

RMU/Show Audit root-file-spec

Command Qualifiers	Defaults
/All	See description
/Daccess[=object-type[,...]]	See description
/Every	See description
/Flush	See description
/Identifiers	See description
/Output[=file-name]	/Output=SYS$OUTPUT
/Protection	See description
/Rmu	See description
/Type={Alarm|Audit}	Alarm and Audit

Description

The RMU Show Audit command is the Oracle Rdb equivalent to the DCL SHOW AUDIT command. Because Oracle Rdb security auditing uses many OpenVMS system-level auditing mechanisms, certain auditing characteristics such as /FAILURE_MODE can only be displayed using the OpenVMS SHOW AUDIT command, which requires the OpenVMS SECURITY privilege.

Command Parameters

root-file-spec
The root file specification of the database for which you want auditing information to be displayed.

Command Qualifiers

All
Displays all available auditing information for the database, including the following: whether security auditing and security alarms are started or stopped; types of security events currently enabled for alarms and audits; identifiers currently enabled for auditing; and whether forced write operations are enabled or disabled.

Daccess[=object-type[, . . .]]
Indicates whether the general DACCESS audit event class is currently enabled. Specifying one or more object types with the Daccess qualifier displays the object types and their associated privileges that are currently enabled for DACCESS event auditing. If you specify more than one object type, enclose the list of object types within parentheses.

The valid object types are:

DATABASE
TABLE
COLUMN
SEQUENCE
ROUTINE
MODULE
VIEW

Every
Displays the current setting for the first or every DACCESS event auditing for the database.

Flush
Displays the current setting for forced write operations on audit journal records for the database.

Identifiers
Displays the user identification codes (UICs) of the users currently enabled for DACCESS event auditing of specified objects.

Output[=file-name]
Controls where the output of the command is sent. If you do not enter the Output qualifier, or if you enter the Output qualifier without a file specification, the output is sent to the current process default output stream or device.

Protection
Displays whether auditing is currently enabled for the PROTECTION audit event class.

Rmu
Displays whether auditing is currently enabled for the RMU event class.

Type={Alarm | Audit}
Displays information about security alarms or security auditing. If you do not specify the Type qualifier, Oracle RMU displays information about both security alarms and security auditing.

Usage Notes

- To use the RMU Show Audit command for a database, you must have the RMU$SECURITY privilege in the root file ACL for the database or the OpenVMS SECURITY or BYPASS privilege.
- If you do not specify any qualifiers with the RMU Show Audit command, the currently enabled alarm and audit security events are displayed.
- Use the RMU Show Audit command to check which auditing features are enabled whenever you plan to enable or disable audit characteristics with a subsequent RMU Set Audit command.
- When the RMU Show Audit command is issued for a closed database, the command executes without other users being able to attach to the database.

Examples

Example 1

The following command shows that alarms are enabled for the RMU and PROTECTION audit classes for the mf_personnel database. Note that the display shows that alarms are also enabled for the AUDIT audit class. The AUDIT audit class is always enabled and cannot be disabled.

```
$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    ACCESS (disabled)

Security alarms STOPPED for:
    PROTECTION (enabled)
    RMU (enabled)
    AUDIT (enabled)
    ACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    None
```

Example 2

In the following example, the first command enables and starts alarms for the RMU audit class for the mf_personnel database. Following the first command is the alarm that is displayed on a security terminal when the first command is executed. The second command displays the auditing characteristics that have been enabled and started. The RMU Show Audit command with the All qualifier causes the alarm at the end of the example to be displayed on the security terminal. Note that security-enabled terminals only receive alarms if alarms have been both enabled *and* started.

```
$ RMU/SET AUDIT/TYPE=ALARM/ENABLE=RMU/START MF_PERSONNEL

%%%%%%%%%%  OPCOM   8-JUL-1996 09:41:01.19  %%%%%%%%%%
Message from user RICK on MYNODE
Oracle Rdb Security alarm (SECURITY) on MYNODE, system id:      32327
Database name:          DDV21:[RICK.SQL]MF_PERSONNEL.RDB;1
Auditable event:        Auditing change
PID:                    21212274
Event time:             8-JUL-1996 09:41:01.17
User name:              RICK
RMU command:            RMU/SET AUDIT/TYPE=ALARM/ENABLE=RMU/START MF_PERSONNEL
Sub status:             RMU required privilege
Final status:           %SYSTEM-S-NORMAL
RMU privilege used:     RMU$SECURITY

$ RMU/SHOW AUDIT/ALL MF_PERSONNEL
Security auditing STOPPED for:
    PROTECTION (disabled)
    RMU (disabled)
    AUDIT (enabled)
    ACCESS (disabled)

Security alarms STARTED for:
    PROTECTION (disabled)
    RMU (enabled)
    AUDIT (enabled)
    ACCESS (disabled)

Audit flush is disabled

Audit every access

Enabled identifiers:
    None
```

RMU Show Audit Command

```
%%%%%%%%%%  OPCOM   8-JUL-1996 09:43:07.94  %%%%%%%%%%
Message from user RICK on MYNODE
Oracle Rdb Security alarm (SECURITY) on MYNODE, system id:      32327
Database name:          DDV21:[RICK.SQL]MF_PERSONNEL.RDB;1
Auditable event:        Attempted RMU command
PID:                    21212274
Event time:              8-JUL-1996 09:43:07.92
User name:              RICK
RMU command:            RMU/SHOW AUDIT/ALL MF_PERSONNEL
Access requested:       RMU$SECURITY
Sub status:             RMU required privilege
Final status:           %SYSTEM-S-NORMAL
RMU privilege used:     RMU$SECURITY
```

1.67.4 RMU Show Corrupt_Pages Command

Indicates which pages, storage areas, or snapshot files are corrupt or inconsistent by displaying the contents of the corrupt page table (CPT). Corrupt pages are logged to the CPT, which is maintained in the database root file.

Format

RMU/Show Corrupt_Pages root-file-spec

Command Qualifiers	Defaults
/Options=({Normal|Debug|Full})	/Options=(Normal)
/Output[=file-name]	/Output=SYS$OUTPUT

Command Parameters

root-file-spec
The root file specification of the database for which you want the corrupt or inconsistent storage areas or snapshot files logged to the CPT to be displayed.

Command Qualifiers

Options=({Normal or Full or Debug})
Specifies the type of information you want displayed, as follows:

- Normal

  Displays the active CPT entries and the corrupt or inconsistent areas sorted by area and page.

- Full

  Displays the same information as Normal plus the disks on which the active CPT entries and the corrupt or inconsistent areas or snapshot files are stored—sorted by disk, area, and page.

- Debug

  Provides a dump of the entire CPT and lists all the storage areas.

Options=(Normal) is the default qualifier.

Output[=file-name]
Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify only a file name.

Usage Notes

- To use the RMU Show Corrupt_Pages command for a database, you must have the RMU$BACKUP, RMU$RESTORE, or RMU$VERIFY privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or OpenVMS BYPASS privilege.
- You can repair and remove a corrupt snapshot file from the CPT by issuing the RMU Repair command with the Initialize=(Snapshots) qualifier. Using the Repair command in this case is faster than performing a restore operation. See Section 1.43 for details.

Examples

Example 1

The following example shows the output from the RMU Show Corrupt_Pages command when page 1 in area 3 is marked as corrupt:

```
$ RMU/SHOW CORRUPT_PAGES MF_PERSONNEL
*-------------------------------------------------------------------
* Oracle Rdb V7.0-00                          8-JUL-1996 13:46:20.77
*
* Dump of Corrupt Page Table
*     Database: USER1:[DB]MF_PERSONNEL.RDB;1
*
*-------------------------------------------------------------------

Entries for storage area EMPIDS_MID
-----------------------------------

    Page 1
        - AIJ recovery sequence number is -1
        - Area ID number is 3
        - Consistency transaction sequence number is 0:0
        - State of page is: corrupt

*-------------------------------------------------------------------
* Oracle Rdb V7.0-00                          8-JUL-1996 13:46:21.17
*
* Dump of Storage Area State Information
*     Database: USER1:[DB]MF_PERSONNEL.RDB;1
*
*-------------------------------------------------------------------

All storage areas are consistent.
```

1.67.5 RMU Show Locks Command

Displays current information about the OpenVMS locks database on your node. It provides information concerning lock activity and contention for all active databases.

Format

RMU/Show Locks [root-file-spec]

Command Qualifiers	Defaults
/Lock = lock-list	None
/Mode = (mode-list)	None
/Options = (option-list)	See description
/Output[=file-name]	/Output=SYS$OUTPUT
/Process = process-list	None
/Resource-type=resource-type-list	None

Description

In a clustered environment, the RMU Show Locks command displays detailed lock information for your current node and may display information about known remote locks.

The RMU Show Locks command displays information about process locks for all active databases on a specific node. A process requesting a lock can have one of three states: owning, blocking, or waiting. A process is considered to be owning when the lock request is granted. A process is considered to be blocking when the lock request is granted and its mode is incompatible with other waiting locks. A process is considered to be waiting when it is prevented from being granted a lock due to the presence of other granted locks whose modes are incompatible with the process' requested mode.

Using the RMU/SHOW LOCKS command can be difficult on systems with multiple open databases due to the amount of output and difficulty in determining what database a particular lock references. The RMU/SHOW LOCKS command, when supplied with a root file specification, can be used to additionally filter lock displays to a specific database. Note that in some cases the RMU/SHOW LOCKS command may be unable to filter locks prior to display. And when using the database "LOCK PARTITIONING IS ENABLED" feature for a database, the RMU/SHOW LOCKS command with a root file specification will be unable to associate area, page, and record locks with the

specified database because the database lock is not the lock tree root for these lock types.

The values for the Mode qualifier: Blocking and Waiting, can be combined with the Process and Lock qualifiers to indicate which of the following types of information is displayed:

- If the Blocking option is specified, information is displayed about processes whose locks are blocking other processes' locks.
- If the Waiting option is specified, information is displayed about processes whose locks are waiting for other processes' locks.
- If the Process qualifier is specified, information is displayed for a specified list of processes.
- If the Lock qualifier is specified, information is displayed for a specified list of locks. When no qualifiers are specified, a list of all active locks in the OpenVMS locks database is displayed.

Use the qualifiers individually or in combination to display the required output. See Table 1–18 for all possible qualifier combinations and the types of output they produce. If you do not specify any qualifiers, a complete list of locks is displayed. The volume of information from this report can be quite large. Therefore, you should use the Output qualifier to direct output to a file, instead of allowing the output to display to SYS$OUTPUT. Each output contains a heading that indicates what qualifiers, if any, were used to generate the output.

Table 1–18 Lock Qualifier Combinations

Object	Mode Argument	Option Argument	Output
Process			Locks for the specified processes
Process	Blocking		Processes blocking the specified processes
Process	Waiting		Processes waiting for the specified processes
Process		All	Process locks for the specified processes
Process		Full	Special process locks for the specified processes

(continued on next page)

Table 1–18 (Cont.) Lock Qualifier Combinations

Object	Mode Argument	Option Argument	Output
Process	Blocking, Waiting		Processes blocking and waiting for the specified processes
Process	Blocking	Full	Special process locks blocking the specified processes
Process	Waiting	Full	Special process locks waiting for the specified processes
Process	Blocking, Waiting	Full	Special process locks blocking and waiting for the specified processes
Process		All, Full	Process and special process locks for the specified processes
Lock			Locks for the specified locks
Lock	Blocking		Processes blocking the specified locks
Lock	Waiting		Processes waiting for the specified locks
Lock		Full	Special process locks for the specified locks
Lock	Blocking	Full	Special process locks blocking the specified locks
Lock	Waiting	Full	Special process locks waiting for the specified locks
Lock	Blocking, Waiting		Processes blocking and waiting for the specified locks
Lock	Blocking, Waiting	Full	Special process locks blocking and waiting for the specified locks
	Blocking		Lock requests that are blocked
	Waiting		Lock requests that are waiting
	Blocking, Waiting		Lock requests that are blocking and waiting
Process Lock			Locks for specified processes and locks
Process Lock	Blocking		Processes blocking the specified processes and locks

(continued on next page)

Table 1–18 (Cont.) Lock Qualifier Combinations

Object	Mode Argument	Option Argument	Output
Process Lock	Waiting		Processes waiting for the specified processes and locks
Process Lock	Blocking, Waiting		Processes blocking and waiting for the specified processes and locks
Process Lock	Blocking	Full	Special process locks blocking the specified processes and locks
Process Lock	Waiting	Full	Special process locks waiting for the specified processes and locks
Process Lock		All	Process locks for the specified processes and locks
Process Lock		Full	Special process locks for the specified processes and locks
Process Lock	Blocking	Full	Special process locks blocking the specified processes and locks
Process Lock		All, Full	Process and special process locks for the specified processes and locks

You can display only those processes that you have privilege to access. Furthermore, certain *special* database processes are not displayed, unless you specifically indicate that all processes are to be displayed. The report heading indicates what qualifiers were used to generate the output.

Command Parameters

root-file-spec
The root file specification of the database for which you want to filter lock displays. Optional parameter.

Command Qualifiers

Lock=lock-list
Displays information for each of the specified locks. When combined with the Mode=Blocking qualifier, the Lock qualifier displays information about processes whose locks are blocking the specified locks. When combined with the Mode=Waiting qualifier, the Lock qualifier displays information about processes whose lock requests are waiting for the specified locks.

One or more locks can be specified; if more than one lock is specified, they must be enclosed in parentheses and separated by commas. The lock identifier is an 8-digit hexadecimal number, and must be local to the node on which the RMU Show Locks command is issued. To see the lock identifier upon which a process is waiting, you can do either of the following:

- Invoke the character cell Performance Monitor "Stall Messages" display.
- Invoke the Performance Monitor from your PC and select Displays ⇒ Process Statistics ⇒ Stall Messages.

Mode=(mode-list)

Indicates the lock mode to be displayed. If you specify more than one option in the mode-list, you must separate the options with a comma, and enclose the mode-list in parentheses. The following lock mode options are available:

- Blocking

  Displays the set of processes whose locks are blocking the lock requests of other processes. A process is considered to be waiting when it has requested a lock mode that is incompatible with existing granted lock modes; in this case, the requestor is the waiting process and the grantors are the blocking processes.

  The first line of output identifies a process that is waiting for a lock request to be granted. All subsequent lines of output identify those processes that are preventing the lock request from being granted. When multiple processes are waiting for the same lock resource, multiple sets of process-specific information, one for each waiting process, are displayed.

- Culprit

  Displays the set of locks for processes that are blocking other processes but are themselves not locked. The output represents the processes that are the source of database stalls and performance degradation.

- Waiting

  Displays the set of processes whose lock requests are waiting due to incompatible granted locks for other processes. A process is considered to be blocking others when it has been granted a lock mode that is incompatible with requested lock modes; in this case, the "Blocker" is the blocking process and the "Waiting" are the waiting processes.

  A requesting process can appear to be waiting for other lock requestors. This condition occurs when there are many processes waiting on the same lock resource. Depending upon the sequence of processes in the wait queue, certain waiting processes appear to be blocking other waiting processes because, eventually, they will be granted the lock first.

The first line of output identifies a process that has been granted a lock on a resource. All subsequent lines of output identify those processes that are waiting for lock requests on the same resource to be granted. When multiple processes are blocking the same lock resource, multiple sets of process-specific information, one for each blocking process, are displayed.

Options=(option-list)

Indicates the type of information and the level of detail the output will include. If you do not specify the Options qualifier, the default output is displayed. If you specify more than one type of output for the Options qualifier, you must separate the options with a comma, and enclose the options list within parentheses. The following options are available:

- All

  Used when you want the complete list of process locks; by default, lock information for only the specified process is displayed. When you specify the All option, information is displayed for all other processes that have a need to know the lock held by the specific process. This method is an easy way to display all of a process' locks and to see what other processes are also using the same resource.

  If the Mode qualifier is specified, the Options=(All) qualifier is ignored.

- Full

  Indicates that special database processes are to be displayed. Some special database processes, such as monitors, perform work on behalf of a database. These database processes frequently request locks that by design conflict with other processes' locks; the granting of these locks indicates an important database event.

  By default, these special database processes are not displayed because they increase the size of the output.

Output[=file-name]

Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify only a file name.

Process=process-list

Displays information for each lock held or requested by the specified processes when used by itself. When the Process qualifier is combined with the Mode=Blocking qualifier, information is displayed about processes whose locks are blocking lock requests by the specified waiting processes.

Note

When the Process qualifier is specified without any Options qualifier values, all locks for the processes are displayed, including owning, blocking, and waiting locks.

One or more processes can be specified; if more than one process is specified, they must be enclosed within parentheses and separated by commas. The process identifier is an 8-digit hexadecimal number, and must be local to the node on which the RMU Show Locks command is issued. The process ID must include all eight characters; the node identifier portion of the process ID cannot be excluded. To get more information, use the Options=All qualifier to display all users using processes' locks.

Resource_type=resource-type-list
Displays information for each lock held or requested by the specified resource type. Only the specific resource types will be displayed. This permits, for example, only PAGE or RECORD lock types to be selected.

One or more resouce types can be specified; if more than one type is specified, they must be enclosed within parentheses and separated by commas.

The following keywords are allowed with the Resource_type qualifier.

Table 1–19 RESOURCE_TYPE Keywords

Internal Lock Type Name	Keyword(s)
ACCESS	ACCESS
ACTIVE	ACTIVE
AIJDB	AIJDB
AIJFB	AIJFB
AIJHWM	AIJHWM, AIJ_HIGH_WATER_MARK
AIJLOGMSG	AIJ_LOG_MESSAGE
AIJLOGSHIP	AIJ_LOG_SHIPPING
AIJOPEN	AIJ_OPEN
AIJSWITCH	AIJ_SWITCH

(continued on next page)

Table 1–19 (Cont.) RESOURCE_TYPE Keywords

Internal Lock Type Name	Keyword(s)
AIJ	AIJ
AIPQHD	AIP
ALS	ALS_ACTIVATION
BCKAIJ	AIJ_BACKUP, BCKAIJ
BCKAIJ_SPD	AIJ_BACKUP_SUSPEND
BUGCHK	BUGCHECK
CHAN	CHAN, FILE_CHANNEL
CLIENT	CLIENT
CLOSE	CLOSE
CLTSEQ	CLTSEQ
CPT	CORRUPT_PAGE_TABLE, CPT
DASHBOARD	DASHBOARD_NOTIFY
DBK_SCOPE	DBKEY_SCOPE
DBR	DBR_SERIALIZATION
DB	DATABASE
FIB	FAST_INCREMENTAL_BACKUP, FIB
FILID	FILID
FRZ	FREEZE
GBL_CKPT	GLOBAL_CHECKPOINT
GBPT_SLOT	GLOBAL_BPT_SLOT
KROOT	KROOT
LAREA	LAREA, LOGICAL_AREA
LOGFIL	LOGFIL
MEMBIT	MEMBIT
MONID	MONID, MONITOR_ID
MONITOR	MONITOR
NOWAIT	NOWAIT
PLN	DBKEY, RECORD, PLN

(continued on next page)

Table 1–19 (Cont.) RESOURCE_TYPE Keywords

Internal Lock Type Name	Keyword(s)
PNO	PAGE, PNO
QUIET	QUIET
RCACHE	RCACHE
RCSREQUEST	RCS_REQUEST
RCSWAITRQST	RCS_WAIT_REQUEST
REL_AREAS	RELEASE_AREAS
REL_GRIC_REQST	RELEASE_GRIC_REQUEST
RMUCLIENT	RMU_CLIENT
ROOT_AREA	DUMMY_ROOT_AREA
RO_L1	L1_SNAP_TRUNCATION
RTUPB	RTUPB
RUJBLK	RUJBLK
RW_L2	L2_SNAP_TRUNCATION
SAC	SNAP_AREA_CURSOR
SEQBLK	SEQBLK
STAREA	STORAGE_AREA, PAREA
STATRQST	STATISTICS_REQUEST
TRM	TERMINATION
TSNBLK	TSNBLK
UTILITY	UTILITY

The RESOURCE_TYPE qualifier is incompatible with the MODE, LIMIT, LOCK and PROCESS qualifiers.

Usage Notes

- To use the RMU Show Locks command for a database, you must have the OpenVMS WORLD privilege.
- When you specify a list of processes or lock identifiers, make sure the processes or locks are local to the node on which the RMU Show Locks command is issued.

- To display the complete list of locks in the OpenVMS locks database, do not specify the Mode=Blocking or Waiting qualifier. The volume of information from this report can be quite large.
- If you have entered an Oracle RMU command and there are no locks on your node, you receive the following message:

```
%RMU-I-NOLOCKSOUT, No locks on this node with the specified
 qualifiers.
```

- When you use the RMU Show Locks command to display locks, the "requested" and "granted" modes of the given lock are displayed. The definitions for the two fields follow:
  - Requested

    This is the mode for which the process has requested the lock. Valid modes are NL, CR, CW, PR, PW, and EX. This mode is not guaranteed to be granted; some locks are intentionally held in conflicting modes forever (for example, the "termination" lock).
  - Granted

    This is the mode that the process was last granted for the lock. Valid modes are NL, CR, CW, PR, PW, and EX. Furthermore, if the lock has never been previously granted, the lock mode is displayed as NL mode.

  Table 1–20 shows the compatibility of requested and granted lock modes.

Table 1–20 Lock Mode Compatibility

	Mode of Currently Granted Locks					
Mode of Requested Lock	**NL**	**CR**	**CW**	**PR**	**PW**	**EX**
NL	Yes	Yes	Yes	Yes	Yes	Yes

Key to Lock Modes

NL—Null Lock
CR—Concurrent Read
CW—Concurrent Write
PR—Protected Read
PW—Protected Write
EX—Exclusive Lock
Yes—Locks compatible
No—Locks not compatible

(continued on next page)

Table 1–20 (Cont.) Lock Mode Compatibility

Mode of Requested Lock	Mode of Currently Granted Locks					
	NL	CR	CW	PR	PW	EX
CR	Yes	Yes	Yes	Yes	Yes	No
CW	Yes	Yes	Yes	No	No	No
PR	Yes	Yes	No	Yes	No	No
PW	Yes	Yes	No	No	No	No
EX	Yes	No	No	No	No	No

Key to Lock Modes

NL—Null Lock
CR—Concurrent Read
CW—Concurrent Write
PR—Protected Read
PW—Protected Write
EX—Exclusive Lock
Yes—Locks compatible
No—Locks not compatible

- If the "requested" and "granted" lock modes differ, then the lock requested is currently blocked on either the "wait" or "conversion" queue. If the modes are the same, then the lock has been granted.
- The OpenVMS distributed lock manager does not always update the requested lock mode. This means that potentially conflicting information can be displayed by the RMU Show Locks utility.
- The requested lock mode is updated only under the following situations:
  - The lock request is for a remote resource.
  - The lock request is a Nowait request.
  - The lock request could not be granted due to a lock conflict (that is, it was canceled by the application or aborted due to lock timeout or deadlock).
  - The lock request is the first for the resource.
- Consider the following RMU Show Locks output:

```
------------------------------------------------------------------------
Resource Name: page 533
Granted Lock Count: 1,  Parent Lock ID: 01000B6C,   Lock Access Mode:
Executive,
Resource Type:
      Global,  Lock Value Block: 03000000 00000000 00000000 00000002

  -Master Node Info-  --Lock Mode Information--     -Remote Node Info-
ProcessID Lock ID  SystemID Requested Granted Queue  Lock ID SystemID
2040021E  0400136A  00010002  EX      CR      GRANT  0400136A 00010002
------------------------------------------------------------------------
```

In this example, it is ordinarily difficult to explain how such a combination of lock modes could occur. Note that the CR (concurrent read) mode is on the Grant queue (not the Conversion queue).

Knowledge of the operating environment is necessary to know that there was only one node on this system. It turns out that two lock requests actually occurred to generate this output, in the *opposite* order of what appears to have occurred.

The first lock request was for EX (exclusive), which was immediately granted. Thus, the Requested and Granted modes were updated according to situation 4. Then, the lock was demoted from EX to CR mode, which was also immediately granted. However, the Requested field was *not* updated because none of the four preceding rules was true, so the Requested mode was never updated to reflect the CR lock request.

- Additional field definitions for the command output are as follows:
  - Conversion Lock Count:

    This is the total number of locks that are currently on the conversion queue of the resource associated with the lock. The conversion queue is a queue containing locks that have been granted but are in the process of being converted to a new, conflicting lock mode. Note that the total number of granted locks on the resource is equal to the sum of the granted and conversion queues.
  - Granted Lock Count:

    This is the total number of locks that are currently in the granted queue of the resource associated with the lock. Note that the total number of granted locks on the resource is equal to the sum of the locks in the granted and conversion queues.
  - Lock Access Mode:

    This is the access mode associated with the first lock request on the indicated resource name. Possible resource names are Kernel = 0, Executive = 1, Supervisor = 2, User = 3.

- Lock ID

  This is the lock identifier for the indicated process on the indicated resource name. Each process lock request on a resource results in a unique lock identifier being issued.

- Lock Value Block:

  This field displays the contents of the lock value block associated with the indicated resource.

- Number of Sub-Resources:

  This is the number of subresources of the indicated resource name. Note, however, that the number of subresources may differ from the number of child locks of the lock because any number of processes may lock the resource. If any of these processes then locks another resource, and in doing so creates the lock as a child of the other lock, then the second resource becomes a subresource of the first resource.

- Parent Lock Count:

  This is the number of locks that have this lock as a parent.

- Parent Lock ID:

  This is the lock identifier of the parent lock of the indicated lock.

- Process Group:

  This is the user identification code (UIC) group number of the process that took out the first lock on the indicated process, thereby creating the resource.

- ProcessID

  This is the process identifier of the process that owns the lock.

- Process Name

  This is the name of the process that owns the lock.

- Queue

  This field identifies the queue on which the lock request has been placed. Valid queues include Grant, Conversion, and Waiting. The Grant queue means the lock has been granted and is not in the process of being converted to another lock mode. The Conversion queue means the lock has been granted but is in the process of being converted to a new, conflicting lock mode. The Waiting queue means the lock request is new and conflicts with a granted or converting lock.

- Requested

This is the mode for which the process requested the lock. Valid modes are NL (null), PR (protected read), PW (protected write), CR (concurrent read), CW (concurrent write), and EX (exclusive).

- Resource:

  That which is being locked.

- Resource Name:

  This is the name of the Oracle Rdb or Oracle CODASYL DBMS resource. Optionally included in the resource name is additional information, such as page and line numbers.

- Resource Type:

  This field indicates if the lock is global or local.

- System ID

  This is the cluster system identification of the node currently overseeing the resource associated with the indicated lock for the indicated process.

Examples

Example 1

The following example shows a portion of the output generated by the command, RMU Show Locks command with the Process=44A047C9 qualifier. The actual report is several pages in length because it shows *all* the locks held by process ID 44A047C9. The report text shows the resource on which the lock is held, ID information, and lock status (Requested and Granted).

```
$ RMU/SHOW LOCKS/PROCESS=44A047C9

================================================================================
SHOW LOCKS/PROCESS Information
================================================================================
   .
   .
   .
--------------------------------------------------------------------------------
Resource: page 352
          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      7CC80BC8  00020025  PR        PR

--------------------------------------------------------------------------------
Resource: cluster membership
```

```
          ProcessID Process Name       Lock ID   System ID Requested Granted
          --------- ---------------    --------- --------- --------- -------
Owner:    44A047C9  USER1.........     16180C1A  00020025  PR        PR

   .
   .
   .
-------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name       Lock ID   System ID Requested Granted
          --------- ---------------    --------- --------- --------- -------
Owner:    44A047C9  USER1.........     45983EC0  00020025  EX        EX

   .
   .
   .
-------------------------------------------------------------------------------
Resource: logical area 33

          ProcessID Process Name       Lock ID   System ID Requested Granted
          --------- ---------------    --------- --------- --------- -------
Owner:    44A047C9  USER1.........     0480973C  00020025  CR        NL
-------------------------------------------------------------------------------
Resource: logical area 53

          ProcessID Process Name       Lock ID   System ID Requested Granted
          --------- ---------------    --------- --------- --------- -------
Owner:    44A047C9  USER1.........     56009774  00020025  EX        EX

   .
   .
   .
```

Example 2

The following display shows the output generated by the RMU Show Locks command with the Process=44A047C9 and the Mode=Waiting qualifiers. The report shows that process ID 44A045D1 is waiting for the exclusive (EX) lock held by the specified process (44A047C9) on logical area 39 to be released.

```
$ RMU/SHOW LOCKS/PROCESS=44A047C9/MODE=WAITING
===============================================================================
SHOW LOCKS/PROCESS/WAITING Information
===============================================================================

-------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name       Lock ID   System ID Requested Granted
          --------- ---------------    --------- --------- --------- -------
Blocker:  44A047C9  USER1.........     45983EC0  00020025  EX        EX
Waiting:  44A045D1  _RTA11:.......     3B5467DA  00020025  CR        NL
```

This command identifies the waiting process. If you specify the ID of a process that is itself a waiting process, Oracle RMU returns the following message:

```
no locks on this node with the specified qualifiers.
```

Example 3

The following display shows the output generated by the RMU Show Locks command with the Process=44A045D1 and the Mode=Blocking qualifiers. The report shows that process ID 44A047C9 has an exclusive (EX) lock on logical area 39, and is blocking the specified process (44A045D1).

```
$ RMU/SHOW LOCKS/PROCESS=44A045D1/MODE=BLOCKING
================================================================================
SHOW LOCKS/BLOCKING Information
================================================================================

--------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Waiting:  44A045D1  _RTA11:.........     3B5467DA  00020025  CR        NL
Blocker:  44A047C9  USER1..........     45983EC0  00020025  EX        EX
```

This command identifies the blocking process. If you specify the ID of a process that is itself the blocking process, Oracle RMU returns the following message:

```
no locks on this node with the specified qualifiers.
```

Example 4

The following display shows the output generated by the RMU Show Locks command with the Lock=45983EC0 and the Mode=Waiting qualifiers. The report is identical to the display shown in Example 2 because process ID 44A047C9 has taken out only one lock. If process ID 44A047C9 held multiple locks, Example 2 would display all of them, but this example would only display lock information for lock ID 45983EC0.

```
$ RMU/SHOW LOCKS/LOCK=45983EC0/MODE=WAITING MF_PERSONNEL
================================================================================
SHOW LOCKS/LOCK/WAITING Information
================================================================================

--------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Blocker:  44A047C9  USER1..........     45983EC0  00020025            EX
Waiting:  44A045D1  _RTA11:.........     3B5467DA  00020025  CR        NL
```

Example 5

The following display shows a portion of the output generated by the RMU Show Locks command and the Process=44A047C9 and the Options=All qualifiers. The actual report is several pages in length because it lists all the resources that have locks held by process ID 44A047C9 and all the locks on the same resource held by other processes. Compare this report with the one shown in Example 1.

```
$ RMU/SHOW LOCKS/PROCESS=44A047C9/OPTIONS=ALL
================================================================================
SHOW LOCKS/PROCESS Information
================================================================================
   .
   .
   .
--------------------------------------------------------------------------------
Resource: page 352

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      7CC80BC8  00020025  PR        PR
Owner:    44A045D1  _RTA11:........     134C0979  00020025  PR        CR

--------------------------------------------------------------------------------
Resource: cluster membership

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      16180C1A  00020025  PR        PR
Owner:    44A045D1  _RTA11:........     333C95ED  00020025  PR        PR

   .
   .
   .
--------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      45983EC0  00020025  EX        EX
Waiting:  44A045D1  _RTA11:........     3B5467DA  00020025  CR        NL

   .
   .
   .
--------------------------------------------------------------------------------
Resource: logical area 33

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      0480973C  00020025  CR        NL
Owner:    44A045D1  _RTA11:........     31900BAE  00020025  CR        CR

--------------------------------------------------------------------------------
Resource: logical area 53
```

```
          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Owner:    44A047C9  USER1.........      56009774  00020025  EX        EX

   .
   .
   .
```

Example 6

The following display shows the output generated by the RMU Show Locks command with the Mode=(Waiting,Blocking) and the Process=(44A045D1,44A047C9) qualifiers. This command combines the output shown in Examples 3 and 4 into a single report.

```
$ RMU/SHOW LOCKS/MODE=(WAITING,BLOCKING)/PROCESS=(44A045D1,44A047C9)
================================================================================
SHOW LOCKS/PROCESS/BLOCKING Information
================================================================================

--------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Waiting:  44A045D1  _RTA11:........     3B5467DA  00020025  CR        NL
Blocker:  44A047C9  USER1.........      45983EC0  00020025  EX        EX

================================================================================
SHOW LOCKS/PROCESS/WAITING Information
================================================================================

--------------------------------------------------------------------------------
Resource: logical area 39

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- ---------------     --------- --------- --------- -------
Blocker:  44A047C9  USER1.........      45983EC0  00020025  EX        EX
Waiting:  44A045D1  _RTA11:........     3B5467DA  00020025  CR        NL
```

Example 7

When possible, the RMU Show Locks command formats the area number for page locks. When the area lock can be determined for the page lock, the area number is included in the output, as seen in the following example:

```
Resource Name: page 566 (area 1)
Granted Lock Count: 4, Parent Lock ID: 7E00FAFC, Lock Access Mode: Executive,
Resource Type: Global, Lock Value Block: 00000000 00000000 00000000 03000000

          -Master Node Info- --Lock Mode Information-- -Remote Node Info-
ProcessID Lock ID   SystemID Requested Granted   Queue Lock ID  SystemID
2504D32C  6400199C  00010028 CR        PR        GRANT 6400199C 00010028
2504D32C  58002984  00010028 CR        PR        GRANT 58002984 00010028
```

1.67.6 RMU Show Logical_Names Command

Displays logical names known by various components of Oracle Rdb.

Format

RMU/Show Logical_Names [logical-name]

Command Qualifiers	Defaults
/Description	None
/Output=file-name	SYS$OUTPUT
/Undefined	None

Description

The RMU Show Logical_Names command displays the definitions of logical names known by various components of Oracle Rdb. You can specify all logical names or just one. The output format is similar to that of the DCL SHOW LOGICALS command.

Command Parameters

logical-name

Use this option to display the definition of one logical name. If you omit the logical name, the definitions of all logical names known to Oracle Rdb are displayed.

Command Qualifiers

Description

The /Description qualifier retrieves a brief description of the logical name and displays it along with the current definition. If wildcards are used for the logical name, then any matching logical names will also include output of the description.

The following example shows the use of the Description qualifier, a wildcard logical name specification and the use of the Undefined qualifier to include output (even for logical names not defined for this process).

```
$ RMU/SHOW LOGICAL_NAME/UNDEFINE/DESCRIPTION RDMS$BIND_WORK*
  "RDMS$BIND_WORK_FILE" = Undefined

       You can define this logical name to redirect the location of
       temporary files that Oracle Rdb creates for use in matching
       operations. These temporary files are deleted when they are no
       longer used.

       See also the logical name RDMS$BIND_WORK_VM.

  "RDMS$BIND_WORK_VM" = Undefined

       This logical name permits you to reduce the overhead of disk
       I/O for matching operations by letting you specify the amount of
       virtual memory (VM), in bytes, to be allocated to your process.
       Once the allocation is exhausted, additional data values will be
       written to a temporary file on disk.

       If the logical name RDMS$BIND_WORK_FILE is undefined, the
       temporary file is located in SYS$LOGIN, otherwise the value
       defined by RDMS$BIND_WORK_FILE will be used.

       The default is 100,000 bytes. The maximum allowed value is
       restricted only by the amount of memory available on your
       system.
```

The definitions of all logical names are maintained in a HELP library called SYS$HELP:RMUDISPLAY73.HLB. Users can also use the DCL HELP command to query this help library.

```
$ HELP/LIBR=SYS$HELP:RMUDISPLAY73.HLB Rdb_Logical_names RDMS$RUJ

RDB_LOGICAL_NAMES

  RDMS$RUJ

       You can use the RDMS$RUJ logical name to locate the .ruj file on
       a different disk and directory from the default directory. This
       can help to reduce contention in that directory.

Topic?
```

Output=file-name

Specifies the name of the file where output is to be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

Undefined

Use the Undefined qualifier to display a list of both defined and undefined logicals.

Examples

Example 1

The following example displays defined logical names known to Oracle Rdb.

```
$ rmu/sho log
  "RDM$BIND_ABS_LOG_FILE" = "ABS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_ALS_OUTPUT_FILE" = "ALS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_DBR_LOG_FILE" = "DBR_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_HOT_OUTPUT_FILE" = "AIJSERVER_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_LCS_OUTPUT_FILE" = "LCS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_LRS_OUTPUT_FILE" = "LRS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_RCS_LOG_FILE" = "RCS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_RCS_LOG_HEADER" = "0" (LNM$SYSTEM_TABLE)
  "RDM$BUGCHECK_DIR" = "DISK$RANDOM:[BUGCHECKS.RDBHR]" (LNM$SYSTEM_TABLE)
  "RDM$MONITOR" = "SYS$SYSROOT:[SYSEXE]" (LNM$SYSTEM_TABLE)
```

Example 2

This example displays both defined and undefined logical names.

```
$ rmu/sho log /undefined ! Display them all
  "RDMS$AUTO_READY" = Undefined
  "RDM$BIND_ABS_GLOBAL_STATISTICS" = Undefined
  "RDM$BIND_ABS_LOG_FILE" = "ABS_PID.OUT" (LNM$SYSTEM_TABLE)
  "RDM$BIND_ABS_OVERWRITE_ALLOWED" = Undefined
  "RDM$BIND_ABS_OVERWRITE_IMMEDIATE" = Undefined
  "RDM$BIND_ABS_QUIET_POINT" = Undefined
  "RDM$BIND_ABS_PRIORITY" = Undefined
  "RDM$BIND_ABW_ENABLED" = Undefined
  "RDM$BIND_AIJ_ARB_COUNT" = Undefined
   .
   .
   .
```

1.67.7 RMU Show Optimizer_Statistics Command

Displays the current values of the optimizer statistics for tables and indexes as stored in the RDB$INDICES, RDB$RELATIONS, and the RDB$WORKLOAD system table.

Format

RMU/Show Optimizer_Statistics root-file

Command Qualifiers	Defaults
/[No]Full	/Nofull
/[No]Indexes[=(index-list)]	/Index
/[No]Log[=file-name]	/Log
/Statistics[=(options)]	/Statistics
/[No]System_Relations	/Nosystem_Relations
/[No]Tables[=(table-list)]	/Tables
/[No]Threshold[=options]	/Nothreshold

Command Parameters

root-file-spec
Specifies the database for which optimizer statistics are to be displayed. The default file type is .rdb.

Command Qualifiers

Full
Nofull
This qualifier can only be used if table, index, or index prefix cardinality statistics are being displayed. If this qualifier is specified, the following cardinality information is displayed:

- Actual cardinality

  Displays the current table, index, or index prefix cardinality value.

- Stored cardinality

  Displays the table, index, or index prefix cardinality value stored in the system relations.

- Difference between the stored and actual cardinality values

This value is negative if the stored cardinality is less than the actual cardinality.

- Percentage cardinality difference from the actual value

  This value is calculated by dividing the difference between the stored and actual cardinality values by the actual cardinality value. It is negative if the stored cardinality is less than the actual cardinality.

The default value is Nofull.

Indexes
Indexes[=(index-list)]
Noindex
Specifies the index or indexes for which statistics are to be displayed. If you do not specify an index-list, statistics for all indexes defined for the tables specified with the Tables qualifier are displayed. If you specify an index-list, statistics are displayed only for the named indexes. If you specify the Noindex qualifier, statistics are not displayed for any indexes.

The default is the Indexes qualifier without an index-list.

Log
Log=file-name
Nolog
Specifies whether the display of statistics are to be logged. Specify the Log qualifier to have the information displayed to SYS$OUTPUT. Specify the Log=file-spec qualifier to have the information written to a file. The Nolog qualifier is valid syntax, but is ignored by Oracle RMU. The default is the Log qualifier.

Statistics[=(options)]
Specifies the type of statistics you want to display for the items specified with the Tables, System_Relations, and Indexes qualifiers. If you specify the Statistics qualifier without an options list, all statistics are displayed for the items specified.

If you specify the Statistics qualifier with an options list, Oracle RMU displays the types of statistics described in the following list. If you specify more than one option, separate the options with commas and enclose the options within parentheses.

The Statistics qualifier options are:

- Cardinality

Displays the table cardinality for the tables specified with the Tables and System_Relations qualifiers and the index and index prefix cardinalities for the indexes specified with the Indexes qualifier.

- Workload

  Displays the Column Group, Duplicity Factor, and Null Factor workload statistics for the tables specified with the Tables and System_Relations qualifiers.

- Storage

  Displays the following statistics:

  – Table Row Clustering Factor for the tables specified with the Tables qualifier

  – Index Key Clustering Factor, the Index Data Clustering Factor, and the Average Index Depth for the indexes specified with the Indexes qualifier.

System_Relations
Nosystem_Relations

The System_Relations qualifier specifies that optimizer statistics are to be displayed for system tables (relations) and their associated indexes.

If you do not specify the System_Relations qualifier, or if you specify the Nosystem_Relations qualifier, optimizer statistics are not displayed for system tables or their associated indexes.

Specify the Noindex qualifier if you do not want statistics displayed for indexes defined on the system tables.

The default is the Nosystem_Relations qualifier.

Tables
Tables=(table-list)
Notables

Specifies the table or tables for which optimizer statistics are to be displayed. If you specify a table-list, optimizer statistics for those tables and their associated indexes are displayed.

If you do not specify the Tables qualifier, or if you specify the Tables qualifier but do not provide a table-list, optimizer statistics for all tables and their associated indexes in the database are displayed.

If you specify the Notables qualifier, optimizer statistics for tables are not displayed.

Specify the Noindex qualifier if you do not want statistics displayed for indexes defined on the specified tables.

The Tables qualifier is the default.

Threshold=options
Nothreshold

The Threshold qualifier can only be used in conjunction with the Full qualifier. If this qualifier is used, an additional Threshold column is added to the display. You can specify the following options with the Threshold qualifier:

- Percent=n

  The value for Percent=*n* can be an integer value from 0 to 99. The default value for *n* is 0. If Percent=n is not specified or if a percent value of 0 is specified, any percentage difference from the actual cardinality value is flagged as "*over*" in the output column. If a percent value of 1 to 99 is specified, any percentage difference from the actual cardinality value that is greater than the percent value specified is flagged as "*over*" in the output column. In the report, the Threshold column displays those cardinality values in which the percent difference exceeds the specified value. If the threshold is not exceeded, the column is blank. If the threshold is exceeded, the column shows the string "*over*".

- Log={All | Over_Threshold}

  If Log is not specified or if Log=All is specified, all cardinality values are displayed. If Log=Over_Threshold is specified, only cardinality values that exceed the threshold percentage are flagged as "*over*" in the output column.

Usage Notes

- To use the RMU Show Optimizer_Statistics command for a database, you must have the RMU$ANALYZE or RMU$SHOW privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

  You must also have the SQL SELECT privilege on the database.

- Cardinality statistics are automatically maintained by Oracle Rdb. Physical storage and workload statistics are only collected when you issue an RMU Collect Optimizer_Statistics command. To get information about the usage of physical storage and workload statistics for a given

query, define the RDMS$SET_FLAGS logical name to be "ESTIMATES". For example:

```
$ DEFINE RDMS$SET_FLAGS "ESTIMATES"
```

When you execute a query, if workload and physical statistics have been used in optimizing the query, you will see a line such as the following in the command output:

```
~O: Workload and Physical statistics used
```

- Use the RMU Show Optimizer Statistics command with the Statistics=Cardinality/Full/Threshold=n qualifier to identify index prefix cardinality drift. This command identifies indexes that need to be repaired. Use the RMU Collect Optimizer_ Statistics command to repair the stored index prefix cardinality values.

Examples

Example 1

The following command displays all optimizer statistics previously collected for the EMPLOYEES table. See Section 1.15 for an example that demonstrates how to collect optimizer statistics.

```
$ RMU/SHOW OPTIMIZER_STATISTICS MF_PERSONNEL.RDB /TABLE=(EMPLOYEES)

------------------------------------------------------------------

Optimizer Statistics for table : EMPLOYEES

  Cardinality            : 100
  Row clustering factor  : 0.5100000

  Workload Column group  :      EMPLOYEE_ID
  Duplicity factor       : 1.0000000
  Null factor            : 0.0000000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:46:10.73

  Workload Column group  : LAST_NAME,   FIRST_NAME,  MIDDLE_INITIAL,
ADDRESS_DATA_1, ADDRESS_DATA_2, CITY,   STATE,  POSTAL_CODE,    SEX,
BIRTHDAY,       STATUS_CODE
  Duplicity factor       : 1.5625000
  Null factor            : 0.3600000
  First created time     :  3-JUL-1996 10:37:36.43
  Last collected time    :  3-JUL-1996 10:46:10.74
```

```
Index name : EMP_LAST_NAME
  Index Cardinality      : 83
  Average Depth          : 2.0000000
  Key clustering factor  : 0.0481928
  Data clustering factor : 1.1686747
  Segment Column                 Prefix cardinality
    LAST_NAME                       0

Index name : EMP_EMPLOYEE_ID
  Index Cardinality      : 0
  Average Depth          : 2.0000000
  Key clustering factor  : 0.0100000
  Data clustering factor : 0.9500000
  Segment Column                 Prefix cardinality
    EMPLOYEE_ID                     0

Index name : EMPLOYEES_HASH
  Index Cardinality      : 0
  Key clustering factor  : 1.0000000
  Data clustering factor : 1.0000000
```

Example 2

The following command displays optimizer statistics for all the tables defined in the database. Because the Noindex qualifier is specified, no index statistics are displayed. Because the Log qualifier is specified with a file specification, the values for the optimizer statistics are written to the specified file.

```
$ RMU/SHOW OPTIMIZER_STATISTICS mf_personnel.rdb -
_$ /NOINDEX/LOG=NOINDEX-STAT.LOG
```

Example 3

The following example displays the output of a command when the Full and Threshold qualifiers are used with the Cardinality option. In the example, table XXX has three indexes. Index XXX_IDX_FULL has index prefix cardinality collection enabled full and the report shows no cardinality drift for this index. Index XXX_IDX_APPROX has index prefix cardinality collection enabled, and cardinality drift is evident. For the first segment of the index (column C1), the stored cardinality is 20% lower than the actual cardinality. Since the command specifies a threshold of 5%, the line is marked "*over*" in the Thresh column. There is also cardinality drift for the second segment of the index (column C2), index prefix (C1, C2). The third index XXX_IDX_NONE has index prefix cardinality collection disabled. This is indicated in the report rather than showing the index segments. If the report were lengthy, you could write it to a disk file and then locate the problem indexes by searching for the string "*over*".

```
$ RMU/SHOW OPTIMIZER/STAT=CARD/FULL/THRESH=(percent=5,log=all) sample.rdb

Optimizer Statistics for table : XXX

(Cardinality: Diff=Stored-Actual, Percent=Diff/Actual, Thresh=Percent exceeded)
                                Table cardinality
    Actual               Stored                Diff             Percent Thresh
    109586               109586                0                0  %

Index name : XXX_IDX_FULL
(Cardinality: Diff=Stored-Actual, Percent=Diff/Actual, Thresh=Percent exceeded)
                                Index cardinality
    Actual               Stored                Diff             Percent Thresh
    109586               109586                0                0  %
                                Prefix cardinality
    Actual               Stored                Diff             Percent Thresh
  Segment Column : C1
    1425                 1425                  0                0  %
  Segment Column : C2
    31797                31797                 0                0  %
 Segment Column : C3
    0                    0                     0                0  %

Index name : XXX_IDX_APPROX
(Cardinality: Diff=Stored-Actual, Percent=Diff/Actual, Thresh=Percent exceeded)
                                Index cardinality
    Actual               Stored                Diff             Percent Thresh
    109586               109586                0                0  %
                                Prefix cardinality
    Actual               Stored                Diff             Percent Thresh
  Segment Column : C1
    1425                 1140                 -285             -20 %    *over*
  Segment Column : C2
    31797                30526                -1271            -4  %
  Segment Column : C3
    0                    0                     0                0  %

Index name : XXX_IDX_NONE
(Cardinality: Diff=Stored-Actual, Percent=Diff/Actual, Thresh=Percent exceeded)
                                Index cardinality
    Actual               Stored                Diff             Percent Thresh
    109586               109586                0                0  %
  ***Prefix cardinality collection is disabled***
```

1.67.8 RMU Show Privilege Command

Allows you to display the root file access control list (ACL) for a database.

Format

RMU/Show Privilege root-file-spec

Command Qualifiers	Defaults
[No]Expand_All	/Noexpand_All
[No]Header	/Header

Command Parameters

root-file-spec
The root file specification for the database whose root file ACL you are displaying. By default, a file extension of .rdb is assumed.

Command Qualifiers

Expand_All
Noexpand_All
Specifies that if a user's access mask was defined with the RMU$ALL keyword on the RMU Set Privilege command, each of the RMU privileges represented by the RMU$ALL keyword is displayed.

The Noexpand_All qualifier specifies that if a user's access mask was defined with the RMU$ALL keyword on the RMU Set Privilege command, only the keyword is displayed; the RMU privileges represented by the keyword are not displayed.

The Noexpand_All qualifier is the default.

Header
Noheader
Specifies that header information is to be displayed. The Noheader qualifier suppresses output of header information.

The Header qualifier is the default.

Usage Notes

- To use the RMU Show Privilege command for a database, you must have the RMU$SECURITY privilege in the root file ACL for the database or the OpenVMS SECURITY or BYPASS privilege.
- Although you can use the DCL SHOW ACL command to display the root file ACL for a database, the DCL SHOW ACL command does not display the names of the Oracle RMU privileges granted to users.

Examples

Example 1

In the following example, the RMU Show Privilege command displays the root file ACL for the mf_personnel database:

```
$ RMU/SHOW PRIVILEGE MF_PERSONNEL.RDB
Object type: file,  Object name: SQL_USER:[USER1]MF_PERSONNEL.RDB;1,
 on 12-FEB-1996 10:48:23.04

 (IDENTIFIER=[SQL,USER1],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
 RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
 RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
 RMU$VERIFY)
 (IDENTIFIER=[SQL,USER2],ACCESS=READ+WRITE+RMU$ALTER+RMU$ANALYZE+
 RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+RMU$MOVE+RMU$OPEN+
 RMU$RESTORE+RMU$SHOW+RMU$UNLOAD+RMU$VERIFY)
 (IDENTIFIER=[SQL,USER3],ACCESS=READ+WRITE+CONTROL+RMU$SECURITY)
```

Example 2

The following examples demonstrate the difference in output when you use the Header and Noheader qualifiers:

```
$ RMU/SHOW PRIV MF_PERSONNEL.RDB/HEADER
Object type: file,  Object name: RDBVMS_USER:[DB]MF_PERSONNEL.RDB;1,
 on 17-SEP-1998 13:47:20.21
          (IDENTIFIER=[RDB,STONE],ACCESS=RMU$ALL)

$ RMU/SHOW PRIVILEGE MF_PERSONNEL.RDB/NOHEADER
          (IDENTIFIER=[RDB,STONE],ACCESS=RMU$ALL)
```

Example 3

The following examples demonstrate the difference in output when you use the Expand and Noexpand qualifiers:

```
$ RMU/SET PRIVILEGE MF_PERSONNEL.RDB /ACL=(I=STONE,A=RMU$ALL)

$ RMU/SHOW PRIVILEGE MF_PERSONNEL.RDB /NOEXPAND/NOHEADER
(IDENTIFIER=[RDB,STONE],ACCESS=READ+WRITE+CONTROL+RMU$ALL)

$ RMU/SHOW PRIVILEGE MF_PERSONNEL.RDB /EXPAND/NOHEADER
(IDENTIFIER=[RDB,STONE],ACCESS=READ+WRITE+CONTROL+RMU$ALTER+
      RMU$ANALYZE+RMU$BACKUP+RMU$CONVERT+RMU$COPY+RMU$DUMP+RMU$LOAD+
      RMU$MOVE+RMU$OPEN+RMU$RESTORE+RMU$SECURITY+RMU$SHOW+RMU$UNLOAD+
      RMU$VERIFY)
```

1.67.9 RMU Show Statistics Command

Opens the Performance Monitor to display, on a character-cell terminal, the usage statistics for a database. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for tutorial information on how to interpret the Performance Monitor displays.

Format

RMU/Show Statistics [root-file-spec]

Command Qualifiers	Defaults
/Access_Log	None
/Alarm=interval	/Alarm=0
/[No]Broadcast	See description
/[No]Cluster=[(node-list)]	/Nocluster
/Configure=file-spec	None
/[No]Cycle=seconds	/Nocycle
/Dbkey_Log=file-spec	See description
/Deadlock_Log=file-spec	None
/[No]Flush_Interval=seconds	/Flush_Interval=60
/[No]Histogram	/Histogram
/Hot_Standby_Log	None
/Input = file-name	See description
/[No]Interactive	See description
/Lock_Timeout_Log=file-spec	None
/[No]Log	See description
/[No]Logical_Area	/Logical_Area
/[No]Notify[=([No]All | operator-classes)]	/Nonotify
/[No]Opcom_Log=filename	/Noopcom_Log
/Options=keywords	/Options=Base
/Output=file-spec	See description
/[No]Prompt_Timeout=seconds	/Prompt_Timeout=60
/Reopen_Interval= minutes	None
/Reset	Statistics are no
/Screen = screen-name	See description
/Stall_Log = file-spec	Stall messages no
/Time = integer	/Time = 3
/Until = date-time	See description
/[No]Write_Report_Delay = integer	See description

Description

The Performance Monitor dynamically samples activity statistics on a database. You can display the statistics at your terminal and can also write them to a formatted binary file.

The statistics show activity only from the node on which you execute the command.

The Performance Monitor operates in one of three modes: online, record, and replay. In online mode, you can display or record current activity on a database. In record mode, you can record statistics in a binary file. In replay mode, you can examine a previously recorded binary statistics file.

If you use the Input qualifier, the Performance Monitor executes in replay mode. In replay mode, this command generates an interactive display from a previously recorded binary statistics file.

If you do not use the Input qualifier, you must specify a database file name. The Performance Monitor then executes in online mode. In online mode, the command generates an interactive display when you use the Interactive qualifier and can also record statistics in a binary file.

The interactive display is made up of numerous output pages. You control the interactive display by means of menus, arrow keys, and the Return key to select options. You select an item by pressing the arrow keys until the desired item is highlighted, then press the Return key.

Display the Select Display options (by typing D) from the Performance Monitor screen to view the available output pages. Items in the Display menu followed by this set of characters: [->, indicate that a submenu is displayed when you select this item.

Once you have selected a display, there are a number of methods you can use to navigate through the screens:

- To move to the next screen of information, do one of the following:
  - Press the right arrow (→) keyboard key.
  - Press the Next Screen keyboard key.
- To move to the previous screen of information, do one of the following:
  - Press the left arrow (←) keyboard key.
  - Press the Prev Screen keyboard key.
- To move forward *n* number of screens, press the plus (+) keyboard key and enter the value *n*.

- To move backward *n* number of screens, press the minus (-) keyboard key and enter the value *n*.
- To move directly from the first screen to the last screen, do one of the following:
  - Press the up arrow (↑) keyboard key.
  - Press the plus (+) keyboard key and enter the value 0.
- To move directly from the last screen to the first screen, do one of the following:
  - Press the down arrow (↓) keyboard key.
  - Press the hyphen (-) keyboard key and enter the value 0.
- To quickly locate a screen in the current submenu group that contains activity, press the space bar on your keyboard.

  This feature works even when you are replaying a binary input file. If there is no screen in the current subgroup that has activity, the next screen is displayed (as though you had used the Next Screen key). The Performance Monitor ignores computational screens, such as Stall Messages, Monitor Log, and so on, when searching for activity.

In interactive mode, enter an exclamation point to open the Select Tool menu. This menu allows you to switch the database for which you are displaying statistics, edit a file, invoke a system command, and so on. (The ability to open a new database is not available if you specify the Input or Output qualifier.) In addition, it provides you the ability to locate a specific statistics screen either by name (or portion thereof) or by a summary-selection menu. Select the Goto screen or Goto screen "by-name" options from the Select Tool menu to use these options.

In interactive mode, you can pause output scrolling on your screen by pressing the P key. Resume output scrolling by pressing the P key again.

An extensive online help facility for the character-cell interface is available by doing the following from the Performance Monitor screen:

1. Type H or PF2.
2. Select the type of help you want (keyboard, screen, or field).
3. Press the Return key.

If you select field level help, you must also do the following:

1. Highlight the field for which you want help information.

2. Press the Return key.

All screens regardless of format or display contents have a standard format as follows:

- First line

  Contains the node name, the utility name and version number, and the current system date and time. The current system date and time are updated at the specified set-rate interval.

- Second line

  Contains the screen refresh rate, in seconds; the current screen name; and the elapsed time since the last set-rate command, which indicates how long the screen information has been collected.

- Third line

  Contains the current page number within the screen (screen *X* of *Y*), the name of the current database, and the statistics utility operation mode (online, record, or replay). Online mode is the normal database activity displayed in real time. Record mode indicates that the database activity being displayed is being recorded to an external file specified by the Output qualifier. Replay mode indicates that the database activity is being displayed from the external file specified by the Input qualifier.

You can display most statistics in either a histogram or a columnar chart, although several display pages have special formats. By default, the initial interactive display appears in histogram mode; by using the Nohistogram qualifier, you can direct Oracle RMU to display statistics in tabular numeric mode.

In addition, you can produce time-plot graphics for individual statistical fields.

Use the Output qualifier to direct statistical output to a file. The output is a formatted binary file and does not produce a legible printed listing. To read the output, you must use the RMU Show Statistics command with the Input qualifier.

The Nointeractive qualifier suppresses the interactive display. Use this qualifier when you want to generate binary statistics output but do not want an online display.

Database statistics are maintained in a global section on each system on which Oracle Rdb is running. Statistics are reset to zero when you close a database. Running the Performance Monitor keeps the database open even when there are no users accessing the database.

The Stall Messages display permits you to display multiple screens of information. Access the Stall Messages display by selecting Per-Process Information from the Select Display Menu; then select the Stall Messages display from the secondary menu.

If you are displaying the last screen of Stall Messages information and the number of stalled processes is reduced such that the last screen is empty, you are automatically moved to the newest last screen of information when you press the Next Screen keyboard key (or the right arrow keyboard key).

You can also use the Alarm, Notify, and Screen qualifiers to simplify monitoring stalled processes. See the description of each of these qualifiers for more information.

Command Parameters

root-file-spec
The root file specification of the database on which you want statistics. If you use the Input qualifier to supply a prerecorded binary statistics file, you cannot specify a database file name. If you do not use the Input qualifier, you must specify a database file name.

Command Qualifiers

Access_Log
Identifies the name of the log file where logical area accesses are to be recorded.

Alarm=interval
Establishes an alarm interval (in seconds) for the Stall Messages screen from the command line. This is useful when you plan to submit the RMU Show Statistics command as a batch job.

Use this qualifier in conjunction with the Notify qualifier to notify an operator or set of operators of stalled processes.

The default value is 0 seconds, which is equivalent to disabling notification.

Broadcast
Nobroadcast
Specifies whether or not to broadcast messages. The Broadcast qualifier is the default, if broadcasting of certain messages has been enabled with DCL SET BROADCAST. If broadcasting has been disabled with the DCL SET BROADCAST=none command, broadcast messages are not displayed, even if you specify the RMU Show Statistics command with the Broadcast qualifier.

Specify the Nobroadcast qualifier if broadcasting has been enabled with the DCL SET BROADCAST command but you do not want broadcast messages displayed while you are running the Performance Monitor.

Cluster=(node-list)
Nocluster
Specifies the list of remote nodes from which statistics collection and presentation are to be performed. The collected statistics are merged with the information for the current node and displayed using the usual statistics screens.

The following list summarizes usage of the Cluster qualifier:

- If the Cluster qualifier is specified by itself, remote statistics collection is performed on all cluster nodes on which the database is currently open.
- If the Cluster=(node-list) qualifier is specified, remote statistics collection is performed on the specified nodes only, even if the database is not yet open on those nodes.
- If the Cluster qualifier is not specified, or the Nocluster qualifier (the default) is specified, cluster statistics collection is not performed. However, you can still enable clusterwide statistics collection online using the Tools menu.

You can specify up to 95 different cluster nodes with the Cluster qualifier. There is a maximum number of 95 cluster nodes because Oracle Rdb supports only 96 nodes per database. The current node is always included in the list of nodes from which statistics collection is to be performed.

It is not necessary to have the RMU Show Statistics command running on the specified remote nodes or to have the database open on the remote nodes. These events are automatically handled by the feature.

The following example shows the use of the Cluster qualifier to initiate statistics collection and presentation from two remote nodes:

```
$ RMU /SHOW STATISTICS /CLUSTER=(BONZAI, ALPHA4) MF_PERSONNEL
```

Remote nodes can also be added and removed online at run time. Use the Cluster Statistics option located in the Tools menu. The Tools menu is displayed by using the exclamation point (!) on-screen menu option.

See the *RMU Show Statistic DBA Handbook* (available in MetaLink if you have a service contract) for information about the Cluster Statistics Collection and Presentation feature.

Configure=file-spec
Specifies the name of a human-readable configuration file to be processed by the RMU Show Statistics command. The configuration file can be created using any editor, or it can be automatically generated from the RMU Show Statistics command using the current run-time configuration settings. The default configuration file type is .cfg.

If you specify the Configure=file-spec qualifier, the configuration file is processed by the RMU Show Statistics command prior to opening the database or the binary input file. If you do not specify this qualifier, all of the variables are the defaults based on command-line qualifiers and logical names.

The configuration file is processed in two passes. The first pass occurs before the database is opened and processes most of the configuration file entries. The second pass occurs after the database is opened and processes those variables that are database-dependent, such as the CUSTOMER_LINE_n variable.

See the *RMU Show Statistic DBA Handbook* (available in MetaLink if you have a service contract) for more information about configuration files.

Cycle=seconds
Nocycle
Directs the Performance Monitor to continually cycle through the set of screens associated with the currently selected menu item. Each menu is displayed for the number of seconds specified.

When you specify the Cycle qualifier, you can change screen modes or change submenus as desired; cycling through the menus associated with your choice continues at whichever menu level is currently selected.

The specified value for the Cycle qualifier must be greater than or equal to the value specified for the Time qualifier. In addition, if you manually change the refresh rate (using the Set_rate onscreen menu option) to a value that is greater than the value you specify with the Cycle qualifier, the cycling is performed at the interval you specify for the Set_rate.

If you do not specify the Cycle qualifier, or if you do not specify the number of seconds, no screen cycling is performed.

Dbkey_Log=file-spec
Logs the records accessed during a given processing period by the various attached processes. The file-spec is the name of the file to which all accessed dbkeys are logged.

The header region of the dbkey log contains four lines. The first line indicates that the RMU Show Statistic utility created the log file. The second line identifies the database. The third line identifies the date and time the dbkey log was created. The fourth line is the column heading line.

The main body of the dbkey log contains six columns. The first column contains the dbkey process ID and stream ID. The second through sixth columns contain the most recently accessed dbkey for a data page, snapshot page, SPAM page, AIP page, and ABM page, respectively.

Only one message per newly accessed dbkey is recorded. However, all dbkey values are displayed, even if some of the dbkeys did not change.

The dbkey information is written at the current screen refresh rate, determined by the Time qualifier or the Set_rate onscreen menu option. Using a larger refresh rate minimizes the size of the file but results in a large number of missed dbkey messages. Using a smaller refresh rate produces a large log file, but contains a much finer granularity of dbkey messages.

Note that you do not need to display the Dbkey Information screen in order to record the dbkey messages to the dbkey log. The dbkey log is maintained regardless of which screen, if any, is displayed.

You can use the Dbkey_Log qualifier to construct a dbkey logging server, as follows:

```
$ RMU/SHOW STATISTICS/NOHISTOGRAM/TIME=1 -
_$ /NOINTERACTIVE/DBKEY_LOG=DBKEY.LOG MF_PERSONNEL -
_$ /NOBROADCAST/UNTIL="15:15:00"
```

Deadlock_Log=file-spec

Records the last deadlock for the processes. There is no method to record each lock deadlock as it occurs.

The file-spec in the qualifier is the name of the file to which you want all lock deadlock messages to be logged. The lock deadlock messages are written in human-readable format similar to the Lock Timeout History and Lock Deadlock History screens.

The header region of the lock deadlock log contains three lines:

- Line 1 indicates that the RMU Show Statistics utility created the log file.
- Line 2 identifies the database.
- Line 3 identifies the date and time the log was created.

The main body of the stall log contains three columns:

- The first column contains the process ID and stream ID that experienced the lock deadlock.
- The second column contains the time the deadlock occurred; however, the date is not displayed.
- The third column contains the deadlock message describing the affected resource. This message is similar to the originating stall message.

For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR)
```

If any lock deadlocks are missed for a particular process (usually because the recording interval is too large), the number of missed lock deadlocks is displayed in brackets after the message. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR) [1 missed]
```

Only one message is logged for each deadlock.

The lock deadlock messages are written at the specified screen refresh rate, determined by specifying the Time qualifier, or online using the Set_rate on-screen menu option. Using a larger refresh rate minimizes the size of the file, but results in a large number of missed deadlock messages. Using a smaller refresh rate produces a large log file, but contains a much finer granularity of deadlock messages.

Using the Time=1 or Time=50 qualifier produces a reasonable log while minimizing the impact on the system.

The affected LockID is not displayed, because this is meaningless information after the lock deadlock has completed.

Use the Tools menu (displayed when you press the exclamation point (!) key from any screen) to enable or disable the lock timeout and lock deadlock logging facility while the RMU Show Statistics utility is running. However, note that the lock timeout log and lock deadlock log are not available during binary file replay.

Flush_Interval=seconds
Noflush_Interval
Allows you to specify a flush interval, in seconds. The default value is 60 seconds if no Flush_Interval is specified. The minimum flush interval that can be specified is 0 seconds. The maximum flush interval that can be specified is 3600 seconds (1 hour). Specifying 0 seconds for the flush interval is equivalent to specifying /NOFLUSH_INTERVAL. If the flush interval is active and less

than the statistics collection interval, to avoid unnecessary buffer flushes, the flush interval will be set to the statistics collection interval, which has a default of 3 seconds and can be set by the existing /TIME qualifier, or by typing an "S" if "Set rate" is displayed at the bottom of the statistics screen. The currrent collection interval is displayed following "Rate:" in the statistics screen header.

The Flush_Interval is important because it insures that, if a system failure occurs, important data is not lost from the RMU/SHOW STATISTICS output files (the binary file used to record Oracle Rdb database statistics for later replay, the logical area access log file, the record access dbkey log file, the process deadlock log file, the lock timeout log file, the OPCOM messages log file, the Hot Standby log file, the online analysis log file, and the stall messages log file). These periodic buffer data flushes initiated by the /Flush_Interval qualifier will help to minimize this loss of diagnostic data.

The first and second command examples below use the default flush interval of 60 seconds. The third and fourth command examples below specify a flush interval of 10 seconds. Note that the flush interval can be set even if the RMU/SHOW STATISTICS command does not specify any log or output files. This is because the TOOLS menu can be used later in the RMU/SHOW STATISTICS session to create log files for which the flush interval of 60 or 10 will be used.

```
$ RMU/SHOW STATISTICS MF_PERSONNEL
$ RMU/SHOW STATISTICS/OUTPUT=SHOWSTAT.DAT/DBKEY_LOG=D.LOG MF_PERSONNEL
$ RMU/SHOW STATISTICS/FLUSH_INTERVAL=10 MF_PERSONNEL
$ RMU/SHOW STATISTICS/FLUSH_INTERVAL=10/OUTPUT=SHOWSTAT.DAT/DBKEY_LOG=D.LOG
MF_PERSONNEL
```

The following command example specifies a statistics collection interval of 12 seconds using the /TIME command qualifier. This is larger than the 10 seconds specified by the new /FLUSH_INTERVAL qualifier. Since the collection interval is larger than the flush interval, the collection interval will be used for the flush interval to avoid excess buffer flushes.

```
$ RMU/SHOW STATISTICS/FLUSH_INTERVAL=10/TIME=12/OUTPUT=SHOWSTAT.DAT
/DBKEY_LOG=D.LOG MF_PERSONNEL
```

The following command examples are equivalent and specify that periodic buffer flushes will not be used for the RMU/SHOW STATISTICS binary output and log files.

```
$ RMU/SHOW STATISTICS/NOFLUSH_INTERVAL/OUTPUT=SHOWSTAT.DAT/DBKEY_LOG=D.LOG
MF_PERSONNEL
$ RMU/SHOW STATISTICS/FLUSH_INTERVAL=0/OUTPUT=SHOWSTAT.DAT/DBKEY_LOG=D.LOG
MF_PERSONNEL
```

Histogram
Nohistogram
Directs Oracle RMU to display the initial statistics screen in the numbers display mode or the graph display mode. The Histogram qualifier specifies the graph display mode. The Nohistogram qualifier specifies the numbers display mode.

The Histogram qualifier is the default.

Hot_Standby_Log
Specifies the name of the Hot Standby log file. The "Start hot standby logging" option of the Tools menu (enter !) can be used to specify the name of the Hot Standby log file at runtime.

Input=file-name
Specifies the prerecorded binary file from which you can read the statistics. This file must have been created by an earlier RMU Show Statistics session that specified the Output qualifier.

You cannot specify a database file name with the Input qualifier. Also, you must not use the Until, Output, or Nointeractive qualifiers with the Input qualifier. However, you can use the Time qualifier to change the rate of the display. This will not change the computed times as recorded in the original session. For example, you can record a session at Time=60. This session will gather statistics once per minute.

You can replay statistics gathered in a file by using the Input and Time qualifiers. To replay a file:

- Use the Output qualifier to create a file of database statistics.
- Use the Input and Time qualifiers to view the statistics again at a rate that you determine. For example, the command RMU/SHOW STATISTICS PERS.LOG/TIME=1, will replay the PERS.LOG file and change the display once per second, thus replaying 10 hours of statistics in 10 minutes.

If you do not specify the Input qualifier, you must specify the root-file-spec parameter.

Interactive
Nointeractive
Displays the statistics dynamically to your terminal. The Interactive qualifier is the default when you execute the RMU Show Statistics command from a terminal. You can use the Nointeractive qualifier with the Output qualifier to generate a binary statistics file without generating a terminal display.

The Nointeractive qualifier is the default when you execute the RMU Show Statistics command from a batch job.

In an interactive session, you can use either the menu interface or the predefined control characters to select display options (see the Performance Monitor online help for further information about the predefined control characters).

Select menu options by using the up (↑) and down (↓) arrow keys followed by pressing the Return or Enter key. Cancel the menu by pressing Ctrl/Z.

Lock_Timeout_Log=file-spec
Records the last lock timeout message for the processes. There is no method to record each lock timeout as it occurs. The lock timeout messages are written in human-readable format.

The header region of the lock timeout log contains three lines:

- Line 1 indicates that the RMU Show Statistics utility created the log file.
- Line 2 identifies the database.
- Line 3 identifies the date and time the log was created.

The main body of the stall log contains three columns:

- The first column contains the process ID and stream ID that experienced the lock timeout.
- The second column contains the time the timeout occurred; however, the date is not displayed.
- The third column contains the timeout message describing the affected resource. This message is similar to the originating stall message.

For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR)
```

If any lock timeouts are missed for a particular process (usually because the recording interval is too large), the number of missed lock timeouts is displayed in brackets after the message. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR) [1 missed]
```

Only one message is logged for each lock timeout.

The lock timeout messages are written at the specified screen refresh rate, determined by specifying the Time qualifier, or online using the Set_rate on-screen menu option. Using a larger refresh rate minimizes the size of the file, but results in a large number of missed lock timeout messages. Using

a smaller refresh rate produces a large log file, but contains a much finer granularity of lock timeout messages.

Using the Time=1 or Time=50 qualifier appears to produce a reasonable log while minimizing the impact on the system.

The affected LockID is not displayed because this is meaningless information after the lock timeout has completed.

Note that you do not need to be displaying the Lock Timeout History or Lock Deadlock History screens to record the stall messages to the stall log. These logs are maintained regardless of which screen, if any, is displayed.

Use the Tools menu (displayed when you press the exclamation point (!) key from any screen) to enable or disable the lock timeout and lock deadlock logging facility while the RMU Show Statistics utility is running. However, note that the lock timeout log and lock deadlock log are not available during binary file replay.

Log
Nolog
Logs the creation of a binary statistics file to your output file. This binary statistics file is created only if you have used the Output qualifier. If you use the Nolog qualifier, no operations will be logged to your output file.

The default is the current setting of the DCL verify switch. See HELP SET VERIFY in DCL HELP for more information on changing the DCL verify switch.

If you use the Interactive qualifier, the Log qualifier is ignored.

Logical_Area
Nological_Area
Specifies that you want the RMU Show Statistics command to acquire the needed amounts of virtual memory to display logical area statistics information. The Logical_Area qualifier is the default.

By default, the RMU Show Statistics command consumes approximately 13,000 bytes of virtual memory per logical area. (The number of logical areas is determined by the largest logical area identifier — not by the actual number of areas.) This can result in the RMU Show Statistics command consuming large amounts of virtual memory, even if you do not want to review logical area statistics information.

Use the NoLogical_Area qualifier to indicate that you do not want to display logical area statistics information. When you specify the NoLogical_Area qualifier, the virtual memory for logical area statistics information presentation is not acquired.

When you specify the NoLogical_Area qualifier, do not also specify the Nolog qualifier, as this causes logical area statistics information to still be collected.

The "Logical Area" statistics are *not* written to the binary output file. Conversely, the "Logical Area" statistics screens are not available during binary input file replay.

There is no corresponding configuration variable. This qualifier cannot be modified at run time. See the *RMU Show Statistic DBA Handbook* (available in MetaLink if you have a service contract) for more information about interpreting logical area screens.

Notify
Notify=[No]All
Notify=operator-classes
Nonotify

Notifies the specified system operator or operators when a stall process exceeds the specified alarm interval by issuing a broadcast message and ringing a bell at the terminal receiving the message.

The valid operator classes are: CENTRAL, CLUSTER, DISKS, OPCOM, SECURITY, and OPER1 through OPER12.

The various forms of the Notify qualifier have the following effects:

- If you specify the Notify qualifier without the operator-classes parameter, the CENTRAL and CLUSTER operators are notified by default.
- If you specify the Nonotify or Notify=Noall qualifiers, operator notification is disabled.
- If you specify the Notify=All qualifier, all operator classes are enabled.
- If you specify the Notify=operator-classes qualifier, the specified classes are enabled. (If you specify more than one operator class, enclose the list in parentheses and separate each class name with a comma.)

  For example, issuing the RMU Show Statistics command with the Notify=(OPER1, OPER2) qualifier sends a notification message to system operator classes OPER1 and OPER2 if the Alarm threshold is exceeded while monitoring the Stall Messages screen.

- When the Notify=OPCOM qualifier is specified with the RMU Show Statistics command along with the Alarm and Cluster qualifiers, Oracle RMU generates an OPCOM message and delivers it to the OPCOM class associated with the Notify qualifier. This message alerts the operator to the fact that the process has stalled for more than *n* seconds, where *n* is the value assigned to the Alarm qualifier. The process that has stalled may be on any node that is included in the node name list assigned to the Cluster qualifier.

The specified system operator(s) are notified only when the alarm threshold is first exceeded. For instance, if three processes exceed the alarm threshold, the specified operator(s) are notified only once. If another process subsequently exceeds the alarm threshold while the other processes are still displayed, the specified system operator(s) are not notified.

However, if the longest-duration stall is resolved and a new process then becomes the newest stall to exceed the alarm threshold, then the specified system operator(s) will be notified of the new process.

To receive operator notification messages, the following three OpenVMS DCL commands must be issued:

1. $ SET TERM /BROADCAST
2. $ SET BROADCAST=OPCOM
3. $ REPLY /ENABLE=(operator-classes)

The operator-classes specified in the REPLY /ENABLE command must match those specified in the Notify qualifier to the RMU Show Statistics command.

The operator notification message will appear similar to the following sample message:

```
%%%%%%%%%%  OPCOM  19-DEC-1994 08:56:39.27  %%%%%%%%%%
                    (from node MYNODE at 19-DEC-1994 08:56:39.30)
Message from user SMITH on MYNODE
Rdb Database USER2:[SMITH.WORK.AIJ]MF_PERSONNEL.RDB;1 Event Notification
Process 2082005F:1 exceeded 5 second stall: waiting for record 51:60:2  (EX)
```

The system operator notification message contains four lines. Line 1 contains the OPCOM broadcast header message. Line 2 identifies the process running the RMU Show Statistics command that sent the message. Line 3 identifies the database being monitored. Line 4 identifies the process that triggered the alarm, including the alarm interval and the stall message.

To establish an alarm interval for the Stall Messages screen, use the Alarm=Interval qualifier.

If you specify the Nointeractive qualifier, bell notification is disabled, but the broadcast message remains enabled.

Opcom_Log=filename
Noopcom_Log
Specifies the name of the file where OPCOM messages broadcast by attached database processes will be sent.

When recording OPCOM messages, it is possible to occasionally miss a few messages for a specific process. When this occurs, the message "n missed" will be displayed in the log file.

You can record specific operator classes of OPCOM messages if you specify the Option=Verbose qualifier. The Option=Verbose qualifier records only those messages that can be received by the process executing the RMU Show Statistics utility. For example, if the process is enabled to receive operator class Central, then if you specify Opcom_Log=opcom.log the Option=Verbose qualifier records all Central operator messages. Conversely, specifying only the Opcom_Log=opcom.log qualifier records all database-specific OPCOM messages generated from this node. Because the output is captured directly from OpenVMS, the operator-specific log file output format is different from the database-specific contents. The following example shows the operator-specific log file contents for the Cluster and Central operator classes:

```
 Oracle Rdb X7.1-00 Performance Monitor OPCOM Log
 Database KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS.RDB;2
 OPCOM Log created 11-JUN-1999 10:52:07.53
11-JUN-1999 10:52:23.85)  Message from user RDBVMS on ALPHA4  Oracle Rdb X7.1-00
Event Notification for Database  _$111$DUA368:[BBENTON.TEST]MF_PERSONNEL.RDB;1
AIJ Log Server terminated
11-JUN-1999 10:52:25.49)  Message from user RDBVMS on ALPHA4  Oracle Rdb X7.1-00
Event Notification for Database  _$111$DUA368:[BBENTON.TEST]MF_PERSONNEL.RDB;1
AIJ Log Roll-Forward Server started
11-JUN-1999 10:52:26.06)  Message from user RDBVMS on ALPHA4  Oracle Rdb X7.1-00
Event Notification for Database  _$111$DUA368:[BBENTON.TEST]MF_PERSONNEL.RDB;1
AIJ Log Roll-Forward Server failed
   .
   .
   .
11-JUN-1999 10:54:21.09)  Message from user RDBVMS on ALPHA4  Oracle Rdb X7.1-00
Event Notification for Database  _$111$DUA368:[BBENTON.TEST.JUNK]T_
PERSONNEL.RDB;1    AIJ Log Server started
11-JUN-1999 10:54:21.13)  Message from user RDBVMS on ALPHA4  Oracle Rdb X7.1-00
Event Notification for Database  _$111$DUA368:[BBENTON.TEST.JUNK]T_
PERSONNEL.RDB;1    Opening "$111$DUA368:[BBENTON.TEST.JUNK]TEST1.AIJ;2"
```

Options=[keywords]

The following keywords may be used with the Options qualifier:

- [No]All

  Indicates whether or not all collectible statistics (all statistics for all areas) are to be collected. The All option indicates that all statistics information is to be collected; the Noall keyword indicates that only the base statistics information is to be collected. You must also specify the Output qualifier. Note: Logical Area information is not written to the binary output file.

- [No]Area

  Indicates whether or not the by-area statistics information is to be collected in addition to the base statistics information. When you specify the Area or Noarea option, the Base statistics are implicitly selected. You must also specify the Output qualifier.

  When the Area option is specified, statistics for all existing storage areas are written to the binary output file; you cannot selectively choose specific storage areas for which statistic information is to be collected.

  The size of the by-area statistics output largely depends on the total number of storage areas in the database, including reserved storage areas. If the database contains a large number of storage areas, it may not be advisable to use the Options=Area qualifier.

  Before you replay a binary output file that contains by-area statistics, specify the following command to format the display correctly:

  ```
  $ SET TERM/NOTAB
  ```

  You can then replay the statistics as follows:

  ```
  $ RMU/SHOW STATISTICS/INPUT=main.stats
  ```

- Base (default)

  Indicates that only the base set of statistics is to be collected; this is the default Options option. You must also specify the Output qualifier. You cannot specify Nobase.

- Compress

  Compresses the statistics records written to the output file specified by the Output qualifier. While replaying the statistics, the RMU Show Statistics command determines if a record was written using compression or not. If the record was written using compression it is automatically decompressed.

  If compression is used, the resultant binary file can be read only by the RMU Show Statistics command. The format and contents of a compressed file are not documented or accessible to other applications.

- Confirm

  Indicates that you wish to confirm before exiting from the utility. You can also specify the Confirm option in the configuration file using the CONFIRM_EXIT variable. A value of TRUE indicates that you want to confirm before exiting the utility and a value of FALSE (the default) indicates you do not want to confirm before exiting the utility.

- Log_Stall_Alarm

  If Log_Stall_Alarm is present when using the Stall_Log qualifier to write stall messages to a log file and the Alarm qualifier to set an alarm interval, only those stalls exceeding the Alarm specified duration are written to the stall log output file.

- Log_Stall_Lock

  If you use the Stall_Log qualifier to write stall messages to a log file, use the Nolog_Stall_Lock option to prevent lock information from being written to the log file. If you use or omit the Log_Stall_Lock option, lock information is written to the log file.

- [No]Row_Cache

  Indicates that all row cache related screens and features of the RMU Show Statistics facility are to be displayed. NoRow_Cache indicates that these features are disabled.

- Screen_Name

  Allows you to identify a screen capture by screen name. If you issue an RMU Show Statistics command with the Options=Screen_Name qualifier, the screen capture is written to a file that has the name of the screen with all spaces, brackets, and slashes replaced by underscores. The file has an extension of .SCR. For example, if you use the Option=Screen_Name qualifier and select the Write option on the Screen Transaction Duration (Read/Write), the screen is written to a file named TRANSACTION_ DURATION_READ_WRITE.SCR.

- Update

  Allows you to update fields in the Database Dashboard. See the Performance Monitor Help or the *Oracle Rdb7 Guide to Database Performance and Tuning* for information about using and updating the Database Dashboard. You must have both the OpenVMS WORLD and BYPASS privileges to update fields in the Database Dashboard.

- Verbose

Causes the stall message logging facility to report a stall message at each interval, even if the stall message has been previously reported.

Note

Use of the Options=Verbose qualifier can result in an enormous stall messages log file. Ensure that adequate disk space exists for the log file when you use this qualifier.

You can enable or disable the stall messages logging Verbose option at run time by using the Tools menu and pressing the exclamation point (!) key.

You can also specify the Verbose option in the configuration file by using the STALL_LOG_VERBOSE variable. Valid keywords are ENABLED or DISABLED.

Lock information is displayed only once per stall, even in verbose mode, to minimize the output file size.

Output=file-spec
Specifies a binary statistics file into which the statistics are written. Information in the Stall Messages screen is not recorded in this file, however. The information in the Stall Messages screen is highly dynamic and thus cannot be replayed using the Input qualifier.

Note

Statistics from the Stall Messages display are *not* collected in the binary output file.

Each release of Oracle Rdb introduces changes to the output statistics file structures (new fields, changes of data type, removal of obsolete fields, moved fields, and so on). If you are having a problem with an application that accesses the RMU Show Statistics field structures, recompile your application with the latest RMU$SHOW_STATISTICSnn.CDO or RMU$SHOW_STATISTICSnn.H (where nn is the version of Oracle Rdb you are using).

Prompt_Timeout=seconds
Noprompt_Timeout
Allows you to specify the user prompt timeout interval, in seconds. The default value is 60 seconds.

If you specify the Noprompt_Timeout qualifier or the Prompt_Timeout=0, the RMU Show Statistics command does not time out any user prompts. Note that this can cause your database to hang.

Note

Oracle Corporation recommends that you do not use the Noprompt_ Timeout qualifier or the Prompt_Timeout= 0 qualifier unless you are certain that prompts will always be responded to in a timely manner.

If the Prompt_Timeout qualifier is specified with a value greater than 0 but less than 10 seconds, the value 10 is used. The user prompt timeout interval can also be specified using the PROMPT_TIMEOUT configuration variable.

Reopen_Interval=minutes

After the specified interval, closes the current output file and opens a new output file without requiring you to exit from the Performance Monitor. The new output file has the same name as the previous output file, but the version number is incremented by 1.

This qualifier allows you to view data written to the output file while the Performance Monitor is running.

If there has been no database activity at the end of the specified interval, the current output file is not closed and a new output file is not created.

Be careful not to use the DCL PURGE command inadvertently. Also note that use of the DCL SET FILE/VERSION_LIMIT command causes older versions of the output file to be deleted automatically.

Use of the Reopen_Interval qualifier is only valid when you also specify the Output qualifier.

Reset

Specifies that you want the Performance Monitor to reset your display to zero. The Reset qualifier has the same effect as selecting the reset option from the interactive screen (except when you specify the Reset qualifier, values are reset before being initially displayed).

Note that this qualifier resets the values being displayed to your output device only, it does *not* reset the values in the database global section nor does it affect the data collected in an output file.

The default behavior of the Performance Monitor is to display each change in values that has occurred since the database was opened. To display only the value changes that have occurred since the Performance Monitor was invoked, specify the Reset qualifier, or immediately select the on-screen reset option when statistics are first displayed.

The Reset qualifier does not affect the values that are written to the binary output file (created when you specify the Output qualifier). Specify the Reset qualifier when you replay the output file if you want the replay to display only the change in values that occurred between the time the Performance Monitor was invoked (with the Output qualifier) and the monitoring session ended.

Screen=screen-name

Specifies the first screen to be displayed. This is particularly useful when you are using the Performance Monitor to interactively monitor stalled processes. For example, the following command automatically warns the system operator of excessive stalls:

```
$ RMU/SHOW STATISTICS/ALARM=5/NOTIFY=OPER12/SCREEN="Stall Messages" -
_$ MF_PERSONNEL
```

The following list describes the syntax of the screen-name argument:

- You can use any unique portion of the desired screen name for the screen-name argument. For example, the following has the same results as the preceding example:

  ```
  $ RMU/SHOW STATISTICS/ALARM=5/NOTIFY=OPER12/SCREEN="Stall" -
  _$ MF_PERSONNEL.RDB
  ```

- Except with regards to case, whatever unique portion of the screen you supply must be an exact match to the equivalent portion of the actual screen name.

  For example Screen="Stall" is equivalent to Screen="STALL"; however Screen="Stalled" is not.

- If the specified screen-name does not match any known screen name, the display starts with the Summary IO Statistics screen (the default first screen). No error message is produced.
- If the screen name contains spaces, enclose the screen-name in quotes.
- You can not specify the "by-lock" or "by-area" screens.

If you specify the Nointeractive qualifier, the Screen qualifier is ignored.

Stall_Log=file-spec

Specifies that stall messages are to be written to the specified file. This can be useful when you notice a great number of stall messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the Stall_Log qualifier can be reviewed later so that the problem can be traced and resolved.

The stall messages are written to the file in a format similar to the Stall Messages screen. Stall messages are written to the file at the same rate as the screen refresh rate. (The refresh rate is set with the Time qualifier or from within the Performance Monitor with the Set_rate on-screen menu option.) Specifying a large refresh rate minimizes the size of the file, but results in a large number of missed stall messages. Specifying a small refresh rate produces a large log file, but contains more of the stall messages generated.

You do not need to be displaying the Stall Messages screen to record the stall messages to the log file. The stall log is maintained regardless of which screen, if any, is displayed.

By default, stall messages are not logged to a file.

Time=integer

Specifies the statistics collection interval in seconds. If you omit this qualifier, a sample collection is made every 3 seconds. The integer has a *normal* range of 1 to 180 (1 second to 3 minutes). However, if you specify a negative number for the Time qualifier, the RMU Show Statistics command interprets the number as hundredths of a second. For example, Time=–20 specifies an interval of 20/100 or 1/5 of a second.

If you are running the RMU Show Statistics command interactively, it updates the screen display at the specified interval.

If you also use the Output qualifier, a binary statistics record is written to the output file at the specified interval. A statistics record is not written to this file if no database activity has occurred since the last record was written.

Until=date-time

Specifies the time the statistics collection ends. When this point is reached, the RMU Show Statistics command terminates and control returns to the system command level. When the RMU Show Statistics command is executed in a batch job, the batch job terminates at the time specified.

An example of using the Until qualifier follows:

```
$ DEFINE LIB$DT_INPUT_FORMAT "!MAU !DB, !Y4 !H04:!M0:!S0.!C2"
$ RMU/SHOW STATISTICS /UNTIL="JUNE 16, 1996 17:00:00.00" -
_$ MF_PERSONNEL
```

This stops execution of the RMU Show Statistics command at 5 P.M. on June 16, 1996. You can omit the date if you wish to use the default of today's date.

You can use either an absolute or delta value to specify the data and time.

If you do not use the Until qualifier, the RMU Show Statistics command continues until you terminate it manually. In an interactive session, terminate the command by pressing Ctrl/Z or by selecting Exit from the menu. When you are running the RMU Show Statistics command with the Nointeractive qualifier from a terminal, terminate the command by pressing Ctrl/C or Ctrl/Y and then selecting Exit. When you are running the RMU Show Statistics command in a batch job, terminate the command by deleting the batch job.

Write_Report_Delay=integer
NoWrite_Report_Delay
The /WRITE_REPORT_DELAY=n qualifier specifies that statistics are to be collected for "n" seconds (default of 60 seconds) and then a report file written and then the RMU /SHOW STATISTICS utility will exit. /WRITE_REPORT_ DELAY implies /NOINTERACTIVE. It can be negated with /NoWrite_Report_ Delay.

Usage Notes

- Refer to the *Oracle Rdb7 Guide to Database Performance and Tuning* for complete information about the RMU Show Statistics command, including information about using formatted binary output files from the RMU Show Statistics command.
- To use the RMU Show Statistics command for a database, you must have the RMU$SHOW privilege in the root file ACL for the database or the OpenVMS SYSPRV, BYPASS, or WORLD privilege.

  To use the RMU Show Statistics command to display statistics about other users, you must have the OpenVMS WORLD privilege.

  To use the RMU Show Statistics command to update fields in the Database Dashboard (specified with the Options=Update qualifier), you must have both the OpenVMS WORLD and BYPASS privileges.
- If a database recovery process is underway, you cannot exit the Performance Monitor using Ctrl/Z or "E" from the interactive display menu. You must use Ctrl/Y or wait for the recovery process to complete. Exiting from the Performance Monitor causes Oracle RMU to request

several locks; however, these locks cannot be granted because the recovery process stalls all new lock requests until the recovery is complete.

- RMU expects a minimum screen dimension of 24 rows and 80 columns. Some screens require more rows and a larger column width. Oracle recommends setting your terminal characteristics large enough to view details on the displays of interest.
- Each release of Oracle Rdb introduces changes to the output statistics file structures (new fields, changes of data type, removal of obsolete fields, moved fields, and so on). If you are having a problem with an application that accesses the RMU Show Statistics field structures, recompile your application with the latest RMU$SHOW_STATISTICSnn.CDO or RMU$SHOW_STATISTICSnn.H (where nn is the version of Oracle Rdb you are using).
- The Oracle Rdb RMU Show Statistics command displays process CPU times in excess of 1 day. Because the width of the CPU time display is limited, the following CPU time display formats are used:
  - For CPU time values less than 1 day: "HH:MM:SS.CC"
  - For CPU time values less than 100 days but more than 1 day: "DD HH:MM"
  - For CPU time values more than 100 days: "DDD HH:MM"
- The following caveats apply to the Cluster Statistics Collection and Presentation feature:
  - Up to 95 cluster nodes can be specified. However, use cluster statistics collection prudently, as the system overhead in collecting the remote statistics may be substantial depending on the amount of information being transmitted on the network.
  - Cluster statistics are collected at the specified display refresh rate. Therefore, set the display refresh rate to a reasonable rate based on the number of cluster nodes being collected. The default refresh rate of 3 seconds is reasonable for most remote collection loads.
  - If you specify the Cluster qualifier, the list of cluster nodes applies to any database accessed during the Show Statistics session. When you access additional databases using the Switch Database option, the same cluster nodes are automatically accessed. However, any nodes that you added manually using the Cluster Statistics menu are not automatically added to the new database's remote collection.

In other words, manually adding and deleting cluster nodes affects only the current database and does not apply to any other database that you may have accessed during the session. For example, when you run the Show Statistics utility on node ALPHA3 with manually added node BONZAI, subsequently switching to BONZAI as the current node will not display cluster statistics from node ALPHA3 unless you manually add that node. Furthermore, switching back to node ALPHA3 as the current node loses the previous collection of node BONZAI because it was manually added.

- Both DECnet and TCP/IP network protocols are supported. By default, the DECnet protocol is used. To explicitly specify which network protocol to use, define the RDM$BIND_STT_NETWORK_TRANSPORT to DECNET or TCPIP respectively. The RDM$BIND_STT_NETWORK_TRANSPORT logical name must be defined to the same definition on both the local and cluster nodes. The RDM$BIND_STT_NETWORK_TRANSPORT logical name can be specified in LNM$FILE_DEV on the local node but must be specified in the LNM$SYSTEM_TABLE on all remote nodes.

Note

There is no command qualifier to specify the network protocol.

- The Output qualifier continues to work as usual, but when in cluster mode writes the cluster statistics information to the binary output file.
- The Cluster qualifier cannot be specified with the Input qualifier. Furthermore, the online selection of cluster nodes is not available when you use the Input qualifier.
- While the collection and presentation feature is active, all on-screen menu options continue to operate as usual. This includes the time-plot, scatter-plot, screen pause, and various other options.
- There is no way to exclude the current node from statistics collection. Log in to another node if you want to do this.
- The cluster collection of per-process stall information automatically detects the binding or unbinding of processes to cluster databases. There is no need to manually refresh the database information on the current node.

- If the database is not currently open on the specified node, Oracle RMU still attempts to collect cluster statistics. However, you must open the remote database prior to regular process attaches.
- When you display any of the per-process screens that support cluster statistics collection, such as the Stall Messages screen, you can zoom in on any of the displayed processes to show which node that process is using.
- Using the Cluster Statistics submenu from the Tools menu, it is also possible to collect statistics from all open database nodes using the Collect From Open Database Nodes menu option. This option simplifies the DBA's job of remembering where the database is currently open. However, subsequently opened nodes are not automatically added to the collection; these must be manually added.
- The cluster statistics collection is an intracluster feature in that it works only on the same database, using the same device and directory specification used to run the initial RMU Show Statistics command (that is, on a shared disk). The cluster statistics collection does not work across clusters (intercluster).
- When you replay a binary output file, the screen header region accurately reflects the number of cluster nodes whose statistics are represented in the output file.
- If you plan on using the cluster capability of RMU/SHOW STATISTICS, Oracle recommends that you install the RDMSTT73.EXE image on OpenVMS with the appropriate privileges.

  The RMONSTART73.COM command procedure provided by the Oracle Rdb installation allows the system manager to optionally install the RDMSTT73.EXE image at monitor startup time. To take advantage of this, you will need to edit the SYS$STARTUP:RMONSTART73.COM procedure and remove the comment characters from the following lines:

```
$ ! DEFX SYS$COMMON:[SYSEXE]RDMSTT73.EXE
$ !     REMOVEX
$ !     ADDX /OPEN/HEAD/PROT/PRIV=(CMKRNL,SYSPRV,SHARE)
```

  Also, edit the command procedure SYS$STARTUP:RMONSTOP73.COM and remove the comment characters from the following lines:

```
$ ! DEFX SYS$COMMON:[SYSEXE]RDMSTT73.EXE
$ !     REMOVEX
```

Examples

Example 1

The following example directs the results of the RMU Show Statistics command to an output file:

```
$ RMU/SHOW STATISTICS MF_PERSONNEL/OUTPUT=PERS.LOG
```

Example 2

The following example formats the binary results created in the previous example and produces a readable display:

```
$ RMU/SHOW STATISTICS/INPUT=PERS.LOG
```

Example 3

The following DCL script shows a complete example of how to create an excessive stall notification server using the operator notification facility. To execute this script, submit it to any queue on the node from which you want to run the script. Supply the parameters as follows:

- P1 is the database pathname.
- P2 is the completion time.
- P3 is the set of operators to be notified. You must enclose the list of operators in quotes.

```
$ VERIFY = F$VERIFY(0)
$ SET NOON
$!
$! Get the database name.
$!
$ IF P1 .EQS. "" THEN INQUIRE P1 "_database"
$!
$! Get the termination date/time.
$!
$ IF P2 .EQS. "" THEN INQUIRE P2 "_until"
$!
$! Get the operator classes.
$!
$ IF P3 .EQS. "" THEN INQUIRE P3 "_operators"
$!
$ RMU/SHOW STATISTICS/TIME=1/NOBROADCAST -
      /NOINTERACTIVE /UNTIL="''P2'" /ALARM=5 /NOTIFY='P3 -
      'P1
$ VERIFY = F$VERIFY(VERIFY)
$ EXIT
```

Example 4

You can use the Lock_Timeout or Deadlock qualifiers to construct a Lock Event Logging server. The following OpenVMS DCL script shows how to create a server that logs both lock timeout and lock deadlock events on the MF_PERSONNEL database for the next 15 minutes:

```
$ RMU/SHOW STATISTICS /NOHISTOGRAM /TIME=1 /NOINTERACTIVE -
_$ /LOCK_TIMEOUT_LOG=TIMEOUT.LOG /DEADLOCK_LOG=DEADLOCK.LOG -
_$ /NOBROADCAST /UNTIL="+15:00" MF_PERSONNEL
```

Example 5

The following example shows stall log information first with and then without the lock information:

```
$ RMU /SHOW STATISTICS /NOINTERACTIVE /STALL_LOG=SYS$OUTPUT: -
_$ DUA0:[DB]MFP.RDB
 Oracle Rdb X7.1-00 Performance Monitor Stall Log
 Database DPA500:[RDB_RANDOM.RDB_RANDOM_TST_247]RNDDB.RDB;1
 Stall Log created  4-SEP-2001 11:27:03.96
11:27:03.96 0002B8A1:1 11:27:03.67 waiting for record 118:2:2 (PR)
   State... Process.ID Process.name... Lock.ID. Rq Gr Queue "record 118:2:2"
   Blocker: 000220A7   RND_TST_24716   0F019E52    EX Grant
   Waiting: 0002B8A1   RND_TST_24715   4500C313 PR    Wait
11:27:03.96 0002B8A8:1 11:27:02.32 waiting for record 101:3:0 (EX)
   State... Process.ID Process.name... Lock.ID. Rq Gr Queue "record 101:3:0"
   Blocker: 000220AD   RND_TST_24710   0B00176A    PR Grant
   Blocker: 000220A7   RND_TST_24716   52018A3F    PR Grant
   Waiting: 0002B8A8   RND_TST_2474    3C00B5AF EX PR Cnvrt
11:27:03.96 0002B89C:1 11:27:00.15 waiting for record 114:4:1 (PR)
   State... Process.ID Process.name... Lock.ID. Rq Gr Queue "record 114:4:1"
   Blocker: 000220A7   RND_TST_24716   180033CC    EX Grant
   Waiting: 0002B89C   RND_TST_2479    110066BA PR    Wait

$ RMU /SHOW STATISTICS /NOINTERACTIVE /STALL_LOG=SYS$OUTPUT: -
_$ DUA0:[DB]MFP.RDB /OPTIONS=NOLOG_STALL_LOCK
 Oracle Rdb X7.1-00 Performance Monitor Stall Log
 Database DPA500:[RDB_RANDOM.RDB_RANDOM_TST_247]RNDDB.RDB;1
 Stall Log created  4-SEP-2001 11:28:34.68
11:28:34.69 0002B8B8:1 11:28:33.69 waiting for logical area 146 (PR)
11:28:34.69 0002B8A8:1 11:28:32.76 waiting for record 114:4:2 (PR)
11:28:34.69 0002B8B3:1 11:28:33.06 waiting for record 114:4:2 (PR)
11:28:34.69 0002B8B0:1 11:28:31.96 waiting for record 111:7:7 (EX)
```

1.67.10 RMU Show System Command

Displays a summary of which databases are in use on a particular node, the monitor log file specification, the number of monitor buffers available, and if after-image journal (AIJ) backup operations have been suspended.

This command is the same as the RMU Show Users command, except that it has no root-file-spec parameter. You can use it to see systemwide user information only.

Format

RMU/Show System

Command Qualifier	Default
/Output[=file-name]	/Output = SYS$OUTPUT

Description

The RMU Show System command displays information about all active database users on a particular node.

Command Qualifiers

Output[=file-name]
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify only a file name without an extension.

Usage Notes

- To use the RMU Show System command, you must have the OpenVMS WORLD privilege.
- When the database monitor is completely idle, identified in the output of the RMU Show Users command by the "no databases accessed on this node" message, the number of available monitor messages should be 1 less than the maximum. During periods of monitor activity, it is normal for the number of available monitor buffers to be less than the maximum, depending on how much work remains for the monitor to process.

Examples

Example 1

The following command lists the file specification for the monitor log file and databases currently in use.

```
$ RMU/SHOW SYSTEM
Oracle Rdb V7.0-64 on node NODEA 27-JUN-2002 16:23:43.92
  - monitor started 26-JUN-2002 06:33:07.33 (uptime 1 09:50:36)
  - monitor log filename is "$111$DUA366:[RDMMON_LOGS]RDMMON701_NODEA.LOG"

database $111$DUA619:[JONES.DATABASES.V70]MF_PERSONNEL.RDB;1
  - first opened 27-JUN-2002 16:23:42.11 (elapsed 0 00:00:01)
  * database is opened by an operator

database NODEB$DKB200:[RDB$TEST_SYSTEM.A70_RMU_4Z.SCRATCH]M_TESTDB.RDB;3
    - first opened 26-JUN-2002 23:24:41.55 (elapsed 0 16:59:02)
    * database is opened by an operator
    * After-image backup operations temporarily suspended from this node
    - current after-image journal file is DISK$RDBTEST8:[RDB$TEST_SYSTEM.A70_RMU
      _4Z]TEST3.AIJ;2
    - AIJ Log Server is active
    - 1 active database user
```

1.67.11 RMU Show Users Command

Displays information about active database users, the monitor log file specification, the number of monitor buffers available, and if after-image journal (AIJ) backup operations have been suspended. It allows you to see the user activity of specified databases on a specific node, and identifies the various nodes in the VMScluster where the database is currently open and available for use. In addition, this command indicates whether or not system space global sections are enabled.

If you are interested in information on users for a cluster, use the RMU Dump command with the Users qualifier.

Format

RMU/Show Users [root-file-spec]

Command Qualifier	Default
/Output[=file-name]	/Output = SYS$OUTPUT

Description

The RMU Show Users command displays information about all active database users or users of a particular database, the file specification for the monitor log file, the number of monitor buffers available, and if AIJ backup operations have been suspended.

This command also displays global buffer information for the node on which the RMU Show Users command is issued and displays global buffer information for the specified database only if global buffers are enabled for that database.

Command Parameters

root-file-spec
The root file specification of the database for which you want information. This parameter is optional. If you specify it, only users of that database are shown. Otherwise, all users of all active databases on your current node are shown.

Command Qualifiers

Output[=file-name]
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify a file name.

Usage Notes

- To use the RMU Show Users command for a specified database, you must have the RMU$SHOW, RMU$BACKUP, or RMU$OPEN privilege in the root file access control list (ACL) of the database, or the OpenVMS WORLD privilege.

  To use the RMU Show Users command without specifying a database, you must have the RMU$SHOW, RMU$BACKUP, or RMU$OPEN privilege in the root file ACL of the database or databases, and the OpenVMS WORLD privilege.
- When the database monitor is completely idle, identified in the output of the RMU Show Users command by the "no databases accessed on this node" message, the number of available monitor messages should be 1 less than the maximum. During periods of monitor activity, it is normal for the number of available monitor buffers to be less than the maximum, depending on how much work remains for the monitor to process.

Examples

Example 1

The following command lists current users information in the file DBUSE.LIS:

```
$ RMU/SHOW USERS/OUTPUT=DBUSE
```

Example 2

The following example shows all active users:

```
$ RMU/SHOW USER

Oracle Rdb V7.0-64 on node NODEA 27-JUN-2002 16:25:49.64
    - monitor started 26-JUN-2002 06:33:07.33 (uptime 1 09:52:42)
    - monitor log filename is "$DISK1:[LOGS]MON701_NODEA.LOG;12"
```

RMU Show Users Command

```
database DISK2:[TEST]M_TESTDB.RDB;3
    - first opened 26-JUN-2002 23:24:41.55 (elapsed 0 17:01:08)
    * database is opened by an operator
    * After-image backup operations temporarily suspended from this node
    - current after-image journal file is DISK3:[TEST1]TEST3.AIJ;2
   - AIJ Log Server is active
    - 1 active database user
    - database also open on these nodes:
      NODEB
    - 23225948:1 - RDM_4 - non-utility server, USER1 - active user
        - image DISK4:[SYS1.SYSCOMMON.][SYSEXE]RDMALS701.EXE;567
```

1.67.12 RMU Show Version Command

Displays the currently executing Oracle Rdb software version number along with information about the system architecture and OpenVMS version.

Format

RMU/Show Version [root-file-spec]

Command Qualifier	Default
/Output[=file-name]	/Output = SYS$OUTPUT

Description

This command is useful when you have multiple versions of Oracle Rdb running on your system and perhaps multiple databases. If the currently executing version of Oracle Rdb is not the version required to access the database, change the current version of Oracle Rdb to the required version. See Example 3 in the Examples section.

Command Parameters

root-file-spec
A database root file specification. The default file extension is .rdb. If you do not specify a database root file, RMU Show Version displays only the version of Oracle Rdb under which Oracle RMU is currently running.

Command Qualifiers

Output[=file-name]
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. The default output file extension is .lis, if you specify a file name.

Usage Notes

- You do not need any special privileges to use the RMU Show Version command.
- When the RMU Show Version command executes, it sets the following two DCL local symbols:
  - RMU$RDB_VERSION

    Set to the currently executing version of Oracle Rdb
  - RMU$DATABASE_VERSION

    Set to the version of Oracle Rdb required to access the specified database

  If you want to set the DCL symbols, RMU$RDB_VERSION and RMU$DATABASE_VERSION only and do not want the RMU Show Version output, specify the null device as the file name with the Output qualifier. For example:

```
$ RMU/SHOW VERSION MF_PERSONNEL /OUTPUT=NL:

$ SHOW SYMBOL RMU$RDB_VERSION
  RMU$RDB_VERSION = "7.0"

$ SHOW SYMBOL RMU$DATABASE_VERSION
  RMU$DATABASE_VERSION = "6.1"
```

Examples

Example 1

The following command displays the current version of Oracle Rdb software:

```
$ RMU/SHOW VERSION
Executing RMU for Oracle Rdb V7.2-400 on OpenVMS IA64 V8.3-1H1
```

Example 2

The following command displays the current version of Oracle Rdb software and the version of Oracle Rdb required to access the mf_personnel database:

```
$ RMU/SHOW VERSION MF_PERSONNEL
Executing RMU for Oracle Rdb V7.0-64
Database DISK:[MYDIR]MF_PERSONNEL.RDB;1 requires version 7.0
```

Example 3

The following example demonstrates how you might use the RMU Show Version command to determine how to access a database that is incompatible with the currently executing version of Oracle Rdb:

```
$ ! The RMU Show Version command tells you that the currently
$ ! executing version of Oracle Rdb is Version 7.0, but
$ ! that mf_personnel requires Version 6.1.
$
$ RMU/SHOW VERSION MF_PERSONNEL
Executing RMU for Oracle Rdb V7.0-00
Database DISK:[MYDIR]MF_PERSONNEL.RDB;1 requires version 6.1
$
$ ! If you ignore this information and attempt to attach to the
$ ! database, you receive an error.
$
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
%SQL-F-ERRATTDEC, Error attaching to database MF_PERSONNEL
-RDB-F-WRONG_ODS, the on-disk structure of database filename is
 not supported by version of facility being used
-RDMS-F-ROOTMAJVER, database format 61.0 is not compatible
 with software version 70.0
SQL> EXIT;
$ ! Assign the currently executing version of Oracle Rdb to
$ ! RMU$PREV_VERSION
$ !
$ rmu$prev_version := 'rmu$rdb_version'
$ !
$ ! Use the RDB$SETVER.COM command file to set the version of
$ ! Oracle Rdb to the version required by mf_personnel.
$ ! (For more information on the RDB$SETVER.COM command
$ ! file, see the Oracle Rdb Installation and Configuration Guide.)
$ !
$ @SYS$LIBRARY:RDB$SETVER 'RMU$DATABASE_VERSION'
$ !
$ ! Re-execute the RMU Show Version command to confirm that you have
$ ! the version of Oracle Rdb set correctly.
$ !
$ RMU/SHOW VERSION MF_PERSONNEL
Executing RMU for Oracle Rdb V6.1-00
Database DISK:[MYDIR]MF_PERSONNEL.RDB;1 requires version 6.1
$ ! Invoke SQL and attach to the mf_personnel database.
$ !
$ SQL
SQL>ATTACH 'FILENAME MF_PERSONNEL';
SQL> SHOW TABLES
User tables in database with filename MF_PERSONNEL
     CANDIDATES
     COLLEGES
     CURRENT_INFO                    A view.
     CURRENT_JOB                     A view.
```

```
     CURRENT_SALARY                 A view.
     DEGREES
     DEPARTMENTS
     EMPLOYEES
     JOBS
     JOB_HISTORY
     RESUMES
     SALARY_HISTORY
     WORK_STATUS
SQL> EXIT
$ !
$ !Reset the executing version of Oracle Rdb to the original setting.
$ !
$ @SYS$LIBRARY:RDB$SETVER 'RMU$PREV_VERSION'
```

1.68 RMU Unload Command

Copies the data from a specified table or view of the database into one of the following:

- A specially structured file that contains both the data and the metadata (.unl).
- An RMS file that contains data only (.unl). This file is created when you specify the Record_Definition qualifier. (The Record_Definition qualifier also creates a second file, with file extension .rrd, that contains the metadata.)

Data from the specially structured file can be reloaded by using the RMU Load command only. Data from the RMS file can be reloaded using the RMU Load command or by using an alternative utility such as is offered by DATATRIEVE.

Format

RMU/Unload root-file-spec table-name output-file-name

Command Qualifiers	Defaults
/Allocation=n	/Allocation=2048
/Buffers=n	See description
/Commit_Every=n	None
/[No]Compression[=options]	/Nocompression
/Debug_Options={options}	See description
/Delete_Rows	None
/[No]Error_Delete	See description
/Extend_Quantity=number-blocks	/Extend_Quantity=2048
/Fields=(column-name-list)	See description
/Flush={Buffer_End|On_Commit}	See description
/[No]Limit_To=n	/Nolimit_To
/Optimize={options}	None
/Record_Definition={([No]File|Path)=name,options}	See description
/Reopen_Count=n	None
/Row_Count=n	See description
/Statistics_Interval=seconds	See description
/Transaction_Type[=(transaction_mode,options...)]	See description
/[No]Virtual_Fields[=[No]Automatic,[No]Computed_By]	/Novirtual_Fields

1.68 RMU Unload Command

Description

The RMU Unload command copies data from a specified table or view and places it in a specially structured file or in an RMS file. Be aware that the RMU Unload command does not remove data from the specified table; it merely makes a copy of the data.

The RMU Unload command can be used to do the following:

- Extract data for an application that cannot access the Oracle Rdb database directly.
- Create an archival copy of data.
- Perform restructuring operations.
- Sort data by defining a view with a sorted-by clause, then unloading that view.

The specially structured files created by the RMU Unload command contain metadata for the table that was unloaded. The RMS files created by the RMU Unload command contain only data; the metadata can be found either in the data dictionary or in the .rrd file created using the Record_Definition qualifier. Specify the Record_Definition qualifier to exchange data with an application that uses RMS files.

The LIST OF BYTE VARYING (segmented string) data type cannot be unloaded into an RMS file; however, it can be unloaded into the specially structured file type.

Data type conversions are valid only if Oracle Rdb supports the conversion.

The RMU Unload command executes a read-only transaction to gather the metadata and user data to be unloaded. It is compatible with all operations that do not require exclusive access.

Command Parameters

root-file-spec
The root file specification of the database from which tables or views will be unloaded. The default file extension is .rdb.

table-name
The name of the table or view to be unloaded, or its synonym.

output-file-name
The destination file name. The default file extension is .unl.

Command Qualifiers

Allocation=n
Enables you to preallocate the generated output file. The default allocation is 2048 blocks; when the file is closed it is truncated to the actual length used.

If the value specified for the Allocation qualifier is less than 65535, it becomes the new maximum for the Extend_Quantity qualifier.

Buffers=n
Specifies the number of database buffers used for the unload operation. If no value is specified, the default value for the database is used. Although this qualifier might affect the performance of the unload operation, the default number of buffers for the database usually allows adequate performance.

Commit_Every=n
Turns the selection query into a WITH HOLD cursor so that the data stream is not closed by a commit. Refer to the *Oracle Rdb7 SQL Reference Manual* for more information about the WITH HOLD clause.

Compression[=options]
NoCompression
Data compression is applied to the user data unloaded to the internal (interchange) format file. Table rows, null byte vector and LIST OF BYTE VARYING data are compressed using either the LZW (Lempel-Ziv-Welch) technique or the ZLIB algorithm developed by Jean-loup Gailly and Mark Adler. Table metadata (column names and attributes) are never compressed and the resulting file remains a structured interchange file. Allowing compression allows the result data file to be more compact, using less disk space and permitting faster transmission over communication lines. This file can also be processed using the RMU Dump Export command.

The default value is Nocompression.

This qualifier accepts the following optional keywords (ZLIB is the default if no compression algorithm is specified):

- LZW

  Selects the LZW compression technique.

- ZLIB

  Selects the ZLIB compression technique. This can be modified using the LEVEL option.

- LEVEL=number

ZLIB allows further tuning with the LEVEL option that accepts a numeric level between 1 and 9. The default of 6 is usually a good trade off between result file size and the CPU cost of the compression.

- EXCLUDE_LIST[=(column-name,...)]

  It is possible that data in LIST OF BYTE VARYING columns is already in a compressed format (for instance images as JPG data) and therefore need not be compressed by RMU Unload. In fact, compression in such cases might actually cause the output to grow. The EXCLUDE_LIST option will disable compression for LIST OF BYTE VARYING columns. Specific column names can be listed, or if omitted, all LIST OF BYTE VARYING columns will be excluded from compression.

Only the user data is compressed. Therefore, additional compression may be applied using various third party compression tools, such as ZIP. It is not the goal of RMU to replace such tools.

The qualifier RECORD_DEFINITION (or RMS_RECORD_DEF) is not compatible /COMPRESSION. Note that the TRIM option for DELIMITED format output can be used to trim trailing spaces from VARCHAR data.

Debug_Options={options}

The Debug_Options qualifier allows you to turn on certain debug functions. The Debug_Options qualifier accepts the following options:

- [NO]TRACE

  Traces the qualifier and parameter processing performed by RMU Unload. In addition, the query executed to read the table data is annotated with the TRACE statement at each Commit (controlled by Commit_Every qualifier). When the logical name RDMS$SET_FLAGS is defined as "TRACE", then a line similar to the following is output after each commit is performed.

  ```
  ~Xt: 2009-04-23 15:16:16.95: Commit executed.
  ```

  The default is NOTRACE.

  ```
  $RMU/UNLOAD/REC=(FILE=WS,FORMAT=CONTROL) SQL$DATABASE WORK_STATUS WS/DEBUG=TRACE
  Debug = TRACE
  * Synonyms are not enabled
  Row_Count = 500
  Message buffer: Len: 13524
  Message buffer: Sze: 27, Cnt: 500, Use: 4 Flg: 00000000
  %RMU-I-DATRECUNL,   3 data records unloaded.
  ```

- [NO]FILENAME_ONLY

When the qualifier Record_Definition=Format:CONTROL is used, the name of the created unload file is written to the control file (.CTL). When the keyword FILENAME_ONLY is specified, RMU Unload will prune the output file specification to show only the file name and type. The default is NOFILENAME_ONLY.

```
$RMU/UNLOAD/REC=(FILE=TT:,FORMAT=CONTROL) SQL$DATABASE WORK_STATUS WS/DEBUG=
FILENAME
--
-- SQL*Loader Control File
--   Generated by: RMU/UNLOAD
--   Version:      Oracle Rdb X7.2-00
--   On:           23-APR-2009 11:12:46.29
--
LOAD DATA
INFILE 'WS.UNL'
APPEND
INTO TABLE "WORK_STATUS"
(
 STATUS_CODE                    POSITION(1:1) CHAR NULLIF (RDB$UL_NB1 = '1')
,STATUS_NAME                    POSITION(2:9) CHAR NULLIF (RDB$UL_NB2 = '1')
,STATUS_TYPE                    POSITION(10:23) CHAR NULLIF (RDB$UL_NB3 = '1')
-- NULL indicators
,RDB$UL_NB1               FILLER POSITION(24:24) CHAR -- indicator for
STATUS_CODE
,RDB$UL_NB2               FILLER POSITION(25:25) CHAR -- indicator for
STATUS_NAME
,RDB$UL_NB3               FILLER POSITION(26:26) CHAR -- indicator for
STATUS_TYPE
)
%RMU-I-DATRECUNL,   3 data records unloaded.
```

- [NO]HEADER

  This keyword controls the output of the header in the control file. To suppress the header use NOHEADER. The default is HEADER.

- APPEND, INSERT, REPLACE, TRUNCATE

  These keywords control the text that is output prior to the INTO TABLE clause in the control file. The default is APPEND, and only one of these options can be specified.

- OFFSET

  This keyword tells RMU/UNLOAD to include a comment containing each field length and offset within the output record. Here is an example of the comment string for each field.

1.68 RMU Unload Command

```
$       RMU/UNLOAD-
            /RECORD=(file=SALARY_HISTORY)-
            /DEBUG_OPTIONS=OFFSET-
            PERSONNEL -
            SALARY_HISTORY -
            SALARY_HISTORY
%RMU-I-DATRECUNL,   729 data records unloaded
$       type SALARY_HISTORY.RRD
DEFINE FIELD EMPLOYEE_ID DATATYPE IS TEXT SIZE IS 5.
DEFINE FIELD SALARY_AMOUNT DATATYPE IS SIGNED LONGWORD SCALE -2.
DEFINE FIELD SALARY_START DATATYPE IS DATE.
DEFINE FIELD SALARY_END DATATYPE IS DATE.
DEFINE RECORD SALARY_HISTORY.
   EMPLOYEE_ID .                            /* Offset = 0 Length = 5 */
   SALARY_AMOUNT .                          /* Offset = 5 Length = 4 */
   SALARY_START .                           /* Offset = 9 Length = 8 */
   SALARY_END .                             /* Offset = 17 Length = 8 */
END SALARY_HISTORY RECORD.                  /* Total Length = 25 */
$
```

Delete_Rows
Specifies that Oracle Rdb delete rows after they have been unloaded from the database. You can use this qualifier with the Commit_Every qualifier to process small batches of rows.

If constraints, triggers, or table protection prevent the deletion of rows, the RMU Unload operation will fail. The Delete_Rows qualifier cannot be used with non-updatable views, those containing joins, or aggregates (union or group by).

Error_Delete
Noerror_Delete
Specifies whether the unload and record definition files should be deleted on error. By default, the RMU Unload command deletes the unload and record definition files if an unrecoverable error occurs that causes an abnormal termination of the unload command execution. Use the Noerror_Delete qualifier to retain the files.

If the Delete_Rows qualifier is specified, the default for this qualifier is Noerror_Delete. This default is necessary to allow you to use the unload and record definition files to reload the data if an unrecoverable error has occurred after the delete of some of the unloaded rows has been committed. Even if the unload file is retained, it may not be able to reload the data using the RMU Load command if the error is severe enough to prevent the RMU error handler from continuing to access the unload file once the error is detected.

If the Delete_Rows qualifier is not specified, the default is Error_Delete.

Extend_Quantity=number-blocks
Sets the size, in blocks, by which the unload file (.unl) can be extended. The minimum value for the number-blocks parameter is 1; the maximum value is 65535. If you provide a value for the Allocation qualifier that is less than 65535, that value becomes the maximum you can specify.

If you do not specify the Extend_Quantity qualifier, the default block size by which .unl files can be extended is 2048 blocks.

Fields=(column-name-list)
Specifies the column or columns of the table or view to be unloaded from the database. If you list multiple columns, separate the column names with a comma, and enclose the list of column names within parentheses. This qualifier also specifies the order in which the columns should be unloaded if that order differs from what is defined for the table or view. Changing the structure of the table or view could be useful when restructuring a database or when migrating data between two databases with different metadata definitions. The default is all the columns defined for the table or view in the order defined.

Flush={Buffer_End | On_Commit}
Controls when internal RMS buffers are flushed to the unload file. By default, the RMU Unload command flushes any data left in the internal RMS file buffers only when the unload file is closed. The Flush qualifier changes that behavior. You must use one of the following options with the Flush qualifier:

- Buffer_End

  The Buffer_End option specifies that the internal RMS buffers be flushed to the unload file after each unload buffer has been written to the unload file.

- On_Commit

  The On_Commit option specifies that the internal RMS buffers be flushed to the unload file just before the current unload transaction is committed.

If the Delete_Rows qualifier is specified, the default for this qualifier is Flush=On_Commit. This default is necessary to allow you to use the unload and record definition files to reload the data if an unrecoverable error has occurred after the delete of some of the unloaded rows has been committed.

If the Delete_Rows qualifier is not specified, the default is to flush the record definition buffers only when the unload files are closed.

More frequent flushing of the internal RMS buffers will avoid the possible loss of some unload file data if an error occurs and the Noerror_Delete qualifer has been specified. Additional flushing of the RMS internal buffers to the unload file can cause the RMU Unload command to take longer to complete.

Limit_To=n
Nolimit_To
Limits the number of rows unloaded from a table or view. The primary use of the Limit_To qualifier is to unload a data sample for loading into test databases. The default is the Nolimit_To qualifier.

Optimize={options}
Controls the query optimization of the RMU Unload command. You must use one or more of the following options with the Optimize qualifier:

- Bitmapped_scan

  This option requests that the Rdb optimizer attempt to perform bitmapped scan when accessing multiple indices during the unload. This option is particularly useful for RMU Unload from complex views.

  This option cannot be specified at the same time as the Sequential_Access option.

- Conformance={Optional | Mandatory}

  This option accepts two keywords, Optional or Mandatory, which can be used to override the settings in the specified query outline.

  If the matching query outline is invalid, the Conformance=Mandatory option causes the query compile, and hence the RMU Unload operation, to stop. The query outline will be one which either matches the string provided by the Using_Outline or Name_As option or matches the query identification.

  The default behavior is to use the setting within the query outline. If no query outline is found, or query outline usage is disabled, then this option is ignored.

- Fast_First

  This option asks the query optimizer to favor strategies that return the first rows quickly, possibly at the expense of longer overall retrieval time. This option does not override the setting if any query outline is used.

This option cannot be specified at the same time as the Total_Time option.

Note

Oracle Corporation does not recommend this optimization option for the RMU Unload process. It is provided only for backward compatibility with prior Rdb releases when it was the default behavior.

- Name_As=query_name

  This option supplies the name of the query. It is used to annotate output from the Rdb debug flags (enabled using the logical RDMS$SET_FLAGS) and is also logged by Oracle TRACE.

  If the Using_Outline option is not used, this name is also used as the query outline name.

- Selectivity=selectivity-value

  This option allows you to influence the Oracle Rdb query optimizer to use different selectivity values.

  The Selectivity option accepts the following keywords:

  – Aggressive — assumes a smaller number of rows is selected compared to the default Oracle Rdb selectivity

  – Sampled — uses literals in the query to perform preliminary estimation on indices

  – Default — uses default selectivity rules

  The following example shows a use of the Selectivity option:

```
$RMU/UNLOAD/OPTIMIZE=(TOTAL_TIME,SELECTIVITY=SAMPLED) -
_$  SALES_DB CUSTOMER_TOP10 TOP10.UNL
```

  This option is most useful when the RMU Unlaod command references a view definition with a complex predicate.

- Sequential_Access

  This option requests that index access be disabled for this query. This is particularly useful for RMU Unload from views against strictly partitioned tables. Strict partitioning is enabled by the PARTITIONING IS NOT UPDATABLE clause on the CREATE or ALTER STORAGE MAP statements. Retrieval queries only use this type of partition optimization during sequential table access.

  This option cannot be specified at the same time as the Using_Outline option.

- Total_Time

  This option requests that total time optimization be applied to the unload query. It does not override the setting if any query outline is used.

  In some cases, total time optimization may improve performance of the RMU Unload command when the query optimizer favors overall performance instead of faster retrieval of the first row. Since the RMU Unload process is unloading the entire set, there is no need to require fast delivery of the first few rows.

  This option may not be specified at the same time as the Fast_First option. The Optimize=Total_Time behavior is the default behavior for the RMU Unload command if the Optimize qualifier is not specified.

- Using_Outline=outline_name

  This option supplies the name of the query outline to be used by the RMU Unload command. If the query outline does not exist, the name is ignored.

  This option may not be specified at the same time as the Sequential_Access option.

Record_Definition={([No]File | Path)=name,options}

Creates an RMS file containing the record structure definition for the output file. The record description uses the CDO record and field definition format. The default file extension is .rrd.

If you omit the File=name or Path=name option you must specify an option.

The date-time syntax in .rrd files generated by this qualifier changed in Oracle Rdb V6.0 to make the .rrd file compatible with the date-time syntax support for Oracle CDD/Repository V6.1. The RMU Unload command accepts both the date-time syntax generated by the Record_Definition qualifier in previous versions of Oracle Rdb and the syntax generated in Oracle Rdb V6.0 and later.

See Appendix A for more information on .rrd files and details on the date-time syntax generated by this qualifier.

The options are:

- Format=(Text)

  If you specify the Format=(Text) option, Oracle RMU converts all data to printable text before unloading it.

- Format=Control

  The Format=Control option provides support for SQL*Loader control files and portable data files. The output file defaults to type .CTL.

FORMAT=CONTROL implicitly uses a portable data format as TEXT rather than binary values. The unloaded data files are similar to that generated by FORMAT=TEXT but includes a NULL vector to represent NULL values ('1') and non-NULL values ('0').

The SQL*Loader control file uses this NULL vector to set NULL for the data upon loading.

When FORMAT=CONTROL is used, the output control file and associated data file are intended to be used with the Oracle Database SQL*Loader (sqlldr) command to load the data into an Oracle Database table. LIST OF BYTE VARYING (SEGMENTED STRING) columns are not unloaded.

The keywords NULL, PREFIX, SEPARATOR, SUFFIX, and TERMINATOR only apply to DELIMITED_TEXT format and may not be used in conjunction with the CONTROL keyword.

DATE VMS data is unloaded including the fractional seconds precision. However, when mapped to Oracle DATE type in the control file, the fractional seconds value is ignored. It is possible to modify the generated control file to use the TIMESTAMP type and add FF to the date edit mask.

Note

The RMU Load command does not support loading data using FORMAT=Control.

- Format=CSV

  The Format=CSV option allows for the CSV (comma separated list of values) format, used by many tools to load data. The output file type defaults to type .CSV.

  There is implicit conversion of DATE VMS to TIMESTAMP(2) so that formatting of text string is compatible with various CSV knowledgeable tools (such as Microsoft EXCEL).

  The first row is a list of column names. These values are formatted using the same PREFIX, SUFFIX, SEPARATOR and TERMINATOR strings as defined for the table data.

  The list of column names is re-generated when the unload file is reopened. See REOPEN_COUNT qualifier.

- Format=(XML[,xml-options])

The Format=XML option causes the output Record_Definition file type to default to .DTD (Document Type Definition). The output file defaults to type .XML. The contents of the data file is in XML format suitable for processing with a Web browser or XML application.

If you use the Nofile option or do not specify the File or Path keyword, the DTD is included in the XML output file (internal DTD). If you specify a name with the File or Path keyword to identify an output file, the file is referenced as an external DTD from within the XML file.

The XML file contains a single table that has the name of the database and multiple rows named <RMU_ROW>. Each row contains the values for each column in printable text. If a value is NULL, then the tag <NULL/> is displayed. Example 16 shows this behavior.

XML options (and their default values if you do not specify XML options) are:

– CHARACTER_ENCODING_XML=string

  Alternate character encoding for the generated XML file. The default is "ISO-8859-1". Other valid values can be found as part of the XML Standards definition at http://www.w3.org/standards/xml/core.

Note

The RMU Load command does not support loading data using FORMAT=XML.

- Format=(Delimited_Text [,delimiter-options])

  If you specify the Format=Delimited_Text option, Oracle RMU applies delimiters to all data before unloading it.

  Note that DATE VMS dates are output in the collatable time format, which is *yyyymmddhhmmsscc*. For example, March 20, 1993 is output as: 1993032000000000.

  If the Format option is not used, Oracle RMU outputs data to a fixed-length binary flat file. If the Format=Delimited_Text options is not used, VARCHAR(n) strings are padded with blanks when the specified string has fewer characters than *n* so that the resulting string is *n* characters long.

  Delimiter options (and their default values if you do not specify delimiter options) are:

  – Prefix=string
    NoPrefix

Specifies a prefix string that begins any column value in the input file. If you omit this option, the column prefix is assumed to consist of a quotation mark ("). If you specify NoPrefix or an empty string, then no prefix will be used.

- Separator=string
  NoSeparator

  Specifies a string that separates column values of a row. If you omit this option, the column separator is assumed to consist of a single comma (,). If you specify NoSeparator or an empty string, then no separator will be used.

- Suffix=string
  NoSuffix

  Specifies a suffix string that ends any column value in the input file. If you omit this option, the column suffix is assumed to consist of a quotation mark ("). If you specify NoSuffix or an empty string, then no suffix will be used.

- Terminator=string

  Specifies the row terminator that completes all the column values corresponding to a row. If you omit this option, the row terminator will be the end of the line.

- Null=string
  NoNull

  Specifies a string that, when found in the database column, is unloaded as "NULL" in the output file.

  The Null option can be specified *on the command line* as one of the following:

  * A quoted string with an alternate replacement string
  * An empty string ("") or NoNull to request no replacement string

  The string that represents the null character must be quoted on the Oracle RMU command line. You cannot specify a blank space or spaces as the null character. You cannot use the same character for the Null value and other Delimited_Text options.

Note

The values of each of the strings specified in the delimiter options must be enclosed within quotation marks. Oracle RMU strips these quotation marks while interpreting the values. If you want to specify

a quotation mark (") as a delimiter, specify a string of four quotation marks. Oracle RMU interprets four quotation marks as your request to use one quotation mark as a delimiter. For example, Suffix = """".

Oracle RMU reads these quotation marks as follows:

- The first quotation mark is stripped from the string.
- The second and third quotation mark are interpreted as your request for one quotation mark (") as a delimiter.
- The fourth quotation mark is stripped.

This results in one quotation mark being used as a delimiter.

Furthermore, if you want to specify a quotation mark as part of the delimited string, you must use two quotation marks for each quotation mark that you want to appear in the string. For example, Suffix = "**""** causes Oracle RMU to use a delimiter of **"**.

- Trim=option

  If you specify the Trim=option keyword, leading and/or trailing spaces area removed from each output field. Option supports three keywords:

  - TRAILING - trailing spaces will be trimmed from CHARACTER and CHARACTER VARYING (VARCHAR) data that is unloaded. This is the default setting if only the TRIM option is specified.
  - LEADING - leading spaces will be trimmed from CHARACTER and CHARACTER VARYING (VARCHAR) data that is unloaded.
  - BOTH - both leading and trailing spaces will be trimmed.

When the Record_Definition qualifier is used with load or unload operations, and the Null option to the Delimited_Text option is not specified, any null values stored in the rows of the tables being loaded or unloaded are not preserved. Therefore, if you want to preserve null values stored in tables and you are moving data within the database or between databases, specify the Null option with Delimited_Text option of the Record_Definition qualifier.

Reopen_Count=n

The Reopen_Count=n qualifier allows you to specify how many records are written to an output file. The output file will be re-created (that is, a new version of the file will be created) when the record count reaches the specified value. The Reopen_Count=n qualifier is only valid when used with the Record_Definition or Rms_Record_Def qualifiers.

Rms_Record_Def=({File | Path}=name [,options])
Synonymous with the Record_Definition qualifier. See the description of the Record_Definition qualifier.

Row_Count=n
Specifies that Oracle Rdb buffer multiple rows between the Oracle Rdb server and the RMU Unload process. The default value for *n* is 500 rows; however, this value should be adjusted based on working set size and length of unloaded data. Increasing the row count may reduce the CPU cost of the unload operation. For remote databases, this may significantly reduce network traffic for large volumes of data because the buffered data can be packaged into larger network packets.

The minimum value you can specify for *n* is 1. The default row size is the value specified for the Commit_Every qualifier or 500, whichever is smaller.

Statistics_Interval=seconds
Specifies that statistics are to be displayed at regular intervals so that you can evaluate the progress of the unload operation.

The displayed statistics include:

- Elapsed time
- CPU time
- Buffered I/O
- Direct I/O
- Page faults
- Number of records unloaded since the last transaction was committed
- Number of records unloaded so far in the current transaction

If the Statistics_Interval qualifier is specified, the seconds parameter is required. The minimum value is 1. If the unload operation completes successfully before the first time interval has passed, you receive only an informational message on the number of files unloaded. If the unload operation is unsuccessful before the first time interval has passed, you receive error messages and statistics on the number of records unloaded.

At any time during the unload operation, you can press Ctrl/T to display the current statistics.

Transaction_Type[=(transaction_mode,options,...)]
Allows you to specify the transaction mode, isolation level, and wait behavior for transactions.

1.68 RMU Unload Command

Use one of the following keywords to control the transaction mode:

- Automatic

  When Transaction_Type=Automatic is specified, the transaction type depends on the current database settings for snapshots (enabled, deferred, or disabled), transaction modes available to this user, and the standby status of the database. Automatic mode is the default.

- Read_Only

  Starts a Read_Only transaction.

- Exclusive

  Starts a Read_Write transaction and reserves the table for Exclusive_Read.

- Protected

  Starts a Read_Write transaction and reserves the table for Protected_Read.

- Shared

  Starts a Read_Write transaction and reserves the table for Shared_Read.

Use one of the following options with the keyword Isolation_Level=[option] to specify the transaction isolation level:

- Read_Committed
- Repeatable_Read
- Serializable. Serializable is the default setting.

Refer to the SET TRANSACTION statement in the Oracle Rdb SQL Reference Manual for a complete description of the transaction isolation levels.

Specify the wait setting by using one of the following keywords:

- Wait

  Waits indefinitely for a locked resource to become available. Wait is the default behavior.

- Wait=n

  The value you supply for *n* is the transaction lock timeout interval. When you supply this value, Oracle Rdb waits *n* seconds before aborting the wait and the RMU Unload session. Specifying a wait timeout interval of zero is equivalent to specifying Nowait.

- Nowait

  Does not wait for a locked resource to become available.

Virtual_Fields=([No]Automatic,[No]Computed_By)
Novirtual_Fields
The Virtual_Fields qualifier unloads any AUTOMATIC AS, COMPUTED BY or IDENTITY columns as real data. This qualifier permits the transfer of computed values to another application. It also permits unloading through a view that is a union of tables or that is comprised of columns from multiple tables. For example, if there are two tables, EMPLOYEES and RETIRED_ EMPLOYEES, the view ALL_EMPLOYEES (a union of EMPLOYEES and RETIRED_EMPLOYEES tables) can be unloaded.

The Novirtual_Fields qualifier is the default, which is equivalent to the Virtual_Fields=(Noautomatic,Nocomputed_By) qualifier.

If you specify the Virtual_Fields qualifier without a keyword, all columns are unloaded, including AUTOMATIC AS, COMPUTED BY and IDENTITY table columns, and calculated VIEW columns.

If you specify the Virtual_Fields=(Automatic,Nocomputed_By) qualifier or the Virtual_Fields=Nocomputed_By qualifier, data is only unloaded from Automatic columns.

If you specify the Virtual_Fields=(Noautomatic,Computed_By) qualifier or the Virtual_Fields=Noautomatic qualifier, data is only unloaded from Computed_ By columns.

Usage Notes

- To use the RMU Unload command for a database, you must have the RMU$UNLOAD privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege. You must also have the SQL SELECT privilege to the table or view being unloaded.
- For tutorial information on the RMU Unload command, refer to the *Oracle Rdb Guide to Database Design and Definition*.
- Detected asynchronous prefetch should be enabled to achieve the best performance of this command. You can determine the setting for your database by issuing the RMU Dump command with the Header qualifier.

  If detected asynchronous prefetch is disabled, and you do not want to enable it for the database, you can enable it for your Oracle RMU operations by defining the following logicals at the process level:

```
$ DEFINE RDM$BIND_DAPF_ENABLED 1
$ DEFINE RDM$BIND_DAPF_DEPTH_BUF_CNT P1
```

  P1 is a value between 10 and 20 percent of the user buffer count.

1.68 RMU Unload Command

- You can unload a table from a database structured under one version of Oracle Rdb and load it into the same table of a database structured under another version of Oracle Rdb. For example, if you unload the EMPLOYEES table from a mf_personnel database created under Oracle Rdb V6.0, you can load the generated .unl file into an Oracle Rdb V7.0 database. Likewise, if you unload the EMPLOYEES table from a mf_ personnel database created under Oracle Rdb V7.0, you can load the generated .unl file into an Oracle Rdb V6.1 database. This is true even for specially formatted binary files (created with the RMU Unload command without the Record_Definition qualifier). The earliest version into which you can load a .unl file from another version is Oracle Rdb V6.0.
- The RMU Unload command with the Record_Definition qualifier creates a fixed-length flat file in which each record begins at an even-numbered address. That is, if unloading a given record would cause the next record to begin on an odd-numbered address, Oracle RMU pads that record—thereby forcing all records to begin at an even-numbered address. This is not a concern if you are using the RMU Load command to load the fixed-length flat file. This is only a concern if you intend to use another application to load the data created by the RMU Unload command. In this case, make sure your application accounts for the padding that the RMU Unload command might add to data records.
- The Fields qualifier can be used with indirect file references. When you use the Fields qualifier with an indirect file reference in the field list, the referenced file is written to SYS$OUTPUT if you have used the DCL SET VERIFY command. See Section 1.3 for more information.
- To view the contents of the specially structured .unl file created by the RMU Unload command, use the RMU Dump Export command.
- To preserve the null indicator in a load or unload operation, use the Null option with the Record_Definition qualifier. Using the Record_Definition qualifier without the Null option replaces all null values with zeros; this can cause unexpected results with computed-by columns.
- Oracle RMU does not allow you to unload a system table.
- The RMU Unload command recognizes character set information. When you unload a table, RMU Unload transfers information about the character set to the record definition file.
- When it creates the record definition file, the RMU Unload command preserves any lowercase characters in table and column names by allowing delimited identifiers. **Delimited identifiers** are user-supplied names enclosed within quotation marks ("").

By default, RMU Unload changes any table or column (field) names that you specify to uppercase. To preserve lowercase characters, use delimited identifiers. That is, enclose the names within quotation marks. In the following example, RMU Unload preserves the uppercase and lowercase characters in "Last_Name" and "Employees":

```
$  RMU/UNLOAD/FIELDS=("Last_name",FIRST_NAME) TEST "Employees" TEST.UNL
```

Note

The data dictionary does not preserve the distinction between uppercase and lowercase identifiers. If you use delimited identifiers, you must be careful to ensure that the record definition does not include objects with names that are duplicates except for the case. For example, the data dictionary considers the delimited identifiers "Employee_ID" and "EMPLOYEE_ID" to be the same name.

- Oracle RMU does not support the multischema naming convention and returns an error if you specify one. For example:

```
$ RMU/UNLOAD CORPORATE_DATA ADMINISTRATION.PERSONNEL.EMPLOYEES -
_$ OUTPUT.UNL
%RMU-E-OUTFILDEL, Fatal error, output file deleted
-RMU-F-RELNOTFND, Relation (ADMINISTRATION.PERSONNEL.EMPLOYEES) not found
```

  When using a multischema database, you must specify the SQL stored name for the database object.

  For example, to find the stored name that corresponds to the ADMINISTRATION.PERSONNEL.EMPLOYEES table in the corporate_ data database, issue an SQL SHOW TABLE command, as follows:

```
SQL> SHOW TABLE ADMINISTRATION.PERSONNEL.EMPLOYEES
Information for table ADMINISTRATION.PERSONNEL.EMPLOYEES
    Stored name is EMPLOYEES
   .
   .
   .
```

  Then to unload the table, issue the following RMU Unload command:

```
$ RMU/UNLOAD CORPORATE_DATA EMPLOYEES OUTPUT.UNL
```

- If the Transaction_Type qualifier is omitted, a Read_Only transaction is started against the database. This behavior is provided for backward compatibility with prior Rdb releases. If the Transaction_Type qualifier is specified without a transaction mode, the default value Automatic is used.

- If the database has snapshots disabled, Oracle Rdb defaults to a READ WRITE ISOLATION LEVEL SERIALIZABLE transaction. Locking may be reduced by specifying Transaction_Type=(Automatic), or Transaction_Type=(Shared,Isolation_Level=Read_Committed).
- If you use a synonym to represent a table or a view, the RMU Unload command translates the synonym to the base object and processes the data as though the base table or view had been named. This implies that the unload interchange files (.UNL) or record definition files (.RRD) that contain the table metadata will name the base table or view and not use the synonym name. If the metadata is used against a different database, you may need to use the Match_Name qualifier to override this name during the RMU load process.

Examples

Example 1

The following command unloads the EMPLOYEE_ID and LAST_NAME column values from the EMPLOYEES table of the mf_personnel database. The data is stored in names.unl.

```
$ RMU/UNLOAD -
_$ /FIELDS=(EMPLOYEE_ID, LAST_NAME) -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-DATRECUNL, 100 data records unloaded.
```

Example 2

The following command unloads the EMPLOYEES table from the mf_personnel database and places the data in the RMS file, names.unl. The names.rrd file contains the record structure definitions for the data in names.unl.

```
$ RMU/UNLOAD/RECORD_DEFINITION=FILE=NAMES.RRD MF_PERSONNEL -
_$ EMPLOYEES NAMES.UNL
%RMU-I-DATRECUNL, 100 data records unloaded.
```

Example 3

The following command unloads the EMPLOYEE_ID and LAST_NAME column values from the EMPLOYEES table of the mf_personnel database and accepts the default values for delimiters, as shown by viewing the names.unl file:

```
$ RMU/UNLOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME) -
-$ /RECORD_DEFINITION=(FILE=NAMES, FORMAT=DELIMITED_TEXT) -
-$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-DATRECUNL, 100 data records unloaded.
$ !
$ ! TYPE the names.unl file to see the effect of the RMU Unload
$ ! command.
$ !
$ TYPE NAMES.UNL

"00164","Toliver       "
"00165","Smith         "
"00166","Dietrich      "
"00167","Kilpatrick    "
"00168","Nash          "
   .
   .
   .
```

Example 4

The following command unloads the EMPLOYEE_ID and LAST_NAME column values from the EMPLOYEES table of the mf_personnel database and specifies the asterisk (*) character as the string to mark the beginning and end of each column (the prefix and suffix string):

```
$ RMU/UNLOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME) -
_$ /RECORD_DEFINITION=(FILE=NAMES, -
_$ FORMAT=DELIMITED_TEXT, SUFFIX="*", -
_$ PREFIX="*") -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL
%RMU-I-DATRECUNL, 100 data records unloaded.
$ !
$ ! TYPE the names.unl file to see the effect of the RMU Unload
$ ! command.
$ !
$ TYPE NAMES.UNL
*00164*,*Toliver       *
*00165*,*Smith         *
*00166*,*Dietrich      *
*00167*,*Kilpatrick    *
*00168*,*Nash          *
*00169*,*Gray          *
*00170*,*Wood          *
*00171*,*D'Amico       *
   .
   .
   .
```

Example 5

1.68 RMU Unload Command

The following command unloads all column values from the EMPLOYEES table of the mf_personnel database, and specifies the Format=Text option of the Record_Definition qualifier. Oracle RMU will convert all the data to printable text, as can be seen by viewing the text_output.unl file:

```
$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=TEXT_RECORD,FORMAT=TEXT) -
_$ MF_PERSONNEL EMPLOYEES TEXT_OUTPUT
%RMU-I-DATRECUNL, 100 data records unloaded.
$ !
$ ! TYPE the text_output.unl file to see the effect of the RMU Unload
$ ! command.
$ !
$ TYPE TEXT_OUTPUT.UNL
00164Toliver       Alvin     A146 Parnell Place
Chocorua            NH03817M19470328000000001
00165Smith         Terry     D120 Tenby Dr.
Chocorua            NH03817M19540515000000002
00166Dietrich      Rick       19 Union Square
Boscawen            NH03301M19540320000000001
   .
   .
   .
```

Example 6

The following command unloads the EMPLOYEE_ID and LAST_NAME column values from the EMPLOYEES table of the mf_personnel database and requests that statistics be displayed on the terminal at 2-second intervals:

```
$ RMU/UNLOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME)   -
_$ /STATISTICS_INTERVAL=2             -
_$ MF_PERSONNEL EMPLOYEES NAMES.UNL

------------------------------------------------------------------------------
ELAPSED:0 00:00:02.16  CPU: 0:00:00.26  BUFIO: 13  DIRIO: 57  FAULTS: 598
  0 records unloaded.
------------------------------------------------------------------------------
------------------------------------------------------------------------------
ELAPSED:0 00:00:04.32  CPU: 0:00:00.68  BUFIO: 18  DIRIO: 102  FAULTS: 2121

  0 records unloaded.
------------------------------------------------------------------------------
------------------------------------------------------------------------------
ELAPSED:0 00:00:06.32  CPU: 0:00:00.92  BUFIO: 31  DIRIO: 158  FAULTS: 2483

  39 records unloaded.
------------------------------------------------------------------------------
%RMU-I-DATRECUNL, 100 data records unloaded.
$
```

Example 7

The following example unloads a subset of data from the EMPLOYEES table, using the following steps:

1. Create a temporary view on the EMPLOYEES table that includes only employees who live in Massachusetts.
2. Use an RMU Unload command to unload the data from this view.
3. Delete the temporary view.

```
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> CREATE VIEW MA_EMPLOYEES
cont>        (EMPLOYEE_ID,
cont>         LAST_NAME,
cont>         FIRST_NAME,
cont>         MIDDLE_INITIAL,
cont>         STATE,
cont>         STATUS_CODE)
cont>   AS SELECT
cont>         E.EMPLOYEE_ID,
cont>         E.LAST_NAME,
cont>         E.FIRST_NAME,
cont>         E.MIDDLE_INITIAL,
cont>         E.STATE,
cont>         E.STATUS_CODE
cont>      FROM EMPLOYEES E
cont>      WHERE E.STATE='MA';
SQL> COMMIT;
SQL> EXIT;

$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=MA_EMPLOYEES,FORMAT=DELIMITED_TEXT) -
_$ MF_PERSONNEL MA_EMPLOYEES MA_EMPLOYEES.UNL
%RMU-I-DATRECUNL, 9 data records unloaded.

$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> DROP VIEW MA_EMPLOYEES;
SQL> COMMIT;
```

Example 8

The following example shows that null values in blank columns are not preserved unless the Null option is specified with the Delimited_Text option of the Record_Definition qualifier:

```
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> --
SQL> -- Create the NULL_DATE table:
SQL> CREATE TABLE NULL_DATE
cont> (COL1 VARCHAR(5),
cont>  DATE1 DATE,
cont>  COL2 VARCHAR(5));
```

1.68 RMU Unload Command

```
SQL> --
SQL> -- Store a row that does not include a value for the DATE1
SQL> -- column of the NULL_DATE table:
SQL> INSERT INTO NULL_DATE
cont>            (COL1, COL2)
cont>    VALUES ('first','last');
1 row inserted
SQL> --
SQL> COMMIT;
SQL> --
SQL> -- The previous SQL INSERT statement causes a null value to
SQL> -- be stored in NULL_DATE:
SQL> SELECT * FROM NULL_DATE;
 COL1    DATE1                     COL2
 first   NULL                      last
1 row selected
SQL> --
SQL> DISCONNECT DEFAULT;
SQL> EXIT;
$ !
$ ! In the following RMU Unload command, the Record_Definition
$ ! qualifier is used to unload the row with the NULL value, but
$ ! the Null option is not specified:
$ RMU/UNLOAD/RECORD_DEFINITION=(FILE=NULL_DATE,FORMAT=DELIMITED_TEXT) -
_$  MF_PERSONNEL NULL_DATE NULL_DATE
%RMU-I-DATRECUNL, 1 data records unloaded.
$ !
$ ! The null_date.unl file created by the previous unload
$ ! operation does not preserve the NULL value in the DATE1 column.
$ ! Instead, the Oracle Rdb default date value is used:
$ TYPE NULL_DATE.UNL
"first","1858111700000000","last"
$ !
$ ! This time, unload the row in NULL_DATE with the Null option to
$ ! the Record_Definition qualifier:
$ RMU/UNLOAD MF_PERSONNEL NULL_DATE NULL_DATE -
_$ /RECORD_DEFINITION=(FILE=NULL_DATE.RRD, FORMAT=DELIMITED_TEXT, NULL="*")
%RMU-I-DATRECUNL,   1 data records unloaded.
$ !
$ TYPE NULL_DATE.UNL
"first",*,"last "
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
```

```
SQL> --
SQL> -- Delete the existing row from NULL_DATE:
SQL> DELETE FROM NULL_DATE;
1 row deleted
SQL> --
SQL> COMMIT;
SQL> EXIT;
$ !
$ ! Load the row that was unloaded back into the table,
$ ! using the null_date.unl file created by the
$ ! previous RMU Unload command:
$ RMU/LOAD MF_PERSONNEL /RECORD_DEFINITION=(FILE=NULL_DATE.RRD, -
_$ FORMAT=DELIMITED_TEXT, NULL="*") NULL_DATE NULL_DATE
%RMU-I-DATRECREAD,  1 data records read from input file.
%RMU-I-DATRECSTO, 1 data records stored.
$ !
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> --
SQL> -- Display the row stored in NULL_DATE.
SQL> -- The NULL value stored in the data row
SQL> -- was preserved by the load and unload operations:
SQL> SELECT * FROM NULL_DATE;
 COL1    DATE1                     COL2
 first   NULL                      last
1 row selected
```

Example 9

The following example demonstrates the use of the Null="" option of the Record_Definition qualifier to signal to Oracle RMU that any data that is an empty string in the .unl file (as represented by two commas with no space separating them) should have the corresponding column in the database flagged as NULL.

The first part of this example shows the contents of the .unl file and the RMU Load command used to load the .unl file. The terminator for each record in the .unl file is the number sign (#). The second part of this example unloads unloads the data and specifies that any columns that are flagged as NULL should be represented in the output file with an asterisk.

```
"90021","ABUSHAKRA","CAROLINE","A","5 CIRCLE STREET",,
"CHELMSFORD", "MA", "02184", "1960061400000000"#
"90015","BRADFORD","LEO","B","4 PLACE STREET",, "NASHUA","NH",
"03030", "1949051800000000"#
```

1.68 RMU Unload Command

```
$ !
$ RMU/LOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#", -
_$ NULL="") -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECREAD,  2 data records read from input file.
%RMU-I-DATRECSTO,   2 data records stored.
$ !
$ ! Unload this data first without specifying the Null option:
$ RMU/UNLOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#") -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECUNL,   102 data records unloaded.
$ !
$ ! The ADDRESS_DATA_2 field appears as a quoted string:
$ TYPE EMPLOYEES.UNL
   .
   .
   .
"90021","ABUSHAKRA     ","CAROLINE  ","A","5 CIRCLE STREET        ","
            ","CHELMSFORD          ","MA","02184","1960061400000000"#
$ !
$ ! Now unload the data with the Null option specified:
$ RMU/UNLOAD/FIELDS=(EMPLOYEE_ID, LAST_NAME, FIRST_NAME, -
_$ MIDDLE_INITIAL, ADDRESS_DATA_1, ADDRESS_DATA_2,  -
_$ CITY, STATE, POSTAL_CODE, BIRTHDAY) -
_$ /RECORD_DEFINITION=(FILE= EMPLOYEES.RRD, -
_$ FORMAT=DELIMITED_TEXT, -
_$ TERMINATOR="#", -
_$ NULL="*") -
_$ MF_PERSONNEL EMPLOYEES EMPLOYEES.UNL
%RMU-I-DATRECUNL,   102 data records unloaded.
$ !
$ ! The value for ADDRESS_DATA_2 appears as an asterisk:
$ !
$ TYPE EMPLOYEES.UNL
   .
   .
   .
"90021","ABUSHAKRA     ","CAROLINE  ","A","5 CIRCLE STREET       ",*,
"CHELMSFORD          ","MA","02184","1960061400000000"#
```

Example 10

The following example specifies a transaction for the RMU Unload command equivalent to the SQL command SET TRANSACTION READ WRITE WAIT 36 RESERVING table1 FOR SHARED READ;

```
$ RMU/UNLOAD-
    /TRANSACTION_TYPE=(SHARED,ISOLATION=REPEAT,WAIT=36)-
    SAMPLE.RDB-
    TABLE1-
    TABLE.DAT
```

Example 11

The following example specifies the options that were the default transaction style in prior releases.

```
$ RMU/UNLOAD-
    /TRANSACTION_TYPE=(READ_ONLY,ISOLATION_LEVEL=SERIALIZABLE)-
    SAMPLE.RDB-
    TABLE1-
    TABLE1.DAT
```

Example 12

If the database currently has snapshots deferred, it may be more efficient to start a read-write transaction with isolation level read committed. This allows the transaction to start immediately (a read-only transaction may stall), and the selected isolation level keeps row locking to a minimum.

```
$ RMU/UNLOAD-
    /TRANSACTION_TYPE=(SHARED_READ,ISOLATION=READ_COMMITTED)-
    SAMPLE.RDB-
    TABLE1-
    TABLE1.DAT
```

Using a transaction type of automatic adapts to different database settings.

```
$ RMU/UNLOAD-
    /TRANSACTION_TYPE=(AUTOMATIC)-
    SAMPLE.RDB-
    TABLE1-
    TABLE1.DAT
```

Example 13

The following example shows the output from the flags STRATEGY and ITEM_LIST which indicates that the Optimize qualifier specified that sequential access be used, and also that Total_Time is used as the default optimizer preference.

1.68 RMU Unload Command

```
$ DEFINE RDMS$SET_FLAGS "STRATEGY,ITEM_LIST"
$ RMU/UNLOAD/OPTIMIZE=SEQUENTIAL_ACCESS PERSONNEL EMPLOYEES E.DAT
   .
   .
   .
~H Request Information Item List: (len=11)
0000 (00000) RDB$K_SET_REQ_OPT_PREF "0"
0005 (00005) RDB$K_SET_REQ_OPT_SEQ "1"
000A (00010) RDB$K_INFO_END
Get     Retrieval sequentially of relation EMPLOYEES
%RMU-I-DATRECUNL,   100 data records unloaded.
```

Example 14

AUTOMATIC columns are evaluated during INSERT and UPDATE operations for a table; for instance, they may record the timestamp for the last operation. If the table is being reorganized, it may be necessary to unload the data and reload it after the storage map and indexes for the table are re-created, yet the old auditing data must remain the same.

Normally, the RMU Unload command does not unload columns marked as AUTOMATIC; you must use the Virtual_Fields qualifier with the keyword Automatic to request this action.

```
$ rmu/unload/virtual_fields=(automatic) payroll_db people people.unl
```

Following the restructure of the database, the data can be reloaded. If the target columns are also defined as AUTOMATIC, then the RMU Load process will not write to those columns. You must use the Virtual_Fields qualifier with the keyword Automatic to request this action.

```
$ rmu/load/virtual_fields=(automatic) payroll_db people people.unl
```

Example 15

This example shows the action of the Delete_Rows qualifier. First, SQL is used to display the count of the rows in the table. The file PEOPLE.COLUMNS is verified (written to SYS$OUTPUT) by the RMU Unload command.

```
$ define sql$database db$:scratch
$ sql$ select count (*) from people;
```

```
        100
1 row selected
$ rmu/unload/fields="@people.columns" -
    sql$database -
    /record_definition=(file:people,format:delimited) -
    /delete_rows -
    people -
    people2.dat
  EMPLOYEE_ID
  LAST_NAME
  FIRST_NAME
  MIDDLE_INITIAL
  SEX
  BIRTHDAY
%RMU-I-DATRECERA,   100 data records erased.
%RMU-I-DATRECUNL,   100 data records unloaded.
```

A subsequent query shows that the rows have been deleted.

```
$ sql$ select count (*) from people;

          0
1 row selected
```

Example 16

The following example shows the output from the RMU Unload command options for XML support. The two files shown in the example are created by this RMU Unload command:

```
$ rmu/unload -
    /record_def=(format=xml,file=work_status) -
    mf_personnel -
    work_status -
    work_status.xml
```

```
Output WORK_STATUS.DTD file
```

```
<?xml version="1.0"?>
<!-- RMU Unload for Oracle Rdb V7.1-00 -->
<!-- Generated: 16-MAR-2001 22:26:47.30 -->
```

```
<!ELEMENT WORK_STATUS (RMU_ROW*)>
<!ELEMENT RMU_ROW (
 STATUS_CODE,
 STATUS_NAME,
 STATUS_TYPE
)>
<!ELEMENT STATUS_CODE (#PCDATA)>
<!ELEMENT STATUS_NAME (#PCDATA)>
<!ELEMENT STATUS_TYPE (#PCDATA)>
<!ELEMENT NULL (EMPTY)>
```

```
Output WORK_STATUS.XML file
```

1.68 RMU Unload Command

```
<?xml version="1.0"?>
<!-- RMU Unload for Oracle Rdb V7.1-00 -->
<!-- Generated: 16-MAR-2001 22:26:47.85 -->

<!DOCTYPE WORK_STATUS SYSTEM "work_status.dtd">

<WORK_STATUS>
 <RMU_ROW>
  <STATUS_CODE>0</STATUS_CODE>
  <STATUS_NAME>INACTIVE</STATUS_NAME>
 <STATUS_TYPE>RECORD EXPIRED</STATUS_TYPE>
 </RMU_ROW>
 <RMU_ROW>
  <STATUS_CODE>1</STATUS_CODE>
  <STATUS_NAME>ACTIVE  </STATUS_NAME>
  <STATUS_TYPE>FULL TIME     </STATUS_TYPE>
 </RMU_ROW>
 <RMU_ROW>
  <STATUS_CODE>2</STATUS_CODE>
  <STATUS_NAME>ACTIVE  </STATUS_NAME>
  <STATUS_TYPE>PART TIME     </STATUS_TYPE>
 </RMU_ROW>
</WORK_STATUS>

<!-- 3 rows unloaded -->
```

Example 17

The following example shows that if the Flush=On_Commit qualifier is specified, the value for the Commit_Every qualifier must be equal to or a multiple of the Row_Count value so the commits of unload transactions occur after the internal RMS buffers are flushed to the unload file. This prevents loss of data if an error occurs.

```
$RMU/UNLOAD/ROW_COUNT=5/COMMIT_EVERY=2/FLUSH=ON_COMMIT MF_PERSONNEL -
_$ EMPLOYEES EMPLOYEES
%RMU-F-DELROWCOM, For DELETE_ROWS or FLUSH=ON_COMMIT the COMMIT_EVERY value must
 equal or be a multiple of the ROW_COUNT value.
The COMMIT_EVERY value of 2 is not equal to or a multiple of the ROW_COUNT value
 of 5.
%RMU-F-FTL_UNL, Fatal error for UNLOAD operation at 27-Oct-2005 08:55:14.06
```

Example 18

The following examples show that the unload file and record definition files are not deleted on error if the Noerror_Delete qualifier is specified and that these files are deleted on error if the Error_Delete qualifier is specified. If the unload file is empty when the error occurs, it will be deleted.

```
$RMU/UNLOAD/NOERROR_DELETE/ROW_ACOUNT=50/COMMIT_EVERY=50 MF_PERSONNEL -
_$ EMPLOYEES EMPLOYEES.UNL

%RMU-E-OUTFILNOTDEL, Fatal error, the output file is not deleted but may not
be useable,
50 records have been unloaded.
-COSI-F-WRITERR, write error
-RMS-F-FUL, device full  (insufficient space for allocation)

$RMU/UNLOAD/ERROR_DELETE/ROW_COUNT=50/COMMIT_EVERY=50 MF_PERSONNEL -
_$ EMPLOYEES EMPLOYEES.UNL

%RMU-E-OUTFILDEL, Fatal error, output file deleted
-COSI-F-WRITERR, write error
-RMS-F-FUL, device full (insufficient space for allocation)
```

Example 19

The following example shows the FORMAT=CONTROL option. This command creates a file EMP.CTL (the SQL*Loader control file) and EMPLOYEES.DAT in a portable format to be loaded.

```
$ RMU/UNLOAD/RECORD_DEFINITION=(FORMAT=CONTROL,FILE=EMP) -
    SQL$DATABASE -
    EMPLOYEES -
    EMPLOYEES
```

Example 20

The following shows an example of using the COMPRESSION qualifier with the RMU Unload command.

```
$ RMU/UNLOAD/COMPRESS=LZW/DEBUG=TRACE COMPLETE_WORKS COMPLETE_WORKS
COMPLETE_WORKS
Debug = TRACE
Compression = LZW
* Synonyms are not enabled
Unloading Blob columns.
Row_Count = 500
Message buffer: Len: 54524
Message buffer: Sze: 109, Cnt: 500, Use: 31 Flg: 00000000
** compress data: input 2700 output 981 deflate 64%
** compress TEXT_VERSION : input 4454499 output 1892097 deflate 58%
** compress PDF_VERSION : input 274975 output 317560 deflate -15%
%RMU-I-DATRECUNL,   30 data records unloaded.
```

Example 21

The following shows an example of using the COMPRESSION qualifier with RMU Unload and using the EXCLUDE_LIST option to avoid attempting to compress data that does not compress.

1.68 RMU Unload Command

```
$ RMU/UNLAOD/COMPRESS=(LZW,EXCLUDE_LIST:PDF_VERSION)/DEBUG=TRACE COMPLETE_WORKS
COMPLETE_WORKS COMPLETE_WORKS
Debug = TRACE
Compression = LZW
Exclude_List:
       Exclude column PDF_VERSION
* Synonyms are not enabled
Unloading Blob columns.
Row_Count = 500
Message buffer: Len: 54524
Message buffer: Sze: 109, Cnt: 500, Use: 31 Flg: 00000000
** compress data: input 2700 output 981 deflate 64%
** compress TEXT_VERSION : input 4454499 output 1892097 deflate 58%
%RMU-I-DATRECUNL,   30 data records unloaded.
```

Example 22

The following example shows using the RMU Unload command to generate a portable data file (CSV format for Microsoft EXCEL support). The output RRD (record definition) file is suppressed using NOFILE as it is usually not useful for the target tool. The TRIM qualifier is used to remove unnecessary padding spaces.

```
$ rmu/unload-
    /record=(nofile,format=csv,trim=trailing,term=";") -
    sql$database -
    work_status -
    ws.csv
%RMU-I-DATRECUNL,   4 data records unloaded.
$ ty ws.csv
"STATUS_CODE","STATUS_NAME","STATUS_TYPE";
"0","INACTIVE","RECORD EXPIRED";
"1","ACTIVE","FULL TIME";
"2","ACTIVE","PART TIME";
```

Example 23

You can use options to change delimiters in a CSV formatted file. This example changes the delimiters for the data as required by the target loading tool.

```
$ rmu/unload-
    /record=(nofile,format=csv,trim=trailing,-
             term=";",pref="{",suff="}") -
    sql$database -
    current_job -
    cj.csv
 . . .
{LAST_NAME},{FIRST_NAME},{EMPLOYEE_ID},{JOB_CODE},{DEPARTMENT_CODE},
{SUPERVISOR_ID},{JOB_START};
{Toliver},{Alvin},{00164},{DMGR},{MBMN},{00228},{1981-09-21 00:00:00.00};
 . . .
```

Example 24

Using the BITMAPPED_SCAN option for /OPTIMIZE.

```
$       define RDMS$SET_FLAGS "item_list,noprefix,strategy,detail(2)"
$       define RDMS$DEBUG_FLAGS_OUTPUT flags.log
$
$       RMU/UNLOAD-
            /OPTIMIZE=BITMAPPED_SCAN-
            RMU_UNLOAD_BITMAPPED_SCAN_4_DB-
            CURRENT_SALARY-
            CURRENT_SALARY
%RMU-I-DATRECUNL,   100 data records unloaded  5-AUG-2013 12:32:11.11.
$       SEARCH/REMAINING FLAGS.LOG "~H Request"
~H Request Information Item List: (len=11)
RDB$K_SET_REQ_OPT_PREF "0"
RDB$K_SET_REQ_OPT_BITMAPPED "1"
RDB$K_INFO_END
Tables:
  0 = SALARY_HISTORY
  1 = EMPLOYEES
Cross block of 2 entries  Q2
  Cross block entry 1
    Leaf#01 BgrOnly 0:SALARY_HISTORY Card=729        Bitmapped scan
      Bool: MISSING (0.SALARY_END)
      BgrNdx1 SH_EMPLOYEE_SS [1:1] Fan=75
        Keys: MISSING (0.SALARY_END)
  Cross block entry 2
    Get     Retrieval by index of relation 1:EMPLOYEES
      Index name  EMP_EMPLOYEE_ID [1:1]      Direct lookup
        Keys: 1.EMPLOYEE_ID = 0.EMPLOYEE_ID
$
$       deassign RDMS$DEBUG_FLAGS_OUTPUT
$       deassign RDMS$SET_FLAGS
```

Example 25

The following example compares the empty string syntax with the negated keyword usage.

1.68 RMU Unload Command

```
$ rmu/unload-
    /record=(nofile,format=delimited,prefix="",suffix="",null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.43.
$ rmu/unload-
    /record=(nofile,format=delimited,noprefix,nosuffix,null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.56.
$
```

1.69 RMU Unload After_Journal Command

Allows you to extract added, modified, committed, and deleted record contents from committed transactions from specified tables in one or more after-image journal files.

Format

RMU/Unload/After_Journal root-file-spec aij-file-name

Command Qualifiers	Defaults
/Before=date-time	None
/Continuous	/NoContinuous
/Deletes_First	/NoDeletes_First
/Extend_Size=integer	/Extend_Size=1000
/Format=options	See description
/Ignore=Old_Version[=table-list]	/Ignore=Old_Version=all
/Include=Action=(include-type)	Include=Action= (NoCommit,Modify,Delete)
/IO_Buffers=integer	/IO_Buffers=2
/[No]Log	Current DCL verify value
/[No]Mbx_Asynch	/Mbx_Asynch
/Options=options-list	See description
/Order_AIJ_files	/NoOrder_aij_files
/Output=file-spec	/Output=SYS$OUTPUT
/Parameter=character-strings	None
/Quick_Sort_Limit=integer	/Quick_Sort_Limit=5000
/Restart=(restart-point)	None
/Restore_Metadata=file-spec	None
/Save_Metadata=file-spec	None
/Select=selection-type	/Select=Commit_Transaction
/Since=date-time	None
/Sort_Workfiles=integer	/Sort_Workfiles=2
/Statistics_Interval=integer	See description
/[No]Symbols	/Symbols
/Table=(Name=table-name, [table-options ...])	See description None
[No]Trace	/Notrace

1.69 RMU Unload After_Journal Command

Description

The RMU Unload After_Journal command translates the binary data record contents of an after-image journal (.aij) file into an output file. Data records for the specified tables for committed transactions are extracted to an output stream (file, device, or application callback) in the order that the transactions were committed.

Before you use the RMU Unload After_Journal command, you must enable the database for LogMiner extraction. Use the RMU Set Logminer command to enable the LogMiner for Rdb feature for the database. Before you use the RMU Unload After_Journal command with the Continuous qualifier, you must enable the database for Continuous LogMiner extraction. See Section 1.61 for more information.

Data records extracted from the .aij file are those records that transactions added, modified, or deleted in base database tables. Index nodes, database metadata, segmented strings (BLOB), views, COMPUTED BY columns, system relations, and temporary tables cannot be unloaded from after-image journal files.

For each transaction, only the final content of a record is extracted. Multiple changes to a single record within a transaction are condensed so that only the last revision of the record appears in the output stream. You cannot determine which columns were changed in a data record directly from the after-image journal file. In order to determine which columns were changed, you must compare the record in the after-image journal file with a previous record.

The database used to create the after-image journal files being extracted must be available during the RMU Unload After_Journal command execution. The database is used to obtain metadata information (such as table names, column counts, record version, and record compression) needed to extract data records from the .aij file. The database is read solely to load the metadata and is then detached. Database metadata information can also be saved and used in a later session. See the Save_MetaData and Restore_MetaData qualifiers for more information.

If you use the Continuous qualifier, the database must be opened on the node where the Continuous LogMiner process is running. The database is always used and must be available for both metadata information and for access to the online after-image journal files. The Save_MetaData and Restore_MetaData qualifiers are not permitted with the Continuous qualifier.

1.69 RMU Unload After_Journal Command

When one or more .aij files and the Continuous qualifier are both specified on the RMU Unload After_Journal command line, it is important that no .aij backup operations occur until the Continuous LogMiner process has transitioned to online mode (where the active online .aij files are being extracted). If you are using automatic .aij backups and wish to use the Continuous LogMiner feature, Oracle recommends that you consider disabling the automatic backup feature (ABS) and use manual .aij backups so that you can explicitly control when .aij backup operations occur.

The after-image journal file or files are processed sequentially. All specified tables are extracted in one pass through the after-image journal file.

As each transaction commit record is processed, all modified and deleted records for the specified tables are sorted to remove duplicates. The modified and deleted records are then written to the output streams. Transactions that were rolled back are ignored. Data records for tables that are not being extracted are ignored. The actual order of output records within a transaction is not predictable.

In the extracted output, records that were modified or added are returned as being modified. It is not possible to distinguish between inserted and updated records in the output stream. Deleted (erased) records are returned as being deleted. A transaction that modifies and deletes a record generates only a deleted record. A transaction that adds a new record to the database and then deletes it within the same transaction generates only a deleted record.

The LogMiner process signals that a row has been deleted by placing a D in the RDB$LM_ACTION field. The contents of the row at the instant before the delete operation are recorded in the user fields of the output record. If a row was modified several times within a transaction before being deleted, the output record contains only the delete indicator and the results of the last modify operation. If a row is inserted and deleted in the same transaction, only the delete record appears in the output.

Records from multiple tables can be output to the same or to different destination streams. Possible output destination streams include the following:

- File
- OpenVMS Mailbox
- OpenVMS Pipe
- Direct callback to an application through a run-time activated shareable image

Refer to Appendix B for more information about using the LogMiner for Rdb feature.

1.69 RMU Unload After_Journal Command

Command Parameters

root-file-spec
The root file specification of the database for the after-image journal file from which tables will be unloaded. The default file extension is .rdb.

The database must be the same actual database that was used to create the after-image journal files. The database is required so that the table metadata (information about data) is available to the RMU Unload After_Journal command. In particular, the names and relation identification of valid tables within the database are required along with the number of columns in the table and the compression information for the table in various storage areas.

The RMU Unload After_Journal process attaches to the database briefly at the beginning of the extraction operation in order to read the metadata. Once the metadata has been read, the process disconnects from the database for the remainder of the operation unless the Continuous qualifier is specified. The Continuous qualifier indicates that the extraction operation is to run non-stop, and the process remains attached to the database.

aij-file-name
One or more input after-image journal backup files to be used as the source of the extraction operation. Multiple journal files can be extracted by specifying a comma-separated list of file specifications. Oracle RMU supports OpenVMS wildcard specifications (using the * and % characters) to extract a group of files. A file specification beginning with the at (@) character refers to an options file containing a list of after-image journal files (rather than the file specification of an after-image journal itself). If you use the at character syntax, you must enclose the at character and the file name in double quotation marks (for example, specify aij-file-name as "@files.opt"). The default file extension is .aij.

Command Qualifiers

Before=date-time
Specifies the ending time and date for transactions to be extracted. Based on the Select qualifier, transactions that committed or started prior to the specified Before date are selected. Information changed due to transactions that committed or started after the Before date is not included in the output.

Continuous
NoContinuous
Causes the LogMiner process to attach to the database and begin extracting records in "near-real" time. When the Continuous qualifier is specified, the

1.69 RMU Unload After_Journal Command

RMU Unload After_Journal command extracts records from the online after-image journal files of the database until it is stopped via an external source (for example, Ctrl/y, STOP/ID, $FORCEX, or database shutdown).

A database must be explicitly enabled for the Continuous LogMiner feature. To enable the Continuous LogMiner feature, use the RMU Set Logminer command with the Enable and Continuous qualifiers; to disable use of the Continuous LogMiner feature, use the RMU Set Logminer command with the Enable and Nocontinuous qualifiers.

The output from the Continuous LogMiner process is a continuous stream of information. The intended use of the Continuous LogMiner feature is to write the changes into an OpenVMS mailbox or pipe, or to call a user-supplied callback routine. Writing output to a disk file is completely functional with the Continuous LogMiner feature, however, no built-in functionality exists to prevent the files from growing indefinitely.

It is important that the callback routine or processing of the mailbox be very responsive. If the user-supplied callback routine blocks, or if the mailbox is not being read fast enough and fills, the RMU Unload After_Journal command will stall. The Continuous LogMiner process prevents backing up the after-image journal that it is currently extracting along with all subsequent journals. If the Continuous LogMiner process is blocked from executing for long enough, it is possible that all available journals will fill and will not be backed up.

When a database is enabled for the Continuous LogMiner feature, an AIJ "High Water" lock (AIJHWM) is utilized to help coordinate and maintain the current .aij end-of-file location. The lock value block for the AIJHWM lock contains the location of the highest written .aij block. The RMU Unload After_Journal command with the Continuous qualifier polls the AIJHWM lock to determine if data has been written to the .aij file and to find the highest written block. If a database is not enabled for the Continuous LogMiner feature, there is no change in locking behavior; the AIJHWM lock is not maintained and thus the Continuous qualifier of the RMU Unload After_ Journal command is not allowed.

In order to maintain the .aij end-of-file location lock, processes that write to the after-image journal file must use the lock to serialize writing to the journal. When the Continuous LogMiner feature is not enabled, processes instead coordinate allocating space in the after-image journal file and can write to the file without holding a lock. The Continuous LogMiner process requires that the AIJHWM lock be held during the .aij I/O operation. In some cases, this can reduce overall throughput to the .aij file as it serves to reduce multiple over-lapped I/O write operations by multiple processes.

The Save_Metadata and Restore_Metadata qualifiers are incompatible with the Continuous qualifier.

Deletes_First
NoDeletes_First
Specifying Deletes_First indicates that all delete operations within each transaction are to be returned before add or modify operations. The default is NoDeletes_First which leaves the ordering unchanged.

Record Order Unpredictable

Within the output stream for a transaction, the order of records returned from the LogMiner remains unpredictable.

Extend_Size=integer
Specifies the file allocation and extension quantity for output data files. The default extension size is 1000 blocks. Using a larger value can help reduce output file fragmentation and can improve performance when large amounts of data are extracted.

Format=options
If the Format qualifier is not specified, Oracle RMU outputs data to a fixed-length binary flat file.

The format options are:

- Format=Binary

  If you specify the Format=Binary option, Oracle RMU does not perform any data conversion; data is output in a flat file format with all data in the original binary state.

  Table 1–21 describes the output fields and data types of an output record in Binary format.

Table 1–21 Output Fields

Field Name	Data Type	Byte Length	Description
ACTION	CHAR (1)	1	Indicates record state. "M" indicates an insert or modify action. "D" indicates a delete action. "E" indicates stream end-of-file (EOF) when a callback routine is being used. "P" indicates a value from the command line Parameter qualifier when a callback routine is being used (see Parameter qualifier). "C" indicates transaction commit information when the Include=Action=Commit qualifier is specified.
RELATION_NAME	CHAR (31)	31	Table name. Space padded to 31 characters.
RECORD_TYPE	INTEGER (Longword)	4	The Oracle Rdb internal relation identifier.
DATA_LEN	SMALLINT (Word)	2	Length, in bytes, of the data record content.
NBV_LEN	SMALLINT (Word)	2	Length, in bits, of the null bit vector content.
DBK	BIGINT (Quadword)	8	Records logical database key. The database key is a 3-field structure containing a 16-bit line number, a 32-bit page number and a 16-bit area number.
START_TAD	DATE VMS (Quadword)	8	Date/time of the start of the transaction.
COMMIT_TAD	DATE VMS (Quadword)	8	Date/time of the commitment of the transaction.
TSN	BIGINT (Quadword)	8	Transaction sequence number of the transaction that performed the record operation.

(continued on next page)

Table 1–21 (Cont.) Output Fields

Field Name	Data Type	Byte Length	Description
RECORD_VERSION	SMALLINT (Word)	2	Record version.
Record Data	Varies		Actual data record field contents.
Record NBV	BIT VECTOR (array of bits)		Null bit vector. There is one bit for each field in the data record. If a bit value is 1, the corresponding field is NULL; if a bit value is 0, the corresponding field is not NULL and contains an actual data value. The null bit vector begins on a byte boundary. Any extra bits in the final byte of the vector after the final null bit are unused.

- Format=Dump

  If you specify the Format=Dump option, Oracle RMU produces an output format suitable for viewing. Each line of Dump format output contains the column name (including LogMiner prefix columns) and up to 200 bytes of the column data. Unprintable characters are replaced with periods (.), and numbers and dates are converted to text. NULL columns are indicated with the string "NULL". This format is intended to assist in debugging; the actual output contents and formatting will change in the future.

- Format=Text

  If you specify the Format=Text option, Oracle RMU converts all data to printable text in fixed-length columns before unloading it. VARCHAR(n) strings are padded with blanks when the specified string has fewer characters than *n* so that the resulting string is *n* characters long.

- Format=(Delimited_Text [,delimiter-options])

  If you specify the Format=Delimited_Text option, Oracle RMU applies delimiters to all data before unloading it.

  DATE VMS dates are output in the collatable time format, which is yyyymmddhhmmsscc. For example, March 20, 1993 is output as: 1993032000000000.

  Delimiter options are:

  – Prefix=string
    NoPrefix

    Specifies a prefix string that begins any column value in the input file. If you omit this option, the column prefix is assumed to consist of a quotation mark ("). If you specify NoPrefix or an empty string, then no prefix will be used.

  – Separator=string
    NoSeparator

    Specifies a string that separates column values of a row. If you omit this option, the column separator is assumed to consist of a single comma (,). If you specify NoSeparator or an empty string, then no separator will be used.

  – Suffix=string
    NoSuffix

    Specifies a suffix string that ends any column value in the input file. If you omit this option, the column suffix is assumed to consist of a quotation mark ("). If you specify NoSuffix or an empty string, then no suffix will be used.

  – Terminator=string

    Specifies the row terminator that completes all the column values corresponding to a row. If you omit this option, the row terminator is the end of the line.

  – Null=string
    NoNull

Specifies a string that, when found in the database column, is unloaded as "NULL" in the output file.

The Null option can be specified *on the command line* as one of the following:

* A quoted string with an alternate replacement string
* An empty string (`""`) or NoNull to request no replacement string

The string that represents the null character must be quoted on the Oracle RMU command line. You cannot specify a blank space or spaces as the null character. You cannot use the same character for the Null value and other Delimited_Text options.

Note

The values for each of the strings specified in the delimiter options must be enclosed within quotation marks. Oracle RMU strips these quotation marks while interpreting the values. If you want to specify a quotation mark (") as a delimiter, specify a string of four quotation marks. Oracle RMU interprets four quotation marks as your request to use one quotation mark as a delimiter. For example, `Suffix = """"`.

Oracle RMU reads these quotation marks as follows:

- The first quotation mark is stripped from the string.
- The second and third quotation mark are interpreted as your request for one quotation mark (") as a delimiter.
- The fourth quotation mark is stripped.

This results in one quotation mark being used as a delimiter.

Furthermore, if you want to specify a quotation mark as part of the delimited string, you must use two quotation marks for each quotation mark that you want to appear in the string. For example, `Suffix = "**""**"` causes Oracle RMU to use a delimiter of **"**.

Ignore=Old_Version[=table-list]

Specifies optional conditions or items to ignore.

The RMU Unload After_Journal command treats non-current record versions in the AIJ file as a fatal error condition. That is, attempting to extract a record that has a record version not the same as the table's current maximum version results in a fatal error.

There are, however, some very rare cases where a verb rollback of a modification of a record may result in an old version of a record being written to the after-image journal even though the transaction did not actually complete a successful modification to the record. The RMU Unload After_ Journal command detects the old record version and aborts with a fatal error in this unlikely case.

When the Ignore=Old_Version qualifier is present, the RMU Unload After_ Journal command displays a warning message for each record that has a non-current record version and the record is not written to the output stream. The Old_Version qualifier accepts an optional list of table names to indicate that only the specified tables are permitted to have non-current record version errors ignored.

Include=Action=include-type
Specifies if deleted or modified records or transaction commit information is to be extracted from the after-image journal. The following keywords can be specified:

- Commit
  NoCommit

  If you specify Commit, a transaction commit record is written to each output stream as the final record for each transaction. The commit information record is written to output streams after all other records for the transaction have been written. The default is NoCommit.

  Because output streams are created with a default file name of the table being extracted, it is important to specify a unique file name on each occurrence of the output stream. The definition of "unique" is such that when you write to a non-file-oriented output device (such as a pipe or mailbox), you must be certain to specify a specific file name on each output destination. This means that rather than specifying `Output=MBA1234:` for each output stream, you should use `Output=MBA1234:MBX`, or any file name that is the same on all occurrences of MBA1234:.

  Failure to use a specific file name can result in additional, and unexpected, commit records being returned. However, this is generally a restriction only when using a stream-oriented output device (as opposed to a disk file).

  The binary record format is based on the standard LogMiner output format. However, some fields are not used in the commit action record. The binary format and contents of this record are shown in Table 1–22. This record type is written for all output data formats.

Table 1–22 Commit Record Contents

Field	Length (in bytes)	Contents
ACTION	1	"C"
RELATION	31	Zero
RECORD_TYPE	4	Zero
DATA_LEN	2	Length of RM_TID_LEN, AERCP_ LEN, RM_TID, AERCP
NBV_LEN	2	Zero
TID	4	Transaction (Attach) ID
PID	4	Process ID
START_TAD	8	Transaction Start Time/Date
COMMIT_TAD	8	Transaction Commit Time/Date
TSN	8	Transaction ID
RM_TID_LEN	4	Length of the Global TID
AERCP_LEN	4	Length of the AERCP information
RM_TID	RM_TID_LEN	Global TID
AERCP	AERCP_LEN	Restart Control Information
USERNAME	12	USERNAME

When the original transaction took part in a distributed, two-phase transaction, the RM_TID component is the Global transaction manager unique transaction ID. Otherwise, this field contains binary zeroes.

The AIJ Extract Recovery Control Point (AERCP) information is used to uniquely identify this transaction within the scope of the database and after-image journal files. It contains the .aij sequence number, VBN and TSN of the last "Micro Quiet Point", and is used by the Continuous LogMiner process to restart a particular point in the journal sequence.

- Delete
  NoDelete

  If you specify Delete, pre-deletion record contents are extracted from the aij file. If you specify NoDelete, no pre-deletion record contents are extracted. The default is Delete.

- Modify
  NoModify

If you specify Modify, modified or added record contents are extracted from the .aij file. If you specify NoModify, then no modified or added record contents are extracted. The default is Modify.

IO_Buffers=integer
Specifies the number of I/O buffers used for output data files. The default number of buffers is two, which is generally adequate. With sufficiently fast I/O subsystem hardware, additional buffers may improve performance. However, using a larger number of buffers will also consume additional virtual memory and process working set.

Log
Nolog
Specifies that the extraction of the .aij file is be reported to SYS$OUTPUT or the destination specified with the Output qualifier. When activity is logged, the output from the Log qualifier provides the number of transactions committed or rolled back. The default is the setting of the DCL VERIFY flag, which is controlled by the DCL SET VERIFY command.

Mbx_Asynch
NoMbx_Asynch
This qualifier controls behavior of the RMU Logminer (LM) utility and how the LM writes records to the mailbox.

The qualifier /Mbx_Asynch is enabled by default. It provides an asynchronous mailbox write mechanism in order to increase throughput. That is, the LM process does not wait for the read operation to finish before writing more unloaded records.

In some high load situations, it is possible for LM to saturate the I/O buffer causing the entire process to hang, which could cause the AIJ Backup Server (ABS) to stall and block other database users. This will typically occur when an after-image journal switchover is attempted. Searching RMU/SHOW STATISTICS stall logs will show many processes stalled with waiting for AIJ journal control messages.

If users are experiencing stalls with "waiting for AIJ journal control" while running LM, you can restart the LM with the /NoMbx_Asynch qualifier to revert to synchronous mailbox write behavior.

The qualifier /NoMbx_Asynch instructs LM to wait for another process to read that record from the mailbox before posting another record.

1.69 RMU Unload After_Journal Command

Options=options-list
The following options can be specified:

- File=file-spec

  An options file contains a list of tables and output destinations. The options file can be used instead of, or along with, the Table qualifier to specify the tables to be extracted. Each line of the options file must specify a table name prefixed with "Table=". After the table name, the output destination is specified as either "Output=", or "Callback_Module=" and "Callback_Routine=", for example:

  ```
  TABLE=tblname,OUTPUT=outfile
  TABLE=tblname,CALLBACK_MODULE=image,CALLBACK_ROUTINE=routine
  ```

  You can use the Record_Definition=file-spec option from the Table qualifier to create a record definition file for the output data. The default file type is .rrd; the default file name is the name of the table.

  You can use the Table_Definition=file-spec option from the Table qualifier to create a file that contains an SQL statement that creates a table to hold transaction data. The default file type is .sql; the default file name is the name of the table.

  Each option in the Options=File qualifier must be fully specified (no abbreviations are allowed) and followed with an equal sign (=) and a value string. The value string must be followed by a comma or the end of a line. Continuation lines can be specified by using a trailing dash. Comments are indicated by using the exclamation point (!) character.

  You can use the asterisk (*) and the percent sign (%) wildcard characters in the table name specification to select all tables that satisfy the components you specify. The asterisk matches zero or more characters; the percent sign matches a single character.

  For table name specifications that contain wild card characters, if the first character of the string is a pound sign (#), the wildcard specification is changed to a "not matching" comparison. This allows exclusion of tables based on a wildcard specification. The pound sign designation is only evaluated when the table name specification contains an asterisk or percent sign.

  For example, a table name specification of "#FOO%" indicates that all table names that are four characters long and do *not* start with the string "FOO" are to be selected.

- Shared_Read

  Specifies that the input after-image journal backup files are to be opened with an RMS shared locking specification.

- Dump

  Specifies that the contents of an input metadata file are to be formatted and displayed. Typically, this information is used as a debugging tool.

Order_AIJ_Files
NoOrder_AIJ_Files
By default, after-image journal files are processed in the order that they are presented to the RMU Unload After_Journal command. The Order_AIJ_Files qualifier specifies that the input after-image journal files are to be processed in increasing order by sequence number. This can be of benefit when you use wildcard (* or %) processing of a number of input files. The .aij files are each opened, the first block is read (to determine the sequence number), and the files are closed prior to the sorting operation.

Output=file-spec
Redirects the log and trace output (selected with the Log and Trace qualifiers) to the named file. If this qualifier is not specified, the output generated by the Log and Trace qualifiers, which can be voluminous, is displayed to SYS$OUTPUT.

Parameter=character-strings
Specifies one or more character strings that are concatenated together and passed to the callback routine upon startup.

For each table that is associated with a user-supplied callback routine, the callback routine is called with two parameters: the length of the Parameter record and a pointer to the Parameter record. The binary format and contents of this record are shown in Table 1–23.

Table 1–23 Parameter Record Contents

Field	Length (in bytes)	Contents
ACTION	1	"P"
RELATION	31	Relation name
RECORD_TYPE	4	Zero
DATA_LEN	2	Length of parameter string
NBV_LEN	2	Zero
LDBK	8	Zero

(continued on next page)

Table 1–23 (Cont.) Parameter Record Contents

Field	Length (in bytes)	Contents
START_TAD	8	Zero
COMMIT_TAD	8	Zero
TSN	8	Zero
DATA	?	Variable length parameter string content

Quick_Sort_Limit=integer
Specifies the maximum number of records that will be sorted with the in-memory "quick sort" algorithm.

The default value is 5000 records. The minimum value that can be specified is 10 and the maximum value is 100,000.

Larger values specified for the /Quick_Sort_Limit qualifier may reduce sort work file I/O at the expense of additional CPU time and/or memory consumption. A value that is too small may result in additional disk file I/O. In general, the default value should be accepted.

Restart=restart-point
Specifies an AIJ Extract Restart Control Point (AERCP) that indicates the location to begin the extraction. The AERCP indicates the transaction sequence number (TSN) of the last extracted transaction along with a location in the .aij file where a known "Micro-quiet point" exists.

When the Restart qualifier is not specified and no input after-image journal files are specified on the command line, the Continuous LogMiner process starts extracting at the beginning of the earliest modified online after-image journal file.

When formatted for text display, the AERCP structure consists of the six fields (the MBZ field is excluded) displayed as unsigned integers separated by dashes; for example, "1-28-12-7-3202-3202".

Restore_Metadata=file-spec
Specifies that the RMU Unload After_Journal command is to read database metadata information from the specified file. The Database parameter is required but the database itself is not accessed when the Restore_Metadata qualifier is specified. The default file type is .metadata. The Continuous qualifier is not allowed when the Restore_Metadata qualifier is present.

Because the database is not available when the Restore_Metadata qualifier is specified, certain database-specific actions cannot be taken. For example, checks for after-image journaling are disabled. Because the static copy of the metadata information is not updated as database structure and table changes are made, it is important to make sure that the metadata file is saved after database DML operations.

When the Restore_Metadata qualifier is specified, additional checks are made to ensure that the after-image journal files were created using the same database that was used to create the metadata file. These checks provide additional security and help prevent accidental mismatching of files.

Save_Metadata=file-spec
Specifies that the RMU Unload After_Journal command is to write metadata information to the named file. The Continuous, Restore_Metadata, Table, and Options=file qualifiers and the aij-file-name parameter are not allowed when the Save_Metadata qualifier is present. The default file type is .metadata.

Also refer to the RMU Dump Metadata_File Command which can render the metadata file created by this qualifier in a readable format.

Select=selection-type
Specifies if the date and time of the Before and Since qualifiers refer to transaction start time or transaction commit time.

The following options can be specified as the selection-type of the Select qualifier:

- Commit_Transaction

  Specifies that the Before and Since qualifiers select transactions based on the time of the transaction commit.

- Start_Transaction

  Specifies that the Before and Since qualifiers select transactions based on the time of the transaction start.

The default for date selection is Commit_Transaction.

Since=date-time
Specifies the starting time for transactions to be extracted. Depending on the value specified in the Select qualifier, transactions that committed or started on or after the specified Since date are selected. Information from transactions that committed or started prior to the specified Since date is not included in the output.

Sort_Workfiles=integer
Specifies the number of sort work files. The default number of sort work files is two. When large transactions are being extracted, using additional sort work files may improve performance by distributing I/O loads over multiple disk devices. Use the SORTWORKn (where n is a number from 0 to 9) logical names to specify the location of the sort work files.

Statistics_Interval=integer
Specifies that statistics are to be displayed at regular intervals so that you can evaluate the progress of the unload operation.

The displayed statistics include:

- Elapsed time
- CPU time
- Buffered I/O
- Direct I/O
- Page faults
- Number of records unloaded for a table
- Total number of records extracted for all tables

If the Statistics_Interval qualifier is specified, the default interval is 60 seconds. The minimum value is one second. If the unload operation completes successfully before the first time interval has passed, you will receive an informational message on the number of files unloaded. If the unload operation is unsuccessful before the first time interval has passed, you will receive error messages and statistics on the number of records unloaded.

At any time during the unload operation, you can press Ctrl/T to display the current statistics.

Symbols
Nosymbols
Specifies whether DCL symbols are to be created, indicating information about records extracted for each table.

If a large enough number of tables is being unloaded, too many associated symbols are created, and the CLI symbol table space can become exhausted. The error message "LIB-F-INSCLIMEM, insufficient CLI memory" is returned in this case. Specify the Nosymbols qualifier to prevent creation of the symbols.

The default is Symbols, which causes the symbols to be created.

Table=(Name=table-name, table-options)
Specifies the name of a table to be unloaded and an output destination. The table-name must be a table within the database. Views, indexes, and system relations may not be unloaded from the after-image journal file.

The asterisk (*) and the percent sign (%) wildcard characters can be used in the table name specification to select all tables that satisfy the components you specify. The asterisk matches zero or more characters and the percent sign matches a single character.

For table name specifications that contain wild card characters, if the first character of the string is a pound sign (#), the wildcard specification is changed to a "not matching" comparison. This allows exclusion of tables based on a wildcard specification. The pound sign designation is only evaluated when the table name specification contains an asterisk or percent sign.

For exmple, a table name specification of "#FOO%" indicates that all table names that are four characters long and do *not* start with the string "FOO" are to be selected.

The following table-options can be specified with the Table qualifier:

- Callback_Module=image-name, Callback_Routine=routine-name

  The LogMiner process uses the OpenVMS library routine LIB$FIND_IMAGE_SYMBOL to activate the specified shareable image and locate the specified entry point routine name. This routine is called with each extracted record. A final call is made with the Action field set to "E" to indicate the end of the output stream. These options must be specified together.

- Control

Use the Control table option to produce output files that can be used by SQL*Loader to load the extracted data into an Oracle database. This option must be used in conjunction with fixed text format for the data file. The Control table option can be specified on either the command line or in an options file.

- Output=file-spec

  If an Output file specification is present, unloaded records are written to the specified location.

- Record_Definition=file-spec

  The Record_Definition=file-spec option can be used to create a record definition file for the output data. The default file type is .rrd; the default file name is the name of the table.

- Table_Definition=file-spec

  You can use the Table_Definition=file-spec option to create a file that contains an SQL statement that creates a table to hold transaction data. The default file type is .sql; the default file name is the name of the table.

Unlike other qualifiers where only the final occurrence of the qualifier is used by an application, the Table qualifier can be specified multiple times for the RMU Unload After_Journal command. Each occurrence of the Table qualifier must specify a different table.

Trace
NoTrace
Specifies that the unloading of the .aij file be traced. The default is Notrace. When the unload operation is traced, the output from the Trace qualifier identifies transactions in the .aij file by TSNs and describes what Oracle RMU did with each transaction during the unload process. You can specify the Log qualifier with the Trace qualifier.

Usage Notes

- To use the RMU Unload After_Journal command for a database, you must have the RMU$DUMP privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.
- Oracle Rdb after-image journaling protects the integrity of your data by recording all changes made by committed transactions to a database in a sequential log or journal file. Oracle Corporation recommends that you enable after-image journaling to record your database transaction activity

between full backup operations as part of your database restore and recovery strategy. In addition to LogMiner for Rdb, the after-image journal file is used to enable several database performance enhancements such as the fast commit, row cache, and hot standby features.

- When the Continuous qualifier is not specified, you can only extract changed records from a backup copy of the after-image journal files. You create this file using the RMU Backup After_Journal command.

  You cannot extract from an .aij file that has been optimized with the RMU Optimize After_Journal command.

- As part of the extraction process, Oracle RMU sorts extracted journal records to remove duplicate record updates. Because .aij file extraction uses the OpenVMS Sort/Merge Utility (SORT/MERGE) to sort journal records for large transactions, you can improve the efficiency of the sort operation by changing the number and location of the work files used by SORT/MERGE. The number of work files is controlled by the Sort_ Workfiles qualifier of the RMU Unload After_Journal command. The allowed values are 1 through 10 inclusive, with a default value of 2. The location of these work files can be specified with device specifications, using the SORTWORKn logical name (where n is a number from 0 to 9). See the OpenVMS documentation set for more information on using SORT/MERGE. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on using these Oracle Rdb logical names.

- When extracting large transactions, the RMU Unload After_Journal command may create temporary work files. You can redirect the .aij rollforward temporary work files to a different disk and directory location than the current default directory by assigning a different directory to the RDM$BIND_AIJ_WORK_FILE logical name in the LNM$FILE_DEV name table. This can help to alleviate I/O bottlenecks that might occur on the default disk.

- You can specify a search list by defining logicals RDM$BIND_AIJ_WORK_FILEn, with each logical pointing to a different device or directory. The numbers must start with 1 and increase sequentially without any gaps. When an AIJ file cannot be created due to a "device full" error, Oracle Rdb looks for the next device in the search list by translating the next sequential work file logical. If RDM$BIND_AIJ_WORK_FILE is defined, it is used first.

- The RMU Unload After_Journal command can read either a backed up .aij file on disk or a backed up .aij file on tape that is in the Old_File format.

- You can select one or more tables to be extracted from an after-image journal file. All tables specified by the Table qualifier and all those specified in the Options file are combined to produce a single list of output streams. A particular table can be specified only once. Multiple tables can be written to the same output destination by specifying the exact same output stream specification (that is, by using an identical file specification).
- At the completion of the unload operation, RMU creates a number of DCL symbols that contain information about the extraction statistics. For each table extracted, RMU creates the following symbols:
  - RMU$UNLOAD_DELETE_COUNT_tablename
  - RMU$UNLOAD_MODIFY_COUNT_tablename
  - RMU$UNLOAD_OUTPUT_tablename

  The tablename component of the symbol is the name of the table. When multiple tables are extracted in one operation, multiple sets of symbols are created. The value for the symbols RMU$UNLOAD_MODIFY_ COUNT_tablename and RMU$UNLOAD_DELETE_COUNT_tablename is a character string containing the number of records returned for modified and deleted rows. The RMU$UNLOAD_OUTPUT_tablename symbol is a character string indicating the full file specification for the output destination, or the shareable image name and routine name when the output destination is an application callback routine.
- When you use the Callback_Module and Callback_Routine option, you must supply a shareable image with a universal symbol or entry point for the LogMiner process to be able to call your routine. See the OpenVMS documentation discussing the Linker utility for more information about creating shareable images.
- Your Callback_Routine is called once for each output record. The Callback_Routine is passed two parameters:
  - The length of the output record, by longword value
  - A pointer to the record buffer

  The record buffer is a data structure of the same fields and lengths written to an output destination.
- Because the Oracle RMU image is installed as a known image, your shareable image must also be a known image. Use the OpenVMS Install Utility to make your shareable image known. You may wish to establish an exit handler to perform any required cleanup processing at the end of the extraction.

- Segmented string data (BLOB) cannot be extracted using the LogMiner process. Because the segmented string data is related to the base table row by means of a database key, there is no convenient way to determine what data to extract. Additionally, the data type of an extracted column is changed from LIST OF BYTE VARYING to BIGINT. This column contains the DBKEY of the original BLOB data. Therefore, the contents of this column should be considered unreliable. However, the field definition itself is extracted as a quadword integer representing the database key of the original segmented string data. In generated table definition or record definition files, a comment is added indicating that the segmented string data type is not supported by the LogMiner for Rdb feature.
- Records removed from tables using the SQL TRUNCATE TABLE statement are not extracted. The SQL TRUNCATE TABLE statement does not journal each individual data record being removed from the database.
- Records removed from tables using the SQL ALTER DATABASE command with the DROP STORAGE AREA clause and CASCADE keyword are not extracted. Any data deleted by this process is not journalled.
- Records removed by dropping tables using the SQL DROP TABLE statement are not extracted. The SQL DROP TABLE statement does not journal each individual data record being removed from the database.
- Database administrators may disable logging of inserted rows to the after-image journal using the NOLOGGING clause on the CREATE TABLE or CREATE STORAGE MAP statements. Logging for this table remains disabled until the transaction that created the table or storage map is committed. Therefore, any rows inserted into that table during that transaction are not available for extraction.

  The following example shows such a transaction. Attempts to use LogMiner for the table TTT will not include these 100 rows.

```
SQL> create table ttt (a int) nologging;
%RDB-W-META_WARN, metadata successfully updated with the reported warning
-RDMS-W-DATACMIT, unjournaled changes made; database may not be recoverable
SQL> insert into ttt select cast(employee_id as int) from employees;
100 rows inserted
SQL> commit;
```

  In a similar way, when the logical name RDMS$CREATE_LAREA_ NOLOGGING is defined, DML operations are not available for extraction between the time the table is created and when the transaction is committed.

- Tables that use the vertical record partitioning (VRP) feature cannot be extracted using the LogMiner feature.

```
$       RMU/UNLOAD/AFTER/NOLOG -
            RMU_LOGMINER_VRP_TABLE_7_DB -
            BACKUP.AIJ -
            /FORMAT=DUMP -
            /INCLUDE=ACTION=(COMMIT,DELETE,MODIFY) -
            /TABLE=(NAME=EMPLOYEES, -
                    OUTPUT=EMPLOYEES.TXT)
%RMU-F-LMNOVRPSUPP, Table "EMPLOYEES" is vertically partitioned, and therefore
can not be unloaded
%RMU-F-FTL_RMU, Fatal error for RMU operation at  3-MAY-2016 22:45:33.47
```

- In binary format output, VARCHAR fields are not padded with spaces in the output file. The VARCHAR data type is extracted as a 2-byte count field and a fixed-length data field. The 2-byte count field indicates the number of valid characters in the fixed-length data field. Any additional contents in the data field are unpredictable.
- You cannot extract changes to a table when the table definition is changed within an after-image journal file. Data definition language (DDL) changes to a table are not allowed within an .aij file being extracted. All records in an .aij file must be the current record version. If you are going to perform DDL operations on tables that you wish to extract using the LogMiner for Rdb, you should:

  1. Back up your after-image journal files.
  2. Extract the .aij files using the RMU Unload After_Journal command.
  3. Make the DDL changes.

- Do not use the OpenVMS Alpha High Performance Sort/Merge utility (selected by defining the logical name SORTSHR to SYS$SHARE:HYPERSORT) when using the LogMiner feature. HYPERSORT supports only a subset of the library sort routines that LogMiner requires. Make sure that the SORTSHR logical name is not defined to HYPERSORT.
- The metadata information file used by the RMU Unload After_Journal command is in an internal binary format. The contents and format are not documented and are not directly accessible by other utilities. The content and format of the metadata information file is specific to a version of the RMU Unload After_Journal utility. As new versions and updates of Oracle Rdb are released, you will proably have to re-create the metadata information file. The same version of Oracle Rdb must be used to both write and read a metadata information file. The RMU Unload After_Journal command verifies the format and version of the metadata

information file and issues an error message in the case of a version mismatch.

- For debugging purposes, you can format and display the contents of a metadata information file by using the Options=Dump qualifier with the Restore_Metadata qualifier. This dump may be helpful to Oracle Support engineers during problem analysis. The contents and format of the metadata information file are subject to change.
- If you use both the Output and Statistics_Interval qualifiers, the output stream used for the log, trace, and statistics information is flushed to disk (via the RMS $FLUSH service) at each statistics interval. This makes sure that an output file of trace and log information is written to disk periodically.
- Please note, unlike SQL and RMU /UNLOAD, the Oracle Rdb LogMiner (tm) RMU /UNLOAD /AFTER_JOURNAL command outputs fields in the database on-disk field order. This difference can become apparant when a table definition has been modified (for example, when adding a new column to appear in a position other than at the end of the record).

  The following example shows one way that the Oracle Rdb LogMiner RMU /UNLOAD /AFTER_JOURNAL command outputs fields in the database on-disk field order.

```
$!
$ SQL$
    CREATE DATA FILE FOO LOGMINER SUPPORT ENA;

    -- Create table then insert another column between COL1 & COL2.

    CREATE TABLE T1 (COL1 INT, COL2 INT);
    ALTER TABLE T1 ADD COLUMN COL3 INT BEFORE COLUMN COL2;
    SHOW TABLE (COLUMN) T1;
Information for table T1

Columns for table T1:
Column Name                    Data Type        Domain
-----------                    ---------        ------
COL1                           INTEGER
COL3                           INTEGER
COL2                           INTEGER

    COMMIT;
```

1.69 RMU Unload After_Journal Command

```
    DISCONNECT ALL;
    ALTER DATA FILE FOO JOU ENA ADD JOU J1 FILE J1;
    EXIT;
$!
$ RMU/BACKUP/NOLOG FOO NLA0:FOO
$ RMU/BACKUP/AFTER/NOLOG FOO NLA0:B1
$!
$ SQL$
    ATTACH 'FILE FOO';
    INSERT INTO T1 VALUES (NULL,2,3); -- COL1=NULL, COL3=2, COL2=3
1 row inserted
    INSERT INTO T1 VALUES (1,NULL,3); -- COL1=1, COL3=NULL, COL2=3
1 row inserted
    INSERT INTO T1 VALUES (1,2,NULL); -- COL1=1, COL3=2, COL2=NULL
1 row inserted
    COMMIT;
   SELECT * FROM T1; -- Show data content
        COL1          COL3          COL2
        NULL             2             3
           1          NULL             3
           1             2          NULL
3 rows selected
    EXIT;
$!
$ RMU/BACKUP/AFTER/NOLOG FOO B1
$!
$ RMU/UNL/RECORD=FILE=T1.RRD1 FOO T1 NLA0:T1
%RMU-I-DATRECUNL,   3 data records unloaded.
$ RMU/UNL/AFTER/NOLOG/FORMAT=DUMP FOO B1 -
    /TABLE=(NAME=T1,OUTPUT=T1.DAT,RECORD=T1.RRD2)
$!
$ SEARCH T1.RRD1 COL ! Show field order - RMU/UNLOAD
DEFINE FIELD COL1 DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD COL3 DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD COL2 DATATYPE IS SIGNED LONGWORD.
   COL1 .
   COL3 .
   COL2 .
$ SEARCH T1.RRD2 COL ! Show field order - RMU/UNLOAD/AFTER_JOURNAL
DEFINE FIELD COL1 DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD COL2 DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD COL3 DATATYPE IS SIGNED LONGWORD.
   COL1.
   COL2.
   COL3.
$ SEARCH T1.DAT RDB$LM_RELATION_NAME, COL ! Show data content
RDB$LM_RELATION_NAME            : T1
COL1                            : NULL
COL2                            : 3
COL3                            : 2
RDB$LM_RELATION_NAME            : T1
COL1                            : 1
```

```
COL2                          : 3
COL3                          : NULL
RDB$LM_RELATION_NAME          : T1
COL1                          : 1
COL2                          : NULL
COL3                          : 2
$!
```

- The /RESTART qualifier is allowed with or without the CONTINUOUS qualifier.

  In addition, when the /RESTART qualifier is specified, it is not enforced that the first AIJ backup file supplied be from after a quiet-point AIJ backup. This implies that it is possible that the transaction returned with the specified AERCP (AIJ Extract Restart Control Point) could be an incomplete transaction (where not every modification made by the transaction is returned in the output data stream). Applications that use /RESTART are expected to understand this behavior and to ignore the transaction with the restarted AERCP.

- RMU Unload After_Journal Format=Control generates definitions and data that are acceptable to Oracle Database release 10g or later and allow these complex types to be shipped in a portable data format and used by SQL*Loader (sqlldr).

  Data transformations include:

  - DATE VMS is unloaded as TIMESTAMP(2). This allows fractional second values (100th of a second) to be unloaded.
  - INTERVAL YEAR and INTERVAL MONTH types are unloaded as INTERVAL YEAR(9) TO MONTH.
  - INTERVAL DAY, INTERVAL HOUR, INTERVAL MINUTE, INTERVAL SECOND, INTERVAL DAY TO HOUR, INTERVAL DAY TO MINUTE, INTERVAL HOUR TO MINUTE, INTERVAL HOUR TO SECOND, and INTERVAL MINUTE TO SECOND are unloaded as INTERVAL DAY(9) TO SECOND(2).

Usage Notes for the Continuous LogMiner Feature

- You can specify input backup after-image journal files along with the Continuous qualifier from the command line. The specified after-image journal backup files are processed in an offline mode. Once they have been processed, the RMU Unload After_Journal command switches to “online” mode and the active online journals are processed.

- When no input after-image journal files are specified on the command line, the Continuous LogMiner starts extracting at the beginning of the earliest modified online after-image journal file. The Restart= qualifier can be used to control the first transaction to be extracted.
- The Continuous LogMiner requires fixed-size circular after-image journals.
- An after-image journal file cannot be backed up if there are any Continuous LogMiner checkpoints in the aij file. The Continuous LogMiner moves its checkpoint to the physical end-of-file for the online .aij file that it is extracting.
- In order to ensure that all records have been written by all database users, Continuous LogMiner processes do not switch to the next live journal file until it has been written to by another process. Live journals SHOULD NOT be backed up while the Continuous LogMiner process is processing a list of .aij backup files. This is an unsupported activity and could lead to the LogMiner losing data.
- If backed up after-image journal files are specified on the command line and the Continuous qualifier is specified, the journal sequence numbers must ascend directly from the backed up journal files to the online journal files.

  In order to preserve the after-image journal file sequencing as processed by the RMU Unload After_Journal /Continuous command, it is important that no after-image journal backup operations are attempted between the start of the command and when the Continuous LogMiner process reaches the live online after-image journals.
- You can run multiple Continuous LogMiner processes at one time on a database. Each Continuous LogMiner process acts independently.
- The Continuous LogMiner reads the live after-image journal file just behind writers to the journal. This will likely increase the I/O load on the disk devices where the journals are located. The Continuous LogMiner attempts to minimize unneeded journal I/O by checking a “High Water Mark” lock to determine if the journal has been written to and where the highest written block location is located.
- Vertically partitioned tables cannot be extracted.

Examples

Example 1

The following example unloads the EMPLOYEES table from the .aij backup file MFP.AIJBCK.

```
RMU /UNLOAD /AFTER_JOURNAL MFP.RDB MFP.AIJBCK -
    /TABLE = (NAME = EMPLOYEES, OUTPUT = EMPLOYEES.DAT)
```

1.69 RMU Unload After_Journal Command

Example 2

The following example simultaneously unloads the SALES, STOCK, SHIPPING, and ORDERS tables from the .aij backup files MFS.AIJBCK_1-JUL-1999 through MFS.AIJBCK_3-JUL-1999. Note that the input .aij backup files are processed sequentially in the order specified.

```
$ RMU /UNLOAD /AFTER_JOURNAL MFS.RDB -
   MFS.AIJBCK_1-JUL-1999, -
   MFS.AIJBCK_2-JUL-1999, -
   MFS.AIJBCK_3-JUL-1999 -
   /TABLE = (NAME = SALES, OUTPUT = SALES.DAT) -
   /TABLE = (NAME = STOCK, OUTPUT = STOCK.DAT) -
   /TABLE = (NAME = SHIPPING, OUTPUT = SHIPPING.DAT) -
   /TABLE = (NAME = ORDER, OUTPUT = ORDER.DAT)
```

Example 3

Use the Before and Since qualifiers to unload data based on a time range. The following example extracts changes made to the PLANETS table by transactions that committed between 1-SEP-1999 at 14:30 and 3-SEP-1999 at 16:00.

```
$ RMU /UNLOAD /AFTER_JOURNAL MFS.RDB MFS.AIJBCK -
   /TABLE = (NAME = PLANETS, OUTPUT = PLANETS.DAT) -
   /BEFORE = "3-SEP-1999 16:00:00.00" -
   /SINCE = "1-SEP-1999 14:30:00.00"
```

Example 4

The following example simultaneously unloads the SALES and STOCK tables from all .aij backup files that match the wildcard specification MFS.AIJBCK_1999-07-*. The input .aij backup files are processed sequentially in the order returned from the file system.

```
$ RMU /UNLOAD /AFTER_JOURNAL MFS.RDB -
   MFS.AIJBCK_1999-07-* -
   /TABLE = (NAME = SALES, OUTPUT = SALES.DAT) -
   /TABLE = (NAME = STOCK, OUTPUT = STOCK.DAT)
```

Example 5

The following example unloads the TICKER table from the .aij backup files listed in the file called AIJ_BACKUP_FILES.DAT (note the double quotation marks surrounding the at (@) character and the file specification). The input .aij backup files are processed sequentially. The output records are written to the mailbox device called MBA127:. A separate program is already running on the system, and it reads and processes the data written to the mailbox.

```
$ RMU /UNLOAD /AFTER_JOURNAL MFS.RDB -
   "@AIJ_BACKUP_FILES.DAT" -
   /TABLE = (NAME = TICKER, OUTPUT = MBA127:)
```

Example 6

You can use the RMU Unload After_Journal command followed by RMU Load commands to move transaction data from one database into a change table in another database. You must create a record definition (.rrd) file for each table being loaded into the target database. The record definition files can be created by specifying the Record_Definition option on the Table qualifier.

```
$ RMU /UNLOAD /AFTER_JOURNAL OLTP.RDB MYAIJ.AIJBCK -
  /TABLE = ( NAME = MYTBL, -
             OUTPUT = MYTBL.DAT, -
             RECORD_DEFINITION=MYLOGTBL) -
  /TABLE = ( NAME = SALE, -
             OUTPUT=SALE.DAT, -
             RECORD_DEFINITION=SALELOGTBL)

$ RMU /LOAD WAREHOUSE.RDB MYLOGTBL MYTBL.DAT -
   /RECORD_DEFINITION = FILE = MYLOGTBL.RRD

$ RMU /LOAD WAREHOUSE.RDB SALELOGTBL SALE.DAT -
   /RECORD_DEFINITION = FILE = SALELOGTBL.RRD
```

Example 7

You can use an RMS file containing the record structure definition for the output file as an input file to the RMU Load command. The record description uses the CDO record and field definition format. This is the same format used by the RMU Load and RMU Unload commands when the Record_Definition qualifier is used. The default file extension is .rrd.

The record definitions for the fields that the LogMiner processs writes to the output .rrd file are shown in the following table. These fields can be manually appended to a record definition file for the actual user data fields being unloaded. The file can be used to load a **transaction table** within a database. A transaction table is the output that the LogMiner process writes to a table consisting of sequential transactions performed in a database.

```
DEFINE FIELD RDB$LM_ACTION          DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD RDB$LM_RELATION_NAME   DATATYPE IS TEXT SIZE IS 31.
DEFINE FIELD RDB$LM_RECORD_TYPE     DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD RDB$LM_DATA_LEN        DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_NBV_LEN         DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_DBK             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_START_TAD       DATETYPE IS DATE
DEFINE FIELD RDB$LM_COMMIT_TAD      DATATYPE IS DATE
DEFINE FIELD RDB$LM_TSN             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_RECORD_VERSION  DATATYPE IS SIGNED WORD.
```

Example 8

Instead of using the Table qualifier, you can use an Options file to specify the table or tables to be extracted, as shown in the following example.

```
$ TYPE TABLES.OPTIONS
TABLE=MYTBL, OUTPUT=MYTBL.DAT
TABLE=SALES, OUTPUT=SALES.DAT
$ RMU /UNLOAD /AFTER_JOURNAL OLTP.RDB MYAIJ.AIJBCK -
   /OPTIONS = FILE = TABLES.OPTIONS
```

Example 9

The following example unloads the EMPLOYEES table from the live database and writes all change records to the MBA145 device. A separate program is presumed to be reading the mailbox at all times and processing the records.

```
$ RMU /UNLOAD /AFTER_JOURNAL /CONTINUOUS MFP.RDB -
 /TABLE = (NAME = EMPLOYEES, OUTPUT = MBA145:)
```

Example 10

This example demonstrates unloading three tables (EMPLOYEES, SALES, and CUSTOMERS) to a single mailbox. Even though the mailbox is not a file-oriented device, the same file name is specified for each. This is required because the LogMiner process defaults the file name to the table name. If the same file name is not explicitly specified for each output stream destination, the LogMiner process assigns one mailbox channel for each table. When the file name is the same for all tables, the LogMiner process detects this and assigns only a single channel for all input tables.

```
$ DEFINE MBX$ LOADER_MBX:X
$ RMU /UNLOAD /AFTER_JOURNAL /CONTINUOUS MFP.RDB -
 /TABLE = (NAME = EMPLOYEES, OUTPUT = MBX$:) -
 /TABLE = (NAME = SALES, OUTPUT = MBX$:) -
 /TABLE = (NAME = CUSTOMERS, OUTPUT = MBX$:)
```

Example 11

In order to include transaction commit information, the /Include =Action =Commit qualifier is specified in this example. Additionally, the EMPLOYEES and SALES tables are extracted to two different mailbox devices (ready by separate processes). A commit record is written to each mailbox after all changed records for each transaction have been extracted.

```
$ RMU /UNLOAD /AFTER_JOURNAL /CONTINUOUS MFP.RDB -
 /INCLUDE = ACTION = COMMIT -
 /TABLE = (NAME = EMPLOYEES, OUTPUT = LOADER_EMP_MBX:X) -
 /TABLE = (NAME = SALES, OUTPUT = LOADER_SAL_MBX:X)
```

Example 12

In this example, multiple input backup after-image journal files are supplied. The Order_AIJ_Files qualifier specifies that the .aij files are to be processed in ascending order of .aij sequence number (regardless of file name). Prior to the extraction operation, each input file is opened and the .aij Open record is read. The .aij files are then opened and extracted, one at a time, by ascending .aij sequence number.

```
$ RMU /UNLOAD /AFTER_JOURNAL /LOG /ORDER_AIJ_FILES -
 MFP.RDB *.AIJBCK -
 /TABLE = (NAME = C1, OUTPUT=C1.DAT)
%RMU-I-UNLAIJFL, Unloading table C1 to DGA0:[DB]C1.DAT;1
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]ABLE.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "5"
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]BAKER.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "4"
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]CHARLIE.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "6"
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]BAKER.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "4"
%RMU-I-AIJMODSEQ, next AIJ file sequence number will be 5
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]ABLE.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "5"
%RMU-I-AIJMODSEQ, next AIJ file sequence number will be 6
%RMU-I-LOGOPNAIJ, opened journal file DGA0:[DB]CHARLIE.AIJBCK;1
%RMU-I-AIJRSTSEQ, journal sequence number is "6"
%RMU-I-AIJMODSEQ, next AIJ file sequence number will be 7
%RMU-I-LOGSUMMARY, total 7 transactions committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
------------------------------------------------------------------------
ELAPSED: 0 00:00:00.15 CPU: 0:00:00.08 BUFIO: 62 DIRIO: 19 FAULTS: 73
Table "C1" : 3 records written (3 modify, 0 delete)
Total : 3 records written (3 modify, 0 delete)
```

Example 13

The SQL record definitions for the fields that the LogMiner process writes to the output are shown in the following example. These fields can be manually appended to the table creation command for the actual user data fields being unloaded. Alternately, the Table_Definition qualifier can be used with the Table qualifier or within an Options file to automatically create the SQL definition file. This can be used to create a transaction table of changed data.

1.69 RMU Unload After_Journal Command

```
SQL> CREATE TABLE MYLOGTABLE (
cont> RDB$LM_ACTION          CHAR,
cont> RDB$LM_RELATION_NAME   CHAR (31),
cont> RDB$LM_RECORD_TYPE     INTEGER,
cont> RDB$LM_DATA_LEN        SMALLINT,
cont> RDB$LM_NBV_LEN         SMALLINT,
cont> RDB$LM_DBK             BIGINT,
cont> RDB$LM_START_TAD       DATE VMS,
cont> RDB$LM_COMMIT_TAD      DATE VMS,
cont> RDB$LM_TSN             BIGINT,
cont> RDB$LM_RECORD_VERSION  SMALLINT ...);
```

Example 14

The following example is the transaction table record definition (.rrd) file for the EMPLOYEES table from the PERSONNEL database:

```
DEFINE FIELD RDB$LM_ACTION          DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD RDB$LM_RELATION_NAME   DATATYPE IS TEXT SIZE IS 31.
DEFINE FIELD RDB$LM_RECORD_TYPE     DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD RDB$LM_DATA_LEN        DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_NBV_LEN         DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_DBK             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_START_TAD       DATATYPE IS DATE.
DEFINE FIELD RDB$LM_COMMIT_TAD      DATATYPE IS DATE.
DEFINE FIELD RDB$LM_TSN             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_RECORD_VERSION  DATATYPE IS SIGNED WORD.

DEFINE FIELD EMPLOYEE_ID            DATATYPE IS TEXT SIZE IS 5.
DEFINE FIELD LAST_NAME              DATATYPE IS TEXT SIZE IS 14.
DEFINE FIELD FIRST_NAME             DATATYPE IS TEXT SIZE IS 10.
DEFINE FIELD MIDDLE_INITIAL         DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD ADDRESS_DATA_1         DATATYPE IS TEXT SIZE IS 25.
DEFINE FIELD ADDRESS_DATA_2         DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD CITY                   DATATYPE IS TEXT SIZE IS 20.
DEFINE FIELD STATE                  DATATYPE IS TEXT SIZE IS 2.
DEFINE FIELD POSTAL_CODE            DATATYPE IS TEXT SIZE IS 5.
DEFINE FIELD SEX                    DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD BIRTHDAY               DATATYPE IS DATE.
DEFINE FIELD STATUS_CODE            DATATYPE IS TEXT SIZE IS 1.
```

```
DEFINE RECORD EMPLOYEES.
   RDB$LM_ACTION .
   RDB$LM_RELATION_NAME .
   RDB$LM_RECORD_TYPE .
   RDB$LM_DATA_LEN .
   RDB$LM_NBV_LEN .
   RDB$LM_DBK .
   RDB$LM_START_TAD .
   RDB$LM_COMMIT_TAD .
   RDB$LM_TSN .
   RDB$LM_RECORD_VERSION .
   EMPLOYEE_ID .
   LAST_NAME .
   FIRST_NAME .
   MIDDLE_INITIAL .
   ADDRESS_DATA_1 .
   ADDRESS_DATA_2 .
   CITY .
   STATE .
   POSTAL_CODE .
   SEX .
   BIRTHDAY .
   STATUS_CODE .
END EMPLOYEES RECORD.
```

Example 15

The following C source code segment demonstrates the structure of a module that can be used as a callback module and routine to process employee transaction information from the LogMiner process. The routine, Employees_ Callback, would be called by the LogMiner process for each extracted record. The final time the callback routine is called, the RDB$LM_ACTION field will be set to "E" to indicate the end of the output stream.

```
#include <stdio>
typedef unsigned char date_type[8];
typedef unsigned char dbkey_type[8];
typedef unsigned char tsn_type[8];
```

```
typedef struct {
    unsigned char       rdb$lm_action;
    char                rdb$lm_relation_name[31];
    unsigned int        rdb$lm_record_type;
    unsigned short int  rdb$lm_data_len;
    unsigned short int  rdb$lm_nbv_len;
    dbkey_type          rdb$lm_dbk;
    date_type           rdb$lm_start_tad;
    date_type           rdb$lm_commit_tad;
    tsn_type            rdb$lm_tsn;
    unsigned short int  rdb$lm_record_version;
    char                employee_id[5];
    char                last_name[14];
    char                first_name[10];
    char                middle_initial[1];
    char                address_data_1[25];
    char                address_data_2[20];
    char                city[20];
    char                state[2];
    char                postal_code[5];
    char                sex[1];
    date_type           birthday;
    char                status_code[1];
} transaction_data;

void employees_callback (unsigned int data_len, transaction_data
                         data_buf)
{    .
     .
     .
 return;}
```

Use the C compiler to compile this module. When linking this module, the symbol EMPLOYEES_CALLBACK needs to be externalized in the shareable image. Refer to the OpenVMS manual discussing the Linker utility for more information about creating shareable images.

You can use a LINK command similar to the following:

```
$ LINK /SHAREABLE = EXAMPLE.EXE EXAMPLE.OBJ + SYS$INPUT: /OPTIONS
SYMBOL_VECTOR = (EMPLOYEES_CALLBACK = PROCEDURE)
<Ctrl/Z>
```

Example 16

You can use triggers and a transaction table to construct a method to replicate table data from one database to another using RMU Unload After_Journal and RMU Load commands. This data replication method is based on transactional changes to the source table and requires no programming. Instead, existing features of Oracle Rdb can be combined to provide this functionality.

For this example, consider a simple customer information table called CUST with a unique customer ID value, customer name, address, and postal code. Changes to this table are to be moved from an OLTP database to a reporting database system on a periodic (perhaps nightly) basis.

First, in the reporting database, a customer table of the same structure as the OLTP customer table is created. In this example, this table is called RPT_CUST. It contains the same fields as the OLTP customer table called CUST.

```
SQL> CREATE TABLE RPT_CUST
cont> CUST_ID               INTEGER,
cont> CUST_NAME             CHAR (50),
cont> CUST_ADDRESS          CHAR (50),
cont> CUST_POSTAL_CODE      INTEGER);
```

Next, a temporary table is created in the reporting database for the LogMiner-extracted transaction data from the CUST table. This temporary table definition specifies ON COMMIT DELETE ROWS so that data in the temporary table is deleted from memory at each transaction commit. A temporary table is used because there is no need to journal changes to the table.

```
SQL> CREATE GLOBAL TEMPORARY TABLE RDB_LM_RPT_CUST (
cont> RDB$LM_RECORD_TYPE    INTEGER,
cont> RDB$LM_DATA_LEN       SMALLINT,
cont> RDB$LM_NBV_LEN        SMALLINT,
cont> RDB$LM_DBK            BIGINT,
cont> RDB$LM_START_TAD      DATE VMS,
cont> RDB$LM_COMMIT_TAD     DATE VMS,
cont> RDB$LM_TSN            BIGINT,
cont> RDB$LM_RECORD_VERSION SMALLINT,
cont> CUST_ID               INTEGER,
cont> CUST_NAME             CHAR (50),
cont> CUST_ADDRESS          CHAR (50),
cont> CUST_POSTAL_CODE      INTEGER) ON COMMIT DELETE ROWS;
```

For data to be populated in the RPT_CUST table in the reporting database, a trigger is created for the RDB_LM_RPT_CUST transaction table. This trigger is used to insert, update, or delete rows in the RPT_CUST table based on the transaction information from the OLTP database for the CUST table.

1.69 RMU Unload After_Journal Command

The unique CUST_ID field is used to determine if customer records are to be modified or added.

```
SQL> CREATE TRIGGER RDB_LM_RPT_CUST_TRIG
cont>  AFTER INSERT ON RDB_LM_RPT_CUST
cont>
cont> -- Modify an existing customer record
cont>
cont>  WHEN (RDB$LM_ACTION = 'M' AND
cont>        EXISTS (SELECT RPT_CUST.CUST_ID FROM RPT_CUST
cont>                WHERE RPT_CUST.CUST_ID =
cont>                RDB_LM_RPT_CUST.CUST_ID))
cont>      (UPDATE RPT_CUST SET
cont>              RPT_CUST.CUST_NAME = RDB_LM_RPT_CUST.CUST_NAME,
cont>              RPT_CUST.CUST_ADDRESS =
cont>              RDB_LM_RPT_CUST.CUST_ADDRESS,
cont>              RPT_CUST.CUST_POSTAL_CODE =
cont>              RDB_LM_RPT_CUST.CUST_POSTAL_CODE
cont>       WHERE RPT_CUST.CUST_ID = RDB_LM_RPT_CUST.CUST_ID)
cont>  FOR EACH ROW
cont>
cont> -- Add a new customer record
cont>
cont>  WHEN (RDB$LM_ACTION = 'M' AND NOT
cont>        EXISTS (SELECT RPT_CUST.CUST_ID FROM RPT_CUST
cont>                WHERE RPT_CUST.CUST_ID =
cont>                RDB_LM_RPT_CUST.CUST_ID))
cont>      (INSERT INTO RPT_CUST VALUES
cont>              (RDB_LM_RPT_CUST.CUST_ID,
cont>               RDB_LM_RPT_CUST.CUST_NAME,
cont>               RDB_LM_RPT_CUST.CUST_ADDRESS,
cont>               RDB_LM_RPT_CUST.CUST_POSTAL_CODE))
cont>  FOR EACH ROW
cont>
cont> -- Delete an existing customer record
cont>
cont>  WHEN (RDB$LM_ACTION = 'D')
cont>      (DELETE FROM RPT_CUST
cont>       WHERE RPT_CUST.CUST_ID = RDB_LM_RPT_CUST.CUST_ID)
cont>  FOR EACH ROW;
```

Within the trigger, the action to take (for example, to add, update, or delete a customer record) is based on the RDB$LM_ACTION field (defined as D or M) and the existence of the customer record in the reporting database. For modifications, if the customer record does not exist, it is added; if it does exist, it is updated. For a deletion on the OLTP database, the customer record is deleted from the reporting database.

1.69 RMU Unload After_Journal Command

The RMU Load command is used to read the output from the LogMiner process and load the data into the temporary table where each insert causes the trigger to execute. The Commit_Every qualifier is used to avoid filling memory with the customer records in the temporary table because as soon as the trigger executes, the record in the temporary table is no longer needed.

```
$ RMU /UNLOAD /AFTER_JOURNAL OLTP.RDB OLTP.AIJBCK -
 /TABLE = (NAME = CUST, -
           OUTPUT = CUST.DAT, -
           RECORD_DEFINITION = RDB_LM_RPT_CUST.RRD)

$ RMU /LOAD REPORT_DATABASE.RDB RDB_LM_RPT_CUST CUST.DAT -
      /RECORD_DEFINITION = FILE = RDB_LM_RPT_CUST.RRD -
      /COMMIT_EVERY = 1000
```

Example 17

The following example shows how to produce a control file that can be used by SQL*Loader to load the extracted data into an Oracle database.

```
$ RMU/UNLOAD/AFTER TEST_DB TEST_DB_AIJ1_BCK -
     /FORMAT=TEXT -
     /TABLE=(NAME=TEST_TBL, -
             OUTPUT=LOGMINER_TEXT.TXT, -
             CONTROL=LOGMINER_CONTROL.CTL, -
             TABLE_DEFINITION=TEST_TBL.SQL)
```

This example produces the following control file. The control file is specific to a fixed length record text file. NULLs are handled by using the NULLIF clause for the column definition that references a corresponding null byte filler column. There is a null byte filler column for each column in the underlying table but not for the LogMiner specific columns at the beginning of the record. If a column is NULL, the corresponding RDB$LM_NBn filler column is set to 1. VARCHAR columns are padded with blanks but the blanks are ignored by default when the file is loaded by SQL*Loader. If you wish to preserve the blanks, you can update the control file and add the "PRESERVE BLANKS" clause.

1.69 RMU Unload After_Journal Command

```
--
-- SQL*Loader Control File
--   Generated by: RMU/UNLOAD/AFTER_JOURNAL (LogMiner)
--   Version:      Oracle Rdb V7.3-120
--   On:           10-JUL-2015 17:10:32.90
--   For table:   "TEST_TBL"
--
LOAD DATA
INFILE 'USER2:[TESTING]LOGMINER_TEXT.TXT;'
APPEND INTO TABLE "RDB_LM_TEST_TBL"
(
-- LogMiner Header fields
 RDB$LM_ACTION                   POSITION(1:1) CHAR
,RDB$LM_RELATION_NAME            POSITION(2:32) CHAR
,RDB$LM_RECORD_TYPE              POSITION(33:44) INTEGER EXTERNAL
,RDB$LM_DATA_LEN                 POSITION(45:50) INTEGER EXTERNAL
,RDB$LM_NBV_LEN                  POSITION(51:56) INTEGER EXTERNAL
,RDB$LM_DBK                      POSITION(57:76) INTEGER EXTERNAL
,RDB$LM_START_TAD                POSITION(77:92) TIMESTAMP "YYYYMMDDHH24MISSFF"
,RDB$LM_COMMIT_TAD               POSITION(93:108) TIMESTAMP "YYYYMMDDHH24MISSFF"
,RDB$LM_TSN                      POSITION(109:128) INTEGER EXTERNAL
,RDB$LM_RECORD_VERSION           POSITION(129:134) INTEGER EXTERNAL
-- Column Definitions
,TEST_COL                        POSITION(135:154) CHAR NULLIF (RDB$LM_NB1 =
'1')
-- NULL indicators
,RDB$LM_NB1               FILLER POSITION(155:155) CHAR
)
```

Example 18

The following example creates a metadata file for the database MFP. This metadata file can be used as input to a later RMU Unload After_Journal command.

```
$ RMU /UNLOAD /AFTER_JOURNAL MFP /SAVE_METADATA=MF_MFP.METADATA /LOG
%RMU-I-LMMFWRTCNT, Wrote 107 objects to metadata file
 "DUA0:[DB]MFMFP.METADATA;1"
```

Example 19

This example uses a previously created metadata information file for the database MFP. The database is not accessed during the unload operation; the database metadata information is read from the file. As the extract operation no longer directly relies on the source database, the AIJ and METADATA files can be moved to another systems and extracted there.

```
$ RMU /UNLOAD /AFTER_JOURNAL /RESTORE_METADATA=MF_MFP.METADATA -
    MFP AIJ_BACKUP1 /TABLE=(NAME=TAB1, OUTPUT=TAB1) /LOG
%RMU-I-LMMFRDCNT, Read 107 objects from metadata file
 "DUA0:[DB]MF_MFP.METADATA;1"
%RMU-I-UNLAIJFL, Unloading table TAB1 to DUA0:[DB]TAB1.DAT;1
%RMU-I-LOGOPNAIJ, opened journal file DUA0:[DB]AIJ_BACKUP1.AIJ;1
%RMU-I-AIJRSTSEQ, journal sequence number is "7216321"
%RMU-I-AIJMODSEQ, next AIJ file sequence number will be 7216322
%RMU-I-LOGSUMMARY, total 2 transactions committed
%RMU-I-LOGSUMMARY, total 0 transactions rolled back
------------------------------------------------------------------------
 ELAPSED:  0 00:00:00.15 CPU: 0:00:00.01 BUFIO: 11 DIRIO: 5 FAULTS: 28
Table "TAB1" : 1 record written (1 modify, 0 delete)
Total : 1 record written (1 modify, 0 delete)
```

Example 20

The following example compares the empty string syntax with the negated keyword usage.

```
$ rmu/unload-
    /record=(nofile,format=delimited,prefix="",suffix="",null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.43.
$ rmu/unload-
    /record=(nofile,format=delimited,noprefix,nosuffix,null="*") -
    db$:mf_personnel -
    work_status -
    sys$output:
0,INACTIVE,RECORD EXPIRED
1,ACTIVE  ,FULL TIME
2,ACTIVE  ,PART TIME
*,*,*
%RMU-I-DATRECUNL,   4 data records unloaded 28-MAR-2017 17:24:07.56.
$
```

1.70 RMU Verify Command

Checks the internal integrity of database data structures. The RMU Verify command does not verify the data itself. You can verify specific portions of a database or the integrity of routines stored in the database by using qualifiers.

Format

RMU/Verify root-file-spec

Command Qualifiers	Defaults
/All	See description
/Areas [= storage-area-list]	No Area checking performed
/Checksum_Only	Full page verification
/[No]Constraints = [(Options)]	/NoConstraint
/[No]Data	/Data when /Indexes is used
/End=page-number	/End=last-page
/[No]Error_Limit [= n]	/Error_Limit = 100
/For_Table [= (table_name,...)]	See description
/[No]Functions	/Nofunctions
/Incremental	See description
/Indexes [= index-list]	No index checking performed
/[No]Key_Values	See description
/Lareas [= logical-area-list]	No LAREA checking performed
/[No]Log	Current DCL verify value
/[No]Orphan_Indexes	NoOrphan_Indexes
/Output=file-spec	SYS$OUTPUT
/[No]Root	/Root
/[No]Routines	/Noroutines
/[No]Segmented_Strings	See description
/Snapshots	No snapshot verification
/Start=page-number	/Start=1
/Transaction_Type=option	/Transaction_Type=Protected

If you specify the RMU Verify command without any qualifiers, a database root file verification and full page verification of the area inventory page (AIP) and the area bit map (ABM) pages in the default RDB$SYSTEM storage area are performed. Also, the snapshot files and after-image journals are validated (even if journaling has been disabled).

1.70 RMU Verify Command

The RMU Verify command checks space area management (SPAM) pages for proper format. The contents of the individual entries are verified as the individual data pages are verified. The command does not attempt to determine if data within rows is reasonable or plausible.

Description

The RMU Verify command checks the internal integrity of database data structures. Oracle Corporation strongly recommends that you verify your database following any kind of serious system malfunction. You should also verify your database as part of routine maintenance, perhaps before performing backup operations. You can use the various qualifiers to perform verification of the maximum number of database areas in the time available.

Command Parameters

root-file-spec
The Oracle Rdb database to verify. The default file extension is .rdb.

Command Qualifiers

All
When you specify the All qualifier, the entire database is checked, including any external routines. Specifying the All qualifier is equivalent to issuing the list of qualifiers shown in the following command:

```
$ RMU/VERIFY/ROOT/CONSTRAINTS/INDEXES/DATA/AREAS -
_$ /SNAPSHOTS/SEGMENTED_STRINGS/LAREAS=*/ROUTINES MF_PERSONNEL.RDB
```

If you do not specify the All qualifier, the verification requested by the other qualifiers you specify is performed.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

Areas[=storage-area-list]
Specifies the storage areas of the database to verify. You can specify storage areas by name or by the area's ID number. When you specify the storage area by name, each storage area name must be the name defined in the SQL CREATE STORAGE AREA statement for the storage area, not the storage area file name. If you list multiple storage areas, separate the storage area names or ID numbers with a comma, and enclose the storage area list within parentheses. The Areas qualifier with no arguments (or Areas=*) directs Oracle RMU to verify all storage areas of the database. With a single-file database, if you do not specify a storage area name, the RDB$SYSTEM storage area is verified.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

The Areas qualifier can be used with indirect file references. See Section 1.3 for more information.

When the Areas qualifier is not specified, Oracle RMU does not verify any storage areas.

Checksum_Only
Specify with the Areas qualifier to perform only checksum verification of pages. This reduces the degree of verification done on a database page. While the RMU Verify command executes faster with the Checksum_Only qualifier than without it, it does not verify pages completely. This qualifier allows you to make trade-offs between speed of verification and thoroughness of verification. For more information on these trade-offs, see the *Oracle Rdb Guide to Database Maintenance*.

If this command finds a problem with a certain page, then that page can be verified in depth by using other qualifiers, such as Indexes, Areas, or Lareas.

Note that you can accomplish the same degree of verification during a backup operation by specifying the Checksum qualifier with the RMU Backup command. The advantage of specifying the Checksum qualifier with the RMU Backup command is that the checksum operation takes place concurrently with the backup operation.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

The default is for full verification of pages.

Constraints[=(Options)]
Noconstraints
Specifies which constraints Oracle RMU is to load and execute to check the integrity of data in the database. In addition, external routines (procedures and functions) referenced by constraints are activated and executed. Any exceptions produced cause the verify operation to report a failure. See the description of the routines qualifier for information on how routines are activated and executed.

The options are as follows:

- Constraints=(list)

Specifies the constraints which you want Oracle RMU to load and execute. If you specify more than one constraint, separate each constraint name with a comma and enclose the list in parentheses. You can specify the wildcard character, the asterisk (*), instead of a constraint list to indicate that you want all constraints checked for the database.

- Fallback

  Algorithms used by prior releases of Oracle Rdb remain available to RMU Verify Constraints using the FALLBACK keyword.

```
$ DEFINE/USER RDMS$SET_FLAGS ITEM_LIST
$ RMU/VERIFY/CONSTRAINTS=FALLBACK PERSONNEL
~H Extension (VERIFY CONSTRAINTS) Item List: (len=0)
~H: ...verify constraint "COLLEGE_CODE_REQUIRED"
~H: ...verify constraint "DEPT_CODE_REQUIRED"
~H: ...verify constraint "EMPLOYEE_ID_REQUIRED"
~H: ...verify constraint "JH_EMP_ID_EXISTS"
~H: ...verify constraint "JOB_CODE_REQUIRED"
~H: ...verify constraint "SH_EMP_ID_EXISTS"
~H: 6 tables processed.
$
```

- Tables=(list)

  Specifies the table for which constraints are to be checked. If you specify more than one table, separate each table name with a comma and enclose the list in parentheses. You can specify the wildcard character, the asterisk (*), instead of a table list to indicate that you want constraints checked for all tables in the database. This option is useful if you issued an RMU Load command with the Noconstraints qualifier.

- (Tables=(list), Constraints=(list))

  You can specify both the Tables and Constraints options to specify which combination of tables and constraints you want Oracle RMU to verify. If you specify the wildcard character, the asterisk (*), for the Tables option and a named constraint or constraints for the Constraint option within the same Oracle RMU command line, Oracle RMU verifies all constraints.

See the *Oracle Rdb Guide to Database Maintenance* for more information on verifying constraints.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

The default is the Noconstraints qualifier. When you specify the Noconstraints qualifier, Oracle RMU does not verify any constraints.

Data
Nodata
Specifies whether consistency checks are made between indexes and tables. When you specify the Data qualifier, Oracle RMU checks that every row to which an index points is a valid row for the table and it checks that every row in a table is pointed to by every index defined on the table. See the description of the Indexes qualifier for more information on how these comparisons are made.

The Data qualifier is valid only when it is used with the Indexes qualifier.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

The default is the Data qualifier.

End=page-number
Specifies the last page to be verified. This qualifier is used in conjunction with the Areas and Lareas qualifiers. If you do not use the End qualifier, Oracle RMU verifies all pages between the first page (or the page specified in the Start qualifier) and the last page of the storage area.

The End qualifier is valid only when you specify the Areas or Lareas qualifier.

See the Usage Notes entry in this command for the rules that determine which other qualifiers can be used in combination on the same RMU Verify command line.

[No]Error_Limit=n
Limits the number of diagnostics output when verifying an Oracle Rdb database. By default, RMU/VERIFY will limit the number of diagnostic messages output by RMU/VERIFY to 100. If this limit is exceeded, the RMU/VERIFY will terminate with a warning messsage.

```
%RMU-W-MAXVERERR, Maximum error limit 100 exceeded - ending verification
```

To disable this behavior and have no limit on the number of diagnostic messages output by the verification, /NOERROR_LIMIT must be specified on the command line.

```
RMU/VERIFY/ALL/NOERROR_LIMIT mf_personnel
```

To override the default of 100, the user can specify a numeric value for the /ERROR_LIMIT between 1 and 2147483647.

```
RMU/VERIFY/ALL/ERROR_LIMIT=50 mf_personnel
```

If /ERROR_LIMIT is specified without a value, the default limit of 100 diagnostics will be used.

```
RMU/VERIFY/ALL/ERROR_LIMIT mf_personnel
```

The /ERROR_LIMIT does not include any logging messages put out when the /LOG qualifier is used with RMU/VERIFY. It only includes diagnostic messages which are defined as messages of any severity which are not logging messages put out when the /LOG qualifier is specified. The %RMU-W-MAXVERERR message is not included in the /ERROR_LIMIT count but is output once the /ERROR_LIMIT in force is exceeded in place of the diagnostic message that would have exceeded the error limit.

We have done everything we can to make the /ERROR_LIMIT count as accurate as possible but related messages output together in one output operation may be counted as one message in a limited number of cases. Therefore, we do not guarantee absolute accuracy in the /ERROR_LIMIT count in all cases but consider it as an acceptably accurate way to limit the potentially large number of diagnostics that can be output by the RMU/VERIFY of a database.

For_Table[=(table_name,...)]

Allows table names to be specified for database index verification. If one or more database table names is specified with the /FOR_TABLE qualifier, all the indexes defined for the named table(s) will be selected for verification. If the /FOR_TABLE qualifier is used, then either the /INDEXES or /INDICES qualifier must also be specified in the same RMU/VERIFY command.

The /INDEXES or /INDICES qualifiers can continue to specify a list of one or more index names to be verified in addition to the /FOR_TABLE list of one or more table names for which all the indexes defined for each table are to be verified. Index names cannot be specified with the /FOR_TABLE qualifier and table names cannot be specified with the /INDEXES or /INDICES qualifiers. Either the syntax RMU/VERIFY/INDEXES/FOR_TABLE=* or RMU/VERIFY/INDEXES/FOR_TABLE can be used to specifiy that all indexes defined for all database tables should be verified. If lower case charcters are not to be converted to upper case in table names, table names must be delimited with double quotes.

The following syntax can be specified for the /FOR_TABLE qualifier.

```
/FOR_TABLE=(table_name,...)
/FOR_TABLE=table_name
/FOR_TABLE=*
/FOR_TABLE
```

Functions
Nofunctions
This qualifier is synonymous with the Routines qualifier. See the description of the Routines qualifier.

Incremental
Directs Oracle RMU to verify database pages that have changed since the last full or incremental verification. Oracle RMU stores timestamps in the database root file for both full and incremental verifications. To determine which pages have changed since the last verify operation, Oracle RMU compares these timestamps with the page timestamps. The page timestamps are updated whenever pages are updated. An incremental verification performs the same number of I/O operations as a full verification, but the incremental verification takes fewer CPU cycles than a full verification, allowing you to perform incremental verifications more frequently than you would perform full ones. The default is to perform a full verification.

Note

If you use the Incremental qualifier with the RMU Verify command, Oracle Corporation recommends that you use it only with the All qualifier and not with any other qualifiers.

The timestamps in the database root file are updated during full and incremental verifications only when the All qualifier is specified. Therefore, if you do not specify the All qualifier, two successive incremental verifications of the same storage area of the database perform the same verifications. This means that the second incremental verification does not pass over pages verified by the first incremental verification, contrary to what you might expect.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

If the Incremental qualifier is not specified, all requested pages are verified, regardless of the timestamp.

Indexes[=index-list]
Verifies the integrity of all but disabled indexes in the database if you specify the Indexes or the Indexes=* qualifier; verifies the integrity of a specific index, or of multiple indexes if you provide an index list. If you list multiple indexes, separate the index names with a comma, and enclose the index list within parentheses.

The verify operation creates a sorted list of all dbkeys for a table and a sorted list of all dbkeys in an index. By comparing these two lists, the verify operation can detect any cases of an index missing an entry for a data row. This comparison of dbkey lists occurs at the end of the verify operation.

Due to the use of SORT during this phase, you can tune the verify by allocating more SORT work files using the logical name RDMS$BIND_ SORT_WORKFILES, and locating these files on different devices using the SORTWORKn logical names. Use RMU Show Logical with the /Description qualifier for more detailed information.

If you specify the Log qualifier, you see messages similar to the following to indicate that the comparison is occurring:

```
%RMU-I-IDXVERSTR, Index data verification of logical area 60 (DEGREES) started.
%RMU-I-IDXVEREND, Index data verification of logical area 60 finished.
```

In addition, when you verify an index with the Data qualifier (the default), Oracle RMU also verifies the logical areas referenced by the indexes. See Example 5 in the Examples section.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

By default, Oracle RMU does not verify indexes.

The Indexes qualifier can be used with indirect file references. See Section 1.3 for more information.

[No]Key_Values

This qualifier verifies the key field values contained in a sorted, sorted ranked or hashed index against the key field values in the matching table row to make sure the key field values contained in the index match the field values in the table. This qualifier can only be specified if indexes are being verified explicitly as with the RMU/VERIFY/INDEX command or if indexes are being verified by default such as with the RMU/VERIFY/ALL command. Key field values will be verified for all indexes or a specified list of one or more indexes. Diagnostic error messages will be put out if the index key field value does not match the table row key field value, if a DBKEY contained in the index node or hash bucket does not match the DBKEY of a table row, or if the DBKEY of a table row is not contained in the index node or hash bucket. Indexes with one key field or multiple key fields can be verified using the /KEY_VALUES qualifier.

This qualifier will ignore index fields that have a collating sequence or are of type character varying. The character encoding for the collating sequence prevents a byte by byte comparison with the column row values. The key encoding for character varying pads with spaces so the precise length cannot

be compared with the column row values. For these index fields, a warning message will be output and a comparison with the column row values will not be done.

Note that /KEY_VALUES is not the default and must be specified.

Lareas[=logical-area-list]
Specifies the storage area pages allocated to a logical area or logical areas that you want verified. If you list multiple logical areas, separate the logical area names with a comma, and enclose the logical area list within parentheses. The Lareas qualifier with no arguments (or Lareas=*) directs Oracle RMU to verify all logical areas of the database. When a logical area is verified, each page in the area is read and verified sequentially starting at the first page.

If an index name is specified with the Lareas qualifier, the index is verified, but it is not verified as a logical area. In this case, the first index record is fetched (which could be on any page) and the verification follows the structure of the index. (For example, if the index record points to other index records, then those records are fetched and verified. If the index node is a leaf node, then the data record is fetched and verified. These data pages might reside in different logical areas.)

Use this qualifier to verify one or more tables.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

The Lareas qualifier can be used with indirect file references. See Section 1.3 for more information.

By default, Oracle RMU does not verify logical areas.

Log
Nolog
Specifies whether the processing of the command is reported to SYS$OUTPUT. By default, SYS$OUTPUT is your terminal. Specify the Log qualifier to request that each verify operation be displayed to SYS$OUTPUT and the Nolog qualifier to prevent this display. If you specify neither, the default is the current setting of the DCL verify switch. (The DCL SET VERIFY command controls the DCL verify switch.)

When you specify the Log qualifier, Oracle RMU displays the time taken to verify each database area specified and the total time taken for the complete verification operation. The display from the Log qualifier is also useful for showing you how much of the verification operation is completed.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

Orphan_Indexes
NoOrphan_Indexes
To ensure Oracle Rdb database integrity, this qualifier can be used to detect "orphan" Hash index buckets on database pages in mixed storage areas which do not belong to any existing Hash index defined for the database. Orphan Hash buckets are either not referenced in a SYSTEM RECORD on a mixed area database page or are not refernced by another Hash bucket.

The default is NoOrphan_Indexes. This feature must be activated by specifying the Orphan_Indexes qualifier. Orphan Hash index buckets will only be reported if RMU/VERIFY is verifying one or more HASH indexes as part of the database verification. For each orphan Hash bucket detected, an error message will be output specifying the storage area name and physical "dbkey" address of the orphan Hash bucket. The physical dbkey address specifies the storage area number, the storage area page number, and the storage area line number of the orphan Hash bucket. This information can be used with the RMU/DUMP command to dump the area page where the orphan Hash bucket is located.

In the short term, these orphaned hash index elements are harmless but consume space which would otherwise be used by new inserts. Eventually, as objects get dropped and created, these elements may be confused with current structures. Therefore, Oracle recommends cleaning them up as soon as practical.

However, these orphaned hash index elements can no longer be removed using the standard DROP commands in SQL. Reclaiming the space used by these elements will require a DROP STORAGE AREA for the affected area. The database administrator should create a replacement storage area and use ALTER or DROP commands to move other tables and indices out of the affected storage area. Then use DROP STORAGE AREA to remove the unused area. Alternatively, you can use the SQL EXPORT DATABASE and IMPORT DATABASE commands to rebuild the whole database.

If RMU/VERIFY detects and reports any structural problems with the database pages or Hash index stuctures for the area being verified, or any problems with VMS sort which is used in the process of detecting orphan Hash buckets, detection of orphan Hash buckets will be aborted for the area and the following error message will be output.

```
$ RMU/VERIFY/ALL/ORPHAN_INDEXES MF_PERSONNEL
%RMU-E-ORPHANERR, Error searching for orphan index nodes in area EMPIDS_LOW
```

The user should correct the reported problems and repeat the verification.

The following example shows the diagnostic error message that will be put out by RMU/VERIFY for each orphan Hash bucket found on a database page in the current storage area. Both the storage area name and the physical dbkey address of the orphan Hash bucket are reported. The dbkey address in the message following the word "at" specifies the storage area number followed by the page number followed by the line number where the orphan Hash bucket is located.

```
$ RMU/VERIFY/ALL/ORPHAN_INDEXES/NOERROR_LIMIT DATABASE.RDB
%RMU-E-ORPHANIDX, Orphan hashed index bucket found in area
 DATABASE_AREA at 21:2:5
%RMU-E-ORPHANIDX, Orphan hashed index bucket found in area
 DATABASE_AREA at 21:2:9
%RMU-E-ORPHANIDX, Orphan hashed index bucket found in area
DATABASE_AREA at 21:2510:32
%RMU-E-ORPHANIDX, Orphan hashed index bucket found in area
DATABASE_AREA at 21:2510:43
```

Output=file-spec
Specifies the name of the file where output will be sent. The default is SYS$OUTPUT. When you specify a file name, the default output file type is .lis.

If you specify both the Log qualifier and the Output qualifier, the messages produced by the Log qualifier and any error messages are directed into the output file specification. If you specify only the Output qualifier, only error messages are captured in the output file. See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

Root
Noroot
Specifies that, in a multifile database, only fields in the database root (.rdb) file and all the pointers to the database (.rda, .snp, .aij) files are verified. The snapshot (.snp) files are validated; that is, only the first page is checked to make sure that it is indeed an .snp file and belongs to the database being verified. If after-image journaling is enabled, the .aij files are validated. The AIP and ABM pages are verified when you specify the Root qualifier.

If you specify the Noroot qualifier, and no other qualifiers, only the AIP pages are verified. If you specify the Noroot qualifier, and the Areas or the Lareas qualifier, ABM and SPAM pages are verified as the other pages in the storage area or logical area are verified.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

You can specify the Root qualifier for a single-file database.

The default is the Root qualifier.

Routines
Noroutines
The Routines qualifier verifies the integrity of all routine (function and procedure) definitions stored in the database. Oracle RMU performs the verification by activating and deactivating each external routine, one at a time. Any exceptions produced cause the verify operation to report a failure.

The Routines qualifier verifies that the shareable image is located where expected, is accessible, and that the correct entry point is at this location. The expected location is that which was specified in the SQL CREATE FUNCTION or CREATE PROCEDURE statement. If the shareable image is not in the expected location, is not accessible, or the entry point is not at the expected location, you receive an error message.

If Oracle RMU is installed with SYSPRV, any external routine image for a routine that is registered with client-site binding must meet the following criteria or the RMU Verify command cannot check for the existence of the entry point for the routine in the image:

- It must be installed.
- It must have been specified with an image file specification that uses only logicals defined with the DCL /SYSTEM and /EXECUTIVE qualifiers.

In addition, the user issuing the RMU Verify command must have OpenVMS SYSPRV in order for the routine to be activated.

The Noroutines qualifier specifies that routine interface not be verified.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

By default, Oracle RMU does not verify any routines.

Segmented_Strings
Nosegmented_Strings
Verifies all list (segmented string) data for each column, in each table in any of the two types of storage areas: read/write and read-only (on read/write disk devices). When you specify the RMU Verify command with the All qualifier, all list data (segmented strings) in all tables is verified in the database. The

Segmented_Strings qualifier can only be used with the Lareas qualifier and has the following meanings when used with this qualifier:

- RMU Verify command with the Lareas=* and the Segmented_Strings qualifiers.

  Segmented strings in all tables are verified.

- RMU Verify command with the Lareas=(LAREA_1, . . . ,LAREA_N) and the Segmented_Strings qualifiers.

  Segmented strings in tables LAREA_1, . . . ,LAREA_N are verified.

  If the Segmented_Strings qualifier is omitted, there is no list data verification.

  The Segmented_Strings qualifier verifies all list data in each column of each row in the database. The verify operation tries to fetch all pointer segments and all data segments from the pointer segments, and verifies all header information, including the total length of the segment, the number of pointer segments, the number of data segments, and the length of the longest segment for the list data.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

Snapshots

Verifies the snapshot area of the specified storage areas up to the page header level. The Snapshots qualifier only performs checksum verification of snapshot pages.

The Snapshots qualifier is valid only when you also specify the Areas qualifier.

See the Usage Notes entry in this command for the rules that determine which other qualifiers can be used in combination on the same RMU Verify command line.

The Snapshots qualifier can be used with indirect file references. See Section 1.3 for more information.

By default, Oracle RMU does not verify snapshots.

Start=page-number

Specifies the first page to be verified. This qualifier is used in conjunction with the Areas and Lareas qualifiers. If you do not use the Start qualifier, the verification begins with the first page of the storage area.

The Start qualifier is valid only when you specify the Areas or Lareas qualifier also.

See the Usage Notes entry in this command for the rules that determine which other qualifiers can be used in combination on the same RMU Verify command line.

Transaction_Type=option
Sets the retrieval lock for the storage areas being verified.

Use one of the following keywords to control the transaction mode:

- Automatic

  When Transaction_Type=Automatic is specified, the transaction type depends on the current database settings for snapshots (enabled, deferred, or disabled), transaction modes available to this user, and the standby status of the database.

- Read_Only

  Starts a Read_Only transaction.

- Exclusive

  Starts a Read_Write transaction and reserves the table for Exclusive_Read.

- Protected

  Starts a Read_Write transaction and reserves the table for Protected_Read. Protected mode is the default.

- Shared

  Starts a Read_Write transaction and reserves the table for Shared_Read.

Use one of the following options with the keyword Isolation_Level=[option] to specify the transaction isolation level:

- Read_Committed
- Repeatable_Read
- Serializable. Serializable is the default setting.

Refer to the SET TRANSACTION statement in the Oracle Rdb SQL Reference Manual for a complete description of the transaction isolation levels.

Specify the wait setting by using one of the following keywords:

- Wait

  Waits indefinitely for a locked resource to become available. Wait is the default behavior.

- Wait=n

The value you supply for *n* is the transaction lock timeout interval. When you supply this value, Oracle Rdb waits *n* seconds before aborting the wait and the RMU Verify session. Specifying a wait timeout interval of zero is equivalent to specifying Nowait.

- Nowait

  Does not wait for a locked resource to become available.

See the Usage Notes entry in this command for the rules that determine which qualifiers can be used in combination on the same RMU Verify command line.

Usage Notes

- To use the RMU Verify command for a database, you must have the RMU$VERIFY privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege. You must also have the SQL DBADM privilege.
- The rules that determine which qualifiers can be used in combination on the same RMU Verify command line are as follows:
  - The Incremental, Log, Output, and Transaction_Type qualifiers can be used in combination with any other qualifiers on the same RMU Verify command line.
  - If the All qualifier is specified, the only other qualifiers you can specify on the same command line are:
    * Noroutines (or Nofunctions)
    * Nosegmented_Strings
  - If the All qualifier is *not* specified, then any combination of the following qualifiers can be specified on the same command line:
    * Areas
    * Constraints
    * [No]Functions
    * Indexes
    * Lareas
    * [No]Root
    * [No]Routines

  - You must specify the Areas qualifier to specify the Checksum_Only or Snapshots qualifier.
  - You must specify the Lareas qualifier to specify the Segmented_Strings qualifier.
  - You must specify either the Areas or Lareas qualifier to specify the Start and End qualifiers.
  - You cannot specify the Indexes qualifier on the same RMU Verify command line with the Start and End qualifiers.
  - You must specify the Indexes qualifier to specify the [No]Data qualifier.

- You can significantly improve the performance of RMU Verify for your database by employing the verification strategies described in the *Oracle Rdb Guide to Database Maintenance*. In addition, detected asynchronous prefetch should be enabled to achieve the best performance of this command. You can determine the setting for your database by issuing the RMU Dump command with the Header qualifier.

  If detected asynchronous prefetch is disabled, and you do not want to enable it for the database, you can enable it for your Oracle RMU operations by defining the following logicals at the process level:

```
$ DEFINE RDM$BIND_DAPF_ENABLED 1
$ DEFINE RDM$BIND_DAPF_DEPTH_BUF_CNT P1
```

  P1 is a value between 10 and 20 percent of the user buffer count.

- The RMU Verify command ignores any constraint that has been disabled (with the SQL ALTER TABLE enable-disable clause) unless you specify the constraint name in the Constraints=(Constraints=list) qualifier of the RMU Verify command. If the Constraints qualifier is specified without a list, disabled constraints are ignored.

  By specifying the name of a disabled constraint in the Constraints=(Constraints=list) qualifier, you can check it periodically without having to reenable it. You might use this to provide a business rule in the database that needs checking only occasionally. This is a useful practice if the overhead of checking the constraint during operating hours is too expensive, or if it is already being enforced by the application.

- The number of work files used by the RMU Verify command is controlled by the RDMS$BIND_SORT_WORKFILES logical name. The allowable values are 1 through 10 inclusive, with a default value of 2. The location of these work files can be specified with device specifications, using the SORTWORKn logical name (where *n* is a number from 0 to 9). See the

OpenVMS documentation set for more information on using SORT/MERGE. See the *Oracle Rdb7 Guide to Database Performance and Tuning* for more information on using these Oracle Rdb logical names.

Because two separate sort streams are used internally by the RMU Verify command when the Index qualifier is specified, the number of work files specified is used for each stream. For example, if RDM$BIND_SORT_ WORKFILES is defined to be 10, twenty work files are created.

- The /SEGMENTED_STRINGS qualifier can be used with the Oracle Rdb RMU/VERIFY command to verify all list (segmented string) data for columns in one or more database tables. To verify segmented strings for all database tables, any of the following commands can be specified:

  * RMU/VERIFY/ALL
  * RMU/VERIFY/SEGMENTED_STRINGS/LAREAS
  * RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=*

```
$ RMU/VERIFY/ALL PERSONNEL
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS PERSONNEL
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=* PERSONNEL
```

  Neither the /SEGMENTED_STRINGS qualifier nor the /LAREAS qualifier can be specified with the /ALL qualifier since the /ALL qualifier defaults to /LAREAS=* and /SEGMENTED_STRINGS in addition to other qualifiers.

```
$ RMU/VERIFY/ALL/SEGMENTED_STRINGS/LAREAS PERSONNEL
%RMU-F-CONFLSWIT, conflicting qualifiers /ALL and /LAREAS
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 16-JAN-2014 15:09:45.22
$ RMU/VERIFY/ALL/SEGMENTED_STRINGS PERSONNEL
%RMU-F-CONFLSWIT, conflicting qualifiers /ALL and /SEGMENTED_STRINGS
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 16-JAN-2014 15:10:10.98
```

  To verify segmented strings for individual tables, the /LAREAS qualifier must be used with the /SEGMENTED_STRING qualifier to specify one or more names of tables to be verified.

```
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=RDB$RELATIONS PERSONNEL
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=(RDB$RELATIONS,RESUMES) PERSONNEL
```

  If the /SEGMENTED_STRINGS qualifier is specified, the /LAREAS qualifier must also be specified.

```
$ RMU/VERIFY/SEGMENTED_STRINGS PERSONNEL
%RMU-F-CONFLSWIT, conflicting qualifiers /SEGMENTED_STRINGS and not LAREA
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 16-JAN-2014 15:11:20.16
```

When the /LAREAS qualifier is used with the /SEGMENTED_STRINGS qualifier, only logical area names that are also database table names can be specified or a fatal error will be output and the verify command will be aborted.

```
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=RESUMES MF_PERSONNEL
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=95 MF_PERSONNEL
%RMU-F-INVSEGTBL, Invalid table name 95 specified for segmented string
verification
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 27-JAN-2014 15:02:39.04
$ RMU/VERIFY/SEGMENTED_STRINGS/LAREAS=SH_EMPLOYEE_ID MF_PERSONNEL
%RMU-F-INVSEGTBL, Invalid table name SH_EMPLOYEE_ID specified for segmented
string verification
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 27-JAN-2014 15:03:42.15
```

If the /LOG qualifier is specified or logging is the default for the verify command, a log message will list the tables containing columns defined for segmented string data for which the segmented string data will be verified.

```
$ RMU/VERIFY/LOG/SEGMENTED_STRINGS/LAREAS=RESUMES MF_PERSONNEL
%RMU-I-BGNROOVER, beginning root verification
%RMU-I-ENDROOVER, completed root verification
%RMU-I-TBLSEGVER, Segmented strings will be verified for table RESUMES
%RMU-I-DBBOUND,   bound to database "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
%RMU-I-OPENAREA,  opened storage area RDB$SYSTEM for protected retrieval
%RMU-I-BGNAIPVER, beginning AIP pages verification
%RMU-I-ENDAIPVER, completed AIP pages verification
%RMU-I-BGNABMSPM, beginning ABM pages verification
%RMU-I-OPENAREA,  opened storage area MF_PERS_SEGSTR for protected retrieval
%RMU-I-ENDABMSPM, completed ABM pages verification
%RMU-I-BSGPGLARE, beginning verification of RESUMES logical area
                  as part of EMP_INFO storage area
%RMU-I-OPENAREA,  opened storage area EMP_INFO for protected retrieval
%RMU-I-ESGPGLARE, completed verification of RESUMES logical area
                  as part of EMP_INFO storage area
%RMU-I-CLOSAREAS, releasing protected retrieval lock on all storage areas
%RMU-S-ENDVERIFY, elapsed time for verification :    0 00:00:00.80
```

- NOT NULL constraints and the not null restriction for PRIMARY KEY constraints are validated using a single table scan for all such constraints on a table. This should reduce the table sequential scans in the absence of a suitable index for each constraint. Note that the side effect is that only the first failing constraint name is reported.
- When multiple tables are verified, the constraints are ordered by table name. The goal is to make use of any buffered table rows for subsequent constraint queries. In prior versions, the constraints were verified in (approximate) definition order which might result in other tables being read and buffered data not being available.

- During RMU Verify Constraints, or during ALTER TABLE ... ENABLE ALL CONSTRAINTS more detailed information on the verify can be seen by defining the ITEM_LIST flag. This can be done using either the SET FLAGS statement in SQL or defining the logical name RDMS$SET_FLAGS.

  In the case of a failing NOT NULL constraint, the DBKEY of the failing row is reported. The failure status (VMS condition code) is also reported along with the associated message text.
- When using the RMU/VERIFY command, a process requires a minimum of the following quotas:
  - FILLM and CHANNELCNT at least 25 more than the total number of database storage areas, snapshot storage areas, and after image journals.
  - Large enough BYTLM, page file quota and working set to open all of the database storage areas, snapshot storage areas, and after image journals.

Examples

Example 1

The following command verifies the entire mf_personnel database because the All qualifier is specified:

```
$ RMU/VERIFY/ALL/LOG MF_PERSONNEL.RDB
```

Example 2

The following command verifies the storage areas EMPIDS_LOW, EMPIDS_MID, and EMPIDS_OVER in the mf_personnel database:

```
$ RMU/VERIFY/AREAS=(EMPIDS_LOW,EMPIDS_MID,EMPIDS_OVER)/LOG -
_$ MF_PERSONNEL.RDB
```

Example 3

The following command performs only a checksum verification on all the storage areas in the database called large_database. The Checksum_Only qualifier quickly detects obvious checksum problems with the database. If a checksum problem is found on a page, you can dump the page by using the RMU Dump command, and verify the appropriate logical areas and indexes.

```
$ RMU/VERIFY/AREAS=*/CHECKSUM_ONLY/LOG LARGE_DATABASE
```

Example 4

1.70 RMU Verify Command

The following command verifies the Candidates and Colleges tables:

```
$ RMU/VERIFY/LAREAS=(CANDIDATES,COLLEGES)/LOG MF_PERSONNEL.RDB
```

Example 5

The following example displays the behavior of the index verification method Oracle RMU employs beginning in Oracle Rdb V7.0. The first RMU Verify command shows the log output when the command is issued under Oracle Rdb V6.1. The second RMU Verify command shows the log output when the command is issued under Oracle Rdb V7.0.

```
$ @SYS$LIBRARY:RDB$SETVER 6.1
$ SET DEF DB1:[V61]
$ RMU/VERIFY/INDEXES=EMP_EMPLOYEE_ID/DATA MF_PERSONNEL.RDB/LOG
%RMU-I-BGNROOVER, beginning root verification
%RMU-I-ENDROOVER, completed root verification
%RMU-I-DBBOUND, bound to database "DB1:[V61]MF_PERSONNEL.RDB;1"
%RMU-I-OPENAREA, opened storage area RDB$SYSTEM for protected retrieval
%RMU-I-BGNAIPVER, beginning AIP pages verification
%RMU-I-ENDAIPVER, completed AIP pages verification
%RMU-I-BGNABMSPM, beginning ABM pages verification
%RMU-I-OPENAREA, opened storage area MF_PERS_SEGSTR for protected retrieval
%RMU-I-ENDABMSPM, completed ABM pages verification
%RMU-I-BGNNDXVER, beginning verification of index EMP_EMPLOYEE_ID
%RMU-I-OPENAREA, opened storage area EMPIDS_LOW for protected retrieval
%RMU-I-OPENAREA, opened storage area EMPIDS_MID for protected retrieval
%RMU-I-OPENAREA, opened storage area EMPIDS_OVER for protected retrieval
%RMU-I-ENDNDXVER, completed verification of index EMP_EMPLOYEE_ID
%RMU-I-CLOSAREAS, releasing protected retrieval lock on all storage areas
%RMU-S-ENDVERIFY, elapsed time for verification :    0 00:00:09.14
$ @SYS$LIBRARY:RDB$SETVER 7.0
$ SET DEF DB1:[V70]
$ RMU/VERIFY/INDEXES=EMP_EMPLOYEE_ID/DATA MF_PERSONNEL.RDB/LOG
%RMU-I-BGNROOVER, beginning root verification
%RMU-I-ENDROOVER, completed root verification
%RMU-I-DBBOUND, bound to database "DB1:[V70]MF_PERSONNEL.RDB;1"
%RMU-I-OPENAREA, opened storage area RDB$SYSTEM for protected retrieval
%RMU-I-BGNAIPVER, beginning AIP pages verification
%RMU-I-ENDAIPVER, completed AIP pages verification
%RMU-I-BGNABMSPM, beginning ABM pages verification
%RMU-I-ENDABMSPM, completed ABM pages verification
%RMU-I-BGNNDXVER, beginning verification of index EMP_EMPLOYEE_ID
%RMU-I-OPENAREA, opened storage area EMPIDS_LOW for protected retrieval
%RMU-I-OPENAREA, opened storage area EMPIDS_MID for protected retrieval
%RMU-I-OPENAREA, opened storage area EMPIDS_OVER for protected retrieval
%RMU-I-ENDNDXVER, completed verification of index EMP_EMPLOYEE_ID
```

```
%RMU-I-BSGPGLARE, beginning verification of EMPLOYEES logical area
                  as part of EMPIDS_LOW storage area
%RMU-I-ESGPGLARE, completed verification of EMPLOYEES logical area
                  as part of EMPIDS_LOW storage area
%RMU-I-BSGPGLARE, beginning verification of EMPLOYEES logical area
                  as part of EMPIDS_MID storage area
%RMU-I-ESGPGLARE, completed verification of EMPLOYEES logical area
                  as part of EMPIDS_MID storage area
%RMU-I-BSGPGLARE, beginning verification of EMPLOYEES logical area
                  as part of EMPIDS_OVER storage area
%RMU-I-ESGPGLARE, completed verification of EMPLOYEES logical area
                  as part of EMPIDS_OVER storage area
%RMU-I-IDXVERSTR, Beginning index data verification of logical area 69
 (EMPLOYEES).
%RMU-I-IDXVEREND, Completed data verification of logical area 69.
%RMU-I-IDXVERSTR, Beginning index data verification of logical area 70
 (EMPLOYEES).
%RMU-I-IDXVEREND, Completed data verification of logical area 70.
%RMU-I-IDXVERSTR, Beginning index data verification of logical area 71
 (EMPLOYEES).
%RMU-I-IDXVEREND, Completed data verification of logical area 71.
%RMU-I-CLOSAREAS, releasing protected retrieval lock on all storage areas
%RMU-S-ENDVERIFY, elapsed time for verification :    0 00:00:11.36
```

Example 6

The following example loads data into a table, verifies the table, and then identifies loaded rows that violated a constraint.

Because the Noconstraints qualifier is specified with the RMU Load command, data that violates database integrity might be added to the database. The second RMU Verify command verifies the table that was just loaded and reveals that data that violates constraints on the table was indeed loaded.

An SQL command is issued to determine which rows violated the constraint so that they can either be removed from the database, or added to the EMPLOYEES table to restore database integrity. The final RMU Verify command checks the constraint again to ensure that changes made have restored the integrity of the database.

1.70 RMU Verify Command

```
$ !
$ ! Load data into the JOB_HISTORY table of the mf_personnel database.
$ ! Specify the Noconstraints qualifier:
$ !
$ RMU/LOAD/RECORD_DEFINITION=(FILE=JOB_HIST.RRD, FORMAT=TEXT) -
_$ MF_PERSONNEL.RDB JOB_HISTORY JOB_HIST.UNL/NOCONSTRAINTS
%RMU-I-DATRECREAD,  18 data records read from input file.
%RMU-I-DATRECSTO,   18 data records stored.
$ !
$ ! Verify the JOB_HISTORY table:
$ !
$ RMU/VERIFY/CONSTRAINTS=(TABLE=JOB_HISTORY) MF_PERSONNEL.RDB
%RMU-W-CONSTFAIL, Verification of constraint "JOB_HISTORY_FOREIGN1"
 has failed.
$ !
$ ! Issue SQL statements to determine what the definition of the
$ ! constraint is and which of the loaded rows violated
$ ! the constraint.  Then issue an SQL command to insert data that will
$ ! restore the data integrity of the database:
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> SHOW TABLE JOB_HISTORY
   .
   .
   .
JOB_HISTORY_FOREIGN1
 Foreign Key constraint
 Column constraint for JOB_HISTORY.EMPLOYEE_ID
 Evaluated on COMMIT
 Source:
        JOB_HISTORY.EMPLOYEE_ID REFERENCES EMPLOYEES (EMPLOYEE_ID)
   .
   .
   .
SQL> SELECT DISTINCT(EMPLOYEE_ID)
cont> FROM JOB_HISTORY
cont> WHERE NOT EXISTS
cont>           (SELECT *
cont>          FROM EMPLOYEES AS E
cont>          WHERE E.EMPLOYEE_ID = JOB_HISTORY.EMPLOYEE_ID);
EMPLOYEE_ID
 10164
 10165
 10166
 10167
 10168
 10169
6 rows selected
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10164', 'Smith');
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10165', 'Frederico');
```

```
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10166', 'Watts');
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10167', 'Risley');
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10168', 'Pietryka');
SQL> INSERT INTO EMPLOYEES (EMPLOYEE_ID, LAST_NAME)
cont> VALUES ('10169', 'Jussaume');
SQL> COMMIT;
SQL> EXIT
$ !
$ ! Check that data integrity has been restored:
$ !
$ RMU/VERIFY/CONSTRAINTS=(CONSTRAINTS=JOB_HISTORY_FOREIGN1, -
_$ TABLE=JOB_HISTORY) MF_PERSONNEL.RDB
$ !
$ ! No messages are returned.  Data integrity has been restored.
```

Example 7

The following example creates an external function in which the external name is incorrect. When the function is verified, Oracle RMU cannot find the entry point and returns an error. The external function is then dropped and then re-created correctly. The verification now succeeds:

```
$ ! Attach to database and create a function.  The external name is
$ ! mistyped:
$ !
SQL> ATTACH 'filename mf_personnel.rdb';
SQL> create function SQRT (in double precision) returns double precision;
cont> external name MTH$SORT location 'SYS$SHARE:MTHRTL'
cont> language GENERAL
cont> GENERAL PARAMETER STYLE;
SQL> COMMIT;
SQL> EXIT;
$ !
$ ! Verify the function:
$ !
$ RMU/VERIFY/ROUTINES MF_PERSONNEL.RDB
%RMU-E-NOENTRPT,  No entry point found for external routine SQRT.
                      Image name is SYS$SHARE:MTHRTL.
                      Entry point is MTH$SORT.
```

1.70 RMU Verify Command

```
$ !
$ ! Oracle RMU cannot find the entry point.  Drop the
$ ! function and reenter correctly:
$ !
$ SQL
SQL> ATTACH 'FILENAME mf_personnel.rdb';
SQL> DROP FUNCTION SQRT;
SQL> create function SQRT (in double precision) returns double precision;
cont> external name MTH$SQRT location 'SYS$SHARE:MTHRTL'
cont> language GENERAL
cont> GENERAL PARAMETER STYLE;
SQL> COMMIT;
SQL> EXIT;
$ !
$ ! Verification is now successful:
$ !
$ RMU/VERIFY/ROUTINES MF_PERSONNEL.RDB
```

Example 8

The following example demonstrates that the RMU Verify command verifies disabled constraints only when you explicitly specify the disabled constraint.

```
$ SQL
SQL> ATTACH 'FILENAME MF_PERSONNEL.RDB';
SQL> -- Disable the EMP_SEX_VALUES constraint.
SQL> ALTER TABLE EMPLOYEES DISABLE CONSTRAINT EMP_SEX_VALUES;
SQL> COMMIT;
SQL> -- Insert a value that violates the EMP_SEX_VALUES constraint.
SQL> INSERT INTO EMPLOYEES
cont> (EMPLOYEE_ID, LAST_NAME, SEX)
cont> VALUES ('99999', 'JICKLING', 'G');
1 row inserted
SQL> COMMIT;
SQL> EXIT;
$ !
$ ! The following two verify commands do not return an error
$ ! because the disabled constraint is not explicitly specified.
$ !
$ RMU/VERIFY MF_PERSONNEL.RDB
$ RMU/VERIFY MF_PERSONNEL.RDB/CONSTRAINTS
$ !
$ ! The following verify command returns an warning message to
$ ! inform you that data that violates the disabled constraint
$ ! has been inserted into the database.
$ !
$ RMU/VERIFY MF_PERSONNEL.RDB/CONSTRAINT=(CONSTRAINT=EMP_SEX_VALUES)
%RMU-W-CONSTFAIL, Verification of constraint "EMP_SEX_VALUES" has failed.
```

Example 9

The following examples show that both the /FOR_TABLE and /INDEXES or /INDICES qualifiers can specify either no values or lists of one or more table or index names in the same RMU/VERIFY command line. Only table name values can be specified with the /FOR_TABLE qualifier and only index name values can be specified with the /INDEXES or /INDICES qualifiers. If table name values are specified, all the indexes defined for each named table will be verified.

```
$ RMU/VERIFY/INDEXES/FOR_TABLE=EMPLOYEES/NOLOG MF_PERSONNEL
$ RMU/VERIFY/INDEXES=(EMPLOYEES_HASH,EMP_EMPLOYEE_ID)/FOR_TABLE=COLLEGES/NOLOG
MF_PERSONNEL
$ RMU/VERIFY/INDICES=EMP_EMPLOYEE_ID/FOR_TABLE=(COLLEGES,JOB_HISTORY)/NOLOG
MF_PERSONNEL
```

Example 10

The following example shows that both of the following commands are equivalent and will verify all indexes defined for all tables in the database.

```
$ RMU/VERIFY/INDEXES/FOR_TABLE=*/NOLOG MF_PERSONNEL
$ RMU/VERIFY/INDICES/FOR_TABLE/NOLOG MF_PERSONNEL
```

Example 11

The following example shows that if the /FOR_TABLE qualifier is specified, either the /INDEXES or /INDICES qualifier must be specified in the same RMU/VERIFY command or a fatal error will occur.

```
$ RMU/VERIFY/FOR_TABLE=EMPLOYEES MF_PERSONNEL
%RMU-F-CONFLSWIT, conflicting qualifiers /FOR_TABLE and /INDEXES or /INDICES
not specified
%RMU-F-FTL_VER, Fatal error for VERIFY operation at 14-FEB-2013 14:58:47.09
```

Example 12

In the following example, the RMU/VERIFY has an error limit of 5 diagnostic messages specified. Therefore, after 5 diagnostic messages are output, the next message specifies that the error limit of 5 has been exceeded.

```
$ rmu/verify/all/error_limit=5 mf_personnel
%RMU-W-PAGCKSBAD, area EMP_INFO, page 2
                  contains an invalid checksum
                  expected: A77B3D6D, found: A7713D6D
%RMU-W-PAGLIXFRS, area EMP_INFO, page 2
                  line index entry 15 maps free space at offset 00000106 (hex)
%RMU-W-PAGLIXSML, area EMP_INFO, page 2
                  line index entry 19, length too small
                  expected at least 2, found: 0
%RMU-W-PAGLIXSML, area EMP_INFO, page 2
                  line index entry 20, length too small
                  expected at least 2, found: 0
%RMU-W-PAGLIXSML, area EMP_INFO, page 2
                  line index entry 21, length too small
                  expected at least 2, found: 0
%RMU-W-MAXVERERR, Maximum error limit 5 exceeded - ending verification
```

Example 13

The following example, using the KEY_VALUES qualifier, shows the error message put out if the index key field value in the index structure does not match the key field value in the table row. The DBKEY pointer to the index node or hash bucket that contains the row DBKEY is output as well as the table row DBKEY.

```
$ RMU/VERIFY/index=JH_EMPLOYEE_ID/key_values MF_PERSONNEL
%RMU-E-BADIDXFLD, Index JH_EMPLOYEE_ID key field EMPLOYEE_ID value at dbkey
93:810:3 does not match stored value for table JOB_HISTORY at dbkey 90:289:5 .
```

Example 14

The following example, using the KEY_VALUES qualifier, shows the error message put out if the key field value(s) returned from index only retrieval of the key fields do not match on row DBKEY with the key fields returned sequentially from the table row. Because the retrieval is out of sequence, the key field values cannot be compared.

```
$ rmu/verify/index/key_values MF_PERSONNEL
%RMU-E-BADIDXDBK, Index JH_EMPLOYEE_ID dbkey 91:413:3 does not match
table JOB_HISTORY row dbkey 90:200:4 .
```

Example 15

The following example, using the KEY_VALUES qualifier, shows the error message put out if a table contains a DBKEY pointing to a table row that is not in the index.

```
$ rmu/verify/index/key_values/ERROR_LIMIT=200 MF_PERSONNEL
%RMU-E-NOTIDXDBK, Table COLLEGES row dbkey 68:2:1 is not in index
 COLL_COLLEGE_CODE .
```

Example 16

The following example, using the KEY_VALUES qualifier, shows the error message put out if an index contains a DBKEY pointing to a table row that is not in the table.

```
$ rmu/verify/index/key_values/ERROR=200 MF_PERSONNEL
%RMU-E-NOTTABDBK, Index COLL_COLLEGE_CODE dbkey 68:2:1 is not in table
 COLLEGES .
```

Example 17

The following example, using the KEY_VALUES qualifier, shows that a warning message is put out for index key fields that have a collating sequence or that are of type character varying. These index fields cannot be compared with row values using the /KEY_VALUES qualifier.

```
$ rmu/verify/index/key_values ABC.RDB
%RMU-W-NOTTYPNDX, Index MANUFACTURING_INDEX field MANUFACTURER_NAME cannot
 be verified with
 the row value in table MANUFACTURING; reason - COLLATING SEQUENCE.
$ rmu/verify/index/key_values DBASE_INDEX.RDB
%RMU-W-NOTTYPNDX, Index FORECAST_HASH field PART_NO cannot be verified with
 the row value in table FORECAST_VOLUME; reason - CHARACTER VARYING.
```

2

RdbALTER Utility Command Syntax

This chapter describes the RdbALTER utility for Oracle Rdb. The RdbALTER utility provides a low-level patch capability that allows you to repair corruption on Oracle Rdb database pages. In addition, it allows you to relocate database root, storage area, and snapshot files to other disks or directories. This is helpful when you move a database from a single node to a clustered environment.

Note

Oracle Corporation recommends that the RdbALTER utility be used only to provide a temporary patch to a corrupt database. The RdbALTER utility should not be used as a routine database management tool.

Use the RdbALTER utility only after you fully understand the internal data structure, know the information the database should contain, and know the full effects of the command. Because of the power of the RdbALTER utility and the cascading effects it can have, Oracle Corporation recommends that you experiment on a copy of the damaged database before applying the RdbALTER utility to a production database.

Issue the RMU Alter command using the following format:

```
$ RMU/ALTER [root-file-spec]
```

The optional root-file-spec parameter identifies the database you want to alter. If you specify this parameter, you automatically attach to the specified database. If you do not specify this parameter, you must use the RdbALTER ATTACH command. See Section 2.2 for more information on the ATTACH command.

To use the RMU Alter command for a database, you must have the RMU$ALTER privilege in the root file access control list (ACL) for the database or the OpenVMS SYSPRV or BYPASS privilege.

If you are using an RMU Alter command to change a file name, you *must* have the OpenVMS SYSPRV or BYPASS privilege.

The RMU Alter command responds with the following prompt:

```
RdbALTER>
```

This prompt indicates that the system expects RdbALTER command input. The RdbALTER commands are:

- AREA . . . PAGE
- ATTACH
- COMMIT
- DEPOSIT
- DEPOSIT AREA_HEADER
- DEPOSIT FILE
- DEPOSIT ROOT
- DEPOSIT ROOT UNIQUE_IDENTIFIER
- DETACH
- DISPLAY
- DISPLAY AREA_HEADER
- DISPLAY FILE
- DISPLAY ROOT
- DISPLAY ROOT UNIQUE_IDENTIFIER
- EXIT
- HELP
- LOG
- MAKE CONSISTENT
- MOVE
- NOLOG
- PAGE

- RADIX
- ROLLBACK
- UNCORRUPT
- VERIFY

When you invoke RdbALTER and the database needs to be recovered, RdbALTER displays a message to that effect. If you receive this message, you should recover the database and verify it before making any changes in RdbALTER. If you make changes in RdbALTER before recovering the database, the recovery procedure may write over your changes.

After you have successfully completed making changes with RdbALTER, you should issue an RMU Backup command. RdbALTER commands are not recorded in the after-image journal (.aij) file, so it will not be possible to recover any changes made with RdbALTER should a database failure occur.

RdbALTER can receive command lines through command files or directly from your terminal.

You can abbreviate any of the RdbALTER command keywords. The only restriction is that you specify enough characters to avoid ambiguity.

RdbALTER interprets the exclamation point (!) as a comment character. If you want RdbALTER to disregard a line of characters, start the line with an exclamation point.

The remainder of this chapter describes the full syntax and usage of these commands. For detailed information on using the RdbALTER utility, see the *Oracle Rdb Guide to Database Maintenance*.

2.1 AREA . . . PAGE Command

Specifies a storage area, a snapshot area, or a page in an area of the database to which you are currently attached.

Description

Only one page of one area is accessible to RdbALTER at a time. This command switches you from one page of the currently attached database to another page.

When you attach to a database, RdbALTER automatically makes area 1 the current area and page 1 of that area the current page. To work on any other area and page, you must use the AREA . . . PAGE command.

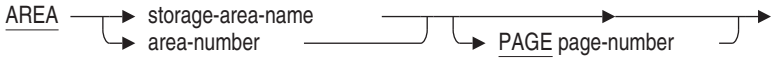
If you specify AREA but not PAGE, RdbALTER makes the area you specify current and fetches page 1 of that area.

If you specify both AREA and PAGE, RdbALTER makes the area you specify current and fetches the page you specify from the new current area.

If you specify an area or page that does not exist, an error occurs and the current area and page do not change.

Use the PAGE option to switch from one page in the current area to another page in the current area.

Format



Command Parameters

storage-area-name
Specifies a storage area of the current database by the storage area name, which is the name defined in the SQL CREATE STORAGE AREA statement. To specify a snapshot area, use the area-number parameter.

area-number
Specifies a storage area or snapshot area of the current database by the area number, which is assigned when the database is created and is given on the first line of a page display.

page-number
Identifies the area page to be altered. Express it as an integer from 1 to the number of pages in the area.

Examples

Example 1

The following example specifies the area JOBS of the currently attached database. No page number has been specified, so RdbALTER fetches page 1 of the area.

```
RdbALTER> AREA JOBS
```

Example 2

The following example is similar to Example 1, except it specifies the area by its area number instead of its area name. If area 4 is the area named JOBS in the database, either command produces the same result.

```
RdbALTER> AREA 4
```

Example 3

The following example fetches area 3, page 100:

```
RdbALTER> AREA 3 PAGE 100
```

2.2 ATTACH Command

Attaches RdbALTER to a database, putting an exclusive update lock on the database. No other user can access the database while the RdbALTER ATTACH command is in effect.

The database specified by the root-file-spec parameter is attached to RdbALTER. Then you can alter the pages of its storage area files. Area 1, page 1 of the database is fetched automatically and remains the current page until you issue an AREA . . . PAGE command.

Description

If the RMU Alter command includes a root-file-spec parameter, the database to which this root-file-spec refers is attached to as part of the RdbALTER startup. In this case, the ATTACH command is unnecessary. Otherwise, no commands changing database pages are allowed until an ATTACH command naming that database is issued.

You can use the ATTACH command to attach to only one database at a time. Before invoking another database for altering, you must use the DETACH command to detach from the current database. See Section 2.9 for more information on the DETACH command.

Format

Command Parameters

root-file-spec
Specifies the database root (.rdb) file whose pages you want to alter. The default file type is .rdb.

Examples

Example 1

The following example enters RdbALTER command level, and then attaches to the PERSONNEL database:

```
$ RMU/ALTER
RdbALTER> ATTACH PERSONNEL
%RMU-I-ATTACH, now altering database "DISK:[USER]PERSONNEL.RDB;1"
```

Example 2

The following example enters RdbALTER command level and attaches to the PERSONNEL database, using a single command:

```
$ RMU/ALTER PERSONNEL
%RMU-I-ATTACH, now altering database "DISK:[USER]PERSONNEL.RDB;1"
```

2.3 COMMIT Command

Writes all page changes back to the database since the last COMMIT or last ROLLBACK command was entered.

Description

The results of DEPOSIT and MOVE commands are kept in virtual memory until you issue a COMMIT or a ROLLBACK command. When you issue a COMMIT command, all pages that you changed since the last COMMIT or last ROLLBACK command are written to the database in their new form.

Changes are not permanent until you issue a COMMIT command. If you issue an EXIT command while altered but uncommitted pages exist, an error results. You cannot issue the EXIT command until you first issue either a COMMIT or a ROLLBACK command.

Oracle Corporation recommends that you make a backup copy of the database after you issue the RdbALTER COMMIT command.

Format

Examples

Example 1

The following example commits all page changes entered since the last COMMIT or last ROLLBACK command:

```
RdbALTER> COMMIT
```

2.4 DEPOSIT Command

Alters specified data fields on the current database page.

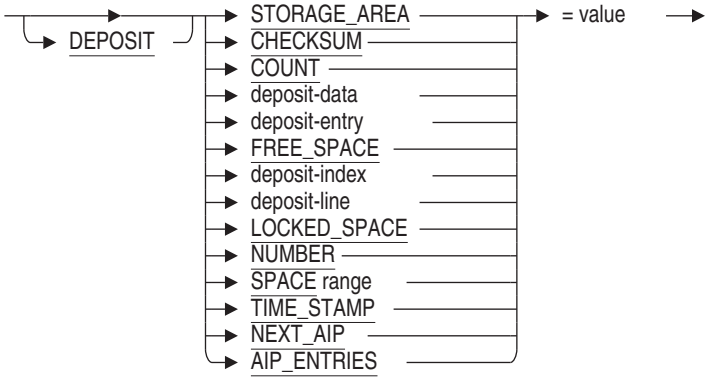
Description

The DEPOSIT command is the default at RdbALTER command level. If no other command is present following the prompt, RdbALTER automatically parses the command line as a DEPOSIT command.

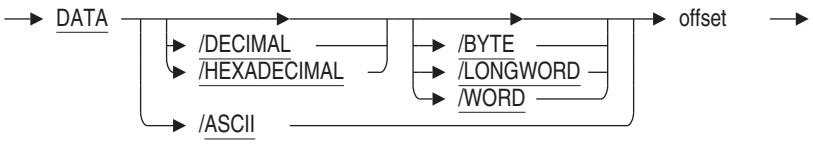
The specification of the field to be altered must be followed by an equal sign (=) and a string of characters specifying the new value of the altered field.

Do not use the DEPOSIT command immediately after a ROLLBACK command. The ROLLBACK command removes current page context. If you issue a DEPOSIT command immediately after a ROLLBACK command, a warning message is returned indicating that there is no current page. For this reason, you must specify your location again by using either the DISPLAY or the AREA . . . PAGE command immediately after a ROLLBACK command but before a DEPOSIT command.

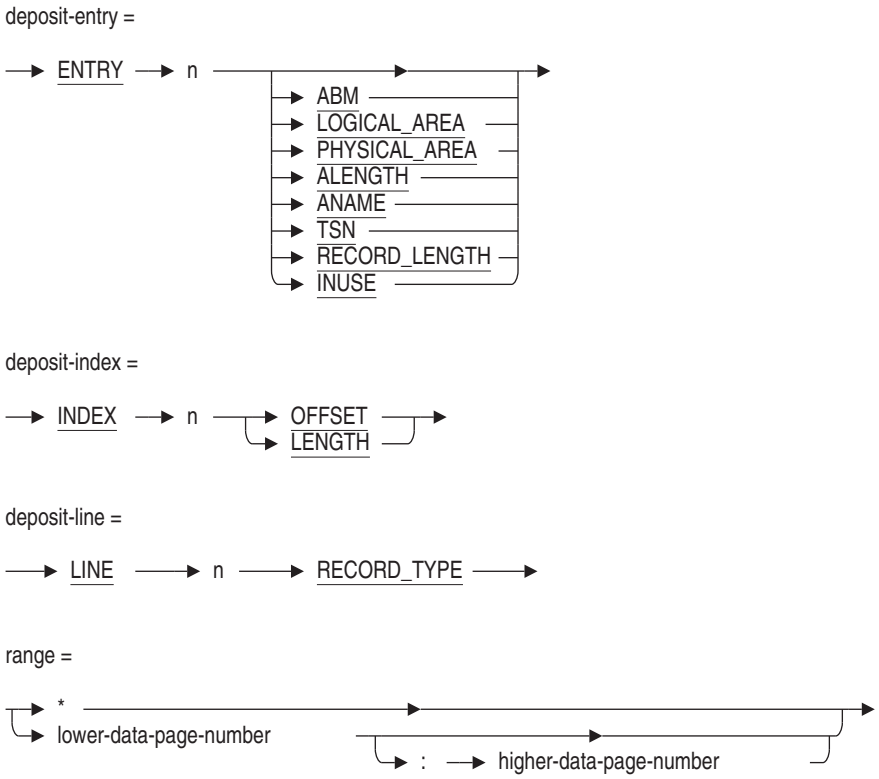
Format



deposit-data =



2.4 DEPOSIT Command



Command Parameters

STORAGE_AREA
Deposits a value for the 2-byte storage area identification.

CHECKSUM
Deposits a value for the 4-byte page checksum field.

COUNT
Deposits a value for the 2-byte field showing the number of line index entries. If this number is 1, the page contains only the SYSTEM record.

DATA offset
Deposits the number of bytes specified. If you do not specify the HEXADECIMAL or the DECIMAL qualifier, the default radix is assumed.

See Section 2.22 for information on how to set a default radix with the RADIX command.

The BYTE, LONGWORD, and WORD qualifiers cannot be used with the ASCII qualifier.

ENTRY
Refers to an area inventory page (AIP) entry on the database page. The value specified for *n* must be a number between zero and the number of AIP entries.

ABM
Deposits the new value on the first area bit map (ABM) page for the specified AIP entry. The ABM value is contained in a longword.

LOGICAL_AREA
Deposits the new value for the number of the logical area for the AIP entry. The LOGICAL_AREA value is contained in a word.

PHYSICAL_AREA
Deposits the new value for the number of the physical area for the AIP entry. The PHYSICAL_AREA value is contained in a word.

ALENGTH
Deposits the new value for the length of the name of the logical area for the AIP entry. The ALENGTH value is contained in 1 byte. The name of the logical area can be from 1 to 31 bytes in length.

ANAME
Deposits the new value for the name of the logical area for the AIP entry. The ANAME value is contained in a 31-character text field.

TSN
Deposits a value for the last transaction sequence number (TSN) to enable snapshot (.snp) files for the logical area of the AIP entry.

Note

Beginning in Oracle Rdb V7.0, Oracle Rdb stores any transaction sequence number that is larger than a longword by using both the TSN field on the page and the page TSN base. Oracle Rdb calculates the actual TSN by applying a formula to these two values. Oracle Corporation recommends that you do not change a TSN value that is larger than a longword. When a TSN is larger than a longword, a nonzero number is stored in the page TSN base (the page tail). The

following example shows the location of the page TSN and the page TSN base:

```
              000A 00000003  0000  page 3, physical area 10
                   9D091204  0006  checksum = 9D091204
          009A2C0F ED786D2E  000A  time stamp = 23-MAY-1996 09:08:53.36
                  0000 03C4  0012  964 free bytes, 0 locked
                       0001  0016  1 line
                  0005 03E4  0018  line 0: offset 03E4, 5 bytes

page TSN ----->    00000000  001C  line 0: TSN 0

   .
   .
   .
                      2001  03E4  line 0 (10:3:0) SYSTEM record
                    00 0001  03E6  1 byte in 0 sets/dynamic items
                  0000000000  03E9  padding '.....'

                    FFFFFFFF  03EE  snap page pointer -1
                    00000000  03F2  snap pointer TSN 0
                        0000  03F6  MBZ '..'
                    00000000  03F8  page sequence number 0
page TSN base -------> 0000  03FC  page TSN base 0
                       0000  03FE  MBZ '..'
```

RECORD_LENGTH
Deposits a value for the length, in bytes, of the record size for an AIP entry. The RECORD_LENGTH value is contained in a word.

INUSE
Deposits the new value for the AIP entry's in-use flag. The INUSE value is contained in 1 byte.

FREE_SPACE
Deposits a value for the 2-byte field indicating how much free space remains on the page.

Note

In the next two parameters, the integers denoting INDEX and LINE are zero based. For example, INDEX 0 refers to the first index, and LINE 3 refers to the fourth line.

References to INDEX and LINE are invalid if the current page is a space area management (SPAM) page, an AIP page, or an ABM page.

INDEX n
Deposits a value for the offset field or the length field for the line index indicated by *n*. For example, if you enter DEPOSIT INDEX 3 OFFSET, the offset address field from the fourth-line index is deposited.

LINE n
Deposits information for an individual storage segment. You can deposit a value for the RECORD_TYPE field.

LOCKED_SPACE
Deposits a value for the 2-byte field indicating how much free space is allocated for exclusive use by a recovery unit.

NUMBER
Deposits a value for the 4-byte page number field.

SPACE range
Deposits a value for a specified range of SPAM entries; it is valid only if the current page is a SPAM page. (The SPACE and DATA parameters are the only ones that you can use in DISPLAY and DEPOSIT commands that access a SPAM page.) The range value can be an asterisk (*), referring to all entries, or a set of consecutive entries, which you describe as follows:

```
lower-data-page-number[:higher-data-page-number]
```

Each entry on a SPAM page consists of 2 bits, containing a value 0 through 3 that represents a fullness threshold. For example, if the *n*th SPAM entry contains a 2, it means that the *n*th data page in the interval has reached a percentage of fullness greater than the second threshold for the area, but less than or equal to the third threshold.

TIME_STAMP
Deposits a value for the 8-byte time and date stamp field.

NEXT_AIP
Deposits a value for the page number of the next area inventory page (AIP).

AIP_ENTRIES
Deposits a value for the number of area inventory page (AIP) entries on the current area inventory page.

value
Specifies the new value of the field you are altering. The value is deposited in the default radix unless you specify otherwise in one of these ways:

- With a prior RADIX command.

- By specifying HEXADECIMAL or DECIMAL in a DEPOSIT DATA command.
- By enclosing ASCII data within quotation marks (" "). Timestamps must always be enclosed within quotation marks because they include punctuation characters.

Examples

Example 1

The following example deposits a value of 1 for SPAM entries on pages 2 and 3 of the current storage area:

```
RdbALTER> AREA 6 PAGE 1
RdbALTER> DISPLAY SPACE 2:3
00000000000000000000000000000000B  0016  page 2: threshold 3
                                         page 3: threshold 2
                                         pages 4-65: threshold 0
RdbALTER> DEPOSIT SPACE 2:3=1
RdbALTER> DISPLAY SPACE 2:3
00000000000000000000000000000005  0016  pages 2-3: threshold 1
                                         pages 4-65: threshold 0
```

Example 2

The following example deposits data 032C with a length of 2 bytes into offset location 0012 (decimal 18):

```
RdbALTER> DEPOSIT DATA/HEXADECIMAL/WORD 0012=032C
```

Example 3

The following example deposits the new value of 0002 in the storage area identification field:

```
RdbALTER> AREA 1 PAGE 1
RdbALTER> DISPLAY STORAGE_AREA
                          0001  0000  area 1
RdbALTER> DEPOSIT STORAGE_AREA=0002
RdbALTER> DISPLAY STORAGE_AREA
                          0002  0000  area 2
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples of how to use the DEPOSIT command.

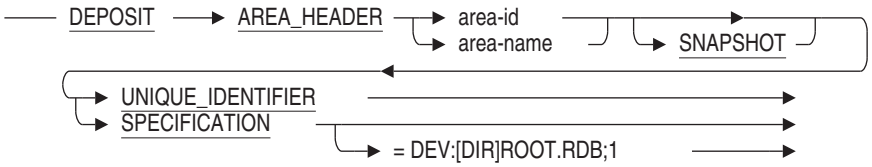
2.5 DEPOSIT AREA_HEADER Command

Stores the current database root UNIQUE_IDENTIFIER value into the current storage area file or storage area snapshot file header for the storage area with the specified name or number when the user executes the next COMMIT command OR deposits the current (original or changed) full root file specification in the database root in the storage area or snapshot file header block for the storage area with the specified name or number. Any specified root file must exist or must have been changed by a previous RMU/ALTER DEPOSIT ROOT SPECIFICATION command. Only full VMS root file specifications are valid and must include a device, directory, extension and version number.

Any changes to the area header UNIQUE_IDENTIFIER values will only be written to the actual area files when the next COMMIT command is executed at the "RdbALTER>" prompt. Any changes to the area file headers since the last COMMIT command was issued can be undone by executing the ROLLBACK command at the "RdbALTER>" prompt. COMMIT and ROLLBACK are existing RMU/ALTER commands and affect any current uncommitted changes made in RMU/ALTER, not just changes to the storage area header UNIQUE_IDENTIFIER values.

To execute the DEPOSIT AREA_HEADER command, the user must be attached to the database which the root and areas belong to, either by specifying the database name when issuing the RMU/ALTER command or by executing the ATTACH command from the "RdbALTER>" prompt.

Format



Command Parameters

name or id
Specifies the name or number of the .RDA or .SNP file where you are storing the current database root UNIQUE_IDENTIFIER value or database root file specification.

2.5 DEPOSIT AREA_HEADER Command

SNAPSHOT
Specifies that the file whose .RDB file fields you are changing is an .SNP file. If Snapshot is not specified, the storage area .RDA file will be modified.

UNIQUE_IDENTIFIER
Specifies that you are storing the UNIQUE_IDENTIFIER in the storage area or the storage area snapshot header.

SPECIFICATION[=DEV:[DIR]ROOT.RDB;1
Deposits the current (original or changed) full root file specification in the database root in the storage area or snapshot file header block for the storage area with the specified name or number. Any specified root file must exist or must have been changed by a previous RMU/ALTER DEPOSIT ROOT SPECIFICATION command. Only full VMS root file specifications are valid and must include a device, directory, extension and version number.

Thisfeature only allows modification of the root file specification in area headers, not other area header data. The "DISPLAY AREA_HEADER" can be used with single file databases but the root file specification will always be blank, which is standard for single file databases. The "DEPOSIT AREA_ HEADER" command cannot be used with single file databases since a root file specification is not specified in area headers in single file databases. To use either the DISPLAY or DEPOSIT AREA_HEADER command, the user must be attached to the database which the areas belong to, either by specifying the database name when issuing the RMU/ALTER command or by executing the "ATTACH" command from the "RdbALTER>" prompt.

Examples

Example 1

The following example shows that for Oracle Rdb single file databases, the UNIQUE_IDENTIFIER value can only be set for the storage area snapshot file since the storage area is part of the root file. Therefore, the DEPOSIT AREA_HEADER command can only specify the SNAPSHOT file or an error will be returned.

```
$ RMU/ALTER PERSONNEL
%RMU-I-ATTACH, now altering database
"DEVICE:[DIRECTORY]PERSONNEL.RDB;1"
  DEPOSIT AREA_HEADER RDB$SYSTEM UNIQUE_IDENTIFIER
%RMU-F-NOTSFDB, This command is not allowed for a single file
 database
  DEPOSIT AREA_HEADER RDB$SYSTEM SNAPSHOT UNIQUE_IDENTIFIER
Area RDB$SYSTEM:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:29.32"
 (00AA557869612643)
```

```
 COMMIT
 EXIT
```

Example 2

Since the DEPOSIT ROOT UNIQUE_IDENTIFIER command always stores the UNIQUE_IDENTIFIER value in ALL storage area file headers when the user executes the RMU/ALTER COMMIT command, it would be redundant to execute the DEPOSIT AREA_HEADER UNIQUE_IDENTIFIER command if a DEPOSIT ROOT UNIQUE_IDENTIFIER command is already pending for the current RMU/ALTER session. Therefore, as the following example shows, in this case a DEPOSIT AREA_HEADER UNIQUE_IDENTIFIER command cannot be executed until the user ends the current session with a COMMIT or ROLLBACK command.

```
$ RMU/ALTER MF_PERSONNEL
%RMU-I-ATTACH, now altering database
 "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DEPOSIT ROOT UNIQUE_IDENTIFIER = NEW
(marked) Root file unique identifier is: "22-OCT-2010 13:49:31.72"
 (00AA55786ACFB115)

  DEPOSIT AREA_HEADER SALARY_HISTORY UNIQUE_IDENTIFIER
%RMU-F-COMROOTCOM, COMMIT or ROLLBACK DEPOSIT ROOT
 UNIQUE_IDENTIFIER command to use this command
  COMMIT
  DEPOSIT AREA_HEADER SALARY_HISTORY UNIQUE_IDENTIFIER
Area SALARY_HISTORY:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:31.72"
 (00AA55786ACFB115)

  COMMIT
  EXIT
```

Example 3

The following example shows that RMU/ALTER is invoked specifying the database MF_PERSONNEL.RDB. The user then displays the current UNIQUE_IDENTIFIER value in the root file. He then executes the DEPOSIT commands to designate that the UNIQUE_IDENTIFIER value in the root file is to be moved to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files, displays the UNIQUE_IDENTIFIER value that is to be moved to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files, and finally specifies COMMIT to actually write the root UNIQUE_IDENTIFIER value to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files. The display messages designate the pending UNIQUE_IDENTIFIER value as "(marked)" until the user either executes COMMIT to write out the UNIQUE_IDENTIFIER value or ROLLBACK to restore the original UNIQUE_IDENTIFIER value. The user

then verifies the database changes. The example shows that the user can use either the storage area name or the storage area identifier number in the root to designate the target storage area.

```
$ RMU/ALTER MF_PESONNEL
%RMU-I-ATTACH, now altering database "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DISPLAY ROOT UNIQUE_IDENTIFIER
           Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DEPOSIT AREA_HEADER DEPARTMENTS UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DEPOSIT AREA_HEADER 2 SNAPSHOT UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DISPLAY AREA_HEADER DEPARTMENTS UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DISPLAY AREA_HEADER 2 SNAPSHOT UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  COMMIT
  EXIT
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
```

Example 4

The following example shows that RMU/ALTER is invoked specifying the multi-file database MULTIFILE_DB.RDB which has been previously backed up using RMU/BACKUP. The DEPOSIT ROOT SPECIFICATION command is used to change the full root file specification in the root file to DEVICE:[DIRECTORY]NEW_ROOT.RDB;1. Then the DEPOSIT AREA_ HEADER command is used without a root file specification to change the root file specification in the storage area header to the current root file specification set by the previous DEPOSIT ROOT SPECIFICATION command for both the ST_AREA.RDA and ST_AREA.SNP files. Then the DEPOSIT AREA_HEADER command is used with a full root file specification "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1" for both the ST_AREA.RDA and ST_AREA.SNP files. This just repeats the previous change but is included to show the use of the DEPOSIT AREA_HEADER COMMAND with and without an explicit root file specification. Then a COMMIT command is used to write the changes to the database files. After exiting RMU/ALTER, the name of the

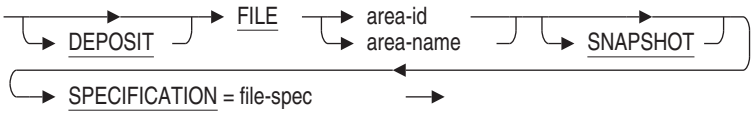
old root file is changed to DEVICE:[DIRECTORY]NEW_ROOT.RDB;1 from the VMS prompt. Then the RMU/VERIFY command is used to verify the integrity of the database. The DISPLAY AREA_HEADER command is used throughout to see the current root file specification in the storage area or snapshot file headers. In the display output "(marked)" means a change has been made but has not yet been comitted. Note that although this is not shown, in this case the area headers of all storage area and snapshot files in the database need to be changed to contain the new root file specification.

```
$ rmu/alter device:[directory]multifile_db
%RMU-I-ATTACH, now altering database "DEVICE:[DIRECTORY]MULTIFILE_DB.RDB;1"

RdbALTER>  deposit root specification = DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
         Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"
RdbALTER> display root specification
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> display area_header st_area specification
Area ST_AREA:
         Root file specification is: "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"

RdbALTER> display area_header st_area snapshot specification
Area ST_AREA:
         Root file specification is: "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"

RdbALTER> deposit area_header st_area specification
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> deposit area_header st_area snapshot specification
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> deposit area_header st_area specification =
 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> deposit area_header st_area snapshot specification =
 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> commit
RdbALTER> exit

$ RENAME DEVICE:[DIRECTORY]MULTIFILE_DB.RDB;1 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
$ RMU/VERIFY/ALL DEVICE:[DIRECTORY]NEW_ROOT.RDB
```

2.6 DEPOSIT FILE Command

Puts a new file specification into the database root (.rdb) file for a storage area (.rda) or snapshot (.snp) file.

Format



Command Parameters

area-id
Specifies the number of the .rda or .snp file whose file specification you are changing in the .rdb file.

area-name
Specifies the name of the .rda or .snp file whose file specification you are changing in the .rdb file.

SNAPSHOT
Specifies that the file whose .rdb file fields you are changing is an .snp file.

SPECIFICATION = file-spec
Specifies the file specification that the .rda or .snp file will have. Use a full file specification (including version number) if the .rda or .snp file is not in your default directory. The file must be in the location you specify, otherwise the DEPOSIT FILE command fails.

Examples

Example 1

The following example deposits a new file specification for the JOBS storage area file. The word *(marked)* in the DEPOSIT FILE display indicates that the JOBS storage area file is marked for the specified location.

```
RdbALTER> DISPLAY FILE JOBS
Area JOBS:
         File specification is: "DISK1:[RICK.RDB]JOBS.RDA;1"
         Corrupt flag is: OFF
         Inconsistent flag is: OFF

RdbALTER> DEPOSIT FILE JOBS SPECIFICATION=DISK1:[RICK]JOBS.RDA;1
Area JOBS:
(marked) File specification is: "DISK1:[RICK]JOBS.RDA;1"
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples of how to use the DEPOSIT FILE command.

2.7 DEPOSIT ROOT Command

Enters new file specifications for database files. The file specification you enter will be the file specification after the change is committed to the database. The change does not occur until you have committed all changes and ended the RdbALTER session.

Note

Prior to Oracle Rdb Version 6.0, you could use the DEPOSIT ROOT command to alter an after-image journal (.aij) file name. This is no longer an option; use the RMU Set After_Journal command instead.

Description

The default is DEPOSIT ROOT SPECIFICATION if no other keyword follows the DEPOSIT ROOT command.

The DEPOSIT ROOT command allows a user to remove recovery-unit journal (.ruj) file references from the database root (.rdb) file, allowing access to the database. However, this action will mark the database as "eternally corrupt," which results in a warning message in all future system management (Oracle RMU) functions against the database. The database is either structurally corrupt, logically corrupt (violates a constraint), or "user-data" corrupt (for example, balance is $250.00 instead of $300.00).

Removing the .ruj file references from the .rdb file permits users to attach to the database in situations where the .ruj files have been accidentally deleted. Users must realize that the database is corrupt when they attach. The database must be restored and recovered from clean backup files to guarantee the consistency of the database contents.

Use the USER n RUJ_FILENAME="" clause to remove an .ruj file reference from the .rdb file.

Note

After you commit a changed .rdb file name, the database cannot be accessed until you copy it to the location (using all the necessary qualifiers for the file specification) that you specify with the DEPOSIT

ROOT command. After that, users attach to the database at the new location.

Format

Command Parameters

SPECIFICATION

Enters the new file specification of the .rdb file.

USER n RUJ_FILENAME

Enters the new file specification for the .ruj file for the specified user. In this syntax, *n* is a valid user number.

file-spec

Specifies the full file specification (including version number) that the .rdb file will have after it has been committed and the RdbALTER session is complete.

The file specification can be a set of double quotation marks ("") if you are removing the .ruj file specification.

Examples

Example 1

The following example enters a new file specification for the .rdb file. You must specify a version number in the new file specification for the .rdb file. The word *(marked)* in the DEPOSIT ROOT display indicates that the .rdb file is marked for the specified location but has not been moved to that location.

```
RdbALTER> DISPLAY ROOT
        Root file specification is: "DISK1:[RICK.RDB]MF_PERSONNEL.RDB;1"
RdbALTER> DEPOSIT ROOT SPECIFICATION="DISK1:[RICK]MF_PERSONNEL.RDB;1"
(marked) Root file specification is: "DISK1:[RICK]MF_PERSONNEL.RDB;1"
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples of how to use the DEPOSIT ROOT command.

2.8 DEPOSIT ROOT UNIQUE_IDENTIFIER Command

Enters the current database root UNIQUE_IDENTIFIER into the storage area header blocks of ALL active storage area and storage area snapshot files which are currently defined in the database root or creates a new database root UNIQUE_IDENTIFIER and then enters it into the storage area header blocks of ALL active storage area and storage area snapshot files which are currently defined in the database root.

Description

To ensure Oracle Rdb database security and integrity, a UNIQUE_ IDENTIFIER has been added to the database root file and the database storage area file and storage area snapshot file headers. The UNIQUE_IDENTIFIER in the root file must match the UNIQUE_IDENTIFIER in the storage area file headers or a storage area cannot be accessed from the database root.

The UNIQUE_IDENTIFIER values are displayed both in VMS date format surrounded by quotes and as a hexadecimal number surrounded by parentheses. The values displayed are the UNIQUE_IDENTIFIER values for the current RMU/ALTER session. The UNIQUE_IDENTIFIER values will not be written to the root or storage area files until the user ends the current session with the RMU/ALTER COMMIT command. If the user ends the current session with the RMU/ALTER ROLLBACK command, the UNIQUE_ IDENTIFIER values will not be written to the root or storage area files and the UNIQUE_IDENTIFIER values in effect at the start of the session just ended will be restored for the new session. Any UNIQUE_IDENTIFIER values that have been changed during the current session will be displayed as "(marked)" before they are committed or rolled back.

If "= NEW" is not specified, this command stores the current database root UNIQUE_IDENTIFIER value into the storage area header blocks of ALL active storage area and storage area snapshot files which are currently defined in the database root when the user executes the next COMMIT command. If "= NEW" is specified, a new UNIQUE_IDENTIFIER value is created and stored in both the root file and ALL active storage area file headers when the user executes the next COMMIT command. Note that to ensure database integrity, ALL storage area file headers will be updated.

To execute the DISPLAY or DEPOSIT ROOT command, the user must be attached to the database which the root and areas belong to, either by specifying the database name when issuing the RMU/ALTER command or by executing the ATTACH command from the "RdbALTER>" prompt.

2.8 DEPOSIT ROOT UNIQUE_IDENTIFIER Command

Format

Command Parameters

= NEW
If "= NEW" is specified, a new UNIQUE_IDENTIFIER value is created and stored in both the root file and ALL active storage area file headers when the user executes the next COMMIT command. Note that to ensure database integrity, ALL storage area file headers will be updated. Use the AREA_ HEADER commands described elsewhere in this chapter for storing the current root UNIQUE_IDENTIFIER value in specific designated storage areas.

If "= NEW" is not specified, this command stores the current database root UNIQUE_IDENTIFIER value into the storage area header blocks of ALL active storage area and storage area snapshot files which are currently defined in the database root when the user executes the next COMMIT command.

Examples

Example 1

The following example shows how to enter a new UNIQUE_IDENTIFIER value using RMU/ALTER.

```
$ RMU/ALTER MF_PERSONNEL
%RMU-I-ATTACH, now altering database
 "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DEPOSIT ROOT UNIQUE_IDENTIFIER = NEW
(marked) Root file unique identifier is: "22-OCT-2010 13:49:31.72"
 (00AA55786ACFB115)
  COMMIT
  EXIT
```

Example 2

The following example shows that RMU/ALTER is invoked specifying the database MF_PERSONNEL.RDB. The user then displays the current UNIQUE_ IDENTIFIER value in the database root, creates a new UNIQUE_ IDENTIFIER value in the database root, displays the new UNIQUE_ IDENTIFIER in the root, and finally specifies COMMIT to write the new UNIQUE_IDENTIFIER value to the database root file and ALL database storage area files. The display messages designate the pending new UNIQUE_ IDENTIFIER value as "(marked)" until the user either executes COMMIT to

write out the new UNIQUE_IDENTIFIER value or ROLLBACK to restore the original UNIQUE_IDENTIFIER value. The user then verifies the database changes.

```
$ RMU/ALTER MF_PERSONNEL
%RMU-I-ATTACH, now altering database "DISK:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DISPLAY ROOT UNIQUE_IDENTIFIER
         Root file unique identifier is: "22-OCT-2010 13:49:27.87"
 (00AA5578688428BB)
  DEPOSIT ROOT UNIQUE_IDENTIFIER = NEW
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)
  DISPLAY ROOT UNIQUE_IDENTIFIER
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)
  COMMIT
  EXIT
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
```

2.9 DETACH Command

Releases a previous database ATTACH command. The exclusive update lock is released, and other users can access the database again. Any uncommitted transactions are rolled back. No database page alterations are allowed until another ATTACH command is issued.

Format

DETACH →

Examples

Example 1

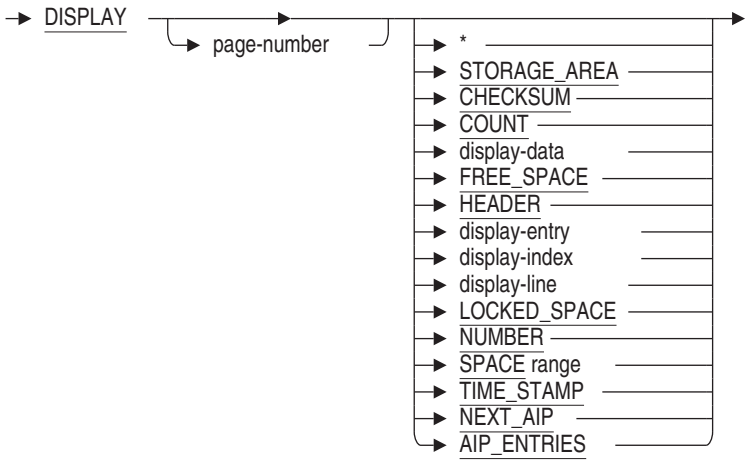
The following command detaches RdbALTER from the current database:

```
RdbALTER> DETACH
```

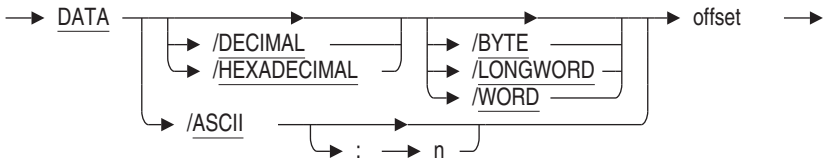
2.10 DISPLAY Command

Requests display of data fields from a database page. An individual DISPLAY command can include only one of the display options shown. If no parameters are specified (you only specify DISPLAY), RdbALTER displays the last object of a PAGE, a DISPLAY, or a DEPOSIT command.

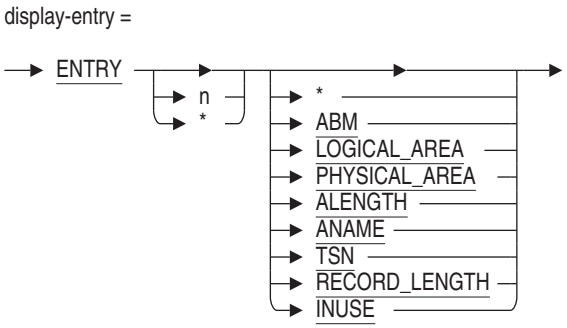
Format



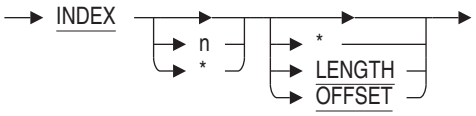
display-data =



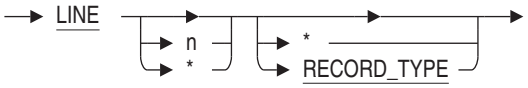
display-entry =



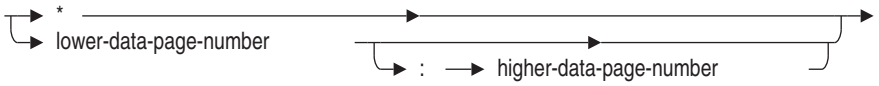
display-index =



display-line =



range =



Command Parameters

page-number
Identifies the page whose information you want to display. The current page is the default.

*** (asterisk)**
Displays the entire page.

STORAGE_AREA
Displays the 2-byte storage area identification.

2.10 DISPLAY Command

CHECKSUM
Displays the 4-byte page checksum field.

COUNT
Displays the 2-byte field showing the number of line index entries. If this number is 1, the page contains only the SYSTEM record.

DATA offset
Displays the number of bytes specified. If you do not specify the HEXADECIMAL or DECIMAL qualifier, the default radix is assumed. See Section 2.22 for information on how to set a default radix by using the RADIX command.

The value you specify for *n* is the number of bytes you want to display. The maximum value that can be specified for *n* is the size of the page minus the offset. If you do not specify *n*, 1 byte is displayed.

The BYTE, LONGWORD, and WORD qualifiers cannot be used with the ASCII qualifier.

FREE_SPACE
Displays the 2-byte field indicating how much free space remains on the page.

HEADER
Displays the entire page header.

ENTRY (n | *)
Refers to an area inventory page (AIP) entry on the database page. If you specify a value for *n*, it must be a number between zero and the number of AIP entries. If you specify the asterisk (*) parameter, the information you request with the other DISPLAY ENTRY parameters is displayed for each AIP entry on the database page.

asterisk (*)
Displays the same information that is displayed when you specify the ABM, LOGICAL_AREA, PHYSICAL_AREA, ALENGTH, ANAME, TSN, RECORD_ LENGTH, and INUSE parameters.

ABM
Displays the first area bit map (ABM) page for the specified area inventory page (AIP) entry. The value is contained in a longword.

LOGICAL_AREA
Displays the number of the logical area of an AIP entry. The value is contained in a word.

PHYSICAL_AREA
Displays the number of the physical area of an AIP entry. The value is contained in a word.

ALENGTH
Displays the length of the name of the logical area of an AIP entry. The value is contained in 1 byte. The name of the logical area can be from 1 to 31 bytes in length.

ANAME
Displays the name of the logical area of an AIP entry. The value is contained in a 31-character text field.

TSN
Displays the value of the last transaction sequence number (TSN) to enable snapshots for the logical area of an AIP entry.

Note

Beginning in Oracle Rdb V7.0, Oracle Rdb stores any transaction sequence number that is larger than a longword by using both the TSN field on the page and the page TSN base. Oracle Rdb calculates the actual TSN by applying a formula to these two values. Oracle Corporation recommends that you do not change a TSN value that is larger than a longword. When a TSN is larger than a longword, a nonzero number is stored in the page TSN base (the page tail). The following example shows the location of the page TSN and the page TSN base:

```
              000A 00000003  0000  page 3, physical area 10
                   9D091204  0006  checksum = 9D091204
          009A2C0F ED786D2E  000A  time stamp = 23-MAY-1996 09:08:53.36
                  0000 03C4  0012  964 free bytes, 0 locked
                       0001  0016  1 line
                  0005 03E4  0018  line 0: offset 03E4, 5 bytes

page TSN ----->    00000000  001C  line 0: TSN 0
   .
   .
   .
                       2001  03E4  line 0 (10:3:0) SYSTEM record
                     00 0001  03E6  1 byte in 0 sets/dynamic items
                 0000000000  03E9  padding '.....'
```

```
                    FFFFFFFF  03EE  snap page pointer -1
                    00000000  03F2  snap pointer TSN 0
                        0000  03F6  MBZ '..'
                    00000000  03F8  page sequence number 0
page TSN base -------> 0000  03FC  page TSN base 0
                        0000  03FE  MBZ '..'
```

RECORD_LENGTH
Displays the value for the length of the record size for an AIP entry. The length is expressed in bytes. The RECORD_LENGTH value is contained in a word.

INUSE
Displays the AIP entry's in-use flag. The value is contained in 1 byte.

Note

In the next two parameters, the integers denoting INDEX and LINE are zero based. For example, INDEX 0 refers to the first index, and LINE 3 refers to the fourth line.

The integer *n* is optional. The present value of the relevant pointer is the default.

References to INDEX and LINE are invalid if the current page is a space area management (SPAM) page.

INDEX n
Displays the offset field, the length field, or both from the line index indicated by *n*. For example, if you enter DISPLAY INDEX 3 OFFSET, the offset address field from the fourth line index is displayed. If you enter DISPLAY INDEX 3 LENGTH, the length field from the fourth line index is displayed. If you enter either DISPLAY INDEX 3 or DISPLAY INDEX 3 *, both the offset and the length fields from the fourth line index are displayed.

INDEX *
Displays the offset field and the length field for all index nodes on a page.

LINE n
Displays information from an individual storage segment. You can display the RECORD_TYPE field or the entire content of the storage segment line indicated by *n*.

LINE *
Displays information from all storage segments on a page.

LOCKED_SPACE
Displays the 2-byte field indicating how much free space is allocated for exclusive use by a recovery unit.

NUMBER
Shows the 4-byte page number field.

SPACE range
Displays SPAM entries; it is valid only if the current page is a SPAM page. The SPACE parameter is the only option that you can use in DISPLAY and DEPOSIT commands that access a SPAM page. The optional range value can be an asterisk (*), referring to all entries, or a set of consecutive entries that you describe as follows:

```
lower-data-page-number[:higher-data-page-number]
```

When you specify a range, you reduce the output display. The specified range of SPAM entries is included in the display; other SPAM entries outside your specified range can also be included in the display.

Each entry on a SPAM page consists of 2 bits, containing a value 0 through 3 that represents a fullness threshold. For example, if the *n*th SPAM entry contains a 2, it means that the *n*th data page in the interval has reached a percentage of fullness greater than the second threshold for the area, but less than or equal to the third threshold.

TIME_STAMP
Displays the 8-byte time stamp field.

NEXT_AIP
Displays the page number of the next area inventory page (AIP).

AIP_ENTRIES
Displays a value for the number of area inventory page (AIP) entries on the current area inventory page.

Examples

Example 1

The following example displays page 94 of area 1:

```
RdbALTER> AREA 1
RdbALTER> DISPLAY 94
                  0001 0000005E  0000  page 94, physical area 1
                       7B429FB0  0006  checksum = 7B429FB0
              0093249F 9EBDB820  000A  time stamp = 14-FEB-1990 10:42:35.81
                      0000 0004  0012  4 free bytes, 0 locked

                       FFFFFFFF  0016  next area bit map page -1
                       00000000  001A  max set bit index 0
                       00000000  001E  MBZ '....'
                       00001E60  0022  bitvector count 7776

00000000000000000000000000000000  0026  bitvector '................'
                                  ::::  (59 duplicate lines)
        00000000000000000000000000  03E6  bitvector '............'

                       00000000  03F2  MBZ '....'

                           801F  03F6  bitmap page for logical area 31
               0000000000000000  03F8  MBZ '........'
```

Example 2

The following example displays the number of the current storage area:

```
RdbALTER> DISPLAY STORAGE_AREA
                             0001  0000  area 1
```

Example 3

This example displays the number of AIP entries on area 1, page 2 of the mf_personnel database, and the value of the highest snapshot-enabled TSN of AIP entry 14. This example also displays the number of the next AIP in the current storage area.

```
RdbALTER> AREA 1 PAGE 2
RdbALTER> DISPLAY AIP_ENTRIES
                             0010  0022  16 logical area entries
RdbALTER> DISPLAY ENTRY 14 TSN

                                         entry #14
                          00000001  0386  snaps enabled TSN 1
RdbALTER> DISPLAY NEXT_AIP
                          00000003  0016  next area inventory page 3
```

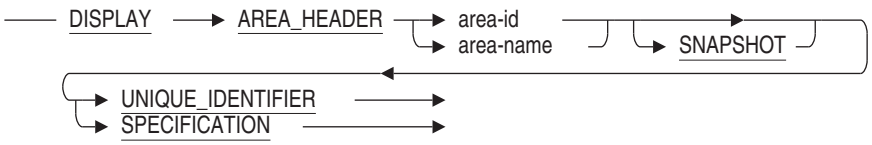
See the *Oracle Rdb Guide to Database Maintenance* for more examples of how to use the DISPLAY command.

2.11 DISPLAY AREA_HEADER Command

Displays the current storage area file or storage area snapshot file header UNIQUE_IDENTIFIER value for the storage area with the specified name or number OR displays the current (original or changed) full root file specification in the storage area file or snapshot file header block for the storage area with the specified name or number.

To execute the DISPLAY AREA_HEADER command, the user must be attached to the database which the root and areas belong to, either by specifying the database name when issuing the RMU/ALTER command or by executing the ATTACH command from the "RdbALTER>" prompt.

Format



Command Parameters

area or id
Specifies the name or number of the storage area for which you want to display the UNIQUE_IDENTIFIER value or the full root file specification in the storage are file or snapshot file header block for the storage area with the specified name or number.

SNAPSHOT
Displays the UNIQUE_IDENTIFIER value for the .SNP file. If SNAPSHOT is not specified, the UNIQUE_IDENTIFIER value for the .RDA file will be displayed.

UNIQUE_IDENTIFIER
Specifies that you are displaying the UNIQUE_IDENTIFIER value for the storage area file or storage area snapshot file.

SPECIFICATION

Displays the current (original or changed) full root file specification in the storage area file or snapshot file header block for the storage area with the specified name or number. This

only allows modification of the root file specification in area headers, not other area header data. The "DISPLAY AREA_HEADER" can be used with single file databases but the root file specification will always be blank, which is standard for single file databases.

Examples

Example 1

The following example shows that RMU/ALTER is invoked specifying the database MF_PERSONNEL.RDB. The user then displays the current UNIQUE_IDENTIFIER value in the root file. He then executes the DEPOSIT commands to designate that the UNIQUE_IDENTIFIER value in the root file is to be moved to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files, displays the UNIQUE_IDENTIFIER value that is to be moved to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files, and finally specifies COMMIT to actually write the root UNIQUE_IDENTIFIER value to the DEPARTMENTS area storage and the DEPARTMENTS area snapshot files. The display messages designate the pending UNIQUE_IDENTIFIER value as "(marked)" until the user either executes COMMIT to write out the UNIQUE_IDENTIFIER value or ROLLBACK to restore the original UNIQUE_IDENTIFIER value. The user then verifies the database changes. The example shows that the user can use either the storage area name or the storage area identifier number in the root to designate the target storage area.

```
$ RMU/ALTER MF_PESONNEL
%RMU-I-ATTACH, now altering database "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DISPLAY ROOT UNIQUE_IDENTIFIER
          Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DEPOSIT AREA_HEADER DEPARTMENTS UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DEPOSIT AREA_HEADER 2 SNAPSHOT UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DISPLAY AREA_HEADER DEPARTMENTS UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)

  DISPLAY AREA_HEADER 2 SNAPSHOT UNIQUE_IDENTIFIER
Area DEPARTMENTS:
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)
```

```
  COMMIT
  EXIT
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
```

Example 2

The following example shows that RMU/ALTER is invoked specifying the multi-file database MULTIFILE_DB.RDB which has been previously backed up using RMU/BACKUP. The DEPOSIT ROOT SPECIFICATION command is used to change the full root file specification in the root file to DEVICE:[DIRECTORY]NEW_ROOT.RDB;1. Then the DEPOSIT AREA_ HEADER command is used without a root file specification to change the root file specification in the storage area header to the current root file specification set by the previous DEPOSIT ROOT SPECIFICATION command for both the ST_AREA.RDA and ST_AREA.SNP files. Then the DEPOSIT AREA_HEADER command is used with a full root file specification "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1" for both the ST_AREA.RDA and ST_AREA.SNP files. This just repeats the previous change but is included to show the use of the DEPOSIT AREA_HEADER COMMAND with and without an explicit root file specification. Then a COMMIT command is used to write the changes to the database files. After exiting RMU/ALTER the name of the old root file is changed to DEVICE:[DIRECTORY]NEW_ROOT.RDB;1 from the VMS prompt. Then the RMU/VERIFY command is used to verify the integrity of the database. The DISPLAY AREA_HEADER command is used throughout to see the current root file specification in the storage area or snapshot file headers. In the display output "(marked)" means a change has been made but has not yet been comitted. Note that although this is not shown, in this case the area headers of all storage area and snapshot files in the database need to be changed to contain the new root file specification.

```
$ rmu/alter device:[directory]multifile_db
%RMU-I-ATTACH, now altering database "DEVICE:[DIRECTORY]MULTIFILE_DB.RDB;1"

RdbALTER>  deposit root specification = DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
         Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> display root specification
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> display area_header st_area specification
Area ST_AREA:
         Root file specification is: "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"

RdbALTER> display area_header st_area snapshot specification
Area ST_AREA:
         Root file specification is: "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"

RdbALTER> deposit area_header st_area specification
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"
```

2.11 DISPLAY AREA_HEADER Command

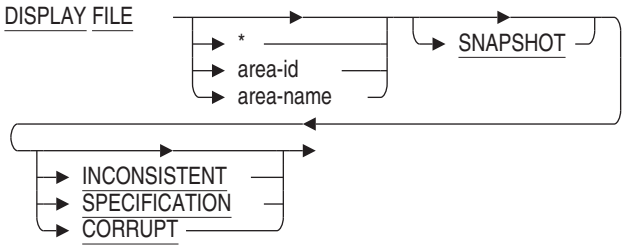
```
RdbALTER> deposit area_header st_area snapshot specification
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> deposit area_header st_area specification =
 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> deposit area_header st_area snapshot specification =
 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
Area ST_AREA:
(marked) Root file specification is: "DEVICE:[DIRECTORY]NEW_ROOT.RDB;1"

RdbALTER> commit
RdbALTER> exit

$ RENAME DEVICE:[DIRECTORY]MULTIFILE_DB.RDB;1 DEVICE:[DIRECTORY]NEW_ROOT.RDB;1
$ RMU/VERIFY/ALL DEVICE:[DIRECTORY]NEW_ROOT.RDB
```

2.12 DISPLAY FILE Command

Displays the file specification in the database root (.rdb) file for a storage area (.rda) or snapshot (.snp) file. You can also use this command to display the current setting of the inconsistent flag or corrupt flag for a storage file.

If you specify the DISPLAY FILE command, but do not specify any parameters, RdbALTER will display the full file specification for the .rda or the .snp file of the storage area, the current setting of the inconsistent flag, and the current setting of the corrupt flag for .rda files.

Format



Command Parameters

*** (asterisk)**
Displays all file characteristics for all files.

area-id
Specifies the number of the storage area for which you want to display information from the .rdb file.

area-name
Specifies the name of the storage area for which you want to display information from the .rdb file.

SNAPSHOT
Displays information about an .snp file. If you select the SNAPSHOT parameter, you can specify the SPECIFICATION parameter; the INCONSISTENT and CORRUPT parameters are not valid options for .snp files.

2.12 DISPLAY FILE Command

INCONSISTENT
Displays the current setting of the inconsistent flag. This parameter applies only to .rda files.

SPECIFICATION
Displays the full file specification (including version number) for the .rda or the .snp file of the storage area.

CORRUPT
Displays the current setting of the corrupt flag. This applies only to .rda files.

Examples

Example 1

The following example displays the file specification for the JOBS storage area file:

```
RdbALTER> DISPLAY FILE JOBS
Area JOBS:
         File specification is: "DISK1:[RICK.RDB]JOBS.RDA;1"
         Corrupt flag is: OFF
         Inconsistent flag is: OFF
```

2.13 DISPLAY ROOT Command

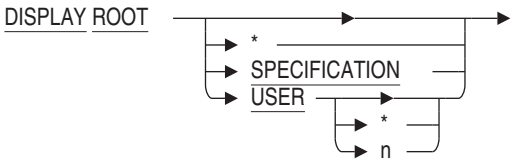
Displays the current database root file specification or the current recovery-unit journal (.ruj) file specification for a specific user of the database. You can use this statement to be sure you assigned the file specification you intended.

The DISPLAY ROOT SPECIFICATION command is the default if no other keyword follows the DISPLAY ROOT command.

Note

Prior to Oracle Rdb Version 6.0, you could use the DISPLAY ROOT command to display an after-image journal (.aij) file specification. This is no longer an option; use the RMU Show After_Journal command instead.

Format



Command Parameters

*** (asterisk)**
Displays the current database root (.rdb) file specification.

SPECIFICATION
Displays the current .rdb file specification for the database.

USER
Displays the current recovery-unit journal (.ruj) file specification for a user of the database. The USER parameter with no qualifier results in all users being displayed; the USER parameter with the * qualifier results in all users being displayed; and the USER parameter with the *n* qualifier results in a specific user being displayed, where *n* is a valid user number.

2.13 DISPLAY ROOT Command

Examples

Example 1

The following command displays the current .rdb file specification of the database:

```
RdbALTER> DISPLAY ROOT
        Root file specification is: "DISK1:[RICK.RDB]MF_PERSONNEL.RDB;1"
```

2.14 DISPLAY ROOT UNIQUE_IDENTIFIER Command

This command displays the current database root UNIQUE_IDENTIFIER value.

Description

To ensure Oracle Rdb database security and integrity, a UNIQUE_ IDENTIFIER has been added to the database root file and the database storage area file and storage area snapshot file headers. The UNIQUE_IDENTIFIER in the root file must match the UNIQUE_IDENTIFIER in the storage area file headers or a storage area cannot be accessed from the database root.

The UNIQUE_IDENTIFIER values are displayed both in VMS date format surrounded by quotes and as a hexadecimal number surrounded by parentheses. The values displayed are the UNIQUE_IDENTIFIER values for the current RMU/ALTER session. The UNIQUE_IDENTIFIER values will not be written to the root or storage area files until the user ends the current session with the RMU/ALTER COMMIT command. If the user ends the current session with the RMU/ALTER ROLLBACK command, the UNIQUE_ IDENTIFIER values will not be written to the root or storage area files and the UNIQUE_IDENTIFIER values in effect at the start of the session just ended will be restored for the new session. Any UNIQUE_IDENTIFIER values that have been changed during the current session will be displayed as "(marked)" before they are committed or rolled back.

To execute the DISPLAY or DEPOSIT ROOT command, the user must be attached to the database which the root and areas belong to, either by specifying the database name when issuing the RMU/ALTER command or by executing the ATTACH command from the "RdbALTER>" prompt.

Format

DISPLAY ROOT UNIQUE_IDENTIFIER

Examples

Example 1

2.14 DISPLAY ROOT UNIQUE_IDENTIFIER Command

The following example shows how to display a UNIQUE_IDENTIFIER value using RMU/ALTER.

```
$ RMU/ALTER MF_PERSONNEL
%RMU-I-ATTACH, now altering database
 "DEVICE:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DISPLAY ROOT UNIQUE_IDENTIFIER
         Root file unique identifier is: "22-OCT-2010 13:49:27.87"
 (00AA5578688428BB)
  EXIT
```

Example 2

The following example shows that RMU/ALTER is invoked specifying the database MF_PERSONNEL.RDB. The user then displays the current UNIQUE_IDENTIFIER value in the database root, creates a new UNIQUE_ IDENTIFIER value in the database root, displays the new UNIQUE_ IDENTIFIER in the root, and finally specifies COMMIT to write the new UNIQUE_IDENTIFIER value to the database root file and ALL database storage area files. The display messages designate the pending new UNIQUE_ IDENTIFIER value as "(marked)" until the user either executes COMMIT to write out the new UNIQUE_IDENTIFIER value or ROLLBACK to restore the original UNIQUE_IDENTIFIER value. The user then verifies the database changes.

```
$ RMU/ALTER MF_PERSONNEL
%RMU-I-ATTACH, now altering database "DISK:[DIRECTORY]MF_PERSONNEL.RDB;1"
  DISPLAY ROOT UNIQUE_IDENTIFIER
         Root file unique identifier is: "22-OCT-2010 13:49:27.87"
 (00AA5578688428BB)
  DEPOSIT ROOT UNIQUE_IDENTIFIER = NEW
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)
  DISPLAY ROOT UNIQUE_IDENTIFIER
(marked) Root file unique identifier is: "22-OCT-2010 13:49:28.34"
 (00AA557868CC9F7A)
  COMMIT
  EXIT
$ RMU/VERIFY/ALL/NOLOG MF_PERSONNEL
```

2.15 EXIT Command

Terminates the RdbALTER session and returns you to DCL command level. You can also press Ctrl/Z to end an RdbALTER session.

If the EXIT command is issued and altered but uncommitted pages exist, you are told to issue either a COMMIT or a ROLLBACK command. RdbALTER does not exit until a COMMIT or a ROLLBACK operation has accounted for all altered pages.

The EXIT command performs an implicit NOLOG command if a log file is open.

Format

EXIT

Examples

Example 1

The following example exits RdbALTER command level and returns you to the DCL command level:

```
RdbALTER> EXIT
$
```

Example 2

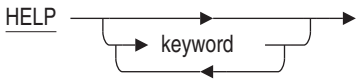
The following example shows that you cannot exit from the RdbALTER session if there are uncommitted changes in your database:

```
RdbALTER> DEPOSIT ROOT SPECIFICATION=DISK1:[RICK]MF_PERSONNEL.RDB;1
(marked) Root file specification is: "DISK1:[RICK]MF_PERSONNEL.RDB;1"
RdbALTER> EXIT
%RMU-F-COMMITROLL, currently modified ROOT fields must be committed or
rolled back
```

2.16 HELP Command

Provides information about RdbALTER commands, terminology, and concepts. If you type HELP at the RdbALTER prompt without specifying a topic, RdbALTER displays a list of topics on which help is available. If you type HELP followed by a topic, you receive a brief description of that topic.

Format



Command Parameters

keyword
Identifies the help topic or subtopic you want explained.

Examples

Example 1

The following example requests two layers of help. First, the user types HELP without specifying a topic, and a list of topics for which help is available is displayed. Then the user selects the VERIFY topic to obtain information for the VERIFY command.

```
RdbALTER> HELP

  Information available:

  AREA-PAGE  ATTACH     COMMIT     DEPOSIT    DETACH     DISPLAY    ERRORS
  EXIT       HELP       LOG        MAKE_CONSISTENT       MOVE       NOLOG
  PAGE       RADIX      ROLLBACK   UNCORRUPT  VERIFY

Topic? VERIFY
```

2.17 LOG Command

Keeps an audit trail of all or part of an RdbALTER session. After you specify the LOG command, RdbALTER commands and their results are logged in the specified log file until you close the log file by entering a NOLOG command, an EXIT command, or another LOG command.

Format

Command Parameters

file-spec
Specifies a file to contain the audit trail log. The default file extension is .lis.

Examples

Example 1

The following command creates the file audit.lis and begins audit trail logging:

```
RdbALTER> LOG AUDIT
```

Example 2

The following command creates the file audit.trl and begins audit trail logging:

```
RdbALTER> LOG AUDIT.TRL
```

2.18 MAKE CONSISTENT Command

Resets an area's inconsistent indication flag, allowing you to use the database.

Description

When a storage area is restored from backup files on a by-area basis, it does not reflect data that has been updated since the backup operation. The transaction level of the restored area reflects the transaction level of the backup file, not the transaction level of the database. Therefore, the transaction level of the restored area differs from that of the database. Oracle Rdb marks the area by setting a flag in the storage area file to inconsistent.

You can perform a recovery by area to upgrade the transaction level of the restored area to that of the database. (After-image journaling must be enabled in order to restore by area.) If you are certain that no updates have been made to the database since the backup operation, you can use the MAKE CONSISTENT command in RdbALTER to change the setting of the flag from inconsistent to consistent.

Note

Beginning in Oracle Rdb Version 6.0, the capabilities provided through this command are also provided through the RMU Set Corrupt_ Pages command. The RdbALTER MAKE CONSISTENT command might be removed in future versions of Oracle Rdb. Therefore, Oracle Corporation recommends that you use the RMU Set Corrupt_Pages command instead of the RdbALTER MAKE CONSISTENT command when these capabilities are needed.

Format

Command Parameters

storage-area-name
Specifies a storage area by the storage area name, which is the name defined with the SQL CREATE STORAGE AREA statement.

storage-area-number
Specifies a storage area by storage area number, which is assigned when the database is created and is given on the first line of a page display.

Examples

Example 1

The following example resets the indication flag from inconsistent to consistent for the area JOBS:

```
RdbALTER> MAKE CONSISTENT JOBS

                ***** WARNING! *****

 BEWARE ATTEMPTING TO MAKE CONSISTENT A STORAGE
 AREA WITHOUT FIRST VERIFYING IT USING THE
 RMU/VERIFY COMMAND.

 AN RdbALTER ROLLBACK COMMAND WILL LEAVE THIS
 AREA MARKED INCONSISTENT.

Area JOBS now marked consistent.
```

2.19 MOVE Command

Moves data (defined by beginning and ending offset addresses) from one page location to another location on the same page.

The number of bytes moved is:

```
(old-offset-end) - (old-offset-start) + 1
```

The sending field, defined by the old-offset arguments, remains unchanged.

The receiving field, defined by the new-offset argument and the length of the sending field, is replaced by the contents of the sending field.

Other information in the page is unchanged.

Format

MOVE old-offset-start:old-offset-end new-offset →

Command Parameters

old-offset-start
Specifies the hexadecimal offset address of the first byte in the data sequence to be moved.

old-offset-end
Specifies the hexadecimal offset address of the last byte in the data sequence to be moved.

new-offset
Specifies the hexadecimal offset address of the first byte in the sequence of bytes receiving the moved data.

Examples

Example 1

The following example moves data from offset location 34A through 34E to the starting offset location of 354:

```
RdbALTER> MOVE 34A:34E 354
```

See the *Oracle Rdb Guide to Database Maintenance* for more examples of how to use the MOVE command.

2.20 NOLOG Command

Stops RdbALTER logging. The NOLOG command stops audit trail logging if a previous LOG command is still in effect. The EXIT command performs an implicit NOLOG if LOG is still active; thus, you need not enter a NOLOG command before exiting the RdbALTER session.

Format

Examples

Example 1

The following example stops audit trail logging:

```
RdbALTER> NOLOG
```

2.21 PAGE Command

Fetches a page from the current storage area. If you specify a valid page number, that page is fetched from the current storage area. If you enter the PAGE command without a page number, the next page in the current storage area is fetched. If you enter the PAGE command without a page number and you are already at the highest numbered page of the storage area, page 1 of the storage area is fetched.

Format

Command Parameters

page-number
Identifies a page to be fetched in the current storage area. If you do not specify the page-number parameter, the next page in the current area is fetched. If you do not specify a page number and you are at the highest numbered page in the storage area, page 1 of the current storage area is fetched.

Example

Example 1

The following example fetches page 14 of the current storage area:

```
RdbALTER> PAGE 14
```

Example 2

If the current page is page 15 of the JOBS storage area, the following command fetches page 16 of the JOBS storage area:

```
RdbALTER> PAGE
```

2.22 RADIX Command

Sets the default radix for entering numeric data for the DISPLAY DATA command and determines how data values are parsed by the DEPOSIT DATA command.

If you do not use this command, RdbALTER expects data values to be entered as decimal values.

This command does not change the radix for specifying page offsets. Page offsets must always be specified as a hexadecimal radix number.

Format

Command Parameters

DECIMAL
Sets the default radix in decimal numbers. This is the default.

HEXADECIMAL
Sets the default radix in hexadecimal numbers.

Examples

Example 1

The following example sets the default radix for entering numeric data to hexadecimal:

```
RdbALTER> RADIX HEXADECIMAL
```

2.23 ROLLBACK Command

Ignores all page changes since the last COMMIT or last ROLLBACK command. Altered pages are stored in virtual memory until you issue a COMMIT or a ROLLBACK command. The ROLLBACK command tells RdbALTER to ignore all changes since the last COMMIT or the last ROLLBACK command was issued.

Do not follow a ROLLBACK command with a DEPOSIT command without first specifying your location again with a DISPLAY or a PAGE command.

RdbALTER does not allow you to issue an EXIT command until all altered pages are either committed or rolled back.

Format

ROLLBACK ⟶

Examples

Example 1

The following example rolls back all page changes entered since the last COMMIT or last ROLLBACK command was issued:

```
RdbALTER> ROLLBACK
```

2.24 UNCORRUPT Command

Resets the storage area's corruption indication flag (FILID_CORRUPT_FLG), allowing you to use the uncorrupted sections of a corrupted storage area. Storage areas are most often corrupted by attempting an SQL (not RdbALTER) roll back with one or more storage areas opened in batch-update transaction mode.

The UNCORRUPT command allows you to access a database that is in an uncertain condition. Accordingly, the following message is displayed when you enter the command:

```
BEWARE ATTEMPTING TO UNCORRUPT A STORAGE AREA
WITHOUT FIRST VERIFYING IT.
```

Note

Beginning in Oracle Rdb Version 6.0, the capabilities provided through this command are also provided through the RMU Set Corrupt_Pages command. The RdbALTER UNCORRUPT command might be removed in future versions of Oracle Rdb. Therefore, Oracle Corporation recommends that you use the RMU Set Corrupt_Pages command instead of the RdbALTER UNCORRUPT command when these capabilities are needed.

Use of the ROLLBACK or the COMMIT command is permitted with the RdbALTER UNCORRUPT command.

Format

Command Parameters

storage-area-name
Specifies a storage area in the current database by storage area name, which is the name defined with the SQL CREATE STORAGE AREA statement.

2.24 UNCORRUPT Command

storage-area-number
Specifies an area of the current database by the storage area number, which is assigned when the database is created and is given on the first line of a page display.

Examples

Example 1

The following example resets the corruption indication flag for the EMPIDS_ LOW area.

```
RdbALTER> UNCORRUPT EMPIDS_LOW

                ***** WARNING! *****

   BEWARE ATTEMPTING TO UNCORRUPT A STORAGE AREA
   WITHOUT FIRST VERIFYING IT USING THE RMU/VERIFY
   COMMAND.

   AN RdbALTER ROLLBACK COMMAND WILL LEAVE THIS
   AREA MARKED CORRUPT.

 Area EMPIDS_LOW now marked uncorrupt.
```

2.25 VERIFY Command

Verifies the current page. When the VERIFY command is issued, the page header and page checksum are verified for the current page, and error messages are issued if header or checksum corruption is found.

Format

VERIFY ⟶

Examples

Example 1

The following example verifies area 3 page 100:

```
RdbALTER> AREA 3 PAGE 100
RdbALTER> VERIFY
```

A

RMU Load Record Definition File Language

The record definition files (.rrd) used by the RMU Load, RMU Unload, and RMU Analyze commands are used to describe the field data types and field ordering for binary and delimited text data files. The .rrd files contain a simple language similar to that accepted by the CDO interface of the Oracle CDD Repository. The RMU Unload command automatically generates a record definition file from the table definition in the database.

This appendix describes the .rrd language which is accepted by the RMU Load command. It describes a useful subset of the language supported by RMU. Clauses from CDO which RMU accepts but ignores are not described.

A.1 DEFINE FIELD statement

Each record definition file must include at least one DEFINE FIELD statement to describe the data type of a field in the unloaded record. This statement has two formats:

- a format that defines a new name

  ```
  define field name_string datatype is text size is 20 characters.
  ```

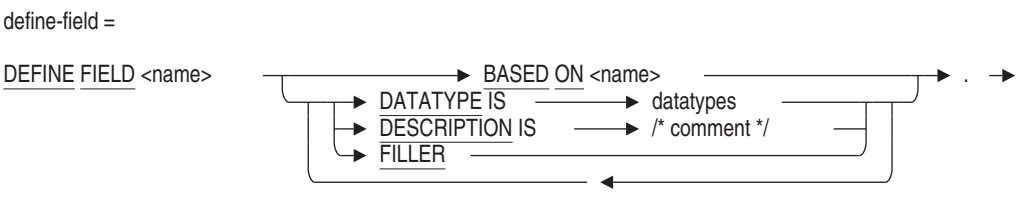
- a format that references another, previously defined, field

  ```
  define field first_name based on name_string.
  ```

RMU Unload generates the DEFINE FIELD statement with just the DATATYPE clause. The full syntax is shown in Figure A–1.

The following example of the DEFINE FIELD statement is more complete, showing the use of annotations (DESCRIPTION clause) and based-on fields.

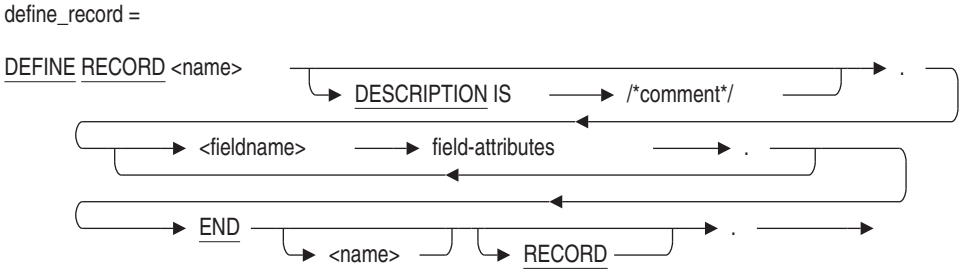
Figure A–1 DEFINE FIELD Statement




```
define field name_string
    description is
        /* This a generic string type to be used for based on */
    datatype is text size is 20 characters.
define field first_name
    based on name_string.
define field last_name
    based on name_string.

define record PERSON
    description is
        /* Record which describes the PERSON.DAT RMS file */.
    first_name.
    last_name.
end.
```

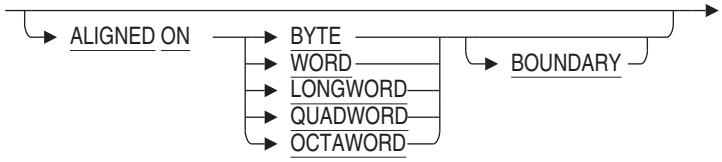
A.2 DEFINE RECORD Statement

The DEFINE RECORD statement defines the ordering of the fields within the file. A field may only be used once. The name of the field is not used for column name matching unless the Corresponding qualifier is used with the RMU Load command.



field-attributes =

alignment-clause =



- The ALIGNED ON clause can be used to adjust for alignment added explicitly or implicitly by host language applications. For instance, on OpenVMS Alpha many 3GL language compilers naturally align fields to take advantage of the Alpha processor hardware which executes more efficiently when data is well aligned. The default is BYTE alignment.

  In the following example, field C is expected to start on a quadword boundary, so A is assigned the first longword, the second longword is ignored, and finally the C is assigned the last longword value.

```
define field A datatype is signed longword.
define field C datatype is signed longword.
define record RMUTEST.
   A .
   C aligned on quadword boundary.
end RMUTEST record.
```

- The STORE USING clause can be used to provide a transformation routine so that the source data being loaded can be modified during the insertion into the target table. For example, some text fields might include the text '(none)' to indicate an unknown state for this field and this might naturally be transformed to NULL during loading.

  The module definition for this example would simply be:

```
create module CVT_LOAD
  function CVT_IF_NULL (in :arg varchar(40))
  returns varchar(40);
  begin
  declare :v varchar(40) = trim(:arg);
  if :v = '(none)'
  then
    return cast (NULL as varchar(40));
  else
    return :v;
  end if;
  end;
end module;
```

The record definition file would appear as follows:

```
DEFINE FIELD LAST_NAME DATATYPE IS TEXT SIZE IS 40.
DEFINE FIELD FIRST_NAME DATATYPE IS TEXT SIZE IS 30.
DEFINE FIELD MIDDLE_NAME BASED ON FIRST_NAME.
DEFINE RECORD NAME_FILE.
  LAST_NAME.
  MIDDLE_NAME STORE USING CVT_IF_NULL.
  FIRST_NAME.
END.
```

A.2.1 Usage Notes

- When the DCL verify process is enabled using the DCL SET VERIFY command or the DCL F$VERIFY lexical function, RMU Load writes the .rrd file being processed to SYS$OUTPUT.
- There is no equivalent to the VARCHAR or VARYING STRING data types in the record definition language because there is no support for these types in the OpenVMS Record Management Services (RMS) environment.
- The VARCHAR or VARYING STRING data type is a two-part type with an UNSIGNED WORD (16 bit integer) length prefix with a fixed TEXT portion. The length defines the actual data in the string.

  There is no equivalent to the VARCHAR or VARYING STRING data types in the record definition language as this type is not supported by the OpenVMS Record Management Services (RMS) environment.

  If you unload a VARCHAR column then it will be converted to a fixed length (space padded) TEXT field. However, TEXT to VARCHAR load and unload is handled appropriately when using the delimited format. In this format RMU Unload only outputs the text as specified by the length prefix of the VARCHAR column. Likewise, RMU Load uses the length of the delimited string to set the length in the database.

- If a field is not to be used during the load into the table, it can be ignored during the load using the FILLER attribute. This allows RMU Load to use a data file which has more fields than there are columns in the database table.
- The <name> referenced in the END RECORD clause must be the same as the name defined by the DEFINE RECORD statement.
- The record definition files are not used when the Record_definition qualifier is omitted. In this case RMU Unload generates a structured internal file format which contains both the record definition and the data. This format allows the unloading of LIST OF BYTE VARYING columns and NULL values. This format is the same as that generated by SQL EXPORT for its interchange (.rbr) file. Use the RMU Dump Export command to format the contents of this file for display.

  ```
  $ rmu/unload mf_personnel employees employees.unl
  $ rmu/dump/export/nodata employees.unl
  ```

- When using the STORE USING clause, the <routine-name> must exist in the target database or be one of the supported built in functions (see below). If the name contains lowercase or other special characters, then it should be delimited. For example, "Convert-If-Null".
- The routine must be a function: either a SQL or external function that accepts at least one parameter. If more than one parameter exists for the function, then the remaining parameters must have DEFAULT values. RMU cannot be used to pass additional values and only passes the source data in the first parameter.
- The routine must be granted EXECUTE privileges for the user executing the RMU Load command or the user must have OpenVMS override privileges such as BYPASS or SYSPRV.
- RMU supports these built in functions: TRIM, LTRIM, RTRIM, UPPER, LOWER and SQRT. They can be used in the STORE USING clause without additional privileges on the database.
- The returned data type must be compatible with the target column. Oracle Rdb will perform some simple implicit conversions, such as text to numeric types.

  The script SQL$SAMPLE:CVT_MODULE.SQL includes some examples of transformation routines that can be used directly or used as models for custom written routines.

- Modules created for use with RMU Load can use the SQL SIGNAL statement to raise an exception which will cause the failing row to be written to the EXCEPTION_FILE (if specified).

  For example, if the previous example wanted to report any blank names, a SIGNAL statement could be used.

```
create module CVT_LOAD
  function CVT_IF_NULL (in :arg varchar(40))
  returns varchar(40);
  begin
  declare :v varchar(40) = trim(:arg);
  if :v = '(none)'
  then
    return NULL;
  end if;

  if length (:v) = 0
  then
      signal '22000' ('Blank names not permitted');
  end if;

  return :v;
  end;
end module;
```

A.3 Additional Data Types

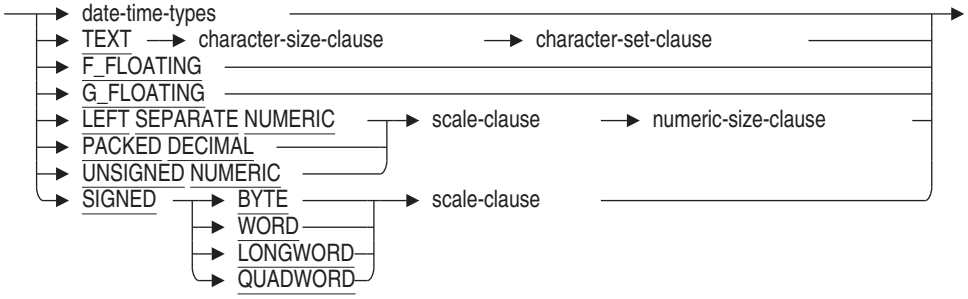
The data types that are supported by Oracle Rdb are described in *Oracle Rdb SQL Reference Manual*.

The cset-name is any character set supported by Oracle Rdb. These character sets are expanded from release to release. Please refer to the Oracle Rdb SQL Reference Manual for new character sets and more information on the character sets listed below.

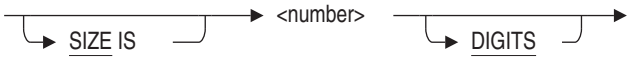
When several character sets are available for the same language (such as Kanji and Hanyu), each is based on a different local or international standard that differs from the others in format and structure. For instance, SHIFT_JIS is widely used with Microsoft Windows systems in Japan, but differs in format from the DEC_KANJI character set supported by Hewlett Packard Company's DECWindows product.

Figure A–2 Data Types

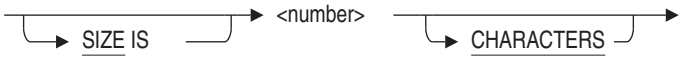
datatypes =



numeric-size-clause =



character-size-clause =



scale-clause =

character-set-clause =

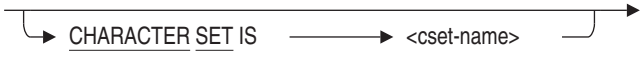


Table A–1 Character sets supported by Oracle RMU Load

Character Set	Description
AL24UTFFSS	UTF-8 encoding based on the Unicode standard 1.1
ASCII	7-bit ASCII characters
BINARY	Special character set used to support the BINARY and BINARY VARYING (VARBINARY) data types
ARABIC	Arabic characters as defined by the ISO9036 standards
BIG5	A set of characters used by the Taiwan information industry
DEC_HANYU	Traditional Chinese characters (Hanyu) as used in Taiwan and defined by standard CNS11643:1986, supplemental characters as defined by DTSCC and ASCII
DEC_HANZI	Chinese (Bopomofo) characters as defined by standard GB2312:1980 and ASCII characters
DEC_KANJI	Japanese characters as defined by the JIS X0208:1990 standard, Hankaku Katakana characters as defined by JIS X0201:1976 prefixed by SS2 (8E hex), user-defined characters, and ASCII characters
DEC_KOREAN	Korean characters as defined by standard KS C5601:1987 and ASCII characters
DEC_MCS	A set of international alphanumeric characters, including characters with diacritical marks
DEC_SICGCC	Traditional Chinese characters (Hanyu) as used in Taiwan and defined by standard CNS11643:1986 and ASCII
DEVANAGARI	Devanagari characters as defined by the ISCII:1988 standard
DOS_LATIN1	DOS Latin 1 code
DOS_LATINUS	DOS Latin US code
GB18030	Simplified Chinese characters as used by the People's Republic of China (PRC) and defined by the GB18030-2000 standard
HANYU	Traditional Chinese characters (Hanyu) as used in Taiwan and defined by the standard CNS11643:1986
HANZI	Chinese (Bopomofo) characters as defined by standard GB2312:1980
HEX	Translation of text data to and from hexadecimal data
ISOLATIN1	Extended European characters as defined by the ISO/IEC 8859-1:1987 standard
ISOLATIN2	Extended East European characters as defined by the ISO/IEC 8859-2 standard

(continued on next page)

Table A–1 (Cont.) Character sets supported by Oracle RMU Load

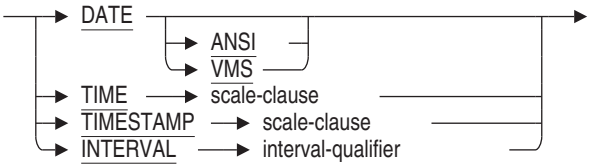
Character Set	Description
ISOLATIN9	Extended European characters as defined by the ISO/IEC 8859-15 standard which includes the Euro Character code-point.
ISOLATINARABIC	Arabic characters as defined by the ISO/IEC 8859-6:1987 standard
ISOLATINCYRILLIC	Cyrillic characters as defined by the ISO/IEC 8859-5:1987 standard
ISOLATINGREEK	Greek characters as defined by the ISO/IEC 8859-7:1987 standard
ISOLATINHEBREW	Hebrew characters as defined by the ISO/IEC 8859-8:1987 standard
KANJI	Japanese characters as defined by the JIS X0208:1990 standard and user-defined characters
KATAKANA	Japanese phonetic alphabet (Hankaku Katakana), as defined by standard JIS X0201:1976
KOREAN	Korean characters as defined by standard KS C5601:1987
SHIFT_JIS	Japanese characters as defined by the JIS X0208:1990 standard using Shift_JIS specific encoding scheme, Hankaku Katakana characters as defined by JIS X0201:1976, and ASCII characters
TACTIS	Thai characters based on TACTIS (Thai API Consortium/Thai Industrial Standard) which is a combination of ISO 646-1983 and TIS 620-2533 standards
UNICODE	Unicode characters as described by Unicode Standard and ISO/IEC 10646 transformation format UTF-16
UNSPECIFIED	Unspecified character set.
UTF8	Unicode characters as described by Unicode Standard and ISO/IEC 10646 UTF-encoding form with restrictions as specified in RFC 3629
WIN_ARABIC	MS Windows Code Page 1256 8-Bit Latin/Arabic
WIN_CYRILLIC	MS Windows Code Page 1251 8-Bit Latin/Cyrillic
WIN_GREEK	MS Windows Code Page 1253 8-Bit Latin/Greek
WIN_HEBREW	MS Windows Code Page 1255 8-Bit Latin/Hebrew
WIN_LATIN1	MS Windows Code Page 1252 8-Bit West European
WIN_LATIN2	MS Windows Code Page 1250 8-Bit East European

A.3.1 Date-Time Syntax

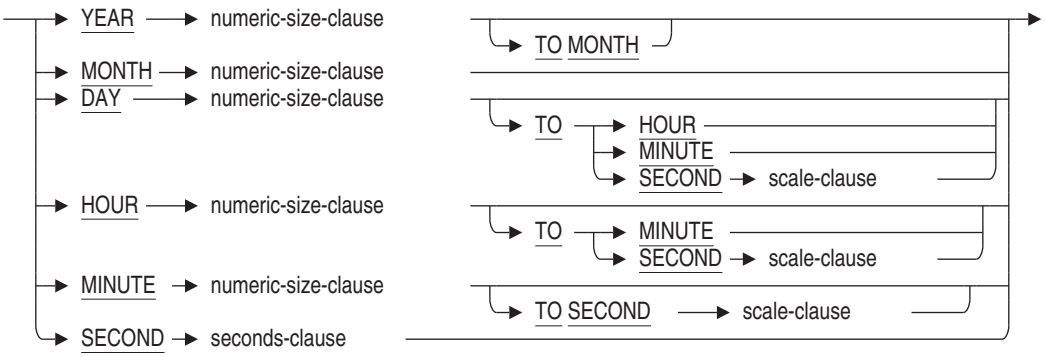
The date-time syntax in .rrd files generated by the RMU Unload command with the Record_Definition=(File=file) command is compatible with the date-time syntax support of Oracle CDD/Repository V6.1 and later versions.

The date-time data type has the following syntax in the .rrd file.

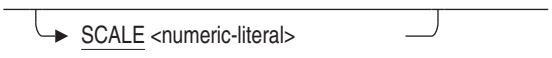
date-time-types =



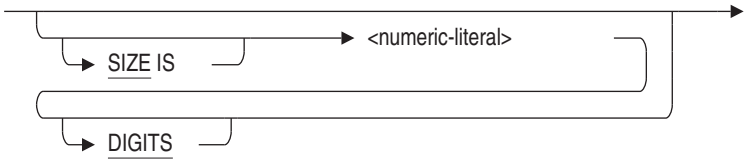
interval-qualifier =



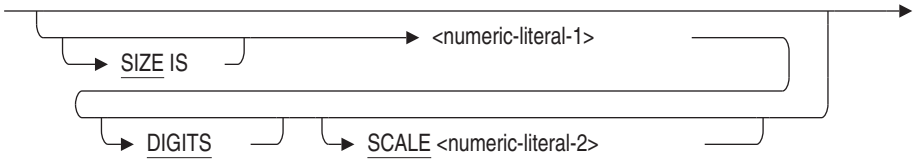
scale-clause =



numeric-size-clause =



seconds-clause =



Note that SCALE values must be between 0 and –2 and that SIZE IS values must be between 2 and 9.

The following are examples of typical field definitions for date-time data types in .rrd files:

```
DEFINE FIELD A DATATYPE IS DATE.
DEFINE FIELD B DATATYPE IS DATE ANSI.
DEFINE FIELD C DATATYPE IS INTERVAL DAY SIZE IS 2 DIGITS.
DEFINE FIELD D DATATYPE IS INTERVAL DAY SIZE IS 2 DIGITS TO HOUR.
DEFINE FIELD E DATATYPE IS INTERVAL DAY SIZE IS 2 DIGITS TO
 SECOND SCALE -2.
DEFINE FIELD F DATATYPE IS INTERVAL HOUR SIZE IS 4 DIGITS.
DEFINE FIELD G DATATYPE IS INTERVAL HOUR SIZE IS 2 DIGITS TO MINUTE.
DEFINE FIELD H DATATYPE IS INTERVAL MINUTE SIZE IS 2 DIGITS TO
 SECOND SCALE -2.
DEFINE FIELD I DATATYPE IS INTERVAL SECOND SIZE IS 2 DIGITS SCALE -2.
DEFINE FIELD J DATATYPE IS TIME.
DEFINE FIELD K DATATYPE IS TIME SCALE -1.
DEFINE FIELD L DATATYPE IS TIMESTAMP SCALE -2.
DEFINE FIELD M DATATYPE IS INTERVAL YEAR SIZE IS 3 DIGITS TO MONTH.
```

B

Using LogMiner for Rdb

This appendix provides information about using the LogMiner for Rdb feature.

Oracle Rdb after-image journal (.aij) files contain a wealth of useful information about the history of transactions in a database. After-image journal files contain all of the data needed to perform database recovery. These files record every change made to data and metadata in the database. The LogMiner for Rdb feature provides an interface to the data record contents of Oracle Rdb after-image journal files. Data records that are added, updated, or deleted by committed transactions may be extracted (unloaded) from the .aij files in a format suitable for subsequent loading into another database or for use by user-written application programs.

Oracle Rdb after-image journaling protects the integrity of your data by recording all changes made by committed transactions to a database in a sequential log or journal file. Oracle Corporation recommends that you enable after-image journaling to record your database transaction activity between full backup operations as part of your database restore and recovery strategy. The after-image journal file is also used to enable several database performance enhancements (such as the fast commit, row cache, and hot standby features).

See the *Oracle Rdb7 Guide to Database Maintenance* for more information about setting up after-image journaling.

To use LogMiner for Rdb, follow these steps:

1. Enable the database for LogMiner operation using the RMU Set Logminer command. See Section 1.61 for additional information.
2. Back up the after-image journal file using the Quiet_Point qualifier to the RMU Backup command.
3. Extract changed records using the RMU Unload After_Journal command. See the Unload After_Journal help topic for additional information.

Oracle Rdb provides several RMU commands which are associated with LogMiner For Rdb.

- RMU Set Logminer Command

  Allows you to change the LogMiner state of a database.

- RMU Unload After_Journal Command

  Allows you to extract added, modified, committed, and deleted record contents from committed transactions from specified tables in one or more after-image journal files.

- RMU Dump Metadata_File Command

  This command dumps the metadata file created by the RMU Unload After_Journal command in a user readable format.

- RMU Show After_Journal Command

  This command will show whether LogMiner or Continuous LogMiner are enabled. This information is also included in the AIJ options file.

B.1 Restrictions and Limitations with LogMiner for Rdb

The following restrictions exist for the LogMiner for Rdb feature:

- Temporary tables cannot be extracted. Modifications to temporary tables are not written to the after-image journal file and, therefore, are not available to LogMiner for Rdb.
- Optimized after-image journal files cannot be used as input to the LogMiner for Rdb. Information needed by the RMU Unload After_Journal command is removed by the optimization process.
- Records removed from tables using the SQL TRUNCATE TABLE statement are not extracted. The SQL TRUNCATE TABLE statement does not journal each individual data record being removed from the database.
- Records removed by dropping tables using the SQL DROP TABLE statement are not extracted. The SQL DROP TABLE statement does not journal each individual data record being removed from the database.
- Tables that use the vertical record partitioning (VRP) feature cannot be extracted using LogMiner for Rdb.
- Segmented string data (BLOB) cannot be extracted using LogMiner for Rdb. Because the segmented string data is related to the base table row by means of a database key, there is no convenient way to determine what data to extract. Additionally, the data type of an extracted column is changed from LIST OF BYTE VARYING to BIGINT. This column contains

the DBKEY of the original BLOB data. Therefore, the contents of this column should be considered unreliable.

- COMPUTED BY columns in a table are not extracted. These columns are not stored in the after-image journal file.
- VARCHAR fields are not space padded in the output file. The VARCHAR data type is extracted as a 2-byte count field and a fixed-length data field. The 2-byte count field indicates the number of valid characters in the fixed-length data field. Any additional contents in the data field are unpredictable.
- You cannot extract changes to a table when the table definition is changed within an after-image journal file. Data definition language (DDL) changes to a table are not allowed within an .aij file being extracted. All records in an .aij file must be the current record version. If you are going to perform DDL operations on tables that you wish to extract using the LogMiner for Rdb, you should:

  1. Back up your after-image journal files.
  2. Extract the .aij files using the RMU Unload After_Journal command.
  3. Make the DDL changes.

- Do not use the OpenVMS Alpha High Performance Sort/Merge utility (selected by defining the logical name SORTSHR to SYS$SHARE:HYPERSORT) when using LogMiner for Rdb. HYPERSORT supports only a subset of the library sort routines that LogMiner requires. Make sure that the SORTSHR logical name is not defined to HYPERSORT.

B.2 Information Returned by LogMiner for Rdb

LogMiner for Rdb appends several output fields to the data fields, creating an output record. The output record contains fixed-length fields in a binary data format (that is, integer fields are not converted to text strings). The data fields correspond to the extracted table columns. This information may or may not be required by all applications and readers of the data. There is currently no available method to restrict or reorder the output fields.

Extracted data field contents are the fields that are actually stored in the Oracle Rdb database. COMPUTED BY fields are not extracted because they are not stored in the database or in the after-image journal file. Segmented string (BLOB) contents are not extracted.

Table B–1 describes the output fields and data types of an output record.

Table B–1 Output Fields

Field Name	Data Type	Byte Length	Description
ACTION	CHAR (1)	1	Indicates record state. "M" indicates an insert or modify action. "D" indicates a delete action. "E" indicates stream end-of-file (EOF) when a callback routine is being used. "P" indicates a value from the command line Parameter qualifier when a callback routine is being used (see Parameter qualifier). "C" indicates transaction commit information when the Include=Action=Commit qualifier is specified.
RELATION_NAME	CHAR (31)	31	Table name. Space padded to 31 characters.
RECORD_TYPE	INTEGER (Longword)	4	The Oracle Rdb internal relation identifier.
DATA_LEN	SMALLINT (Word)	2	Length, in bytes, of the data record content.
NBV_LEN	SMALLINT (Word)	2	Length, in bits, of the null bit vector content.
DBK	BIGINT (Quadword)	8	Records logical database key. The database key is a 3-field structure containing a 16-bit line number, a 32-bit page number and a 16-bit area number.
START_TAD	DATE VMS (Quadword)	8	Date-time of the start of the transaction.
COMMIT_TAD	DATE VMS (Quadword)	8	Date-time of the commitment of the transaction.
TSN	BIGINT (Quadword)	8	Transaction sequence number of the transaction that performed the record operation.
RECORD_VERSION	SMALLINT (Word)	2	Record version.
Record Data	Varies		Actual data record field contents.

(continued on next page)

Table B–1 (Cont.) Output Fields

Field Name	Data Type	Byte Length	Description
Record NBV	BIT VECTOR (array of bits)		Null bit vector. There is one bit for each field in the data record. If a bit value is 1, the corresponding field is NULL; if a bit value is 0, the corresponding field is not NULL and contains an actual data value. The null bit vector begins on a byte boundary. Any extra bits in the final byte of the vector after the final null bit are unused.

B.3 Record Definition Prefix for LogMiner Fields

An RMS file containing the record structure definition for the output file can be used as an input file to the RMU Load command if extracted data is to be loaded into an Oracle Rdb database. The record description uses the CDO record and field definition format (this is the format used by the RMU Load and RMU Unload commands when the Record_Definition qualifier is used). The default file extension is .rrd.

The record definition for the fields that LogMiner for Rdb writes to the output is shown in the following example. These fields can be manually appended to a record definition file for the actual user data fields being unloaded. Alternately, the Record_Definition qualifier can be used with the Table qualifier or within an Options file to automatically create the record definition file. This can be used to load a transaction table within a database. A **transaction table** is the output that LogMiner for Rdb writes to a table consisting of sequential transactions performed in a database.

```
DEFINE FIELD RDB$LM_ACTION          DATATYPE IS TEXT SIZE IS 1.
DEFINE FIELD RDB$LM_RELATION_NAME      DATATYPE IS TEXT SIZE IS 31.
DEFINE FIELD RDB$LM_RECORD_TYPE     DATATYPE IS SIGNED LONGWORD.
DEFINE FIELD RDB$LM_DATA_LEN        DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_NBV_LEN         DATATYPE IS SIGNED WORD.
DEFINE FIELD RDB$LM_DBK             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_START_TAD       DATETYPE IS DATE
DEFINE FIELD RDB$LM_COMMIT_TAD      DATATYPE IS DATE
DEFINE FIELD RDB$LM_TSN             DATATYPE IS SIGNED QUADWORD.
DEFINE FIELD RDB$LM_RECORD_VERSION  DATATYPE IS SIGNED WORD.
```

B.4 SQL Table Definition Prefix for LogMiner Fields

The SQL record definition for the fields that LogMiner for Rdb writes to the output is shown in the following example. These fields can be manually appended to the table creation command for the actual user data fields being unloaded. Alternately, the Table_Definition qualifier can be used with the Table qualifier or within an Options file to automatically create the SQL definition file. This can be used to create a transaction table of changed data.

```
SQL> create table MYLOGTABLE (
cont> RDB$LM_ACTION          CHAR,
cont> RDB$LM_RELATION_NAME      CHAR (31),
cont> RDB$LM_RECORD_TYPE     INTEGER,
cont> RDB$LM_DATA_LEN        SMALLINT,
cont> RDB$LM_NBV_LEN         SMALLINT,
cont> RDB$LM_DBK             BIGINT,
cont> RDB$LM_START_TAD       DATE VMS,
cont> RDB$LM_COMMIT_TAD      DATE VMS,
cont> RDB$LM_TSN             BIGINT,
cont> RDB$LM_RECORD_VERSION  SMALLINT ...);
```

B.5 Segmented String Columns

Segmented string (also called BLOB or LIST OF BYTE VARYING) column data is not extracted. However, the field definition itself is extracted as a quadword integer representing the database key of the original segmented string data. In generated table definition or record definition files, a comment is added indicating that the segmented string data type is not supported by LogMiner for Rdb.

B.6 Using LogMiner to Minimize Application Downtime for Maintenance

Lengthy offline application or database maintenance operations can pose a significant problem in high-availability production environments. The LogMiner for Rdb feature can help reduce the length of downtime to a matter of minutes.

If a back up of the database is used for maintenance operations, the application can continue to be modified during lengthy maintenance operations. Once the maintenance is complete, the application can be shut down, the production system .aij file or files can be backed up, and LogMiner for Rdb can be used to extract changes made to production tables since the database was backed up. These changes can then be applied (using an application program or the trigger technique previously described) to the new database. Once the new database has been updated, the application can be restarted using the new database.

The sequence of events required would be similar to the following:

1. Perform a full online, quiet-point database backup of the production database.
2. Restore the backup to create a new database that will eventually become the production database.
3. Perform maintenance operations on the new database. (Note that the production system continues to run.)
4. Perform an online, quiet-point after-image journal backup of the production database.
5. Use the RMU Unload After_Journal command to unload all database tables into individual output files from the .aij backup file.
6. Using either the trigger technique or an application program, update the tables in the new database with the changed data.
7. Shut down the production application and close the database.
8. Perform an offline, quiet-point after-image journal backup of the production database.
9. Use the RMU Unload After_Journal command to unload all database tables into individual output files from the .aij backup file.
10. Using either the trigger technique or an application program, update the tables in the new database with the changed data.
11. Start an online, quiet-point backup of the new database.
12. Change logical names or the environment to specify the new database root file as the production database.
13. Restart the application on the new database.

Depending on the amount of application database activity, steps 4, 5, and 6 can be repeated to limit the amount of data that needs to be applied (and the amount of downtime required) during the final after-image journal backup and apply stage in steps 8, 9, and 10.

B.7 Using an OpenVMS Pipe

You can use an OpenVMS pipe to pass data from the RMU Unload After_ Journal command to another application (for example, RMU Load). Do not use any options (such as the Log or Verify qualifiers) that could cause LogMiner to send extra output to the SYS$OUTPUT device, as that information would be part of the input data source stream to the next pipeline segment.

You may find that the OpenVMS default size of the pipe is too small if the records being extracted (including LogMiner fields) are larger than 256 bytes. If the pipe is too small, increase the SYSGEN parameter DEFMBXMXMSG (mailbox maximum message size) to accommodate the largest size required. This might also require that the SYSGEN parameter MAXBUF (maximum allowable size for a buffered I/O packet) also be increased - it defaults to 8192.

The following example uses LogMiner for Rdb to direct output to an OpenVMS pipe device and uses RMU Load to read the pipe device as the input data record stream. Using the pipeline allows parallel processing and also avoids the need for an intermediate disk file. Note that you must have created the record definition (.rrd) file prior to executing the command.

```
$ PIPE (RMU /UNLOAD /AFTER_JOURNAL OLTP.RDB AIJ1.AIJ -
    /TABLE = (NAME = MYTBL, OUTPUT = SYS$OUTPUT:)) -
     |  (RMU /LOAD REPORTS.RDB MYLOGTBL SYS$PIPE: -
        /RECORD_DEFINITION = FILE = MYLOGTBL.RRD)
```

C

Using Per-Process Monitoring for RMU Show Statistics

This appendix provides information about using Per-Process Monitoring for RMU Show Statistics.

RMU Show Statistics has a Per-Process Monitoring facility to provide a powerful drill-down capability to allow a database administrator to analyze process-specific information for a single process, class of processes, or all attached database processes. The facility also provides several screens that display a side-by-side comparison of individual process statistic information such as I/O, transaction, and record statistics.

The Per-Process Monitoring facility presents real-time information and, consequently, does not write its information to the binary output file. Therefore, the Per-Process Monitoring facility is not available during the replay of a binary input file.

The Per-Process Monitoring facility is also not available if cluster wide statistic collection is active. Conversely, the cluster wide statistic collection facility is not available when the Per-Process Monitoring facility is active.

The RMU Show Statistics utility requires SYSGBL privilege in order to use the Per-Process Monitoring facility. This privilege is required even if the facility activation is implicitly performed by the database monitor. See Section C.2, Per-Process Monitoring Facility Activation for activation details.

C.1 Per-Process Monitoring Operational Modes

The RMU Show Statistics utility provides two separate operational modes for the Per-Process Monitoring facility. These are:

* Process Overview Information

This operational mode allows you to compare statistic information for various activated processes for the purpose of easily identifying performance and behavioral discrepancies. The information presented is very similar to the main-menu summary screen information.

* Process Drill-Down Information

  This operational mode allows you to operate the RMU Show Statistics utility against a specific attached process. All of the statistic screens display information for that process only.

Together, these operational modes allow you to completely analyze process information, especially as it correlates to the statistic information of other processes.

C.2 Per-Process Monitoring Facility Activation

You can select the Per-Process Monitoring facility using any of the following methods:

* Global Activation

  Global per-process statistic collection means that all processes attached to the database are automatically eligible to be monitored using the RMU Show Statistics utility.

  The RMU Open command, with the Statistics qualifier, now supports the optional keyword [No]Process_Global, which indicates whether or not you are able to collect per-process statistics. The Statistics=Process_Global qualifier indicates that all processes attached to the database are eligible for per-process monitoring. The default qualifier, Statistics=Noprocess_ Global, indicates that attached processes are not automatically eligible; however, you can still activate these processes at run-time using one of the other activation methods described below.

  When you use the global activation method, the RMU Show Statistics utility changes the screen header Mode attribute from Online to Global.

  Oracle Corporation recommends this method only when there are a small number of active processes. The Statistics=Process_Global qualifier causes each process attached to the database on that node to create a sizable global section into which the process global statistic collection occurs. Oracle Corporation recommends that you activate process global statistic collection on a per-process basis. That is, you should activate only those processes you are currently interested in, and deactivate them when you are finished with them.

The following formula (expressed in bytes) defines the global section memory requirements for each process when it is activated:

```
4096 +
(512 * "# storage areas") +
((128 * "# row caches") / 4) +
(512 * "max # logical areas")
```

The maximum number of logical areas is determined using the RMU Dump Header command. Search the generated output for the line containing "Logical area count is n". Note that the logical area count is not necessarily the actual number of tables and indexes contained in the database.

For example, the MF_PERSONNEL.RDB database requires approximately 300K (580 pages) of global section backing store memory for each attached process by the facility. Obviously, this memory requirement increases as the size of the database increases.

* Local Activation

  Local per-process statistic collection means that selected processes attached to the database are automatically eligible to be monitored from the RMU Show Statistics utility.

  You can define the RDM$BIND_GLOBAL_STATISTICS logical name, located in the LNM$FILE_DEV name table, to identify those non-server database processes that are accessible. The default value "0" indicates that the process is not eligible for monitoring. The value "1" indicates that the process is eligible for monitoring.

  You can designate the database server processes eligibility using server-specific logical names (see below) also located in the LNM$SYSTEM_ TABLE name table.

  Note that the database recovery process (DBR) does not participate in the per-process monitoring facility.

  Oracle Corporation recommends this method only when you know in advance which processes, or class of processes, you intend to monitor. However, this is seldom the case.

* Dynamic Activation

  You can dynamically activate processes attached to the database, at run-time, for per-process monitoring using the RMU Show Statistics utility. You commonly use this method when you want to monitor a few specific processes that exhibit run-time behavior that requires more investigation.

  Refer to Section C.4, Per-Process Monitoring Run-Time Options for more information on this using this method.

The following table summarizes the logical names and their respective name table:

Logical Name	Name Table	Affected Process
RDM$BIND_GLOBAL_STATISTICS	LNM$FILE_DEV	Application processes
RDM$BIND_ABS_GLOBAL_STATISTICS	LNM$SYSTEM_TABLE	AIJ Backup Server
RDM$BIND_ALS_GLOBAL_STATISTICS	LNM$SYSTEM_TABLE	AIJ Log Server
RDM$BIND_LCS_GLOBAL_STATISTICS	LNM$SYSTEM_TABLE	Log Catch-up Server
RDM$BIND_LRS_GLOBAL_STATISTICS	LNM$SYSTEM_TABLE	Log Replication Server
RDM$BIND_RCS_GLOBAL_STATISTICS	LNM$SYSTEM_TABLE	Row Cache Server

C.3 Per-Process Monitoring Facility Process Activation

When you activate the Per-Process Monitoring facility, attached database processes are in one of the following states:

* Inactive State

  By default, attached database processes are inactive for per-process monitoring. This means that their per-process statistic information is not globally available to the RMU Show Statistics utility.

  You can activate inactive processes at run-time. Refer to Section C.4, Per-Process Monitoring Run-Time Options for more information.

* Eligible State

  Processes that are eligible for per-process monitoring, but that you are not currently monitoring, are identified on RMU Show Statistics utility screens with the "G" suffix in the Process.ID field. The "G" suffix means statistics are being collected globally for this process, but cannot be displayed by the RMU Show Statistics utility.

* Activated State

  The RMU Show Statistics utility is able to display overview information for any eligible process. The RMU Show Statistics utility identifies any process activated for per-process monitoring with the "A" suffix in the Process.ID field.

The fundamental difference between eligible and activated processes is that the RMU Show Statistics utility displays process-specific information for activated processes. You are only able to activate eligible processes.

For example, if you open the database using the RMU Open Statistics=Process_Global command, the "G" suffix identifies all eligible processes, as shown in the following screen:

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 16-JUN-2014 15:47:28.89
Rate: 1.00 Second          Active User Stall Messages   Elapsed: 159 01:35:39.64
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;3       Mode: Global
------------------------------------------------------------------------------
Process.ID  Elapsed.... T Stall.reason.............................. Lock.ID.
2A51334D:1s               waiting for AIJ journal lock 0 (PW)       3E007869
2A50D94E:1s               waiting for standby database activity request
2A54A759:53 02:40:05.77 W waiting for page 1:1091 (PR)              5D00FE93
2A51A75A:1G 02:58:00.85 W waiting for page 3:2181 (PR)              1D005613
2A55155B:1G 02:40:04.45 W waiting for page 1:1091 (PW)              7000495D
2A53135C:1G 02:47:42.57 W waiting for page 1:1200 (PW)              1D0087FB
2A51C603:1G 02:40:09.20 W waiting for page 1:1091 (PW)              7500B779
2A559A06:1G 02:47:36.84 W waiting for page 33:1711 (PW)             0700EDD2
2A50FA1A:1G 02:56:11.84 R waiting for page 1:1006 (PR)              5500EC46
2A469E1F:1G 02:47:05.52 W waiting for record 50:9591:0 (PR)         4600FC19
2A49061C:1G               locking page 33:1713

------------------------------------------------------------------------------
Config Exit Help LockID Menu >next_page <prev_page PageInfo Set_rate Write Zoom
```

When the RMU Show Statistics utility activates eligible processes, the "A" suffix identifies these processes, as shown in the following screen:

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 16-JUN-2014 15:47:28.89
Rate: 1.00 Second          Active User Stall Messages   Elapsed: 159 01:35:39.64
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;3       Mode: Global
------------------------------------------------------------------------------
Process.ID  Elapsed.... T Stall.reason.............................. Lock.ID.
2A51334D:1s               waiting for AIJ journal lock 0 (PW)       3E007869
2A50D94E:1s               waiting for standby database activity request
2A54A759:53 02:40:05.77 W waiting for page 1:1091 (PR)              5D00FE93
2A51A75A:1G 02:58:00.85 W waiting for page 3:2181 (PR)              1D005613
2A55155B:1G 02:40:04.45 W waiting for page 1:1091 (PW)              7000495D
2A53135C:1G 02:47:42.57 W waiting for page 1:1200 (PW)              1D0087FB
2A51C603:1A 02:40:09.20 W waiting for page 1:1091 (PW)              7500B779
2A559A06:1A 02:47:36.84 W waiting for page 33:1711 (PW)             0700EDD2
2A50FA1A:1G 02:56:11.84 R waiting for page 1:1006 (PR)              5500EC46
2A469E1F:1G 02:47:05.52 W waiting for record 50:9591:0 (PR)         4600FC19
2A49061C:1G               locking page 33:1713
------------------------------------------------------------------------------
Config Exit Help LockID Menu >next_page <prev_page PageInfo Set_rate Write Zoom
```

Note that it is possible to activate specific processes and leave others as eligible for future activation.

Upon startup, the RMU Show Statistics utility automatically activates any eligible processes. You can either implicitly or explicitly activate eligible processes that attach to the database after the RMU Show Statistics utility, depending on which screen you are currently displaying.

The following table summarizes the possible Process.ID suffix values:

Type Tag	Description
(blank)	Ordinary application process.
D	Process is being recovered by a database recovery (DBR) process.
R	Process is a remote-connection server.
s	Process is a database server (note that the "s" suffix is lowercase to differentiate it from the character "8").
u	Process is a database utility (note that the "u" suffix is lowercase to differentiate it from the character "0").
*	Process is attached on a remote node.
A	Process is activated by the RMU Show Statistics utility for global statistic collection.
G	Process is eligible for global statistic collection but is not activated.

C.4 Per-Process Monitoring Run-Time Options

The RMU Show Statistics utility provides run-time options to manage the Per-Process Monitoring facility. The run-time options are available in the Tools menu, obtained using the exclamation mark (!). Selecting the Process Monitoring option displays one or more of the following menu options:

* Activate all eligible processes

  The RMU Show Statistics utility displays this option when the current screen is not one of the process overview screens. Process overview screens automatically activate all eligible processes.

  Selecting this option activates all eligible processes on the current node.

* De-activate all processes

  The RMU Show Statistics utility displays this option when the current screen is not one of the process overview screens. Process overview screens automatically activate all eligible processes.

Selecting this option de-activates all previously activated processes. However, they are still eligible to be re-activated if needed.

* Activate specific eligible process

  The RMU Show Statistics utility displays this option when the current screen is not one of the process overview screens. Process overview screens automatically activate all eligible processes.

  Selecting this option activates the selected eligible processes on the current node that are not currently activated.

  Note that you cannot activate database server processes using this option. You must explicitly define the RDM$BIND_xxx_GLOBAL_STATISTICS logical name to activate database server processes.

  Also, it may take a few seconds for the selected process to activate itself. Activating an already activated process has no effect on the selected process.

* De-activate specific process

  The RMU Show Statistics utility displays this option when the current screen is not one of the process overview screens. Process overview screens automatically activate all eligible processes.

  Selecting this option de-activates the selected, previously activated processes.

* Select activated process to monitor

  The RMU Show Statistics utility displays this option when there are one or more processes activated for per-process monitoring.

  Selecting this option allows the RMU Show Statistics utility screens to display information for the specified process only.

  The Node portion of the screen header displays the identifier of the selected process you are currently monitoring.

* Cancel process monitoring

  The RMU Show Statistics utility displays this option when you have previously selected a specific activated process to monitor.

  Selecting this option reverts the RMU Show Statistics utility screens to display information for all database processes.

C.5 Detached Process Monitoring

If the process actively being monitored detaches from the database, the RMU Show Statistics utility screens will inquire whether or not you wish to continue monitoring the information of the detached process.

If you wish to continue monitoring the detached process, the process identifier in the screen header will blink. The blinking process identifier lets you know that the process information you are viewing is stale. In the event of an abnormal process termination, reviewing the stale information can be useful for determining the cause of the termination.

If you do not wish to continue monitoring the detached process, the RMU Show Statistics utility will automatically revert to displaying information for all database processes.

There is no method available to activate a process once it has detached from the database.

C.6 Per-Process Monitoring Overview Information

The RMU Show Statistics utility contains seven new screens, located in the Process Information menu, to provide overview and side-by-side comparison of information for all attached processes. The screens provide information similar to the main menu summary screens.

All of the process overview screens implicitly attach to any eligible process not already attached.

Note that you are unable to transfer statistic information from these screens onto the Custom Statistics screen.

The following is a list of the new screens:

* Process IO Overview
* Process Journal IO Overview
* Process Lock Overview
* Process Object Overview
* Process Transaction Overview
* Process Record Overview
* Process Snapshot Overview

Process IO Overview Screen

The Process IO Overview screen summarizes database I/O activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:29.66
Rate: 1.00 Second             Process IO Overview       Elapsed: 160 00:32:40.41
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2       Mode: Global
------------------------------------------------------------------------------
Process.ID Sync.Reads SyncWrites Read.Stall WriteStall AsyncReads AsyncWrits
2A46BE1F:1s         0          0          0          0          0          0
2A562A21:1s         0          0          0          0          0          0
2A56002D:1A      1058       4221       8230      11400          0          0
2A53DC2E:1A       566        560        333       9729          0          0
2A540E31:1A      5948       6764      32374       8377          0          0
2A532E37:1A       543        453        689       9240          0          0
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A       198       1634        743       3660         13          0
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* Sync.Reads

  This field gives the number of synchronous read queued I/O requests (QIO) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot (.SNP) files. This operation reads database pages synchronously from the database.

* SyncWrites

  This field gives the number of synchronous write queued I/O requests (QIO) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot (.SNP) files. This operation writes modified database pages synchronously back to the database.

* Read.Stall

  This field gives the time in hundredths of a second spent reading database pages from the database rootfile (.RDB), storage area (.RDA) files, and snapshot (.SNP) files. An excessively high number often indicates disk contention that might be alleviated by moving some files to other disks.

This statistic field includes both synchronous and asynchronous I/O read stall durations.

* WriteStall

  This field gives the time in hundredths of a second spent writing database pages to the database rootfile (.RDB), storage area (.RDA), and snapshot (.SNP) files. An excessively high number often indicates disk contention that might be alleviated by moving some files to other disks.

  This statistic field includes both synchronous and asynchronous I/O write stall durations.

* AsyncReads

  This field gives the number of asynchronous read queued I/O requests (QIO) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot (.SNP) files. This operation reads database pages asynchronously from the database.

* AsyncWrits

  This field gives the number of asynchronous write queued I/O requests (QIO) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot (.SNP) files. This operation writes modified database pages asynchronously back to the database.

Process Journal IO Overview Screen

The Process Journal IO Screen summarizes journal I/O activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:31.20
Rate: 1.00 Second         Process Journal IO Overview   Elapsed: 160 00:32:41.95
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2      Mode: Global
-------------------------------------------------------------------------------
Process.ID RUJ.Reads. RUJ.Writes RUJ.Extend AIJ.Reads. AIJ.Writes AIJ.Extend
2A46BE1F:1s         0          0          0         41       3960          0
2A562A21:1s         0          0          0        197          1          0
2A56002D:1A         0         71          1          4          0          0
2A53DC2E:1A         0        152          1          4          0          0
2A540E31:1A         0         13          1          3          0          0
2A532E37:1A         0        117          1          3          0          0
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A         0          2          1          2          0          0
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
-------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* RUJ.Reads

  This field gives the number of read queued I/O requests (QIO) issued to the database recovery unit journal (.RUJ) files. This operation reads before-image records from the .RUJ file to roll back a verb or a transaction.

  This statistic field includes both synchronous and asynchronous I/O read requests.

* RUJ.Writes

  This field gives the number of write queued I/O requests (QIO) issued to the database recovery unit journal (.RUJ) files. This operation writes before-image records to the .RUJ file in case a verb or transaction must be rolled back. Before-images must be written to the .RUJ file before the corresponding database page can be written back to the database.

  This statistic field includes both synchronous and asynchronous I/O write requests.

* RUJ.Extend

  This field identifies the number of times an RUJ file has been extended. Ideally, this value should be "0" or as close to "0" as possible. Each .RUJ file extension represents a performance bottleneck that is easily resolved.

* AIJ.Reads.

  The number of read queued I/O requests (QIO) issued to the database after-image journal (.AIJ) file. If after-image journaling is not enabled for the database, this statistic will be zero.

  This statistic field includes both synchronous and asynchronous I/O read requests.

* AIJ.Writes

  This field gives the total number of write queued I/O requests (QIO) issued to the database after-image journal (.AIJ) file. If after-image journaling is not enabled for the database, this statistic will be zero. This operation writes after-image records to the after-image journal to facilitate roll-forward recovery using the RMU Recover command.

  This statistic field includes both synchronous and asynchronous I/O write requests.

* AIJ.Extend

  This field identifies the number of times an AIJ journal has been extended. Ideally, this value should be "0" or as close to "0" as possible. Each AIJ journal extension represents a performance bottleneck that is easily resolved.

Process Lock Overview Screen

The Process Lock Overview screen summarizes database locking activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:32.23
Rate: 1.00 Second            Process Lock Overview      Elapsed: 160 00:32:42.98
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2       Mode: Global
------------------------------------------------------------------------------
Process.ID Lock.Rqsts Prom.Rqsts Deadlocks. Blasted... Demotes... Unlocks...
2A46BE1F:1s        64       7519          0        214       7456         22
2A562A21:1s      1098       1355          0        247        346       1087
2A56002D:1A      8293       8803          7       2779      13331       5347
2A53DC2E:1A     10104       4359          8       2621       3581       5908
2A540E31:1A     18831       3409          7        788      13662      16509
2A532E37:1A      7337       2982          4       1694       2508       4109
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A      6170       3836          0       1384       5604       3093
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* Lock.Rqsts

  This field gives the number of lock requests (also referred to as "enqueue" lock requests) for new locks. Whether the lock request succeeds or fails, it is included in this count.

* Prom.Rqsts

  This field gives the number of enqueue lock requests to promote an existing lock to a higher lock mode. Whether or not the lock request succeeds, it is included in this count.

* Deadlocks.

This field gives the number of stalled enqueue lock requests for both new and promoted locks that ultimately resulted in a deadlock. Most deadlocks are tried again and resolved by Oracle Rdb without the application program ever knowing there was a deadlock. Therefore, the number shown in this field does not necessarily reflect the number of deadlocks reported to the application program.

* Blasted...

  This field gives the number of blocking ASTs, sometimes referred to as "blasts", delivered to Oracle Rdb by the lock manager. A blocking AST is delivered to the holder of a lock when a lock conflict is detected, which is a good indication of contention problems. When Oracle Rdb receives a blocking AST, it often demotes or releases a lock in an attempt to avoid unnecessary deadlocks.

  The number of blocking ASTs reported is composed of two different types of blocking ASTs: those generated externally and those generated internally.

  An externally generated blocking AST occurs when a blocking AST is actually received by the process from the operating system in response to some lock conflict with another process. A blocking AST routine is executed and the RMU Show Statistics utility records the blocking AST activity.

  An internally generated blocking AST occurs when a lock-blocking AST routine is executed by the process in anticipation that the same work would have to be performed anyway if a blocking AST were to be received from the operating system. This algorithm serves as an optimistic code optimization. The process, assuming it would eventually receive a blocking AST for the particular lock, executes the blocking AST routine. The RMU Show Statistics utility does not differentiate between these two types of blocking ASTs.

* Demotes...

  This field gives the number of enqueue lock requests to demote an existing lock to a lower lock mode. These requests always succeed.

* Unlocks...

  This field gives the number of deallocating lock requests to release an existing lock. These requests always succeed.

Process Object Overview Screen

The Process Object Overview screen summarizes rootfile object activity for each attached process. The rootfile objects are the KROOT, FILID, SEQBLK, TSNBLK, AIJDB, AIJFB, RTUPB, ACTIVE bitmap, CPT, RCACHE, CLIENT, UTILITY, and the CLIENT SEQUENCE.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:33.34
Rate: 1.00 Second          Process Object Overview     Elapsed: 160 00:32:44.09
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2       Mode: Global
-------------------------------------------------------------------------------
Process.ID Objs.Shrd. Objs.Excl. Objs.Rfrsh Objs.Updt. Objs.Write Objs.Relsd
2A46BE1F:1s    135255         25         52         24         24     135280
2A562A21:1s      1497         48        103         20         20       1545
2A56002D:1A      6162       1712        107        510        510       7874
2A53DC2E:1A      5111       1620         70        479        479       6731
2A540E31:1A      1844        416         42        150        150       2260
2A532E37:1A      3190        953         61        297        297       4143
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A      3959       1016         57        302        302       4975
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
-------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* Objs.Shrd

  This field displays the number of objects that are fetched for shared retrieval access.

* Objs.Excl

  This field displays the number of objects that are fetched for exclusive access with the intention of subsequently being updated. This statistic does not indicate that the object was actually updated.

* Objs.Rfrsh

  This field displays the number of objects whose information in the global section was detected as being stale, so the information was read again from the database root file.

* Objs.Updt

  This field displays the number of objects whose information was modified. Only objects fetched for exclusive access can be modified.

* Objs.Write

  This field displays the number of objects whose information was written back to the database root file.

* Objs.Relsd

  This field displays the number of objects whose shared or exclusive access was released to other processes.

Process Transaction Overview Screen

The Process Transaction Overview screen summarizes transaction and checkpoint activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 18-JUN-2014 13:09:48.19
Rate: 1.00 Second         Process Transaction Overview  Elapsed: 160 22:57:58.94
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;3       Mode: Global
------------------------------------------------------------------------------
Process.ID Transactns TxCommited TxRollback Checkpoint VerbSucces VerbFailur
2A565E77:1s         2          2          0          0          2          0
2A53B278:1s         1          1          0          0          1          0
2A55DC81:38        22         22          0          1        204          0
2A54348E:1A         4          4          0          0         73          0
2A562A89:1A        53         53          0          8       3056          0
2A520A84:1A       542        542          0         43       3357          5
2A43A88C:1A       489        489          0         40       3030          4
2A561A8D:1A         1          1          0          0         60          0
2A539A90:1A         1          1          0          0         60          0
2A55FA8F:1A         1          1          0          0         60          0
------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* Transactns

  This field gives the number of completed database transactions. This is the count of the transaction COMMIT and ROLLBACK statements that have executed.

* TxCommited

  This field identifies the actual number of transactions that committed successfully to the database.

* TxRollback

This field identifies the actual number of transactions that aborted and were not applied to the database.

* Checkpoint

  This field identifies the number of checkpoints performed by users. This field does not include the initial checkpoint when the user first attaches to the database.

* VerbSucces

  This field gives the number of completed verbs that returned a successful status code.

  A verb is an atomic SQL statement or action. For example, a record insert is a verb, as is a record deletion.

  Also, within a compound statement, each individual statement is atomic and Oracle Rdb performs a verb-success operation after processing each one. To avoid this overhead, you can use the SQL BEGIN ATOMIC compound statement to treat the entire block as a single verb.

* VerbFailur

  This field gives the number of completed verbs that returned an error status code. Errors include end-of-collection and deadlocks, as well as all other exception conditions.

  A verb is an atomic SQL statement or action. For example, a record insert is a verb, as is a record deletion.

  Excessive verb failures are usually an indication of a failed constraint, such as uniqueness criteria or an invalid data definition language (DDL) statement. Note that in the case of cursors and scans, reaching the end-of-stream always results in a verb failure.

  Note that SQL performs its own internal queries to identify metadata, such as relation or index names.

  Oracle Rdb rarely issues a verb-failure unless there is an exception of some kind, such as a constraint failure.

Process Record Overview

The Process Record Overview screen summarizes database record activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:34.65
Rate: 1.00 Second           Process Record Overview     Elapsed: 160 00:32:45.40
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2       Mode: Global
-------------------------------------------------------------------------------
Process.ID RecFetched Rec.Marked Rec.Stored Pag.Checkd PagDiscard Rec.Erased
2A46BE1F:1s         0          0          0          0          0          0
2A562A21:1s         0          0          0          0          0          0
2A56002D:1A    276020      50497      23890      23890          0          0
2A53DC2E:1A      2676        475        448        448          0          0
2A540E31:1A     42975      12291      10341      10492        151          0
2A532E37:1A      1845        286        273        273          0          0
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A    164922      32011      15046      15046          0          0
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
-------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* RecFetched

  This field gives the number of records fetched, including snapshot records. This field does not include records retrieved from a temporary table.

  Note that this value may be more than the actual number of records returned by a query. The reason is that queries may fetch records during the search phase and then re-fetch the selected records so that they may be returned to the user. Also, for uniform format storage areas, a sequential scan needs to fetch the next record on each page of the clump, even if there are no records on that page. In addition, every page in a uniform format storage area incurs an extra fetch to verify that there are no more records residing on that page.

* Rec.Marked

  This field gives the number of records marked. A record is marked when it is modified or erased, but not when it is stored. This field does not include records modified in a temporary table.

* Rec.Stored

  This field gives the number of records stored in the database. This field does not include records stored in temporary tables.

* Pag.Checkd

This field indicates the number of pages checked in order to store a record. Ideally, very few candidate pages need to be checked when storing a record. However in certain cases, depending on record size, access method, locked space on a page, and SPAM thresholds, storing a record requires a number of page fetches.

* PagDiscard

  This field identifies the number of pages checked but discarded because the actual free space on that page did not meet the physical requirements needed to store a new record.

* Rec.Erased

  This field gives the number of records erased from the database. This field does not include records erased from temporary tables.

Process Snapshot Overview Screen

The Process Snapshot Overview screen summarizes snapshot area activity for each attached process.

```
Node: GANDLF (1/3/24)   Oracle Rdb V7.3-13 Perf. Monitor 17-JUN-2014 14:44:36.44
Rate: 1.00 Second          Process Snapshot Overview    Elapsed: 160 00:32:47.19
Page: 1 of 1       USER_TEST:[DB_TESTING.TCS_MASTER]TCS.RDB;2      Mode: Global
-------------------------------------------------------------------------------
Process.ID SnpRecRtrv SnpLinFtch SnpPagRead SnpRecStor SnpPagFull SnpLckCnft
2A46BE1F:1s         0          0          0          0          0          0
2A562A21:1s         0          0          0          0          0          0
2A56002D:1A       303          0          0       6581        916          0
2A53DC2E:1A         0          0          0        193         50          2
2A540E31:1A       303          0          0       3467         29          0
2A532E37:1A         0          0          0        157         38          0
2A52A43B:1A         0          0          0          0          0          0
2A536E3A:1A       303          0          0         70          1          0
2A50C047:1A         0          0          0          0          0          0
2A4CDE49:1A         0          0          0          0          0          0
-------------------------------------------------------------------------------
Exit Help Menu >next_page <prev_page Options Set_rate Write Zoom !
```

The screen fields are the following:

* Process.ID

  Identifies the attached database process. All processes on this screen contain the "A" suffix unless the process is a database server.

* SnpRecRtrv

  This field gives the number of records retrieved by read-only transactions.

* SnpLinFtch

  This field gives the number of lines fetched by read-only transactions. To retrieve a single record, a transaction might actually check a number of lines, some of which may be empty.

* SnpPagRead

  This field gives the number of snapshot pages fetched by read-only transactions. If this count is high relative to the other read fields, read-only transactions are fetching records that are being updated frequently and the snapshot file is being used extensively.

* SnpRecStor

  This field gives the number of records stored in the snapshot file by update transactions. Every snapshot record stored by an update transaction implies that a snapshot page was found and utilized. In the best case, this is a single-page fetch. The page in use, page too full, page conflict, and extended file sub-fields indicate some of the extra overhead involved in finding suitable snapshot pages on which to store snapshot records.

* SnpPagFull

  This field gives the number of pages fetched that were unsuitable for storing snapshot records because there was not enough room on the snapshot page to include another version of a record. In this case, a new snapshot page must be fetched and linked with the full page. This allows read-only transactions to follow a chain of snapshot pages to find the correct version of a record.

* SnpLckCnft

  This field gives the number of times a snapshot page fetch was requested but aborted due to a lock conflict with another process. When a page fetch conflicts with another process, another target page is fetched and checked so the lock conflict does not cost a disk I/O operation.

Index

A

ABM pages
  repairing, 1–511
  verifying, 1–902
Access Control Entry
  See ACE
Access control list
  See ACL
Access privilege set
  See APS
ACE
  for root file, 1–696
ACL
  auditing changes to, 1–664
  delete protections, 1–303
  extracting, 1–302
  modifying the root file, 1–696
Activating the Oracle Rdb monitor process, 1–414
Adding .aij files, 1–634
Adding AIJ cache, 1–634
After-image journal (.aij) file, 1–117
  See also Journaling, After-image journaling
  adding
    using RMU Copy_Database command, 1–195
    using RMU Move_Area command, 1–423
    using RMU Restore command, 1–536
    using RMU Restore Only_Root command, 1–588
    using RMU Set After_Journal command, 1–634
After-image journal (.aij) file (cont'd)
  automatic recovery of, 1–527
  backing up, 1–118
  backing up to tape, 1–120, 1–126
    See also Tapes
  backup file, 1–118
  copying, 1–117
  creating
    using RMU Copy_Database command, 1–195
    using RMU Move_Area command, 1–423
    using RMU Restore command, 1–536
    using RMU Restore Only_Root command, 1–588
    using RMU Set After_Journal command, 1–635
  database convert operation and, 1–182
  displaying output, 1–240
  displaying unresolved transactions, 1–250
  dropping
    using RMU Set After_Journal command, 1–634
  extracting record contents, 1–861
  initialization of I/O buffers, 1–145
    RDM$BIND_AIJ_INITIALIZE_IO_COUNT logical name, 1–145
    RDM$BIND_AIJ_INITIALIZE_IO_SIZE logical name, 1–145
  managing, 1–117
  modifying
    using RMU Set After_Journal command, 1–634
  optimizing, 1–452
  overwriting, 1–634
  protection on backup file, 1–135

After-image journal (.aij) file (cont'd)
reserving
using RMU Convert command, 1–185
using RMU Copy_Database command, 1–195
using RMU Move_Area command, 1–423
using RMU Restore command, 1–536
using RMU Restore Only_Root command, 1–588
using RMU Set After_Journal command, 1–634
suppressing use of, 1–634
unloading
using RMU Unload After_Journal command, 1–861
using to recover a database, 1–476, 1–527
After-image journal (AIJ) log server
See ALS
After-image journaling
disabling, 1–634
enabling, 1–634
resuming backup of, 1–627
suspending backup of, 1–629
AIJ Backup Server, 1–629
AIJ cache
adding, 1–642
disabling, 1–642
enabling, 1–642
AIJ object definition
extracting, 1–299
AIJ temporary work file
RDM$BIND_AIJ_WORK_FILE logical name, 1–465, 1–491
RDM$BIND_DBR_WORK_FILE logical name, 1–491
AIP
count in a storage area, 1–21
verifying, 1–902
AIP entries
correcting, 1–519
Alarms
security, 1–658
Allocating storage
improving performance, 1–519
Allocating tape drive
for .aij backup file, 1–144
for database backup file, 1–99
ALS
reopening the output file, 1–621
starting, 1–623
stopping, 1–625
Alter command, 1–12
See also RdbAlter
correcting AIP entries, 1–519
Analyze Cardinality command, 1–29
Confirm qualifier, 1–31
description, 1–29
examples, 1–34
Output qualifier, 1–32
storing cardinality of tables and indexes, 1–29
Transaction_Type qualifier, 1–32
Update qualifier, 1–33
Analyze command, 1–14
Areas qualifier, 1–14
Binary_Output qualifier, 1–15
description, 1–14
End qualifier, 1–16
examples, 1–22
Exclude=options qualifier, 1–16
gathering statistics, 1–14
Lareas qualifier, 1–17
Option qualifier, 1–17
Output qualifier, 1–17
Partition qualifier, 1–17
Start qualifier, 1–18
syntax for .rrd file, A–10
Analyze Indexes command, 1–35
Binary_Output qualifier, 1–38
description, 1–35
examples, 1–45
Exclude=Metadata qualifier, 1–39
gathering statistics, 1–35
Option qualifier, 1–39
Output qualifier, 1–40
Partitions qualifier, 1–40
Transaction_Type qualifier, 1–40

Analyze Placement command, 1–50
Areas qualifier, 1–51
Binary_Output qualifier, 1–52
description, 1–50
examples, 1–57
Exclude=Metadata qualifier, 1–52
gathering statistics, 1–50
Option qualifier, 1–53
Output qualifier, 1–53
Partitions qualifier, 1–53
Transaction_Type qualifier, 1–54
Analyzing cardinality of disabled indexes, 1–31
Analyzing compressed index keys, 1–39
Analyzing databases, 1–14, 1–50
Analyzing disabled index structures, 1–38
Analyzing placement of an index, 1–51
Analyzing placement of disabled indexes, 1–51
Analyzing record placement relative to indexes, 1–50
APS
auditing changes to, 1–664
Area bit map pages
See ABM pages
Area inventory page
See AIP
AREA . . . PAGE command (RdbALTER), 2–4
Area qualifiers
See Parameter qualifiers
ATTACH command (RdbALTER), 2–6
releasing, 2–27
Attaching RdbALTER to a database, 2–6
Audit event
disabling DACCESS, 1–659
enabling DACCESS, 1–659
Audit event classes, 1–659
Auditing
ACL changes, 1–664
APS changes, 1–664
characteristics, 1–752
disabling, 1–658
enabling, 1–658, 1–659
specific objects, 1–661
specific object types, 1–660
starting, 1–665
Auditing (cont'd)
stopping, 1–665
use of Oracle RMU commands, 1–664
Audit journal, 1–658
Audit journal records
creating a database for storing, 1–377
defining a table for storing, 1–377, 1–378
Automatic .aij file recovery during a restore operation, 1–527, 1–563

B

Backing up a database, 1–65, 1–151
to tape
See also Tape, 1–66
Backing up a database by area
See Backup command
Backing up an .aij file, 1–118
to tape, 1–118
improving performance of, 1–123
Backup After_Journal command, 1–117
Accept_Label qualifier, 1–122
Active_IO qualifier, 1–123
Block_Size qualifier, 1–123
Compression qualifier, 1–123
Continuous qualifier, 1–124
Crc=Autodin_II qualifier, 1–125
Crc=Checksum qualifier, 1–125
Density qualifier, 1–126
Edit_Filename qualifier, 1–127
Encrypt qualifier, 1–128
examples, 1–146
Format qualifier, 1–129
Group_Size qualifier, 1–130
Interval qualifier, 1–130
Label qualifier, 1–131
Librarian qualifier, 1–132
Lock_Timeout qualifier, 1–133
Log qualifier, 1–134
Media_Loader qualifier, 1–134
Nocrc qualifier, 1–126
Owner qualifier, 1–134
Prompt qualifier, 1–135
Protection=file-protection qualifier, 1–135
Quiet_Point qualifier, 1–135

Backup After_Journal command (cont'd)
Rename qualifier, 1–136
Rewind qualifier, 1–137
Sequence=(n,m) qualifier, 1–137
Space_Check qualifier, 1–138
Tape_Expiration=date-time qualifier, 1–139
Threshold qualifier, 1–140
Until qualifier, 1–140
Wait qualifier, 1–141
Backup command, 1–65
Accept_Label qualifier, 1–71
Acl qualifier, 1–71
Active_IO qualifier, 1–72
Allocation qualifier, 1–72
Assist qualifier, 1–72, 1–123, 1–241, 1–263, 1–455, 1–480, 1–538, 1–589
backing up a database, 1–65
backing up a database by area, 1–77
Block_Size qualifier, 1–72
by area, 1–77
Checksum_Verification qualifier, 1–72
Compression qualifier, 1–73
Crc=Autodin_II qualifier, 1–73
Crc=Checksum qualifier, 1–74
Database_Verification qualifier, 1–74
Density qualifier, 1–75
Disk_File qualifier, 1–75
Encrypt qualifier, 1–76
examples, 1–108
Exclude qualifier, 1–77
Execute qualifier, 1–78
Exit_Error qualifier, 1–264
Extend_Quantity qualifier, 1–79
format, 1–65
Group_Size qualifier, 1–79
Include qualifier, 1–79
Incremental qualifier, 1–80
Journal qualifier, 1–82
Label qualifier, 1–82
Librarian qualifier, 1–83
List_Plan qualifier, 1–86
Loader_Synchronization qualifier, 1–86
Lock_Timeout qualifier, 1–87
Log qualifier, 1–88
Master qualifier, 1–88
Backup command (cont'd)
Media_Loader qualifier, 1–88
Nocrc qualifier, 1–74
No_Read_Only qualifier, 1–89
Online qualifier, 1–89
options file, 1–95
Output qualifier, 1–90, 1–551
Owner qualifier, 1–92
Owner_Uic qualifier, 1–92
Page_Buffers qualifier, 1–92
Parallel qualifier, 1–92
plan file, 1–67, 1–86, 1–151
Progress_Report qualifier, 1–94
Prompt qualifier, 1–94
Protection=file-protection qualifier, 1–94
Quiet_Point qualifier, 1–95
Reader_Thread_Ratio qualifier, 1–95
Record qualifier, 1–89
Restore_Options qualifier, 1–95
Rewind qualifier, 1–96
Scan_Optimization qualifier, 1–96
Tape_Expiration=date-time qualifier, 1–97
Threads qualifier, 1–98, 1–204, 1–429
truncating file names, 1–103
Backup file
.aij file, 1–117
protection on, 1–135
database
protection on, 1–94
displaying output, 1–261
size of, 1–102
truncation of name, 1–103
Backup operation
after-image journal (.aij) file, 1–117
by area, 1–77
database, 1–65, 1–527
multithreaded, 1–66
online, 1–89
parallel, 1–66
to tape
See Tape
Backup performance
job priority and, 1–103

Backup Plan command, 1–151
Database_Verifcation qualifier, 1–151
examples, 1–153
Execute qualifier, 1–152
List_Plan qualifier, 1–152
Progress_Report qualifier, 1–152
Buffers
communications, 1–349, 1–365
database, 1–357, 1–365
global, 1–198, 1–542
local, 1–199, 1–547
monitor buffers available, 1–818, 1–820
page, 1–92, 1–201, 1–428, 1–552
Buffer_Object features
disabling, 1–678
enabling, 1–678
By-area backup operation, 1–77
By-area restore operation, 1–534, 1–537

C

Cardinality of tables and indexes
RMU Analyze Cardinality command, 1–29
Cardinality statistics
collecting, 1–167
Case sensitivity, 1–379
command qualifiers, 1–2
RMU Extract command and, 1–296
RMU Unload command and, 1–844
Characteristics
security auditing, 1–752
Character set
with RMU Extract command, 1–296
with RMU Load command, 1–379
with RMU Unload command, 1–844
Character sets, A–6
Checking internal structures, 1–902
Checkpoint command, 1–158
Wait qualifier, 1–159
Checkpoint operation, 1–158
Classes of audit events, 1–659
Close command, 1–161
Abort qualifier, 1–162
Delprc option, 1–162
Forcex option, 1–162
Close command (cont'd)
Cluster qualifier, 1–163
eliminating active users, 1–161, 1–165
examples, 1–166
Path qualifier, 1–164
Wait qualifier, 1–165
Closing a database, 1–161
Closing a monitor log file, 1–412
Collating sequence
extracting from database, 1–299
Collecting cardinality statistics, 1–167
Collecting optimizer statistics, 1–167
Collecting storage statistics, 1–167
Collecting workload statistics, 1–167
Collect Optimizer_Statistics command, 1–167
examples, 1–177
Exclude_Tables qualifier, 1–170
Indexes qualifier, 1–171
Log qualifier, 1–171
Row_Count qualifier, 1–171
Statistics qualifier, 1–171
System_Relations qualifier, 1–172
Tables qualifier, 1–173
Transaction_Type qualifier, 1–173
Command parameters, 1–2
Command qualifiers, 1–2
case sensitivity of, 1–2
Commands
auditing the use of, 1–664
Command syntax, xii, xiii, 1–11
COMMIT command (RdbALTER), 2–8
Committing changes
with RdbALTER, 2–8
Communications buffers, 1–349, 1–365
Compatibility
of lock modes, 1–768t
Compressed index keys
analyzing, 1–39
Compression ratio
of index keys, 1–37
Constraint
extracting from database, 1–299
Constraint evaluation
RMU Load command and, 1–358

Constraint violations
RMU Load command and, 1–368
Continuous LogMiner
Using the Unload After_Journal command, 1–864
Conventions
for command format, xii
used in manual, xv
Conversion of data types
by RMU Load command, 1–354t
Conversion of multifile databases, 1–183
Convert command, 1–182
after-image journaling and, 1–182
called by RMU Restore command, 1–527
Commit qualifier, 1–184
committing a database conversion, 1–184
Confirm qualifier, 1–185
enabling user input during conversion, 1–185
function, 1–182
multisegment index cardinality update, 1–186
Path qualifier, 1–185
Reserve qualifier, 1–185
reserving .aij files, 1–185
reserving storage areas, 1–185
Rollback qualifier, 1–186
rolling back a converted database, 1–186
specifying a path name
Path qualifier, 1–185
versions of Oracle Rdb supported, 1–182
Converting a database
RMU Convert command, 1–182
RMU Restore command, 1–527
Copying a database
See Copy_Database command
Copying an .aij file, 1–117
Copy_Database command, 1–193
After_Journal qualifier, 1–194
Aij_Options qualifier, 1–195
Attributes qualifier, 1–196, 1–539
Blocks_Per_Page qualifier, 1–205
Cdd_Integrate qualifier, 1–196
Checksum_Verification qualifier, 1–197
Directory qualifier, 1–197
Duplicate qualifier, 1–198
examples, 1–209
Copy_Database command (cont'd)
Extension qualifier, 1–205
File qualifier, 1–206
Global_Buffers qualifier, 1–198
Local_Buffers qualifier, 1–199
Lock_Timeout qualifier, 1–199
Log qualifier, 1–200
Nodes_Max qualifier, 1–200
Online qualifier, 1–200
Open_Mode qualifier, 1–200
Option qualifier, 1–201
Page_Buffers qualifier, 1–201
Path qualifier, 1–201
Quiet_Point qualifier, 1–201
Read_Only qualifier, 1–206
Read_Write qualifier, 1–206
Root qualifier, 1–202
Row_Cache_Options qualifier, 1–202
Snapshots qualifier, 1–206
Spams qualifier, 1–207
Thresholds qualifier, 1–207
Transaction_Mode qualifier, 1–203
Users_Max qualifier, 1–205
Correcting ABM errors, 1–510
Correcting AIP entries
with RMU Repair command, 1–519
Correcting list area problems, 1–510
Correcting page tail errors, 1–510
Correcting performance problems, 1–510
Correcting segmented string area problems, 1–510
Correcting SPAM problems, 1–510
Corrupted database
recovery of, 1–507
Corrupt pages
setting to consistent, 1–681
Creating a backup .aij file, 1–117
Creating a backup copy of database, 1–65
Creating a duplicate database
with RMU Copy_Database command, 1–193
CSN values, 1–590, 1–595
initializing, 1–590
setting, 1–584, 1–588, 1–589, 1–595, 1–602

.csv file, 1–351

D

DACCESS audit events
  enabling or disabling, 1–659
    for specific objects, 1–660
  privileges for database objects, 1–662
Database
  analyzing, 1–14
  changing
    after-image journal (.aij) file specification, 1–634
    recovery-unit journal (.ruj) file specification, 2–22
    recovery-unit journal (.ruj) unique identifier, 2–24
    root file specification, 2–22
    root unique identifier, 2–24
  closing, 1–161
  creating a duplicate of, 1–193
  displaying
    recovery-unit journal (.ruj) unique identifier, 2–43
    root unique identifier, 2–43
  displaying .aij file configuration, 1–733
  displaying contents of database, 1–221
  displaying corrupt page table entries, 1–757
  displaying fields, 2–28
  displaying security auditing characteristics, 1–752
  extracting attributes and characteristics, 1–300
  for storing security audit journal records, 1–377
  moving, 2–1
    data, 2–50
  online backup operation, 1–65
  opening, 1–443
  patching, 2–1
  performing a checkpoint operation, 1–158
  reclaiming, 1–471
  recovery, 1–476, 1–507
    single-file databases, 1–489
  repairing page corruption, 1–510
Database (cont'd)
  resolving an unresolved transaction, 1–507, 1–523
  restoring, 1–527
    from full backup file, 1–527, 1–530
    from incremental backup file, 1–527, 1–530
    root file, 1–583
  uncorrupting, 2–56
  unloading tables or views, 1–827
  verifying integrity, 1–902
Database backup operation, 1–65
Database buffers, 1–357, 1–365
Database migration
  converting to higher version, 1–182
Database pages
  checksum verification of, 1–904
  initializing, 1–513
  purging unused space from, 1–513
Database recovery (DBR) process
  using RMU Recover command, 1–476
Database table
  for storing security audit journal records, 1–377, 1–378
Database users
  eliminating
    with RMU Close command, 1–161
Data structures
  checking integrity, 1–902
Data types
  conversion of by RMU Load command, 1–354t
  date-time, A–10
  .rrd file, A–6
Date-time data type, A–10
DBR process
  See Database recovery (DBR) process
DEFINE FIELD statement
  .rrd file, A–1
DEFINE RECORD statement
  .rrd file, A–2
Delete Optimizer_Statistics command, 1–216
  Column_Group qualifier, 1–217
  examples, 1–219
  Log qualifier, 1–217
  Tables qualifier, 1–217

Deleting
column duplicity factor from optimizer statistics, 1–216
null factor statistics from optimizer statistics, 1–216
optimizer statistics, 1–216
workload statistics, 1–216
Delimited identifier
with RMU Load command, 1–379
with RMU Unload command, 1–844
DEPOSIT AREA_HEADER command (RdbALTER), 2–15
DEPOSIT command (RdbALTER), 2–9
DEPOSIT FILE command (RdbALTER), 2–20
DEPOSIT ROOT command (RdbALTER), 2–22
DEPOSIT ROOT UNIQUE_IDENTIFIER command (RdbALTER), 2–24
DETACH command (RdbALTER), 2–27
Detaching RdbALTER from a database, 2–27
Disabling .aij file overwriting, 1–634
Disabling after-image journaling, 1–634
Disabling AIJ caching, 1–634
Disabling DACCESS audit events, 1–659
Disabling Galaxy features, 1–690
Disabling logminer, 1–695
Disabling security alarms, 1–658
DISPLAY AREA_HEADER command (RdbALTER), 2–35
DISPLAY command (RdbALTER), 2–28
DISPLAY FILE command (RdbALTER), 2–39
Displaying .aij file in ASCII format, 1–240
Displaying .metadata file in ASCII format, 1–285
Displaying .oaij file in ASCII format, 1–240
Displaying .rbr file in ASCII format, 1–281
Displaying .ruj file in ASCII format, 1–288
Displaying Area Inventory Pages characteristics, 1–747
Displaying Audit file in ASCII format, 1–254
Displaying cardinality statistics, 1–780
Displaying contents
of database root (.rdb) file, 1–221
of snapshot (.snp) file, 1–221
of storage area (.rda) file, 1–221
Displaying database information, 1–1, 1–818
Displaying formatted .unl file in ASCII format, 1–281
Displaying information about users, 1–820
for a cluster, 1–230
for a node, 1–820
Displaying locks, 1–759
Displaying logical names, 1–777
Displaying optimizer statistics, 1–780
Displaying Oracle Rdb version number, 1–823
Displaying output
after-image journal, 1–240
Audit file, 1–254
backup file, 1–261
database, 1–221
metadata file, 1–285
optimized after-image journal, 1–240
recovery-unit journal, 1–288
Displaying root file ACL, 1–787
Displaying security audit characteristics, 1–752
Displaying statistics, 1–790
Displaying storage statistics, 1–780
Displaying workload statistics, 1–780
DISPLAY ROOT command (RdbALTER), 2–41
DISPLAY ROOT UNIQUE_IDENTIFIER command (RdbALTER), 2–43
Distributed transaction
resolving, 1–523
Documentation
conventions, xv
Domain definition
extracting, 1–300
Dropping after-image journal (.aij) files, 1–634
Dump After_Journal command, 1–240
Active_IO qualifier, 1–241
Area qualifier, 1–241
Data qualifier, 1–242
Encrypt qualifier, 1–242
End qualifier, 1–242
examples, 1–251
First qualifier, 1–242
Format qualifier, 1–243
Label qualifier, 1–244
Larea qualifier, 1–244
Last qualifier, 1–244

Dump After_Journal command (cont'd)
Library qualifier, 1–245
Line qualifier, 1–246
Media_Loader qualifier, 1–246
Only qualifier, 1–247
Option qualifier, 1–248
Output qualifier, 1–249
Page qualifier, 1–249
Prompt qualifier, 1–249
Rewind qualifier, 1–249
Start qualifier, 1–249
State=Prepared qualifier, 1–250
Dump Audit command, 1–254
Before qualifier, 1–255
Format qualifier, 1–255
Log qualifier, 1–255
Output qualifier, 1–255
Since qualifier, 1–255
Type qualifier, 1–255
Dump Backup_File command, 1–261
Active_IO qualifier, 1–263
Area qualifier, 1–263
Disk_File qualifier, 1–263
Encrypt qualifier, 1–264
End qualifier, 1–264
examples, 1–274
Header_Only qualifier, 1–265
Journal qualifier, 1–265
Label qualifier, 1–266
Librarian qualifier, 1–266
Media_Loader qualifier, 1–269
Options qualifier, 1–269
Output qualifier, 1–271
Page_Number qualifier, 1–271
Process qualifier, 1–272
Prompt qualifier, 1–272
Restore_Options qualifier, 1–272
Rewind qualifier, 1–272
Skip qualifier, 1–273
Start qualifier, 1–273
Dump command, 1–221
ABMS_Only qualifier, 1–222
Areas qualifier, 1–223
Directory_Option qualifier, 1–223
End qualifier, 1–224
Dump command (cont'd)
examples, 1–231
Header qualifier, 1–224
Lareas qualifier, 1–227
Option qualifier, 1–227
options file, 1–228
Output qualifier, 1–228
Page_Number qualifier, 1–228
Restore_Options qualifier, 1–228
Snapshots qualifier, 1–228
Spams_Only qualifier, 1–229
Start qualifier, 1–229
State=Blocked qualifier, 1–229
Users qualifier, 1–230
Dump Export command, 1–281
Data qualifier, 1–281
examples, 1–283
Options qualifier, 1–281
Output qualifier, 1–282
Dump Metadata_File command, 1–285
Data qualifier, 1–286
Output qualifier, 1–286
Dump Recovery_Journal command, 1–288
Data qualifier, 1–288
Output qualifier, 1–289
Dump Row_Cache command, 1–290
Cache_Name qualifier, 1–291
Data qualifier, 1–291
Output qualifier, 1–291
Dump Symbols command, 1–293
Execute qualifier, 1–293
Output qualifier, 1–293
Prefix qualifier, 1–294
Duplicating a database, 1–198

E

Eliminating active users
RMU Close command, 1–161
Enabling .aij file overwriting, 1–634
Enabling after-image journaling, 1–634
Enabling AIJ caching, 1–634
Enabling Buffer_Object features, 1–678

Enabling DACCESS audit events, 1–659
Enabling Galaxy features, 1–690
Enabling logminer, 1–695
Enabling security alarms, 1–659
Enabling security auditing, 1–658, 1–659
Exclusive update lock (RdbALTER), 2–6, 2–27
EXIT command (RdbALTER), 2–45
Expiration date
  specifying for .aij backup file, 1–139
  specifying for database backup file, 1–97
  specifying for optimized .aij file, 1–463
Export file
  exporting output, 1–281
  RMU Dump Export command, 1–281
External procedure
  extracting, 1–302
Extract command, 1–295
  access control list (ACL) definition, 1–302
  AIJ object definition, 1–299
  character set and, 1–296
  collating sequence, 1–299
  command procedure for database load, 1–301
  command procedure for database unload, 1–301
  command procedure for database verify, 1–305
  constraints, 1–299
  contents of catalog created for an SQL multischema database, 1–299
  database, 1–300
  Defaults qualifier, 1–297
  delete protections from ACLs, 1–303
  disabling header output, 1–317
  disabling hidden objects output, 1–317
  display profiles, 1–302
  Domains, 1–300
  examples, 1–328
  external function, 1–300
  external procedure, 1–302
  grouping by table, 1–317
  index definition, 1–301
  Items qualifier, 1–299
  Journals keyword, 1–317
  Language qualifier, 1–310
  Log qualifier, 1–313
Extract command (cont'd)
  Options qualifier, 1–313
  Output qualifier, 1–321
  parameters for generated verify command file, 1–306t
  query outline definition, 1–302
  recompile entry, 1–302
  revoke entry, 1–303
  RMU Set Audit command, 1–303
  role definitions, 1–303
  schema definition for SQL multischema database, 1–303
  SQL ALTER SEQUENCE statement, 1–303
  SQL CREATE SEQUENCE statement, 1–303
  SQL IMPORT script, 1–300
  storage map definition, 1–303
  stored procedure and function definition, 1–302
  table definition, 1–304
  trigger, 1–305
  user definitions, 1–305
  using qualifiers to determine output selection, 1–322t
  view, 1–307
  work load, 1–309

F

File extension
  .aij, 1–634
  .aij_rbf, 1–122, 1–240
  default, 1–4
  .metadata, 1–285
  .oaij, 1–117
  .opt, 1–4
  .plan, 1–151, 1–364, 1–404
  .rbf, 1–68
  .rbr, 1–281
  .rda, 1–69
  .rdb, 1–468, 2–6
  .rrd, 1–16
  .ruj, 1–288
  .snp, 1–90
  .unl, 1–828

File name
truncation of during backup operation, 1–103
File qualifiers
See Parameter qualifiers
File specification
changing with RdbALTER, 2–15, 2–20, 2–22, 2–24
DEPOSIT AREA_HEADER command, 2–15
DEPOSIT FILE command, 2–20
indirect, 1–4
Format
conventions, xii
Function
extracting, 1–300

G

Galaxy features
disabling, 1–690
enabling, 1–690
Gathering statistics
for the optimizer, 1–167
on database logical area space, 1–14
on database page space, 1–14
on database storage area space, 1–14
on index structure, 1–35
on row placement relative to index structures, 1–50
Global buffers, 1–198, 1–542
Global process symbols, 1–120, 1–738

H

Help command, 1–12
HELP command (RdbALTER), 2–46

I

Importing data, 1–301, 1–349
Improving sequential access performance, 1–513
Index
analyzing, 1–35
analyzing placement of, 1–51
disabled
analyzing cardinality of, 1–31
Index
disabled (cont'd)
analyzing placement of, 1–51
analyzing structure of, 1–38
RMU Load command and, 1–360
showing statistics
on index structure, 1–35
RMU Show Statistics command, 1–790
Index definition
extracting, 1–301
Index keys
compressed
analyzing, 1–39
compression ratio of, 1–37
Indirect command file, 1–4
Initializing CSN values, 1–590
Initializing database pages, 1–513
Initializing TSN values, 1–590
Inserting
column duplicity factor into optimizer statistics, 1–341
null factor statistics into optimizer statistics, 1–341
optimizer statistics, 1–341
workload statistics, 1–341
Insert Optimizer_Statistics command, 1–341
Column_Group qualifier, 1–342
Duplicity_Factor qualifier, 1–342
examples, 1–344
Log qualifier, 1–342
Null_Factor qualifier, 1–342
Tables qualifier, 1–342
Invoking RdbALTER utility, 1–12
Invoking the Performance Monitor, 1–790

J

Journal
security audit, 1–658
Journaling
after-image (.aij) file
automatic, 1–530
mechanism described, 1–117
using to recover a database, 1–476

L

Librarian command, 1–346
  List qualifier, 1–346
  Remove Qualifier, 1–348
LIST OF BYTE VARYING data
  repairing corruption to, 1–510
Load command, 1–349
  Audit qualifier, 1–356
  Before qualifier, 1–356
  Buffers qualifier, 1–357
  case sensitivity and, 1–379
  Commit_Every qualifier, 1–357
  constraint evaluation, 1–358
  Constraints qualifier, 1–358
  constraint violations and, 1–368
  Corresponding qualifier, 1–360
  Defer_Index_Updates qualifier, 1–360
  delimited identifier and, 1–379
  delimited text null string, 1–369
  delimited text prefix string, 1–368
  delimited text separator string, 1–369
  delimited text suffix string, 1–369
  delimited text terminator string, 1–369
  Dialect qualifier, 1–362
  examples, 1–381
  Execute qualifier, 1–362
  Fields qualifier, 1–362
  Limit_To qualifier, 1–363
  List_Plan=output-file qualifier, 1–363
  Log_Commits qualifier, 1–364
  Match_Name qualifier, 1–364
  null value, 1–369
  Parallel qualifier, 1–364
  Place qualifier, 1–366
  Record_Definition qualifier, 1–366
  Replace_Rows qualifier, 1–371
  Restricted_Access qualifier, 1–371
  Rms_Record_Def qualifier, 1–371
  Row_Count qualifier, 1–371
  Since qualifier, 1–372
  Skip qualifier, 1–373
  Statistics qualifier, 1–373
  syntax for .rrd file, A–1
Load command (cont'd)
  Transaction_Type qualifier, 1–374
  Trigger_Relations qualifier, 1–375
  .unl file
    and Oracle Rdb version, 1–376
  version of .unl file, 1–376
  Virtual_Fields qualifier, 1–375
  with character set, 1–379
Loading relations, 1–349, 1–404
Loading rows into an existing table, 1–349
Loading tables, 1–349, 1–404
  from security audit journal, 1–354
  generating a command procedure for, 1–301
Load Plan command
  Audit qualifier, 1–404
  Before qualifier, 1–406
  examples, 1–407
  Execute qualifier, 1–406
  List_Plan qualifier, 1–406
  Since qualifier, 1–407
Local buffers, 1–199, 1–547
Local symbol
  RMU$RDB_VERSION, 1–824
Lock mode compatibility, 1–768t
Lock qualifier combinations, 1–760t
Locks
  database, 1–759
  process, 1–759
Lock timeout for quiet point
  specifying the duration of, 1–87, 1–133
LOG command (RdbALTER), 2–47
Log file
  closing, 1–412
Logical name
  nonconcealed rooted, 1–436, 1–560
  RDM$BIND_ABS_QUIET_POINT, 1–648
  RDM$BIND_AIJ_WORK_FILE, 1–465, 1–491
  RDM$BIND_BUFFERS, 1–357
  RDM$BIND_DBR_WORK_FILE, 1–491
  RDM$BIND_SNAP_QUIET_POINT, 1–100
  RDMS$BIND_SORT_WORKFILES, 1–366, 1–464, 1–917
Logical names
  displaying, 1–777

LogMiner for Rdb
See the Unload After_Journal command
disabling, 1–695
enabling, 1–695
information returned, B–3
minimize application downtime, B–6
record definition prefix, B–5
restrictions and limitations, B–2
segmented string columns, B–6
table definition prefix, B–6
transaction table, B–5
using an OpenVMS pipe, B–8
LogMiner Mbx_Asynch
Using the Unload After_Journal command, 1–873

M

MAKE CONSISTENT command (RdbALTER), 2–48
Managing .aij files, 1–117
Modifying .aij files, 1–634
Modifying root file ACLs, 1–696
Monitor buffers available, 1–820
Monitor log specification, 1–820
Monitor process
closing log file, 1–412
refreshing log file, 1–412
starting, 1–414
stopping, 1–418
Monitor Reopen_Log command, 1–412
examples, 1–413
Monitor Start command, 1–414
examples, 1–417
Output qualifier, 1–414
Priority qualifier, 1–415
Swap qualifier, 1–415
Monitor Stop command, 1–418
Abort qualifier, 1–418
examples, 1–420
using with Oracle Trace, 1–419
Wait qualifier, 1–419
MOVE command (RdbALTER), 2–50
Move_Area command, 1–421
After_Journal qualifier, 1–422
Aij_Options qualifier, 1–423
All_Areas qualifier, 1–424
Area qualifier, 1–424
Attributes qualifier, 1–425
Blocks_Per_Page qualifier, 1–430
Cdd_Integrate qualifier, 1–425
Checksum_Verification qualifier, 1–426
Data_File qualifier, 1–431
Directory qualifier, 1–426
examples, 1–436
Extension qualifier, 1–432
File qualifier, 1–432
Log qualifier, 1–427
Nodes_Max qualifier, 1–427
Online qualifier, 1–427
Option qualifier, 1–427
Page_Buffers qualifier, 1–428
Path qualifier, 1–428
Quiet_Point qualifier, 1–428
Read_Only qualifier, 1–432
Read_Write qualifier, 1–432
Row_Cache_Options qualifier, 1–429
Snapshots qualifier, 1–433
Spams qualifier, 1–434
Thresholds qualifier, 1–434
Users_Max qualifier, 1–430
Moving a database, 1–421, 2–1
See also Move_Area command
Moving a root file, 1–421
Moving storage areas, 1–421
Multinational character set, A–6, A–8
Multischema database
extracting contents of, 1–299
extracting schema definitions, 1–303
Multithreaded backup, 1–66

N

NOLOG command (RdbALTER), 2–51
Nonconcealed rooted logical name
using to create database files, 1–436, 1–560

Null value
RMU Load command and, 1–369
RMU Unload After Journal command and, 1–869
RMU Unload command and, 1–839

O

.oaij file
See Optimized after-image journal (.oaij) file
Object
auditing, 1–660, 1–661
for specific privileges, 1–662
Object types
auditing, 1–660
for specific privileges, 1–662
Open command, 1–443
Access qualifier, 1–444
examples, 1–450
Global_Buffers qualifier, 1–446
Path qualifier, 1–447
Process_Global qualifier, 1–448
Row_Cache=Disable qualifier, 1–447
Statistics qualifier, 1–448
Wait qualifier, 1–449
Opening databases, 1–443
Opening root files, 1–102
OpenVMS security audit journal
loading into database, 1–354
Optimize After_Journal command, 1–452
Accept_Label qualifier, 1–455
Active_IO qualifier, 1–455
Block_Size qualifier, 1–456
Crc=Autodin_II qualifier, 1–456
Crc=Checksum qualifier, 1–456
Density qualifier, 1–457
Encrypt qualifier, 1–457
examples, 1–467
Format qualifier, 1–458
Group_Size qualifier, 1–459
Label qualifier, 1–459
Librarian qualifier, 1–460
Log qualifier, 1–461
Media_Loader qualifier, 1–461
Nocrc qualifier, 1–456
Optimize After_Journal command (cont'd)
Owner_Uic qualifier, 1–462
Protection qualifier, 1–462
Recovery_Method qualifier, 1–463
Rewind qualifier, 1–463
Tape_Expiration qualifier, 1–463
Trace qualifier, 1–464
Optimized after-image journal (.oaij) file
displaying output, 1–240
Optimizer statistics
collecting, 1–167
deleting, 1–216
displaying, 1–780
inserting, 1–341
Optimizing after-image journal (.aij) files, 1–452
logging, 1–461
to tape
See Tape, 1–455
tracing, 1–464
Optional root file parameter
RMU Alter command, 2–1
Oracle Media Management
Librarian qualifier, 1–83, 1–132, 1–245, 1–266, 1–460, 1–484, 1–544, 1–592
Oracle Rdb metadata
decoding, 1–295
extracting, 1–295
reading, 1–295
Oracle RMU command syntax, xii
Oracle Trace software
interaction with RMU Monitor Stop command, 1–419
Overwriting after-image journal (.aij) files, 1–634

P

Page buffers, 1–92, 1–201, 1–428, 1–552
PAGE command (RdbALTER), 2–52, 2–57e
Pages
setting to consistent, 1–681
setting to corrupt, 1–681
Parallel backup, 1–66

Parallel backup plan file, 1–151
Parallel load, 1–349, 1–364
Parameter qualifiers, 1–3
  defined, 1–3
  positional semantics of, 1–3
Parameters
  for Oracle RMU commands, 1–2
Patching databases, 2–1
Performance Monitor, 1–790
Plan file
  RMU Backup Plan command and, 1–151
  RMU Load Plan command and, 1–404
Populate_Cache
  examples, 1–470
Populate_Cache command
  Index qualifier, 1–468
  Log qualifier, 1–469
  Only_Cached qualifier, 1–469
  Statistics_Interval qualifier, 1–469
  Table qualifier, 1–469
  Transaction_Type qualifier, 1–469
Positional qualifiers
  See Parameter qualifiers
Privilege
  creating, 1–696
  deleting, 1–696
  modifying, 1–696
  required for using Oracle RMU commands, 1–5t
Procedure
  extracting, 1–302
Process locks, 1–759
Process statistics, 1–790
Profiles
  extracting, 1–302
Purging unused space
  from database pages, 1–513

Q

Qualifiers
  case sensitivity of, 1–2
  for Oracle RMU commands, 1–2
  positional, 1–3
  using indirect command files, 1–4
Query outline definition
  extracting, 1–302
Quiet-point-lock timeout
  specifying the duration of, 1–87, 1–133

R

RADIX command (RdbALTER), 2–53
RAID technology, 1–426
RDB$INDICES system table, 1–780
  displaying rows in, 1–167
RDB$RELATIONS system table, 1–780
  displaying rows in, 1–167
RDB$WORKLOAD system table, 1–780
  deleting rows from, 1–216
  displaying rows in, 1–167
  inserting rows into, 1–341
RdbALTER
  altering data fields, 2–9
  attaching to a database, 2–6
  changing area (.rda) and snapshot (.snp) file specifications, 2–15, 2–20
  changing recovery-unit journal (.ruj) file specification, 2–22
  changing recovery-unit journal (.ruj) unique identifier, 2–24
  changing root (.rdb) file specification, 2–22
  changing root (.rdb) unique identifier, 2–24
  commands, 2–2
    AREA . . . PAGE, 2–4
    ATTACH, 2–6
    COMMIT, 2–8
    DEPOSIT, 2–9
    DEPOSIT AREA_HEADER, 2–15
    DEPOSIT FILE, 2–20
    DEPOSIT ROOT, 2–22
    DEPOSIT ROOT UNIQUE_IDENTIFIER, 2–24
    DETACH, 2–27
    DISPLAY, 2–28
    DISPLAY AREA_HEADER, 2–35
    DISPLAY FILE, 2–39
    DISPLAY ROOT, 2–41
    DISPLAY ROOT UNIQUE_IDENTIFIER, 2–43

RdbALTER
commands (cont'd)
EXIT, 2–45
HELP, 2–46
LOG, 2–47
MAKE CONSISTENT, 2–48
MOVE, 2–50
NOLOG, 2–51
PAGE, 2–52
RADIX, 2–53
ROLLBACK, 2–54
UNCORRUPT, 2–55
VERIFY, 2–57
command syntax, 2–1
committing changes, 2–8
detaching from a database, 2–27
displaying area (.rda) or snapshot (.snp) file specifications, 2–39
displaying area (.rda) or snapshot (.snp) unique identifier, 2–35
displaying data fields, 2–28
displaying full root file spec, 2–35
displaying recovery-unit journal (.ruj) file specifications, 2–41
displaying recovery-unit journal (.ruj) unique identifier, 2–43
displaying root (.rdb) file specifications, 2–41
displaying root (.rdb) unique identifier, 2–43
ending the session, 2–45
exclusive update lock, 2–6, 2–27
fetching pages, 2–52
getting information on, 2–46
invoking, 1–12
keeping an audit trail, 2–47
moving data, 2–50
patching fields, 2–9
resetting an inconsistent flag, 2–48
resetting the corruption flag, 2–55
setting default radix, 2–53
specifying a page, 2–4
specifying a snapshot area, 2–4
specifying a storage area, 2–4
static verification, 2–57
stopping logging, 2–51
undoing changes, 2–54
RdbALTER (cont'd)
verifying a page, 2–57
RDM$BIND_ABS_QUIET_POINT logical name, 1–648
RDM$BIND_AIJ_INITIALIZE_IO_COUNT logical name, 1–145
RDM$BIND_AIJ_INITIALIZE_IO_SIZE logical name, 1–145
RDM$BIND_AIJ_WORK_FILE logical name, 1–465, 1–491
RDM$BIND_BUFFERS logical name, 1–357
RDM$BIND_DBR_WORK_FILE logical name, 1–491
RDM$BIND_SNAP_QUIET_POINT logical name, 1–100
RDMS$BIND_SORT_WORKFILES logical name
RMU Load command, 1–366
RMU Optimize After_Journal command, 1–464
RMU Verify command, 1–917
Reclaim command, 1–471
Area qualifier, 1–472
Log qualifier, 1–472
Page_Skip_Limit qualifier, 1–472
Retry_Time qualifier, 1–472
Spam_Skip_Limit qualifier, 1–472
Recover command, 1–476
Active_IO qualifier, 1–478
After_Journal qualifier, 1–478
Aij_Buffers qualifier, 1–479
Areas qualifier, 1–479
Automatic qualifier, 1–480
Confirm qualifier, 1–481
Directory qualifier, 1–481
Encrypt qualifier, 1–482
examples, 1–493
Format qualifier, 1–483
Just_Corrupt qualifier, 1–484
Just_Pages qualifier, 1–484
Label qualifier, 1–484
Librarian qualifier, 1–484
Log qualifier, 1–486
Media_Loader qualifier, 1–486
Online qualifier, 1–486
Option qualifier, 1–487

Recover command (cont'd)
Order_Aij_Files qualifier, 1–487
Output qualifier, 1–488
Prompt qualifier, 1–488
reentering lost transactions, 1–476
Resolve qualifier, 1–488
Rewind qualifier, 1–488
Root qualifier, 1–488
Trace qualifier, 1–489
Until qualifier, 1–490
Recover Resolve command, 1–507
Confirm qualifier, 1–508
examples, 1–509
State qualifier, 1–508
Recovery of .aij files
during a restore operation, 1–527
Recovery of databases, 1–476
single-file, 1–489
specifying new location for root file, 1–488
specifying time limits, 1–490
Recovery-unit journal (.ruj) file
displaying file specification, 2–41
displaying output, 1–288
Redundant arrays of inexpensive disks (RAID) technology, 1–426
Reopening the AIJ log server (ALS) output file, 1–621
Repair command, 1–510
Abm qualifier, 1–511
All_Segments qualifier, 1–511
Areas qualifier, 1–512
Checksum qualifier, 1–512
correcting AIP entries, 1–519
examples, 1–521
Initialize qualifier, 1–512
Spams qualifier, 1–519
Tables qualifier, 1–519
Repairing ABM pages, 1–511
Repairing database page corruption, 1–510
Repairing segmented string corruption, 1–510
Repairing SPAM page corruption, 1–519
Repairing storage areas, 1–512
Replicating a database, 1–193
Required privileges
See Privilege
Reserving after-image journal (.aij) files
using RMU Convert command, 1–185
using RMU Copy_Database command, 1–195
using RMU Move_Area command, 1–423
using RMU Restore command, 1–536
using RMU Restore Only_Root command, 1–588
using RMU Set After_Journal command, 1–634
Reserving storage area (.rda) files
using RMU Convert command, 1–185
Resolve command, 1–523
Confirm qualifier, 1–524
examples, 1–525
Log qualifier, 1–524
Parent_Node qualifier, 1–524
Process qualifier, 1–524
State qualifier, 1–524
Tsn qualifier, 1–525
Resolving an unresolved database transaction, 1–507, 1–523
Restore/Only_Root command
Attributes qualifier, 1–589
Encrypt qualifier, 1–590
Restore command, 1–527
Acl qualifier, 1–535
Active_IO qualifier, 1–535
After_Journal qualifier, 1–535
Aij_Options qualifier, 1–536
Area qualifier, 1–537
Blocks_Per_Page qualifier, 1–556, 1–557
by area, 1–534, 1–537
calling RMU Convert command, 1–527
Cdd_Integrate qualifier, 1–540
Close_Wait qualifier, 1–197, 1–540
Commit qualifier, 1–540
Confirm qualifier, 1–541
Directory qualifier, 1–541
Disk_File qualifier, 1–541
Duplicate qualifier, 1–542
Encrypt qualifier, 1–542
examples, 1–565
Extension qualifier, 1–557

Restore command (cont'd)
File qualifier, 1–557
generating an options file for
using the RMU Backup command, 1–95
using the RMU Dump Backup command, 1–272
using the RMU Dump command, 1–228
Global_Buffers qualifier, 1–542
Incremental qualifier, 1–543
Journal qualifier, 1–543
Just_Corrupt qualifier, 1–557
Just_Pages qualifier, 1–559
Label qualifier, 1–544
Librarian qualifier, 1–544
Loader_Synchronization qualifier, 1–547
Local_Buffers qualifier, 1–547
Log qualifier, 1–548
Master qualifier, 1–548
Media_Loader qualifier, 1–549
New_Version qualifier, 1–549
Nodes_Max qualifier, 1–549
Online qualifier, 1–550
Open_Mode qualifier, 1–550
Options qualifier, 1–550
Page_Buffers qualifier, 1–552
Path qualifier, 1–552
Progress_Report qualifier, 1–553
Prompt qualifier, 1–553
Read_Only qualifier, 1–559
Read_Write qualifier, 1–559
Recovery qualifier, 1–553
Rewind qualifier, 1–554
Root qualifier, 1–554
Row_Cache_Options qualifier, 1–554
Snapshot qualifier, 1–560
Spams qualifier, 1–561
Thresholds qualifier, 1–561
Transaction_Mode qualifier, 1–555
Users_Max qualifier, 1–556
Volumes qualifier, 1–556
Restore Only_Root command, 1–583
Active_IO qualifier, 1–587
After_Journal qualifier, 1–587
Aij_Options qualifier, 1–588
Blocks_Per_Page qualifier, 1–598
Restore Only_Root command (cont'd)
Directory qualifier, 1–590
examples, 1–604
File qualifier, 1–598
Initialize_Tsns qualifier, 1–590
Label qualifier, 1–591
Librarian qualifier, 1–592
Log qualifier, 1–593
Media_Loader qualifier, 1–593
New_Snapshots qualifier, 1–594
Nodes_Max qualifier, 1–594
Noset_Tsn qualifier, 1–595
Options qualifier, 1–594
Read_Only qualifier, 1–599
Read_Write qualifier, 1–599
Rewind qualifier, 1–594
Root qualifier, 1–595
Set_Tsn qualifier, 1–595
Snapshot qualifier, 1–599
Spams qualifier, 1–599
Thresholds qualifier, 1–600
Transaction_Mode qualifier, 1–596
Update_Files qualifier, 1–597
Users_Max qualifier, 1–597
Restore operation
See also Restore command, Restore Only_Root command
automatic .aij file recovery and, 1–527
by area, 1–533, 1–534, 1–537
full restore operation, 1–530
incremental restore operation, 1–530
automatic .aij recovery and, 1–563
Restore_Database command
Attributes qualifier, 1–539
Restoring a database, 1–527
See also Restore command, Restore Only_Root command
from a backup file, 1–527
from an incremental backup file, 1–530
from by-area backup files, 1–583
from tape
improving performance of, 1–535, 1–543
root file, 1–583

Resuming suspended AIJ backup operations, 1–627
RMU$FLAGS bits, 1–43t, 1–56t
RMU$RDB_VERSION local symbol, 1–824
RMU Alter command, 1–12
  See also RdbAlter
  correcting AIP entries, 1–519
  invoking the RdbALTER utility, 1–12
ROLLBACK command (RdbALTER), 2–54
Root file
  displaying contents, 1–221
  moving, 1–421
  problems opening, 1–102
  restoring, 1–583
Root file ACL
  displaying, 1–787
  modifying, 1–696
.rrd file
  data types, A–6
  DEFINE FIELD statement, A–1
  DEFINE RECORD statement, A–2
  syntax of, A–1, A–10
.ruj file
  See Recovery-unit journal (.ruj) file

S

Security alarms, 1–658
  disabling, 1–658
  enabling, 1–659
Security auditing
  disabling, 1–658
  displaying characteristics, 1–752
  enabling, 1–658, 1–659
  starting, 1–665
  stopping, 1–665
Security audit journal, 1–658
  loading into database, 1–354
Security audit journal records
  creating a database for storing, 1–377
  creating a table for storing, 1–377, 1–378
Segmented strings
  repairing corruption to, 1–510
  verifying, 1–913
Sequential access
  improving performance of, 1–513
Server After_Journal Reopen_Output command, 1–621
  examples, 1–622
  reopening the AIJ log server (ALS) output file, 1–621
Server After_Journal Start command, 1–623
  examples, 1–624
  Output qualifier, 1–623
  starting the AIJ log server (ALS), 1–623
Server After_Journal Stop command, 1–625
  examples, 1–626
  Output qualifier, 1–625
  stopping the AIJ log server (ALS), 1–625
Server Backup_Journal Resume command, 1–627
  examples, 1–628
  Log qualifier, 1–627
  resuming suspended AIJ backup operations, 1–627
Server Backup_Journal Suspend command, 1–629
  examples, 1–630
  Log qualifier, 1–630
  suspending AIJ backup operations, 1–629
Server Record_Cache Checkpoint command, 1–632
  Log qualifier, 1–632
  Wait qualifier, 1–633
Set After_Journal command, 1–634
  Add qualifier, 1–635
  AIJ_Options qualifier, 1–637
  Allocation qualifier, 1–637
  Alter qualifier, 1–638
  Backups qualifier, 1–639
  Cache qualifier, 1–642
  changing .aij file specification, 1–634
  Disable qualifier, 1–643
  Drop qualifier, 1–643
  Enable qualifier, 1–643
  examples, 1–649
  Extent qualifier, 1–643
  Log qualifier, 1–644
  Notify qualifier, 1–644

Set After_Journal command (cont'd)
Overwrite qualifier, 1–645
Reserve qualifier, 1–646
Shutdown_Timeout qualifier, 1–646
Suppress qualifier, 1–647
Switch_Journal qualifier, 1–647
Set AIP command, 1–653
examples, 1–656
Larea qualifier, 1–654
Length qualifier, 1–654
Log qualifier, 1–654
Rebuild_Spams qualifier, 1–654
Rename_To qualifier, 1–654
Threshold qualifier, 1–654
Set Audit command, 1–658
Disable qualifier, 1–658
Enable qualifier, 1–659
Every qualifier, 1–664
examples, 1–668
extracting, 1–303
First qualifier, 1–665
Flush qualifier, 1–665
Start qualifier, 1–665
Stop qualifier, 1–665
Type=Alarm option, 1–666
Type=Audit option, 1–666
Type qualifier, 1–666
Set Buffer_Object command, 1–678
example, 1–680
Set Corrupt_Pages command, 1–681
Area qualifier, 1–683
Consistent qualifier, 1–684
Corrupt qualifier, 1–684
Disk qualifier, 1–684
examples, 1–685
Page qualifier, 1–684
Set Database command, 1–687
Transaction_Mode qualifier, 1–687
Set Galaxy command, 1–690
example, 1–691
Set Global_Buffers command, 1–692
Disabled qualifier, 1–692
Enabled qualifier, 1–692
Large_Memory qualifier, 1–693
Log qualifier, 1–693
Set Global_Buffers command (cont'd)
Number qualifier, 1–693
User_Limit qualifier, 1–693
Set Logminer command, 1–694
examples, 1–695
Set Privilege command, 1–696
Acl qualifier, 1–698
Acl_File qualifier, 1–699
After qualifier, 1–699
Delete qualifier, 1–699
Edit qualifier, 1–699
examples, 1–704
Header qualifier, 1–787
Journal qualifier, 1–700
Keep qualifier, 1–700
Like qualifier, 1–700
Log qualifier, 1–700
Mode qualifier, 1–700
New qualifier, 1–701
Recover qualifier, 1–701
Replace qualifier, 1–701
Set Row_Cache command, 1–709
Alter=Rad_Hint qualifier, 1–710
Alter=Type option, 1–710
Alter qualifier, 1–709
Backing_Store_Location qualifier, 1–711
Disable qualifier, 1–711
Enable qualifier, 1–712
examples, 1–292, 1–713
NoBacking_Store_Location qualifier, 1–712
Set Server command, 1–715
Set Server Output command
examples, 1–717
Log qualifier, 1–716
Set Shared_Memory command, 1–719
Log qualifier, 1–719
Rad_Hint qualifier, 1–719
Section_Name qualifier, 1–720
Type=Process option, 1–721
Type=Resident option, 1–721
Type=System option, 1–721
Type qualifier, 1–721
Set Statistics command, 1–724
Checkpoint qualifier, 1–725
Export qualifier, 1–726

Set Statistics command (cont'd)
Import qualifier, 1–728
Log qualifier, 1–729
Setting CSN values, 1–584, 1–588, 1–589, 1–595, 1–602
Setting TSN values, 1–584, 1–588, 1–589, 1–595, 1–602
Show After_Journal command, 1–733
Backup_Context qualifier, 1–738
Checkpoint qualifier, 1–740
examples, 1–741
Output qualifier, 1–741
Show AIP command, 1–747
Brief qualifier, 1–748
examples, 1–749
Larea qualifier, 1–748
Option qualifier, 1–748
Output qualifier, 1–748
Parea qualifier, 1–748
Type qualifier, 1–748
Show Audit command, 1–752
All qualifier, 1–752
Daccess qualifier, 1–753
Every qualifier, 1–753
examples, 1–754
Flush qualifier, 1–753
Identifiers qualifier, 1–753
Output qualifier, 1–753
Protection qualifier, 1–753
Rmu qualifier, 1–753
Type=Alarm option, 1–753
Type=Audit option, 1–753
Show command, 1–732
See also specific Show commands
Show Corrupt_Pages command, 1–757
examples, 1–758
Options qualifier, 1–757
Output qualifier, 1–757
Show Locks command, 1–759
Lock qualifier, 1–762
Mode qualifier, 1–763
Options qualifier, 1–764
Output qualifier, 1–764
Process qualifier, 1–764
Resource_type qualifier, 1–765
Show Logical_Names command, 1–777
Description qualifier, 1–777
Output qualifier, 1–778
Undefined qualifier, 1–778
Show Optimizer_Statistics command, 1–780
examples, 1–784
Full qualifier, 1–780
Index qualifier, 1–781
Log qualifier, 1–781
Statistics qualifier, 1–781
System_Relations qualifier, 1–782
Tables qualifier, 1–782
Threshold qualifier, 1–783
Show Privilege command, 1–787
examples, 1–788
Expand_All qualifier, 1–787
Show Statistics command, 1–790
Alarm qualifier, 1–794
Broadcast qualifier, 1–794
Cluster qualifier, 1–795
Configure qualifier, 1–796
Cycle qualifier, 1–796
Dbkey_Log qualifier, 1–796
Deadlock_Log qualifier, 1–797
exiting display, 1–812
Flush_Interval qualifier, 1–798
Histogram qualifier, 1–800
Input qualifier, 1–800
Interactive qualifier, 1–800
Lock_Timeout_Log qualifier, 1–801
Logical_Area qualifier, 1–802
Log qualifier, 1–802
Notify qualifier, 1–803
Opcom_Log qualifier, 1–805
Options qualifier, 1–806
Output qualifier, 1–808
Prompt_Timeout qualifier, 1–808
Reopen_Interval qualifier, 1–809
Reset qualifier, 1–809
Screen qualifier, 1–810
Stall_Log qualifier, 1–811
Time qualifier, 1–811
Until qualifier, 1–811
Write_Report_Delay qualifier, 1–812

Show System command, 1–818
  available monitor message buffers, 1–818
  examples, 1–819
  Output qualifier, 1–818
Show Users command, 1–820
  available monitor message buffers, 1–820
  examples, 1–821
  Output qualifier, 1–820
Show Version command, 1–823
  examples, 1–824
  Output qualifier, 1–823
Shutdown timeout
  displaying
    using RMU Show After_Journal command, 1–735
  setting
    using RMU Set After_Journal command, 1–634, 1–646
Single-file database
  recovery of, 1–489
Snapshot (.snp) file
  displaying contents, 1–221
  moving, 1–560
Snapshot area
  specifying with RdbALTER utility, 2–4
Snapshot pages
  checksum verification of, 1–914
Space area management (SPAM) page
  See SPAM page
SPAM page
  dumping, 1–229
  output from RMU Verify command, 1–903
  repairing, 1–519
SQL ALTER SEQUENCE statement
  extracting, 1–303
SQL CREATE SEQUENCE statement
  extracting, 1–303
SQL IMPORT script
  extracting, 1–300
Starting security auditing, 1–665
Starting the AIJ log server (ALS), 1–623
Starting the monitor, 1–414
Statistics
  collecting for the optimizer, 1–167
  database, 1–790
Statistics (cont'd)
  displaying those used by the optimizer, 1–780
  gathering
    on database page space, 1–14
    on index structure, 1–35
    on logical area space, 1–14
    on row placement relative to index structures, 1–50
    on storage area space, 1–14
  process, 1–790
Stopping security auditing, 1–665
Stopping the AIJ log server (ALS), 1–625
Stopping the monitor process, 1–418
Storage allocation
  improving, 1–519
Storage area
  displaying contents, 1–221
  moving, 1–421
  repairing, 1–512
  reserving, 1–185
  specifying with RdbALTER utility, 2–4
Storage map definition
  extracting, 1–303
Storage statistics
  collecting, 1–167
Stored function
  extracting, 1–302
Stored procedure
  extracting, 1–302
Suppressing use of after-image journal (.aij) file, 1–634
Suspending AIJ backup operations, 1–629
Symbols
  process, 1–120, 1–738
Syntax diagram, reading, xiii

T

Table
  loading data into, 1–349
  unloading from the database, 1–827
Table definition
  extracting, 1–304

Tape
backing up .aij files to, 1–120, 1–121
backing up database to, 1–103
backing up to
accepting current tape labels and, 1–71, 1–122
database backup file name length and, 1–69
identifying volumes
for dump of database backup file, 1–270
improving performance of
dump of database backup file, 1–265
improving reliability of
database backup file on, 1–74
for .aij backup file on, 1–126
for optimized .aij file on, 1–456
maximum number of read operations
for dump of .aij backup, 1–241
for dump of database backup, 1–263
for recovery operation, 1–478
maximum number of write operations to
for database backup, 1–72
optimizing aij file to
accepting current tape labels and, 1–455
parallel backup to, 1–66, 1–92
preloading
for a database backup operation, 1–86
for a database restore operation, 1–547
protection
on .aij backup file, 1–135
on database backup file, 1–94
on optimized .aij file, 1–462
recovering .aij file from
accepting current tape labels and, 1–486
rewinding
before .aij backup operation begins, 1–137
before database backup operation begins, 1–96
before dump of database backup file begins, 1–272
before optimization of .aij backup file begins, 1–463
before recovery operation begins, 1–488
before restore-only-root operation begins, 1–594
Tape
rewinding (cont'd)
before restore operation begins, 1–554
rewinding before dump of .aij backup file begins, 1–249
specifying concurrent access
for a restore operation, 1–556
specifying density
for .aij backup file on, 1–126
for database backup file on, 1–75
for optimized .aij file on, 1–457
specifying expiration date
for .aij backup file, 1–139
for database backup file, 1–97
for optimized .aij file, 1–463
specifying format
for .aij backup file, 1–129
for .aij backups, 1–142, 1–144
for optimized .aij file, 1–458
for recovery operation, 1–483
specifying format for dump of .aij backup file, 1–243
specifying labels
for .aij backup file, 1–131
for database backup file, 1–82
for dump of .aij backup file, 1–244
for dump of database backup file, 1–266
for optimized .aij file, 1–459
for recovery operation, 1–484
for restore-only-root operation, 1–591
for restore operation, 1–544
specifying maximum number of blocks
for database backup file, 1–72
for optimized .aij file, 1–455, 1–456
specifying owner of
.aij backup file, 1–134
database backup file, 1–92
optimized .aij file, 1–462
XOR recovery blocks
for .aij backup file, 1–130
for database backup file, 1–79
for optimized .aij file, 1–459
Tape device
alerting RMU to presence or absence of
for .aij backup file, 1–134, 1–269

Tape device
alerting RMU to presence or absence of (cont'd)
for database backups, 1–88
for dump of .aij backups, 1–246
for optimized .aij file, 1–461
for restore-only-root operation, 1–593
for restore operation, 1–549
Tape drive
allocating
for .aij backup file, 1–144
for database backup file, 1–99
designating a master
for database backup operation, 1–88
for database restore operation, 1–548
Tape labels
accepting current labels, 1–71, 1–122, 1–455
Temporary work file
AIJ
location of, 1–465, 1–491
Transaction
See Unresolved transaction, Distributed transaction
Transaction Sequence Number (TSN)
See TSN values
Trigger
extracting, 1–305
Truncation
of file name during backup operation, 1–103
TSN values, 1–518, 1–525, 1–590, 1–595
initializing, 1–590
setting, 1–584, 1–588, 1–589, 1–595, 1–602

U

UNCORRUPT command (RdbALTER), 2–55
.unl file, 1–351, 1–827
Unload After Journal command
delimited text null string, 1–869
delimited text prefix string, 1–869
delimited text separator string, 1–869
delimited text suffix string, 1–869
null value, 1–869
Unload After_Journal command, 1–861
Before qualifier, 1–864
Continuous qualifier, 1–864
Deletes_First qualifier, 1–866
enable for LogMiner extraction, 1–862
examples, 1–889
Extend_Size qualifier, 1–866
Format qualifier, 1–866
Ignore qualifier, 1–870
Include=Action qualifier, 1–871
IO_Buffers qualifier, 1–873
Log qualifier, 1–873
Mbx_Asynch qualifier, 1–873
Options qualifier, 1–874
Order_AIJ_Files qualifier, 1–875
Output qualifier, 1–875
Parameter qualifier, 1–875
Quick_Sort_Limit qualifier, 1–876
Restart qualifier, 1–876
Restore_Metadata qualifier, 1–876
Save_Metadata qualifier, 1–877
Select qualifier, 1–877
Since qualifier, 1–878
Sort_Workfiles qualifier, 1–878
Statistics_Interval qualifier, 1–878
Symbols qualifier, 1–879
Table qualifier, 1–879
Trace qualifier, 1–880
Unload command, 1–827
Allocation qualifier, 1–828
Buffers qualifier, 1–829
case sensitivity and, 1–844
Commit_Every qualifier, 1–829
Compression qualifier, 1–829
Debug_Options qualifier, 1–830
Delete_Rows qualifier, 1–832
delimited identifier and, 1–844
delimited text null string, 1–839
delimited text prefix string, 1–838
delimited text separator string, 1–839
delimited text suffix string, 1–839
Error_Delete qualifier, 1–832
examples, 1–846
Extend_Quantity qualifier, 1–833
Fields qualifier, 1–833

Unload command (cont'd)
Flush qualifier, 1–833
Limit_To qualifier, 1–834
null value, 1–839
Optimize qualifier, 1–834
Record_Definition qualifier, 1–836
Reopen_Count qualifier, 1–840
Rms_Record_Def qualifier, 1–841
Row_Count qualifier, 1–841
Statistics_Interval qualifier, 1–841
syntax for .rrd file, A–10
Transaction_Type qualifier, 1–323, 1–841
viewing contents of .unl file, 1–281
Virtual_Fields qualifier, 1–843
with character set, 1–844
Unloading tables, 1–827
Unloading views, 1–827
Unresolved transaction
aborting in the .aij file, 1–507
committing in the .aij file, 1–507
displaying a list of, 1–229
resolving, 1–507, 1–523
Upgrading software version
with RMU Convert command, 1–182
Users
displaying information about, 1–820

V

Verb
locks, 1–759
statistics, 1–790
Verify command, 1–902
All qualifier, 1–903
Areas qualifier, 1–903
Checksum_Only qualifier, 1–904
Constraints qualifier, 1–904
Data qualifier, 1–906
End qualifier, 1–906
Error_Limit qualifier, 1–906
examples, 1–920
For_Table qualifier, 1–907
Functions qualifier, 1–908
Incremental qualifier, 1–908
Indexes qualifier, 1–908
Key_Values qualifier, 1–909
Verify command (cont'd)
Lareas qualifier, 1–910
Log qualifier, 1–910
Orphan_Indexes qualifier, 1–911
Output qualifier, 1–912
Root qualifier, 1–912
Routines qualifier, 1–913
Segmented_Strings qualifier, 1–913
Snapshots qualifier, 1–914
Start qualifier, 1–914
Transaction_Type qualifier, 1–915
VERIFY command (RdbALTER), 2–57
Verify command file generated by RMU Extract command
parameters for, 1–306t
Verifying database integrity, 1–902
generating a command procedure for, 1–305
Version number
displaying information about, 1–823
View
extracting definition of, 1–307
unloading from the database, 1–827

W

Work load
extract, 1–309
Workload statistics
collecting, 1–167
deleting, 1–216
displaying, 1–780
inserting, 1–341

X

XOR recovery blocks
specifying frequency of
for .aij backup file, 1–130
for database backup file, 1–79
for optimized .aij file, 1–459