Oracle® Rdb Connectivity Manager RMU Statistics User Guide

Release 7.3.2.0.0
August 2017

Oracle Rdb Connectivity Manager RMU Statistics User Guide, Release 7.3.2.0.0

Primary Author: Jim Murray.
Contributing Author: Rick Anderson.

Contents

Preface

Purpose of This Manual

The Oracle Rdb Connectivity Manager 7.3.1 RMU Statistics User Guide describes concepts, features and usage of the Oracle Rdb Connectivity Manager RMU Statistic tool.

Intended Audience

This document is intended for users responsible for:

- System management
- Database administration
- Application programming

Document Structure

This document consists of the following chapters:

Chapter 1	Overview of ORCM RMU Statistics user interface
Chapter 2	Describes the RMU Statistics menus
Chapter 3	Lists the RMU Statistics panels
Chapter 4 onwards	Describes the various Statistics panels available

Conventions

Oracle Rdb Connectivity Manager is often referred to as ORCM.

Oracle Rdb Connectivity Manager RMU Statistics tool often referred to as RMU Statistics.

Oracle Rdb is often referred to as Rdb.

Hewlett-Packard Company is often referred to as HP.

The following conventions are used in this document:

word	A lowercase word in a format example indicates a syntax element that you supply.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
{ }	Braces enclose clauses from which you must choose one alternative.
...	A horizontal ellipsis means you can repeat the previous item.
. . .	A vertical ellipsis in an example means that information not directly related to the example has been omitted.

Conventions in Code Examples

Code examples illustrate SQL or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT last_name FROM employees WHERE last_name = 'TOLIVER';
```

Contents

Chapter 1

Overview of ORCM RMU Statistics user interface

Oracle Rdb collects and maintains an immense amount of runtime statistical and analytical information about database activities as they occur over time. From OpenVMS DCL command line the **RMU Show Statistic** utility may be used to present and analyze this statistical information.

Alternatively the ORCM RMU Statistics tool may be used, which may be invoked by selecting the ***RMU Statistics*** menu item from the **RMU** submenu of the ORCM main menu.

The following sections describe the invocation, layout and operation of the ORCM **RMU Statistics** user interface:

- Invoking RMU Statistics
- ORCM RMU Statistics
- Layout

1.1 Invoking RMU Statistics

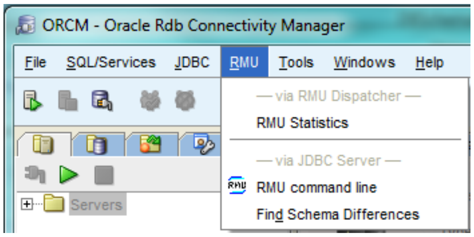
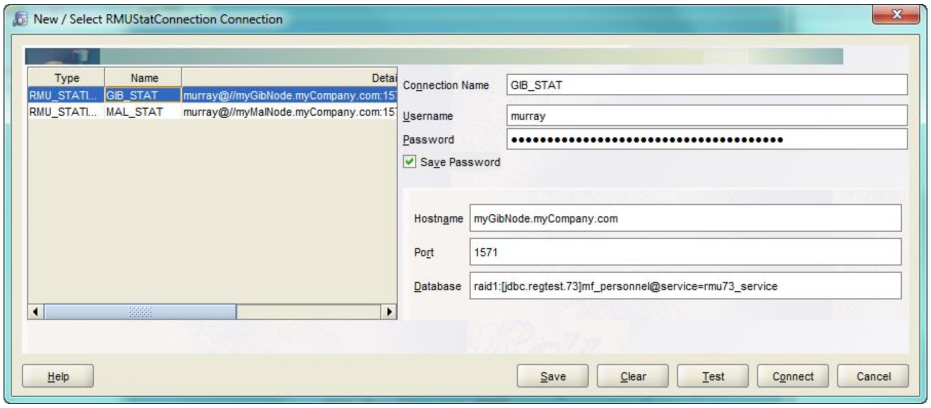


Figure 1 - Invoking RMU Statistics

Select the **RMU Statistics** option from the **RMU** sub-menu within the ORCM main menu.

When the ORCM **RMU Statistics** menu option is selected a connection dialog will be shown allowing you to enter connection information to connect to the RMU server you wish to use to access the monitored database.



Type	Name	Detai
RMU_STATI...	GIB_STAT	murray@//myGibNode.myCompany.com:15
RMU_STATI...	MAL_STAT	murray@//myMalNode.myCompany.com:15

Connection Name: GIB_STAT
Username: murray
Password: ••••••••••••••••••••••••••••••••••••••
☑ Save Password
Hostname: myGibNode.myCompany.com
Port: 1571
Database: raid1:[jdbc.regtest.73]mf_personnel@service=rmu73_service

Figure 2 - RMU Statistics Connection

ORCM **RMU Statistics** uses the standard RMU server to obtain the database statistics. Thus before you can connect you should ensure that both the SQL/Services RMU Dispatcher and RMU services are running on the node you wish to access.

The connection information is similar to JDBC control connection information; however the server that will be connected to is a RMU server, which by default listens on port 1571.

Note:

✓ The RMU Server would normally be started up by SQL/Services as the RMU Dispatcher.

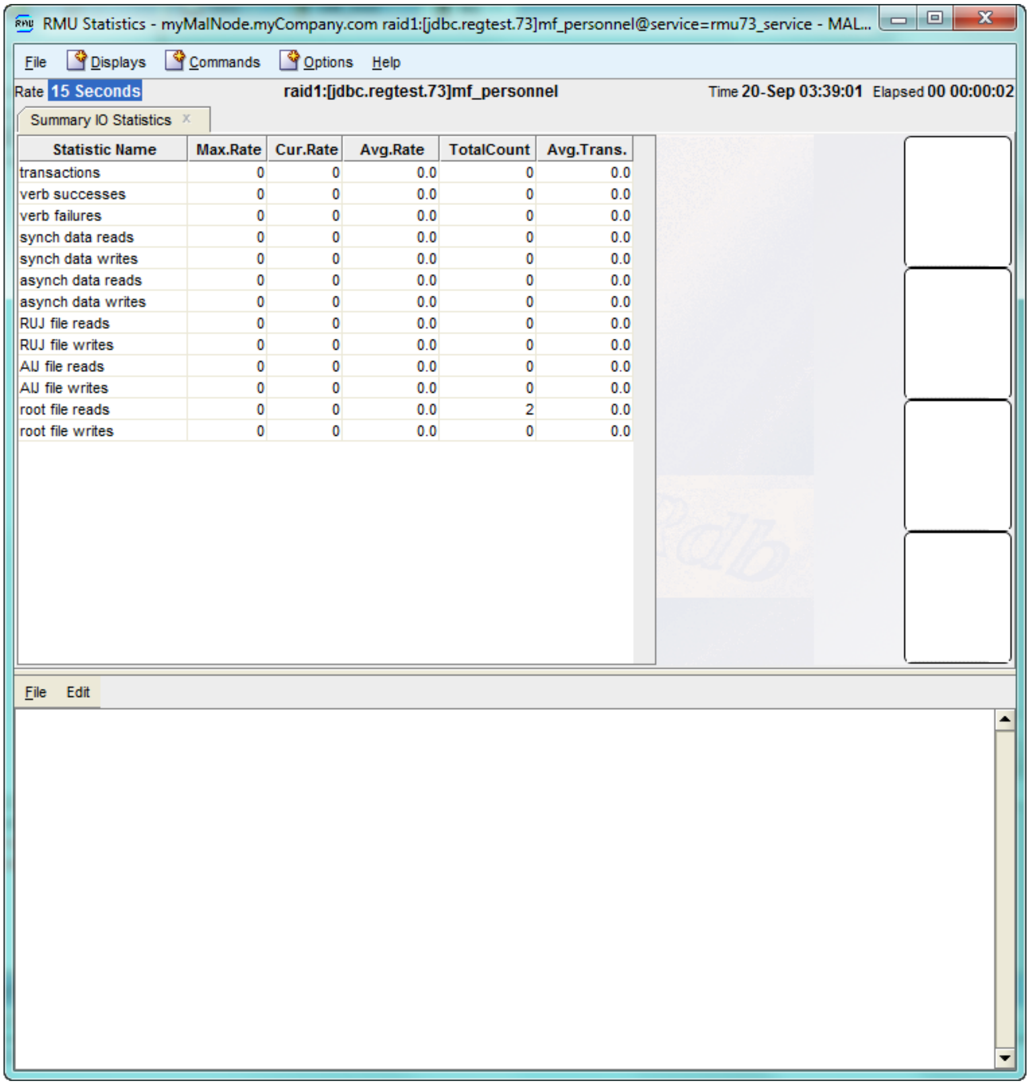
You may choose a different RMU Service to connect to. The default service is called RMU_SERVICE, however if you have multiple versions of Rdb installed on your system the appropriate RMU service may be called something else. Your database or SQL/Services administrator will be able to tell you the correct service name to use.

If you choose to use a RMU service name other than RMU_SERVICE, you can tell ORCM the name of the service by appending "`@service=<service_name>`" to the database connection string as shown in the example above.

The username and password must be correct for the OpenVMS system you will be connecting to and valid for access to the RMU Server and the specified database.

Once correct connection information has been entered, press the **Connect** key to connect. If successful the ORCM RMU Statistics window will appear.

1.2 ORCM RMU Statistics


RMU Statistics - myMalNode.myCompany.com raid1:[jdbc.regtest.73]mf_personnel@service=rmu73_service - MAL...
File
Displays
Commands
Options
Help
Summary IO Statistics
File
Edit

Rate 15 Seconds

raid1:[jdbc.regtest.73]mf_personnel

Time 20-Sep 03:39:01 Elapsed 00 00:00:02

Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
synch data reads	0	0	0.0	0	0.0
synch data writes	0	0	0.0	0	0.0
asynch data reads	0	0	0.0	0	0.0
asynch data writes	0	0	0.0	0	0.0
RUJ file reads	0	0	0.0	0	0.0
RUJ file writes	0	0	0.0	0	0.0
AIJ file reads	0	0	0.0	0	0.0
AIJ file writes	0	0	0.0	0	0.0
root file reads	0	0	0.0	2	0.0
root file writes	0	0	0.0	0	0.0

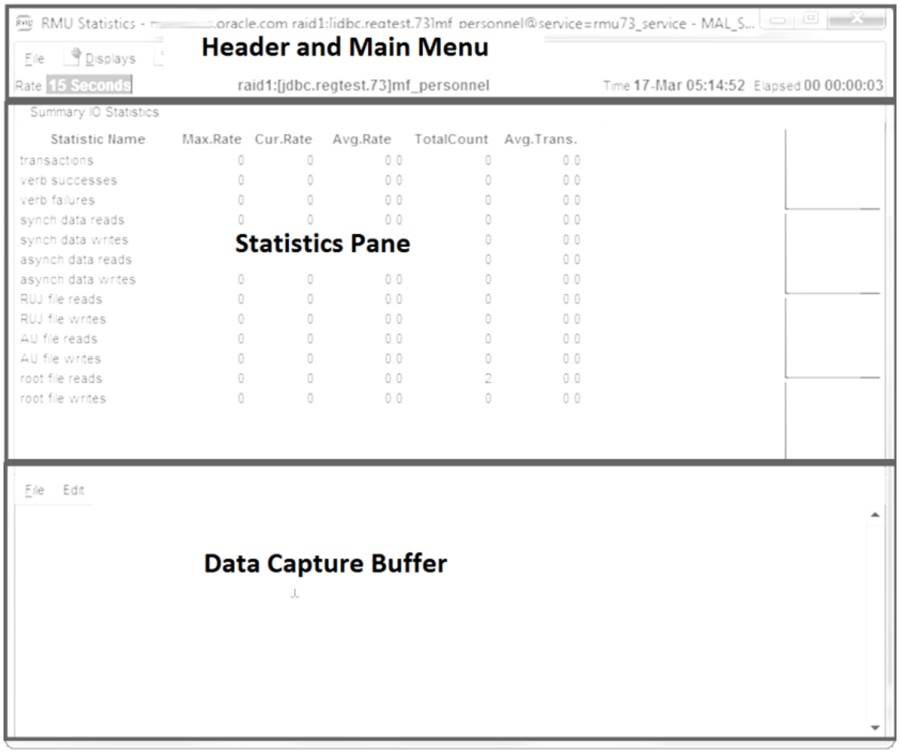
Figure 3 - ORCM RMU Statistics

ORCM RMU Statistics provides a GUI-based interface for presentation of RMU statistical data.

Note:

- ✓ Currently the ORCM RMU Statistics interface provides a subset of the complete set of features found in the **RMU Show Statistic** utility; much of the presentation and reporting features are available, however the analytical and replay features are not.

1.3 Layout



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
synch data reads	0	0	0.0	0	0.0
synch data writes				0	0.0
asynch data reads				0	0.0
asynch data writes	0	0	0.0	0	0.0
RUJ file reads	0	0	0.0	0	0.0
RUJ file writes	0	0	0.0	0	0.0
AU file reads	0	0	0.0	0	0.0
AU file writes	0	0	0.0	0	0.0
root file reads	0	0	0.0	2	0.0
root file writes	0	0	0.0	0	0.0

Figure 1 - layout

The interface is comprised of three main display areas:

- The header and main menu area
- The Statistics pane
- The Data Capture buffer

1.3.1 The header and main menu area

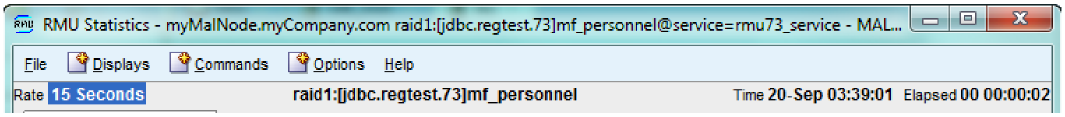

RMU Statistics - myMalNode.myCompany.com raid1:[jdbc.regtest.73]mf_personnel@service=rmu73_service - MAL...
File
Displays
Commands
Options
Help
Rate 15 Seconds
raid1:[jdbc.regtest.73]mf_personnel
Time 20-Sep 03:39:01
Elapsed 00 00:00:02

Figure 4 - ORCM RMU Statistics header area

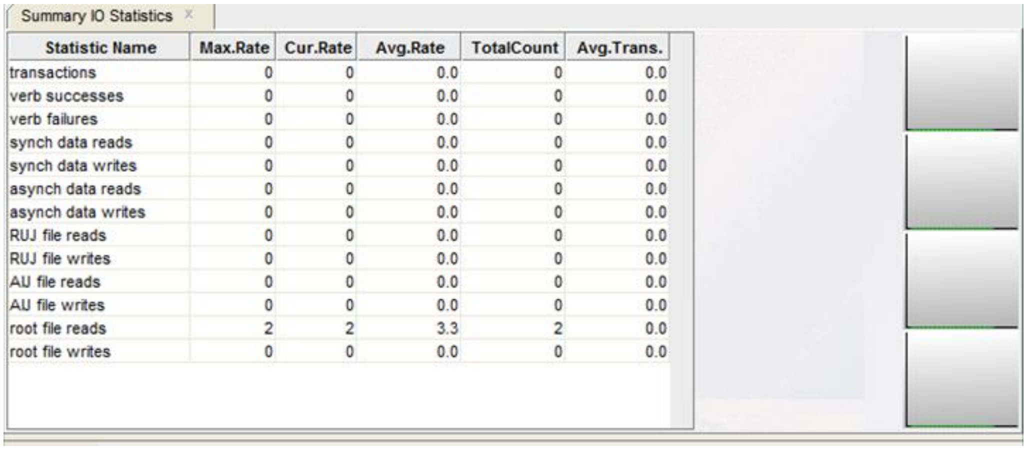
The frame header displays the connection string used for invocation. Under this is the main RMU Statistics menu followed by the statistics header.

The main menu is where the various displays and options for ORCM RMU Statistics may be selected. See RMU Statistics Menus for more details on the Main Menu.

The statistics header shows:

- The refresh rate used.   See Setting refresh rate for more details.
- The file specification of the database the statistics pertain to.
- The time stamp of the last refresh of the statistical data.
- The elapsed time since this statistics session began.

1.3.2 The Statistics pane



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
synch data reads	0	0	0.0	0	0.0
synch data writes	0	0	0.0	0	0.0
asynch data reads	0	0	0.0	0	0.0
asynch data writes	0	0	0.0	0	0.0
RUJ file reads	0	0	0.0	0	0.0
RUJ file writes	0	0	0.0	0	0.0
AIJ file reads	0	0	0.0	0	0.0
AIJ file writes	0	0	0.0	0	0.0
root file reads	2	2	3.3	2	0.0
root file writes	0	0	0.0	0	0.0

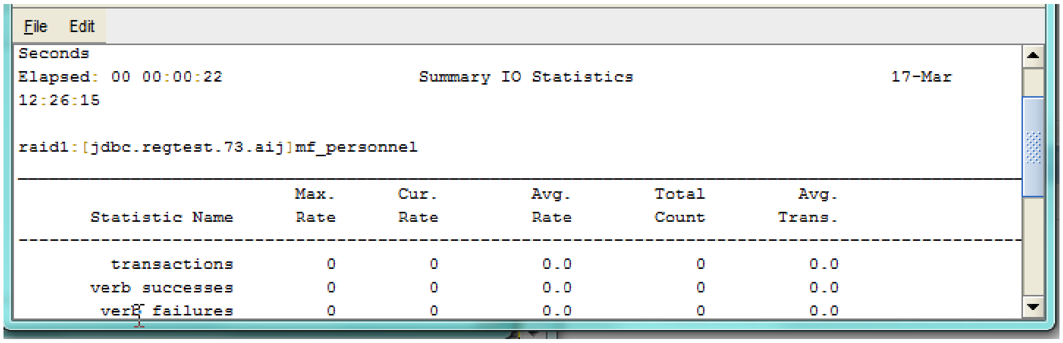
Figure 5 - ORCM RMU Statistics pane

The Statistics pane is where the data will be displayed. Each selected statistic will be displayed as a separate tab within this area.

Statistics data is generally displayed in table format, the column headers and contents depending on the specific statistic being displayed. Details about the data displayed can be found in the specific panel descriptions. See DisplaysMenu for more details.

To the right of the data area is a set of four small graph panels, the **Statistics Mini Graphs**. Each panel displays a timeline of a selected statistic count from the current data.

1.3.3 The Data Capture buffer



```
Seconds
Elapsed: 00 00:00:22                    Summary IO Statistics                       17-Mar
12:26:15

raid1:[jdbc.regtest.73.aij]mf_personnel

-----------------------------------------------------------------------------------------------------
                           Max.      Cur.         Avg.        Total         Avg.
      Statistic Name       Rate      Rate         Rate        Count        Trans.
-----------------------------------------------------------------------------------------------------

        transactions          0         0          0.0            0           0.0
      verb successes          0         0          0.0            0           0.0
       verb failures          0         0          0.0            0           0.0
```

Figure 6 - Data Capture buffer

See [Data Capture Buffer](#) for more details.

1.3.4 Setting refresh rate

By default the statistical data is requested from the connected RMU server every 15 seconds. This may be changed by selecting the **Timer Rate** menu option from the **Options** menu:

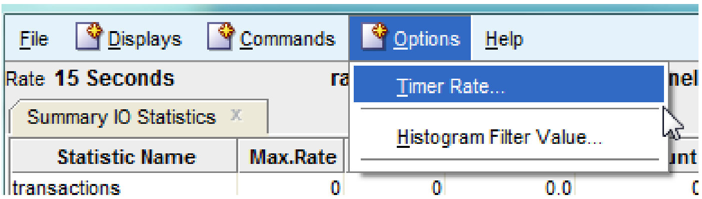
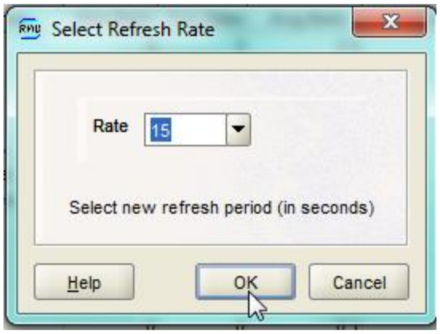


Figure 7 - Timer Rate

When selected a dialog will be displayed allowing you to change the frequency of statistic data refreshes.



Rate 15

Select new refresh period (in seconds)

Figure 8 - Timer Rate Selection

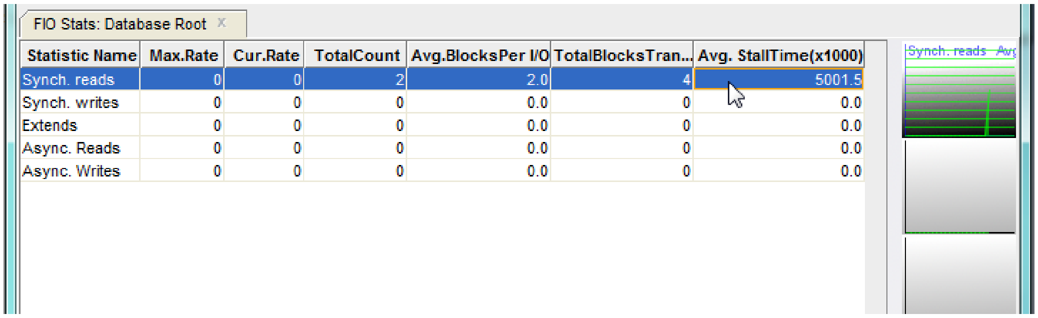
Oracle advises not to set this value to less than 5 seconds. Depending on your network latency setting this value to a smaller period may cause excessive network traffic and may cause timeouts on the ORCM GUI application.

1.4 Statistics Mini Graphs

Depending on the statistics data displayed the Statistic pane may contain a set of four graphical panels. If the panels are present, they will be displayed to the right of the main data table area. These mini graphs may be used to watch the change of selected statistics over time.

By default no data is displayed in the mini graphs.

To select which data should be watched, double-click within any numeric cell within the statistics data table. The selected statistic will then be graphed in the next free mini graph panel. The panels are allocated in a round-robin fashion, each double-click will assign the selected statistic to the next mini graph panel available, from top to bottom and then it will cycle around again to the top of the panel stack.

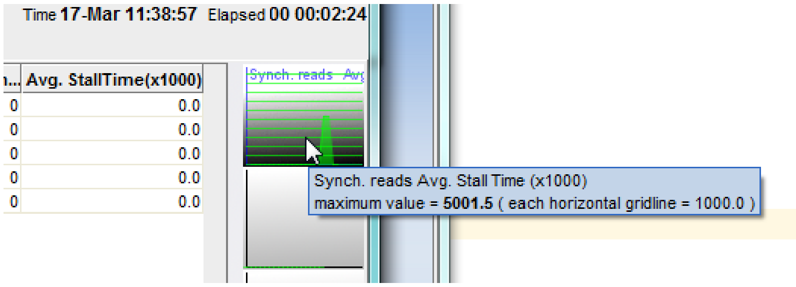


Statistic Name	Max.Rate	Cur.Rate	TotalCount	Avg.BlocksPer I/O	TotalBlocksTran...	Avg. StallTime(x1000)
Synch. reads	0	0	2	2.0	4	5001.5
Synch. writes	0	0	0	0.0	0	0.0
Extends	0	0	0	0.0	0	0.0
Async. Reads	0	0	0	0.0	0	0.0
Async. Writes	0	0	0	0.0	0	0.0

Figure 9 - Double-click to assign mini graph

In the example shown above, double-clicking in the **Avg.StallTime(x1000)** column of the **Synch. reads** table row displays the time variant data for the average stall time for synchronized reads.

Hover the mouse over the mini graph to see a synopsis of the display:



Time **17-Mar 11:38:57** Elapsed **00 00:02:24**

...	Avg. StallTime(x1000)
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0

Synch. reads Avg. Stall Time (x1000)
maximum value = **5001.5** (each horizontal gridline = 1000.0)

Figure 10 - Mini graph synopsis

The mini graphs are updated each time the statistics are refreshed. Each graph will display around 5 minutes worth of recent data.

1.5 Reset and Pause

The ORCM RMU Statistics window main menu has a ***Commands*** submenu that allows you to reset, pause and resume the presentation of statistic information:

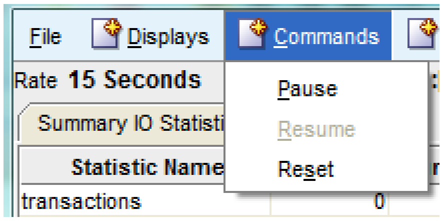


Figure 11 - Reset menu option

Selecting the ***Reset*** menu option does not reset the database statistic information; rather, it resets the per-process view of the statistic information. The original database statistic information can always be retrieved.

Often, it is desirable to temporarily pause the collection of statistic information so that a snapshot view of the statistic information can be analyzed. Since Oracle Rdb is a high-performance mission-critical database management system, there is no method available to temporarily pause the collection of statistic information.

However, you can temporarily pause the presentation of the statistic information when using the ORCM RMU Statistics utility. The ***Pause*** menu option provides this capability on most statistical presentations.

The ***Resume*** menu option will resume a paused session.

1.6 Data Capture

Statistical Data is captured by selecting one of the capture options on the File submenu of the RMU Statistics main menu:

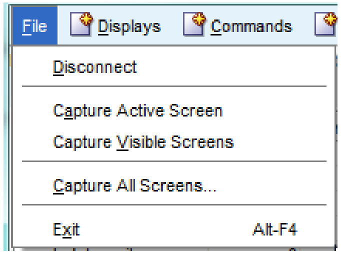


Figure 12 - Capture menu options

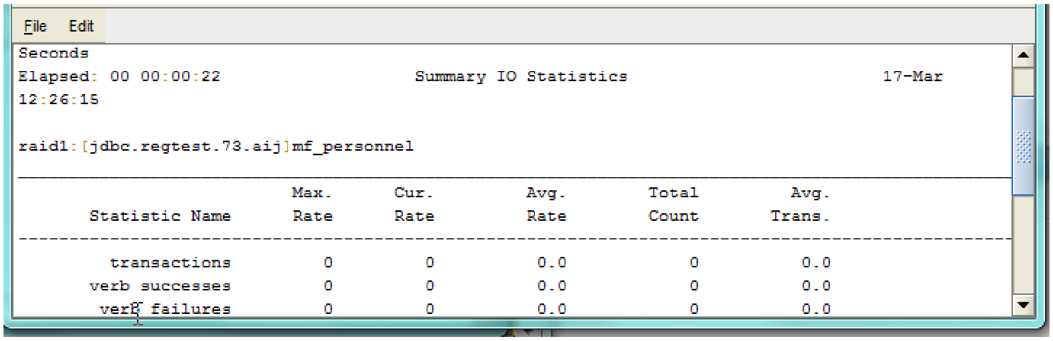
When selected the specified data is captured to text and appended to the Data Capture buffer.

The Capture menu options are:

- **Capture Active Screen** – capture the data from the top-most (active tab) statistics pane currently selected in the statistics pane area.

- **Capture Visible Screens** – captures the data from all the panes currently displayed in the statistics pane area.
- **Capture All Screens** – captures all the data that ORCM RMU Statistics collects. Basically, each statistic selectable from the Displays Menu will be captured.

1.7 Data Capture Buffer



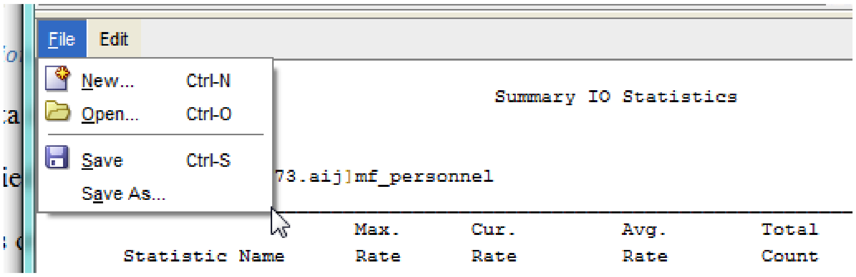
```
Seconds
Elapsed: 00 00:00:22                    Summary IO Statistics                         17-Mar
12:26:15

raid1:[jdbc.regtest.73.aij]mf_personnel
---------------------------------------------------------------------------------------------------
       Statistic Name           Max.      Cur.        Avg.        Total        Avg.
                                Rate      Rate        Rate        Count        Trans.
---------------------------------------------------------------------------------------------------
         transactions             0         0         0.0           0           0.0
        verb successes            0         0         0.0           0           0.0
         verb failures            0         0         0.0           0           0.0
```

Figure 13 - Data Capture buffer

The Data Capture buffer is an editor pane where the captured statistic data is displayed. This textual data may then be copied into the local system's Paste buffer or written out to file.

The Data Capture buffer has its own sub-menu allowing basic editing and file save options.

1.7.1 Data Capture Buffer File menu



```
                          Summary IO Statistics

73.aij]mf_personnel
---------------------------------------------------------------
                       Max.      Cur.        Avg.        Total
   Statistic Name      Rate      Rate        Rate        Count
```

Figure 14 - Data Capture Buffer File menu

The File menu within the Data Capture buffer has the following options:

- **New...** – will create a new save file.
- **Open…** – will open an existing text file for display; subsequent captures will be append to it.
- **Save** – saves the current buffer content to the current save file, if no file has been set, you will be prompted for a destination file specification.
- **Save As…** – allows you to select the file specification of the file the buffer will be saved to.

1.7.2 Data Capture Buffer Edit menu

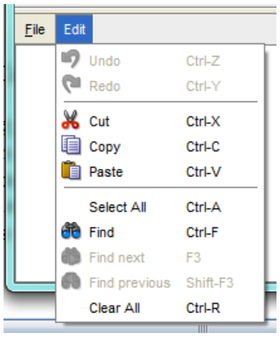


Figure 15 - Data Capture Buffer Edit menu

The Edit menu associated the Data Capture buffer has basic editing functions such as select, cut and paste as well as search capabilities.

The **Clear All** menu option may be used to clear the buffer contents.

The buffer is editable; as well as captured textual data, you may also add your own text to the buffer, as well as changing or removing existing text.

1.8 Statistical Information Collection

The database statistic information is collected and maintained in a global section that resides on each node for which the database is open. The statistic information is only physically reset when the database is closed on that node. Optionally, the statistic information can be exported for future collection.

Each database application process performs the work of collecting the numerical statistical information and storing that information in the database global section. Statistical information actively collected by each process also includes the columnar numeric statistics, such as the number of transactions, as well as tabular information such as the name of the lock for which the process is currently stalled.

Processes contribute collected statistic information to the aggregated database information.

This aggregation permits ORCM RMU Statistics to display statistic information for the entire database as well as for specific application processes.

Some statistical information is not stored in the global section but will be actively collected when a ORCM RMU Statistics session is active. This information is typically oriented towards process actions, such as checkpoint information, or after-image journal operations. If ORCM RMU Statistics is not active, this information is not collected.

The database statistic information remains available as long as the database is open, even if ORCM RMU Statistics is not currently active; however the process-specific statistic information remains available only as long as the process is attached to the database.

The database statistic information is collected until the database is either implicitly or explicitly closed. A database is implicitly closed when the database open mode is automatic and the last process detaches from the database. In this mode, an active ORCM RMU Statistics session will keep the database open.

Issuing the RMU Close command will explicitly close a database, regardless of the database open mode. Specifying a database close mode also controls when the database is actually closed.

Database statistics are reset to zero only when the database is closed.

1.9 Cost of Statistic Collection

The cost of collecting the statistic information varies depending on the size of the database and the operations being performed by the application. The statistic collection region in the database global section is based on a large number of factors, including number of users, storage areas, database files such as after-image journals and recovery- unit journals, logical areas and row caches. In general, however, collecting statistics information incurs a 7-10% overhead.

Oracle Rdb has optimized the runtime statistic collection code to minimize the runtime impact on the application.

Obviously, the actual impact on the system and database varies greatly depending on the specific database and application. However, without the ability to use the **RMU Show Statistic** utility and the **ORCM RMU Statistics** tool, you will never know how well your application is performing. In most cases, the benefits provided by the information greatly outweigh the cost of collecting the information.

1.10 Enabling & Disabling Statistics Collection

By default statistics collection is enabled on Oracle Rdb databases.

Note:

- ✓ Oracle recommends that you leave the statistics collection enabled, so that the information is available should you need it. When statistics collection is disabled for a database, no statistics are displayed for any of the processes attached to the database.

You can disable statistics using the ALTER DATABASE and IMPORT statements.

You can also enable statistics collection by defining the logical name RDM$BIND_STATS_ENABLED.

1.11 More Information

For more information on the STATISTICS COLLECTION IS ENABLED clause of the ALTER statements see the ***Oracle Rdb SQL Reference Manual Volume 2.***

For more information on the **RMU Show Statistics** utility, see the ***Oracle RMU Reference Manual***.

For more information about when to use statistics collection, see the ***Oracle Rdb Guide to Database Performance and Tuning***

Contents

Chapter 2
RMU Statistics Menus

The ORCM RMU Statistics main menu allows you to invoke the various options and features of the tool.

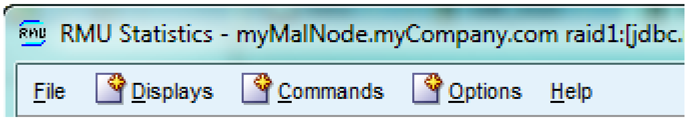


Figure 16 - Main Menu

The flowing sections detail the various sub menus found within the main menu.

2.1 File Menu

This menu allows you exit the application and capture screen data.

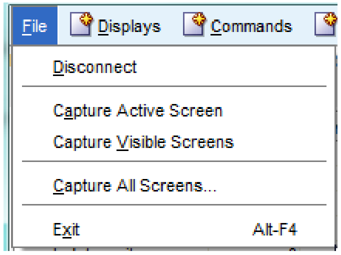


Figure 17 - File Menu options

The menu allows the following menu options:

Menu Option	Description
Disconnect	Disconnects from the current RMU server.
Capture Active Screen	Capture the data from the top-most (active tab) statistics pane
Capture Visible Screens	Capture the data from all the panes currently displayed

Capture All Screens	Capture all the data that ORCM RMU Statistics collects.
Exit	Disconnects from the current RMU server and exits the RMU Statistics tool.

See Data Capture for more details on the Capture options.

2.2 Displays Menu

This menu allows you to choose the statistics you wish to display.

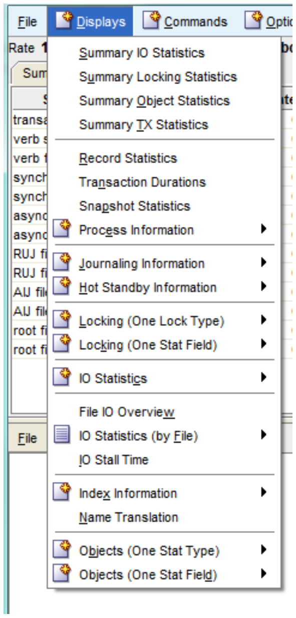
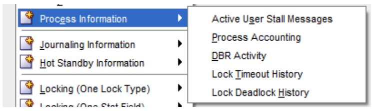
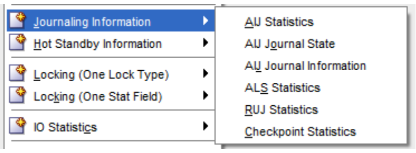
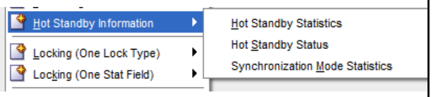


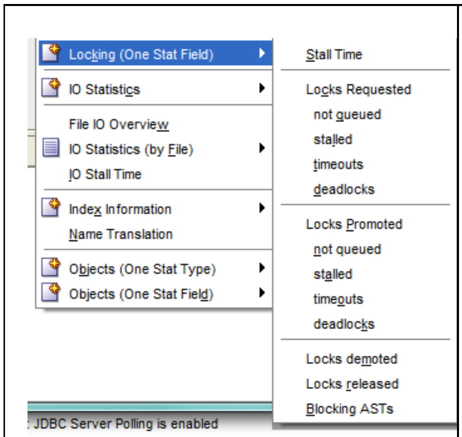
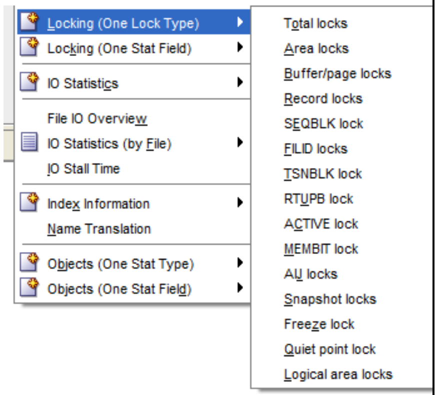
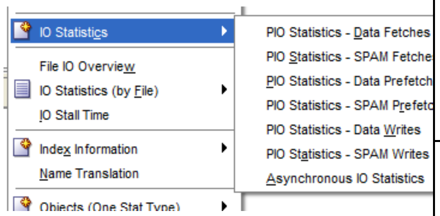
Figure 18 - Displays Menu options

The menu allows the following menu options:

Menu Option	Submenu Option	Description
Summary IO Statistics		Shows a summary of database I/O activity.
Summary Locking Statistics		Shows a summary of the locking activity.

Summary Object Statistics		Shows cumulative information for all database root file objects.
Summary Tx Statistics		Shows a summary of database transaction activity.
Record Statistics		Shows a summary of data row activity.
Transaction Duration		Shows overall real-time transaction processing performance.
Snapshot Statistics		Shows snapshot activity for update and read-only transactions.
Rdb Executive Statistics		Shows activity within the Rdb Executive sub-system
Process Information  Process Information Journaling Information Hot Standby Information Locking (One Lock Type) Active User Stall Messages Process Accounting DBR Activity Lock Timeout History Lock Deadlock History	Active User Stall Messages	Shows all processes attached to the database on the current node and, if a process stalled, shows the reason why the stall occurred.
	Process Accounting	Shows continuously updated OpenVMS accounting information about local processes in a cluster environment.
	DBR Activity	Shows one line of information for each database recovery process active on the node.
	Lock Timeout History	Identifies the object that causes a timeout event.
	Lock Deadlock	Identifies the object that causes a

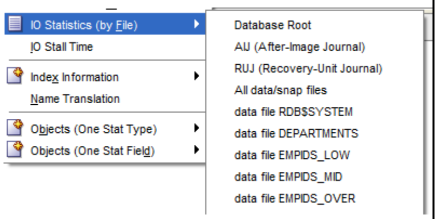
	History	deadlock event.
Journaling Information  Journaling Information Hot Standby Information Locking (One Lock Type) Locking (One Stat Field) IO Statistics AIJ Statistics AIJ Journal State AIJ Journal Information ALS Statistics RUJ Statistics Checkpoint Statistics	AIJ Statistics	Shows a summary of after-image journaling activity.
	Group Commit Statistics	Shows transaction group commit activity
	AIJ Journal State	Shows information relating to AIJ journaling in general.
	AIJ Journal Information	Shows information about all of a database's after-image journals on the current node.
	ALS Statistics	Shows AIJ log server activity.
	RUJ Statistics	Provides summary information for all active update transactions.
	Checkpoint Statistics	Shows transaction and checkpoint activity.
Hot Standby Information  Hot Standby Information Locking (One Lock Type) Locking (One Stat Field) Hot Standby Statistics Hot Standby Status Synchronization Mode Statistics	Hot Standby Statistics	Shows information about the performance of the Hot Standby feature.
	Hot Standby Status	Shows information about the state of the Hot Standby feature.
	Synchronization Mode Statistics	Shows statistics on the breakdown of each type of synchronization mode.
Locking (One Stat Field)	List of individual	Shows locking information for a specific lock statistic, compared to the various lock types.

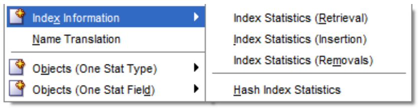
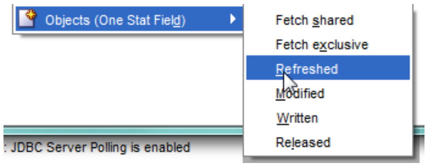
<table>
<tr>
<td>
</td>
<td>stat fields</td>
<td></td>
</tr>
<tr>
<td>Locking (One Lock Type)

</td>
<td>List of individual lock types</td>
<td>Shows locking information for a specific lock type.</td>
</tr>
<tr>
<td rowspan="3">IO Statistics

IO Statistics
File IO Overview
IO Statistics (by File)
IO Stall Time
Index Information
Name Translation
Objects (One Stat Type)
PIO Statistics - Data Fetches
PIO Statistics - SPAM Fetche
PIO Statistics - Data Prefetch
PIO Statistics - SPAM Prefetc
PIO Statistics - Data Writes
PIO Statistics - SPAM Writes
Asynchronous IO Statistics

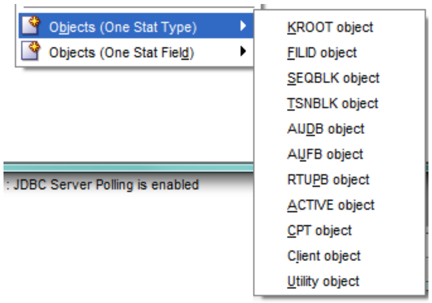
</td>
<td>PIO Statistics Data Fetches</td>
<td>Shows handling of buffer page reads in the PIO sub-system.

<b>Note</b>: If Global Buffers are enabled the Global Buffer Data Fetches panel will display.</td>
</tr>
<tr>
<td>PIO Statistics SPAM Fetches</td>
<td>Shows handling of SPAM page reads in the PIO sub-system.

<b>Note</b>: If Global Buffers are enabled the Global Buffer SPAM Fetches panel will display.</td>
</tr>
<tr>
<td>PIO Statistics Data Prefetches</td>
<td>Shows handling of buffer page pre-fetches in the PIO sub-system.</td>
</tr>
</table>

	PIO Statistics SPAM Prefetches	Shows handling of SPAM page pre-fetches in the PIO sub-system.
	PIO Statistics SPAM Access	Shows statistics about why the SPAM page was accessed.
	PIO Statistics Data Writes	Shows handling of buffer page writes in the PIO sub-system.
	PIO Statistics SPAM Writes	Shows handling of SPAM page writes in the PIO sub-system.
	Asynchronous IO Statistics	Shows information concerning asynchronous reads and writes to the database files.
File IO Overview		Shows a summary comparison of I/O activity for all database files.
IO Statistics (by File)  IO Statistics (by File) IO Stall Time Index Information Name Translation Objects (One Stat Type) Objects (One Stat Field) Database Root AIJ (After-Image Journal) RUJ (Recovery-Unit Journal) All data/snap files data file RDB$SYSTEM data file DEPARTMENTS data file EMPIDS_LOW data file EMPIDS_MID data file EMPIDS_OVER	List of individual database files	Shows I/O statistics for each individual file in the database.
IO Stall Time		Shows a summary of I/O stall activity.
Index Information	Index Statistics (Retrieval)	Shows the update activity of a database's sorted indexes when you perform any retrieval operation.
	Index Statistics	Shows the update activity of a

 Index Information Name Translation Objects (One Stat Type) Objects (One Stat Field) Index Statistics (Retrieval) Index Statistics (Insertion) Index Statistics (Removals) Hash Index Statistics	[(Insertion)]()	database's sorted indexes when you perform any insertion operation.
	[Index Statistics (Removal)]()	Shows the update activity of a database's sorted indexes when you perform any removal operation.
	[Hash Index Statistics]()	Shows the update and retrieval activity of a database's hashed indexes.
[Name Translation]()		Shows statistics on database dashboard updates and logical name translation.
[Objects (One Stat Field)]()  Objects (One Stat Field) Fetch shared Fetch exclusive Refreshed Modified Written Released : JDBC Server Polling is enabled	[Fetch Shared]()	Shows the statistics for the “objects fetch shrd” collection category.
	[Fetch Exclusive]()	Shows the statistics for the “objects fetch excl” collection category.
	[Refreshed]()	Shows the statistics for the “objects refreshed” collection category.
	[Modified]()	Shows the statistics for the “objects modified” collection category.
	[Written]()	Shows the statistics for the “objects written” collection category.
	[Released]()	Shows the statistics for the “objects released” collection

		category.
Objects (One Stat Type)  Objects (One Stat Type) Objects (One Stat Field) : JDBC Server Polling is enabled KROOT object FILID object SEQBLK object TSNBLK object AIJDB object AIJFB object RTUPB object ACTIVE object CPT object Client object Utility object	KROOT object	Shows the database control information that describes all of the other database objects.
	FILID object	Shows the storage area information.
	SEQBLK object	Shows information on the allocation of sequence numbers.
	TSNBLK object	Shows the information on the last committed transaction.
	AIJDB object	Shows the after-image journal control information.
	AIJFB object	Shows the after-image journal information.
	RTUPB object	Shows information on active users.
	ACTIVE object	Shows information on active transactions.
	CPT object	Shows information on the corrupt page table.
	RCACHE object	Shows the row cache information.
	CLIENT object	Shows client-specific information.
	CLTSEQ object	Shows the client “sequence” information.

	[UTILITY object]()	Shows information used by the Oracle RMU utility.

2.3 Commands Menu

This menu allows you to issue a session command.

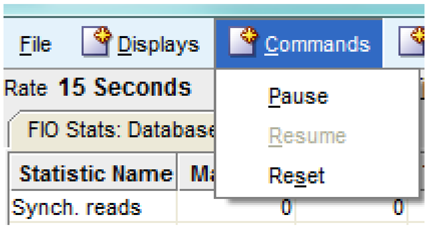


Figure 19 - Commands Menu options

The menu allows the following menu options:

Menu Option	Description
Pause	Temporarily pauses the collection of statistic information so that a snapshot view of the statistic information can be analyzed.
Resume	Resume a paused session.
Reset	Resets the per-process view of the statistic information. The original database statistic information can always be retrieved. **Note**: Selecting the ***Reset*** menu option does not reset the database statistic information.

2.4 Options Menu

This menu allows you to change the statistics refresh rate and histogram filters.

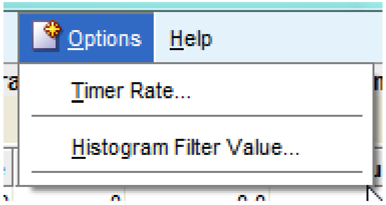


Figure 20 - Options Menu options

How often statistical data is requested from the connected RMU server may be changed by selecting the **Timer Rate…** option. See Setting refresh rate for more details.

2.5 Help Menu

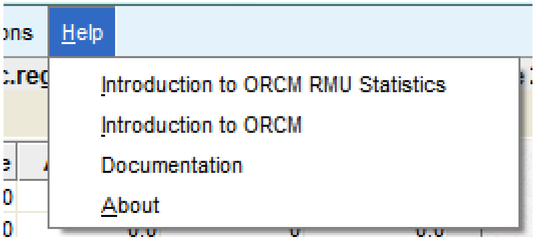


Figure 21 - Help Menu options

This menu allows you to display help on various ORCM RMU Statistics topics and to display ORCM documentation.

Contents

Chapter 3

RMU Statistics Panels

The following is a list of RMU Statistic Panels sorted by alphabetic order.

Panels
ACTIVE object
Active User Stall Messages
AIJ Journal Information
AIJ Journal State
AIJ Statistics
AIJDB object
AIJFB object
ALS Statistics
CLIENT object
CLTSEQ object
CPT object
DBR Activity

File IO Overview
FILID object
Group Commit Statistics
Hash Index Statistics
Hot Standby Statistics
Hot Standby Status
Index Statistics (Insertion)
Index Statistics (Removal)
Index Statistics (Retrieval)
IO Stall Time
KROOT object
Lock Deadlock History
Lock Timeout History
Name Translation
Objects Fetch Exclusive
Objects Fetch Shared

Objects Modified
Objects Refreshed
Objects Released
Objects Written
PIO Statistics Data Fetches
PIO Statistics Data Prefetches
PIO Statistics Data Writes
PIO Statistics SPAM Access
PIO Statistics SPAM Fetches
PIO Statistics SPAM Prefetches
PIO Statistics SPAM Writes
Process Accounting
RCACHE object
Rdb Executive Statistics
Record Statistics
RTUPB object

RUJ Statistics
SEQBLK object
Snapshot Statistics
Summary IO Statistics
Summary Locking Statistics
Summary Object Statistics
Summary Tx Statistics
Transaction Duration
TSNBLK object
UTILITY object

Contents

Chapter 4

Summary Statistics

The following panels provide summary statistical information:

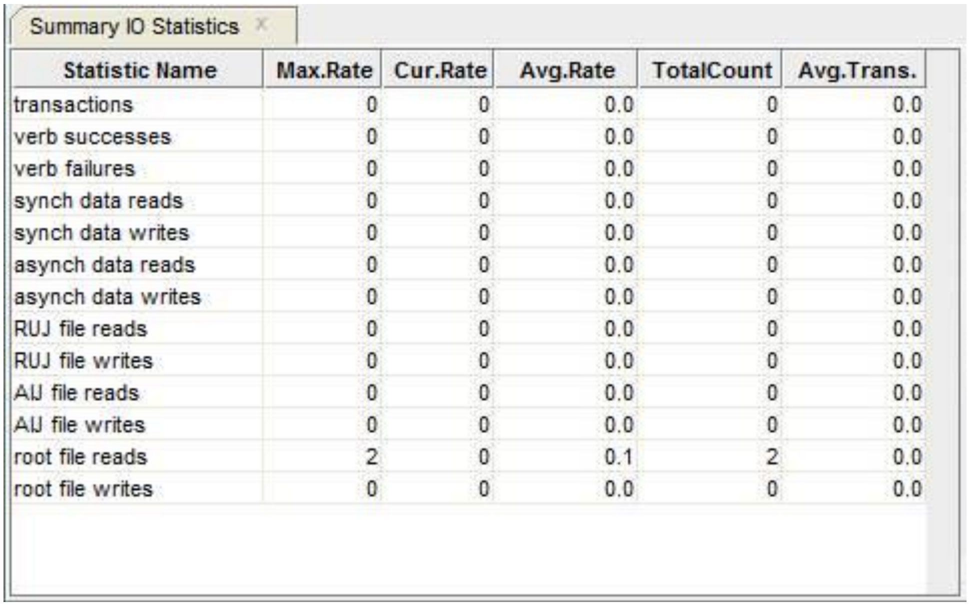
- Summary IO Statistics
- Summary Locking Statistics
- Summary Object Statistics
- Summary Tx Statistics
- Record Statistics

4.1 Summary IO Statistics

This panel summarizes database I/O activity and indicates transaction and verb execution rates.

This panel should be used as a starting point; use the **IO Statistics (by file)** panel and the ***File IO Overview*** panel to narrow down I/O problems.

The information displayed in the panel is a summation of the information displayed on a per-storage-area basis in the **IO Statistics (by file)** panels.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
synch data reads	0	0	0.0	0	0.0
synch data writes	0	0	0.0	0	0.0
asynch data reads	0	0	0.0	0	0.0
asynch data writes	0	0	0.0	0	0.0
RUJ file reads	0	0	0.0	0	0.0
RUJ file writes	0	0	0.0	0	0.0
AIJ file reads	0	0	0.0	0	0.0
AIJ file writes	0	0	0.0	0	0.0
root file reads	2	0	0.1	2	0.0
root file writes	0	0	0.0	0	0.0

Figure 22 - Summary IO Statistics Pane

The following sections detail the information presented in this panel.

4.1.1 transactions

The number of completed database transactions.

This is the count of the COMMIT and ROLLBACK statements that have executed.

4.1.2 verb successes

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 statement to treat the entire block as a single verb.

4.1.3 verb failures

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 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.

4.1.4 synch data reads

The number of synchronous read QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases and snapshot files.

This operation reads database pages synchronously from the database.

4.1.5 synch data writes

The number of synchronous write QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot files (.SNP). This operation writes modified database pages synchronously back to the database.

4.1.6 asynch data reads

The number of asynchronous read QIOs (queued I/O requests) 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.

4.1.7 asynch data writes

The number of asynchronous write QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot files (.SNP). This operation writes modified database pages asynchronously back to the database.

4.1.8 RUJ file reads

The number of read QIOs (queued I/O requests) 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.

4.1.9 RUJ file writes

The number of write QIOs (queued I/O requests) 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.

4.1.10 AIJ file reads

The number of read QIOs 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.

4.1.11 AIJ file writes

The total number of write QIOs (queued I/O requests) 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 utility.

This statistic field includes both synchronous and asynchronous I/O write requests.

4.1.12 root file reads

The number of read QIOs (queued I/O requests) issued to the database root (.RDB) file. Oracle Rdb reads the .RDB file when a new user attaches to the database and when an .RDB file control block needs to be updated because of database activity on another OpenVMS cluster node.

4.1.13 root file writes

The number of write QIOs (queued I/O requests) issued to the database root (.RDB) file. Oracle Rdb writes to the .RDB file when a user issues a COMMIT or ROLLBACK statement. Other events also cause updates to the .RDB file.

Notes:

- ✓ Compare the **verb successes** statistic to the **verb failures** statistic. A high failure rate compared to the success rate indicates that many things are failing. The problem might be too many deadlocks, but this is really application-dependent.

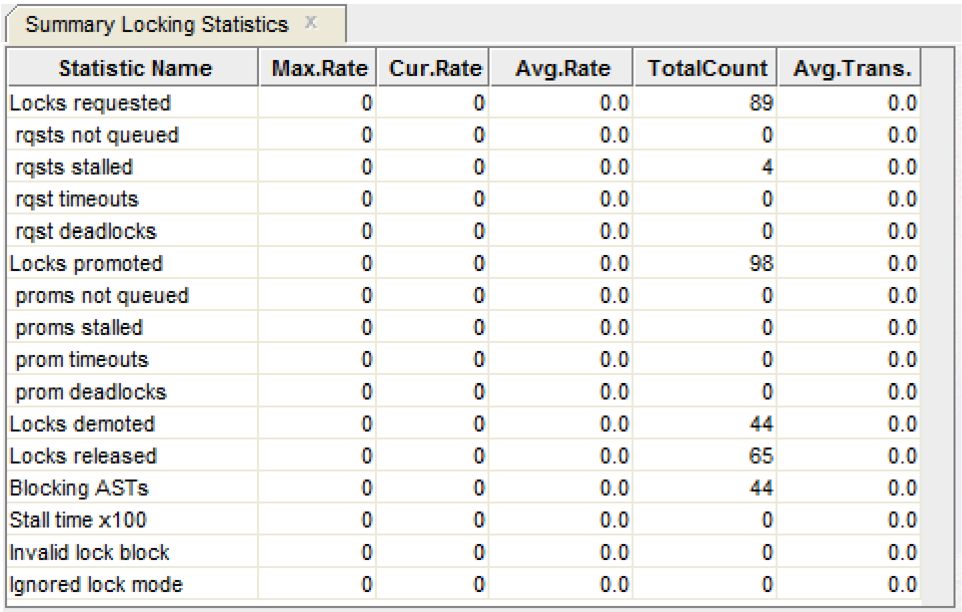
- ✓ Compare the **RUJ file reads** and the **RUJ file writes** statistics. Too many RUJ file reads indicate too many rollbacks. Too many RUJ file writes might indicate data page contention problems or transactions that modify a lot of data. More than one write for each transaction indicates that marked pages are being written back to the database before the actual commit, meaning either another recovery-unit requested the page, the buffer pool overflowed, or there are very large transactions with lots of updates.

- ✓ Compare the **synchronous data file reads** with the **asynchronous data file reads**. If the ratio is high, it is likely that there are too many sequential scans being performed.

  These scans can be either sequential access to tables, or full btree index scans. In either case, you should further investigate this issue to understand if this is normal behavior of the application. You may need to add or re-define your btree indexes to minimize the number of sequential scans being performed.

4.2 Summary Locking Statistics

This pane monitors the total database locking activity. The statistics in this panel are the totals for all types of database locks.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
Locks requested	0	0	0.0	89	0.0
rqsts not queued	0	0	0.0	0	0.0
rqsts stalled	0	0	0.0	4	0.0
rqst timeouts	0	0	0.0	0	0.0
rqst deadlocks	0	0	0.0	0	0.0
Locks promoted	0	0	0.0	98	0.0
proms not queued	0	0	0.0	0	0.0
proms stalled	0	0	0.0	0	0.0
prom timeouts	0	0	0.0	0	0.0
prom deadlocks	0	0	0.0	0	0.0
Locks demoted	0	0	0.0	44	0.0
Locks released	0	0	0.0	65	0.0
Blocking ASTs	0	0	0.0	44	0.0
Stall time x100	0	0	0.0	0	0.0
Invalid lock block	0	0	0.0	0	0.0
Ignored lock mode	0	0	0.0	0	0.0

Figure 23 - Summary Locking Statistics Pane

The following sections detail the information presented in this panel.

4.2.1 Locks requested

The number of lock requests, also referred to as “enqueue” lock re-quests, for new locks. Whether the lock request succeeds or fails, it is included in this count.

The **rqsts no queued**, **rqsts_stalled** and **rqst_deadlocks** count provide further detail for enqueue lock request statistics.

4.2.2 rqsts not queued

The number of enqueue lock requests for new locks that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting and, when a conflict is detected, resorts to a secondary locking protocol to avoid unnecessary deadlocks. This number is one measure of lock contention.

4.2.3 rqsts stalled

The number of enqueue lock requests for new locks that were stalled because of a lock conflict. Whether or not the lock request ultimately succeeds, it is included in this count. This number is one measure of lock contention.

4.2.4 rqst timeouts

The total number of lock requests that could not be granted because they timed out. These are typically logical areas.

Each lock timeout reported in the **rqst timeouts** field is also reported in the **rqsts stalled** field. This is because each timed out request is also a stalled request.

4.2.5 rqst deadlocks

The number of stalled enqueue lock requests for new 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.

Each lock deadlock reported in the **rqst deadlocks** field is also reported in the **rqsts stalled** field. This is because each deadlocked request is also a stalled request

4.2.6 Locks promoted

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.

The **proms not queued**, **proms stalled**, and **prom deadlocks** counts provide further detail for the locks promotion statistics.

4.2.7 proms not queued

The number of enqueue lock requests to promote an existing lock that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting. When a conflict is detected, Oracle Rdb resorts to a secondary locking protocol to avoid unnecessary deadlocks. This number is one measure of lock contention.

4.2.8 proms stalled

This field gives the number of enqueue lock requests to promote an existing lock to a higher lock mode that were stalled because of a lock conflict. Whether or not the lock request ultimately succeeds, it is included in this count. This number is one measure of lock contention.

4.2.9 prom timeouts

The total number of lock promotions that could not be granted because they timed out. These are typically logical areas.

Each promotion timeout reported in the **prom timeouts** field is also reported in the **proms stalled** field. This is because each timed out request is also a stalled request.

4.2.10 prom deadlocks

The number of stalled enqueue lock requests to promote an existing lock 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, this number does not necessarily reflect the number of deadlocks reported to the application program.

Each promotion deadlock reported in the **prom deadlocks** field is also reported in the **proms stalled** field. This is because each deadlocked request is also a stalled request.

4.2.11 Locks demoted

The number of enqueue lock requests to demote an existing lock to a lower lock mode. These requests always succeed.

4.2.12 Locks released

The number of deallocating lock requests to release an existing lock. These requests always succeed. The number of outstanding locks can be determined by the formula:

(locks requested) - (rqsts not queued) - (rqst deadlocks) - (locks released).

4.2.13 Blocking ASTs

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 ORCM RMU Statistic 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 per-formed 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. ORCM RMU Statistic does not differentiate between these two types of blocking ASTs.

4.2.14 Stall time x100

The total time in hundredths of a second spent by all users waiting for a lock. Since more than one user can be waiting for a lock at the same time, this total can be greater than the actual elapsed clock time. This statistic gives a relative measure of work lost due to lock conflicts.

4.2.15 Invalid lock block

The number of invalid lock value blocks encountered during a lock request or lock promote operation. The lock value block is the means by which Oracle Rdb communicates information between processes, either on the local node or to remote nodes in the cluster.

The following events may cause a lock value block to become invalid:

- The premature failure of a process while holding a lock in a strong mode, such as protected-update (PU) or exclusive-update (EX).
- The failure of a node on which a process is holding a lock in a strong mode.
- The OpenVMS lock tree is dynamically re-mastered to a new node in the cluster, while processes are holding locks in a strong mode.

Note that there is no storage area specific statistic for invalid lock value blocks.

4.2.16 Ignored lock mode

The number of lock release operations that could not be performed due to the lock release being requested from AST context. The lock is instead demoted to NL mode, and can be released later.

Note that there is no storage area specific statistic for ignored lock mode statistics.

Notes:

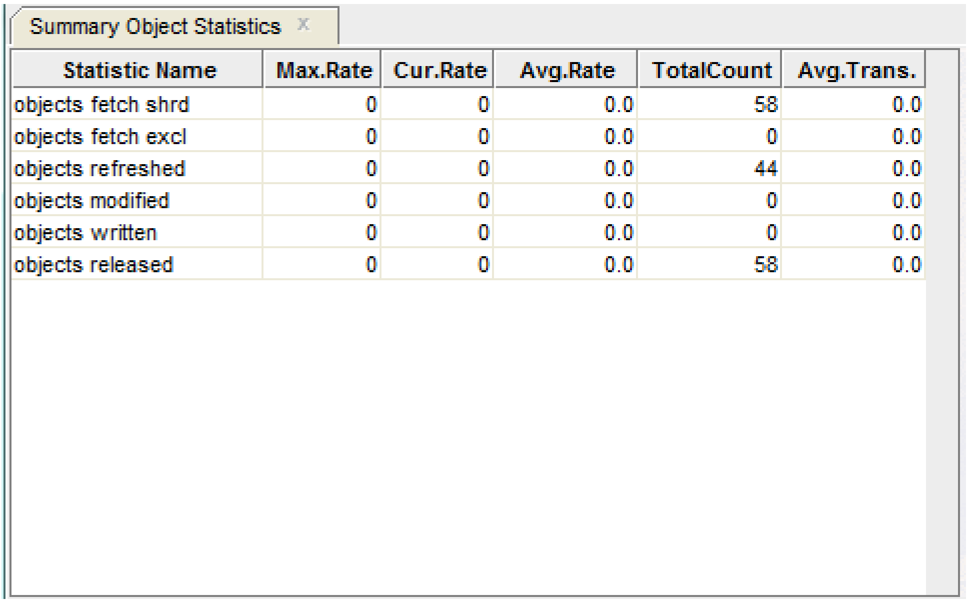
- ✓ The most important statistic is the **stall time x100** field. This field provides a measure of lost work as a result of waiting for lock requests to be granted. Eliminating lock stalls will probably provide the largest increase in performance throughput.

- ✓ Look at the **rqsts not queued**, **rqsts stalled**, and **rqst deadlocked** statistics. High numbers for these statistics, relative to the norm for your database, indicate contention problems. The problem might be that your database design is not optimal.

  The problem could also be a high number of blocking asynchronous system traps (ASTs) or it may be that users are contending for frequently accessed data.

4.3 Summary Object Statistics

This pane provides cumulative information for all database root file objects.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
objects fetch shrd	0	0	0.0	58	0.0
objects fetch excl	0	0	0.0	0	0.0
objects refreshed	0	0	0.0	44	0.0
objects modified	0	0	0.0	0	0.0
objects written	0	0	0.0	0	0.0
objects released	0	0	0.0	58	0.0

Figure 24 - Summary Object Statistics Pane

The following sections detail the information presented in this panel.

4.3.1 objects fetch shrd

The number of objects that were fetched for shared retrieval access.

4.3.2 objects fetch excl

The number of objects that were fetched for exclusive access with the intention of subsequently being updated.

Note:

✓ This statistic does not indicate that the object was actually updated.

4.3.3 objects refreshed

The number of objects for which information in the global section was detected as being stale, so the information was read again from the database root file.

4.3.4 objects modified

The number of objects for which information was modified. Only objects fetched for exclusive access can be modified.

4.3.5 objects written

The number of objects for which information was written back to the database root file.

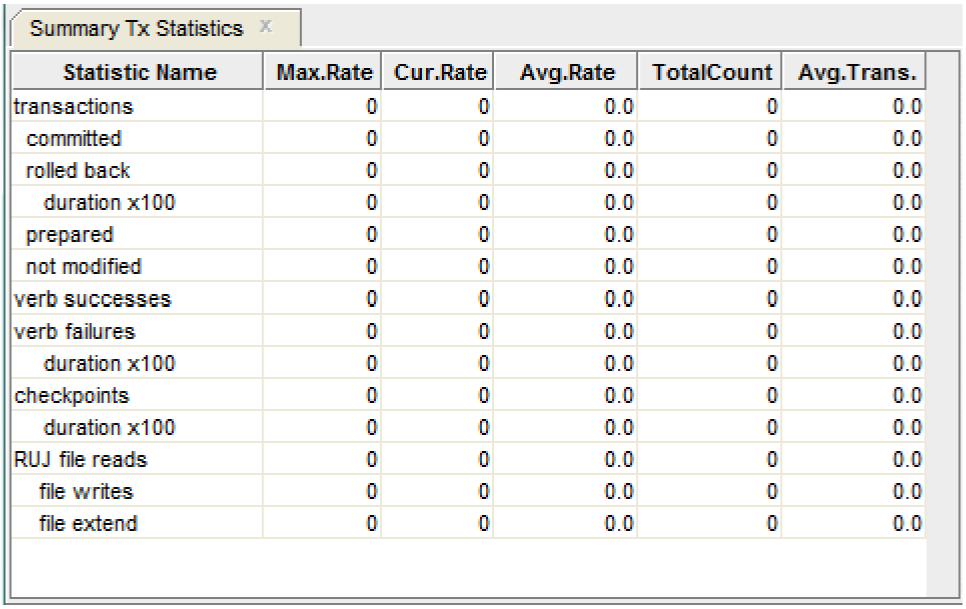
4.3.6 objects released

The number of objects where their shared or exclusive access was released to other processes.

Contents

4.4 Summary Tx Statistics

This pane summarizes database transaction activity and indicates transaction and verb execution rates.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
committed	0	0	0.0	0	0.0
rolled back	0	0	0.0	0	0.0
duration x100	0	0	0.0	0	0.0
prepared	0	0	0.0	0	0.0
not modified	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
duration x100	0	0	0.0	0	0.0
checkpoints	0	0	0.0	0	0.0
duration x100	0	0	0.0	0	0.0
RUJ file reads	0	0	0.0	0	0.0
file writes	0	0	0.0	0	0.0
file extend	0	0	0.0	0	0.0

Figure 25 - Summary Tx Statistics Pane

The following sections detail the information presented in this panel.

4.4.1 transactions

The number of completed database transactions. This is the count of the transaction COMMIT and ROLLBACK statements that have executed.

4.4.2 committed

The actual number of transactions that committed successfully to the database.

4.4.3 rolled back

The actual number of transactions that aborted and were not applied to the database.

4.4.4 duration x100

The duration of the transaction rollback operation, expressed in hundredths of a second displayed as a whole number. For example, the value “500” is actually “5” seconds.

4.4.5 prepared

The number of distributed transactions that have successfully “prepared” themselves for subsequent transaction commit.

4.4.6 not modified

The number of committed transactions that did not modify any database objects. This field includes both read-only and read/write transactions.

4.4.7 verb successes

The number of completed verbs that returned a successful status code.

Note:

- ✓ 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 statement to treat the entire block as a single verb.

4.4.8 verb failures

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.

Excessive verb failures are usually an indication of a failed constraint, such as uniqueness criteria, or an invalid 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.

4.4.9 duration x100

The duration of the verb failure rollback operation, expressed as hundredths of a second displayed as a whole number. For example, the value “500” is actually “5” seconds.

4.4.10 checkpoints

The number of checkpoints performed by users. This field does not include the initial checkpoint when the user first attaches to the database.

4.4.11 duration x100

The checkpoint duration, expressed in hundredths of a second displayed as a whole number. For example, the value “500” is actually “5” seconds.

4.4.12 RUJ file reads

The total number of read I/O operations performed on the RUJ journal during the transaction undo phase. The RUJ file is never written by the Database Recovery process (DBR).

This statistic field includes both synchronous and asynchronous I/O read requests.

4.4.13 file writes

The total number of write I/O operations performed on the RUJ journal during the transaction phase.

This statistic field includes both synchronous and asynchronous I/O write requests.

4.4.14 file extends

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.

Notes:

- ✓ Compare the **verb successes** statistic to the **verb failures** statistic. A high failure rate compared to the success rate indicates that many things are failing. The problem might be too many deadlocks, but this is really application-dependent.

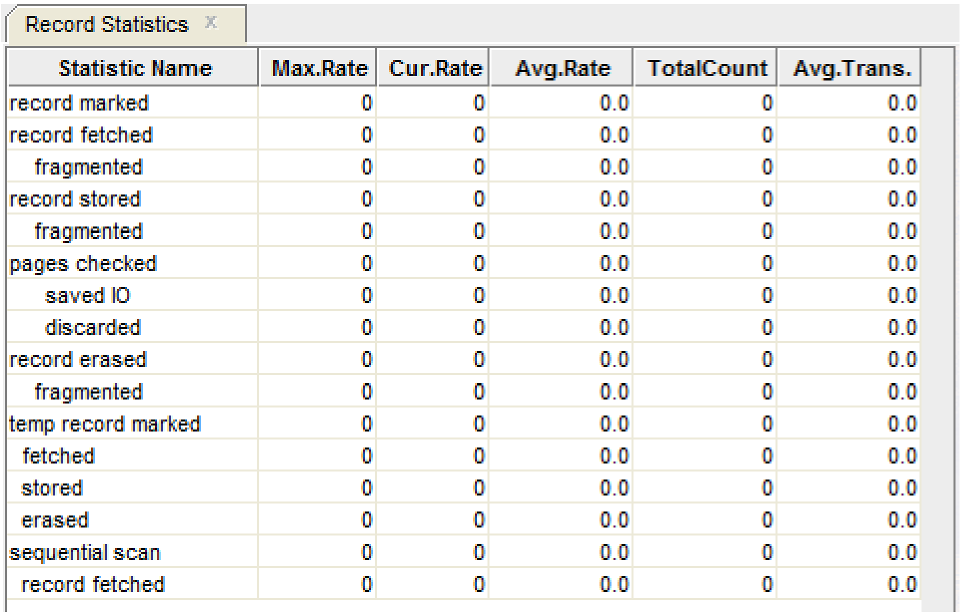
- ✓ Compare the **RUJ file reads** and the **RUJ file writes** statistics. Too many RUJ file reads indicate too many rollbacks. Too many RUJ file writes might indicate data page contention problems or transactions that modify a lot of data. More than one write for each transaction indicates that marked pages are being written back to the database before the actual commit, meaning either another recovery-unit requested the page, the buffer pool overflowed, or there are very large transactions with lots of updates.

- ✓ Examine the **rolled back duration x100** statistic field to ensure that transaction rollback operations are not excessively long.

Contents

4.5 Record Statistics

This pane shows a summary of data row activity for storage areas in the database. Data row activity includes modification (marked), retrieval (fetch), store or erase operations.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
record marked	0	0	0.0	0	0.0
record fetched	0	0	0.0	0	0.0
fragmented	0	0	0.0	0	0.0
record stored	0	0	0.0	0	0.0
fragmented	0	0	0.0	0	0.0
pages checked	0	0	0.0	0	0.0
saved IO	0	0	0.0	0	0.0
discarded	0	0	0.0	0	0.0
record erased	0	0	0.0	0	0.0
fragmented	0	0	0.0	0	0.0
temp record marked	0	0	0.0	0	0.0
fetched	0	0	0.0	0	0.0
stored	0	0	0.0	0	0.0
erased	0	0	0.0	0	0.0
sequential scan	0	0	0.0	0	0.0
record fetched	0	0	0.0	0	0.0

Figure 26 - Record Statistics Pane

The following sections detail the information presented in this panel.

4.5.1 record marked

The number of records marked. A record is marked when it is modified or it is erased, but not when it is stored. This field does not include records modified in a temporary table.

4.5.2 record fetched

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.

4.5.3 fragmented

The number of record fragments that Oracle Rdb had to fetch. A record is fragmented if it is too large to fit on one page. A fragmented record requires more CPU time and more virtual memory, and often requires additional I/O operations because each record fragment must be fetched.

4.5.4 record stored

The number of records stored in the database. This field does not include records stored in temporary tables.

4.5.5 fragmented

The number of rows stored as fragmented records in the database. This number indicates that a page size is smaller than a record's uncompressed size, including overhead. You should use the RMU Analyze Areas utility to further analyze the problem and then increase the page size for the storage area that has the problem.

4.5.6 pages checked

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.

4.5.7 saved IO

The number of pages checked that did not result in an I/O because the page was already in the buffer. It is essentially a “for free” pages checked.

4.5.8 discarded

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.

4.5.9 record erased

The number of records erased from the database. This field does not include records erased from temporary tables.

4.5.10 fragmented

The number of fragmented records erased from the database.

4.5.11 temp record marked

The number of records marked in a temporary table. A record is marked when it is modified or it is erased, but not when it is stored.

4.5.12 fetched

The number of records fetched from a temporary table.

4.5.13 stored

The number of records stored in a temporary table.

4.5.14 erased

The number of records erased from a temporary table.

4.5.15 sequential scan

The number of sequential scans made to retrieve records from the database.

4.5.16 record fetched

The number of records retrieved using sequential scan.

Note:

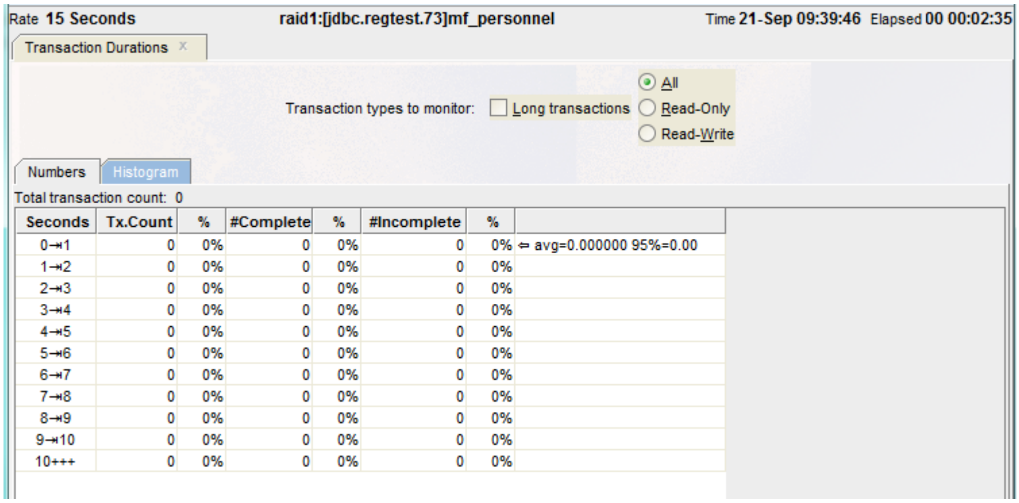
- ✓ If the value of the **fragmented records fetched** statistic is high compared to the number of records fetched, Oracle Corporation recommends that you use the RMU Analyze Areas utility to further analyze the problem to see how many records are actually fragmented.
  If the number of pages discarded is high compared to the number of pages checked, one or more of the storage areas may contain a SPAM threshold problem.

  Determine if you are using compression on data records; if you are using data record compression, you should review the ***Logical Area Statistics*** screen of the **RMU Show Statistics** utility to identify the tables that are experiencing the SPAM threshold problem. Use the RMU Analyze utility to determine the average compression ratio, then recalculate the storage area thresholds based on the information you have gathered. The problem could also be related to non-unique indexes. If an index has a duplicates allowed you need to follow the instructions contained in the ***Oracle Rdb Guide to Database Performance and Tuning*** manual on how to properly calculate thresholds.

Contents

Chapter 5
Transaction Duration Statistics

This pane shows a graphical or numerical representation of the duration of each application transaction in the database.



Rate **15 Seconds** **raid1:[jdbc.regtest.73]mf_personnel** Time **21-Sep 09:39:46** Elapsed **00 00:02:35**

Transaction types to monitor: ☐ Long transactions

● All
○ Read-Only
○ Read-Write

Total transaction count: 0

Seconds	Tx.Count	%	#Complete	%	#Incomplete	%	
0⇥1	0	0%	0	0%	0	0%	⇦ avg=0.000000 95%=0.00
1⇥2	0	0%	0	0%	0	0%	
2⇥3	0	0%	0	0%	0	0%	
3⇥4	0	0%	0	0%	0	0%	
4⇥5	0	0%	0	0%	0	0%	
5⇥6	0	0%	0	0%	0	0%	
6⇥7	0	0%	0	0%	0	0%	
7⇥8	0	0%	0	0%	0	0%	
8⇥9	0	0%	0	0%	0	0%	
9⇥10	0	0%	0	0%	0	0%	
10+++	0	0%	0	0%	0	0%	

Figure 27 - Transaction Durations Pane

The presentation granularity can be configured as seconds, the default, or minutes for those long-running transaction environments.

You may also choose whether to see all transactions or just read-only or read-write transactions.

The pane contains two sub-panes, Numbers, which shows the information in numeric form, and Histogram which shows a histogram of the transaction durations.

The panel also allows you to change the type of transactions you wish to display.

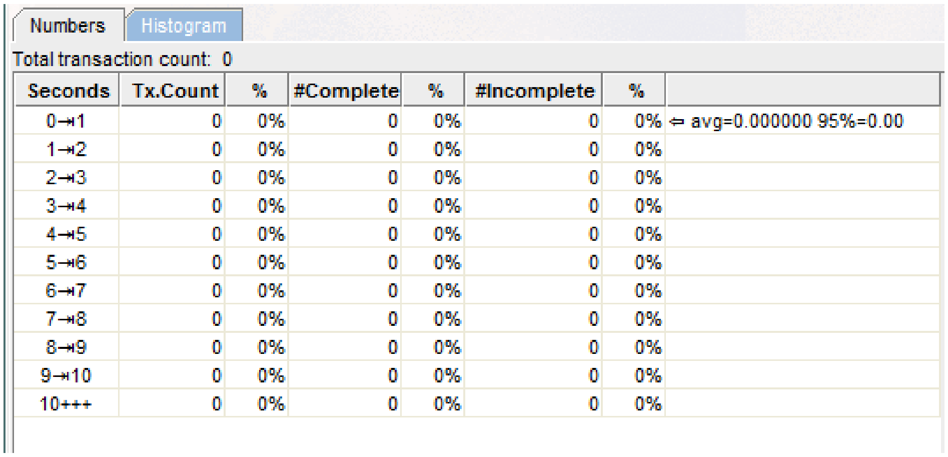
5.1 Long Transaction check box

Check this box if you wish to show long lasting transaction information. The data will be displayed showing duration intervals of minutes rather than the default of seconds.

5.2 Transaction Type

Select either **All** (the default), **Read-Only** or **Read-Write** to select the type of transaction you wish to display.

5.3 Numbers sub-panel



Total transaction count: 0

Seconds	Tx.Count	%	#Complete	%	#Incomplete	%	
0⇥1	0	0%	0	0%	0	0%	⇐ avg=0.000000 95%=0.00
1⇥2	0	0%	0	0%	0	0%	
2⇥3	0	0%	0	0%	0	0%	
3⇥4	0	0%	0	0%	0	0%	
4⇥5	0	0%	0	0%	0	0%	
5⇥6	0	0%	0	0%	0	0%	
6⇥7	0	0%	0	0%	0	0%	
7⇥8	0	0%	0	0%	0	0%	
8⇥9	0	0%	0	0%	0	0%	
9⇥10	0	0%	0	0%	0	0%	
10+++	0	0%	0	0%	0	0%	

Figure 28 - Transaction Durations Pane – Numbers sub-panel

The following sections detail the information presented in this panel.

5.3.1 Total transaction count

This is the total number of transactions represented by the information on the panel.

Reset the number of transactions with the Reset option from the Commands menu:

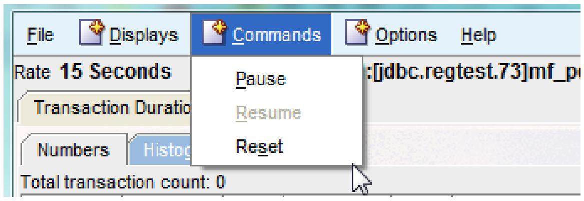


Figure 29 - Reset Menu option

5.3.2 Duration

This column contains the transaction duration categories. For the short-duration transaction display, the format “n⇥m” means that each category includes transactions of n seconds to less than but not including m seconds.

For the long-duration transaction display, the format “n⇥m” means that each category includes transactions of n minutes to less than but not including m minutes.

The 10+++ category contains all transactions of 10 seconds/minutes or more.

5.3.3 Tx.Count, %

The **Tx.Count** field contains the number of transactions in each transaction duration category. The associated “**%**” column shows the percentage of the total transactions in each transaction duration category.

5.3.4 #Complete, %

The **#Complete** field contains the number of transactions that completed within the time specified by each transaction duration category. The “**%**” column shows the percentage of the total transactions that completed within the time specified by each transaction duration category.

5.3.5 #Incomplete, %

The **#Incomplete** field contains the number of transactions that did not complete within the time specified by each transaction duration category. The associated “**%**” column shows the percentage of the total transactions that did not complete within the time specified by each transaction duration category.

The **Complete %** and **Incomplete %** fields should always add up to 100% for each transaction duration category.

5.3.6 ← average

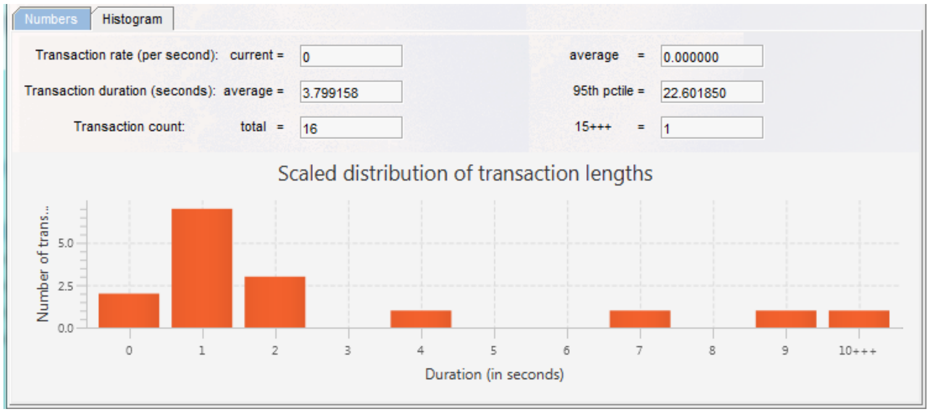
The “← **average**” indicator on the right side of the panel points to the transaction duration category that contains the average transaction duration time. The position of this indicator changes at runtime to maintain an accurate indication of the average transaction duration time.

5.3.7 ← 95th %

The “← **95th %**” indicator on the right side of the panel points to the transaction duration category that contains the 95th percentile transaction duration time. The position of this indicator changes at runtime to maintain an accurate indication of the 95th percentile transaction duration time.

Contents

5.4 Histogram sub-panel



Transaction rate (per second): current = 0

average = 0.000000

Transaction duration (seconds): average = 3.799158

95th pctile = 22.601850

Transaction count: total = 16

15+++ = 1

Figure 30 - Transaction Durations Pane – Histogram sub-panel

The following sections detail the information presented in this panel.

5.4.1 transaction rate current

The total number of transactions being performed, per second.

5.4.2 average

The average number of transactions being performed, per second.

5.4.3 transaction duration average

The average transaction duration.

5.4.4 95th pctile

The 95th percentile transaction duration. If the 95th percentile occurs in the “10+++” category, then the exact percentile is approximate only, and is displayed using the “~” character.

5.4.5 transaction count total

This is the total number of transactions represented by the information on the panel.

5.4.6 transaction count 15+++

The number of transactions that are 15 seconds/minutes or longer.

Note:

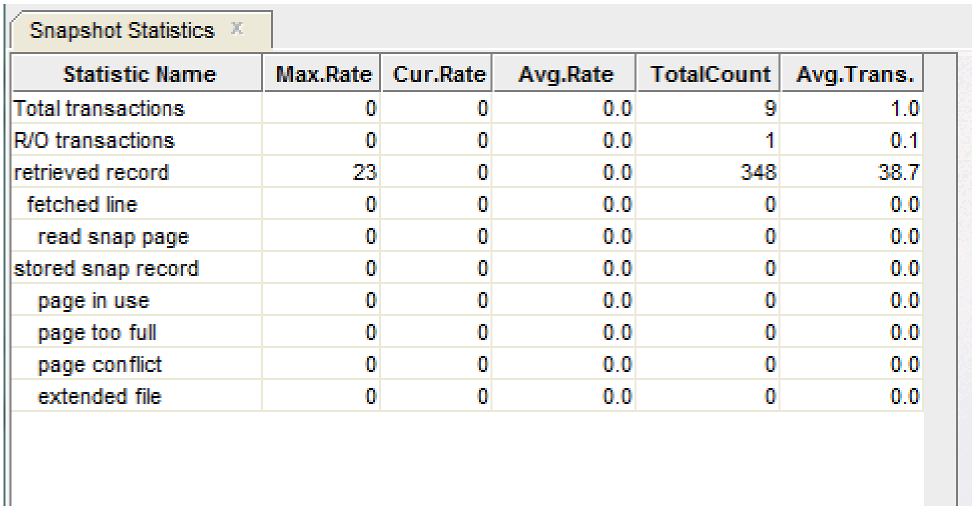
- ✓ The Histogram sub-panel presents a graph that indicates the overall real-time processing performance of the database. It shows the transaction rate (number per second, by default) and the time required to complete transactions. The duration of each transaction is measured from the first SQL SET TRANSACTION statement to the completion of the COMMIT or ROLLBACK verb. In addition to displaying average transaction duration, this panel shows a graph of the transaction durations. As each transaction completes, it is added to the cumulative histogram display. The 95 percentile response time is continuously updated to show the time in which 95 percent or more of the transactions complete. The panel gives an indication of subjective system response time.

- ✓ The Numbers sub-panel provides the exact number of transactions in each of the 16 time categories. The numeric display also shows the number of transactions that completed and the number of transactions that did not complete within each time category.

Contents

Chapter 6
Snapshot Statistics

This pane shows snapshot activity for both update and read-only transactions.



Snapshot Statistics

Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
Total transactions	0	0	0.0	9	1.0
R/O transactions	0	0	0.0	1	0.1
retrieved record	23	0	0.0	348	38.7
fetched line	0	0	0.0	0	0.0
read snap page	0	0	0.0	0	0.0
stored snap record	0	0	0.0	0	0.0
page in use	0	0	0.0	0	0.0
page too full	0	0	0.0	0	0.0
page conflict	0	0	0.0	0	0.0
extended file	0	0	0.0	0	0.0

Figure 31 - Snapshot Statistics Pane

The **retrieved record**, **fetched line**, and **read snap page** statistics relate to read-only transactions, and the last five statistics are relevant for update transactions.

The following sections detail the information presented in this panel.

6.1.1 Total transactions

The total number of transactions that have been committed or rolled back, including read-only transactions.

6.1.2 R/O transactions

The total number of read-only transactions that have been committed or rolled back.

6.1.3 retrieved record

The number of records retrieved by read-only transactions.

6.1.4 fetched line

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.

6.1.5 read snap page

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.

6.1.6 stored snap record

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.

6.1.7 page in use

The number of pages fetched that were unsuitable for storing snapshot records because the page was owned by another transaction. A new snapshot page cannot be used again until the Transaction Sequence Number (TSN) that denotes the age of the page is less than the oldest active TSN in the database. This ensures that read-only transactions always find the correct version of a record.

6.1.8 page too full

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.

6.1.9 page conflict

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.

6.1.10 extended file

The number of times the snapshot file has been extended. The snapshot file is extended when an update operation cannot find a suitable page on which to store a snapshot record after checking a certain number of pages.

Note:

- ✓ Look at the **retrieved record**, the **fetched line**, and the **read snap page** statistics. If the fetched line number is much higher than that for retrieved record, read-only transactions are doing a lot of work to access their records. The problem might be that the access paths are not optimal.

- ✓ The **read snap page** also indicates the amount of database activity. If the read snap page number is higher than the number for retrieved record, the records fetched may be those that have been frequently updated. A zero indicates that read-only transactions are rarely accessing data modified by update transactions.

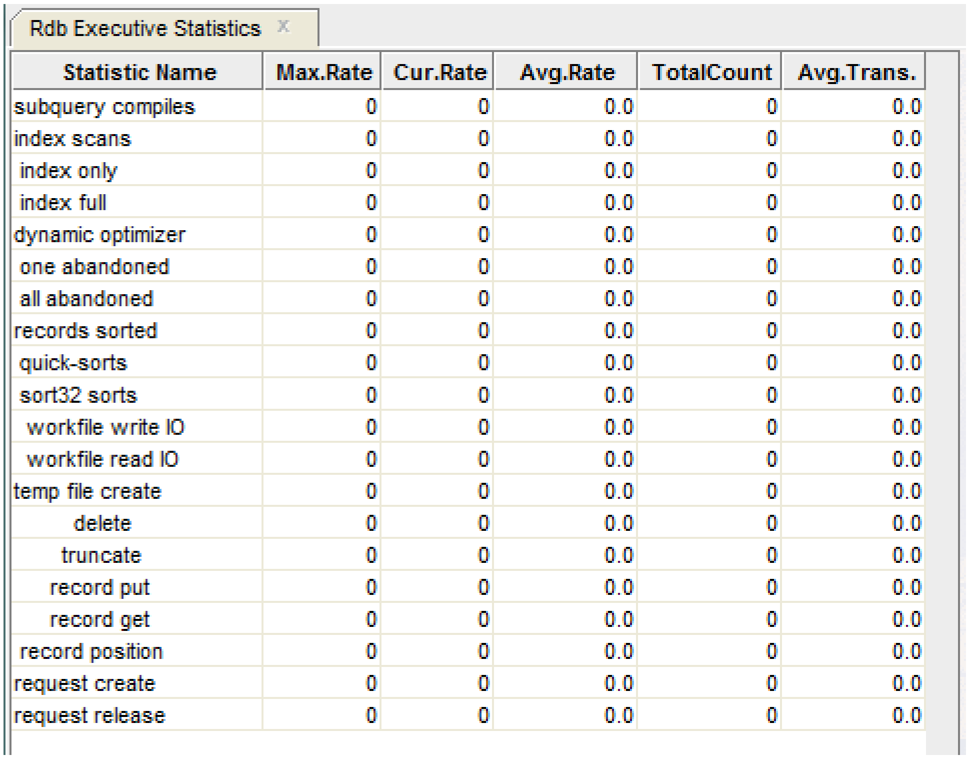
- ✓ Check the **page too full** statistic, which applies to read/write transactions. This statistic indicates that some data pages contain frequently accessed data that form chains of snapshot pages.

- ✓ Look at the **page in use** or **page conflict** statistics. These statistics indicate contention within the snapshot file. The problem might be that the snapshot file should be extended. The file might not extend itself because it can resolve the contention by doing extra work. However, you might want to extend the snapshot file manually to see if it helps alleviate the problem.

Contents

Chapter 7
Rdb Executive Statistics

This panel monitors activity within the Rdb Executive sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
subquery compiles	0	0	0.0	0	0.0
index scans	0	0	0.0	0	0.0
index only	0	0	0.0	0	0.0
index full	0	0	0.0	0	0.0
dynamic optimizer	0	0	0.0	0	0.0
one abandoned	0	0	0.0	0	0.0
all abandoned	0	0	0.0	0	0.0
records sorted	0	0	0.0	0	0.0
quick-sorts	0	0	0.0	0	0.0
sort32 sorts	0	0	0.0	0	0.0
workfile write IO	0	0	0.0	0	0.0
workfile read IO	0	0	0.0	0	0.0
temp file create	0	0	0.0	0	0.0
delete	0	0	0.0	0	0.0
truncate	0	0	0.0	0	0.0
record put	0	0	0.0	0	0.0
record get	0	0	0.0	0	0.0
record position	0	0	0.0	0	0.0
request create	0	0	0.0	0	0.0
request release	0	0	0.0	0	0.0

Figure 32 - Rdb Executive Statistics Pane

The following sections detail the information presented in this panel.

7.1.1 subquery compiles

Counts each time a table context is compiled, such as during compilation of each branch of a UNION, join or nested sub-query.

7.1.2 index scans

The number of times an index scan is initiated. This statistic accumulates for all index types and for all scan types including direct lookup.

7.1.3 index only

The number of scans that are started that are index only scans.

7.1.4 index full

The number of index scans started that do not have a lower or upper bound. In this case, the entire index will be scanned. If a key value range is provided by the query that includes all keys in the index, the entire index will be scanned. However, this statistic will not be incremented.

7.1.5 dynamic optimizer

The number of times the dynamic optimizer has been invoked.

7.1.6 one abandoned

The number of times that the dynamic optimizer abandons a background index scan because it is considered too costly.

7.1.7 all abandoned

The number of times that all background indexes have been abandoned and the dynamic optimizer has switched to sequential retrieval.

7.1.8 records sorted

The number of data items passed in to a sort routine (note that the elements being sorted may not be actual database rows).

7.1.9 quick−sorts

Sort operations that were handled entirely within an internally buffered quick sort routine. These are generally sorts of smaller cardinalities of simple sort keys.

7.1.10 sort32 sorts

Sort operations that were handled by the internal SORT32 interface.

7.1.11 workfile write IO

Write IO operations to SORT32 on−disk work files.

7.1.12 workfile read IO

Read IO operations from SORT32 on−disk work files.

7.1.13 temp file create

Creation of temporary relation work files.

7.1.14 delete

Deletion of temporary relation work files.

7.1.15 truncate

Rewind and truncate of temporary relation work files using RMS.

7.1.16 record put

Data written using RMS to temporary relation work files.

7.1.17 record get

Data read using RMS from temporary relation work files.

7.1.18 record position

Rewind or backspace operation of temporary relation work files using RMS.

7.1.19 request create

The loading of queries, stored procedures, and stored functions. Note that this count will also include queries executed by the SQL runtime environment.

7.1.20 request release

The release of loaded queries, stored procedures, and stored functions.

Notes:

- The **request create** and **request release** statistics closely correspond to the dynamic SQL DECLARE CURSOR and RELEASE statements. In general, interactive SQL will create and release a request for each interactive query, unless it is a cursor which will be retained until DISCONNECT. Compiled (SQL$MOD or SQL$PRE) applications will release queries only upon DISCONNECT (unless they are using dynamic SQL).
- The **request create** statistic is a better metric of the count of compiled queries than the statistic **subquery compiles.**

Contents

Chapter 8

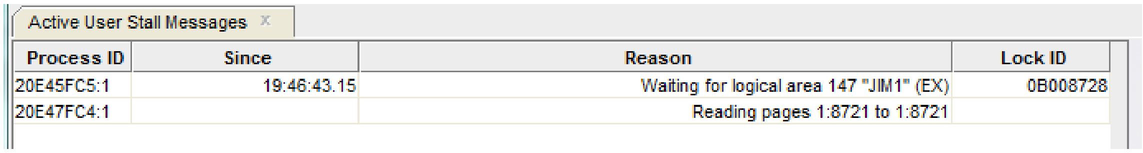
Process Information

The following process-centric statistics are available:

- Active User Stall Messages
- Process Accounting
- DBR Activity
- Lock Timeout History
- Lock Deadlock History

8.1 Active User Stall Messages

This panel shows a summary of database users' stall activity, even if the user is not actively stalled. A user stalls whenever Oracle Rdb issues a system call or long-duration operation on behalf of the user's process. For example, a stall occurs while a user waits for a lock or for completion of a physical disk read or write operation. A stall can also occur when switching AIJ journals and none are available.



Process ID	Since	Reason	Lock ID
20E45FC5:1	19:46:43.15	Waiting for logical area 147 "JIM1" (EX)	0B008728
20E47FC4:1		Reading pages 1:8721 to 1:8721	

Figure 33 - Active User Stall Messages Pane

The following sections detail the information presented in this panel.

8.1.1 Process ID

The process ID and the Oracle Rdb stream ID of the database user. Normally the stream ID will be one (1). However, if the user is attached to multiple databases or has explicitly detached and attached to different database sessions during the same image activation, the stream ID will uniquely identify the database session. A process ID with an “R” appended indicates a server for a remote process.

8.1.2 Since

The time at which the current stall started. If this field is blank, the process is not actively stalled.

8.1.3 Stall Reason

The reason for the stall. For example, “waiting for. . .“ messages indicate a stalled lock request, along with the requested lock mode. If the **Since** field is blank, this was the last known stall message for the process.

For more information on the stall messages, please see Stall Messages.

8.1.4 lock ID

The optional lock ID field is displayed only when the stall is the result of a lock request.

When other types of stalls occur, such as stalls due to I/O activity, the lock ID field is cleared from the panel.

When displayed, the lock ID field shows the lock identification of the resource that is stalled. You can use the lock identification number as input to the ***RMU Show Locks*** command to obtain information about processes that own, are blocking, or are waiting for locks.

Note that on the ***Active User Stall Messages*** panel, stalled locks that are no longer stalled are still displayed. Even if the displayed lock is no longer stalled, you can still attempt to display information on that stalled lock. However, the lock that was stalled is probably released. If the information is still there then this would seem an obvious candidate for further examination.

Notes:

✓ The ***Active User Stall Messages*** panel should be the first panel to examine when investigating reports of diminished application performance. Check if there are any stalls for logical areas, as this could be indicative that the application may need to modify the manner in which logical areas are readied.

✓ If there are frequent stalls for data pages, this may be indicative of insufficient buffers, or inappropriate specification of the “Asynchronous Pre-Fetch” feature or “Detected Asynchronous Pre-Fetch” feature attributes. If the database uses local buffers, review the PIO Statistics— Data Writes panel to verify that there are not too many buffer overflows occurring.

If a process is active and running on the same node as the attached RMU server, the process' PID displays and:

- The process information remains on that line until it detaches from the database.
- The process stall text remains on the panel until it is replaced by a new stall message.
- An active stall is identified by the stall starting time being displayed.
- If the stall is no longer active, the stall starting time is not displayed, but the stall message text remains displayed.

✓ The ***Active User Stall Messages*** panel has several advantages over the ***Stall Messages*** screen of the **RMU Show Statistics** utility.

The advantages are:

- The location of a process remains static; because it is fixed on a given page, it is always easy to locate.

- The process' last stall message (and lock ID, if applicable) are displayed, even if the process is not currently stalled; this is useful for identifying possible hot spots.

However, the ***Active User Stall Messages*** panel display does have the following disadvantages:

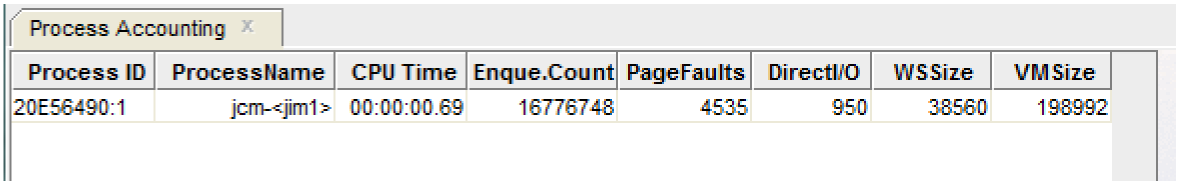
- It is difficult, but possible, to isolate the source of a potential deadlock or a long duration stall; the ***Stall Messages*** screen is more useful for this purpose.
- It is difficult, but possible, to isolate the set of actively stalled processes from the complete set of processes doing normal database accesses.

Contents

8.2 Process Accounting

The ***Process Accounting*** panel provides continuously updated accounting information about local processes. This panel is an alternative to the OpenVMS SHOW PROCESS /CONTINUOUS utility. The panel provides equivalent information, except that all active database processes on a specific node can be monitored at the same time.

The ***Process Accounting*** panel shows direct operating system accounting information, thereby enabling a database administrator to evaluate the system resources used by database processes. The values in the ***Process Accounting*** panel are for all process activity, not just the activity that occurs while in the database. Therefore, this panel is useful for monitoring the complete application behavior.



Process ID	ProcessName	CPU Time	Enque.Count	PageFaults	DirectI/O	WSSize	VMSize
20E56490:1	jcm-<jim1>	00:00:00.69	16776748	4535	950	38560	198992

Figure 34 - Process Accounting Pane

Note:

✓ Use the horizontal scroll bar at the bottom of this panel to scroll through columns.

The following sections detail the information presented in this panel.

8.2.1 Process ID

The process ID for the current process, assigned by the operating system, and the stream ID, assigned by the database. If “D” is appended to the process ID, the process is being recovered by a Database Recovery process (DBR). If “R” is appended to the process ID, the process is a server for a remote process.

Note:

- ✓ If the process has more than one active database session, only the first active stream ID will be displayed because the process accounting information is the same for all active database sessions.

8.2.2 Process Name

The name of the process.

8.2.3 CPUtime

The total accumulated CPU time since the process was logged in.

8.2.4 Enque Count

The remaining number of locks the process may request. This field should not approach zero. If it does approach zero, the ENQLM value should be raised for the process.

8.2.5 Page Faults

The total accumulated number of page faults.

8.2.6 NumDio

The total accumulated count of the direct I/O operations.

8.2.7 WSSize

The total accumulated number of active pages (512-byte pages) in the working set for the process.

8.2.8 VMSize

The total accumulated number of pages (512-byte pages) of virtual memory allocated by the process. This value includes the size of the database global section.

8.2.9 User Name

The name of the user (used when logging in to the system).

8.2.10 Image Name

The name of the image file currently being executed by the process.

8.2.11 State

The current process state, which will be one of the following keywords:

Keyword	**Description**
CEF	Common event flag wait
COM	Computable
COMO	Computable, out of balance set
CUR	Current process
COLPG	Collided page wait
FPG	Free page wait
HIB	Hibernate wait
HIBO	Hibernate wait, out of balance set
LEF	Local event flag wait

LEFO	Local event flag wait, out of balance set
MWAIT	Mutex and miscellaneous resource wait
PFW	Page fault wait
SUSP	Suspended
SUSPO	Suspended, out of balance set

8.2.12 PGfCnt

The remaining number of pages in the paging file the process may request. This field should not approach zero. If it does approach zero, the PGFLQUOTA value should be raised for the process.

8.2.13 DioCnt

The remaining number of direct I/O operations the process may request. This field should not approach zero. If it does approach zero, the DIOLM value should be raised for the process.

8.2.14 NumBio

The total accumulated count of the buffered I/O operations.

8.2.15 BioCnt

The remaining number of buffered I/O operations the process may request. This field should not approach zero. If it does approach zero, the BIOLM value should be raised for this process.

Notes:

- This panel shows dynamically changing process information only. That is, quotas and other information that are fixed for each process are not displayed because that information can be obtained in other ways. The ***Process Accounting*** panel has brief and full modes that control the amount of information displayed for each active database process.

- Note that information on the ***Process Accounting*** panel cannot be reset.

- This panel provides continuously updated OpenVMS accounting information about local processes in a cluster environment. This panel is an alternative to the OpenVMS **SHOW PROCESS /CONTINUOUS** utility. The panel provides equivalent information, except that all active database processes on a specific node can be monitored at the same time.

- The ***Process Accounting*** panel does not provide information about database processes on other nodes.
  This panel displays operating system accounting information, thereby enabling a database administrator to evaluate the system resources used by database processes. The values in the ***Process Accounting*** panel are for all process activity, not just the activity that occurs while in the database. Therefore, this panel is useful for monitoring the complete application behavior.

- This panel displays dynamically changing process information only. That is, quotas and other information that are fixed for each process are not displayed because that information can be obtained in other ways.

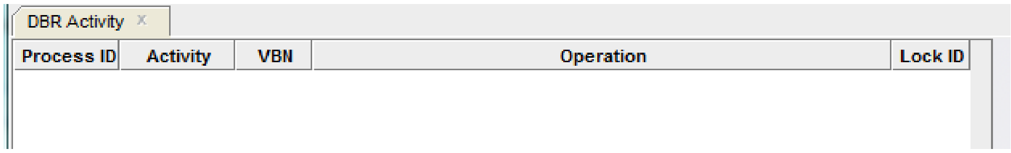
Contents

8.3 DBR Activity

This table displays one line of information for each Database Recovery process (DBR) active on the node. If there is no active DBR process, the table is empty.

If the ***DBR Activity*** panel is not used, the only method available to users to determine if the DBR process is running is to use the **RMU Show Users** utility, which only indicates that the DBR process is running; it does not indicate what type of progress DBR is making in the recovery operation.

The ***DBR Activity*** panel does not identify which user process is being recovered. This information can be obtained from the Active User Stall Messages panel.



Process ID	Activity	VBN	Operation	Lock ID

Figure 35 - DBR Activity Pane

The following sections detail the information presented in this panel.

8.3.1 Process ID

The process identifier of the database recovery process (DBR). The process ID contains the suffix “D”.

8.3.2 Activity

This field contains a description of the recovery activity being performed by the Database Recovery process (DBR).

This field may not change for a DBR process running on another node of the cluster.

The following table describes the various DBR process activities:

Activity	**Description**
Activation	The DBR process is being activated by the monitor.
Database Attach	The DBR process is attaching to the database.
AIJ Recovery	The DBR process is recovering any pending AIJ operations.
Root Update	The DBR process is recovering any pending database root information updates.
GB Recovery	The DBR process is recovering any pending global buffer transactions.
Recovery Setup	The DBR process is initializing its recovery context information.
Transaction Redo	The DBR process is redoing committed transactions that have not yet been flushed to the database.

Transaction Undo	The DBR process is undoing uncommitted transactions that have already been flushed to the database.
Buffer Flush	The DBR process is flushing its cache buffers to the database.
Database Detach	The DBR process is detaching from the database and terminating the image.

8.3.3 VBN

This column identifies the current block number of the AIJ journal or RUJ journal being accessed. If the block number displayed is for the AIJ journal, the block number will normally prefixed with the AIJ sequence number, so it is easy to identify which AIJ journal is being backed up.

The VBN is for the RUJ journal during the “Transaction UNDO” activity phase.

The VBN is for the AIJ journal during the “Transaction REDO” activity phase.

This field may not change for a database recovery operation on another node of the cluster.

8.3.4 Operation

This column identifies the activity-specific operation being performed by the Database Recovery process (DBR). This column contains messages similar to those displayed by the **RMU Show Statistics** utility ***Stall Messages*** screen.

This field may not change for a database recovery operation on another node of the cluster.

8.3.5 Lock.ID

This column identifies any lock the DBR process may be trying to acquire. This lock is typically a page lock.

Note:

- ✓ The DBR process does not acquire record locks.

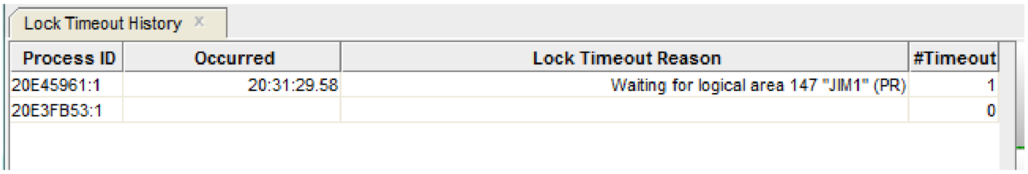
Contents

8.4 Lock Timeout History

For each active process on the current node, the ***Lock Timeout History*** panel shows the process ID, the time that the process most recently timed out while waiting for a lock, the reason for the most recent timeout experienced by the process, and the number of timeouts experienced by the process since attaching to the database.

The ***Stall Messages*** screen of the **RMU Show Statistics** utility provides active stall information, but when a process times out while waiting for a lock, the stall is terminated and the information is no longer available on the ***Stall Messages*** screen.

By first examining the ***Lock Deadlock History*** panel and then the ***Lock Timeout History*** panel, the DBA can determine which records represent database hot spots and sources of contention.



Process ID	Occurred	Lock Timeout Reason	#Timeout
20E45961:1	20:31:29.58	Waiting for logical area 147 "JIM1" (PR)	1
20E3FB53:1			0

Figure 36 - Lock Timeout History Pane

The following sections detail the information presented in this panel.

8.4.1 Process ID

The process ID and the Oracle Rdb stream ID of the database user. Normally the stream ID will be one (1). However, if the user is attached to multiple databases or has explicitly detached and attached to different database sessions during the same image activation, the stream ID will uniquely identify the database session. A process ID with an “R” appended indicates a server for a remote process.

8.4.2 Occurred

This field displays the time at which the lock timeout occurred.

8.4.3 Lock Timeout Reason

The reason for the lock timeout. For example, “waiting for. . .“ messages indicates a stalled lock request, along with the requested lock mode, that timed out on the identified lock resource. See Stall Messages for descriptions of the reason text that may be displayed in this column.

8.4.4 #Timeouts

This field displays the number of lock timeout operations that have occurred to this process.

Note that his field does **not** represent the number of lock timeouts that have occurred for the displayed resource in the **Lock Timeout Reason** field.

Note:

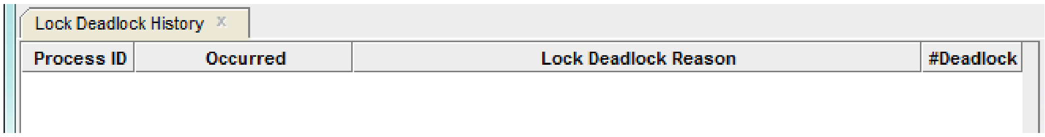
✓ Lock timeouts are considerably rarer than lock deadlocks, so they should be reviewed more closely.

See Stall Messages for the Lock Timeout Reason text messages.

Contents

8.5 Lock Deadlock History

For each active process on the current node, the ***Lock Deadlock History*** panel shows the process ID, the time that the process was most recently involved in a deadlock, the reason for the most recent deadlock, and the number of deadlocks the process has encountered since attaching to the database.



Process ID	Occurred	Lock Deadlock Reason	#Deadlock

Figure 37 - Lock Deadlock History Pane

By using the ***Lock Deadlock History*** panel to examine the most recent deadlocks, a DBA can more easily identify potential database hot spots and application bottlenecks.

Looking at a full panel of deadlock information can help the DBA differentiate between frequent and infrequent problems and correlate common causes of the problems.

In general, record deadlocks are undesirable and the cause of such deadlocks should be discovered and fixed. Page deadlocks are not always indicative of an application problem, but they need to be analyzed as all deadlocks represent performance problems.

The table shows the following information:

8.5.1 Process

The process ID and the Oracle Rdb stream ID of the database user. Normally the stream ID will be one (1). However, if the user is attached to multiple databases or has explicitly detached and attached to different database sessions during the same image activation, the stream ID will uniquely identify the database session. A process ID with an R appended indicates a server for a remote process.

8.5.2 Occurred

This field displays the time at which the lock deadlock occurred.

8.5.3 Lock Deadlock Reason

The reason for the lock deadlock. For example, "waiting for. . ." messages indicates a stalled lock request, along with the requested lock mode, that deadlocked on the identified lock resource.

8.5.4 #Deadlocks

This field displays the number of lock deadlock operations that have occurred to this process. This field does not represent the number of lock deadlock that have occurred for the displayed resource in the "reason" field.

See Stall Messages for the Lock Timeout Reason text messages.

Contents

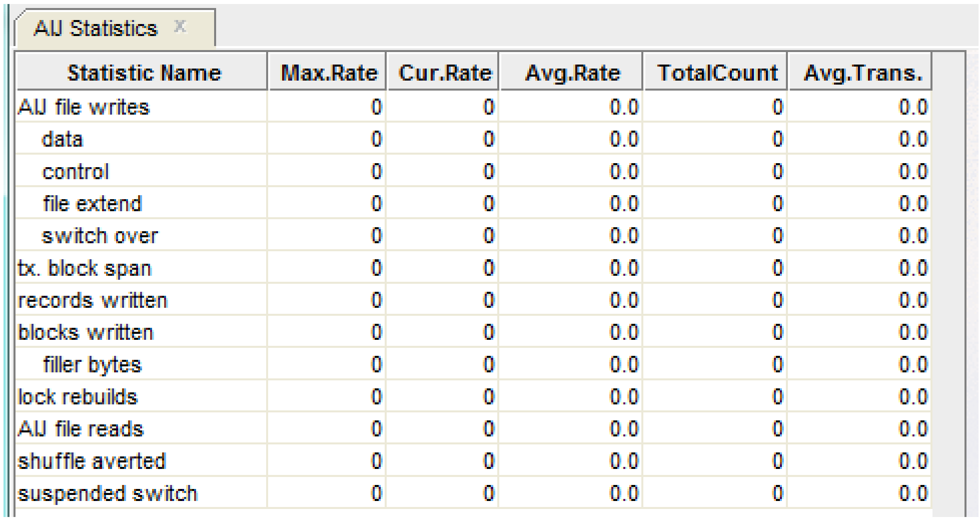
Chapter 9
Journaling Information

The following journaling information and statistics are available:

- AIJ Statistics
- Group Commit Statistics
- AIJ Journal State
- AIJ Journal Information
- ALS Statistics
- RUJ Statistics
- Checkpoint Statistics

9.1 AIJ Statistics

This panel monitors both logical and physical after-image journaling activity.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
AIJ file writes	0	0	0.0	0	0.0
data	0	0	0.0	0	0.0
control	0	0	0.0	0	0.0
file extend	0	0	0.0	0	0.0
switch over	0	0	0.0	0	0.0
tx. block span	0	0	0.0	0	0.0
records written	0	0	0.0	0	0.0
blocks written	0	0	0.0	0	0.0
filler bytes	0	0	0.0	0	0.0
lock rebuilds	0	0	0.0	0	0.0
AIJ file reads	0	0	0.0	0	0.0
shuffle averted	0	0	0.0	0	0.0
suspended switch	0	0	0.0	0	0.0

Figure 38 - AIJ Statistics Pane

The following sections detail the information presented in this panel.

9.1.1 AIJ file writes

The total number of write I/Os (queued I/O requests) issued to the database after-image journal (.AIJ) file. The **data**, **control**, **file extend**, and **switch over** counts further subdivide the statistic.

This statistic field includes both synchronous and asynchronous I/O write requests.

9.1.2 data

This field gives the number of write-I/Os to the database after-image journal file that contained only data records.

9.1.3 control

The number of write I/Os to the database after-image journal file that contained OPEN, COMMIT, ROLLBACK, and checkpoint records. These writes may also include data records.

9.1.4 file extend

The number of write I/Os issued to the database after-image journal file to initialize new disk blocks when the file is extended.

9.1.5 switch over

The number of times AIJ switch-over has occurred. That is, the number of times journaling has switched from one journal file to another.

9.1.6 tx block span

The number of blocks each transaction spans in the AIJ journal.

9.1.7 records written

The number of logical after-image journal records written to the after-image journal file. Because many logical records can be buffered into a single write I/O, this number is usually much larger than the number of after-image journal writes.

No count is kept for the number of logical AIJ records read by the Database Recovery process (DBR).

9.1.8 blocks written

The number of disk blocks written to the after-image journal file.

Because many disk blocks may be written with a single write I/O, this number may be larger than the number of after-image journal writes. In addition, the same disk block can be written more than once, because it might not have been completely full the first time it was written.

This statistic field includes both synchronous and asynchronous I/O write requests.

9.1.9 filler bytes

The number of filler bytes needed to pad AIJ records to a disk block boundary. Filler bytes are bytes that contain nothing, that is, memory allocated but not used. Filler bytes are necessary to ensure the readability of the after-image journal file in the event of a system failure during a write I/O operation.

9.1.10 lock rebuilds

The number of times that the after-image journal lock value block had to be rebuilt. The AIJ lock value block contains the current end-of-file information about the after-image journal file. The lock value block can be lost when there is a cluster configuration change, or when a user process holding the lock terminates abnormally, with a Ctrl-Y/STOP, for example.

9.1.11 AIJ file reads

The number of read-I/Os executed during lock rebuilds. To rebuild the lock value block, the after-image journal file is read backwards to find the last record in the file.

This statistic field includes both synchronous and asynchronous I/O read requests.

9.1.12 shuffle averted

The number of times the group commit buffer has had a shuffle operation averted. The shuffle operation can be averted for a number of reasons, including setting the RDM$BIND_AIJ_SHUFFLE_DISABLED logical to the value “0”.

9.1.13 suspended switch

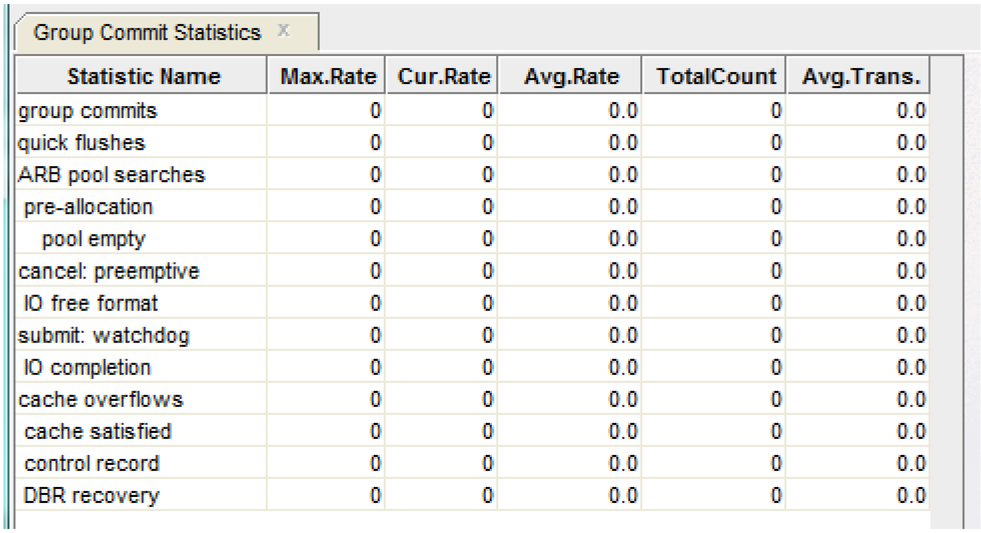
The number of times the AIJ switch-over operation enters the suspended state.

A database enters the “AIJ suspended” state when the AIJ switch-over operation cannot complete because there are no available AIJ journals. During this state, the DBA can add new AIJ journals or perform database backups, but all other AIJ-related activities are temporarily suspended until an AIJ journal becomes available.

Contents

9.2 Group Commit Statistics

This panel monitors transaction group commit activity.



Group Commit Statistics

Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
group commits	0	0	0.0	0	0.0
quick flushes	0	0	0.0	0	0.0
ARB pool searches	0	0	0.0	0	0.0
pre-allocation	0	0	0.0	0	0.0
pool empty	0	0	0.0	0	0.0
cancel: preemptive	0	0	0.0	0	0.0
IO free format	0	0	0.0	0	0.0
submit: watchdog	0	0	0.0	0	0.0
IO completion	0	0	0.0	0	0.0
cache overflows	0	0	0.0	0	0.0
cache satisfied	0	0	0.0	0	0.0
control record	0	0	0.0	0	0.0
DBR recovery	0	0	0.0	0	0.0

Figure 40 - Group Commit Statistics Pane

A “group commit” is an operation whereby several transactions writing to the AIJ journal at the same approximate time are “grouped” together for efficiency, and written using a single I/O operation. This improves the AIJ performance since the expense of the I/O operation is amortized across many transactions, with a corresponding increase of transaction throughput.

Ideally, the AIJ Log Server (ALS) process is the group commit writer. However, when the ALS process is not used, one of the user processes is arbitrarily selected as the group commit writer.

The following sections detail the information presented in this panel.

9.2.1 group commits

The number of group commit operations performed by the AIJ Log Server (ALS) process. Dividing the number of transactions by the number of groups gives the average group size. That is, the inverse of the transaction average display.

9.2.2 quick flushes

The number of times users requested their .AIJ records be flushed to the after-image journal and those records had already been flushed by the cooperative actions of another user (group commit).

9.2.3 ARB pool searches

The number of times the global after-image journal request block pool was searched. AIJ records are globally buffered using AIJ Request Blocks (ARB).

9.2.4 pre-allocation

The number of ARBs that are able to be pre-allocated, resulting in reduced transaction commit or rollback processing (increased throughput).

9.2.5 pool empty

The number of times that the global after-image journal request block (ARB) pool was found to be empty. In this event, the active user must force a write operation to the after-image journal file to free an ARB.

9.2.6 cancel: preemptive

The number of times the background group-commit operation was canceled because there was no additional data to be formatted.

9.2.7 IO free format

The number of times the foreground group-commit operation was canceled because there was no additional data to be formatted.

9.2.8 submit: watchdog

The number of times the ALS process attempted to perform a group-commit operation in anticipation of new work but found none.

9.2.9 io completion

The number of times the group-commit formatting was completed because the pending asynchronous I/O operation for the previous group-commit operation completed.

9.2.10 cache overflows

The number of times the 127-block .AIJ cache has overflowed.

9.2.11 cache satisfied

The number of times the group-commit formatting was completed because the cache-size constraints have been reached.

9.2.12 control record

The number of times the group-commit formatting was completed because of encountering a control record for which a process was waiting (and there was no I/O in progress).

9.2.13 DBR recovery

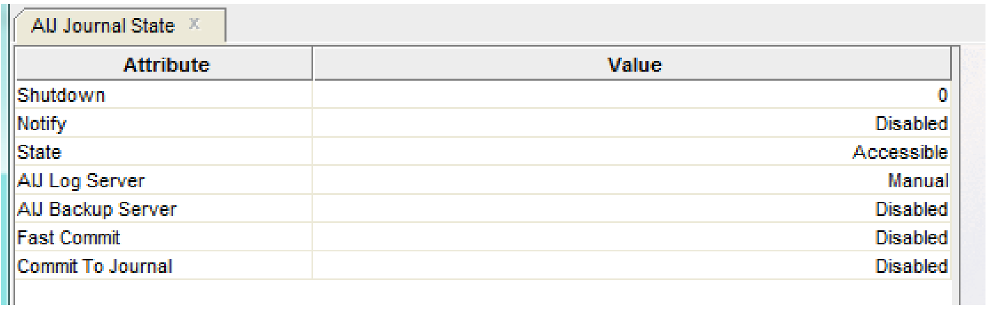
The number of times the group-commit formatting was completed because the current process is Database Recovery process (DBR) and the database is frozen. Ideally, this field should always be “0”.

Contents

9.3 AIJ Journal State

This panel contains information relating to AIJ journaling in general.

The information on the ***AIJ Journal State*** panel pertains to the node the connected RMU server is running on only.



Attribute	Value
Shutdown	0
Notify	Disabled
State	Accessible
AIJ Log Server	Manual
AIJ Backup Server	Disabled
Fast Commit	Disabled
Commit To Journal	Disabled

Figure 41 - AIJ Journal State Pane

The following sections detail the information presented in this panel.

9.3.1 Journaling

Indicates whether or not AIJ journaling is available. The valid keywords are:

Keyword	**Description**
ENABLED	after-image journaling is enabled

DISABLED	after-image journaling is disabled

9.3.2 Shutdown

If highlighted, indicates that an AIJ switch-over is in progress and identifies the number of minutes remaining until the after-image journaling system shutdown is complete.

If no switch-over is in progress, the default shutdown time, in minutes, is displayed.

Note:

✓ This field displays shutdown information for the node the connected RMU server is running on only.

9.3.3 Notify

Indicates whether or not the system operator notification facility is active.

This field does not identify the specific operator classes that are enabled. The valid keywords are:

Keyword	Description
ENABLED	operator notification is enabled
DISABLED	operator notification is disabled

9.3.4 State

The current state of the AIJ journals. The valid keywords are:

Keyword	Description

ACCESSIBLE	all after-image journals are available
INACCESSIBLE	all after-image journals are unavailable

9.3.5 AIJ Log Server

The state of the AIJ Log Server (ALS) process. The valid keywords are:

Keyword	**Description**
AUTOMATIC	The ALS process is started automatically at database open time, but is not currently running
MANUAL	The ALS process must be manually started by the DBA, and it is not currently running
RUNNING	The ALS process is currently active

9.3.6 AIJ Backup Server

Indicates whether or not the AIJ Backup Server (ABS) is enabled. The valid keywords are:

Keyword	**Description**
ENABLED	The ABS is enabled
DISABLED	The ABS is not used
SUSPENDED	The ABS is enabled but currently suspended. This usually occurs when using the “Hot Standby“

	feature
DENIED	The ABS cannot be used because one or more AIJ journals have been over-written or are currently inaccessible

---

Note:

- ✓ The AIJ backup server is not always invoked even though it is enabled; output from the RMU Dump Header command provides more information in those situations where the AIJ backup server is enabled but not invoked.

9.3.7 Fast Commit

Indicates whether or not the “Fast Commit“ feature is available. Note that this field does not indicate whether or not the Fast Commit feature is active. The Fast Commit options are not displayed. The valid keywords are:

Keyword	**Description**
ENABLED	the “Fast Commit“ feature is enabled
DISABLED	the “Fast Commit“ feature is not used

9.3.8 Commit To Journal

Indicates whether or not the “Commit To Journal optimization“(CTJ) feature is available. Note that this field does not indicate whether or not the commit to journal optimization feature is active. The valid keywords are:

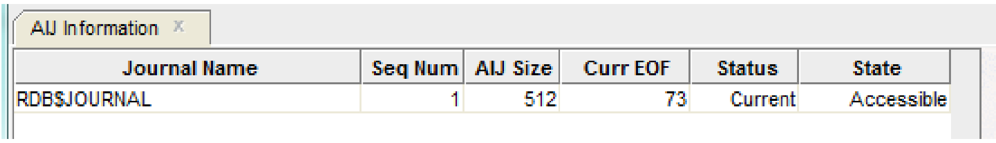
Keyword	**Description**

ENABLED	the “Commit To Journal optimization “ feature is enabled
DISABLED	the “Commit To Journal optimization “ feature is not used

Contents

9.4 AIJ Journal Information

The ***AIJ Journal Information*** panel provides online information about all of a database's after-image journals on the current node, that is, the node on which the connected RMU server is running. Most of the information displayed on the panel is obtained in real time, which means that the panel is automatically updated as AIJ database parameters are modified, or as AIJ operations such as a backup or journal switch-over are performed.



Journal Name	Seq Num	AIJ Size	Curr EOF	Status	State
RDB$JOURNAL	1	512	73	Current	Accessible

Figure 42 - AIJ Journal Information Pane

The following sections detail the information presented in this panel.

9.4.1 image journal name

Identifies the name of the after-image journal (this is not the journal filename). If the journal is not allocated, identifies the number of the available journal slot. The current journal is highlighted for quick identification.

9.4.2 Seq Num

Identifies the sequence number of the journal. If the journal is unused, “Unused” is displayed.

9.4.3 AIJ size

Identifies the size, in blocks, of the current after-image journal. This value typically corresponds to the after-image journal allocation size for fixed-size journals. If the physical end-of-file has not yet been established, the default allocation size, in blocks, is displayed.

9.4.4 Current EOF

Identifies the block number of the current end-of-file of the current journal; this information is only displayed for the current journal because it is meaningless for other journals. This value corresponds to the location in the journal being written with data. If the journal is current but the logical end-of-file has not yet been established,

“Unknown” is displayed. If the journal is not current, “Empty” is displayed.

9.4.5 Status

This field displays the following keywords:

Keyword	**Description**
Current	the journal is currently being written to
Written	the journal was written to in the past
Latent	the journal has not yet been written to
BACKUP NEEDED	A journal that has been written to and is not current should be backed up, this message identifies when a journal needs to be backed up

9.4.6 State

This field displays the following keywords:

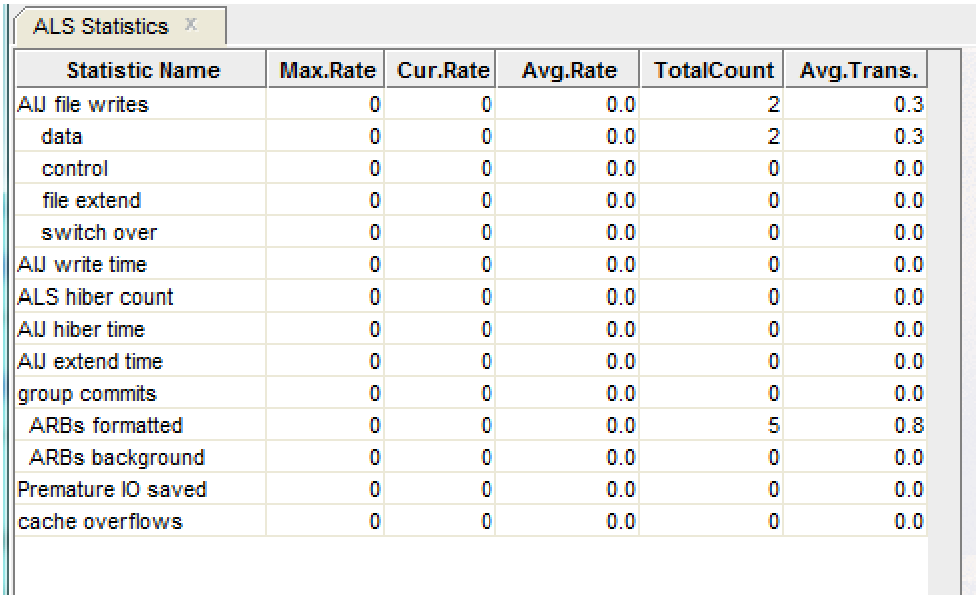
Keyword	**Description**
Backing Up	the journal is in the process of being backed up
Overwritten	the journal has been overwritten

Accessible	the journal can be written to
Inaccessible	the journal cannot be written to

Contents

9.5 *ALS Statistics*

This panel contains information specific to the operation of the **AIJ Log Server** (ALS) process.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
AIJ file writes	0	0	0.0	2	0.3
data	0	0	0.0	2	0.3
control	0	0	0.0	0	0.0
file extend	0	0	0.0	0	0.0
switch over	0	0	0.0	0	0.0
AIJ write time	0	0	0.0	0	0.0
ALS hiber count	0	0	0.0	0	0.0
AIJ hiber time	0	0	0.0	0	0.0
AIJ extend time	0	0	0.0	0	0.0
group commits	0	0	0.0	0	0.0
ARBs formatted	0	0	0.0	5	0.8
ARBs background	0	0	0.0	0	0.0
Premature IO saved	0	0	0.0	0	0.0
cache overflows	0	0	0.0	0	0.0

Figure 43 - ALS Statistics Pane

The following sections detail the information presented in this panel.

9.5.1 AIJ file writes

This field gives the total number of write-I/Os (queued I/O requests) issued to the database after-image journal (.AIJ) file. The **data**, **control**, **file extend**, and **switch over** counts further subdivide the statistic.

This statistic field includes both synchronous and asynchronous I/O write requests.

9.5.2 data

The number of write-I/Os to the database after-image journal file that contained only data records.

9.5.3 control

The number of write-I/Os to the database after-image journal file that contained OPEN, COMMIT, ROLLBACK, and checkpoint records. These writes may also include data records.

9.5.4 file extend

The number of write-I/Os issued to the database after-image journal file to initialize new disk blocks when the file is extended.

9.5.5 switch over

The number of times AIJ switch-over has occurred. In other words, this field indicates the number of times journaling has switched from one journal file to another.

9.5.6 AIJ Write Time

The time in hundredths of a second spent writing to the after-image journal.

9.5.7 ALS Hiber Count

The number of times the AIJ Log Server process spent hibernating while waiting for more work.

9.5.8 AIJ Hiber Time

The amount of time in hundredths of a second that processes have spent hibernating while the AIJ log server processes their requests.

9.5.9 AIJ Extend Time

The time in hundredths of a second spent extending the after-image journal. An excessively high number often indicates a full disk or disk fragmentation, or may be an indication of the need for a larger allocation.

Use the JOURNAL ALLOCATION IS clause of the SQLALTER DATABASE statement to specify a larger allocation.

9.5.10 Group Commits

The number of group commit operations performed by the ALS process. Dividing the number of transactions by the number of groups gives the average group size, which is the inverse of the transaction average display.

9.5.11 ARBs formatted

The number of AIJ Request Blocks that were formatted by the ALS process.

9.5.12 ARBs background

The number of AIJ Request Blocks that were formatted while asynchronous I/O was being performed to the AIJ journal.

9.5.13 Premature io saved

The number of I/Os saved by waiting for more work.

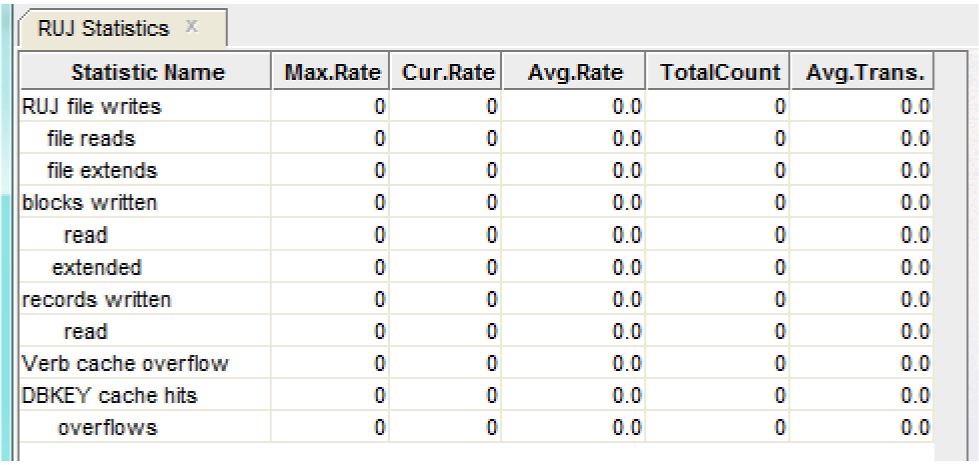
9.5.14 Cache Overflows

The number of times the 127-block .AIJ cache has overflowed.

Contents

9.6 *RUJ Statistics*

This panel contains information for all active update transactions on the node of the connected RMU Server.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
RUJ file writes	0	0	0.0	0	0.0
file reads	0	0	0.0	0	0.0
file extends	0	0	0.0	0	0.0
blocks written	0	0	0.0	0	0.0
read	0	0	0.0	0	0.0
extended	0	0	0.0	0	0.0
records written	0	0	0.0	0	0.0
read	0	0	0.0	0	0.0
Verb cache overflow	0	0	0.0	0	0.0
DBKEY cache hits	0	0	0.0	0	0.0
overflows	0	0	0.0	0	0.0

Figure 44 - RUJ Statistics Pane

The following sections detail the information presented in this panel.

9.6.1 RUJ file writes

The number of write I/Os (queued I/O requests) 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.

9.6.2 file reads

The number of read I/Os (queued I/O requests) 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.

9.6.3 file extends

The total number of times the recovery-unit journal files have been extended.

9.6.4 blocks written

The number of disk blocks written to the recovery-unit journal file.

Because many disk blocks may be written with a single write I/O, this number may be larger than the number of recovery-unit journal writes. In addition, the same disk block can be written more than once, because it might not have been completely full the first time it was written.

This statistic field includes both synchronous and asynchronous I/O write requests.

9.6.5 read

The total number of 512-byte blocks that have been read from the recovery-unit journals (.RUJ) on this node.

This statistic field includes both synchronous and asynchronous I/O read requests.

9.6.6 extended

The total number of 512-byte blocks that have been written to the recovery-unit journals (.RUJ) on this node during file extensions.

9.6.7 records written

The number of journal records written to the recovery-unit journal (.RUJ) file.

9.6.8 read

The number of journal records read from the recovery-unit journal (.RUJ) file.

9.6.9 Verb cache overflows

The number of times the recovery-unit data cache has overflowed, causing a premature synchronous write I/O operation. Overflowing the data cache indicates update-intensive transactions.

9.6.10 DBKEY cache hits

The number of times the same DBKEY (row) was modified within the same transaction and found in the DBKEY cache.

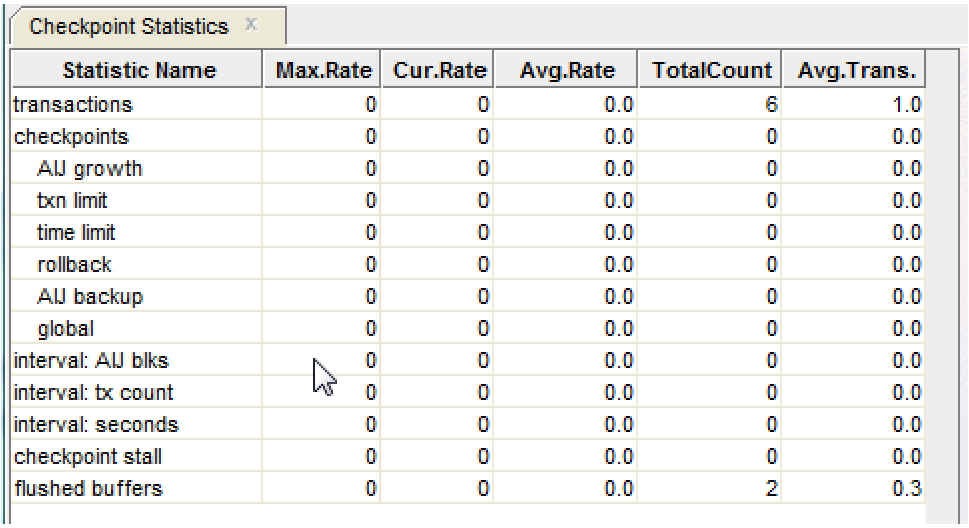
9.6.11 overflows

The number of times the DBKEY cache ran out of space and new space had to be allocated.

Contents

9.7 *Checkpoint Statistics*

This panel shows transaction and checkpoint activity. This panel is useful only if your database has enabled the AIJ Fast Commit feature.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	6	1.0
checkpoints	0	0	0.0	0	0.0
AIJ growth	0	0	0.0	0	0.0
txn limit	0	0	0.0	0	0.0
time limit	0	0	0.0	0	0.0
rollback	0	0	0.0	0	0.0
AIJ backup	0	0	0.0	0	0.0
global	0	0	0.0	0	0.0
interval: AIJ blks	0	0	0.0	0	0.0
interval: tx count	0	0	0.0	0	0.0
interval: seconds	0	0	0.0	0	0.0
checkpoint stall	0	0	0.0	0	0.0
flushed buffers	0	0	0.0	2	0.3

Figure 45 - Checkpoint Statistics Pane

Statistics are displayed for all processes on the node for a particular database. The total number of checkpoints, with a breakdown of the reasons for checkpointing, is displayed.

The sum of all checkpoint intervals is also displayed, using three different metrics. You can use this information to compute the average interval between checkpoints, allowing you to decide if a checkpointing interval should be adjusted, and by how much.

Some of the columns provided by the ***Checkpoint Statistics*** panel measure events on a per-second or per transaction basis. These columns are useful for measuring frequently occurring events such as I/O operations. Because checkpointing typically occurs at a slower rate, you will find the most meaningful checkpointing information in the “total count” column of the ***Checkpoint Statistics*** panel.

Oracle Corporation recommends that you refer to this column when you use checkpoint statistics to determine optimal checkpoint intervals.

The following sections detail the information presented in this panel.

9.7.1 transactions

The total number of transactions (both read/write and read-only transactions). This statistic represents all transactions (committed or rolled back) completed by all users of the database, including transactions that read from the database as well as transactions that modify the database.

9.7.2 checkpoints

The total number of checkpoints. The total checkpoints category is further broken down into categories of reasons for checkpointing. The statistics for these categories are included in the **AIJ growth**, **txn limit**, **time limit**, **rollback**, **AIJ backup**, and **global** sub-fields.

This field does not include the initial checkpoint that is performed when users attach to the database.

Note:

- ✓ There may be more reasons for checkpoints than there are total checkpoints. For example, you might have a total count of 100 for checkpoints, but when you add the number of checkpoint reasons (**AIJ growth**, **txn limit**, **time limit**, **rollback**, **AIJ backup**, and **global**), the total could be greater than 100. This occurs because a single checkpoint may be triggered by more than one event. For example, a checkpoint may occur because of time and AIJ file growth. Although the **TotalCount** columns for the **interval: seconds** and **interval: AIJ blks** fields are both incremented by one, the **TotalCount** column for **checkpoints** is only incremented by one.

9.7.3 AIJ growth

Checkpoint for all processes due to the after-image journal growth checkpoint limit.

9.7.4 txn limit

Checkpoints for all processes due to the logical-defined transaction limit.

9.7.5 time limit

Checkpoints for all processes due to the time interval checkpoint limit.

9.7.6 rollback

Checkpoints automatically triggered by rollback of transactions that updated the database.

9.7.7 AIJ backup

System-generated checkpoints due to periodic backups to tape of the after-image journal by the AIJ backup utility.

9.7.8 global

The number of system-wide checkpoints, issued from an RMU Checkpoint, RMU Backup, or RMU Backup After Journal command.

9.7.9 interval: AIJ blks

This field displays the sum of the intervals between checkpoints due to AIJ growth in block checkpoints for all processes. For example, if Process 1 checkpoints at virtual block number (VBN) 100, then checkpoints again at VBN 250, the AIJ block interval category is incremented by 150. If Process 2 checkpoints at VBN 125, then checkpoints again at VBN 200, the AIJ block interval is incremented by an additional 75.

Statistics for the other two interval categories are displayed in the **interval: tx count** and **interval: seconds** fields.

If CHECKPOINT INTERVAL IS 1000 BLOCKS is specified with the SQL ALTER DATABASE statement, each process checkpoints when the .AIJ file has grown 1000 blocks since the process' last checkpoint.

Keep in mind that checkpointing influences recovery time. The main reason to consult checkpoint statistics is to find the average interval per checkpoint. You can use the information in the total count column to compute this average. For each category of checkpoint reason, use the average interval per checkpoint to help you decide if a checkpointing interval should be adjusted, and by how much.

If most of the checkpoints for a database are triggered by a particular checkpoint limit, that limit may be set too high, or the other two limits may be set too low. You can determine the average interval per checkpoint for each type of checkpoint limit. After you have this information, you can reset the limits so that each type of checkpoint limit triggers approximately the same number of checkpoints, which results in optimal performance.

To compute the average interval in AIJ blocks, divide the total count for the AIJ block interval by the total number of checkpoints minus the number caused by AIJ backups. Although checkpoints caused by AIJ backups are counted in the total number of checkpoints, they are not counted in the total of AIJ block intervals. If the total count of AIJ block intervals is 70000, the total count of checkpoints is 100, and the number of checkpoints caused by AIJ backups is 1, then the average AIJ block interval is 707:

$$70000 / (100 - 1) = 707$$

The description of the **interval: tx count** field explains how to determine the average interval for transaction checkpoints.

The description of the **interval: seconds** field explains how to determine the average interval for time checkpoints.

9.7.10 interval: tx count

This field displays the sum of the intervals between checkpoints due to the transactions count checkpoint for all processes. For example, if Process 1 checkpoints after 20 transactions, the transactions count category is incremented by 20. If Process 2 checkpoints after 30 transactions, the transactions count category is incremented by an additional 30. Statistics for the other two interval categories are displayed in the **interval: AIJ blks** and **interval: seconds** fields.

The transactions limit for checkpoints is determined by the setting of the RDM$BIND_CKPT_TRANS_INTERVAL logical name. If RDM$BIND_CKPT_TRANS_INTERVAL is defined as a system logical set to 10, each process will checkpoint after 10 transactions unless a user redefines the logical to a different value. That is, if a user defines

RDM$BIND_CKPT_TRANS_INTERVAL as a process logical and sets a value of 5, that user will checkpoint after 5transactions.

Keep in mind that checkpointing influences recovery time. The main reason to consult checkpoint statistics is to find the average interval per checkpoint. You can use the information in the total count column to compute this average. For each category of checkpoint reason, use the average interval per checkpoint to help you decide if a checkpointing interval should be adjusted, and by how much.

If most of the checkpoints for a database are triggered by a particular checkpoint limit, that limit may be set too high, or the other two limits may be set too low. You can determine the average interval per checkpoint for each type of checkpoint limit. After you have this information, you can reset the limits so that each type of checkpoint limit triggers approximately the same number of checkpoints, which results in optimal performance.

To compute the average transactions interval, divide the total count for transaction intervals by the total number of checkpoints. If the total count for transaction intervals is 800 and the total number of checkpoints is 100, then the average number of transactions between checkpoints is 8.

800 / 100 = 8

The description of the **interval: AIJ blks** field explains how to determine the average interval for after-image journal growth checkpoints.

The description of the **interval: seconds** field explains how to determine the average interval for time checkpoints.

9.7.11 interval: seconds

This field displays the sum of the intervals between time in seconds checkpoints for all processes. For example, if Process 1 checkpoints after 500 seconds, the time in seconds category is incremented by 500. If Process 2 checkpoints after 600 seconds, the time in seconds category is incremented by an additional 600. Statistics for the other two interval categories are displayed in the **interval: AIJ blks** and **interval: tx count** fields.

If CHECKPOINT TIMED EVERY 600 SECONDS is specified with the SQL ALTER DATABASE statement, each process checkpoints every 10 minutes.

Keep in mind that checkpointing influences recovery time. The main reason to consult checkpoint statistics is to find the average interval per checkpoint. You can use the information in the total count column to compute this average. For each category of checkpoint reason, use the average interval per checkpoint to help you decide if a checkpointing interval should be adjusted, and by how much.

If most of the checkpoints for a database are triggered by a particular checkpoint limit, that limit may be set too high, or the other two limits may be set too low. You can determine the average interval per checkpoint for each type of checkpoint limit. After you have this information, you can reset the limits so that each type of checkpoint limit triggers approximately the same number of checkpoints, which results in optimal performance.

To compute the average time interval, divide the total count for seconds interval by the total number of checkpoints. If the total count for the seconds field is 59,300 and the total number of checkpoints is 100, the average number of seconds between each time-triggered checkpoint is 593.

59,300 / 100 = 593

The description of the **interval: AIJ blks** field explains how to determine the average interval for after-image journal growth checkpoints.

The description of the **interval: tx count** field explains how to determine the average interval for transaction checkpoints.

9.7.12 checkpoint stall

This field displays the checkpoint duration, expressed in hundredths of a second displayed as a whole number. For example, the value “500” is actually “5” seconds.

9.7.13 flushed buffers

This field displays the number of buffers flushed to disk during a checkpoint operation.

Chapter 10

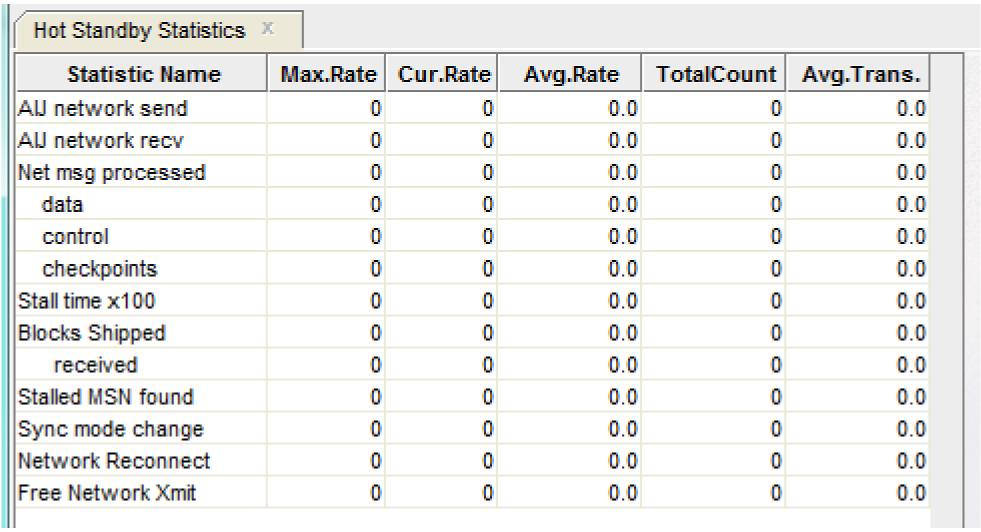
Hot Standby Information

The following Hot Standby information and statistics are available:

- Hot Standby Statistics
- Hot Standby Status
- Synchronization Mode

10.1 *Hot Standby Statistics*

This panel provides information about the performance of the Hot Standby feature.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
AIJ network send	0	0	0.0	0	0.0
AIJ network recv	0	0	0.0	0	0.0
Net msg processed	0	0	0.0	0	0.0
data	0	0	0.0	0	0.0
control	0	0	0.0	0	0.0
checkpoints	0	0	0.0	0	0.0
Stall time x100	0	0	0.0	0	0.0
Blocks Shipped	0	0	0.0	0	0.0
received	0	0	0.0	0	0.0
Stalled MSN found	0	0	0.0	0	0.0
Sync mode change	0	0	0.0	0	0.0
Network Reconnect	0	0	0.0	0	0.0
Free Network Xmit	0	0	0.0	0	0.0

Figure 46 - Hot Standby Statistics Pane

The following sections detail the information presented in this panel.

10.1.1 AIJ network send

The number of network messages sent.

10.1.2 AIJ network recv

The number of network messages received.

10.1.3 Net msg processed

The number of network messages processed. This field is only updated on the standby database, and is intended to be used as a comparison against the **AIJ network recv** field.

10.1.4 data

The number of data messages sent to the standby database or received from the master database.

10.1.5 control

The number of control (non-data) messages sent to the standby database or received from the master database. Controls messages are typically requests for information about the status of the other database, used for consistency verification.

10.1.6 checkpoints

The number of checkpoint operations performed. Note that these are Hot Standby checkpoints, not to be confused with the Fast Commit feature transactional checkpoints.

10.1.7 Stall time x100

The network I/O duration, in hundredths of seconds. For instance, the value “500” indicates “5” seconds.

10.1.8 blocks shipped

The number of 512-byte blocks sent.

10.1.9 received

The number of 512-byte blocks received.

10.1.10 Stalled MSN found

The number of stalled messages identified.

10.1.11 Synch mode change

The number of times synch mode changed between master and standby.

10.1.12 Network Reconnect

The number of times the network connection was lost between the master and standby databases. This statistic only applies to the master database. It could also indicate that the network object server (router) died prematurely.

10.1.13 Free Network Xmit

The number of messages that have been sent for “free”. That is, a group commit buffer that does not contain any transaction “commit” records can always be transmitted with a synchronization mode of “cold” since no replication of the buffer can be performed on the standby side. This is an indication of master database performance optimization.

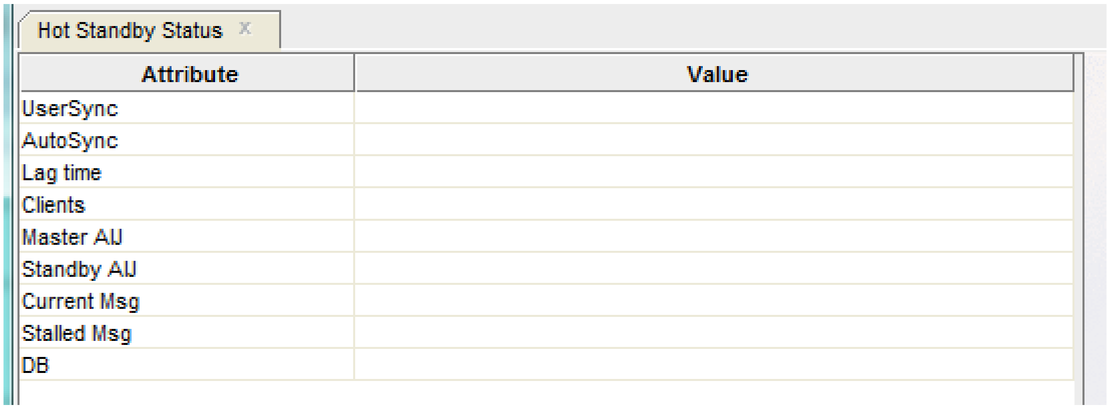
Note:

✓ On the standby database, comparing the **rate.per.second** column of the **AIJ network recv** and **Net msg processed** statistics is a useful way to determine if the performance of the master database exceeds the ability of the standby database to apply the changes. The **AIJ network recv** statistic represents the rate of in-coming messages from the master database, and the **Net msg processed** statistic represents the rate at which these messages are processed. Obviously, if the in-coming rate exceeds the process rate, then the standby database will lag behind the master database, or the master database will need to reduce its network throughput.

Contents

10.2 *Hot Standby Status*

This panel provides information about the state of the Hot Standby feature.



Attribute	Value
UserSync	
AutoSync	
Lag time	
Clients	
Master AIJ	
Standby AIJ	
Current Msg	
Stalled Msg	
DB	

Figure 47 - Hot Standby Status Pane

The following sections detail the information presented in this panel.

10.2.1 UserSync

The minimum message synchronization mode to be used by the master database AIJ Log Server (ALS) when sending messages to the standby database.

The keywords are the following:

Keyword	**Description**
Cold	cold synchronization
Warm	warm synchronization
Hot	hot synchronization
Commit	commit synchronization

10.2.2 AutoSync

This field indicates the message synchronization mode currently being used by the master database AIJ Log Server (ALS) when it sends messages to the standby database.

This field always has the same or stronger synchronization mode as the **User Sync** field.

The keywords are the following:

Keyword	**Description**
Cold	cold synchronization
Warm	warm synchronization
Hot	hot synchronization
Commit	commit synchronization

10.2.3 Lag Time

This field provides an “estimate” of the time that the standby database lags behind the master database. The concept is the following: If the master database were to fail right now, this field indicates how long it would take the standby database to catch-up to the master database’s current transactional state.

Note that this field varies widely depending on the size and scope of the database transactions. When transactions are relatively short, the standby database lag time estimate should be closely in sync with the master database. Very long transactions tend to cause the standby database lag time to slowly increase and then suddenly catch-up as the long transactions are applied to the standby database.

Ideally, this value should display “00:00:00” although this is not always practical.

The database lag time can be controlled using the synchronization modes and the “Replication Governor” feature.

10.2.4 Clients

The number of master database clients attached to the standby database. A “client” is the AIJ Log Server (ALS).

The value of this field is displayed underneath the field name.

10.2.5 Master AIJ

This field identifies the current AIJ end-of-file on the master database. Comparing this field to the value of the Stby.AIJ field provides an estimate of the work remaining to be applied on the standby database.

10.2.6 Standby AIJ

This field identifies the current AIJ end-of-file on the standby database. Comparing this field to the value of the **Master AIJ** field provides an estimate of the work remaining to be applied on the standby database.

10.2.7 Current Msg

The current message sequence number being used by this node.

10.2.8 Stalled Msg

This field indicates the missing message number that the standby database is waiting for. This message is almost always from a different node on the master database.

If the standby database is not waiting on a message, which is the normal case, then the value “none” is displayed for this field.

10.2.9 DB

The current state of the Hot Standby feature. Possible values are:

State	**Description**
Inactive	This keyword indicates that the Hot Standby feature is not being used.
DB Bind	This keyword indicates that the database is being set up for replication.
Net Bind	This keyword indicates that the network connection is being established.
Restart	This keyword indicates that the standby database is performing replication “restart” processing.
Connecting	This keyword indicates that master and standby databases are connecting together to establish the proper replication modes.
DB Synch.	This keyword indicates that the master and standby databases are being synchronized.
Activating	This keyword indicates that the AIJ Log Servers (ALS) on the master database are being activated. That is, replication is commencing.
SyncCmpltn	This keyword indicates that the master and standby databases have been synchronized and that the two databases are being verified for transactional consistency.
DECnet	This keyword indicates that the Hot Standby feature is active and using the DECnet communications protocol.

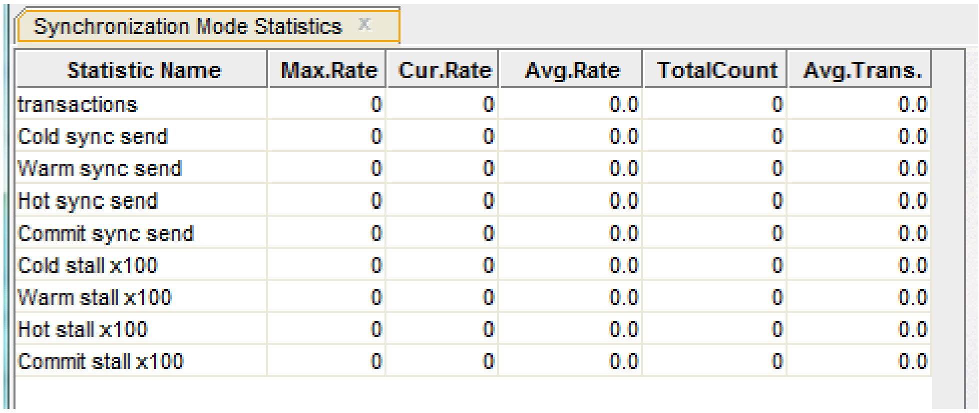
TCP/IP	This keyword indicates that the Hot Standby feature is active and using the TCP/IP communications protocol.
Active	This keyword indicates that the Hot Standby feature is active and being replicated locally. Local replication means that the master and standby databases are on the same node of the Cluster. When displaying this panel for the standby database, a number will occasionally be displayed to the right of the “Active” keyword. This number indicates the total number of “pending” messages from the master database that remain to be applied to the standby database.
Suspended	This keyword indicates that the Hot Standby feature is temporarily suspended.
Resumption	This keyword indicates that a temporarily suspended master database is in the process of being resumed. Transactional activity that occurred while the Hot Standby feature was suspended is being transferred to the standby database.
Completion	This keyword indicates that the Hot Standby feature is being shut down, and that final completion processing is being performed.
Shutdown	This keyword indicates that the Hot Standby feature is being shut down.
Net Unbind	This keyword indicates that the network connection between the master and standby databases is being disconnected.
DB Unbind	This keyword indicates that the database is being removed from Hot Standby replication mode.
Recovery	This keyword indicates that the database is being recovered following a premature failure of the Hot Standby feature, usually as a result of network failure or server failure.

Contents

10.3 *Synchronization Mode*

This panel provides a breakdown of each type of synchronization mode. The count and stall duration for each of the four synchronization modes are displayed.

This panel is important for analyzing network bandwidth utilization and standby database resource allocation effectiveness.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
Cold sync send	0	0	0.0	0	0.0
Warm sync send	0	0	0.0	0	0.0
Hot sync send	0	0	0.0	0	0.0
Commit sync send	0	0	0.0	0	0.0
Cold stall x100	0	0	0.0	0	0.0
Warm stall x100	0	0	0.0	0	0.0
Hot stall x100	0	0	0.0	0	0.0
Commit stall x100	0	0	0.0	0	0.0

Figure 48 - Synchronization Mode Pane

The following sections detail the information presented in this panel.

10.3.1 transactions

The number of completed database transactions. This is the count of the COMMIT and ROLLBACK statements that have executed.

10.3.2 cold sync send

The number of cold mode messages sent to the standby database.

10.3.3 warm sync send

The number of warm mode messages sent to the standby database.

10.3.4 hot sync send

The number of hot mode messages sent to the standby database.

10.3.5 commit sync send

The number of commit mode messages sent to the standby database.

10.3.6 cold stall x100

The cold message duration, in hundredths of seconds.

10.3.7 warm stall x100

The warm message duration, in hundredths of seconds.

10.3.8 hot stall x100

The hot message duration, in hundredths of seconds.

10.3.9 commit stall x100

The commit message duration, in hundredths of seconds.

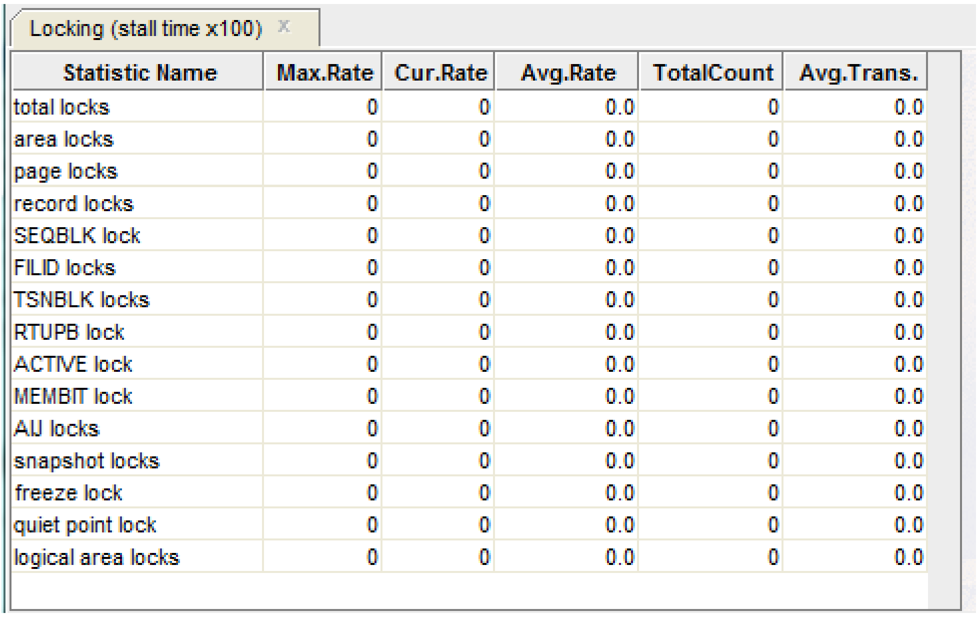
Note:

✓ The purpose of this panel is to verify that the database synchronization mode you chose is effective and appropriate for the runtime performance characteristics of the database. For example, if you specified the “Cold” synchronization mode, but you are encountering mostly “Warm” and “Hot” message sends, it could be indicative of insufficient network band-width. It could also be indicative of insufficient resources on the standby database. Try increasing the standby database number of buffers.

Contents

Chapter 11
Locking (One Stat Field)

The ***Locking (One Stat Field)*** panels display locking information for a specific lock statistic, compared to the various lock types.



Locking (stall time x100)

Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
total locks	0	0	0.0	0	0.0
area locks	0	0	0.0	0	0.0
page locks	0	0	0.0	0	0.0
record locks	0	0	0.0	0	0.0
SEQBLK lock	0	0	0.0	0	0.0
FILID locks	0	0	0.0	0	0.0
TSNBLK locks	0	0	0.0	0	0.0
RTUPB lock	0	0	0.0	0	0.0
ACTIVE lock	0	0	0.0	0	0.0
MEMBIT lock	0	0	0.0	0	0.0
AIJ locks	0	0	0.0	0	0.0
snapshot locks	0	0	0.0	0	0.0
freeze lock	0	0	0.0	0	0.0
quiet point lock	0	0	0.0	0	0.0
logical area locks	0	0	0.0	0	0.0

Figure 49 - Example Locking One Stat Field Pane

All the **One Stat Field** panels display a table similar to the example provided above.

The following sections detail the information presented in this panel.

11.1.1 total locks

Statistics for all types of database locks. Note that this count includes all locks, not just those listed below.

11.1.2 area locks

Statistics for database storage area locks.

11.1.3 buffer page locks

Statistics for database page locks. Page locks manage the database page buffer pool.

11.1.4 record locks

Statistics for database record locks. Record locks maintain the logical consistency of the database. This set of statistics includes all record locks in the adjustable lock granularity tree.

11.1.5 SEQBLK lock

Statistics for the database sequence block (SEQBLK) locks. The SEQBLK locks maintain global sequence numbers or transaction and commit sequence numbers and control COMMIT and ROLLBACK operations.

11.1.6 FILID locks

Statistics for the database file identification (FILID) locks. The FILID locks maintain consistent end-of-file information for the database root file (.RDB), live storage areas (.RDA) and snapshot storage areas (.SNP).

11.1.7 TSNBLK locks

Statistics for the database transaction block (TSNBLK) locks. The TSNBLK locks control the COMMIT and ROLLBACK operations on each cluster node. TSNBLK locks are also used to control SQL SET TRANSACTION statements for read-only transactions.

11.1.8 RTUPB lock

Statistics for the database runtime user process block (RTUPB) lock.

The RTUPB locks maintain a consistent list of the users who are attached to the database. They also maintain the process checkpoint information when the Fast Commit feature is enabled.

11.1.9 ACTIVE lock

Statistics for the database active user bit map (ACTIVE) lock. The ACTIVE lock maintains a consistent list (in bit map form) of the users who are attached to the database.

11.1.10 MEMBIT lock

Statistics for the database membership node bit map (MEMBIT) lock.

The MEMBIT locks maintain a consistent list (in bit map form) of the cluster nodes on which the database is currently accessed.

11.1.11 AIJ locks

Statistics for the after-image journal (AIJ) locks. AIJ locks control reading from and writing to the after-image journal. One global AIJ lock maintains current end-of-file information. In addition, there is one local AIJ lock on each cluster node that manages the global AIJ buffer on that node.

11.1.12 snapshot locks

Statistics for the database snapshot area cursor (SAC) locks. Snapshot locks manage the allocation of snapshot pages to users who are updating the database.

Snapshot locks are only used if snapshots are enabled for a storage area.

11.1.13 freeze lock

Statistics for the database freeze lock. The freeze lock suspends database activity while a database recovery process is running.

11.1.14 quiet point lock

Statistics for the database quiet-point lock. The quiet-point lock suspends starting new transactions while the AIJ backup utility is trying to finish backing up the contents of the after-image journal when you use the RMU Backup After_Journal utility or the AIJ Backup Server (ABS) process. The quiet-point lock also suspends starting new transactions during the startup of an online RMU Backup command.

11.1.15 logical area locks

Logical area locks are obtained when Oracle Rdb readies tables. Lock carryover can help reduce the number of logical area locks.

11.1.16 nowait transaction

Statistics for the database nowait transaction lock.

11.1.17 CLIENT locks

Monitors the database client information (CLIENT) lock. The CLIENT locks are used to provide serialized access to the database metadata stored in the system relations. The CLIENT locks are also used to serialize operations such as creating tables and indices.

Contents

11.2 Individual Panels

Details of the individual panels within this group:

11.2.1 Stall Time x100

This panel monitors the total time (in hundredths of a second) spent by all users waiting for a lock. Since more than one user can be waiting for a lock at the same time, this total can be greater than the actual elapsed clock time. These statistics give a relative measure of work lost due to lock conflicts.

This is probably the most best locking panel to examine first, as it presents an overall identification of lost work. Eliminating stalls from this panel almost always directly improves application throughput.

11.2.2 Locks Requested

This panel monitors the number of enqueue lock requests for new locks. Whether the lock request succeeds or fails, it is included in these counts.

11.2.3 Rqsts Not Queued

This panel monitors the number of enqueue lock requests for new locks that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting, and, when a conflict is detected, resorts to a secondary locking protocol to avoid unnecessary deadlocks. These numbers are one measure of lock contention.

11.2.4 Rqsts Stalled

This panel monitors the number of enqueue lock requests for new locks that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting, and, when a conflict is detected, resorts to a secondary locking protocol to avoid unnecessary deadlocks. These numbers are one measure of lock contention.

11.2.5 Rqst Timeouts

This panel monitors the total number of lock requests that could not be granted because they timed out.

These are typically logical areas or record locks.

Each lock timeout reported in the **rqst timeouts** panel is also reported in the **rqsts stalled** panel. This is because each timed out request is also a stalled request.

11.2.6 Rqst Deadlocks

This panel monitors the number of stalled enqueue lock requests for new locks that ultimately resulted in a deadlock. Most deadlocks are resolved and retried by Oracle Rdb transparently to the application program, therefore these numbers do not necessarily reflect the number of deadlocks reported to the application program.

These are typically page locks.

Each lock deadlock reported in the **rqst deadlocks** panel is also reported in the **rqsts stalled** panel. This is because each deadlocked request is also a stalled request.

11.2.7 Locks Promoted

This panel monitors the number of enqueue lock requests to promote an existing lock to a higher lock mode. Whether the lock requests succeed or fail, they are included in these counts.

11.2.8 Proms Not Queued

This panel monitors the number of enqueue lock requests to promote an existing lock that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting, and, when a conflict is detected, resorts to a secondary locking protocol to avoid unnecessary deadlocks. These numbers are one measure of lock contention.

11.2.9 Proms Stalled

This panel monitors the number of enqueue lock requests to promote an existing lock to a higher lock mode that were stalled because of a lock conflict. Whether the lock requests ultimately succeed or fail (resulting in a deadlock), they are included in these counts. These numbers are one measure of lock contention.

11.2.10 Prom Timeouts

This panel monitors the number of lock promotions that could not be granted because they timed out.

These are typically logical areas or record locks.

Each promotion timeout reported in the **prom timeouts** panel is also reported in the **proms stalled** panel. This is because each timed out request is also a stalled request.

11.2.11 Prom Deadlocks

This panel monitors the number of stalled enqueue lock requests to promote an existing lock to a higher lock mode that ultimately resulted in a deadlock. Most deadlocks are resolved and retried by Oracle Rdb transparently to the application program, therefore these numbers do not necessarily reflect the number of deadlocks reported to the application program.

These are typically page locks.

Each promotion deadlock reported in the **prom deadlocks** field is also reported in the **proms stalled** panel. This is because each deadlocked request is also a stalled request.

11.2.12 Blocking ASTs

This panel monitors the number of blocking ASTs, sometimes referred to as blasts, delivered to Oracle Rdb by the OpenVMS lock manager. A blocking AST is delivered to the holder of a lock when a lock conflict is detected. 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 actually comprised of two different types of blocking ASTs, those blocking ASTs externally generated and those blocking ASTs internally generated.

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 **ORCM RMU Statistics** tool records the blocking AST activity.

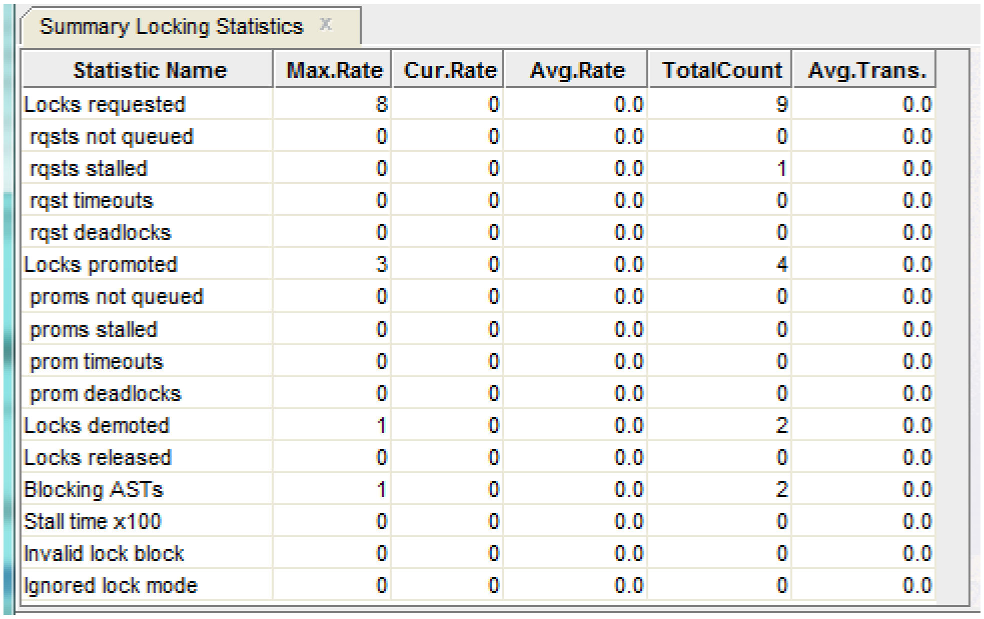
An internally generated blocking AST occurs when a lock blocking AST routine is executed by the process in anticipation that the same work would have to be performed anyway if a blocking AST were to be received from the operating system, even when no blocking AST from the operating system actually occurred. This algorithm serves as an optimistic code optimization; it is assumed that the process would eventually receive a blocking AST for the particular lock, so it optimistically executes the blocking AST routine. The **ORCM RMU Statistics** tool does not differentiate between these two types of blocking ASTs.

Contents

Chapter 12

Locking (One Lock Type)

The ***Locking (One Lock Type)*** panels display locking information for a specific lock type.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
Locks requested	8	0	0.0	9	0.0
rqsts not queued	0	0	0.0	0	0.0
rqsts stalled	0	0	0.0	1	0.0
rqst timeouts	0	0	0.0	0	0.0
rqst deadlocks	0	0	0.0	0	0.0
Locks promoted	3	0	0.0	4	0.0
proms not queued	0	0	0.0	0	0.0
proms stalled	0	0	0.0	0	0.0
prom timeouts	0	0	0.0	0	0.0
prom deadlocks	0	0	0.0	0	0.0
Locks demoted	1	0	0.0	2	0.0
Locks released	0	0	0.0	0	0.0
Blocking ASTs	1	0	0.0	2	0.0
Stall time x100	0	0	0.0	0	0.0
Invalid lock block	0	0	0.0	0	0.0
Ignored lock mode	0	0	0.0	0	0.0

Figure 50 - Example Locking One Lock Type Pane

All the **One Lock Type** panels display a table similar to the example provided above.

The following sections detail the information presented in this panel.

12.1.1 locks requested

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.

The **rqsts not queued**, **rqsts stalled**, and **rqst deadlocks** counts provide further detail for enqueue lock requests statistics.

12.1.2 rqsts not queued

This field gives the number of enqueue lock requests for new locks that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting and, when a conflict is detected, resorts to a secondary locking protocol to avoid unnecessary deadlocks. This number is one measure of lock contention.

12.1.3 rqsts stalled

This field gives the number of enqueue lock requests for new locks that were stalled because of a lock conflict. Whether or not the lock request ultimately succeeds, it is included in this count. This number is one measure of lock contention.

12.1.4 rqst timeouts

This field shows the total number of lock requests that could not be granted because they timed out. These are typically logical areas.

Each lock timeout reported in the **rqst timeouts** field is also reported in the **rqsts stalled** field. This is because each timed out request is also a stalled request.

12.1.5 rqst deadlocks

This field gives the number of stalled enqueue lock requests for new 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.

Each lock deadlock reported in the **rqst deadlocks** field is also reported in the **rqsts stalled** field. This is because each deadlocked request is also a stalled request.

12.1.6 locks promoted

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. The **proms not queued**, **proms stalled**, and **prom deadlocks** counts provide further detail for the locks promotion statistics.

12.1.7 proms not queued

This field gives the number of enqueue lock requests to promote an existing lock that were rejected immediately because of a lock conflict. Oracle Rdb often requests a lock without waiting. When a conflict is detected, Oracle Rdb resorts to a secondary locking protocol to avoid unnecessary deadlocks. This number is one measure of lock contention.

12.1.8 proms stalled

This field gives the number of enqueue lock requests to promote an existing lock to a higher lock mode that were stalled because of a lock conflict. Whether or not the lock request ultimately succeeds, it is included in this count. This number is one measure of lock contention.

12.1.9 prom timeouts

This field shows the total number of lock promotions that could not be granted because they timed out. These are typically logical areas.

Each promotion timeout reported in the **prom timeouts** field is also reported in the **proms stalled** field. This is because each timed out request is also a stalled request.

12.1.10 prom deadlocks

This field gives the number of stalled enqueue lock requests to promote an existing lock 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, this number does not necessarily reflect the number of deadlocks reported to the application program.

Each promotion deadlock reported in the **prom deadlocks** field is also reported in the **proms stalled** field. This is because each deadlocked request is also a stalled request.

12.1.11 locks demoted

This field gives the number of enqueue lock requests to demote an existing lock to a lower lock mode. These requests always succeed.

12.1.12 locks released

This field gives the number of deallocating lock requests to release an existing lock.

These requests always succeed. The number of outstanding locks can be determined by the formula:

(locks requested) - (rqsts not queued) - (rqst deadlocks) - (locks released).

12.1.13 blocking ASTs

This field gives the number of blocking ASTs, sometimes referred to as “blasts”, delivered to Oracle Rdb by the OpenVMS 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 actually comprised of two different types of blocking ASTs, those blocking ASTs externally generated and those blocking ASTs internally generated.

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 **ORCM RMU Statistics** tool 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, even when no blocking AST from the operating system actually occurred. This algorithm serves as an optimistic code optimization; it is assumed that the process would eventually receive a blocking AST for the particular lock, so it optimistically executes the blocking AST routine. The **ORCM RMU Statistics** tool does not differentiate between these two types of blocking ASTs.

12.1.14 stall time x100

This field gives the total time (in hundredths of a second) spent by all users waiting for a lock. Since more than one user can be waiting for a lock at the same time, this total can be greater than the actual elapsed clock time. This statistic gives a relative measure of work lost due to lock conflicts.

Contents

12.2 Individual Panels

Details of the individual panels within this group:

12.2.1 Total Locks

This panel monitors the total database locking activity. The statistics in this panel are the totals for all types of database locks.

Note:

✓ The **Total Locks** and the Summary Locking Statistics information are identical. This panel includes locks that may have their own panel, such as the AIJ switch-over or Hot Standby locks.

12.2.2 Area Locks

This panel monitors the database storage area locks. Physical areas are simply another name for “storage areas”.

12.2.3 Buffer/Page Locks

This panel monitors the database page locks. Page locks are used to manage the database page buffer pool.

12.2.4 Record Locks

This panel monitors the database record locks. Record locks are used to maintain the logical consistency of the database. All record locks in the adjustable lock granularity tree are included here.

12.2.5 SEQBLK Locks

This panel monitors the database sequence block (SEQBLK) locks. The SEQBLK locks maintain global transaction sequence numbers or transaction and commit sequence numbers and control COMMIT and ROLLBACK operations.

12.2.6 FILID Locks

This panel monitors the database file identification (FILID) locks. The FILID locks are used to maintain consistent end-of-file information for the database root file (.RDB), live storage areas (.RDA) and snapshot storage areas (.SNP).

12.2.7 TSNBLK Locks

This panel monitors the database transaction block (TSNBLK) locks. The TSNBLK locks are used to control the COMMIT and ROLLBACK operations on each cluster node. TSNBLK locks are also used to control SQL SET TRANSACTION statements for read-only transactions.

12.2.8 RTUPB Locks

This panel monitors the database runtime user process block (RTUPB) lock. The RTUPB lock is used to maintain a consistent list of the users who are attached to the database.

12.2.9 ACTIVE Locks

This panel monitors the database active user bit map (ACTIVE) lock. The ACTIVE lock is used to maintain a consistent list (in bit map form) of the users who are attached to the database.

12.2.10 MEMBIT Locks

This panel monitors the database membership node bit map (MEMBIT) lock. The MEMBIT lock is used to maintain a consistent list (in bit map form) of the nodes on which the database is currently accessed.

12.2.11 AIJ Locks

This panel monitors the after-image journal (AIJ) locks. The AIJ locks are used to control reading and writing to the after-image journal. One global AIJ lock maintains current end-of-file information. In addition, one local AIJ lock on each cluster node manages the global AIJ buffers on that node.

Note that there are actually two separate AIJ locks monitored by this panel. The “global” AIJ lock serializes access to the AIJ end-of-file, while the “local” AIJ lock serializes access to the node-specific AIJ cache located in the database global section.

12.2.12 Snapshot Locks

This panel monitors the database snapshot area cursor (SAC) locks. The snapshot locks are used to manage the allocation of snapshot pages to users who are updating the database. Snapshot locks are only used if snapshots are enabled for a storage area.

12.2.13 Freeze Locks

This panel monitors the database freeze lock. The freeze lock is used to suspend database activity on all nodes of the cluster while a database recovery process is running.

12.2.14 Quiet Point Locks

This panel monitors the database quiet-point lock.

The quiet-point lock suspends starting new transactions while the AIJ backup utility is trying to finish backing up the contents of the after-image journal when you use the RMU Backup After_Journal utility or AIJ Backup Server (ABS). The quiet-point lock also suspends starting new transactions during the startup of an online RMU Backup utility. The Oracle Rdb Hot Standby facility also uses the quiet-point lock during startup.

Note:

- ✓ Using the RDM$BIND_SNAP_QUIET_POINT logical with a value of “0” in almost all cases improves the quiet-point lock acquisition, by allowing snapshot transactions to not be affected by the lock.

12.2.15 Logical Area Locks

This panel monitors database logical area locks. Logical area locks are obtained when Oracle Rdb readies tables. Lock carryover can help reduce the number of logical area locks.

12.2.16 Nowait Transaction Locks

This panel monitors the database nowait transaction lock. See NOWAIT transactions for more information.

Note:

- ✓ The Oracle Rdb “Carry-over Lock Optimization” feature allows transactions to avoid some logical area and storage area lock request overhead at commit time. See CarryOver Locks.

12.2.17 CLIENT Locks

This panel monitors the database client information (CLIENT) locks. The CLIENT locks are used to provide serialized access to the database metadata stored in the system relations. The CLIENT locks are also used to serialize operations such as creating tables and indices.

Contents

12.3 NOWAIT transactions

NOWAIT transactions do not wait for locks. If a lock requested by a NOWAIT transaction cannot be granted immediately, Oracle Rdb issues an error message and the transaction aborts. As part of carry-over lock optimization, a NOWAIT transaction requests, acquires, and holds a NOWAIT lock. This signals other processes accessing the database that a NOWAIT transaction exists and results in the release of all carry-over locks. If carry-over locks were not released, a NOWAIT transaction could not access an area held by a WAIT transaction's carry-over lock until the WAIT transaction's process detached from the database.

Every NOWAIT transaction requests the NOWAIT lock at transaction start in CW mode and waits until this lock is granted. When a transaction acquires the NOWAIT lock in CW mode, this indicates that all other users know that a NOWAIT transaction is running and indicates that all carry-over locks have been released, thus reducing the possibility of lock contention.

All transactions request the NOWAIT lock in PR mode at commit time. If the NOWAIT lock is granted in PR mode, it indicates that there are no NOWAIT transactions attached to the database and carry-over locks are permitted. If the NOWAIT lock request is not granted (because a NOWAIT transaction holds the lock in CW mode), carry-over locks are not permitted.

However, a NOWAIT transaction can experience a delay in acquiring the NOWAIT lock if another transaction is holding the lock. This can result in the following stall message: “waiting for NOWAIT signal (CW)”.

12.4 CarryOver Locks

This section provides a general description of how carry-over lock optimization works, and then describes its particular effects on WAIT and NOWAIT transactions.

An area lock requested by a current transaction is called an active lock. At commit time, Oracle Rdb tries to avoid demoting area locks. Those area locks that are not demoted at commit time are called “carry-over” locks.

While attached to the database, a process can have some active locks (locks used by the current transaction) and some carry-over locks (locks requested in earlier transactions that have not been demoted). If a transaction needs a lock that it has currently marked as carry-over, it can reuse the lock by changing it to an active lock. Thus, the same lock can go from active to carry-over to active multiple times without paying the cost of lock request and demotion. This substantially reduces the number of lock requests if a process accesses the same set of areas repeatedly.

Whenever a WAIT transaction requests an area lock, Oracle Rdb must distinguish between the following two cases in which process A has a lock on area X and process B wants to access the same area:

- If the lock that process A has on area X is a carry-over lock, A gives up the lock on demand, process B gets it, and B continues to process.
- If the lock that process A has on area X is an active lock, A cannot give up the lock before its transaction has completed. In this case, Oracle Rdb sets a flag to indicate that this lock must be demoted when A's transaction commits, so that B can acquire it. Because process B cannot get the lock on demand, B must wait.

For WAIT transactions, the reduced number of locks associated with carry-over lock optimization can result in an increase in blocking ASTs. You can see an increase in blocking ASTs by using the various **ORCM RMU Statistics** tool Locking panels.

Carry-over lock optimization works well when applications are designed so that each transaction accesses its own set of data; that is, transactions do not randomly access data in all partitions, thereby increasing contention. For example, consider the EMPLOYEE_ID column, which partitions the EMPLOYEES table to three areas.

Applications that access the EMPLOYEES table should be designed so that transactions access a particular area or set of areas instead of randomly selecting any area.

Furthermore, carry-over lock optimization works best if transactions repeatedly access the same area or set of areas. The partitioning and placement features available in Oracle Rdb should help in this regard.

If NOWAIT transactions are noticeably slow in executing, you can disable carry-over lock optimization by using the CARRY OVER LOCKS ARE [ENABLED | DISABLED] clause with either the SQL CREATE DATABASE or SQL ALTER DATABASE statements. By default, carry-over locks are enabled.

The following example shows how to disable carry-over locks that have been enabled by default.

```
SQL> ALTER DATABASE FILENAME test1 CARRY OVER LOCKS ARE DISABLED;
```

Contents

Chapter 13

IO Statistics

The following IO Statistics are available:

- PIO Statistics – Data Fetches
- PIO Statistics – SPAM Fetches
- PIO Statistics – Data Prefetches
- PIO Statistics – SPAM Prefetches
- PIO Statistics – SPAM Access
- PIO Statistics – Data Writes
- PIO Statistics – SPAM Writes
- Asynchronous IO Statistics

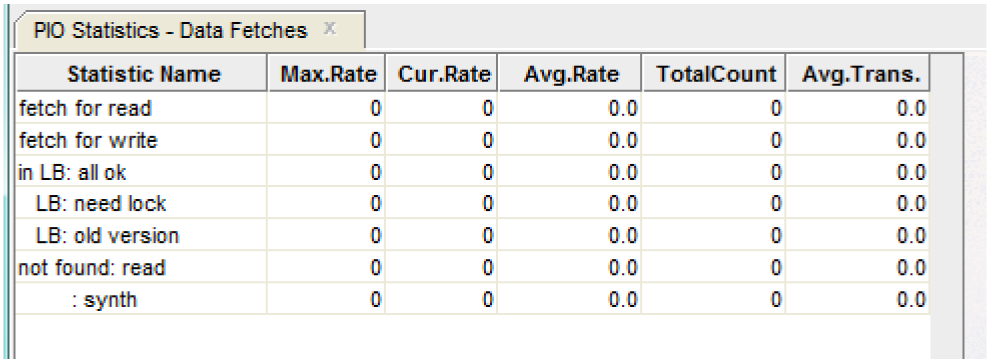
13.1 PIO Statistics – Data Fetches

Depending on the Global Buffers state of the database the ***PIO Statistics – Data Fetches*** pane may be displayed in either of the following formats:

- PIO Statistics – Data Fetches (Local Buffers)
- PIO Statistics – Data Fetches (Global Buffers)

13.2 PIO Statistics – Data Fetches (Local Buffers)

When Global Buffers are not enabled, the ***PIO Statistics – Data Fetches*** panel provides statistics on how local buffer data page requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
fetch for read	0	0	0.0	0	0.0
fetch for write	0	0	0.0	0	0.0
in LB: all ok	0	0	0.0	0	0.0
LB: need lock	0	0	0.0	0	0.0
LB: old version	0	0	0.0	0	0.0
not found: read	0	0	0.0	0	0.0
: synth	0	0	0.0	0	0.0

Figure 51 - PIO Statistics Data Fetches Pane (Local Buffer variant)

The following sections detail the information presented in this panel.

13.2.1 fetch for read

The number of synchronous data page requests to the PIO sub-system where only read privileges are being requested for the page.

If Oracle Rdb reads any area inventory pages (AIP) and area bit map (ABM) pages while fetching the data page, the requests for the AIP and ABM pages are included in the total count field.

The sum of the fetch for read and fetch for write fields equals the total number of synchronous data page requests to the PIO sub-system.

13.2.2 fetch for write

The number of data page requests to the PIO sub-system where update as well as read privileges are being requested for the page.

If Oracle Rdb reads any AIP and ABM pages while fetching the data page, the requests for the AIP and ABM pages are included in the total count field.

The sum of the fetch for read and fetch for write fields equals the total number of data page requests to the PIO sub-system.

13.2.3 in LB: all ok

The correct version of the requested page was found in the user's local buffer pool and the user already held the needed locks on that page.

This line and the next four lines categorize data page requests to the PIO sub-system in terms of what work the sub-system had to do to satisfy the request. The sum of this line and the next four lines should be equal to the sum of the first two lines.

The three lines with the LB heading further categorize those data page fetch requests to the PIO sub-system where the requested page was found in the user's local buffer pool.

13.2.4 LB: need lock

The correct version of the requested page was found in the user's local buffer pool but additional locking was required (that is, the user did not have the page locked in the needed mode).

13.2.5 LB: old version

The requested page was found in the user's local buffer pool but it was an obsolete version of the page (that is, the page has been changed by another user since it was read into this user's local buffer). Thus, the page must be read again from disk. In addition, locks will need to be obtained for this page.

13.2.6 not found: read

Because the requested page was not found in the buffer pool, it was read in from disk.

This required obtaining appropriate locks on the page.

This line and the “not found: synth” line further categorize data page fetch requests to the PIO sub-system in which the requested page did not reside in the user's buffer pool.

13.2.7 synth

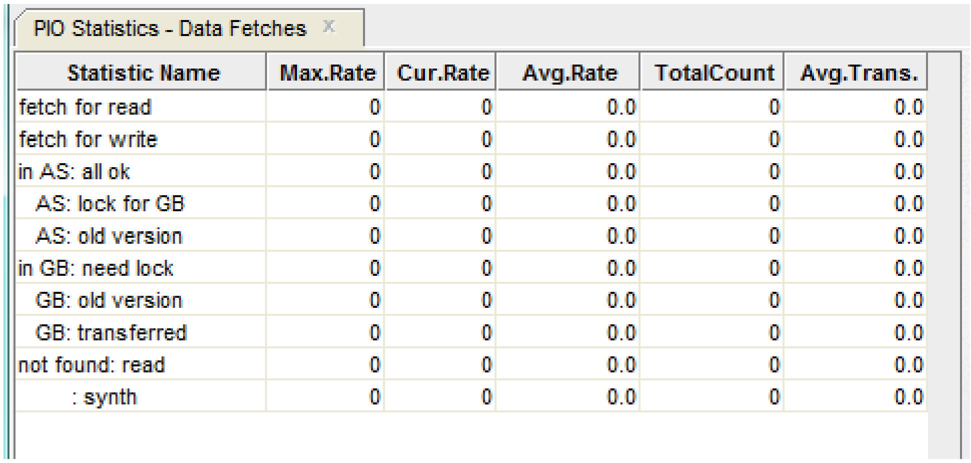
Because the requested page was not found in the buffer pool, it was synthesized into the pool without being read from disk. This required obtaining appropriate locks on the page.

A requested page that is synthesized is “Read” from a uniform format area and the clump is unallocated.

Contents

13.3 PIO Statistics – Data Fetches (Global Buffers)

If Global Buffers are enabled, the ***PIO Statistics – Data Fetches*** panel provides statistics on how global buffer data page requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
fetch for read	0	0	0.0	0	0.0
fetch for write	0	0	0.0	0	0.0
in AS: all ok	0	0	0.0	0	0.0
AS: lock for GB	0	0	0.0	0	0.0
AS: old version	0	0	0.0	0	0.0
in GB: need lock	0	0	0.0	0	0.0
GB: old version	0	0	0.0	0	0.0
GB: transferred	0	0	0.0	0	0.0
not found: read	0	0	0.0	0	0.0
: synth	0	0	0.0	0	0.0

Figure 52 - PIO Statistics Data Fetches Pane (Global Buffer variant)

The following sections detail the information presented in this panel.

13.3.1 fetch for read

The number of synchronous data page requests to the PIO sub-system where only read privileges are being requested for the page.

If Oracle Rdb reads any area inventory pages (AIP) and area bit map (ABM) pages while fetching the data page, the requests for the AIP and ABM pages are included in the total count field.

The sum of the fetch for read and fetch for write fields equals the total number of synchronous data page requests to the PIO sub-system.

13.3.2 fetch for write

The number of data page requests to the PIO sub-system where update as well as read privileges are being requested for the page.

If Oracle Rdb reads any AIP and ABM pages while fetching the data page, the requests for the AIP and ABM pages are included in the total count field.

The sum of the fetch for read and fetch for write fields equals the total number of data page requests to the PIO sub-system.

13.3.3 in AS: all ok

The correct version of the requested page was found in the user's allocate set and the user already held the needed locks on that page.

This line and the next eight lines categorize data page requests to the PIO sub-system in terms of what work the sub-system had to do to satisfy the request. The sum of this line and the next eight lines should be equal to the sum of the first two lines.

The four lines with the AS: heading further categorize those data page fetch requests to the PIO sub-system where the requested page was found in the user's allocate set.

13.3.4 AS: lock for GB

The page was found in the user's allocate set and the user held sufficient locks to satisfy the request. But because of global buffers, additional locking was needed to verify that the version was correct. The version was, in fact, correct, so the extra locking overhead was due solely to the fact that global buffers were being used.

13.3.5 AS: need lock

The correct version of the requested page was found in the user's allocate set but additional locking was required (that is, the user did not have the page locked in the needed mode).

13.3.6 AS: old version

The requested page was found in the user's allocate set but it was an obsolete version of the page (that is, the page has been changed by another user since it was added to the requester's allocate set). Thus, the page must be read again from disk. In addition, locks will need to be obtained for this page.

13.3.7 in GB: need lock

The correct version of the page was found in the global buffer pool. It was necessary to bring this page into the user's allocate set and to obtain a lock on the page.

This line and the “GB: old version” line further categorize data page fetch requests to the PIO sub-system in which the requested page was not found in the user's allocate set but was found in the global buffer pool.

13.3.8 GB: old version

The page was found in the global buffer pool but it was the wrong version; i.e. the buffer is stale. Thus, a lock had to be obtained, the page had to be read in from disk to a global buffer, and that buffer had to be brought into the user's allocate set.

13.3.9 GB: transferred

The count of the number of data pages that were transferred from one process to another without being written to the disk.

This field is active only when the “transfer via memory” feature is enabled.

13.3.10 not found: read

Because the requested page was not found in the global buffer pool, it was read in from disk.

This required obtaining appropriate locks on the page.

This line and the **synth** line further categorize data page fetch requests to the PIO sub-system in which the requested page did not reside in the user's buffer pool.

13.3.11 synth

Because the requested page was not found in the global buffer pool, it was synthesized into the pool without being read from disk. This required obtaining appropriate locks on the page.

A requested page that is synthesized is “Read” from a uniform format area and the clump is unallocated.

Contents

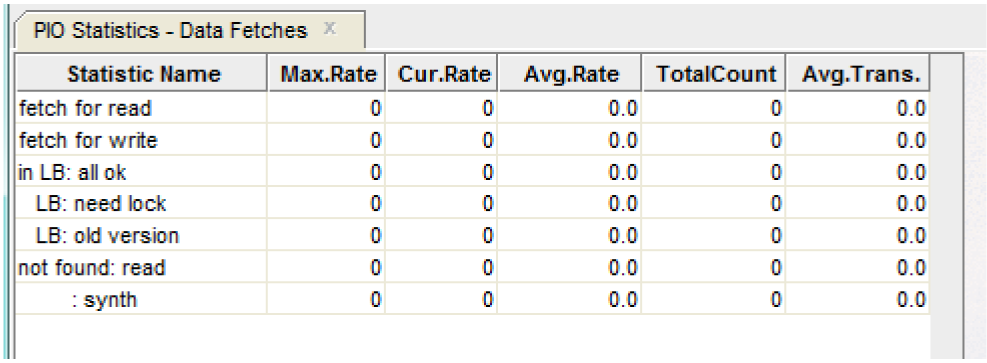
13.4 PIO Statistics – SPAM Fetches

Depending on the Global Buffers state of the database the ***PIO Statistics – SPAM Fetches*** pane may be displayed in either of the following formats:

- PIO Statistics – SPAM Fetches (Local Buffers)
- PIO Statistics – SPAM Fetches (Global Buffers)

13.5 PIO Statistics – SPAM Fetches (Local Buffers)

If Global Buffers are not enabled, the ***PIO Statistics – SPAM Fetches*** panel provides statistics on how local buffer SPAM page requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
fetch for read	0	0	0.0	0	0.0
fetch for write	0	0	0.0	0	0.0
in LB: all ok	0	0	0.0	0	0.0
LB: need lock	0	0	0.0	0	0.0
LB: old version	0	0	0.0	0	0.0
not found: read	0	0	0.0	0	0.0
: synth	0	0	0.0	0	0.0

Figure 53 - PIO Statistics Data Fetches Pane (Local Buffer variant)

The following sections detail the information presented in this panel.

13.5.1 fetch for read

The number of SPAM page requests to the PIO subsystem where only read privileges are being requested for the page.

The sum of the fetch for read and fetch for write fields equals the total number of SPAM page requests to the PIO subsystem.

13.5.2 fetch for write

The number of SPAM page requests to the PIO subsystem where update as well as read privileges are being requested for the page.

The sum of the fetch for read and fetch for write fields equals the total number of SPAM page requests to the PIO subsystem.

13.5.3 in LB: all ok

The correct version of the requested page was found in the user's local buffer pool and the user already held the needed locks on that page.

This line and the next four lines categorize SPAM page requests to the PIO subsystem in terms of what work the subsystem had to do to satisfy the request. The sum of this line and the next four lines should be equal to the sum of the first two lines.

The three lines with the “LB:” heading further categorize those SPAM page fetch requests to the PIO subsystem where the requested page was found in the user's local buffer pool.

13.5.4 LB: need lock

The correct version of the requested page was found in the user's local buffer pool but additional locking was required (that is, the user did not have the page locked in the needed mode).

13.5.5 LB: old version

The requested page was found in the user's local buffer pool but it was an obsolete version of the page (that is, the page has been changed by another user since it was read into this user's local buffer). Thus, the page must be read again from disk. In addition, locks will need to be obtained for this page.

13.5.6 not found: read

Because the requested page was not found in the buffer pool, it was read in from disk.

This required obtaining appropriate locks on the page.

This line and the **synth** line further categorize SPAM page fetch requests to the PIO subsystem in which the requested page did not reside in the user's buffer pool.

13.5.7 synth

Because the requested page was not found in the buffer pool, it was synthesized into the pool without being read from disk. This required obtaining appropriate locks on the page.

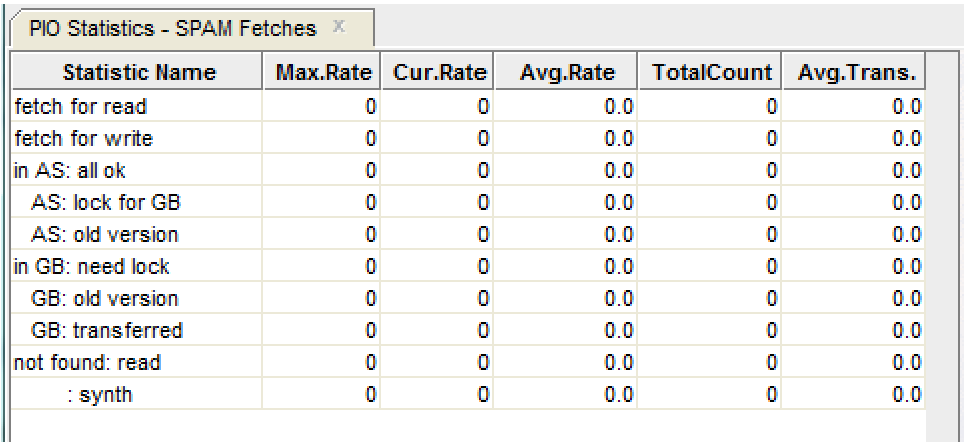
A requested page that is synthesized is “Read” from a uniform format area and the clump is unallocated.

The page does not actually have to be read from disk because the page contents are known (that is, Oracle Rdb can determine what a formatted unused page looks like). Therefore, rather than actually reading the page from disk, Oracle Rdb synthesizes the page (produces a properly-formatted unused page). Page and buffer locking must still occur to keep other users from accessing the same page.

Contents

13.6 PIO Statistics – SPAM Fetches (Global Buffers)

If Global Buffer are enabled, the ***PIO Statistics – Spam Fetches*** panel provides statistics global buffer SPAM page requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
fetch for read	0	0	0.0	0	0.0
fetch for write	0	0	0.0	0	0.0
in AS: all ok	0	0	0.0	0	0.0
AS: lock for GB	0	0	0.0	0	0.0
AS: old version	0	0	0.0	0	0.0
in GB: need lock	0	0	0.0	0	0.0
GB: old version	0	0	0.0	0	0.0
GB: transferred	0	0	0.0	0	0.0
not found: read	0	0	0.0	0	0.0
: synth	0	0	0.0	0	0.0

Figure 54 - PIO Statistics Spam Fetches Pane (Global Buffer variant)

The following sections detail the information presented in this panel.

13.6.1 fetch for read

The number of SPAM page requests to the PIO subsystem where only read privileges are being requested for the page.

The sum of the fetch for read and fetch for write fields equals the total number of SPAM page requests to the PIO subsystem.

13.6.2 fetch for write

The number of SPAM page requests to the PIO subsystem where update as well as read privileges are being requested for the page.

The sum of the fetch for read and fetch for write fields equals the total number of SPAM page requests to the PIO subsystem.

13.6.3 in AS: all ok

The correct version of the requested page was found in the user's allocate set and the user already held the needed locks on that page.

This line and the next eight lines categorize SPAM page requests to the PIO subsystem in terms of what work the subsystem had to do to satisfy the request. The sum of this line and the next eight lines should be equal to the sum of the first two lines.

The four lines with the “AS:” heading further categorize those SPAM page fetch requests to the PIO subsystem where the requested page was found in the user's allocate set.

13.6.4 AS: lock for GB

The page was found in the user's allocate set and the user held sufficient locks to satisfy the request. But because of global buffers, additional locking was needed to verify that the version was correct. The version was, in fact, correct, so the extra locking overhead was due solely to the fact that global buffers were being used.

13.6.5 AS: need lock

The correct version of the requested page was found in the user's allocate set but additional locking was required (that is, the user did not have the page locked in the needed mode).

13.6.6 AS: old version

The requested page was found in the user's allocate set but it was an obsolete version of the page (that is, the page has been changed by another user since it was added to the requester’s allocate set). Thus, the page must be read again from disk. In addition, locks will need to be obtained for this page.

13.6.7 in GB: need lock

The correct version of the page was found in the global buffer pool. It was necessary to bring this page into the user's allocate set and to obtain a lock on the page.

This line and the **GB: old version** line further categorize SPAM page fetch requests to the PIO subsystem in which the requested page was not found in the user's allocate set but was found in the global buffer pool.

13.6.8 GB: old version

The page was found in the global buffer pool but it was the wrong version; i.e. the buffer is stale. Thus, a lock had to be obtained, the page had to be read in from disk to a global buffer, and that buffer had to be brought into the user's allocate set.

13.6.9 GB: transferred

The count of the number of SPAM pages that were transferred from one process to another without being written to the disk. This field is active only when the “transfer via memory” feature is enabled.

13.6.10 not found: read

Because the requested page was not found in the global buffer pool, it was read in from disk.

This required obtaining appropriate locks on the page.

This line and the **synth** line further categorize data page fetch requests to the PIO sub-system in which the requested page did not reside in the user's buffer pool.

13.6.11 synth

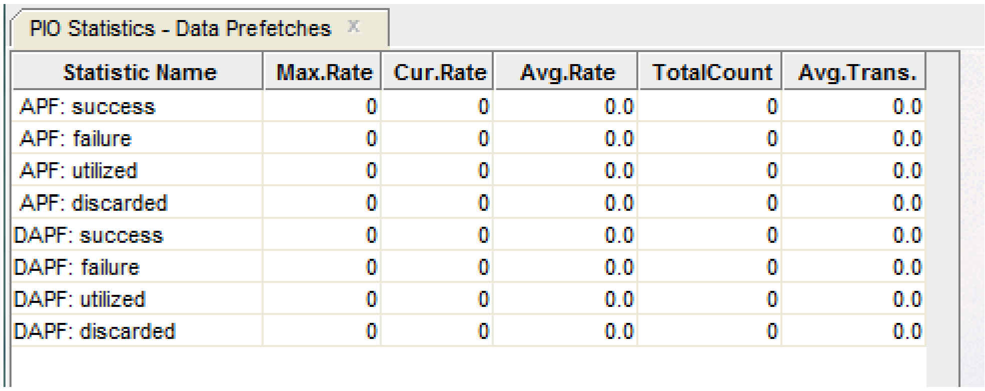
Because the requested page was not found in the global buffer pool, it was synthesized into the pool without being read from disk. This required obtaining appropriate locks on the page.

A requested page that is synthesized is “Read” from a uniform format area and the clump is unallocated.

Contents

13.7 PIO Statistics – Data Prefetches

The ***PIO Statistics – Data Prefetches*** panel provides statistics on how buffer data page pre-fetch requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
APF: success	0	0	0.0	0	0.0
APF: failure	0	0	0.0	0	0.0
APF: utilized	0	0	0.0	0	0.0
APF: discarded	0	0	0.0	0	0.0
DAPF: success	0	0	0.0	0	0.0
DAPF: failure	0	0	0.0	0	0.0
DAPF: utilized	0	0	0.0	0	0.0
DAPF: discarded	0	0	0.0	0	0.0

Figure 55 - PIO Statistics Data Prefetches Pane

The following sections detail the information presented in this panel.

13.7.1 APF: success

The number of times the “Asynchronous Pre-Fetch” (APF) feature operation initiated a buffer fetch attempt that was successful in fetching the buffer.

13.7.2 APF: failure

The number of times an APF-initiated buffer fetch attempt failed due to locking conflicts.

13.7.3 APF: utilized

If a buffer is fetched by APF, it is marked APF FETCHED. If the buffer is accessed again, then this statistic is incremented to indicate that pre-fetching the buffer by APF was worthwhile. The statistic is incremented only the first time the buffer is accessed.

13.7.4 APF: discarded

If an APF-fetched buffer is thrown out of the buffer pool without being accessed, then this statistic is incremented to indicate that fetching the buffer was not worthwhile.

13.7.5 DAPF: success

The number of times the “Detected Asynchronous Pre-Fetch” (DAPF) feature operation initiated a buffer fetch attempt that was successful in fetching the buffer. This is possible only if an EX lock can be obtained for the whole buffer. A successful DAPF fetch may actually issue an asynchronous read or simply ensure that a buffer that was already in the buffer pool now moves to the front of the LRU queue. In the latter case, the buffer then becomes least likely to be thrown out of the buffer pool.

13.7.6 DAPF: failure

The number of times a DAPF-initiated buffer fetch attempt failed due to locking conflicts. If an EX lock cannot be obtained on the whole buffer, then the fetch attempt will fail.

13.7.7 DAPF: utilized

If a buffer is fetched by DAPF, it is marked DAPF FETCHED. If the buffer is accessed again, then this statistic is incremented to indicate that pre-fetching the buffer by DAPF was worthwhile. The statistic is incremented only the first time the buffer is accessed.

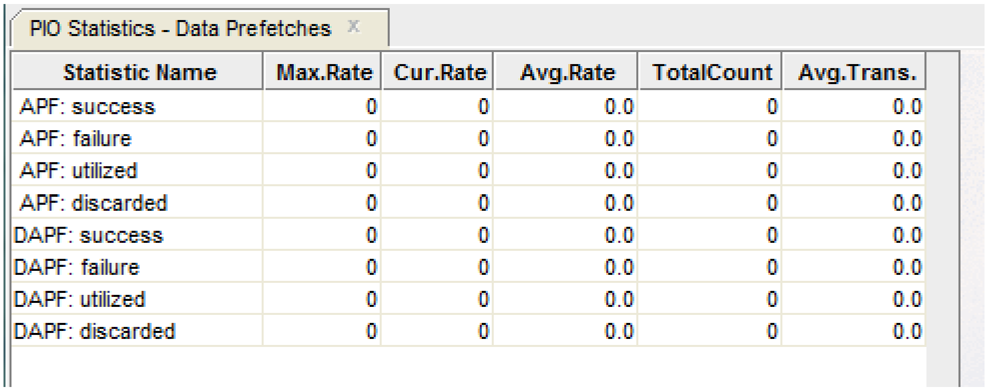
13.7.8 DAPF: discarded

If a DAPF-fetched buffer is thrown out of the buffer pool without being accessed, then this statistic is incremented to indicate that fetching the buffer was not worthwhile.

Contents

13.8 PIO Statistics – SPAM Prefetches

The ***PIO Statistics – SPAM Prefetches*** panel provides statistics on how SPAM page pre-fetch requests are handled by the physical-I/O (PIO) sub-system.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
APF: success	0	0	0.0	0	0.0
APF: failure	0	0	0.0	0	0.0
APF: utilized	0	0	0.0	0	0.0
APF: discarded	0	0	0.0	0	0.0
DAPF: success	0	0	0.0	0	0.0
DAPF: failure	0	0	0.0	0	0.0
DAPF: utilized	0	0	0.0	0	0.0
DAPF: discarded	0	0	0.0	0	0.0

Figure 56 - PIO Statistics SPAM Prefetches Pane

The following sections detail the information presented in this panel.

13.8.1 APF: success

The number of times the “Asynchronous Pre-Fetch” (APF) feature operation initiated a buffer fetch attempt that was successful in fetching the buffer.

13.8.2 APF: failure

The number of times an APF-initiated buffer fetch attempt failed due to locking conflicts.

13.8.3 APF: utilized

If a buffer is fetched by APF, it is marked APF FETCHED. If the buffer is accessed again, then this statistic is incremented to indicate that pre-fetching the buffer by APF was worthwhile. The statistic is incremented only the first time the buffer is accessed.

13.8.4 APF: discarded

If an APF-fetched buffer is thrown out of the buffer pool without being accessed, then this statistic is incremented to indicate that fetching the buffer was not worthwhile.

13.8.5 DAPF: success

The number of times the “Detected Asynchronous Pre-Fetch” (DAPF) feature operation initiated a buffer fetch attempt that was successful in fetching the buffer. This is possible only if an EX lock can be obtained for the whole buffer. A successful DAPF fetch may actually issue an asynchronous read or simply ensure that a buffer that was already in the buffer pool now moves to the front of the LRU queue. In the latter case, the buffer then becomes least likely to be thrown out of the buffer pool.

13.8.6 DAPF: failure

The number of times a DAPF-initiated buffer fetch attempt failed due to locking conflicts. If an EX lock cannot be obtained on the whole buffer, then the fetch attempt will fail.

13.8.7 DAPF: utilized

If a buffer is fetched by DAPF, it is marked DAPF FETCHED. If the buffer is accessed again, then this statistic is incremented to indicate that pre-fetching the buffer by DAPF was worthwhile. The statistic is incremented only the first time the buffer is accessed.

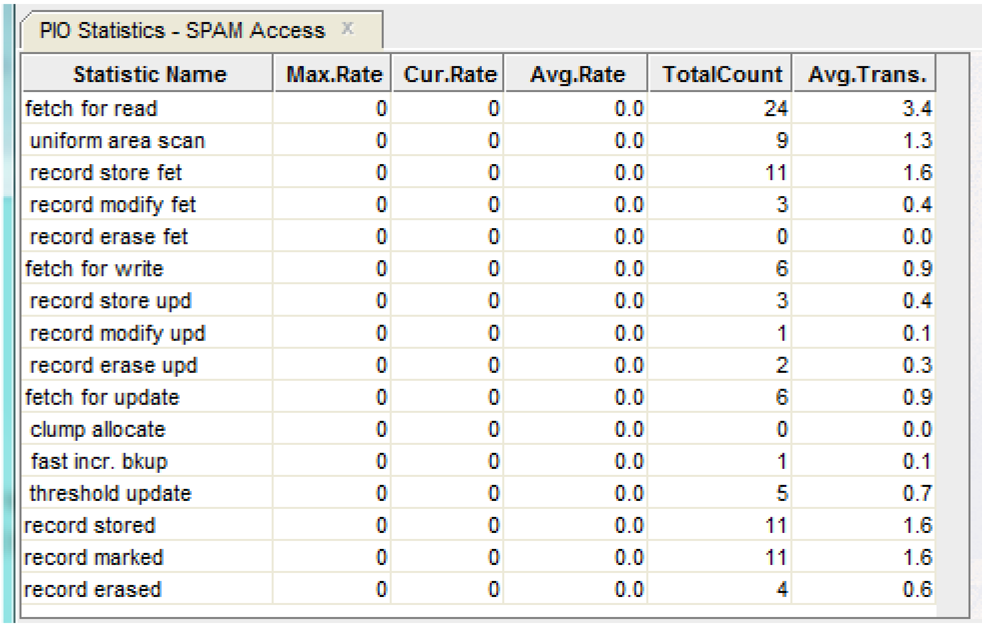
13.8.8 DAPF: discarded

If a DAPF-fetched buffer is thrown out of the buffer pool without being accessed, then this statistic is incremented to indicate that fetching the buffer was not worthwhile.

Contents

13.9 PIO Statistics – SPAM Access

The ***PIO Statistics – SPAM Access*** panel displays statistics about why the SPAM page was accessed. You can identify excessive SPAM fetches by comparing the **record store fet** field to the **record store upd** and **record stored** fields.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
fetch for read	0	0	0.0	24	3.4
uniform area scan	0	0	0.0	9	1.3
record store fet	0	0	0.0	11	1.6
record modify fet	0	0	0.0	3	0.4
record erase fet	0	0	0.0	0	0.0
fetch for write	0	0	0.0	6	0.9
record store upd	0	0	0.0	3	0.4
record modify upd	0	0	0.0	1	0.1
record erase upd	0	0	0.0	2	0.3
fetch for update	0	0	0.0	6	0.9
clump allocate	0	0	0.0	0	0.0
fast incr. bkup	0	0	0.0	1	0.1
threshold update	0	0	0.0	5	0.7
record stored	0	0	0.0	11	1.6
record marked	0	0	0.0	11	1.6
record erased	0	0	0.0	4	0.6

Figure 57 - PIO Statistics – SPAM Access Pane

The following sections detail the information presented in this panel.

13.9.1 fetch for read

The total number of times the SPAM page was fetched for retrieval.

The SPAM page is fetched for four sub-categories: **uniform area scan**, **record store**, **record modify** and **record erase**.

13.9.2 uniform area scan

The total number of times the SPAM page was fetched for retrieval during a uniform area scan operation. This is primarily used to check if SPAM thresholds need to be adjusted.

13.9.3 record store fet

The total number of times the SPAM page was fetched for retrieval during a record store operation. This is primarily used to check if SPAM thresholds need to be adjusted. Typically, this is the most common reason for accessing a SPAM page.

13.9.4 record modify fet

The total number of times the SPAM page was fetched for retrieval during a record modification operation. This is primarily used to check if SPAM thresholds need to be adjusted. The value of this field is typically “0” since SPAM pages are only occasionally fetched for retrieval during a record modification operation.

13.9.5 record erase fet

The total number of times the SPAM page was fetched for retrieval during a record erase operation. This is primarily used to check if SPAM thresholds need to be adjusted. The value of this field is typically “0” since SPAM pages are only occasionally fetched for retrieval during a record erase operation.

13.9.6 fetch for write

The total number of times the SPAM page was fetched for update.

This field is always the same as the **fetch for update** field.

13.9.7 record store upd

The total number of times the SPAM page was fetched for update during a record store operation. This is primarily used to modify the SPAM thresholds.

13.9.8 record modify upd

The total number of times the SPAM page was fetched for update during a record modification operation. This is primarily used to modify the SPAM thresholds.

13.9.9 record erase upd

The total number of times the SPAM page was fetched for update during a record erase operation. This is primarily used to modify the SPAM thresholds.

13.9.10 fetch for update

The total number of times the SPAM page was fetched for update.

This field is always the same as the **fetch for write** field.

13.9.11 clump allocate

The total number of times the SPAM page was updated for a clump allocation operation.

13.9.12 fast incr bkup

The total number of times the SPAM page was updated for a “Fast Incremental Backup” feature modification. A large value in this field may indicate that the “Fast Incremental Backup” feature is not providing any benefit and may actually degrading runtime through-put.

13.9.13 threshold update

The total number of times the SPAM page was updated to change a data page's threshold information.

13.9.14 record stored

The number of records stored in the database.

13.9.15 record marked

The number of records marked. A record is marked when it is modified or it is erased, but not when it is stored.

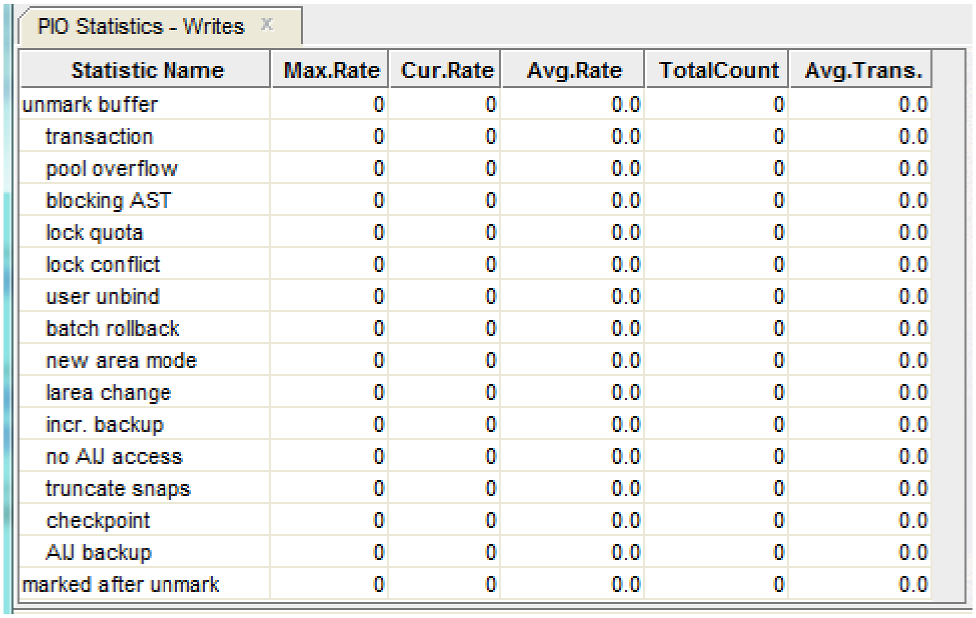
13.9.16 record erased

The number of records erased from the database.

Contents

13.10 PIO Statistics – Data Writes

The ***PIO Statistics – Data Writes*** panel provides information concerning data file writes and buffer unmarking activity (buffer writes to the disk).



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
unmark buffer	0	0	0.0	0	0.0
transaction	0	0	0.0	0	0.0
pool overflow	0	0	0.0	0	0.0
blocking AST	0	0	0.0	0	0.0
lock quota	0	0	0.0	0	0.0
lock conflict	0	0	0.0	0	0.0
user unbind	0	0	0.0	0	0.0
batch rollback	0	0	0.0	0	0.0
new area mode	0	0	0.0	0	0.0
larea change	0	0	0.0	0	0.0
incr. backup	0	0	0.0	0	0.0
no AIJ access	0	0	0.0	0	0.0
truncate snaps	0	0	0.0	0	0.0
checkpoint	0	0	0.0	0	0.0
AIJ backup	0	0	0.0	0	0.0
marked after unmark	0	0	0.0	0	0.0

Figure 58 - PIO Statistics Data Writes Pane

The following sections detail the information presented in this panel.

13.10.1 unmark buffer

This field is incremented by one each time a modified buffer is written back to disk.

Its value should be equal to the sum of the14 following fields. These fields as well as the SPAM page field provide further detail about buffer unmark activity. Write operations of buffers that contain SPAM pages are included in the total although they are also counted separately by the SPAM page field.

Note that unmarking one buffer may entail more than one I/O.

13.10.2 transaction

This field is incremented by one for each modified buffer that is written back to disk as a result of a COMMIT or ROLLBACK statement.

13.10.3 pool overflow

This field is incremented by one for each modified buffer that is written back to disk as a result of a request to read in a new page from disk.

13.10.4 blocking AST

This field is incremented by one for each modified buffer that is written back to disk in response to a blocking AST (that is, a page in the buffer was requested by another user).

13.10.5 lock quota

This field is incremented by one for each modified buffer that is written back to disk due to a user reaching his lock quota.

13.10.6 lock conflict

This field is incremented by one for each modified buffer that is written back to disk to reduce the possibility of a deadlock when Oracle Rdb discovers a lock conflict.

13.10.7 user unbind

This field is incremented by one for each modified buffer that is written back to disk as a result of a user's detaching from the database. This includes database detaches that occur as part of an Oracle RMU operation.

13.10.8 batch rollback

This field is incremented by one for each modified buffer that is written back to disk because of a failed or rolled back batch-update transaction.

13.10.9 new area mode

This field is incremented by one for each modified buffer that is written back to disk as a result of a physical area being readied in a new lock mode.

13.10.10 larea change

This field is incremented by one for each modified buffer that is written back to disk as a part of creating, deleting, or extending a logical area.

13.10.11 incr backup

This field is incremented by one for each modified buffer that is written back to disk to support the requirements of the “Fast Incremental Backup“ feature. These buffers are always SPAM page buffers.

13.10.12 no AIJ access

This field is incremented by one for each modified buffer that is written back to disk as a safety precaution after the failure of an I/O to the after-image journal. The buffers are unmarked under such circumstances only when the “Fast Commit“ feature for transaction processing is enabled to ensure that all committed data is safely in the database.

13.10.13 truncate snaps

This field is incremented by one for each modified buffer that is written back to disk as a part of an online snapshot file truncation.

13.10.14 checkpoint

This field is incremented by one for each modified buffer that is written back to disk as a result of a checkpoint.

13.10.15 AIJ backup

This field is incremented by one for each modified buffer that is written back to disk to optimize the performance of the quiet-point backup of the after-image journal.

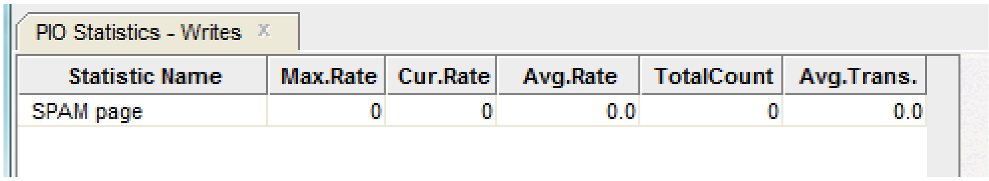
13.10.16 marked after unmark

This field is incremented by one for each modified buffer written back to disk and immediately read again for modification, potentially wasting IO.

Contents

13.11 PIO Statistics – SPAM Writes

The ***PIO Statistics – SPAM Writes*** panel provides information concerning SPAM file writes.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
SPAM page	0	0	0.0	0	0.0

Figure 59 - PIO Statistics Spam Writes Pane

The following sections detail the information presented in this panel.

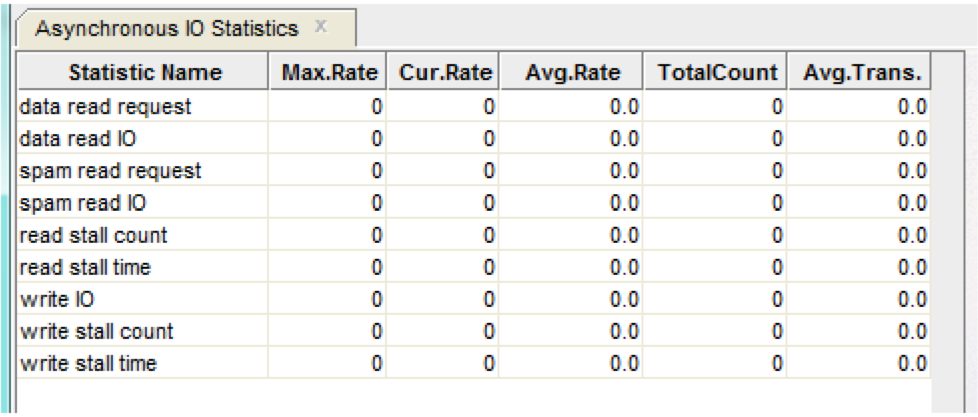
13.11.1 SPAM page

This field is incremented by one each time a modified buffer that contains a space management (SPAM) page is written back to disk. These write operations are also included in the “unmark buffer” field.

Contents

13.12 Asynchronous IO Statistics Panel

The ***Asynchronous IO* Statistics** panels displays information concerning asynchronous reads and writes to the database files. The stall times are displayed in hundredths of a second.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
data read request	0	0	0.0	0	0.0
data read IO	0	0	0.0	0	0.0
spam read request	0	0	0.0	0	0.0
spam read IO	0	0	0.0	0	0.0
read stall count	0	0	0.0	0	0.0
read stall time	0	0	0.0	0	0.0
write IO	0	0	0.0	0	0.0
write stall count	0	0	0.0	0	0.0
write stall time	0	0	0.0	0	0.0

Figure 60 - Asynchronous IO Statistics Pane

The following sections detail the information presented in this panel.

13.12.1 data read request

The number of requests issued to the PIO subsystem to read data pages asynchronously.

13.12.2 data read IO

The number of asynchronous read I/O operations done by the PIO sub-system to get data pages from the disk.

13.12.3 spam read request

The number of requests issued to the PIO subsystem to read SPAM pages asynchronously.

13.12.4 spam read IO

The number of asynchronous read I/O operations done by the PIO sub-system to get SPAM pages from the disk.

13.12.5 read stall count

The total number of times the PIO sub-system had to wait for the completion of asynchronous read I/O operations.

13.12.6 read stall time

The total time in hundredths of a second spent waiting for asynchronous read requests to complete. A high number means the number of buffers being pre-fetched is too low. You can specify that more buffers be pre-fetched by specifying a higher value with the RDM$BIND_APF_DEPTH logical name.

13.12.7 write IO

The number of asynchronous writes done by the PIO sub-system to write data and SPAM pages to the disk.

13.12.8 write stall count

The total number of times the PIO subsystem had to wait for the completion of asynchronous write I/O. This statistic includes both asynchronous batch write operations as well as asynchronous I/O initiated during a transaction commit operation when the “Fast Commit“ feature is not active.

13.12.9 write stall time

The total time in hundredths of a second spent waiting for asynchronous write requests to complete. An excessively high number often indicates that the number of clean buffers being maintained at the end of a process's least recently used queue of buffers for replacement is too low. You can increase the number of clean buffers being maintained at the end of a process's least recently used queue of buffers for replacement by specifying a higher value with the RDM$BIND_CLEAN_BUF_CNT logical name.

Contents

Chapter 14
File IO

The File IO Statistics panels include:

- File IO Overview
- IO Statistic (By File)
- IO Stall Time

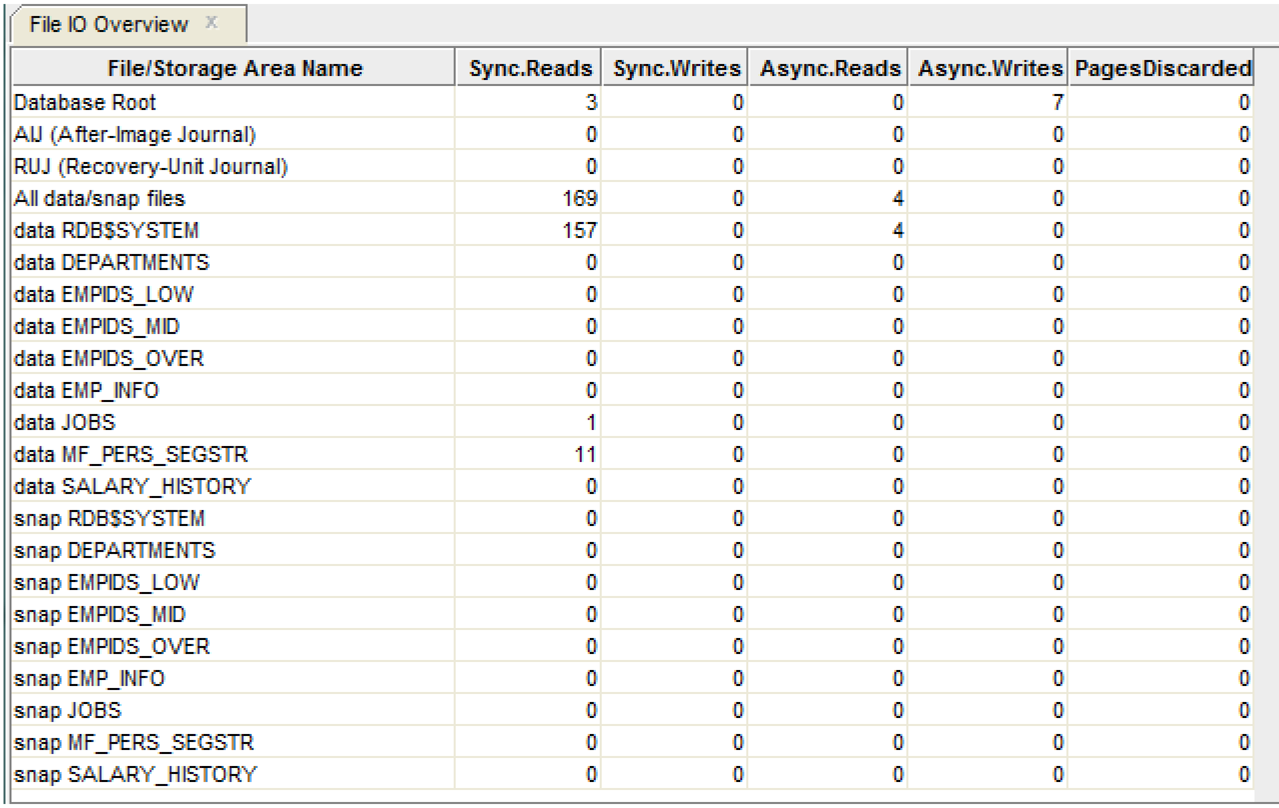
14.1 File IO Overview

This panel shows a summary comparison of I/O activity for all database files, including the rootfile, after-image journals, RUJ journals, data and snapshot storage areas. The types of database files can be filtered by name and file type, and the statistic information displayed on this panel can be sorted using various criteria.

This panel displays comparison information about I/O activities that are specific to storage areas and snapshot files. This information is vital in determining which storage areas have the most I/O activity, and analyzing the validity of storage area partitioning.

Note that, unlike the ***File Locking Overview*** panel, the ***File IO Overview*** panel does display information about after-image journals and RUJ journals.

Also, information on the statistic information files (.RDS) is not available.



File/Storage Area Name	Sync.Reads	Sync.Writes	Async.Reads	Async.Writes	PagesDiscarded
Database Root	3	0	0	7	0
AIJ (After-Image Journal)	0	0	0	0	0
RUJ (Recovery-Unit Journal)	0	0	0	0	0
All data/snap files	169	0	4	0	0
data RDB$SYSTEM	157	0	4	0	0
data DEPARTMENTS	0	0	0	0	0
data EMPIDS_LOW	0	0	0	0	0
data EMPIDS_MID	0	0	0	0	0
data EMPIDS_OVER	0	0	0	0	0
data EMP_INFO	0	0	0	0	0
data JOBS	1	0	0	0	0
data MF_PERS_SEGSTR	11	0	0	0	0
data SALARY_HISTORY	0	0	0	0	0
snap RDB$SYSTEM	0	0	0	0	0
snap DEPARTMENTS	0	0	0	0	0
snap EMPIDS_LOW	0	0	0	0	0
snap EMPIDS_MID	0	0	0	0	0
snap EMPIDS_OVER	0	0	0	0	0
snap EMP_INFO	0	0	0	0	0
snap JOBS	0	0	0	0	0
snap MF_PERS_SEGSTR	0	0	0	0	0
snap SALARY_HISTORY	0	0	0	0	0

Figure 61 - File IO Overview Pane

The following sections detail the information presented in columns this table.

14.1.1 File/Storage Area Name

This column identifies the particular database rootfile, live or snapshot storage area, or the after-image journal and RUJ journal. Note that the summation information for all live and snapshot storage areas is identified as “All data/snap files”.

14.1.2 Sync Reads

The number of synchronous read QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases and snapshot files. This operation reads database pages synchronously from the database.

14.1.3 Sync Writes

The number of synchronous write QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot files (.SNP). This operation writes modified database pages synchronously back to the database.

14.1.4 Async Reads

The number of asynchronous read QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases (.RDA) and snap-shot (.SNP) files. This operation reads database pages asynchronously from the database.

14.1.5 Asynch Writes

The number of asynchronous write QIOs (queued I/O requests) issued to the database storage area for single-file and multi-file databases (.RDA) and snapshot files (.SNP). This operation writes modified database pages asynchronously back to the database.

14.1.6 Pages Discarded

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.

Note:

- ✓ A discarded page is an indication of wasted resources and decreased through-put and should be more closely examined. Ideally, the "Pages Discarded" column should always display the value "0".

14.1.7 Information

The ***File IO Overview*** panel shows the synchronous and asynchronous read and write I/O counts for all storage areas, the after-image journal (.AIJ), the recovery-unit journal (.RUJ), the database root file and, finally, all data and snapshot areas combined.

In the default display configuration of the File IO Overview display, the first five rows displayed are the database root (.RDB) file, the after- image journal (.AIJ) file, the recovery-unit journal (.RUJ) file and the total read and write statistics for all the database storage (.RDA) and snapshot (.SNP) areas.

Following the first five rows is the list of database storage areas, identified by the prefix 'data', followed by the list of snapshot areas, identified by the prefix 'snap'. Storage areas that are added to the database are automatically shown in the display.

Using the ***File IO Overview*** panel makes it easier for you to identify the set of storage areas that are performing an excessive number of synchronous I/O operations. For example, a large number of synchronous write I/O operations should cause you to examine the storage area thresholds for this area to determine the cause of the problem.

When Oracle Rdb is attempting to store a record, it sometimes reads a target page based on an acceptable threshold on the SPAM page, but determines after reading the data page that not enough space is available on the page to store the record.

This behavior causes a high ratio of pages checked to records stored on the Record Statistics panel. This behavior can potentially cause very slow insert performance if a large number of pages needs to be examined for each record stored.

If you detect that Oracle Rdb is checking excessive pages while attempting to insert a record, it is very difficult to determine which storage area is exhibiting the behavior and the exact cause of the problem (which, for example, could be locked free space, incorrect SPAM thresholds, unique indexes, or estimated record sizes).

For high volume transaction processing applications or applications with a large number of storage areas, it is not practical to manually examine the IO Statistics (by file) panel for each storage area in the database, trying to identify the particular storage area with excessive read I/O operations. This is where the ***File IO Overview*** panel comes in useful.

Contents

14.2 IO Statistic (By File)

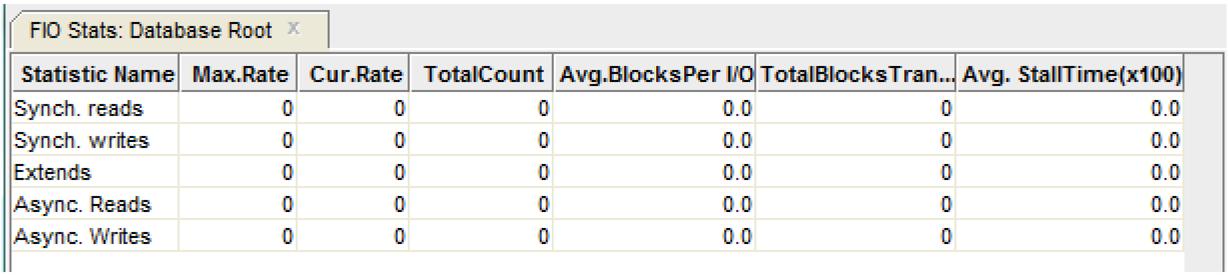
This panel allows you to display I/O statistics for each of individual files that form the database.

When you select **IO Statistics (by file)** from the display menu, Oracle Rdb displays a list of files that comprise the database and for which you can choose to view statistics.

With the exception of the **all data/snap files** panel, each panel shows the I/O activity for a specific database file. The **all data/snap files** panel shows a summation of I/O activity for all data and snapshot files.

The information in this panel applies from the time that your ORCM **RMU Statistics** session began, or since the accumulators were last reset (using the **Reset** menu option).

Note that the accumulators on this panel can be reset using the **Reset** menu option.



Statistic Name	Max.Rate	Cur.Rate	TotalCount	Avg.BlocksPer I/O	TotalBlocksTran...	Avg. StallTime(x100)
Synch. reads	0	0	0	0.0	0	0.0
Synch. writes	0	0	0	0.0	0	0.0
Extends	0	0	0	0.0	0	0.0
Async. Reads	0	0	0	0.0	0	0.0
Async. Writes	0	0	0	0.0	0	0.0

Figure 62 - Example File IO Pane

The following sections detail the information presented in the rows of this table.

14.2.1 Synch reads

Database pages read synchronously from the database.

14.2.2 Synch writes

Database pages written synchronously back to the database.

14.2.3 Extends

File extension operations issued to the database.

14.2.4 Asynch reads

Database pages read asynchronously from the database.

14.2.5 Asynch writes

Database pages written asynchronously back to the database.

The following sections detail the information presented in the column of this table.

14.2.6 Max Rate

The maximum occurrence-per-second rate of synchronous read QIOs (queued I/O requests) issued to the file being displayed.

14.2.7 Cur Rate

The current occurrence-per-second rate of synchronous read QIOs (queued I/O requests) issued to the file being displayed.

14.2.8 Total Count

The total number of the specific I/O operations made to the file being displayed.

14.2.9 Avg Blocks per IO

The average number of blocks transferred IO for the specified operation issued to the file being displayed.

14.2.10 Avg Stall Time(x100)

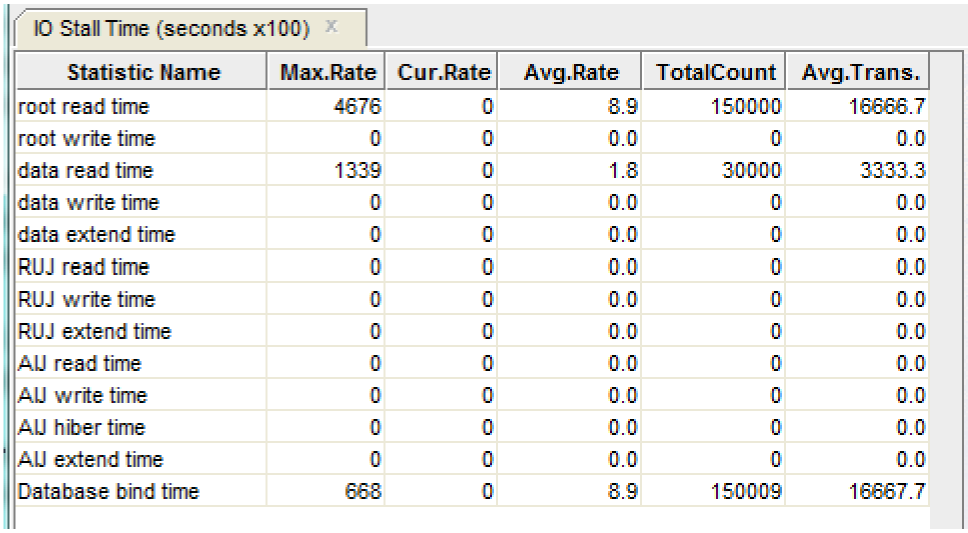
The time in hundredths of a second spent waiting for the specified operation issued to the file being displayed.

Contents

14.3 IO Stall Time

This panel shows a summary of I/O stall activity. All times are displayed in hundredths of a second.

This panel is important for analyzing disk device controller utilization as well as backplane and inter-connect effectiveness.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
root read time	4676	0	8.9	150000	16666.7
root write time	0	0	0.0	0	0.0
data read time	1339	0	1.8	30000	3333.3
data write time	0	0	0.0	0	0.0
data extend time	0	0	0.0	0	0.0
RUJ read time	0	0	0.0	0	0.0
RUJ write time	0	0	0.0	0	0.0
RUJ extend time	0	0	0.0	0	0.0
AIJ read time	0	0	0.0	0	0.0
AIJ write time	0	0	0.0	0	0.0
AIJ hiber time	0	0	0.0	0	0.0
AIJ extend time	0	0	0.0	0	0.0
Database bind time	668	0	8.9	150009	16667.7

Figure 63 - IO Stall Time Pane

The following sections detail the information presented in this panel.

14.3.1 root read time

This field gives the time in hundredths of a second spent reading the database root (.RDB) file.

This statistic field includes both synchronous and asynchronous I/O read stall durations.

14.3.2 root write time

This field gives the time in hundredths of a second spent writing to the database root (.RDB) file.

This statistic field includes both synchronous and asynchronous I/O write stall durations.

14.3.3 data read time

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.

14.3.4 data write time

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.

14.3.5 data extend time

This field gives the time in hundredths of a second spent extending the database rootfile (.RDB), storage area (.RDA) and snapshot (.SNP) files. A very high number often indicates a full disk or disk fragmentation.

14.3.6 RUJ read time

This field gives the time in hundredths of a second spent reading records from the recovery-unit journals (.RUJ) during verb or transaction rollback. 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.

14.3.7 RUJ write time

This field gives the time in hundredths of a second spent writing records to the recovery-unit journals (.RUJ). 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.

14.3.8 RUJ extend time

This field gives the time in hundredths of a second spent extending the recovery-unit journals (.RUJ). An excessively high number often indicates a full disk or disk fragmentation.

14.3.9 AIJ extend time

This field gives the time in hundredths of a second spent reading records from the after-image journal. AIJ reads almost always occur as the result of opening the afterimage journal, or performing failed process recovery.

This statistic field includes both synchronous and asynchronous I/O read stall durations.

14.3.10 AIJ write time

This field gives the time in hundredths of a second spent writing to the after-image journal.

This statistic field includes both synchronous and asynchronous I/O write stall durations.

14.3.11 AIJ hiber time

This field shows the stall time spent hibernating for the AIJ I/O to complete.

14.3.12 AIJ read time

This field gives the time in hundredths of a second spent extending the after-image journal. An excessively high number often indicates a full disk or disk fragmentation, or may be an indication of the need for a larger allocation.

Use the JOURNAL ALLOCATION IS clause of the SQL ALTER DATABASE statement to specify a larger allocation does that journal extension operations are avoided.

14.3.13 database bind time

This field gives the length of time it takes users to attach to the database.

Contents

Chapter 15
Index Information

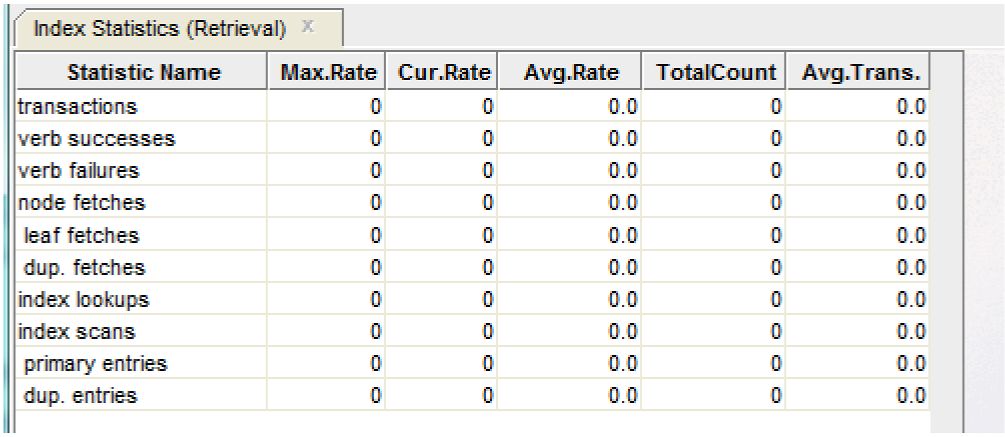
The following index statistics are provided:

- Index Statistics (Retrieval)
- Index Statistics (Insertion)
- Index Statistics (Removal)
- Hash Index Statistics

15.1 Index Statistics (Retrieval)

This panel monitors how much retrieval activity is taking place in a database's sorted indexes.

Oracle Rdb often uses direct index lookups and index scans to access records in the database. This panel monitors these operations as well as the number of index nodes fetched.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
transactions	0	0	0.0	0	0.0
verb successes	0	0	0.0	0	0.0
verb failures	0	0	0.0	0	0.0
node fetches	0	0	0.0	0	0.0
leaf fetches	0	0	0.0	0	0.0
dup. fetches	0	0	0.0	0	0.0
index lookups	0	0	0.0	0	0.0
index scans	0	0	0.0	0	0.0
primary entries	0	0	0.0	0	0.0
dup. entries	0	0	0.0	0	0.0

Figure 64 - Index Statistics (Retrieval) Pane

The following sections detail the information presented in this panel.

15.1.1 transactions

This field gives the number of completed verbs that returned a successful status code.

15.1.2 verb successes

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 statement to treat the entire block as a single verb.

15.1.3 verb failures

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 DDL statement.

Notes:

- ✓ In the case of cursors and scans, reaching the end-of-stream always results in a verb failure.
- ✓ 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.

15.1.4 node fetches

This field gives the number of times Oracle Rdb fetched an index node during index retrievals. This number includes the number of leaf nodes and duplicate nodes fetched. Therefore, the calculation for the number of upper-level index nodes accessed is:

node fetches minus the **sum** of the **leaf fetches** and **dup fetches.**

The result can indicate the depth of the database indexes.

15.1.5 leaf fetches

This field gives the number of times Oracle Rdb fetched bottom level (leaf) nodes during index retrievals. This number, along with the **index scans** field, can indicate the length of scans in terms of index nodes accessed. There is one leaf node fetch for each **index lookups** retrieval.

15.1.6 dup fetches

This field gives the number of times Oracle Rdb fetched a duplicate node (as opposed to a leaf node) during index retrievals. This number can indicate the lengths of duplicate node chains in the database indexes. When a duplicate node is retrieved, the operation always includes one leaf fetch.

15.1.7 index lookups

This field gives the number of direct single-key retrievals performed on the database indexes. This statistic shows up only on unique key retrievals and not on duplicate key retrievals.

15.1.8 index scans

This field gives the number of scans, or range retrievals, performed on the database indexes. In an index scan, Oracle Rdb searches an index from top to bottom to find the starting point (low value) of the retrieval. Oracle Rdb then searches the bottom level nodes of the index, including duplicate nodes, until the scan's end condition is met.

15.1.9 primary entries

This field gives the number of unique keys found during the index scan.

15.1.10 dup entries

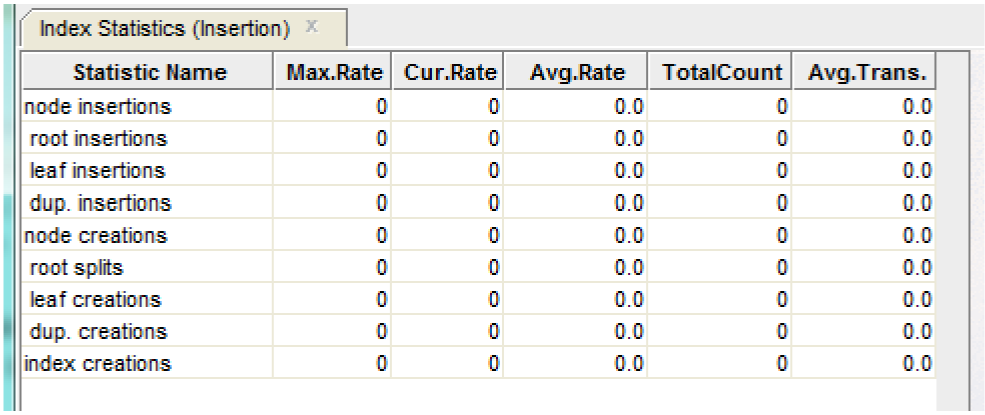
This field gives the number of duplicate keys found during the index scans. If an index has two entries with the same key value, the first one is a primary entry and the second is a duplicate entry.

Contents

15.2 Index Statistics (Insertion)

This panel monitors the update activity of a database's sorted indexes during insertions; that is, when you store or modify an index key field or when you use the SQL CREATE INDEX statement on a table. This panel also indicates in which type of index node the insertions occur and displays node creations by node type.

By examining this panel, you can monitor how a database balances its sorted indexes after insertions into the database.



Index Statistics (Insertion)

Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
node insertions	0	0	0.0	0	0.0
root insertions	0	0	0.0	0	0.0
leaf insertions	0	0	0.0	0	0.0
dup. insertions	0	0	0.0	0	0.0
node creations	0	0	0.0	0	0.0
root splits	0	0	0.0	0	0.0
leaf creations	0	0	0.0	0	0.0
dup. creations	0	0	0.0	0	0.0
index creations	0	0	0.0	0	0.0

Figure 65 - Index Statistics (Insertion) Pane

The following sections detail the information presented in this panel.

15.2.1 node insertions

The number of index entries inserted into all index nodes. This number includes root, leaf, and duplicate entries within user-and system-defined indexes.

This number is greater than the number of records being stored in the database because it usually takes one to two insertions into an index for each record for each index.

The calculation of node insertions minus the sum of the root, leaf, and duplicate insertions yields the number of entries inserted into mid-level nodes. This number and the **root insertions** field indicate sorted balancing activity.

15.2.2 root insertions

The number of entries inserted into the root (top-level) index nodes.

The number of insertions should be small except for when you load the database. Also, if an index consists of only one node, insertions into this node are not included in this field, but are included in the **leaf insertions** field.

15.2.3 leaf insertions

The number of unique keys inserted into the database's indexes. This field indicates the number of entries inserted into the leaf (bottom-level) index nodes.

15.2.4 dup insertions

The number of duplicate index keys inserted into the database's indexes. There should be a one-to-one correspondence to the number of duplicate records being stored in the tables.

15.2.5 node creations

The total number of index nodes created during insertion of index entries into the index trees. This includes root, leaf, and duplicate nodes created within user- and system-defined indexes.

Nodes are created three ways:

- When an index is first defined
- When a node cannot accommodate an insertion, causing it to overflow into a new node (node splitting)
- When the first duplicate for a particular key is inserted into an index, causing a duplicate node to be created

The total number of nodes created and the associated fields should be relatively small, except for an initial load of the database with indexes already defined, or for creation of indexes on already-stored data.

15.2.6 root splits

The number of times the root nodes have split because they overflowed after an insertion. A root node split causes the index to grow by one level. A parent node must be created to point to the two halves of the overflowed root node.

Therefore, two nodes are created, the parent node and the node for the second half of the root node. Increasing the number of tree levels means Oracle Rdb must search more index nodes to access a data row. This can result in additional I/O operations.

15.2.7 leaf creations

The number of times a leaf (bottom level) node was created because an existing leaf node had become full and needed to accommodate another unique index key entry.

15.2.8 dup creations

The number of times a duplicate node was created to accommodate more duplicated entries within the duplicate index node or on the first store of a duplicate key entry.

15.2.9 index creations

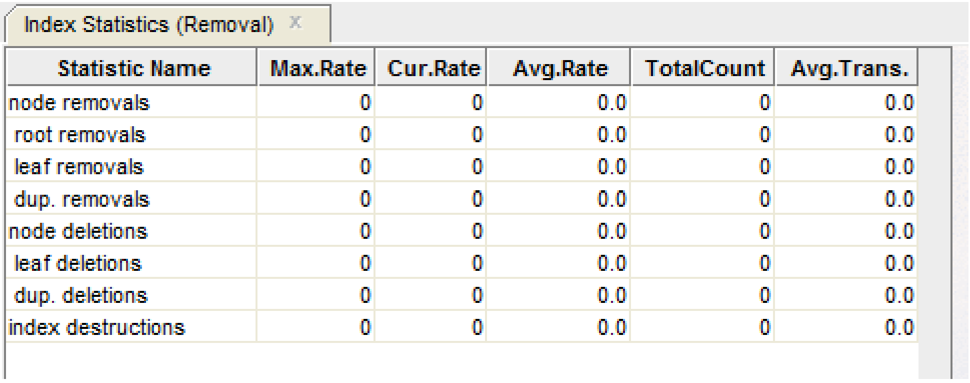
The number of times an index was created on a particular table. This count is the number of CREATE INDEX statements. Also, if an index is partitioned over three areas, for example, there will be a count of three index creations.

Contents

15.3 Index Statistics (Removal)

This panel monitors the update activity of a database's sorted indexes when you perform any removal operation; that is, erase, alter, or modify an index key field or drop or delete an index. This panel indicates from which type of index node the removals occur. It also shows node deletions by node type.

This panel lets you monitor how a database balances its sorted indexes when nodes are removed from the indexes.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
node removals	0	0	0.0	0	0.0
root removals	0	0	0.0	0	0.0
leaf removals	0	0	0.0	0	0.0
dup. removals	0	0	0.0	0	0.0
node deletions	0	0	0.0	0	0.0
leaf deletions	0	0	0.0	0	0.0
dup. deletions	0	0	0.0	0	0.0
index destructions	0	0	0.0	0	0.0

.

Figure 66 - Index Statistics (Removal) Pane

The following sections detail the information presented in this panel.

15.3.1 node removals

The total number of index entries within the root, leaf, and duplicate nodes that have been removed. This removal can be triggered by erasing rows, deleting tables, or deleting indexes. The calculation of node removals minus the sum of the root, leaf, and duplicate node removals yields the number of entries removed from mid-level nodes. A node is not deleted until all its entries are removed.

15.3.2 root removals

The number of index entries removed from a root node due to deletion of entries within lower-level nodes. If an index consists of only one node, removals from this node are not included in this field, but are included in the **leaf removals** field.

15.3.3 leaf removals

The number of unique index keys removed from the leaf nodes during an SQL DELETE operation.

15.3.4 dup removals

The number of duplicate index keys removed from duplicate nodes due to the deletion of duplicate records. This should be a one-to-one correspondence to the number of erased duplicate records within the database.

15.3.5 node deletions

This field gives the total number of index nodes deleted due to an SQL DROP.

When an index is deleted, this number should be equal to the total number of index nodes within the index. This field minus the sum of leaf and duplicate node deletions yields the number of mid-level node deletions.

15.3.6 leaf deletions

The number of leaf (bottom level) nodes deleted from the database's indexes. A leaf node is deleted only when it becomes empty.

15.3.7 dup deletions

The number of duplicate node deletions within the indexes.

15.3.8 index destructions

The number of indexes deleted with an SQL DROP INDEX statement.

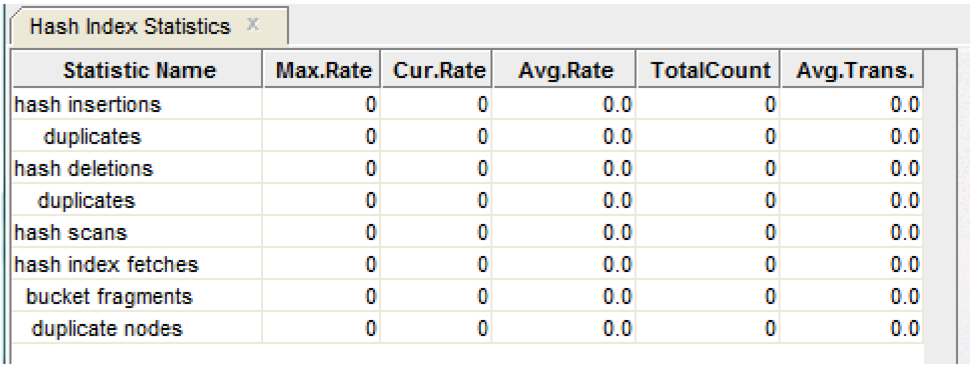
This count will be 1 if the index is not partitioned. If an index that is partitioned over three areas is deleted, for example, then the count will be 3. This count also gives the number of root node deletions.

Contents

15.4 Hash Index Statistics

This panel monitors the update and retrieval activity of a database's hashed indexes.

It indicates the total number of key insertions and deletions. It also indicates the number of scans that were opened. For retrievals (successful fetches), the panel indicates the total number of nodes (either bucket fragments or duplicate nodes) that were fetched.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
hash insertions	0	0	0.0	0	0.0
duplicates	0	0	0.0	0	0.0
hash deletions	0	0	0.0	0	0.0
duplicates	0	0	0.0	0	0.0
hash scans	0	0	0.0	0	0.0
hash index fetches	0	0	0.0	0	0.0
bucket fragments	0	0	0.0	0	0.0
duplicate nodes	0	0	0.0	0	0.0

Figure 67 - Hash Index Statistics Pane

The following sections detail the information presented in this panel.

15.4.1 hash insertions

The number of hash key insertions in the database's hashed indexes. It includes unique key insertions as well as duplicate key insertions.

15.4.2 duplicates

The number of duplicate key updates in the database's hashed indexes.

15.4.3 hash deletions

The number of hash key deletions from the database's hashed indexes. It includes unique key deletions as well as duplicate key deletions.

15.4.4 duplicates

The number of duplicate key deletions in the database's hashed indexes.

15.4.5 hash scans

The number of hashed index scans, including both retrieval and update scans that were opened on the database's hashed indexes. A scan is defined as the sequential processing of the records that meet the search criteria of a query. Hashed scans then refer to the case where duplicate records are returned that meet the search criteria of a query from a scan of the hashed index.

15.4.6 hash index fetches

The number of hashed index nodes that were fetched on a successful search of the database's hashed indexes. This includes fetches of duplicate nodes as well as bucket fragment nodes.

15.4.7 bucket fragments

The number of bucket fragments that were fetched on a successful search of the database's hashed indexes.

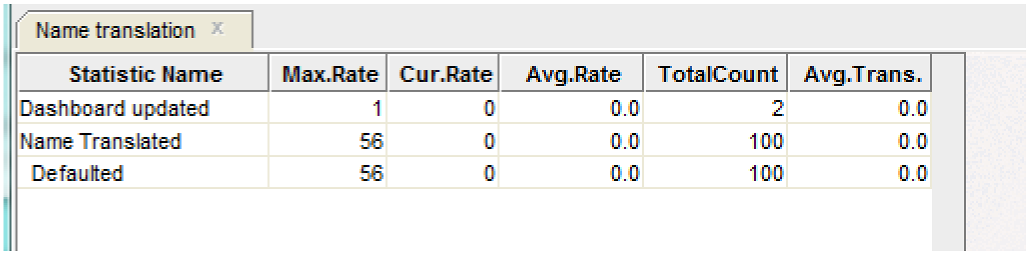
15.4.8 duplicate nodes

The number of duplicate nodes that were fetched on a successful search of the database's hashed indexes.

Contents

Chapter 16
Name Translation

The **Logical Name Translations** panel shows statistics on database dashboard updates and logical name translation. This panel provides a measure of how many logical names are actually being used instead of using the default values stored in the database.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
Dashboard updated	1	0	0.0	2	0.0
Name Translated	56	0	0.0	100	0.0
Defaulted	56	0	0.0	100	0.0

Figure 68 - Logical Name Translations Pane

The following sections detail the information presented in this panel.

16.1.1 dashboard updated

This field displays the number of times a user has received notification that the database dashboard has been updated.

16.1.2 name translated

This field displays the number of times a logical name has been translated.

16.1.3 defaulted

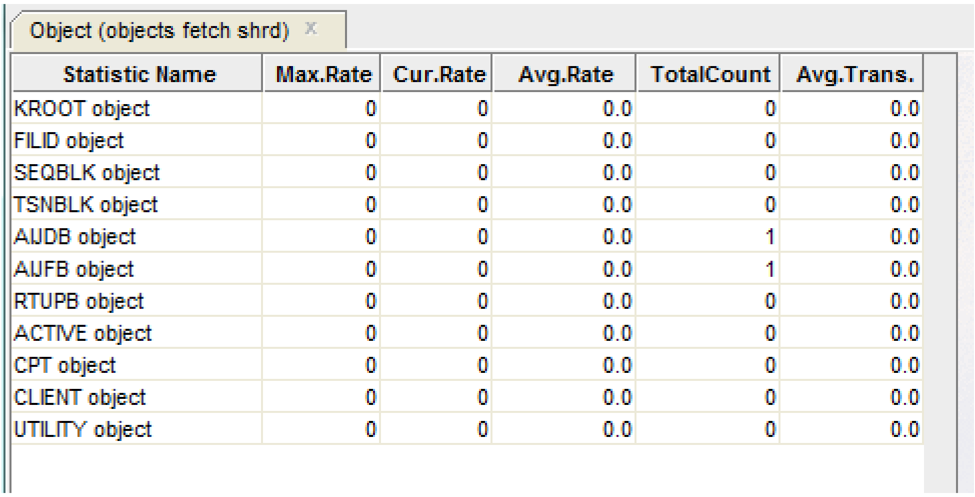
This field displays the number of times the default value was used for a logical name.

Contents

Chapter 17

Objects (One Stat Field)

The ***Objects (One Stat Field)*** panels display information for a specific database root file object.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
KROOT object	0	0	0.0	0	0.0
FILID object	0	0	0.0	0	0.0
SEQBLK object	0	0	0.0	0	0.0
TSNBLK object	0	0	0.0	0	0.0
AIJDB object	0	0	0.0	1	0.0
AIJFB object	0	0	0.0	1	0.0
RTUPB object	0	0	0.0	0	0.0
ACTIVE object	0	0	0.0	0	0.0
CPT object	0	0	0.0	0	0.0
CLIENT object	0	0	0.0	0	0.0
UTILITY object	0	0	0.0	0	0.0

Figure 69 - Example Objects One Field Pane

All the **One Stat Field** panels display a table similar to the example provided above.

The following sections detail the information presented in this panel.

17.1.1 KROOT object

This field displays the database control information that describes all of the other database objects.

17.1.2 FILID object

This field displays the storage area information.

17.1.3 SEQBLK object

This field displays the information on the allocation of transaction sequence numbers.

17.1.4 TSNBLK object

This field displays the information on the last committed transaction.

17.1.5 AIJDB object

This field displays the after-image journal control information.

17.1.6 AIJFB object

This field displays the after-image journal information.

17.1.7 RTUPB object

This field displays information on active users.

17.1.8 ACTIVE object

This field displays information on active transactions.

17.1.9 CPT object

This field displays information on the corrupt page table.

17.1.10 RCACHE object

This field displays information on row caches.

17.1.11 CLIENT object

This field displays client-specific information.

17.1.12 CLTSEQ object

This field displays client “sequence” information.

17.1.13 UTILITY object

This field displays Oracle RMU utility information.

17.2 Individual Panels

Details of the individual panels within this group:

17.2.1 Objects Fetch Shrd Panel

This panel displays the statistics for the “objects fetch shrd” collection category. This category shows the number of objects that are fetched for the shared retrieval operation.

17.2.2 Objects Fetch Excl Panel

This panel displays the statistics for the “objects fetch excl” collection category. This category shows the number of objects fetched for exclusive access with the intention of subsequently being updated. This statistic does not indicate that the object was actually updated.

17.2.3 Objects Refreshed Panel

This panel displays the statistics for the “objects refreshed” collection category. This category shows the number of objects whose information was detected as being stale, so the information was read again from the database root file.

17.2.4 Objects Modified Panel

This panel displays the statistics for the “objects modified” collection category. This category shows the number of objects whose information was modified. Only objects fetched for exclusive access can be modified.

17.2.5 Objects Written Panel

This panel displays the statistics for the “objects written” collection category. This category shows the number of objects whose information was written back to the database root file.

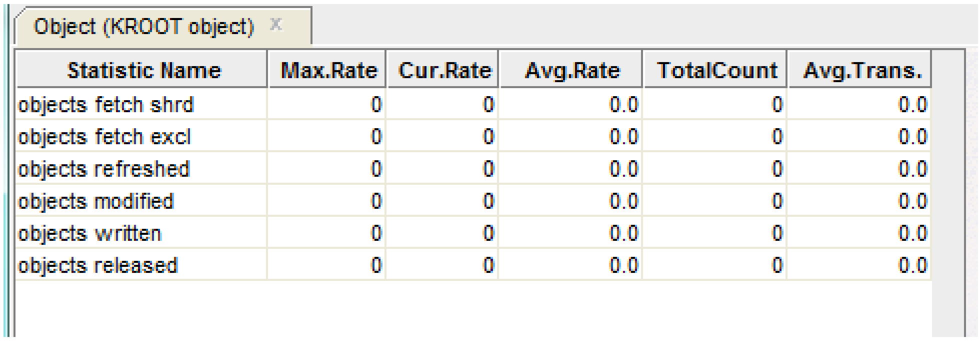
17.2.6 Objects Released Panel

This panel displays the statistics for the “objects released” collection category. This category shows the number of objects whose shared or exclusive access was released to other processes.

Chapter 18

Objects (One Stat Type)

The ***Objects (One Stat Type)*** panels display information for a specific statistic field for all database root file objects.



Statistic Name	Max.Rate	Cur.Rate	Avg.Rate	TotalCount	Avg.Trans.
objects fetch shrd	0	0	0.0	0	0.0
objects fetch excl	0	0	0.0	0	0.0
objects refreshed	0	0	0.0	0	0.0
objects modified	0	0	0.0	0	0.0
objects written	0	0	0.0	0	0.0
objects released	0	0	0.0	0	0.0

Figure 70 - Example Objects One Stat Type Pane

All the **One Stat Type** panels display a table similar to the example provided above.

The following sections detail the information presented in this panel.

18.1.1 objects fetch shrd

This field displays the number of objects that are fetched for the shared retrieval process.

18.1.2 objects fetch 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 actually was updated. This object is sometimes acquired in exclusive mode to serialize access to an operation.

18.1.3 objects refreshed

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.

18.1.4 objects written

This field displays the number of objects whose information was modified. Only objects fetched for exclusive access can be modified.

18.1.5 objects modified

This field displays the number of objects whose information was written back to the database root file.

18.1.6 objects released

This field displays the number of objects whose shared or exclusive access was released to other processes.

18.2 Individual Panels

Details of the individual panels within this group:

18.2.1 KROOT Object Panel

The KROOT object contains the database control information that describes all of the other database objects.

The majority KROOT updates occur as the result of the SQL ALTER DATABASE Statement.

18.2.2 FILID Object Panel

The FILID object contains the storage area information.

18.2.3 SEQBLK Object Panel

The SEQBLK object contains the information on the allocation of sequence numbers such as transaction sequence numbers (TSN) and commit sequence numbers (CSN).

The SEQBLK is also used to serialize access to the TSNBLK locks.

18.2.4 TSNBLK Object Panel

The TSNBLK object contains the information on the last committed transaction.

The number of TSNBLK objects is a function of the maximum number of users in the database; there is one TSNBLK object for every 28 database users (rounded up).

For example, a database containing a maximum of 512 users would contain 19TSNBLK objects.

18.2.5 AIJDB Object Panel

The AIJDB object contains the after-image journal control information.

18.2.6 AIJFB Object Panel

The AIJFB object contains the after-image journal information.

18.2.7 RTUPB Object Panel

The RTUPB object contains information on active users.

18.2.8 ACTIVE Object Panel

The ACTIVE object contains information on active transactions.

18.2.9 CPT Object Panel

The CPT object contains information on the corrupt page table.

18.2.10 RCACHE Object Panel

The RCACHE object contains the row cache information.

18.2.11 CLIENT Object Panel

The CLIENT object contains client-specific information.

18.2.12 CLTSEQ Object Panel

The CLTSEQ object contains the client “sequence” information.

18.2.13 UTILITY Object Panel

The UTILITY object contains information used by the Oracle RMU utility.

Contents

Chapter 19

Stall Messages

The following table describes the possible stall messages, in alphabetical order:

Stall Message	**Description**
binding to database	This message indicates that the process is binding or attaching to the database.
Bugcheck: *name*	This message indicates that the corresponding application process has terminated abnormally and is producing a bugcheck dump for diagnostic analysis.
committing TSN *number : number*	This message indicates that the specified transaction is being committed to the database.
connecting to remote database (*number*)	This message indicates that the AIJ Log Server (“ALS”) process, the AIJ Catch-up Server (“LCS”), AIJ Backup Server (“ABS”) or Database Recovery server (“DBR”) is attaching to the standby database. The *number* value indicates the number of connection attempts remaining before failure.
creating AIJ backup file	This message indicates that the AIJ Backup Server (“ABS”) or RMU Backup After_Journal utility is creating an after-image journal backup file.
creating temporary AIJ journal	This message indicates that the AIJ Backup

	Server (“ABS”) or RMU Backup After_Journal utility is creating a temporary afterimage journal. This occurs when the Fast Commit feature is enabled, or “no quiet point” after-image journal backups are performed.
extending AIJ file	This message displays whenever the after-image journal is logically or physically extended, which should occur infrequently.
extending .RUJ file	This message displays whenever the .RUJ file is physically extended, which should occur infrequently.
extending storage area *number*	This message displays whenever a storage area file (identified by its numeric identifier, which can be determined using the RMU Dump utility) is physically extended. You can determine the numeric identifier for a database's storage areas by using the RMU Dump utility. This message should occur infrequently.
finding buffer for record cache *number*	This message indicates that the specified row cache is being searched for an available buffer.
finding next free buffer	This message indicates that a search for the next available buffer in a row cache is being performed.
hibernating for *number* record cache latch	This message indicates that the latch for the specified row cache could be acquired within a reasonable period of time, so the process will hibernate until the latch is released by the holding process.

hibernating on AIJ I/O completion	This message indicates that the AIJ Log Server (“ALS”) is hibernating until the current asynchronous write I/O operation to the after-image journal completes.
hibernating on AIJ submission	This message indicates that the process has submitted after-image journal information and is hibernating while the AIJ Log Server (“ALS”) processes the information.
hibernating until next checkpoint (*time*)	This message indicates that the Row Cache Server (“RCS”) is hibernating until the next checkpoint, which will occur at the indicated date/time.
Initializing AIJ journal	This message indicates that the current after-image journal is being initialized.
initializing AIJ journal	This message indicates that the after-image journal is being initialized. This occurs when an after-image journal has been backed up.
initializing new AIJ journal	This message indicates that the after-image journal is being initialized. This occurs when an after-image journal is originally created
latching page *number : number* in GB	This message indicates that the indicated page is being latched for exclusive access in the global buffer pool.
locking page *number : number*	This message indicates that the specified page is being locked.

opening storage area *number* file	This message indicates that the specified storage area file is being opened.
performing cluster statistics collection	This message indicates that the RMU Show Statistic utility is performing statistic collection cluster-wide.
performing local statistics collection	This message indicates the RMU Show Statistic utility is performing statistic collection on the local node only.
prepared, waiting to commit distributed transaction	This message indicates that the process is part of a distributed transaction and that the transaction has been successfully prepared.
Processing pending messages	This message indicates that the standby database log replication server (“LRS”) on the standby database is processing messages received from the master database.
Processing pending redo	This message indicates that the standby database log replication server (“LRS”) on the standby database is applying the after-image journal information from the master database.
querying standby database for state information	This message indicates that the Log Catch-up Server (“LCS”) on the master database is querying the Log Replication Server (“LRS”) on the standby database for information about the state of the Hot Standby replication.
RCS waiting for *number* record cache record latch *number*	This message indicates that the row cache server (“RCS”) is waiting for the latch for the specified row cache.

reading *number* AIJ file blocks from VBN *number*	This message displays whenever the after-image journal lock information needs to be refreshed; this typically only occurs the first time a user attaches to the database. The after-image journal is read to determine the after-image journal logical EOF (not to be confused with the OpenVMS logical EOF). It is also read by the database recovery ("DBR") process.
reading AIJ open record (block 1)	This message indicates the after-image journal "open" record is being read to verify information about the state of the after-image journal.
reading pages *number : number* to *number :* *number*	This message displays whenever one or more pages are read into either a user's local buffer or the global buffer. One buffer full of pages is being read. The format string "*number : number*" identifies the physical area and the page number.
reading ROOT file (*name* VBN *number*)	This message displays whenever the in-memory database root information is determined to be out-of-date and must be read again from the disk. This message normally occurs only when a database parameter is modified by a user on line or some information in the database root is modified by the system (such as the after-image journal sequence number).
reading .RUJ file block *number*	This message displays whenever an undo operation needs to read the next RUJ page to acquire the rollback information necessary to complete the operation. The .RUJ file is read one block at a time. Sometimes a process that is not being rolled back receives this message because it was necessary to

	read the .RUJ file in order to refresh cached recovery information.
sending AIJ replication data to standby database	This message indicates that a Hot Standby replication message is being sent from the master database to the standby database.
sending AIJ replication reply to master database	This message indicates that the Log Replication Server ("LRS") on the standby database is sending a reply message to the master database.
switching AIJ journals	This message indicates that an after-image journal switch-over operation is being performed; this message only occurs when circular after-image journals are used.
waiting *number* seconds for next pass through AIJ files	This message indicates that an after-image journal backup operation is suspended for the indicated number of seconds before performing an additional pass through the after-image journals.
waiting for *resource* (*mode*)	This message displays whenever a process requests a lock "with wait" and another process is holding the lock in an incompatible mode. This message may indicate a database hot spot and should be investigated using the RMU Show Locks utility. The format string "*resource*" identifies the lock type (that is, storage area, page, MEMBIT, etc.) and the string "*mode*" identifies the requested lock mode (PR, CR, EX, etc.).
waiting for *number* record cache hash	This message indicates that the process is waiting for the specified row cache hash table latch.

latch	
waiting for *number* record cache record latch	This message indicates that the process is waiting for the specified row cache record image latch.
waiting for *number*-block unmodified AIJ (*number* minutes)	This message indicates that an after-image journal switch-over operation is suspended, and that an unmodified after-image journal containing at least the indicated number of available blocks is required for the switch-over operation to continue. The "minutes" indicates the time remaining before the database is shutdown.
waiting for active AIJ backups to complete	This message indicates that the process is waiting for active afterimage journal backup operations to complete. This message is typically displayed when using the RMU Close utility to close the database.
waiting for AIJ initialization	This message indicates that the process is waiting for an after-image journal initialization operation to complete.
waiting for AIJ message *number* from master database	This message indicates that the process is waiting for a reply from the Log Replication Server ("LRS") on the standby database.
waiting for async-prefetch of pages *number : number* to *number : number*	This message indicates that the process is waiting for the asynchronous pre-fetch (read) of the specified pages to complete. This message may indicate that the "Asynchronous Pre-Fetch" feature attributes may not be appropriate.
waiting for async-write of pages	This message indicates that the process is waiting for the asynchronous write of the specified pages

number : *number* to *number : number*	to complete. This message may indicate that the “Asynchronous Batch Write” feature attributes may not be appropriate.
waiting for async-write of ROOT file	This message indicates that the process is waiting for the asynchronous write to the database rootfile to complete.
waiting for busy AIJ sequence *number* (*number* minutes)	This message indicates that the specified after-image journal has an active checkpoint held by an active process; this message almost always results during an after-image journal backup operation of circular after-image journals.
waiting for checkpoint completion	This message indicates that a checkpoint operation is being performed.
waiting for global buffer latch	This message indicates that the process is waiting for the global buffer latch. Note that this latch differs from the global buffer page latch.
waiting for reply from standby database	This message indicates that the process is waiting for a reply from the Log Replication Server (“LRS”) on the standby database.
waiting for standby database activity request	This message indicates that the Log Replication Server (“LRS”) on the standby database is idle and waiting for messages from the master database.
writing *number* AIJ file blocks from VBN *number*	This message displays whenever a group commit process writes the commit information to the after-image journal. In a high throughput environment, the write buffer length will be as close to 64K as possible.

writing *number* pages back to database	This message displays whenever one or more data pages are written to the database. This is typically caused by a request to access those pages from another process or by detaching from the database.
writing AIJ sequence *number* block *number*	This message displays whenever a group commit process writes the commit information to the after-image journal. In a high throughput environment, the write buffer length will be as close to 64K as possible.
writing ROOT file (*name* VBN *number*)	This message displays whenever the in-memory database root information is modified by a user on line or some information in the database root is modified by the system (such as the after-image journal sequence number).
writing .RUJ file block *number*	This message displays whenever a user process writes data page modification information to the .RUJ file. This message always precedes the next message

Contents