SQL Update

Ian Smith

Relational Language Group
Oracle Rdb

Summary

- Documentation Update
- RMU Extract
- Query Governor
- New SQL native functions
- DDL Changes
- Some other items of interest

Documentation

- Regenerated SQL HELP for 7.1.3
- SQL Reference Manual has been updated for V7.1.4.1, and will be available soon

RMU Extract

- Major rewrite of VIEW extract for V7.1.3
  - Support for nested derived tables
  - Various join combinations
  - Better UNION, EXCEPT and INTERSECT support
- Decided to move V7.1 RMU Extract into the V7.0.8 release (subset functionality)
- Look for updated documentation in 7.0.8 release notes
- Includes support new options MATCH, GROUP_TABLE, etc

RMU Extract

- New /DEFAULTS qualifier to establish dialect, language, allocation amount, etc
- Currently script header sets QUOTING RULES to ‘SQL92’ and DEFAULT DATE FORMAT ‘SQL92’
- Use qualifier to modify the generated script /DEFAULT=(DATE_FORMAT=“VMS”, LANGUAGE=SWEDISH,NOALLOCATION)

Query Governor

Query Timeout [review]

- V4.1 introduced query governor controls
  - RDMS$BIND_QG_TIMEOUT
  - RDMS$BIND_QG_REC_LIMIT
- V4.2 added extra controls
  - RDMS$BIND_QG_CPU_TIMEOUT
- Timeouts were query compile time
- Row limit for delivered rows

Query Timeout [New]

- Now in V7.1.2.4 new additions
  - RDMS$BIND_QG_EXEC_CPU_TIMEOUT
  - RDMS$BIND_QG_EXEC_ELAPSED_TIMEOUT
- Controls query execution
  note: ‘exec’ in the name

QG Interface

- Controls are provided through several interfaces
  - Logical Names
  - Interactive SQL SET statement
  - Dynamic SQL SET statement
  - Module Language and Pre-compiler qualifiers
  - RMU Show Statistics Dashboard (exec controls)
- New dynamic support useful for OCI Services for Rdb SQL.INIT files

Example 1

```
SQL> set query execution
cont> limit elapsed time 5 minutes;
.
.
.
SQL> delete from EMPLOYEES;
%RDB-E-EXQUOTA, Oracle Rdb runtime quota exceeded
-RDMS-E-MAXTIMLIM, query governor maximum
timeout has been reached
SQL>
```

Units. Defaults to second

Example 2

```
SQL> set query execution limit elapsed time 2 seconds;
SQL> delete from EMPLOYEES;
%RDB-E-EXQUOTA, Oracle Rdb runtime quota exceeded
-RDMS-E-MAXTIMLIM, query governor maximum
timeout has been reached
SQL> show query limit;
Query limit Time is OFF
Query limit Row count is OFF
Query limit CPU time is OFF
Execution limit CPU time is OFF
Execution limit Elapsed time    0 00:00:02.00 (2 seconds)
Execution limit Row count is OFF
SQL> set query execution nolimit elapsed time;
```

QG Notes

- ‘exec’ options are process wide
- Affect all database attaches in the session
- Lazy timer used to reduce overhead, so may not see exact timeouts
- Only checked during database I/O so external functions, or CPU intensive SQL procedures will not timeout
- Use RMU Show Statistics Dashboard to adjust timeout dynamically

SQL Functions

ROUND and TRUNC

- ROUND and TRUNC of numeric values now native Rdb functions
- In V7.0 and up until V7.1.3 existed as DOUBLE PRECISION functions in SQL$FUNCTIONS library
- These remain for backward compatibility but did not work consistently

ROUND and TRUNC

- Now support TINYINT, SMALLINT, INTEGER, BIGINT, REAL, and DOUBLE PRECISION values
- DATE variant now dispatched to OCI Services for Rdb via Rdb Server. Expanded support to include DATE ANSI, TIMESTAMP as well as DATE VMS

RDB$$IS_ROW_FRAGMENTED

- Optional function to detect fragmented rows
- Must activate in Interactive SQL using DECLARE FUNCTION
- Must use exact syntax as show in release notes
- Accepts a DBKEY and returns 0 (not fragmented) or 1 (fragmented)

Example 3

```
SQL> declare function RDB$$IS_ROW_FRAGMENTED
cont>    (in :dbk char(8) character set unspecified)
cont>    returns integer;
SQL>
SQL> select dbkey,
cont>   RDB$$IS_ROW_FRAGMENTED (dbkey)
cont> from work_status;
                    DBKEY
                    99:10:12          0
                    99:10:13          0
                    99:10:14          0
3 rows selected
```

DDL Changes

Computed Columns

- ALTER TABLE … ALTER COLUMN now allows change of COMPUTED BY or AUTOMATIC AS computed expressions
- Previously had to DROP/CREATE the column
- Problem if it was used by a procedure, trigger or constraint

Computed columns

- Can not change the ‘class’ of column
  - Computed by
  - Automatic as
  - Automatic Insert as
  - Automatic Update as
- New row version is created, old AUTOMATIC data is converted to new type

DECLARE and UNDECLARE

- Interactive SQL allows you to declare a local variable, and assign it a default value.
- To eliminate the variable, you use the UNDECLARE statement
- These statements also supported in Dynamic SQL
- Note: closely matches SQL$PRE usage of host variables

Declare Example

```
SQL> declare :employee_id ID_NUMBER = '00164';
SQL> select last_name from employees
cont> where :employee_id = employee_id;
  LAST_NAME
  Toliver
SQL> undeclare :employee_id;
```

- Note: When using DECLARE edit string is inherited from the domain.

Alter Sequence

- Support SQL:2003 syntax
- RESTART WITH clause
  - Resets the sequence to start with the specified value
  - Must be in the range MINVALUE to MAXVALUE
- Used by TRUNCATE TABLE for IDENTITY columns
  - When a table is truncated the identity column starts at 1

Alter View

- Allow modification of view attributes
  - COMMENT IS
    synonymous with COMMENT ON VIEW statement
  - RENAME TO
    synonymous with RENAME VIEW statement
  - WITH CHECK OPTION
    enforces constraint on the view WHERE condition
  - WITH NO CHECK OPTION

Alter View

- AS SELECT … FROM … WHERE
  - Allows redefinition of table select expression
  - Allows redefinition of column source expressions
  - Preserves dependencies on view
  - Can not change column names, order or number

Alter View

```
SQL> create view DEPARTMENTS_SUMMARY
cont> as
cont> select department_code, d.department_name,
cont>        d.manager_id, jh.employee_count
cont> from departments d inner join
cont>      (select department_code, count (*)
cont>         from job_history
cont>         where job_end is null
cont>         group by department_code)
cont>           as jh (department_code, employee_count)
cont>      using (department_code);
SQL>
```

Alter View (example)

- The view DEPARTMENTS_SUMMARY is used by stored procedures, other views, etc
- To make changes to the view requires DROP VIEW … CASCADE and rebuild of dependency tree using CREATE VIEW (at least)
- The DBA decides to tune the queries by making EMPLOYEE_COUNT a static column and avoid continual computation by reporting applications.

Alter View

```
SQL> alter view DEPARTMENTS_SUMMARY
cont> as
cont> select department_code, d.department_name,
cont>        d.manager_id, d.employee_count
cont> from departments d;
SQL>
```

- View name, column names, column ordering all stay the same - no dependencies break and no rebuild of hierarchy is required

Alter View

- A view may change from allowing updates to being read-only
- Columns might change to be read-only
- Care must be taken or exceptions will be reported at run-time

Storage Maps

- Create and Alter Storage Map now create a special ‘system’ routine in module Rdb$Storage_Maps
- Can be used by applications to determine target partition for specified columns (see USING clause)
- Use ALTER STORAGE MAP … COMPILE to add to existing storage maps

Example 4

```
SQL> show system module
Modules in database with filename mf_personnel
    RDB$STORAGE_MAPS
SQL> show system function
Functions in database with filename mf_personnel
    CANDIDATES_MAP
    COLLEGES_MAP
    DEGREES_MAP
    DEPARTMENTS_MAP
    EMPLOYEES_MAP
…
```

Example 5

```
SQL> show function EMPLOYEES_MAP
Information for function EMPLOYEES_MAP

 Function ID is: -2
 Source:
return
    case
        when (:EMPLOYEE_ID <= '00200') then 1
        when (:EMPLOYEE_ID <= '00400') then 2
        else 3
    end case;
 Comment:       Return value for select partition - range 1 .. 3
…
```

Example 5 (cont)

```
Module name is: RDB$STORAGE_MAPS
 Module ID is: -1
 Number of parameters is: 1
Parameter Name    Data Type       Domain or Type
--------------    ---------       --------------
                  INTEGER
        Function result datatype
        Return value is passed by value

EMPLOYEE_ID                 CHAR(5)
        Parameter position is 1
        Parameter is IN (read)
        Parameter is passed by reference
```

Pre-compiler and Module Language

ORACLE

/PROTOTYPES

- In prior version supported /C_PROTOTYPES for C language only
- Now expanded to support Pascal, Bliss and soon Basic
- Generates external declarations of the module language routines
- C support enhanced for ANSI C and C++ compatibility

Prototypes

- Simplifies interface for module language routines
- Used by Rdb Engineering for RMU tools that use SQL Module Language
- Generate an extra output file for the LANGUAGE clause of the module

Example

- Use LANGUAGE C
- Compile using SQL$MOD/PROTOTYPE
  - Default filename is derived from source .SQLMOD
  - Default file type is on language
- In the C source include the generated file

Source Module Language

```
module SAMPLE
language C
parameter COLONS

declare alias filename SQL$DATABASE

procedure START_TRANSACTION (sqlcode);
    start transaction;

procedure COMMIT_TRANSACTION (sqlstate);
    commit work;
```

Generate .h header file

```
/* Generated by Oracle Rdb SQL X7.1-00 at 10-FEB-2005 13:47:52.70 */
/* Source file is USER2:[TESTER.RDB$TEST_SYSTEM]XX.SQLMOD;1 */
#ifndef _SAMPLE_H_
#define _SAMPLE_H_
#ifdef __cplusplus
extern "C" {
#endif /* __cplusplus */

extern void START_TRANSACTION (
            long *SQLCODE                       /* out    */
        );
extern void COMMIT_TRANSACTION (
            void *SQLSTATE                      /* out    */
        );
#ifdef __cplusplus
}
#endif /* __cplusplus */
#endif /* _SAMPLE_H_ */
```

Miscellaneous

- New GET DIAGNOSTICS keywords
  - LIMIT_CPU_TIME
  - LIMIT_ROWS_FETCHED
  - LIMIT_ELAPSED_TIME
  - TRACE_ENABLED (7.1.4)
- New Chinese character set: GB18030 (defined by the GB18030-2000 standard) as used by the People's Republic of China (7.1.4.1)
- Dynamic EXECUTE statement now supports INTO variable list
  - Previously this had to be an SQLDA

TRACE_ENABLED

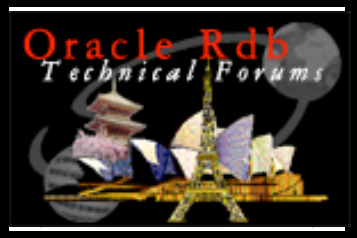
```
SQL> declare :x integer;
SQL> begin get diagnostics :x = TRACE_ENABLED; end;
SQL> print :x;
            X
            0
SQL> set flags 'trace';
SQL> begin get diagnostics :x = TRACE_ENABLED; end;
SQL> print :x;
            X
            1
```

Authentication

- Starting with V7.1.2.4 we have enhanced support for the DCL SET UIC command
- Command requires CMKRNL just like SUBMIT/USERNAME
- We use the UIC to find the first matching user
- Better integration with SQL/Services and OCI Services
- Simplifies specialized impersonation

SET UIC Example

```
$ set uic jain
$ sql$
SQL> attach 'file sql$database';
SQL> show privileges on database rdb$dbhandle;
Privileges on Alias RDB$DBHANDLE
    (IDENTIFIER=[RDB,JAIN],ACCESS=SELECT+INSERT+
       UPDATE+DELETE+SHOW+CREATE+ALTER+DROP+
       DBCTRL+OPERATOR+DBADM+SECURITY+
       DISTRIBTRAN)
```



For More Information

- www.oracle.com/rdb
- metalink.oracle.com
- www.hp.com/products/openvms
- ian.e.smith@oracle.com
- Sample script – SQL and log file

QUESTIONS & ANSWERS

ORACLE


ORACLE®