Oracle® Rdb7 for OpenVMS

Release Notes

Release 7.0.3.1

October 1999

Oracle Rdb7 Release Notes

Release 7.0.3.1

This document was prepared using VAX DOCUMENT, Version 2.0.

Contents

Preface vii

1 Installing Oracle Rdb7 Release 7.0.3.1

1.1 Requirements 1–1
1.2 Invoking VMSINSTAL 1–1
1.3 Stopping the Installation 1–2
1.4 After Installing Oracle Rdb7 1–2
1.5 New Documentation HTML Save Sets Available 1–2
1.6 Alpha EV6 Processor Support Added 1–3
1.7 Maximum OpenVMS Version Check Added 1–3

2 Software Errors Fixed in Oracle Rdb7 Release 7.0.3.1

2.1 Software Errors Fixed That Apply to All Interfaces 2–1
2.1.1 Leaf Cardinality Problem in Sorted Ranked Indexes 2–1
2.1.2 Corruption of Ranked Index Node During Duplicate Deletion 2–1
2.1.3 Deadlock on AIP Larea Sync 2–2
2.1.4 Online ALTER STORAGE AREA Operations and Row Cache 2–2
2.1.5 Inconsistent Enforcement of RESERVE Limits 2–2
2.1.6 ABS Initialization of AIJ Journal Causes System Slowdown 2–3
2.1.7 Transaction Checkpoint Determination of Switchover or Backup 2–4
2.1.8 Non-Update Read/Write Transactions Do Not Validate Checkpoint Thresholds 2–4
2.1.9 Lowercase Characters Cannot Be Used as Escape Character in a LIKE Clause 2–4
2.1.10 ACCVIO if RDM$BIND_STT_NETWORK_TRANSPORT Logical Defined 2–5
2.1.11 RMU/SHOW Statistic Cluster Collection Failure 2–5
2.1.12 Error Creating RUJ When Default RUJ Directory Does Not Exist . . . 2–5
2.1.13 Process May Stall in an Infinite Wait for a GBPT Slot Latch 2–6
2.1.14 RDMS-F-NOREQIDT Errors 2–6
2.1.15 AIJ Work File Search List Capability 2–7
2.1.16 Failed Backup of Extensible AIJ Causes Gap in Sequence Numbers 2–7
2.1.17 Improper Column Reference in CREATE TRIGGER Now Detected . . . 2–7
2.1.18 Bugcheck in DIO$FREE_CURRENT_LOCK 2–8
2.2 SQL Errors Fixed 2–8
2.2.1 Improved Optimization for NULLIF Expression with Subselect 2–8
2.2.2 Correct JOIN Syntax Generated Error and Bugcheck 2–10
2.2.3 Incorrect Results from Oracle7 Outer Join Syntax 2–10
2.2.3.1 Outer Joins 2–12
2.2.3.2 Outer Join Examples 2–13

2.2.4 MAPPING VALUES Not Supported for CREATE INDEX in IMPORT 2–16
2.2.5 SELECT Returns Incorrect Values for GROUP BY Queries 2–16
2.2.6 Restrictions on CURRENT_TIMESTAMP Default Value Lifted 2–17
2.2.7 SQL Uses Read/Write Transactions Against Standby Database 2–18
2.2.8 Domain Constraints Did Not Support Function Calls Passing VALUE 2–19
2.2.9 CREATE VIEW May Fail on Large or Complex View Definition 2–19
2.2.10 Incomplete Support for Multischema Databases for Query Outlines 2–20
2.2.11 UNKNOWN_VAR Error Reported During Select from a View 2–21
2.2.12 SQL Module Language Compiler Fails Unexpectedly 2–21
2.2.13 ORDER BY Select Query from a View with ORDER BY on the Same COMPUTED BY Column Returns Wrong Result 2–22
2.3 Oracle RMU Errors Fixed 2–23
2.3.1 Lock ID Hidden by Stall Message in RMU/SHOW Statistics 2–23
2.3.2 Terminal Width Checked by RMU/SHOW Statistics 2–23
2.3.3 RMU/SHOW Statistic Physical-Area Event Creation Bugchecks 2–24
2.3.4 Ignore Row Caches when Writing RMU/SHOW Statistic Report 2–24
2.3.5 RMU/LOAD from a Record-Oriented Device Caused RMS-F-IOP Error 2–24
2.3.6 RMU/EXTRACT Sometimes Bugchecks when Processing Many Storage Areas 2–25
2.3.7 Problem in Reporting Recovered AIJ Sequence Number 2–25
2.3.8 Truncate Table Could Generate RMU/VERIFY Errors 2–25
2.4 Row Cache Errors Fixed 2–26
2.4.1 Row Cache of One Slot Causes Loop 2–26
2.5 Hot Standby Errors Fixed 2–26
2.5.1 Stopping Hot Standby on Standby then Master Hangs Standby for 15 Minutes 2–26
2.5.2 Monitor ENQLM Minimum Increased to 32767 2–26
2.6 Oracle Trace Errors Fixed 2–26
2.6.1 Incorrect Completion Status Reported by Oracle Trace 2–27

3 Documentation Corrections

3.1 Documentation Corrections 3–1
3.1.1 Partition Clause is Optional on CREATE STORAGE MAP 3–1
3.1.2 Oracle Rdb Logical Names 3–1
3.1.3 Waiting for Client Lock Message 3–1
3.1.4 Documentation Error in *Oracle Rdb7 Guide to Database Performance and Tuning* 3–3
3.1.5 SET FLAGS Option IGNORE_OUTLINE Not Available 3–3
3.1.6 SET FLAGS Option INTERNALS Not Described 3–3
3.1.7 Documentation for VALIDATE_ROUTINE Keyword for SET FLAGS 3–4
3.1.8 Documentation for Defining the RDBSERVER Logical Name 3–4
3.1.9 Undocumented SET Commands and Language Options 3–5
3.1.9.1 QUIET COMMIT Option 3–5
3.1.9.2 COMPOUND TRANSACTIONS Option 3–6
3.1.10 Undocumented Size Limit for Indexes with Keys Using Collating Sequences 3–7
3.1.11 Changes to RMU/REPLICATE AFTER/BUFFERS Command 3–8
3.1.12 Change in the Way RDMAIJ Server is Set Up in UCX 3–8

3.1.13 CREATE INDEX Supported for Hot Standby 3–9
3.1.14 Dynamic OR Optimization Formats 3–9

4 Known Problems and Restrictions

4.0.1 AIJSERVER Privileges 4–1
4.0.2 Lock Remastering and Hot Standby 4–1
4.0.3 RDB_SETUP Privilege Error 4–2
4.0.4 Dynamic Optimizer Problem with Zigzag Match 4–2
4.0.5 Starting Hot Standby on Restored Standby Database May Corrupt Database 4–3
4.0.6 Restriction on Compound Statement Nesting Levels 4–3
4.0.7 Back Up All AIJ Journals Before Performing a Hot Standby Switchover Operation 4–4
4.0.8 Concurrent DDL and Read-Only Transaction on the Same Table Not Compatible 4–4
4.0.9 Oracle Rdb and the SRM_CHECK Tool 4–5
4.0.10 Oracle RMU Checksum_Verification Qualifier 4–5
4.0.11 Do Not Use HYPERSORT with RMU/OPTIMIZE/AFTER_JOURNAL (Alpha) 4–6
4.0.12 Restriction on Using /NOONLINE with Hot Standby 4–6
4.0.13 SELECT Query May Bugcheck with PSII2SCANGETNEXTBBCDUPLICATE Error 4–7
4.0.14 DBAPack for Windows 3.1 is Deprecated 4–7
4.0.15 Determining Mode for SQL Non-Stored Procedures 4–7
4.0.16 DROP TABLE CASCADE Results in %RDB-E-NO_META_UPDATE Error 4–9
4.0.17 Bugcheck Dump Files with Exceptions at COSI_CHF_SIGNAL 4–10
4.0.18 Interruptions Possible when Using Multistatement or Stored Procedures 4–10
4.0.19 Row Cache Not Allowed on Standby Database While Hot Standby Replication Is Active 4–11
4.0.20 Hot Standby Replication Waits when Starting if Read-Only Transactions Running 4–12
4.0.21 Error when Using the SYS$LIBRARY:SQL_FUNCTIONS70.SQL Oracle Functions Script 4–12
4.0.22 DEC C and Use of the /STANDARD Switch 4–12
4.0.23 Excessive Process Page Faults and Other Performance Considerations During Oracle Rdb Sorts 4–13
4.0.24 Performance Monitor Column Mislabeled 4–14
4.0.25 Restriction Using Backup Files Created Later than Oracle Rdb7 Release 7.0.1 4–14
4.0.26 RMU Backup Operations and Tape Drive Types 4–15
4.0.27 Use of Oracle Rdb from Shared Images 4–15
4.0.28 Interactive SQL Command Line Editor Rejects Eight-Bit Characters 4–15
4.0.29 Restriction Added for CREATE STORAGE MAP on Table with Data 4–16
4.0.30 ALTER DOMAIN...DROP DEFAULT Reports DEFVALUNS Error . . . 4–16
4.0.31 Oracle Rdb7 Workload Collection Can Stop Hot Standby Replication 4–17
4.0.32 RMU Convert Command and System Tables 4–18
4.0.33 Converting Single-File Databases 4–18

4.0.34 Restriction when Adding Storage Areas with Users Attached to Database 4–18
4.0.35 Restriction on Tape Usage for Digital UNIX V3.2 4–19
4.0.36 Support for Single-File Databases to be Dropped in a Future Release 4–19
4.0.37 DECdtm Log Stalls 4–19
4.0.38 Cannot Run Distributed Transactions on Systems with DECnet/OSI and OpenVMS Alpha Version 6.1 or OpenVMS VAX Version 6.0 4–20
4.0.39 Multiblock Page Writes May Require Restore Operation 4–20
4.0.40 Oracle Rdb7 Network Link Failure Does Not Allow DISCONNECT to Clean Up Transactions 4–21
4.0.41 Replication Option Copy Processes Do Not Process Database Pages Ahead of an Application 4–21
4.0.42 SQL Does Not Display Storage Map Definition After Cascading Delete of Storage Area 4–22
4.0.43 ARITH_EXCEPT or Incorrect Results Using LIKE IGNORE CASE 4–22
4.0.44 Different Methods of Limiting Returned Rows from Queries 4–22
4.0.45 Suggestions for Optimal Usage of the SHARED DATA DEFINITION Clause for Parallel Index Creation 4–23
4.0.46 Side Effect when Calling Stored Routines 4–25
4.0.47 Nested Correlated Subquery Outer References Incorrect 4–26
4.0.48 Considerations when Using Holdable Cursors 4–28
4.0.49 INCLUDE SQLDA2 Statement Is Not Supported for SQL Precompiler for PL/I in Oracle Rdb Release 5.0 or Higher 4–29
4.0.50 SQL Pascal Precompiler Processes ARRAY OF RECORD Declarations Incorrectly 4–29
4.0.51 RMU Parallel Backup Command Not Supported for Use with SLS . . . 4–30
4.0.52 Oracle RMU Commands Pause During Tape Rewind 4–30
4.0.53 TA90 and TA92 Tape Drives Are Not Supported on Digital UNIX 4–30
4.1 Oracle CDD/Repository Restrictions 4–30
4.1.1 Oracle CDD/Repository Compatibility with Oracle Rdb Features 4–30
4.1.2 Multischema Databases and CDD/Repository 4–32
4.1.3 Interaction of Oracle CDD/Repository Release 5.1 and Oracle RMU Privileges Access Control Lists 4–32
4.1.3.1 Installing the Corrected CDDSHR Images 4–34
4.1.3.2 CDD Conversion Procedure 4–34

5 Enhancements

5.1 Enhancements Provided in Oracle Rdb7 Release 7.0.3.1 5–1
5.1.1 Per-Process Monitoring for SHOW STATS 5–1
5.1.2 New DOMAINS Option for RMU/EXTRACT 5–1
5.1.3 New NO REORGANIZE clause for ALTER STORAGE MAP 5–2
5.1.4 New Options for the GET DIAGNOSTICS Statement 5–4
5.1.5 RMU/SHOW Statistic OPCOM Message Tracking 5–5
5.1.6 New Restricted_Access Qualifier for RMU/LOAD 5–8
5.1.7 RDO EDT Editor on OpenVMS Alpha Now Available 5–8
5.1.8 New Options Added to SQL EXTRACT Function 5–8

Tables

3–1 Object Type Values 3–2
4–1 Oracle CDD/Repository Compatibility for Oracle Rdb Features 4–31

Preface

Purpose of This Manual

This manual contains release notes for Oracle Rdb7 Release 7.0.3.1. The notes describe changed and enhanced features; upgrade and compatibility information; new and existing software problems and restrictions; and software and documentation corrections. These release notes cover both Oracle Rdb7 for OpenVMS Alpha and Oracle Rdb7 for OpenVMS VAX, which are referred to by their abbreviated name, Oracle Rdb7.

Intended Audience

This manual is intended for use by all Oracle Rdb7 users. Read this manual before you install, upgrade, or use Oracle Rdb7 Release 7.0.3.1.

Document Structure

This manual consists of five chapters:

Chapter 1	Describes how to install Oracle Rdb7 Release 7.0.3.1.
Chapter 2	Describes software errors corrected in Oracle Rdb7 Release 7.0.3.1.
Chapter 3	Provides information not currently available in the Oracle Rdb7 documentation set.
Chapter 4	Describes problems, restrictions, and workarounds known to exist in Oracle Rdb7 Release 7.0.3.1.
Chapter 5	Describes enhancements introduced in Oracle Rdb7 Release 7.0.3.1.

1

Installing Oracle Rdb7 Release 7.0.3.1

This software update is installed using the standard OpenVMS Install Utility.

1.1 Requirements

The following conditions must be met in order to install this software update:

- Oracle Rdb7 must be shutdown before you install this update kit. That is, the command file SYS$STARTUP:RMONSTOP(70).COM should be executed before proceeding with this installation. If you have an OpenVMS cluster, you must shutdown all versions of Oracle Rdb7 on all nodes in the cluster before proceeding.
- Oracle Rdb7 Release 7.0.3 must have already been installed on your system.
- The installation requires approximately 100,000 free blocks on your system disk for OpenVMS VAX systems; 200,000 blocks for OpenVMS Alpha systems.

1.2 Invoking VMSINSTAL

To start the installation procedure, invoke the VMSINSTAL command procedure:

```
@SYS$UPDATE:VMSINSTAL variant-name device-name OPTIONS N
```

variant-name

The variant names for the software update for Oracle Rdb7 Release 7.0.3.1 are:

- RDBSE1C070 for Oracle Rdb7 for OpenVMS VAX standard version.
- RDBASE1C070 for Oracle Rdb7 for OpenVMS Alpha standard version.
- RDBMVE1C070 for Oracle Rdb7 for OpenVMS VAX multiversion.
- RDBAMVE1C070 for Oracle Rdb7 for OpenVMS Alpha multiversion.

device-name

Use the name of the device on which the media is mounted.

- If the device is a disk drive, such as a CD-ROM reader, you also need to specify a directory. For CD-ROM distribution, the directory name is the same as the variant name. For example:

  ```
  DKA400:[RDBSE1C070.KIT]
  ```

- If the device is a magnetic tape drive, you need to specify only the device name. For example:

  ```
  MTA0:
  ```

OPTIONS N

This parameter prints the release notes.

The following example shows how to start the installation of the VAX standard kit on device MTA0: and print the release notes:

```
$ @SYS$UPDATE:VMSINSTAL RDBSE1C070 MTA0: OPTIONS N
```

1.3 Stopping the Installation

To stop the installation procedure at any time, press Ctrl/Y. When you press Ctrl/Y, the installation procedure deletes all files it has created up to that point and exits. You can then start the installation again.

If VMSINSTAL detects any problems during the installation, it notifies you and a prompt asks if you want to continue. You might want to continue the installation to see if any additional problems occur. However, the copy of Oracle Rdb7 installed will probably not be usable.

1.4 After Installing Oracle Rdb7

This update provides a new Oracle Rdb7 Oracle TRACE facility definition. Any Oracle TRACE selections that reference Oracle Rdb7 will need to be redefined to reflect the new facility version number for the updated Oracle Rdb7 facility definition, "RDBVMSV7.0-31".

If you have Oracle TRACE installed on your system and you would like to collect for Oracle Rdb7, you must insert the new Oracle Rdb7 facility definition included with this update kit.

The installation procedure inserts the Oracle Rdb7 facility definition into a library file called EPC$FACILITY.TLB. To be able to collect Oracle Rdb7 event-data using Oracle TRACE, you must move this facility definition into the Oracle TRACE administration database. Perform the following steps:

1. Extract the definition from the facility library to a file (in this case, RDBVMS.EPC$DEF).

   ```
   $ LIBRARY /TEXT /EXTRACT=RDBVMSV7.0-31 -
   _$ /OUT=RDBVMS.EPC$DEF SYS$SHARE:EPC$FACILITY.TLB
   ```

2. Insert the facility definition into the Oracle TRACE administration database.

   ```
   $ COLLECT INSERT DEFINITION RDBVMS.EPC$DEF /REPLACE
   ```

Note that if you are installing the multiversion variant of Oracle Rdb7, the process executing the INSERT DEFINITION command must use the version of Oracle Rdb7 that matches the version used to create the Oracle TRACE administration database or the INSERT DEFINITION command will fail.

1.5 New Documentation HTML Save Sets Available

Included with this release is a new backup save set (RDB_702_HTML.BCK) and a new self-extracting archive file (RDB_702_HTML.EXE) for Windows NT and Windows 95. These new files contain the Oracle Rdb V7.0 (and related products) documentation in HTML format. Documentation is included for the following products:

- Oracle Rdb, Release 7.0
- Rdb Web Agent, Release 2.2
- SQL*Net for Rdb7, Release 7.1.2
- Hot Standby for Oracle Rdb and CODASYL DBMS, Release 7.0

- Distributed Option for Rdb, Release 7.0

When you expand the RDB_702_HTML.BCK backup save set, the WWW and DOC sub-directories are created and product-specific sub-directories are created below DOC. Be sure to maintain the directory structure by specifying the following command:

```
$ BACKUP RDB_702_HTML.BCK/SAVE disk:[directory...]
```

To access this library of documentation, point to LIBRARY.HTML using your favorite web browser.

When you expand the RDB_702_HTML.EXE self-extracting archive file for your Windows NT or Windows 95 system, the rdbhtmldocs directory is created with product-specific directories below that. Again, access this library of documentation by pointing to LIBRARY.HTML from your favorite web browser.

The PostScript format of this documentation is available in the RDB7PS.BCK backup save set.

1.6 Alpha EV6 Processor Support Added

As of Oracle Rdb7 Release 7.0.3, the Alpha EV6 processor is supported. This is the only thing that changed between Oracle Rdb7 Release 7.0.2.1 and Oracle Rdb7 Release 7.0.3.

1.7 Maximum OpenVMS Version Check Added

As of Oracle Rdb7 Release 7.0.1.5, a maximum OpenVMS version check has been added to the product. Oracle Rdb has always had a minimum OpenVMS version requirement. With 7.0.1.5 and for all future Oracle Rdb releases, we have expanded this concept to include a maximum VMS version check and a maximum supported processor hardware check. The reason for this check is to improve product quality.

OpenVMS Version 7.2-n is the maximum supported version of OpenVMS.

As of Oracle Rdb7 Release 7.0.3, the Alpha EV6 processor is supported.

The check for the OpenVMS operating system version and supported hardware platforms is performed both at installation time and at runtime. If either a non-certified version of OpenVMS or hardware platform is detected during installation, the installation will abort. If a non-certified version of OpenVMS or hardware platform is detected at runtime, Oracle Rdb will not start.

2

Software Errors Fixed in Oracle Rdb7 Release 7.0.3.1

This chapter describes software errors that are fixed by Oracle Rdb7 Release 7.0.3.1.

2.1 Software Errors Fixed That Apply to All Interfaces

2.1.1 Leaf Cardinality Problem in Sorted Ranked Indexes

A problem in storage of the cardinality fields of sorted ranked index entries resulted in errors in RMU index verification.

This problem only occurred in duplicate entries where the number of duplicate dbkeys stored for a single entry is greater than 65535. Update of the entry cardinality caused problems with correct interpretation of the associated leaf cardinality.

The following is an example of the verification error that may be encountered due to this problem:

```
%RMU-W-BTRLEACAR, Inconsistent leaf cardinality (C2) of 6 specified
                  for entry 3 at dbkey 62:1087:0 using precision of 33.
                 Dbkey 62:3057:0 at level 2 specified a cardinality of 2
%RMU-I-BTRERPATH, parent B-tree node of 62:1087:0 is at 62:696:0
%RMU-I-BTRROODBK, root dbkey of B-tree is 62:696:0
```

A workaround for the problem is to use a normal sorted index instead of a sorted ranked index.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

2.1.2 Corruption of Ranked Index Node During Duplicate Deletion

On very rare occasions the deletion of a duplicate record from a sorted ranked index may cause corruption of the index node from which the duplicate dbkey was removed.

This problem occurred in duplicate entries where the number of duplicate dbkeys stored in the bitmap was already greater than could be held in a single index node and thus the entry had overflow nodes attached to it.

Most deletions of duplicate dbkeys from the ranked index bitmap result in the reduction of the size of the bitmap, however, on rare occasions is it possible for the bitmap to increase in size. If this occurs and there is insufficient room in the current index node for the new larger bitmap, the node will be split.

A problem with the splitting of the duplicate node during duplicate dbkey deletion resulted in the corruption of the duplicate entry.

The corruption of the index showed up during RMU index verification and caused a bugcheck dump when attempting to access duplicate dbkeys held within the corrupt bitmap.

The following is an example of the routine stack as may be seen in a bugcheck caused by this problem:

```
%SYSTEM-F-ACCVIO, access violation, reason mask=00,
virtual address=000000003A800210, PC=FFFFFFFF8090A13C, PS=00000009
Saved PC = 00E7E8C8 : PSII2SCANINVALIDATESCANSREM + 000000A8
Saved PC = 00E62B68 : PSII2REMOVEDUPBBC + 000000C8
Saved PC = 00E5FBC0 : PSII2REMOVEBOTTOM + 00000490
Saved PC = 00E5C908 : PSII2REMOVET + 00000228
Saved PC = 00E5CB54 : PSII2REMOVET + 00000474
Saved PC = 00E5D238 : PSII2REMOVETREE + 000001D0
Saved PC = 0102FC54 : RDMS$$KOD_REMOVE_TREE + 00001784
Saved PC = 00FF0F10 : RDMS$$EXE_ACTION + 00000A40
```

A workaround for the problem is to use a normal sorted index instead of a sorted ranked index.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

2.1.3 Deadlock on AIP Larea Sync

When concurrently creating indexes, occasionally deadlocks would be returned with the message: -RDMS-F-DEADLOCK, deadlock on AIP larea synch. Customers encountering other deadlocks are encouraged to reoprt them to their support personnel.

The following example shows a complete set of the errors encountered when this problem occurs:

```
%RDB-E-DEADLOCK, request failed due to resource deadlock
-RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-DEADLOCK, deadlock on AIP larea synch
```

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

2.1.4 Online ALTER STORAGE AREA Operations and Row Cache

Previously, it was possible to cause database recovery failures when using the row cache feature and making online modifications to storage areas. For example, if storage areas were marked read-only after they had been modified, it was possible for cached changes to be unable to be written back to the database at a later time.

This problem has been corrected. Online ALTER STORAGE AREA operations now request that the RCS process flush all modified rows from all caches back to the database before the storage area modification is performed. This means that a ALTER STORAGE AREA operation may stall while the RCS flushes modified rows back to the database.

2.1.5 Inconsistent Enforcement of RESERVE Limits

Bug 972253

Oracle Rdb7 did not consistently enforce the number of reserved storage areas, journals, or caches that could be created in a database. For example, Oracle Rdb7 would allow a user to repeatedly add reserved storage areas to a database until the database had more storage areas than are allowed by Oracle Rdb7, as shown:

```
$ SQL
CREATE DATABASE FILENAME SLOT_LIMITS
    RESERVE 1000 STORAGE AREAS
CREATE STORAGE AREA AREA_1 FILENAME AREA_1;
DISCONNECT ALL;
ALTER DATABASE FILENAME SLOT_LIMITS
    RESERVE 200 STORAGE AREAS;
%RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
DISCONNECT ALL;
EXPORT DATABASE FILENAME SLOT_LIMITS INTO SLOT_LIMITS;
DROP DATABASE FILENAME SLOT_LIMITS;
IMPORT DATABASE FROM SLOT_LIMITS FILENAME SLOT_LIMITS;
Exported by Oracle Rdb V7.0-3 Import/Export utility
.
.
.
IMPORT DATABASE FROM SLOT_LIMITS FILENAME SLOT_LIMITS;
Exported by Oracle Rdb V7.0-3 Import/Export utility
.
.
.
IMPORTing STORAGE AREA: RDB$SYSTEM
IMPORTing STORAGE AREA: AREA_1
%SQL-F-ERRCRESCH, Error creating database filename SLOT_LIMITS
-RDB-E-BAD_DPB_CONTENT, invalid database parameters in the database parameter block (DPB)
-RDMS-F-BADPARAM, reserved storage area count (1200) is out of valid range (0...1024)
```

A workaround for the failed IMPORT is to respecify the RESERVE...STORAGE AREAS clause on the IMPORT statement to a lower value.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. Oracle Rdb7 will now disallow adding more reserved areas, journals, or caches than the total allowed for a database.

2.1.6 ABS Initialization of AIJ Journal Causes System Slowdown

The initialization of AIJ journals, which occurs after the journal has been backed up, is extremely I/O intensive and often impacts the ability of the database to concurrently generate new AIJ information. Currently, there are two logicals that control the impact of the AIJ initialization operation. However, the default values were originally defined when only extensible AIJ journals existed. These default values are not conducive to circular AIJ journal initialization.

The workaround is to define explicit values for the RDM$BIND_AIJ_INITIALIZATION_IO_COUNT and RDM$BIND_AIJ_INITIALIZATION_IO_SIZE logicals when using fixed-size circular AIJ journals.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

The default values for the RDM$BIND_AIJ_INITIALIZATION_IO_COUNT and RDM$BIND_AIJ_INITIALIZATION_IO_SIZE logicals are now based on whether extensible or circular AIJ journals are active. The new default values are the following:

Logical Name	Extensible AIJ Default	Circular AIJ Default
RDM$BIND_AIJ_INITIALIZATION_IO_COUNT	15	2
RDM$BIND_AIJ_INITIALIZATION_IO_SIZE	127	32

In addition, these two logicals have been added to the database "AIJ dashboard." This means that the AIJ journal initialization values can be non-persistently modified at runtime using the RMU Show Statistics database dashboard utility.

2.1.7 Transaction Checkpoint Determination of Switchover or Backup

It is sometimes possible for a committing transaction to checkpoint even though no checkpoint thresholds have been exceeded. This problem occurs more frequently when the current AIJ journal is located on an extremely fast I/O device or serviced by an extremely fast device controller such as an HSJ disk controller with write-back caching enabled.

The problem is caused by a race condition between the process committing the transaction and the process performing the AIJ group commit I/O to the AIJ journal.

There is no workaround to this problem. However, this problem does not cause database corruption or cause other database integrity problems.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.8 Non-Update Read/Write Transactions Do Not Validate Checkpoint Thresholds

Currently, a read/write transaction that does not modify any database objects does not validate its checkpoint thresholds. This strategy has the advantage of allowing the transaction to re-use the "Transaction Sequence Number" and reducing the amount of AIJ journal information being submitted. However, this strategy has the disadvantage of the process checkpoint not getting advanced, which can result in an increased recovery duration if the process were to terminate prematurely.

Users can see the transaction commit behavior, using the new logical name RDM$BIND_RW_TX_CKPT_ADVANCE. This logical name can be placed in the LNM$FILE_DEV table.

The value "0" indicates that read/write non-update transactions do not advance the checkpoint; this is the current behaviour and the default value. The value "1" indicates that read/write non-update transactions will advance the checkpoint if the checkpoint thresholds are exceeded.

2.1.9 Lowercase Characters Cannot Be Used as Escape Character in a LIKE Clause

Bug 905784

When lowercase characters are used as escape characters within an SQL statement, Rdb may not be able to correctly locate matching records.

This problem only occurs when a sorted index is used by the optimizer to preselect a range of records which will be filtered using the LIKE pattern.

The following example shows the problem:

```
SQL> create table ttable (tname char(10));
SQL> insert into ttable (tname) value ('my_table1');
SQL> insert into ttable (tname) value ('my_table2');
SQL> set flags 'strategy';
SQL> select tname from ttable
cont> where tname like 'myz_table%' escape 'z';
Conjunct        Get     Retrieval sequentially of relation TTABLE
 TNAME
 my_table1
 my_table2
2 rows selected
SQL> create index tind on ttable(tname);
SQL> select tname from ttable
cont> where tname like 'myz_table%' escape 'z';
Conjunct        Index only retrieval of relation TTABLE
  Index name  TIND [1:1]
0 rows selected
```

Two possible workarounds can be used to prevent this problem:

- Use only uppercase or case-independent characters as the escape character within the LIKE clause.
- Prevent the optimizer from using index retrieval by either deleting the index or by using a query outline to change the selected strategy.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.10 ACCVIO if RDM$BIND_STT_NETWORK_TRANSPORT Logical Defined

It is possible for the RMU/SHOW Statistic utility to access violate when the RDM$BIND_STT_NETWORK_TRANSPORT logical is defined to the value "DECNET". This problem occurs on OpenVMS Alpha systems only, and appears to occur for OpenVMS Version 6.2 and later.

The only workaround is to not define the RDM$BIND_STT_NETWORK_ TRANSPORT logical.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.11 RMU/SHOW Statistic Cluster Collection Failure

It is sometimes possible for the RMU/SHOW Statistic utility to not connect to the remote statistics collection server (RDMSTTnn.EXE). This problem occurs on OpenVMS Alpha systems only, and appears to occur for OpenVMS Version 7.1 and later.

There is no workaround to this problem.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.12 Error Creating RUJ When Default RUJ Directory Does Not Exist

Bug 860794

Errors are returned if a database is altered such that the default recovery journal (RUJ) location has a non-existent directory. For example:

```
SQL> create database filename foo;
SQL> alter database filename foo recover journal
cont> (location is 'sys$disk:[nosuchdir]');
SQL> disconnect all;
SQL> att 'fi foo';
SQL> create table foo (f1 int);
%RDB-F-SYS_REQUEST, error from system services request
-RDMS-F-FILACCERR, error creating run-unit journal file DEV:[NOSUCHDIR]FOO$0001‘
-RMS-E-DNF, directory not found
-SYSTEM-W-NOSUCHFILE, no such file
SQL>
SQL> alter database filename foo recover journal
cont> (location is 'sys$disk:[gooddir]');
%RDB-F-SYS_REQUEST, error from system services request
-RDMS-F-FILACCERR, error creating run-unit journal file DEV:[NOSUCHDIR]FOO$0001‘
-RMS-E-DNF, directory not found
-SYSTEM-W-NOSUCHFILE, no such file
SQL>
```

The workarounds are:

1. Create the directory so that it becomes valid.
2. EXPORT/IMPORT the database.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. If the default recovery journal location contains an invalid directory, it will now be ignored.

2.1.13 Process May Stall in an Infinite Wait for a GBPT Slot Latch

Bug 714899

Under some rare conditions, it is possible to get a deadlock between a global buffer page table (GBPT) latch and a global page lock. Such a deadlock is not detected and so you get a process stall waiting indefinitely f or a GBPT slot latch, blocking all other processes with its locks.

The code has been changed to bugcheck after 3 minutes if you do not get a latch instead of looping infinitely.

There is no workaround to this problem.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.14 RDMS-F-NOREQIDT Errors

Bugs 870987 and 890405

In situations with high Oracle Rdb7 lock contention and substantial RMS activity, it was possible to have an application fail with the following error:

```
%RDMS-F-NOREQIDT, reached internal maximum number of simultaneous timer requests
```

This error could be returned to the application program or sometimes a bugcheck dump would be written with this error as the exception.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. Oracle Rdb7 has been changed to no longer use as many timer requests when handling blocking AST requests for database locks.

2.1.15 AIJ Work File Search List Capability

In previous releases of Oracle Rdb, the RDB$BIND_AIJ_WORK_FILE logical could be used to designate a device and directory in which the AIJ work files would be created. However, when that device became full and no more work files could be created or extended, Hot Standby would be shutdown.

In this release, a search list can be specified by defining logicals RDB$BIND_AIJ_WORK_FILE1, RDB$BIND_AIJ_WORK_FILE2, ... RDB$BIND_AIJ_WORK_FILEn with each logical pointing to a different device/directory. The numbers must start with 1 and increase sequentially without any gaps. When an AIJ work file cannot be created due to a "device full" error, Rdb will look for the next device in the search list by translating the next sequential work file logical. If RDB$BIND_AIJ_WORK_FILE is defined, it will be used first.

This AIJ work file search list capability is available in Oracle Rdb7 Release 7.0.3.1.

2.1.16 Failed Backup of Extensible AIJ Causes Gap in Sequence Numbers

In previous releases of Oracle Rdb, a failed backup of an extensible AIJ caused a gap in the AIJ sequence numbers. This gap caused RMU/RECOVER and Hot Standby to fail.

In this release, a failed backup of an extensible AIJ no longer causes a gap in the sequence numbers.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.1.17 Improper Column Reference in CREATE TRIGGER Now Detected

Bug 592660

In prior versions of Oracle Rdb, an application error in a trigger definition was not detected by the SQL interface. The resulting trigger generated a confusing diagnostic. This error is shown in the following example:

```
SQL> create table T (ta integer);
SQL> create table S (sa integer, sb integer);
SQL>
SQL> -- bad column reference in INSERT
SQL> create trigger T_S
cont>     before update of TA on T
cont>     referencing new as C2
cont>     (insert into S (ta) values (ta))
cont>     for each row;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDB-E-INVALID_BLR, request BLR is incorrect at offset 13
-RDMS-E-INVASSIGNMENT, illegal target for assignment
```

The problem was that the SQL interface allowed a column of the triggering table (T in the example) to be listed in the column list of an INSERT statement, even when it did not belong on the table targeted by the INSERT statement. This problem only occurs for the INSERT statement if the REFERENCING clause is present in a trigger definition.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. SQL now detects the invalid column reference as shown in this example:

```
SQL> -- bad column reference in INSERT
SQL> create trigger T_S
cont>     before update of TA on T
cont>     referencing new as C2
cont>     (insert into S (ta) values (ta))
cont>     for each row;
%SQL-F-FLDNOTCRS, Column TA was not found in the tables in current scope
```

2.1.18 Bugcheck in DIO$FREE_CURRENT_LOCK

Bug 728613

A bugcheck occurred with an exception in the routine DIO$FREE_CURRENT_ LOCK if the transaction type was READ COMMITTED and a hold cursor was used containing a query whose strategy employed a backward scan and the rows from the cursor were fetched over multiple transactions. The problem occurred because Oracle Rdb mistakenly released a lock prematurely. Later, when Oracle Rdb was truly finished with the resource, a bugcheck occurred trying to release the lock that had previously been released.

The following query on the PERSONNEL database shows an example of the problem:

```
SQL> ATTACH 'FILE PERSONNEL';
SQL>
SQL> DECLARE TRANSACTION READ WRITE ISOLATION LEVEL READ COMMITTED;
SQL>
SQL> DECLARE T2 TABLE CURSOR WITH HOLD PRESERVE ALL FOR
cont> SELECT    *
cont> FROM      EMPLOYEES
cont> WHERE     EMPLOYEE_ID > '00300'
cont> AND       EMPLOYEE_ID < '00400'
cont> ORDER BY EMPLOYEE_ID DESC;
SQL>
SQL> OPEN T2;
SQL> FETCH T2;
 EMPLOYEE_ID   LAST_NAME        FIRST_NAME   MIDDLE_INITIAL
   ADDRESS_DATA_1              ADDRESS_DATA_2              CITY
      STATE   POSTAL_CODE   SEX    BIRTHDAY      STATUS_CODE
 00374         Andriola         Leslie       Q.
   111 Boston Post Rd.         NULL                        Salisbury
      NH      03268         M      19-Mar-1955   1

SQL>
SQL> COMMIT;
SQL>
SQL> FETCH T2;
%RDMS-I-BUGCHKDMP, generating bugcheck dump file DISK:[DIR]RDSBUGCHK.DMP;
```

The only workaround the problem is to restructure the transaction to use isolation level serializable; or use a standard, non-hold cursor; or avoid the backward scan; or fetch all the rows using one transaction.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.2 SQL Errors Fixed

2.2.1 Improved Optimization for NULLIF Expression with Subselect

Bug 903619

In prior versions of Oracle Rdb, queries using NULLIF might have a sub-optimzal optimizer strategy if it included a subquery, as shown in this example:

```
SQL> set flags 'strat';
SQL> select nullif ((select count(*)
cont>                 from job_history jh
cont>                 where jh.employee_id = e.employee_id),
cont>               0),
cont>       employee_id
cont> from employees e
cont> where employee_id < '00166';
Match
  Outer loop
    Cross block of 2 entries
      Cross block entry 1
        Index only retrieval of relation EMPLOYEES
          Index name  EMP_EMPLOYEE_ID [0:1]
      Cross block entry 2
        Aggregate       Index only retrieval of relation JOB_HISTORY
          Index name  JOB_HISTORY_HASH [1:1]
  Inner loop      (zig-zag)
    Aggregate       Conjunct        Index only retrieval of relation JOB_HISTORY
      Index name  JOB_HISTORY_SORT [0:0]
                EMPLOYEE_ID
            2   00164
            4   00165
2 rows selected
```

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. NULLIF now shares an optimization with COALESCE and the simple CASE expression where the subselect is evaluated just once. This allows the Oracle Rdb server to perform less table accesses as shown in the example run on the latest release:

```
SQL> set flags 'strategy';
SQL> select nullif ((select count(*)
cont>                 from job_history jh
cont>                 where jh.employee_id = e.employee_id),
cont>               0),
cont>       employee_id
cont> from employees e
cont> where employee_id < '00166';
Match
  Outer loop
    Index only retrieval of relation EMPLOYEES
      Index name  EMP_EMPLOYEE_ID [0:1]
  Inner loop      (zig-zag)
    Aggregate       Conjunct        Index only retrieval of relation JOB_HISTORY
      Index name  JOB_HISTORY_SORT [0:0]
                EMPLOYEE_ID
            2   00164
            4   00165
2 rows selected
```

Note that a side effect of this change is that the query outline ID generated for the query and the query structure will change and thus the query outline will not be used for the query. This will require regeneration of these query outlines.

2.2.2 Correct JOIN Syntax Generated Error and Bugcheck

Bug 941621

In a SELECT expression, a result table is derived from some combination of the table references identified in the FROM clause of the expression. In the following example, the table reference is a joined table in which one of the latter table references is a derived table. The syntax is correct, yet was generating an error. This problem has been fixed and the syntax is now accepted. This example shows the syntax error which had been generated:

```
select col1,table1.col2 from table1
     left join
    (
       (select col1 from table2 where col2 = 1)  t2
        inner join table3 using (col1)
    )
    using (col1);

    %SQL-W-LOOK_FOR_STT, Syntax error, looking for:
    %SQL-W-LOOK_FOR_CON,            JOIN, LEFT, FULL, UNION, CROSS, RIGHT,
    %SQL-W-LOOK_FOR_CON,            INNER, LIMIT, ORDER, NATURAL, ),
    %SQL-F-LOOK_FOR_FIN,    found T2 instead
```

In the next example, the correlation name clause has been omitted. In such a case, SQL was generating a bugcheck dump. For example:

```
    select col1,table1.col2 from table1
     left join
     (
       (select col1 from table2 where col2 = 1)
        inner join table3 using (ch1)
      ) using (ch1);

%SQL-I-BUGCHKDMP, generating bugcheck dump file DISK1:[TEST]SQLBUGCHK.DMP;
%SQL-F-BUGCHK, There has been a fatal error. Please contact your Oracle
support representative. SQL$SEMRSE - WALK_TABLE_REF - UNKNOWN TYPE
```

The following example shows the error message SQL now generates:

```
select ch1,table1.ch2 from table1
 left join
(
        (select ch1 from table2 where ch2 = 1)
         inner join table3 using (ch1)
)
using (ch1);
%SQL-F-CORNAMREQ, A correlation name is required for a derived table
```

These problems have been corrected.

2.2.3 Incorrect Results from Oracle7 Outer Join Syntax

Bugs 782901 and 693636

Oracle Rdb release 7.0.1.1 introduced new SQL syntax for performing outer joins between two or more tables. The syntax and semantics were designed to duplicate the syntax available in Oracle7 and Oracle8 SQL and therefore enhance the compatibility between these two products. The special operator (+) can be placed in the WHERE clause to instruct SQL to join tables using outer join semantics.

In prior releases, this syntax could produce the incorrect results listed below. However, the ANSI/ISO SQL standard syntax (SQL92) for LEFT, RIGHT, and FULL OUTER JOIN does not have these problems and can be used as a workaround for these limitations.

- Using outer join syntax to join a table to itself may only perform an inner join. For example:

```
select e.eid, e.ename, m.ename as "Manager"
 from employees e, employees m
 where e.manid = m.eid (+)
 order by m.eid, e.eid;
```

- If a table was joined to itself and three or more table sources were specified, the query did not work and the error FLDNOTCRS would be erroneously generated as shown in this example:

```
SQL> select o.order_id, b1.comp_name, b2.comp_name
cont>  from t_order o, company b1, company b2
cont>  where b1.comp_id (+) = o.customer
cont>     and   b1.comp_type(+) = o.cust_type
cont>     and   b2.comp_id(+)   = o.transport
cont>     and   b2.comp_type(+) = o.trans_type;
%SQL-F-FLDNOTCRS, Column B1.COMP_NAME was not found in the tables in current
scope
```

  In some cases on OpenVMS Alpha, an SQL bugcheck could be generated.

- Predicates which compared an outer join marked table column with a simple expression were not correctly used for the outer join.

  In this example, the condition `job (+) = 'Clerk'` was treated as though it had no outer join marker as in `job = 'Clerk'`. If the column is not qualified with (+), it becomes part of the WHERE clause as a filter to the final result table. However, the query had wanted to place this condition on the outer join to force NULL-filled rows to be returned. The difference in the result can be seen in the correct results from these queries:

```
SQL> select ename, job, dept.deptno, dname
cont>  from emp, dept
cont>  where emp.deptno (+) = dept.deptno
cont>     and job (+) = 'Clerk';
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 Miller       Clerk                 10   Accounting
 Smith        Clerk                 20   Research
 Adams        Clerk                 20   Research
 James        Clerk                 30   Sales
 NULL         NULL                  40   Operations
5 rows selected

SQL>
SQL> select ename, job, dept.deptno, dname
cont>  from emp, dept
cont>  where emp.deptno (+) = dept.deptno
cont>     and job = 'Clerk';
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 Miller       Clerk                 10   Accounting
 Smith        Clerk                 20   Research
 Adams        Clerk                 20   Research
 James        Clerk                 30   Sales
4 rows selected
SQL>
```

- Complex multitable outer joins were not supported by Oracle Rdb.

When more than one column from a table is referenced for an outer join SQL erroneously diagnoses an invalid form of join. The following example is a valid SQL query and should not have caused an error message to be displayed:

```
SQL> select e.employee_id, sh.salary_amount, jh.job_start
cont> from employees e, job_history jh, salary_history sh
cont> where
cont>     e.employee_id = sh.employee_id (+)
cont>     and e.employee_id = jh.employee_id (+)
cont>     and e.department_code = jh.department_code (+)
cont>     and e.employee_id = '00164';
%SQL-F-ONETABLEJOIN, a table may be outer joined to at most one other table
```

These problems have been corrected in Oracle Rdb7 Release 7.0.3.1. SQL now processes this style of join operator correctly and more generally than in previous releases.

2.2.3.1 Outer Joins

An outer join extends the result of a simple join. An outer join returns all rows that satisfy the join condition and those rows from one table for which no rows from the other satisfy the join condition. Such rows are not returned by a simple join. To write a query that performs an outer join of tables A and B and returns all rows from A, apply the outer join operator (+) to all columns of B in the join condition. For all rows in A that have no matching rows in B, Oracle Rdb returns NULL for any select list expressions containing columns of B.

Outer join queries are subject to the following rules and restrictions:

- The (+) operator can appear only in the WHERE clause and can be applied only to a column of a table or view.
- If A and B are joined by multiple join conditions, you must use the (+) operator in all these conditions. If you do not, Oracle Rdb will return only the rows resulting from a simple join, but without a warning or error to advise you that you do not have the results of an outer join.
- The (+) operator can be applied only to a column, not to an arbitrary expression. However, an arbitrary expression can contain a column marked with the (+) operator.
- A condition containing the (+) operator cannot be combined with another condition using the OR logical operator.
- A condition cannot use the IN comparison operator to compare a column marked with the (+) operator with an expression.
- A condition cannot compare any column marked with the (+) operator with a subquery.

If the WHERE clause contains a condition that compares a column from table B with a constant, the (+) operator must be applied to the column so that Oracle Rdb returns the rows from table A for which it has generated NULLs for this column. Otherwise Oracle Rdb will return only the results of a simple join.

In a query that performs outer joins of more than two pairs of tables, a single table can be the NULL-generated table for only one other table. For this reason, you cannot apply the (+) operator to columns of B in the join condition for A and B, and the join condition for B and C.

2.2.3.2 Outer Join Examples

The examples in this section extend the results of an inner join (Equijoin) between the EMP and the DEPT tables.

```
SQL> SELECT ename, job, dept.deptno, dname
cont>  FROM emp, dept
cont>  WHERE emp.deptno = dept.deptno;
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 King         President             10   Accounting
 Blake        Manager               30   Sales
 Clark        Manager               10   Accounting
 Jones        Manager               20   Research
 Ford         Analyst               20   Research
 Smith        Clerk                 20   Research
 Allen        Salesman              30   Sales
 Ward         Salesman              30   Sales
 Martin       Salesman              30   Sales
 Scott        Analyst               20   Research
 Turner       Salesman              30   Sales
 Adams        Clerk                 20   Research
 James        Clerk                 30   Sales
 Miller       Clerk                 10   Accounting
14 rows selected
```

The following query uses an outer join to extend the results of the Equijoin example above:

```
SQL> SELECT ename, job, dept.deptno, dname
cont>  FROM emp, dept
cont>  WHERE emp.deptno (+) = dept.deptno;
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 King         President             10   Accounting
 Clark        Manager               10   Accounting
 Miller       Clerk                 10   Accounting
 Jones        Manager               20   Research
 Ford         Analyst               20   Research
 Smith        Clerk                 20   Research
 Scott        Analyst               20   Research
 Adams        Clerk                 20   Research
 Blake        Manager               30   Sales
 Allen        Salesman              30   Sales
 Ward         Salesman              30   Sales
 Martin       Salesman              30   Sales
 Turner       Salesman              30   Sales
 James        Clerk                 30   Sales
 NULL         NULL                  40   Operations
15 rows selected
```

In this outer join, Oracle Rdb returns a row containing the Operations department even though no employees work in this department. Oracle Rdb returns NULL in the ENAME and JOB columns for this row. The join query in this example selects only departments that have employees.

The following query uses an outer join to extend the results of the preceding example:

```
SQL> SELECT ename, job, dept.deptno, dname
cont>  FROM emp, dept
cont>  WHERE emp.deptno (+) = dept.deptno
cont>     AND job (+) = 'Clerk';
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 Miller       Clerk                 10   Accounting
 Smith        Clerk                 20   Research
 Adams        Clerk                 20   Research
 James        Clerk                 30   Sales
 NULL         NULL                  40   Operations
5 rows selected
```

In this outer join, Oracle Rdb returns a row containing the Operations department even though no clerks work in this department. The (+) operator on the JOB column ensures that rows for which the JOB column is NULL are also returned. If this (+) were omitted, the row containing the Operations department would not be returned because its JOB value is not CLERK.

```
SQL> SELECT ename, job, dept.deptno, dname
cont>  FROM emp, dept
cont>  WHERE emp.deptno (+) = dept.deptno
cont>     AND job = 'Clerk';
 EMP.ENAME    EMP.JOB      DEPT.DEPTNO   DEPT.DNAME
 Miller       Clerk                 10   Accounting
 Smith        Clerk                 20   Research
 Adams        Clerk                 20   Research
 James        Clerk                 30   Sales
4 rows selected
```

This example shows four outer join queries on the CUSTOMERS, ORDERS, LINEITEMS, and PARTS tables:

```
SQL> SELECT custno, custname
cont>  FROM customers
cont>  ORDER BY custno;
      CUSTNO   CUSTNAME
           1   Angelic Co
           2   Believable Co
           3   Cables R Us
3 rows selected

SQL>
SQL> SELECT orderno, custno, orderdate
cont>  FROM orders
cont>  ORDER BY orderno;
     ORDERNO        CUSTNO   ORDERDATE
        9001             1   1999-10-13
        9002             2   1999-10-13
        9003             1   1999-10-20
        9004             1   1999-10-27
        9005             2   1999-10-31
5 rows selected

SQL>
SQL> SELECT orderno, lineno, partno, quantity
cont>  FROM lineitems
cont>  ORDER BY orderno, lineno;
     ORDERNO        LINENO        PARTNO      QUANTITY
        9001             1           101            15
        9001             2           102            10
        9002             1           101            25
        9002             2           103            50
        9003             1           101            15
        9004             1           102            10
        9004             2           103            20
7 rows selected
```

```
SQL>
SQL> SELECT partno, partname
cont>  FROM parts
cont>  ORDER BY partno;
      PARTNO   PARTNAME
         101   X-Ray Screen
         102   Yellow Bag
         103   Zoot Suit
3 rows selected
```

The customer Cables R Us has placed no orders, and order number 9005 has no line items.

The following outer join returns all customers and the dates they placed orders. The (+) operator ensures that customers who placed no orders are also returned:

```
SQL> SELECT custname, orderdate
cont>  FROM customers, orders
cont>  WHERE customers.custno = orders.custno (+)
cont>  ORDER BY customers.custno, orders.orderdate;
 CUSTOMERS.CUSTNAME   ORDERS.ORDERDATE
 Angelic Co           1999-10-13
 Angelic Co           1999-10-20
 Angelic Co           1999-10-27
 Believable Co        1999-10-13
 Believable Co        1999-10-31
 Cables R Us          NULL
6 rows selected
```

The following outer join builds on the result of the previous one by adding the LINEITEMS table to the FROM clause, columns from this table to the select list, and a join condition joining this table to the ORDERS table to the WHERE clause. This query joins the results of the previous query to the LINEITEMS table and returns all customers, the dates they placed orders, the part number, and the quantity of each part they ordered. The first (+) operator serves the same purpose as in the previous query. The second (+) operator ensures that orders with no line items are also returned:

```
SQL> SELECT custname, orderdate, partno, quantity
cont>  FROM customers, orders, lineitems
cont>  WHERE customers.custno = orders.custno (+)
cont>     AND orders.orderno = lineitems.orderno (+)
cont>  ORDER BY customers.custno, orders.orderdate, lineitems.partno;
 CUSTOMERS.CUSTNAME  ORDERS.ORDERDATE  LINEITEMS.PARTNO  LINEITEMS.QUANTITY
 Angelic Co          1999-10-13                   101                  15
 Angelic Co          1999-10-13                   102                  10
 Angelic Co          1999-10-20                   101                  15
 Angelic Co          1999-10-27                   102                  10
 Angelic Co          1999-10-27                   103                  20
 Believable Co       1999-10-13                   101                  25
 Believable Co       1999-10-13                   103                  50
 Believable Co       1999-10-31                  NULL                NULL
 Cables R Us         NULL                        NULL                NULL
9 rows selected
```

The following outer join builds on the result of the previous one by adding the PARTS table to the FROM clause, the PARTNAME column from this table to the select list, and a join condition joining this table to the LINEITEMS table to the WHERE clause. This query joins the results of the previous query to the PARTS table to return all customers, the dates they placed orders, the quantity, and the name of each part they ordered. The first two (+) operators serve the same purposes as in the previous query. The third (+) operator ensures that rows with NULL part numbers are also returned:

```
SQL> SELECT custname, orderdate, quantity, partname
cont>  FROM customers, orders, lineitems, parts
cont>  WHERE customers.custno = orders.custno (+)
cont>     AND orders.orderno = lineitems.orderno (+)
cont>     AND lineitems.partno = parts.partno (+)
cont>  ORDER BY customers.custno, orders.orderdate, parts.partno;
 CUSTOMERS.CUSTNAME   ORDERS.ORDERDATE   LINEITEMS.QUANTITY   PARTS.PARTNAME
 Angelic Co           1999-10-13                       15   X-Ray Screen
 Angelic Co           1999-10-13                       10   Yellow Bag
 Angelic Co           1999-10-20                       15   X-Ray Screen
 Angelic Co           1999-10-27                       10   Yellow Bag
 Angelic Co           1999-10-27                       20   Zoot Suit
 Believable Co        1999-10-13                       25   X-Ray Screen
 Believable Co        1999-10-13                       50   Zoot Suit
 Believable Co        1999-10-31                     NULL   NULL
 Cables R Us          NULL                           NULL   NULL
9 rows selected
```

2.2.4 MAPPING VALUES Not Supported for CREATE INDEX in IMPORT

In previous releases of Oracle Rdb, the SQL IMPORT statement did not allow an index to be created or replaced which used the MAPPING VALUES clause. This restriction has now been lifted. The SQL IMPORT statement now supports the full CREATE INDEX syntax as a subclause, allowing indexes to be created and replaced during the IMPORT operation.

The following example shows the error which was generated in previous releases:

```
SQL> import database
cont>     from ppp
cont>     filename MAPP
cont> create index PERSON_INDEX
cont>     on PERSON (employee_id mapping values 0000.00 to 2001.9)
cont> ;
%SQL-F-NOMAPIMPO, Mapping Values on CREATE INDEX within an IMPORT is not
supported
```

A workaround to this problem is to use the DROP INDEX statement to remove the index (either before the EXPORT or after the IMPORT) and then re-create the index when the database is complete. The corrected behavior in this update to Oracle Rdb will be more efficient, as the index will only be built once.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.2.5 SELECT Returns Incorrect Values for GROUP BY Queries

Bug 882722

Prior releases of Oracle Rdb7 would return incorrect results when similar expressions appeared in the select list of a GROUP BY query. SQL attempted to eliminate common expressions and incorrectly folded together the results of these similar expressions.

This problem affected the use of the functions CAST, EXTRACT, TRIM and the subtraction operator that resulted in different INTERVAL data types.

The following example shows this problem where the EXTRACT of the MONTH field is incorrectly returning the year:

```
SQL> select fundnum,
cont>       extract(year from distribution_date) as yr,
cont>       extract(month from distribution_date) as mth,
cont>       count(*),
cont>       sum(distribution_rate)
cont> from fund_distributions
cont> group by fundnum,
cont>          extract(year from distribution_date),
cont>          extract(month from distribution_date);
     FUNDNUM            YR           MTH
         202          1999          1999             1  2.533465999999999E-003
         245          1999          1999             1  6.772000000000000E-002
.
.
.
```

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.2.6 Restrictions on CURRENT_TIMESTAMP Default Value Lifted

In previous versions of Oracle Rdb, the CURRENT_TIMESTAMP built-in FUNCTION was controlled by the setting of the DEFAULT DATE FORMAT, or DIALECT of the session. When these were set to SQL92, CURRENT_ TIMESTAMP could only be used as a default for a TIMESTAMP domain or column. When set to VMS or the default dialect SQL040, then CURRENT_ TIMESTAMP could only be used as a default for a DATE VMS domain or column.

This restriction was problematic when tables were created with columns of both DATE VMS and TIMESTAMP and these columns wished to use the default of CURRENT_TIMESTAMP. SQL would reject the CREATE TABLE definition no matter what setting was active for the DEFAULT DATE FORMAT. In a similar way, when different tables were created the DEFAULT DATE FORMAT or DIALECT had to be changed between each CREATE/ALTER DOMAIN or TABLE statement.

This restriction has been lifted in this release of Oracle Rdb. SQL now uses the column or domain data type when checking the usage of the CURRENT_ TIMESTAMP as a default value. If the column or domain is of type TIMESTAMP then CURRENT_TIMESTAMP returns a value of type TIMESTAMP, otherwise it will be DATE VMS.

Oracle Rdb has also relaxed the fractional seconds precision used by the DEFAULT clause for CURRENT_TIMESTAMP, CURRENT_TIME, TIMESTAMP, TIME, and INTERVAL literals. They no longer need to exactly match those of the column or domain. Obviously, if the DEFAULT allows more precision than the column, then truncation will occur during assignment.

The following example shows the errors reported when the DIALECT is set to SQL92:

```
SQL> create table C_T_TEST1
cont>     (a timestamp default current_timestamp);
SQL>
SQL> create table C_T_TEST2
cont>     (a timestamp(1) default current_timestamp(2));
%SQL-F-DEFVALINC, You specified a default value for A which is inconsistent
with its data type
SQL>
SQL> create table C_T_TEST3
cont>     (a integer,
cont>      b timestamp(0) default timestamp'1999-7-2:12:00:00.99');
%SQL-F-DEFVALINC, You specified a default value for B which is inconsistent
with its data type
SQL>
SQL> create table C_T_TEST4
cont>     (a date vms default current_timestamp);
%SQL-F-DEFVALINC, You specified a default value for A which is inconsistent
with its data type
```

The following example shows the new behavior:

```
SQL> set dialect 'sql92';
SQL> attach 'file db$:scratch';
SQL>
SQL> create table C_T_TEST1
cont>     (a timestamp default current_timestamp);
SQL>
SQL> create table C_T_TEST2
cont>     (a timestamp(1) default current_timestamp(2));
SQL>
SQL> create table C_T_TEST3
cont>     (a integer,
cont>      b timestamp(0) default timestamp'1999-7-2:12:00:00.99');
SQL> insert into C_T_TEST3 (a) values (0);
cont>     (a integer,
cont>      b timestamp(0) default timestamp'1999-7-2:12:00:00.99');
SQL> insert into C_T_TEST3 (a) values (0);
1 row inserted
SQL> select * from C_T_TEST3;
           A   B
           0   1999-07-02 12:00:00
1 row selected
SQL>
SQL> create table C_T_TEST4
cont>     (a date vms default current_timestamp);
```

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.2.7 SQL Uses Read/Write Transactions Against Standby Database

In prior releases of Oracle Rdb7, the SQL interface, in particular interactive and dynamic SQL, would attempt to execute a read/write transaction on a database which was currently a standby database. This type of database is considered read-only and starting a read/write transaction fails. This problem may cause connections from ODBC, SQL*Net for Rdb, and SQL/Services to fail.

A workaround for this problem is to restore the standby database using the RMU/RESTORE/TRANSACTION_MODE=READ_ONLY command which permits only read-only transactions on the database. Current releases of SQL do detect this setting and will then operate correctly. When the standby database is required as the master database (following a system failure), the ALTER DATABASE ... ALTER TRANSACTION MODE (READ WRITE) statement should be used to enable read/write transactions for the database.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. SQL now determines that the database is a standby database and uses by default READ ONLY transactions for metadata queries.

2.2.8 Domain Constraints Did Not Support Function Calls Passing VALUE

Bug 808378

In prior releases of Oracle Rdb, attempts to pass the special VALUE keyword to a user-defined function (external or stored) would cause the CREATE or ALTER DOMAIN statement to bugcheck.

The following example shows the resulting exception:

```
SQL> create module myModule language sql
cont> function myFunction( in :x char(1)) return char(1);
cont> begin
cont>  return case :x when 'A' then :x else null end;
cont> end;
cont> end module;
SQL> create domain myDomain char(1)
cont>     check( myFunction(value) is not null) not deferrable;
%SQL-I-BUGCHKDMP, generating bugcheck dump file DISK1:[TEST]SQLBUGCHK.DMP;
%SQL-F-BUGCHK, There has been a fatal error. Please contact your Oracle support
representative. SQL$SEMASS - 9
```

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. SQL now supports the passing of the VALUE keyword (or the name of the domain) to a user-defined function in a domain CHECK constraint.

2.2.9 CREATE VIEW May Fail on Large or Complex View Definition

Bug 909035

Definitions of very complex views, view definitions referencing long table and column names, or views based on tables with many columns may fail with one of the following errors:

- Obsolete metadata

  ```
  %RDB-E-NO_META_UPDATE, metadata update failed
  -RDB-E-OBSOLETE_METADA, request references metadata objects that no longer
  exist
  -RDMS-F-BAD_SYM, unknown field symbol - XXX
  ```

  Here XXX represents a partial or corrupt column name from the view definition.

- Invalid BLR

  ```
  %RDB-E-NO_META_UPDATE, metadata update failed
  -RDB-E-INVALID_BLR, request BLR is incorrect at offset NNN
  ```

  Here NNN represents some numeric offset into the definition.

These problems occur when SQL attempts to build a binary version of the view row selection expression (RSE) which exceeds 65,535 (64K) bytes in length. This was the maximum size supported in prior versions of Oracle Rdb, and remains the limit for Oracle CDD/Repository and older versions of Oracle Rdb.

There is no simple workaround for this restriction. However, you could use shorter table and column names, or break the view definition into references to other views.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. Oracle Rdb has been enhanced to support much larger view definitions. SQL will use a new format for the view definition if it requires more than 65,535 bytes to describe. However, this new format is not supported by Oracle CDD/Repository or older versions of Oracle Rdb, and these view definitions cannot be created in these products.

2.2.10 Incomplete Support for Multischema Databases for Query Outlines

Bug 886825

In prior releases of Oracle Rdb, the support for multischema databases was incomplete for query outlines for the following reasons:

- The CREATE OUTLINE statement did not support the STORED NAME IS clause. Therefore, user-specified names could not be provided for query outlines.

  The workaround for this problem was to accept the unique names generated by SQL.

- The OPTIMIZE USING clause on the DECLARE CURSOR, SELECT, UPDATE, and DELETE statements did not allow a fully-qualified query outline name.

  A workaround for this problem was to specify the query outline name without the catalog or schema reference.

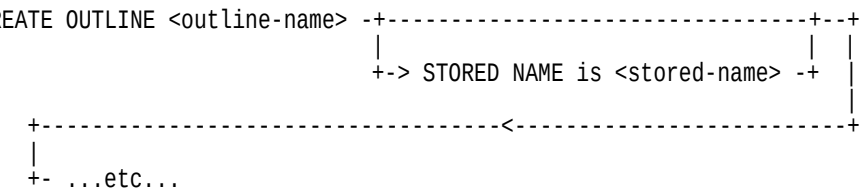
The following example shows the error reported when an attempt was made to use a fully-qualified name for the outline SAMPLE:

```
SQL> select count(a)
cont>  from SAMPLE_TABLE
cont>  where b <> CURRENT_TIMESTAMP
cont>  optimize using outer_space.inner_space.sample;
 optimize using outer_space.inner_space.sample;
                           ^
%SQL-W-LOOK_FOR_STT, Syntax error, looking for:
%SQL-W-LOOK_FOR_CON,            AS, FOR, USING, ;,
%SQL-F-LOOK_FOR_FIN,    found . instead
```

These problems have been corrected in Oracle Rdb7 Release 7.0.3.1. SQL now supports the STORED NAME IS clause for the CREATE OUTLINE statement, and the fully-qualified name can be used in the OPTIMIZE USING clause.

Note

If a fully-qualified name is used for the OPTIMIZE USING clause, then the catalog and schema must exist within the database. However, for consistency with single schema databases, the outline need not exist at the time of the query processing. If the outline is not created prior to execution of the query, then this name will be ignored by the Oracle Rdb optimizer.



2.2.11 UNKNOWN_VAR Error Reported During Select from a View

Bug 903619

In prior versions of Oracle Rdb, the error, "unknown variable 1 found in the query string" may be reported when selecting rows from a view.

The problem only occurs when the conditional expressions CASE, COALESCE, NVL, and DECODE are used in a SELECT DISTINCT clause as part of a CREATE VIEW statement. Typically these conditional expressions contain subselects that reference other table contexts. Although the view can be created, it could never be used successfully, as shown in this example:

```
SQL> select * from sample_view;
%RDB-E-INVALID_BLR, request BLR is incorrect at offset 66
-RDMS-E-UNKNOWN_VAR, unknown variable 1 found in the query string
```

This error is caused by an incorrect definition stored in the database by the CREATE VIEW statement. Once this release of Oracle Rdb is installed, the view can be deleted and re-created to correct this problem.

A workaround for this problem is to rewrite the view SELECT clause to use GROUP BY or a derived table in place of the DISTINCT clause. The resulting view will probably be more efficient and perform less table accesses because the complex expressions will not be used for the unique row selection.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. The CREATE VIEW statement now correctly handles conditional expressions with subselects in the view SELECT clause.

2.2.12 SQL Module Language Compiler Fails Unexpectedly

Bug 496320

It is possible that the SQL module language compiler could fail when processing a FETCH statement. The failure occurs when parameters have the same names as columns in the table and are not prefixed with colons. The SQL module language compiler should diagnose this error, but instead it fails to complete the compile.

On OpenVMS VAX systems, the SQL module language compile fails with error similar to those in the following example:

```
$ sql$mod appl
              00000207'GF   14   28  06A7   797         MOVC3   #20,
G^MESSAGE$1_1+MESSAGE$1_1$VAR$3, G^MESSAGE$0_0+MESSAGE$0_0$VAR$0
%MACRO-E-UNDEFSYM, Undefined symbol
                                  06AE
%MACRO-E-UNDEFSYM, Undefined symbol
              00000223'GF   01   28  06CA   804         MOVC3   #1,
G^MESSAGE$1_1+MESSAGE$1_1$VAR$6, G^MESSAGE$0_0+MESSAGE$0_0$VAR$0
%MACRO-E-UNDEFSYM, Undefined symbol
                                  06D1
%MACRO-E-UNDEFSYM, Undefined symbol

There were 4 errors, 0 warnings and 0 information messages, on lines:
  797 (2)       804 (2)

MACRO/NOLIST MAR_SPEC
```

On OpenVMS Alpha systems the following is produced:

```
$ sql$mod appl
%SQL-E-CHKINIT, 2 integrity check errors in initial IL & ST for module
READ_BD_FOR_BATCH
%SQL-I-BUGCHKDMP, generating bugcheck dump file DISK1:[TEST]SQLBUGCHK.DMP;
```

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. The SQL module language compiler has been enhanced to detect this situation and report a new error message, as shown in the following example:

```
$ sql$mod appl
   ARCHIVED
   1
%SQL-F-INVINTOTAR, (1) target for an INTO clause must be a variable or
parameter
```

2.2.13 ORDER BY Select Query from a View with ORDER BY on the Same COMPUTED BY Column Returns Wrong Result

Bugs 880630 and 934057

The following ORDER BY select query from a view with ORDER BY on the same COMPUTED BY column returns wrong results:

```
    select F_COMPUTE_BY from test_view order by F_COMPUTE_BY;
               F_COMPUTE_BY
                    5.50
                    5.50
2 rows selected
where the view test_view is defined as :
create view test_view (
LEV_ID                          ,
F_COMPUTE_BY
) as
   select
     T1.LEV_ID,
     T1.F_COMPUTE_BY
     from T1 T1
     order by T1.F_COMPUTE_BY asc ;
;
```

where the column F_COMPUTE_BY is defined as:

```
        COMPUTED BY (select C2.F2 from T2 C2
            where (T1.F1 = C2.F1)));
```

The correct result should be the same as the following query:

```
SQL> sel * from test_view;
 LEV_ID                 F_COMPUTE_BY
 123456                      5.50
 123456                     10.10
2 rows selected
```

Here is the script to reproduce the problem:

```
create table T2 (
    F1 CHAR(26),
    F2 INTEGER
    );

INSERT INTO T2 VALUE ('F1_01', 10);
INSERT INTO T2 VALUE ('F1_02',  5);

create table T1 (
    F1 CHAR(26),
    REV_ID INTEGER,
    PROJ_ID CHAR(6),
    LEV_ID  CHAR(6),
    F_COMPUTE_BY
        computed by (select T2.F2 from T2 T2
            where (T1.F1 = T2.F1)));
```

```
INSERT INTO T1 (F1, REV_ID, PROJ_ID, LEV_ID) VALUE
    ('F1_01', 990413001, '2800', '123456');
INSERT INTO T1 (F1, REV_ID, PROJ_ID, LEV_ID) VALUE
    ('F1_02', 990413002, '2800', '123456');

create index T1_NDX
    on T1 (
    PROJ_ID        asc,
    REV_ID        asc);

create index T2_NDX
    on T2 (
    F1)
    ;
```

This problem occurs when the ORDER BY clause of both the main select query and view query is placed upon the same column where the column is COMPUTED BY as a subquery select.

The current problem is caused by Oracle Rdb 7.0.1.6 where a fix was introduced for bug 651184 to explicitly join the subselect query under the view query if the view query contains ORDER BY clause. This fix has a serious flaw when the subselect query is defined under a COMPUTED BY definition for the column.

Workaround: None.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

2.3 Oracle RMU Errors Fixed

2.3.1 Lock ID Hidden by Stall Message in RMU/SHOW Statistics

Previously, RMU/SHOW Statistics could "hide" the Lock ID value when displaying stall messages if the stall message was long, even when the terminal width was set wide enough to display everything correctly. For example:

```
Node: LLDV00        Oracle Rdb V7.0-01 Performance Monitor  11-AUG-1999 22:39:0
Rate: 3.00 Seconds              Stall Messages          Elapsed: 6 04:26:42.2
Page: 1 of 1          DEV_DISKE:[DEV.DATA]RGN_DB.RDB;2             Mode: Onlie

Process.ID Since..... T Stall.reason.............................. Lock.ID.
20202B96:1 10:37:11.23 W Waiting for client '....Q...' 0000000510000000040000005
```

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. RMU/SHOW Statistics now displays the Lock ID value with an offset from the right margin of the display. Setting the display wider (to 132 columns, for example) allows both the stall message text and the Lock ID to be displayed at one time.

2.3.2 Terminal Width Checked by RMU/SHOW Statistics

RMU/SHOW Statistics assumes a minimum terminal width of 80 columns. When run on a terminal set to less than 80 columns, RMU/SHOW Statistics could bugcheck or be unable to properly format the display.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. RMU/SHOW Statistics now enforces a minimum terminal width of 80 columns. It also enforces a minimum row count of 24. If the terminal is set to less than the minimum number of rows or columns, an error message is displayed.

2.3.3 RMU/SHOW Statistic Physical-Area Event Creation Bugchecks

Attempting to create a user-defined event involving a physical or logical area when used in conjunction with the /NOINTERACTIVE qualifier causes the RMU/SHOW Statistic utility to bugcheck.

The following configuration file entry shows an example of a user-defined event that will cause the problem when the RMU/SHOW Statistic utility is invoked using the NOINTERACTIVE qualifier:

```
EVENT_DESCRIPTION = "ENABLE '  (Extends)' \
    MAX_CUR_TOTAL \
    AREA DEPARTMENTS\
    INITIAL 0 EVERY 1 LIMIT 0 \
    NOTIFY OPER12";
```

The workaround is to use the /INTERACTIVE qualifier, or do not specify an event using a physical or logical area statistic name.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. User-defined events using physical or logical areas in conjunction with the /NOINTERACTIVE qualifier now work as expected.

2.3.4 Ignore Row Caches when Writing RMU/SHOW Statistic Report

Bug 944656

Customers frequently use the RMU/SHOW Statistic utility Write Report (Numbers) to generate screen snapshots which are then processed using various analysis tools. When the database has row caches enabled, the RMU/SHOW Statistic utility maps all available caches into memory. When the database contains a large number of row caches, or the row caches have a large number of slots, or both, this can cause the process P0 space to be exhausted.

The best workaround is to put the row caches in system space (VLM).

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

When row caching is allowed on the database, the "Options" on-screen menu will display the following new options:

- Include row cache information

  This option allows you to include all row cache information in the report. This is the default setting.

- Exclude row cache information

  This option allows you to exclude all row cache information in the report.

You can also use the new configuration variable REPORT_IGNORE_RCACHE to specify the initial report option. The value TRUE indicates that the report should exclude all row cache information. The default value FALSE indicates that the report should include row cache information. This variable can be changed at run time using the Options on-screen menu options that was previously described.

2.3.5 RMU/LOAD from a Record-Oriented Device Caused RMS-F-IOP Error

Previously, attempting to use the RMU/LOAD utility using a record-oriented device (a terminal or mailbox, for example) for data input would fail with an RMS-F-IOP error.

For example:

```
$ RMU/LOAD DB.RDB /RECORD_DEFINITION=FILE=TBL.RRD TBL MBA500:
%RMU-F-FILACCERR, error opening input file MBA500:[DB].UNL;
-RMS-F-IOP, operation invalid for file organization or device
%RMU-I-DATRECSTO,   0 data records stored.
%RMU-F-FTL_LOAD, Fatal error for LOAD operation at 13-JUN-1999 15:13:33.41
```

This problem was caused by an attempt to search for the input file on the specified device to return the full file specification.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. When the input source is a record-oriented device, no search for the input file is done.

2.3.6 RMU/EXTRACT Sometimes Bugchecks when Processing Many Storage Areas

Bug 633334

In previous versions of Oracle Rdb, the RMU/EXTRACT storage area output could be truncated or RMU might bugcheck at EXTRACT_DATABASE +789 if the database had many storage areas. This affects the /ITEM options for DATABASE, IMPORT, and ALL.

RMU uses a special interface call (API) to retrieve all the names of the storage areas. This problem occured when the buffer was incorrectly truncated. This truncated buffer appeared corrupted and so RMU/EXTRACT would bugcheck.

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1. The buffer returned from the storage area query is no longer truncated.

2.3.7 Problem in Reporting Recovered AIJ Sequence Number

In previous releases of Oracle Rdb, when multiple, after-image journals were combined into one journal file, such as a backup of multiple journals, the RMU/RECOVER command's AIJONEDONE trace message could display incorrect information. The RMU/RECOVER command might display the sequence number of the next journal to be recovered instead of the journal just recovered. In the following example, sequence 5 has just been recovered and sequence 6 will be recovered next, but the AIJONEDONE message indicates an incorrect sequence number of 6:

```
%RMU-I-AIJONEDONE, AIJ file sequence 6 roll-forward operations completed
```

This problem has been corrected in Oracle Rdb7 Release 7.0.3.1.

2.3.8 Truncate Table Could Generate RMU/VERIFY Errors

Bug 919653

In rare cases, where the area bit map (ABM) pages are at the beginning of a space management page (SPAM) interval, a truncate of that logical area could corrupt the linkages between the area inventory page (AIP), the area bit map page, and the SPAM page. This corruption was reported by RMU/VERIFY.

The following example shows the output from the VERIFY. The reference to larea_dbid : -1 indicates a deleted logical area on the SPAM page.

```
%RMU-W-AIPLAREID, area inventory page 7 entry #14 contains a
                  reference to logical area 95 that is nonexistent
%RMU-W-BADABMPTR, invalid larea for ABM page 192151 in storage area 1.
                  The SPAM page entry for this page is for a different larea.
                  SPAM larea_dbid : -1 page larea_dbid: 95.
%RMU-W-BADABMPTR, invalid larea for ABM page 192152 in storage area 1.
                  The SPAM page entry for this page is for a different larea.
                  SPAM larea_dbid : -1 page larea_dbid: 95.
```

Issuing an RMU/REPAIR/SPAM/ABM command will correct this error.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

2.4 Row Cache Errors Fixed

2.4.1 Row Cache of One Slot Causes Loop

When the Oracle Rdb7 row cache feature is enabled and if a cache is created specifying only one slot in the cache, a second insert into the cache may cause a process to enter a seemingly infinite loop.

As a possible workaround, always create row caches with at least 2 slots in the cache. Oracle recommends that you create all caches with at least 100 slots.

This problem as been corrected. Caches with only 1 slot no longer cause the to CPU loop.

2.5 Hot Standby Errors Fixed

2.5.1 Stopping Hot Standby on Standby then Master Hangs Standby for 15 Minutes

If the Hot Standby product is properly shutdown on the standby database first, and then immediately thereafter also shutdown properly on the master database, it is possible for the standby database to wait 15 minutes before actually shutting down.

The workaround is to do any of the following:

- Stop Hot Standby on the standby database using the /ABORT qualifier.
- Restart the master database, and properly shutdown the master database.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1. Shutting down the Hot Standby product from the standby database, master database, or both now works as expected.

2.5.2 Monitor ENQLM Minimum Increased to 32767

All OpenVMS Systems

Bug 357439

In previous versions, the Oracle Rdb7 monitor process (RDMMON) was created with a minimum lock limit (ENQLM) of 8192 locks. This minimum has been increased to 32767 locks (the OpenVMS maximum value).

2.6 Oracle Trace Errors Fixed

2.6.1 Incorrect Completion Status Reported by Oracle Trace

Bug 949283

An incorrect value for the status of a request was being stored by Oracle Trace. The stored value was larger than the field definition and also corrupted the value for the Client_PC address. When the Oracle Trace information was formatted in a database, incorrect information was being returned in the queries against tables holding this information.

There is no workaround for this problem.

This problem has been corrected in Oracle Rdb7 Version 7.0.3.1.

3
Documentation Corrections

This chapter provides information not currently available in the Oracle Rdb7 documentation set.

3.1 Documentation Corrections

3.1.1 Partition Clause is Optional on CREATE STORAGE MAP

Bug 642158

In the *Oracle Rdb7 SQL Reference Manual*, the syntax diagram for the CREATE STORAGE MAP statement incorrectly shows the partition clause as required syntax. The partition clause is not a required clause.

This correction will appear in the next publication of the *Oracle Rdb SQL Reference Manual*.

3.1.2 Oracle Rdb Logical Names

The *Oracle Rdb7 Guide to Database Performance and Tuning* contains a table in Chapter 2 summarizing the Oracle Rdb logical names and configuration parameters. The information in the following table supersedes the entries for the RDM$BIND_RUJ_ALLOC_BLKCNT and RDM$BIND_RUJ_EXTEND_BLKCNT logical names.

Logical Name Configuration Parameter	Function
RDM$BIND_RUJ_ALLOC_BLKCNT	Allows you to override the default value of the .ruj file. The block count value can be defined between 0 and 2 billion with a default of 127.
RDM$BIND_RUJ_EXTEND_BLKCNT	Allows you to pre-extend the .ruj files for each process using a database. The block count value can be defined between 0 and 65535 with a default of 127.

3.1.3 Waiting for Client Lock Message

The *Oracle Rdb7 Guide to Database Performance and Tuning* contains a section in Chapter 3 that describes the Performance Monitor Stall Messages screen. The section contains a list describing the "Waiting for" messages. The description of the "waiting for client lock" message was missing from the list.

A client lock indicates that an Oracle Rdb metadata lock is in use. The term client indicates that Oracle Rdb is a client of the Oracle Rdb locking services. The metadata locks are used to guarantee memory copies of the metadata (table, index, and column definitions) are consistent with the on-disk versions.

The "waiting for client lock" message means the database user is requesting an incompatible locking mode. For example, when trying to delete a table which is in use, the drop operation requests a PROTECTED WRITE lock on the metadata object (such as a table) which is incompatible with the existing PROTECTED READ lock currently used by others of the table.

These metadata locks consist of three longwords. The lock is displayed in text format first, followed by its hexadecimal representation. The text version masks out nonprintable characters with a period (.).

The leftmost value seen in the hexadecimal output contains the ID of the object. The following ID describes the tables, routines, modules and storage map areas.

- For tables and views, the ID represents the unique value found in the RDB$RELATION_ID column of the RDB$RELATIONS system table for the given table.
- For routines, the ID represents the unique value found in the RDB$ROUTINE_ID column of the RDB$ROUTINES system table for the given routine.
- For modules, the ID represents the unique value found in the RDB$MODULE_ID column of the RDB$MODULES system table for the given module.
- For storage map areas, the ID presents the physical area ID. The "waiting for client lock" message on storage map areas is very rare. This may be raised for databases that have been converted from versions prior to Oracle Rdb 5.1.

The next value displayed signifies the object type. The following table describes objects and their hexadecimal type values:

Table 3–1 Object Type Values

Object	Hexadecimal Value
Tables or views	00000004
Routines	00000006
Modules	00000015
Storage map areas	0000000E

The last value in the hexadecimal output represents the lock type. The value 55 indicates this is a client lock.

The following example shows a "waiting for client" lock message from the Stall Messages screen:

```
Process.ID Since......   Stall.reason.............................. Lock.ID.
46001105:2 10:40:46.38 - waiting for client '........' 000000190000000400000055
                                                      1        2        3       4
```

The following list describes each part of the client lock:

1 ..... indicates nonprintable characters.

2 00000019 indicates unique identifier hex value 19 (RDB$RELATION_ID = 25).

3 00000004 indicates object type 4 which is a table.

4 00000055 indicates this is a client lock.

To determine the name of the referenced object given the Lock ID the following queries can be used based on the object type:

```
SQL> SELECT RDB$RELATION_NAME FROM RDB$RELATIONS WHERE RDB$RELATION_ID = 25;
SQL> SELECT RDB$MODULE_NAME FROM RDB$MODULES WHERE RDB$MODULE_ID = 12;
SQL> SELECT RDB$ROUTINE_NAME FROM RDB$ROUTINES WHERE RDB$ROUTINE_ID = 7;
```

Note

Because the full client lock output is long, it may require more space than is allotted for the Stall.reason column and therefore can be overwritten by the Lock.ID. column output.

For more detailed lock information, perform the following steps:

1. Press the L option from the horizontal menu to display a menu of Lock IDs.
2. Select the desired Lock ID.

3.1.4 Documentation Error in *Oracle Rdb7 Guide to Database Performance and Tuning*

The *Oracle Rdb7 Guide to Database Performance and Tuning, Volume 2* contains an error in section C.7, "Displaying Sort Statistics with the R Flag".

When describing the output from this debugging flag, bullet 9 states:

> **Work File Alloc** indicates how many work files were used in the sort operation. A zero (0) value indicates that the sort was accomplished completely in memory.

This is incorrect. This statistic should be described as shown:

> **Work File Alloc** indicates how much space (in blocks) was allocated in the work files for this sort operation. A zero (0) value indicates that the sort was accomplished completely in memory.

This error will be corrected in a future release of *Oracle Rdb Guide to Database Performance and Tuning*.

3.1.5 SET FLAGS Option IGNORE_OUTLINE Not Available

Bug 510968

The *Oracle Rdb7 SQL Reference Manual* described the option IGNORE_ OUTLINE in Table 7-6 of the SET FLAGS section. However, this keyword was not implemented in Oracle Rdb7.

This has been corrected in this release of Oracle Rdb7. This keyword is now recognized by the SET FLAGS statement. As a workaround the logical name RDMS$BIND_OUTLINE_FLAGS "I" can be used to set this attribute.

3.1.6 SET FLAGS Option INTERNALS Not Described

The *Oracle Rdb7 SQL Reference Manual* does not describe the option INTERNALS in Table 7-6 in the SET FLAGS section. This keyword was available in first release of Oracle Rdb7 and is used to enable debug flags output for internal queries such as constraints and triggers. It can be used in conjunction with other options such as STRATEGY, BLR, and EXECUTION. For example, the following flag settings are equivalent to defining the RDMS$DEBUG_FLAGS as

ISn and shows the strategy used by the trigge's actions on the AFTER DELETE trigger on the EMPLOYEES table.

```
SQL> SET FLAGS 'STRATEGY, INTERNAL, REQUEST_NAME';
SQL> SHOW FLAGS

Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   INTERNALS,STRATEGY,PREFIX,REQUEST_NAMES
SQL> DELETE FROM EMPLOYEES WHERE EMPLOYEE_ID = '00164';
~S: Trigger name  EMPLOYEE_ID_CASCADE_DELETE
Get     Temporary relation      Retrieval by index of relation DEGREES
  Index name  DEG_EMP_ID [1:1]
~S: Trigger name  EMPLOYEE_ID_CASCADE_DELETE
Get     Temporary relation      Retrieval by index of relation JOB_HISTORY
  Index name  JOB_HISTORY_HASH [1:1]
~S: Trigger name  EMPLOYEE_ID_CASCADE_DELETE
Get     Temporary relation      Retrieval by index of relation SALARY_HISTORY
  Index name  SH_EMPLOYEE_ID [1:1]
~S: Trigger name  EMPLOYEE_ID_CASCADE_DELETE
Conjunct        Get     Retrieval by index of relation DEPARTMENTS
  Index name  DEPARTMENTS_INDEX [0:0]
Temporary relation      Get     Retrieval by index of relation EMPLOYEES
  Index name  EMPLOYEES_HASH [1:1]       Direct lookup
1 row deleted
```

3.1.7 Documentation for VALIDATE_ROUTINE Keyword for SET FLAGS

The SET FLAGS section of the *Oracle Rdb7 SQL Reference Manual* omitted the description of the VALIDATE_ROUTINE keyword (which can be negated as NOVALIDATE_ROUTINE). This keyword enables the re-validation of an invalidated stored procedure or function. This flag has the same action as the logical RDMS$VALIDATE_ROUTINE or the UNIX environment variable SQL_VALIDATE_ROUTINE described in the *Oracle Rdb7 Guide to Database Performance and Tuning*.

This example shows the re-validation of a stored procedure. When the stored routine is successfully prepared (but not executed), the setting of VALIDATE_ ROUTINE causes the entry for this routine in the RDB$ROUTINES system table to be set as valid.

```
SQL> SET TRANSACTION READ WRITE;
SQL> SET FLAGS 'VALIDATE_ROUTINE';
SQL> SET NOEXECUTE;
SQL> CALL ADD_EMPLOYEE ('Smith');
SQL> SET EXECUTE;
SQL> COMMIT;
```

In this example, the use of the SET NOEXECUTE statement in interactive SQL allows the stored routine to be successfully compiled, but it is not executed.

3.1.8 Documentation for Defining the RDBSERVER Logical Name

Bugs 460611 and 563649.

Sections 4.3.7.1 and 4.3.7.2 in the *Oracle Rdb7 for OpenVMS Installation and Configuration Guide* provide the following examples for defining the RDBSERVER logical name:

```
$ DEFINE RDBSERVER SYS$SYSTEM:RDBSERVER70.EXE
 and
$ DEFINE RDBSERVER SYS$SYSTEM:RDBSERVER61.EXE
```

These definitions are inconsistent with other command procedures that attempt to reference the RDBSERVERxx.EXE image. The following is one example where the RDBSERVER.COM procedure references SYS$COMMON:<SYSEXE> and SYS$COMMON:[SYSEXE], rather than SYS$SYSTEM:

```
$   if .not. -
       ((f$locate ("SYS$COMMON:<SYSEXE>",rdbserver_image) .ne. log_len) .or. -
        (f$locate ("SYS$COMMON:[SYSEXE]",rdbserver_image) .ne. log_len))
$   then
$       say "''rdbserver_image' is not found in SYS$COMMON:<SYSEXE>"
$       say "RDBSERVER logical is ''rdbserver_image'"
$       exit
$   endif
```

In this case, if the logical name were defined as instructed in the *Oracle Rdb7 for OpenVMS Installation and Configuration Guide*, the image would not be found.

The *Oracle Rdb7 for OpenVMS Installation and Configuration Guide* should define the logical name as follows:

```
DEFINE RDBSERVER SYS$COMMON:<SYSEXE>RDBSERVER70.EXE
 and
DEFINE RDBSERVER SYS$COMMON:<SYSEXE>RDBSERVER61.EXE
```

3.1.9 Undocumented SET Commands and Language Options

The following SET statements were omitted from the Oracle Rdb7 documentation.

3.1.9.1 QUIET COMMIT Option

The SET QUIET COMMIT statement (for interactive and dynamic SQL), the module header option QUIET COMMIT, the /QUIET_COMMIT (and /NOQUIET_COMMIT) qualifier for SQL module language, or the /SQLOPTIONS=QUIET_COMMIT (and NOQUIET_COMMIT) option for the SQL language precompiler allows the programmer to control the behavior of the COMMIT and ROLLBACK statements in cases where there is no active transaction.

By default, if there is no active transaction, SQL will raise an error when COMMIT or ROLLBACK is executed. This default is retained for backward compatibility for applications that may wish to detect the situation. If QUIET COMMIT is set to ON, then a COMMIT or ROLLBACK executes successfully when there is no active transaction.

Note

Within a compound statement, the COMMIT and ROLLBACK statements in this case are ignored.

Examples

In interactive or dynamic SQL, the following SET command can be used to disable or enable error reporting for COMMIT and ROLLBACK when no transaction is active. The parameter to the SET command is a string literal or host variable containing the keyword ON or OFF. The keywords may be in any case (upper, lower, or mixed).

```
SQL> COMMIT;
%SQL-F-NO_TXNOUT, No transaction outstanding
SQL> ROLLBACK;
%SQL-F-NO_TXNOUT, No transaction outstanding
SQL> SET QUIET COMMIT 'on';
SQL> ROLLBACK;
SQL> COMMIT;
SQL> SET QUIET COMMIT 'off';
SQL> COMMIT;
%SQL-F-NO_TXNOUT, No transaction outstanding
```

In the SQL module language or precompiler header, the clause QUIET COMMIT can be used to disable or enable error reporting for COMMIT and ROLLBACK when no transaction is active. The keyword ON or OFF must be used to enable or disable this feature. The following example enables QUIET COMMIT so that no error is reported if a COMMIT is executed when no transaction is active. For example:

```
MODULE TXN_CONTROL
LANGUAGE BASIC
PARAMETER COLONS
QUIET COMMIT ON

PROCEDURE S_TXN (SQLCODE);
SET TRANSACTION READ WRITE;

PROCEDURE C_TXN (SQLCODE);
COMMIT;
```

3.1.9.2 COMPOUND TRANSACTIONS Option

The SET COMPOUND TRANSACTIONS statement (for interactive and dynamic SQL) and the module header option COMPOUND TRANSACTIONS allows the programmer to control the SQL behavior for starting default transactions for compound statements.

By default, if there is no current transaction, SQL will start a transaction before executing a compound statement or stored procedure. However, this may conflict with the actions within the procedure, or may start a transaction for no reason if the procedure body does not perform any database access. This default is retained for backward compatibility for applications that may expect a transaction to be started for the procedure.

If COMPOUND TRANSACTIONS is set to EXTERNAL, then SQL starts a transaction before executing the procedure; otherwise, if it is set to INTERNAL, it allows the procedure to start a transaction as required by the procedure execution.

Examples

In interactive or dynamic SQL, the following SET command can be used to disable or enable transactions started by the SQL interface. The parameter to the SET command is a string literal or host variable containing the keyword INTERNAL or EXTERNAL. The keywords may be in any case (upper, lower, or mixed). For example:

```
SQL> SET COMPOUND TRANSACTIONS 'internal';
SQL> CALL START_TXN_AND_COMMIT ();
SQL> SET COMPOUND TRANSACTIONS 'external';
SQL> CALL UPDATE_EMPLOYEES (...);
```

In the SQL module language or precompiler header, the clause COMPOUND TRANSACTIONS can be used to disable or enable starting a transaction for procedures. The keyword INTERNAL or EXTERNAL must be used to enable or disable this feature.

```
MODULE TXN_CONTROL
LANGUAGE BASIC
PARAMETER COLONS
COMPOUND TRANSACTIONS INTERNAL

PROCEDURE S_TXN (SQLCODE);
BEGIN
SET TRANSACTION READ WRITE;
END;

PROCEDURE C_TXN (SQLCODE);
BEGIN
COMMIT;
END;
```

3.1.10 Undocumented Size Limit for Indexes with Keys Using Collating Sequences

Bug 586079

When a column is defined with a collating sequence, the index key is specially encoded to incorporate the correct ordering (collating) information. This special encoding takes more space than keys encoded for ASCII (the default when no collating sequence is used). Therefore, the encoded string uses more than the customary one byte per character of space within the index. This is true for all versions of Oracle Rdb that support collating sequences.

For all collating sequences, except Norwegian, the space required is approximately 9 bytes for every 8 characters. So, a CHAR (24) column will require approximately 27 bytes. For Norwegian collating sequences, the space required is approximately 10 bytes for every 8 characters.

The space required for encoding the string must be taken into account when calculating the size of an index key against the limit of 255 bytes. Suppose a column defined with a collating sequence of GERMAN was used in an index. The length of that column is limited to a maximum of 225 characters because the key will be encoded in 254 bytes.

The following example demonstrates how a 233 character column, defined with a German collating sequence and included in an index, exceeds the index size limit of 255 bytes, even though the column is defined as less than 255 characters in length:

```
SQL> CREATE DATABASE
cont>      FILENAME 'TESTDB.RDB'
cont>      COLLATING SEQUENCE GERMAN GERMAN;
SQL> CREATE TABLE EMPLOYEE_INFO (
cont>      EMP_NAME CHAR (233));
SQL> CREATE INDEX EMP_NAME_IDX
cont>      ON EMPLOYEE_INFO (
cont>      EMP_NAME     ASC)
cont>      TYPE IS SORTED;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-INDTOOBIG, requested index is too big
```

3.1.11 Changes to RMU/REPLICATE AFTER/BUFFERS Command

The behavior of the RMU/REPLICATE AFTER/BUFFERS command has been changed. The /BUFFERS qualifier may be used with either the CONFIGURE option or the START option.

When using local buffers, the AIJ log roll-forward server (LRS) will use a minimum of 4096 buffers. The value provided to the /BUFFERS qualifier will be accepted, but it will be ignored if it is less than 4096. In addition, further parameters will be checked and the number of buffers may be increased if the resulting calculations are greater than the number of buffers specified by the /BUFFERS qualifier. If the database is configured to use more than 4096 AIJ request blocks (ARBs), then the number of buffers may be increased to the number of ARBs configured for the database. The LRS ensures that there are at least 10 buffers for every possible storage area in the database. Thus, if the total number of storage areas (both used and reserved) multiplied by 10 results in a greater number of buffers, that number will be used.

When global buffers are used, the number of buffers used by the AIJ log roll-forward server is determined as follows:

- If the /BUFFERS qualifier is omitted and the /ONLINE qualifier is specified, the number of buffers will default to the previously configured value, if any, or 256, whichever is larger.
- If the /BUFFERS qualifier is omitted and the /ONLINE qualifier is not specified or the /NOONLINE is specified, the number of buffers will default to the maximum number of global buffers allowed per user ("USER LIMIT"), or 256, whichever is larger.
- If the /BUFFERS qualifier is specified, that value must be at least 256, and it may not be greater than the maximum number of global buffers allowed per user ("USER LIMIT").

The /BUFFER qualifier now enforces a minimum of 256 buffers for the AIJ log roll-forward server. The maximum number of buffers allowed is still 524288 buffers.

3.1.12 Change in the Way RDMAIJ Server is Set Up in UCX

Starting with Oracle Rdb V7.0.2.1, the RDMAIJ image has become a varianted image. Therefore, the information in section 2.12, "Step 10: Specify the Network Transport Protocol," of the *Oracle Rdb7 and Oracle CODASYL DBMS Guide to Hot Standby Databases* has become outdated in regards to setting up the RDMAIJSERVER object when using UCX as the network transport protocol. The UCX SET SERVICE command should now look similar to the following:

```
$ UCX SET SERVICE RDMAIJ -
      /PORT=<port_number> -
      /USER_NAME=RDMAIJ -
      /PROCESS_NAME=RDMAIJ -
      /FILE=SYS$SYSTEM:RDMAIJSERVER.com -
      /LIMIT=<limit>
```

And for Oracle Rdb multiversion, it should look similar to the following:

```
$ UCX SET SERVICE RDMAIJ70 -
      /PORT=<port_number> -
      /USER_NAME=RDMAIJ70 -
      /PROCESS_NAME=RDMAIJ70 -
      /FILE=SYS$SYSTEM:RDMAIJSERVER70.com -
      /LIMIT=<limit>
```

The installation procedure for Oracle Rdb creates a user named RDMAIJ(nn) and places a file called RDMAIJSERVER(nn).com in SYS$SYSTEM and the RMONSTART(nn).COM command procedure will try to enable a service called RDMAIJ(nn) if UCX is installed and running.

Changing the RDMAIJ server to a multivarianted image does not impact installations using DECNet since the correct DECNet object is created during the Rdb installation.

3.1.13 CREATE INDEX Supported for Hot Standby

On page 1-13 of the *Guide to Hot Standby Databases*, the add new index operation is incorrectly listed as an offline operation not supported by Hot Standby. The CREATE INDEX operation is now fully supported by Hot Standby, as long as the transaction does not span all available AIJ journals, including emergency AIJ journals.

3.1.14 Dynamic OR Optimization Formats

Bug 711643

In Table C-2 on Page C-7 of the *Oracle Rdb7 Guide to Database Performance and Tuning,* the dynamic OR optimization format is incorrectly documented as [l:h...]n. The correct formats for Oracle Rdb Release 7.0 and later are [(l:h)n] and [l:h,l2:h2].

4

Known Problems and Restrictions

This chapter describes problems, restrictions, and workarounds known to exist in Oracle Rdb7 Release 7.0.3.1.

4.0.1 AIJSERVER Privileges

For security reasons, the AIJSERVER account ("RDMAIJSERVER") is created with only NETMBX and TMPMBX privileges. These privileges are sufficient to start Hot Standby, in most cases.

However, for production Hot Standby systems, these privileges are not adequate to ensure continued replication in all environments and workload situations. Therefore, Oracle recommends that the DBA provide the following additional privileges for the AIJSERVER account:

- ALTPRI

  This privilege allows the AIJSERVER to adjust its own priority to ensure adequate quorum (CPU utilization) to prompt message processing.

- PSWAPM

  This privilege allows the AIJSERVER to enable and disable process swapping, also necessary to ensure prompt message processing.

- SETPRV

  This privilege allows the AIJSERVER to temporarily set any additional privileges it may need to access the standby database or its server processes.

- SYSPRV

  This privilege allows the AIJSERVER to access the standby database rootfile, if necessary.

- WORLD

  This privilege allows the AIJSERVER to more accurately detect standby database server process failure and handle network failure more reliably.

4.0.2 Lock Remastering and Hot Standby

When using the Hot Standby feature, Oracle recommends that the VMS distributed lock manager resource tree be mastered on the standby node where Hot Standby is started. This can be using any of the following methods:

- Disable dynamic lock remastering. This can be done dynamically by setting the SYSGEN parameter PE1 to the value 1.

  When using this option, be sure that Hot Standby is started on the node where the standby database is first opened.

- Increasing the LOCKDIRWT value for the LRS node higher than any other node in the same cluster. However, this is not a dynamic SYSGEN parameter, and a node re-boot is required.

Failure to prevent dynamic lock remastering may cause severe performance degradation for the standby database, which ultimately may be reflected by decreased master database transaction throughput.

4.0.3 RDB_SETUP Privilege Error

Rdb Web Agent V3.0 exposes a privilege problem with Rdb V7.0 and later. This will be fixed in the next Rdb7 release.

The RDB_SETUP function fails with %RDB-E-NO_PRIV, privilege denied by database facility.

It appears that the only workaround is to give users DBADM privilege. Oracle Corporation does not recommend giving users the DBADM privilege.

4.0.4 Dynamic Optimizer Problem with Zigzag Match

In some queries when utilizing zigzag match retrieval, a problem with the interaction of the zigzag match and the dynamic optimizer may cause the query to fail to deliver appropriate records.

The queries affected contain a join of two or more tables where the optimizer has chosen to utilize a zigzag match retrieval strategy and dynamic optimization (LEAF) retrieval of data for the inner leg of the match.

The following is an example of the type of strategy associated with the affected queries:

```
Match
  Outer loop      (zig-zag)
    Conjunct        Get     Retrieval by index of relation TABLE1
     Index name  TABLE1_INDEX_01 [1:1]
  Inner loop      (zig-zag)
    Leaf#02 Sorted TABLE2 Card=128800
      FgrNdx  TABLE2_INDEX_01 [0:0] Fan=41
      BgrNdx1 TABLE2_INDEX_02 [0:0] Fan=27
```

A problem in the delivery of data by the inner leg of the match from the dynamic optimizer data buffers prevented the appropriate match records in the outer from leg being found.

A workaround for the problem is to use the RDMS$DISABLE_ZIGZAG_MATCH logical name or the SQL SET FLAGS statement to disable the zigzag match:

```
VMS> define RDMS$DISABLE_ZIGZAG_MATCH 2
```

or

```
SQL> set flags 'nozigzag_match';
```

Alternatively, dynamic optimization may be disabled by using the RDMS$MAX_ STABILITY logical name or the SQL SET FLAGS statement:

```
VMS> define RDMS$MAX_STABILITY "TRUE"
```

or

```
SQL> set flags 'max_stability';
```

4.0.5 Starting Hot Standby on Restored Standby Database May Corrupt Database

If a standby database is modified outside of Hot Standby, then backed up and restored, Hot Standby will appear to start up successfully but will corrupt the standby database. A subsequent query of the database will return unpredictable results, possibly in a bugcheck in DIOFETCH$FETCH_ONE_LINE. When the standby database is restored from a backup of itself, the database is marked as unmodified. Therefore, Hot Standby cannot tell whether the database had been modified before the backup was taken.

WORKAROUND: None.

4.0.6 Restriction on Compound Statement Nesting Levels

The use of multiple nesting levels of compound statements such as CASE or IF-THEN-ELSE within multistatement procedures can result in excessive memory usage during the compile of the procedure. Virtual memory problems have been reported with 10 or 11 levels of nesting. The following example shows an outline of the type of nesting that can lead to this problem.

```
CREATE MODULE MY_MOD LANGUAGE SQL
PROCEDURE MY PROCEDURE
    ( PARAMETERS .....);

BEGIN
  DECLARE ....;

SET :VARS = 0;

SELECT .....;
GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
CASE :FLAG                      ! Case #1
       WHEN 100 THEN SET ...;
       WHEN -811 THEN SET ...;
       WHEN 0 THEN
         SET ...; SELECT ...;
         GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
         CASE :FLAG            ! Case #2
            WHEN 0 THEN SET ...;
            WHEN -811 THEN SET ...;
            WHEN 100 THEN
               UPDATE...; SET ...;
               GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
               IF :FLAG= 100 THEN SET ...;              ! #1
               ELSE
                IF :FLAG < 0 THEN SET...;               ! #2
               ELSE
                   DELETE  ...
                   GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
                   IF :FLAG= 100 THEN SET...;           ! #3
                      SET ...;
                   ELSE
                    IF :FLAG < 0 THEN SET...;           ! #4
                    ELSE
                     IF IN_CHAR_PARAM = 'S' THEN        ! #5
                      UPDATE ...
                      GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
                        IF :FLAG= 100 THEN SET ...;     ! #6
                        ELSE
                         IF :FLAG < 0 THEN SET...;      ! #7
                         END IF;                        ! #7
                       END IF;                         ! #6
                     END IF;                           ! #5
```

```
                    IF :FLAG = 0 THEN                  ! #5
                      UPDATE ...
                      GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
                      IF :FLAG= 100 THEN SET ...;      ! #6
                      ELSE
                         IF :FLAG < 0 THEN SET ...;    ! #7
                         ELSE
                            DELETE ...
                            GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE:
                            IF :FLAG= 100 THEN SET ...;        ! #8
                            ELSE
                              IF :FLAG < 0 THEN SET ...;       ! #9
                              ELSE
                                DELETE ...;
                                GET DIAGNOSTICS EXCEPTION 1 :FLAG = RETURNED_SQLCODE;
                                IF :FLAG= 100 THEN SET ...;    ! #10
                                   SET ...;
                                ELSE
                                   IF :FLAG < 0 THEN SET ...;  ! #11
                                   END IF; (11 end if's  for #11 - #1)
          ELSE SET ...;
          END CASE;                ! Case #2
      ELSE SET ...;
      END CASE;                    ! Case #1
END;
END MODULE;
```

Workaround: Reduce the complexity of the multistatement procedure. Use fewer levels of compound statement nesting by breaking the multistatement procedure into smaller procedures or by using the CALL statement to execute nested stored procedures.

4.0.7 Back Up All AIJ Journals Before Performing a Hot Standby Switchover Operation

Prior to performing a proper Hot Standby switchover operation from the old master database to the new master database (old standby database), be sure to back up ALL AIJ journals.

If you do not back up the AIJ journals on the old master database prior to switchover, they will be initialized by the Hot Standby startup operation, and you will not have a backup of those AIJ journals.

Failure to back up these journals may place your new master database at risk of not being able to be recovered, requiring another fail-over in the event of system failure.

4.0.8 Concurrent DDL and Read-Only Transaction on the Same Table Not Compatible

It is possible that a read-only transaction could generate a bugcheck at DIOBND$FETCH_AIP_ENT + 1C4 if there is an active, uncommitted transaction that is making metadata changes to the same table. Analysis shows that the snapshot transaction is picking up stale metadata information. Depending on what metatdata modifications are taking place, it is possible for metadata information to be removed from the system tables but still exist in the snapshot file. When the read-only transaction tries to use that information, it no longer exists and causes a bugcheck.

The following example shows the actions of the two transactions:

```
A:                              B:
attach
set transaction read write
                                attach
                                set transaction read only
drop index emp_last_name
                                select * from employees
                                ...bugcheck...
```

The only workaround is to avoid running the two transactions together.

4.0.9 Oracle Rdb and the SRM_CHECK Tool

The Alpha Architecture Reference Manual, Third Edition (AARM) describes strict rules for using interlocked memory instructions. The Compaq Alpha 21264 (EV6) processor and all future Alpha processors are more stringent than their predecessors in their requirement that these rules be followed. As a result, code that has worked in the past despite noncompliance may now fail when executed on systems featuring the new 21264 processor.

Oracle Rdb Release 7.0.3 supports the Compaq Alpha 21264 (EV6) processor. Oracle has performed extensive testing and analysis of the Rdb code to ensure that it is compliant with the rules for using interlocked memory instructions.

However, customers using the Compaq supplied SRM_CHECK tool may find that several of the Oracle Rdb images cause the tool to report potential alpha architecture violations. Although SRM_CHECK can normally identify a code section in an image by the section's attributes, it is possible for OpenVMS images to contain data sections with those same attributes. As a result, SRM_CHECK may scan data as if it were code, and occasionally, a block of data may look like a noncompliant code sequence. This is the case with the Oracle Rdb supplied images. There is no actual instruction stream violation.

However, customers must use the SRM_CHECK tool on their own application executable image files. It is possible that applications linked with very old version of Oracle Rdb (versions prior to Oracle Rdb Release 6.0-05) could have included illegal interlocked memory instruction sequences produced by very old versions of compilers. This code was included in the Oracle Rdb object library files for some very old versions of Oracle Rdb.

If errant instruction sequences are detected in the objects supplied by the Oracle Rdb object libraries, the correct action is to relink the application with a more-current version of Oracle Rdb.

Additional information about the Compaq Alpha 21264 (EV6) processor interlocked memory instructions issues is available at:

http://www.openvms.digital.com/openvms/21264_considerations.html

4.0.10 Oracle RMU Checksum_Verification Qualifier

The Oracle Rdb RMU BACKUP database backup command includes a Checksum_Verification qualifier.

Specifying Checksum_Verification requests that the RMU Backup command verify the checksum stored on each database page before it is backed up, thereby providing end-to-end error detection on the database I/O.

The Checksum_Verification qualifier uses additional CPU resources but can provide an extra measure of confidence in the quality of the data backed up. Use of the Checksum_Verification qualifier offers an additional level of data security and use of the Checksum_Verification qualifier permits Oracle RMU to detect the possibility that the data it is reading from these disks has only been partially updated.

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

Oracle Corporation recommends that you use the Checksum_Verification qualifier with all database backup operations because of the improved data integrity this qualifier provides.

Unfortunately, due to an oversight, for versions of Oracle Rdb prior to Version 8.0, the default for online backups is the Nochecksum_Verification qualifier. When you do not specify the Checksum_Verification qualifie on all of your RMU database backup commands.

4.0.11 Do Not Use HYPERSORT with RMU/OPTIMIZE/AFTER_JOURNAL (Alpha)

OpenVMS Alpha V7.1 introduced the high-performance Sort/Merge utility (also known as HYPERSORT). This utility takes advantage of the Alpha architecture to provide better performance for most sort and merge operations.

The high-performance Sort/Merge utility supports a subset of the SOR routines. Unfortunately, the high-performance Sort/Merge utility does not support several of the interfaces used by the RMU/OPTIMIZE/AFTER_JOURNAL command. In addition, the high-performance Sort/Merge utility reports no error or warning when being called with the unsupported options used by the RMU/OPTIMIZE/AFTER_JOURNAL command.

For this reason, the use of the high-performance Sort/Merge utility is not supported for the RMU/OPTIMIZE/AFTER_JOURNAL command. Do not define the logical name SORTSHR to reference HYPERSORT.EXE.

4.0.12 Restriction on Using /NOONLINE with Hot Standby

When a user process is performing a read-only transaction on a standby database, an attempt to start replication on the standby database with the /NOONLINE qualifier will fail with the following error, and the database will be closed clusterwide:

```
%RDMS-F-OPERCLOSE, database operator requested database shutdown
```

In a previous release, the following error was returned and the process doing the read-only transaction was not affected:

```
%RDMS-F-STBYDBINUSE, standby database cannot be exclusively accessed for
replication
```

As a workaround, if exclusive access is necessary to the standby database, terminate any user processes before starting replication with the /NOONLINE qualifier.

This restriction is due to another bug fix and will be lifted in a future release of Oracle Rdb.

4.0.13 SELECT Query May Bugcheck with PSII2SCANGETNEXTBBCDUPLICATE Error

Bug 683916

A bugcheck could occur when a ranked B-tree index is used in a query after a database has been upgraded to Release 7.0.1.3. This is a result of index corruption that was introduced in previous versions of Oracle Rdb7. This corruption has been fixed and indexes created using Release 7.0.1.3 will not be impacted.

As a workaround, delete the affected index and re-create it under Oracle Rdb7 Release 7.0.1.3 or later.

4.0.14 DBAPack for Windows 3.1 is Deprecated

Oracle Enterprise Manager DBAPack will no longer be supported for use on Windows 3.1.

4.0.15 Determining Mode for SQL Non-Stored Procedures

Bug 506464.

Although stored procedures allow parameters to be defined with the modes IN, OUT, and INOUT, there is no similar mechanism provided for SQL module language or SQL precompiled procedures. However, SQL still associates a mode with a parameter using the following rules:

Any parameter which is the target of an assignment is considered an OUT parameter. Assignments consist of the following:

- The parameter is assigned a value with the SET or GET DIAGNOSTICS statement. For example:

```
set :p1 = 0;
get diagnostics :p2 = TRANSACTION_ACTIVE;
```

- The parameter is assigned a value with the INTO clause of an INSERT, UPDATE, or SELECT statement. For example:

```
insert into T (col1, col2)
    values (...)
    returning dbkey into :p1;

update accounts
    set account_balance = account_balance + :amount
    where account_number = :p1
    returning account_balance
    into :current_balance;

select last_name
    into :p1
    from employees
    where employee_id = '00164';
```

- The parameter is passed on a CALL statement as an OUT or INOUT argument. For example:

```
begin
call GET_CURRENT_BALANCE (:p1);
end;
```

Any parameter that is the source for a query is considered an IN parameter. Query references include:

- The parameter appears in the SELECT list, WHERE or HAVING clauses of a SELECT, or DELETE statement. For example:

```
select :p1 || last_name, count(*)
    from T
    where last_name like 'Smith%'
    group by last_name
    having count(*) > :p2;

delete from T
    where posting_date < :p1;
```

- The parameter appears on the right side of the assignment in a SET statement or SET clause of an UPDATE statement. For example:

```
set :p1 = (select avg(salary)
            from T
            where department = :p2);
update T
    set col1 = :p1
    where ...;
```

- The parameter is used to provide a value to a column in an INSERT statement. For example:

```
insert into T (col1, col2)
    values (:p1, :p2);
```

- The parameter is referenced by an expression in a TRACE, CASE, IF/ELSEIF, WHILE statement, or by the DEFAULT clause of a variable declaration. For example:

```
begin
declare :v integer default :p1;
DO_LOOP:
while :p2 > :p1
loop
    if :p1 is null then
        leave DO_LOOP;
    end if;
    set :p2 = :p2 + 1;
    ...;
    trace 'Loop at ', :p2;
end loop;
end;
```

- The parameter is passed on a CALL statement as an INOUT or IN argument. For example:

```
begin
call SET_LINE_SPEED (:p1);
end;
```

SQL only copies values from the client (application parameters) to the procedure running in the database server if it is marked as either an IN or INOUT parameter. SQL only returns values from the server to the client application parameter variables if the parameter is an OUT or INOUT parameter.

If a parameter is considered an OUT only parameter, then it must be assigned a value within the procedure, otherwise the result returned to the application is considered undefined. This could occur if the parameter is used within a conditional statement such as CASE or IF/ELSEIF. In the following example, the value returned by :p2 would be undefined if :p1 were negative or zero:

```
begin
if :p1 > 0 then
    set :p2 = (select count(*)
                from T
                where col1 = :p1);
end if;
end;
```

It is the responsibility of the application programmer to ensure that the parameter is correctly assigned values within the procedure. A workaround is to either explicitly initialize the OUT parameter, or make it an INOUT parameter. For example:

```
begin
if :p1 > 0 then
    set :p2 = (select count(*)
                from T
                where col1 = :p1);
elseif :p2 is null then
    begin
    end;
end if;
end;
```

The empty statement will include a reference to the parameter to make it an IN parameter as well as an OUT parameter.

4.0.16 DROP TABLE CASCADE Results in %RDB-E-NO_META_UPDATE Error

An error could result when a DROP TABLE CASCADE statement is issued. This occurs when the following conditions apply:

- A table is created with an index defined on the table.
- A storage map is created with a placement via index.
- The storage map is a vertical record partition storage map with two or more STORE COLUMNS clauses.

The error message given is %RDB-E-NO_META_UPDATE, metadata update failed.

The following example shows a table, index, and storage map definition followed by a DROP TABLE CASCADE statement and the resulting error message:

```
SQL> CREATE TABLE VRP_TABLE ( ID INT, ID2 INT);
SQL> COMMIT;
SQL> CREATE UNIQUE INDEX VRP_IDX ON VRP_TABLE (ID)
SQL> STORE IN EMPIDS_LOW;
SQL> COMMIT;
SQL> CREATE STORAGE MAP VRP_MAP
cont> FOR VRP_TABLE
cont> PLACEMENT VIA INDEX VRP_IDX
cont> ENABLE COMPRESSION
cont> STORE COLUMNS (ID)
cont> IN EMPIDS_LOW
cont> STORE COLUMNS (ID2)
cont> IN EMPIDS_MID;
SQL> COMMIT;
SQL>
SQL> DROP TABLE VRP_TABLE CASCADE;
SQL> -- Index VRP_IDX is also being dropped.
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-E-WISH_LIST, feature not implemented yet
-RDMS-E-VRPINVALID, invalid operation for storage map "VRP_MAP"
```

The workaround to this problem is to first delete the storage map, and then delete the table using the CASCADE option. The following example shows the workaround. The SHOW statement indicates that the table, index, and storage map were deleted:

```
SQL> DROP STORAGE MAP VRP_MAP;
SQL> DROP TABLE VRP_TABLE CASCADE;
SQL> -- Index VRP_IDX is also being dropped.
SQL> COMMIT;
SQL> SHOW TABLE VRP_TABLE
No tables found
SQL> SHOW INDEX VRP_IDX
No indexes found
SQL> SHOW STORAGE MAP VRP_MAP
No Storage Maps Found
```

This problem will be corrected in a future version of Oracle Rdb.

4.0.17 Bugcheck Dump Files with Exceptions at COSI_CHF_SIGNAL

In certain situations, Oracle Rdb bugcheck dump files will indicate an exception at COSI_CHF_SIGNAL. This location is, however, not the address of the actual exception. The actual exception occurred at the previous call frame on the stack (the one listed as the next "Saved PC" after the exception).

For example, consider the following bugcheck file stack information:

```
$ SEARCH RDSBUGCHK.DMP "EXCEPTION","SAVED PC","-F-","-E-"

***** Exception at 00EFA828 : COSI_CHF_SIGNAL + 00000140
%COSI-F-BUGCHECK, internal consistency failure
Saved PC = 00C386F0 : PSIINDEX2JOINSCR + 00000318
Saved PC = 00C0BE6C : PSII2BALANCE + 0000105C
Saved PC = 00C0F4D4 : PSII2INSERTT + 000005CC
Saved PC = 00C10640 : PSII2INSERTTREE + 000001A0
    .
    .
    .
```

In this example, the exception actually occurred at PSIINDEX2JOINSCR offset 00000318. If you have a bugcheck dump with an exception at COSI_CHF_ SIGNAL, it is important to note the next "Saved PC" because it will be needed when working with Oracle Rdb Support Services.

4.0.18 Interruptions Possible when Using Multistatement or Stored Procedures

Long running multistatement or stored procedures can cause other users in the database to be interrupted by holding resources needed by those other users. Some resources obtained by the execution of a multistatement or stored procedure will not be released until the multistatement or stored procedure finishes. This problem can be encountered even if the statement contains COMMIT or ROLLBACK statements.

The following example demonstrates the problem. The first session enters an endless loop; the second session attempts to backup the database, but it is permanently interrupted:

Session 1

```
SQL> ATTACH 'FILE MF_PERSONNEL';
SQL> CREATE FUNCTION LIB$WAIT (IN REAL BY REFERENCE)
cont> RETURNS INT;
cont> EXTERNAL NAME LIB$WAIT
cont> LOCATION 'SYS$SHARE:LIBRTL.EXE'
cont> LANGUAGE GENERAL
cont> GENERAL PARAMETER STYLE
cont> VARIANT;
SQL> COMMIT;
SQL> EXIT;
```

```
$ SQL
SQL> ATTACH 'FILE MF_PERSONNEL';
SQL> BEGIN
cont> DECLARE :LAST_NAME LAST_NAME_DOM;
cont> DECLARE :WAIT_STATUS INTEGER;
cont> LOOP
cont> SELECT LAST_NAME INTO :LAST_NAME
cont>   FROM EMPLOYEES WHERE EMPLOYEE_ID = '00164';
cont> ROLLBACK;
cont> SET :WAIT_STATUS = LIB$WAIT (5.0);
cont> SET TRANSACTION READ ONLY;
cont> END LOOP;
cont> END;
```

Session 2

```
$ RMU/BACKUP/LOG/ONLINE MF_PERSONNEL MF_PERSONNEL
```

From a third session we can see that the backup process is waiting for a lock held in the first session:

```
$ RMU/SHOW LOCKS /MODE=BLOCKING MF_PERSONNEL
================================================================================
SHOW LOCKS/BLOCKING Information
================================================================================

--------------------------------------------------------------------------------
Resource: nowait signal

          ProcessID Process Name        Lock ID   System ID Requested Granted
          --------- --------------      --------- --------- --------- -------
Waiting:  20204383  RMU BACKUP.....     5600A476  00010001  CW        NL
Blocker:  2020437B  SQL.............    3B00A35C  00010001  PR        PR
$
```

There is no workaround for this restriction. When the multistatement or stored procedure finishes execution, the resources needed by other processes will be released.

4.0.19 Row Cache Not Allowed on Standby Database While Hot Standby Replication Is Active

The row cache feature may not be active on a Hot Standby database while replication is taking place. The Hot Standby feature will not start if row cache is active on the standby database.

This restriction exists because rows in the row cache are accessed using logical dbkeys. However, information transferred to the Hot Standby database from the after-image journal facility only contains physical dbkeys. Because there is no way to maintain rows in the cache using the Hot Standby processing, the row cache must be disabled on the standby database when the standby database is open and replication is active. The master database is not affected; the row cache feature and the Hot Standby feature may be used together on a master database.

The row cache feature should be identically configured on the master and standby databases in the event failover occurs, but the row cache feature must not be activated on the standby database until it becomes the master.

A new command qualifier, ROW_CACHE=DISABLED, has been added to the RMU/OPEN command to disable the row cache feature on the standby database. To open the Hot Standby database prior to starting replication, use the ROW_ CACHE=DISABLED qualifier on the RMU/OPEN command.

4.0.20 Hot Standby Replication Waits when Starting if Read-Only Transactions Running

Hot Standby replication will wait to start if there are read-only (snapshot) transactions running on the standby database. The log roll-forward server (LRS) will wait until the read-only transactions commit, and then replication will continue.

This is an existing restriction of the Hot Standby software. This release note is intended to complement the Hot Standby documentation.

4.0.21 Error when Using the SYS$LIBRARY:SQL_FUNCTIONS70.SQL Oracle Functions Script

If your programming environment is not set up correctly, you may encounter problems running the SYS$LIBRARY:SQL_FUNCTIONS70.SQL script used to set up the Oracle7 functions being supplied with Oracle Rdb.

The following example shows the error:

```
%RDB-E-EXTFUN_FAIL, external routine failed to compile or execute successfully
-RDMS-E-INVRTNUSE, routine RDB$ORACLE_SQLFUNC_INTRO can not be used, image
"SQL$FUNCTIONS" not activated
-RDMS-I-TEXT, Error activating image
DISK:[DIR]SQL$FUNCTIONS.;, File not found
```

To resolve this problem, use the @SYS$LIBRARY:RDB$SETVER to set up the appropriate logical names. This will be necessary for programs that use the functions as well.

In a standard environment, use the command shown in the following example:

```
$ @SYS$LIBRARY:RDB$SETVER S
```

In a multiversion environment, use the command shown in the following example:

```
$ @SYS$LIBRARY:RDB$SETVER 70
```

4.0.22 DEC C and Use of the /STANDARD Switch

Bug 394451

The SQL$PRE compiler examines the system to know which dialect of C to generate. That default can be overwritten by using the /CC=[DECC/VAXC] switch. The /STANDARD switch should not be used to choose the dialect of C.

Support for DEC C was added to the product with V6.0 and this note is meant to clarify that support, not to indicate a change. It is possible to use /STANDARD=RELAXED_ANSI89 or /STANDARD=VAXC correctly, but this is not recommended.

The following example shows both the right and wrong way to compile an Oracle Rdb SQL program. Assume a symbol SQL$PRE has been defined, and DEC C is the default C compiler on the system:

```
$ SQL$PRE/CC  ! This is correct.
$ SQL$PRE/CC=DECC ! This is correct.
$ SQL$PRE/CC=VAXC ! This is correct.

$ SQL$PRE/CC/STANDARD=VAXC   ! This is incorrect.
```

Notice that the /STANDARD switch has other options in addition to RELAXED_ANSI89 and VAX C. Those are also not supported.

4.0.23 Excessive Process Page Faults and Other Performance Considerations During Oracle Rdb Sorts

Excessive hard or soft page faulting can be a limiting factor of process performance. Sometimes this page faulting occurs during Oracle Rdb sort operations. This note describes how page faulting can occur and some ways to help control, or at least understand, it.

One factor contributing to Oracle Rdb process page faulting is sorting operations. Common causes of sorts include the SQL GROUP BY, ORDER BY, UNION, and DISTINCT clauses specified for query and index creation operations. Defining the logical name RDMS$DEBUG_FLAGS to "RS" can help determine when Oracle Rdb sort operations are occurring and to display the sort keys and statistics.

Oracle Rdb includes its own copy of the OpenVMS SORT32 code within the Oracle Rdb images and does not generally call the routines in the OpenVMS run-time library. A copy of the SORT32 code is used to provide stability between versions of Oracle Rdb and OpenVMS and because Oracle Rdb calls the sort routines from executive processor mode which is difficult to do using the SORT32 sharable image. Database import and RMU load operations call the OpenVMS sort run-time library.

At the beginning of a sort operation, the sort code allocates some memory for working space. The sort code uses this space for buffers, in-memory copies of the data, and sorting trees.

Sort code does not directly consider the process quotas or parameters when allocating memory. The effects of WSQUOTA and WSEXTENT are indirect. At the beginning of each sort operation, the sort code attempts to adjust the process' working set to the maximum possible size using the $ADJWSL system service specifying a requested working set limit of %X7FFFFFFF pages (the maximum possible). Sort code then uses a value of 75% of the returned working set for virtual memory scratch space. The scratch space is then initialized and the sort begins.

The initialization of the scratch space generally causes page faults to access the pages newly added to the working set. Pages that were in the working set already may be faulted out as new pages are faulted in. Once the sort operation completes, the pages that may have been faulted out of the working set are likely to be faulted back into the working set.

When a process' working set is limited by the working set quota (WSQUOTA) parameter and the working set extent (WSEXTENT) parameter is a much larger value, the first call to the sort routines can cause many page faults as the working set grows. Using a value of WSEXTENT that is closer to WSQUOTA can help reduce the impact of this case.

With some OpenVMS versions, AUTOGEN sets the SYSGEN parameter PQL_MWSEXTENT equal to the WSMAX parameter. This means that all processes on the system end up with WSEXTENT the same as WSMAX. Because WSMAX might be quite high, sorting might result in excessive page faulting. You may want to explicitly set PQL_MWSEXTENT to a lower value if this is the case on your system.

Sort work files are another factor to consider when tuning Oracle Rdb sort operations. When the operation cannot be done in available memory, sort code will use temporary disk files to hold the data as it is being sorted. The *Oracle Rdb Guide to Performance and Tuning* contains more detailed information about sort work files.

The logical name RDMS$BIND_SORT_WORKFILES specifies how many work files sort code is to use if work files are required. The default is 2, and the maximum number is 10. The work files can be individually controlled by the SORTWORKn logical names (where n is from 0 through 9). You can increase the efficiency of sort operations by assigning the location of the temporary sort work files to different disks. These assignments are made by using up to 10 logical names, SORTWORK0 through SORTWORK9.

Normally, sort code places work files in the user's SYS$SCRATCH directory. By default, SYS$SCRATCH is the same device and directory as the SYS$LOGIN location. Spreading the I/O load over many disks improves efficiency as well as performance by taking advantage of the system resources and helps prevent disk I/O bottlenecks. Specifying that a user's work files will reside on separate disks permits overlap of the sort read/write cycle. You may also encounter cases where insufficient space exists on the SYS$SCRATCH disk device, such as when Oracle Rdb builds indexes for a very large table. Using the SORTWORK0 through SORTWORK9 logical names can help you avoid this problem.

Note that sort code uses the work files for different sorted runs, and then merges the sorted runs into larger groups. If the source data is mostly sorted, then not every sort work file may need to be accessed. This is a possible source of confusion because even with 10 sort work files, it is possible to exceed the capacity of the first sort file, and the sort operation will fail never having accessed the remaining 9 sort work files.

Note that the logical names RDMS$BIND_WORK_VM and RDMS$BIND_WORK_FILE do not affect or control the operation of sort. These logical names are used to control other temporary space allocations within Oracle Rdb.

4.0.24 Performance Monitor Column Mislabeled

The File IO Overview statistics screen, in the Rdb Performance Monitor, contains a column labeled Pages Checked. The column should be labeled Pages Discarded to correctly reflect the statistic displayed.

4.0.25 Restriction Using Backup Files Created Later than Oracle Rdb7 Release 7.0.1

Bug 521583

Backup files created using Oracle Rdb7 releases later than 7.0.1 cannot be restored using Oracle Rdb7 Release 7.0.1. To fix a problem in a previous release, some internal backup file data structures were changed. These changes are not backward compatible with Oracle Rdb7 Release 7.0.1.

If you restore the database using such a backup file, then any attempt to access the restored database may result in unpredictable behavior, even though a verify operation may indicate no problems.

There is no workaround to this problem. For this reason, Oracle Corporation strongly recommends performing a full and complete backup both before and after the upgrade from Release 7.0.1 to later releases of Oracle Rdb7.

4.0.26 RMU Backup Operations and Tape Drive Types

When using more than one tape drive for an RMU backup operation, all the tape drives must be of the same type. For example, all the tape drives must be either TA90s or TZ87s or TK50s. Using different tape drive types (one TK50 and one TA90) for a single database backup operation may make database restoration difficult or impossible.

Oracle RMU attempts to prevent using different tape drive densities during a backup operation, but is not able to detect all invalid cases and expects that all tape drives for a backup are of the same type.

As long as all the tapes used during a backup operation can be read by the same type of tape drive during a restore operation, the backup is likely to be valid. This may be the case, for example, when using a TA90 and a TA90E.

Oracle recommends that, on a regular basis, you test your backup and recovery procedures and environment using a test system. You should restore the databases and then recover them using AIJs to simulate failure recovery of the production system.

Consult the *Oracle Rdb Guide to Database Maintenance*, the *Oracle Rdb Guide to Database Design and Definition*, and the *Oracle RMU Reference Manual* for additional information about Oracle Rdb backup and restore operations.

4.0.27 Use of Oracle Rdb from Shared Images

Bug 470946

If code in the image initialization routine of a shared image makes any calls into Oracle Rdb, through SQL or any other means, access violations or other unexpected behavior may occur if Oracle Rdb's images have not had a chance to do their own initialization.

To avoid this problem, applications must do one of the following:

- Do not make Oracle Rdb calls from the initialization routines of shared images.
- Link in such a way that the RDBSHR.EXE image initializes first. This can be done by placing the reference to RDBSHR.EXE and any other Oracle Rdb shared images last in the linker options file.

4.0.28 Interactive SQL Command Line Editor Rejects Eight-Bit Characters

Digital UNIX Systems

The interactive SQL command line editor on Digital UNIX can interfere with entering eight-bit characters from the command line. The command line editor assumes that a character with the eighth bit set will invoke an editing function. If the command line editor is enabled and a character with the eighth bit set is entered from the command line, the character will not be inserted on the command line. If the character has a corresponding editor function, the function will be invoked; otherwise, the character is considered invalid and rejected.

There are two ways to enter eight-bit characters from the SQL command line: either disable the command line editor or use the command line editor character quoting function to enter each eight-bit character. To disable the command line editor, set the configuration parameter RDB_NOLINEDIT in the configuration file. For example:

```
! Disable the interactive SQL command line editor.
RDB_NOLINEDIT   ON
```

To place quotation marks around a character using the command line editor, type Ctrl/V before each character to be place in quotation marks.

4.0.29 Restriction Added for CREATE STORAGE MAP on Table with Data

Oracle Rdb7 added support that allows a storage map to be added to an existing table which contains data. The restrictions listed for Oracle Rdb7 were:

- The storage map must be a simple map that references only the default storage area and represents the current (default) mapping for the table. The default storage area is either RDB$SYSTEM or the area name provided by the CREATE DATABASE...DEFAULT STORAGE AREA clause.
- The new map cannot change THRESHOLDS or COMPRESSION for the table, nor can it use the PLACEMENT VIA INDEX clause. It can only contain one area and cannot be vertically partitioned. This new map simply describes the mapping as it exists by default for the table.

This release of Rdb7 adds the additional restriction that the storage map may not include a WITH LIMIT clause for the storage area. The following example shows the reported error:

```
SQL> CREATE TABLE MAP_TEST1 (A INTEGER, B CHAR(10));
SQL> CREATE INDEX MAP_TEST1_INDEX ON MAP_TEST1 (A);
SQL> INSERT INTO MAP_TEST1 (A, B) VALUES (3, 'Third');
1 row inserted
SQL> CREATE STORAGE MAP MAP_TEST1_MAP FOR MAP_TEST1
cont> STORE USING (A) IN RDB$SYSTEM
cont>     WITH LIMIT OF (10);  -- can't use WITH LIMIT clause
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-RELNOTEMPTY, table "MAP_TEST1" has data in it
-RDMS-E-NOCMPLXMAP, can not use complex map for non-empty table
```

4.0.30 ALTER DOMAIN...DROP DEFAULT Reports DEFVALUNS Error

Bug 456867

If a domain has a DEFAULT of CURRENT_USER, SESSION_USER, or SYSTEM_USER and attempts to delete that default, it may fail unexpectedly. The following example shows the error:

```
SQL> ATTACH 'FILENAME PERSONNEL';
SQL> CREATE DOMAIN ADDRESS_DATA2_DOM CHAR(31)
cont>  DEFAULT CURRENT_USER;
SQL> COMMIT;
SQL> ALTER DOMAIN ADDRESS_DATA2_DOM
cont> DROP DEFAULT;
%SQL-F-DEFVALUNS, Default values are not supported for the data type of
ADDRESS_DATA2_DOM
```

To work around this problem you must first alter the domain to have a default of NULL, as shown, and then use DROP DEFAULT:

```
SQL> ALTER DOMAIN ADDRESS_DATA2_DOM
cont> SET DEFAULT NULL;
SQL> ALTER DOMAIN ADDRESS_DATA2_DOM
cont> DROP DEFAULT;
SQL> COMMIT;
```

This problem will be corrected in a future release of Oracle Rdb.

4.0.31 Oracle Rdb7 Workload Collection Can Stop Hot Standby Replication

If you are replicating your Oracle Rdb7 database using the Oracle Hot Standby option, you must not use the workload collection option. By default, workload collection is disabled. However, if you enabled workload collection, you must disable it on the master database prior to performing a backup operation on that master database if it will be used to create the standby database for replication purposes. If you do not disable workload collection, it could write workload information to the standby database and prevent replication operations from occurring.

The workaround included at the end of this section describes how to disable workload collection on the master database and allow the Hot Standby software to propagate the change to the standby database automatically during replication operations.

Background Information

By default, workload collection and cardinality collection are automatically disabled when Hot Standby replication operations are occurring on the standby database. However, if replication stops (even for a brief network failure), Oracle Rdb7 potentially can start a read/write transaction on the standby database to write workload collection information. Then, because the standby database is no longer synchronized transactionally with the master database, replication operations cannot restart.

Note

The Oracle Rdb7 optimizer can update workload collection information in the RDB$WORKLOAD system table even though the standby database is opened exclusively for read-only queries. A read/write transaction is started during the disconnection from the standby database to flush the workload and cardinality statistics to the system tables.

If the standby database is modified, you receive the following messages when you try to restart Hot Standby replication operations:

```
%RDMS-F-DBMODIFIED, database has been modified; AIJ roll-forward not possible
%RMU-F-FATALRDB, Fatal error while accessing Oracle Rdb.
```

Workaround

To work around this problem, perform the following:

- On the master database, disable workload collection using the SQL clause WORKLOAD COLLECTION IS DISABLED on the ALTER DATABASE statement. For example:

  ```
  SQL> ALTER DATABASE FILE mf_personnel
  cont> WORKLOAD COLLECTION IS DISABLED;
  ```

This change is propagated to the standby database automatically when you restore the standby database and restart replication operations. Note that, by default, the workload collection feature is disabled. You need to disable workload collection only if you previously enabled workload collection with the WORKLOAD COLLECTION IS ENABLED clause.

- On the standby database, include the Transaction_Mode qualifier on the RMU/Restore command when you restore the standby database. You should set this qualifier to read-only to prevent modifications to the standby database when replication operations are not active. The following example shows the Transaction_Mode qualifier used in a typical RMU/Restore command:

```
$ RMU/RESTORE /TRANSACTION_MODE=READ_ONLY
              /NOCDD
              /NOLOG
              /ROOT=DISK1:[DIR]standby_personnel.rdb
              /AIJ_OPT=aij_opt.dat
              DISK1:[DIR]standby_personnel.rbf
```

If, in the future, you fail over processing to the standby database (so that the standby database becomes the master database), you can re-enable updates to the "new" master database. For example, to re-enable updates, use the SQL statement ALTER DATABASE and include the SET TRANSACTION MODES (ALL) clause. The following example shows this statement used on the new master database:

```
SQL> ALTER DATABASE FILE mf_personnel
cont> SET TRANSACTION MODES (ALL);
```

4.0.32 RMU Convert Command and System Tables

When the RMU Convert command converts a database from a previous version to Oracle Rdb V7.0 or higher, it sets the RDB$CREATED and RDB$LAST_ ALTERED columns to the timestamp of the convert operation.

The RDB$xxx_CREATOR columns are set to the current user name (which is space filled) of the converter. Here xxx represents the object name, such as in RDB$TRIGGER_CREATOR.

The RMU Convert command also creates the new index on RDB$TRANSFER_ RELATIONS if the database is transfer enabled.

4.0.33 Converting Single-File Databases

Because of a substantial increase in the database root file information for Release 7.0, you should ensure that you have adequate disk space before you use the RMU Convert command with single-file databases and Release 7.0 or higher.

The size of the database root file of any given database will increase a minimum of 13 blocks and a maximum of 597 blocks. The actual increase depends mostly on the maximum number of users specified for the database.

4.0.34 Restriction when Adding Storage Areas with Users Attached to Database

If you try to interactively add a new storage area where the page size is less than the existing page size and the database has been manually opened or users are active, the add operation fails with the following error:

```
%RDB-F-SYS_REQUEST, error from system services request
-RDMS-F-FILACCERR, error opening database root DKA0:[RDB]TEST.RDB;1
-SYSTEM-W-ACCONFLICT, file access conflict
```

You can make this change only when no users are attached to the database and, if the database is set to OPEN IS MANUAL, the database is closed. Several internal Oracle Rdb data structures are based on the minimum page size, and these structures cannot be resized if users are attached to the database.

Furthermore, because this particular change is not recorded in the AIJ file, any recovery scenario will fail. Note also that if you use .aij files, you must backup the database and restart after-image journaling because this change invalidates the current AIJ recovery.

4.0.35 Restriction on Tape Usage for Digital UNIX V3.2

Digital UNIX Systems

You can experience a problem where you are unable to use multiple tapes with the Oracle RMU Backup command with Digital UNIX V3.2. Every attempt to recover fails. If this happens and device errors are logged in the system error log, it is possible that the operation succeeded, but the device open reference count is zeroed out. This means that any attempt to use the drive by the process holding the open file descriptor will fail with EINVAL status but another process will be able to open and use the drive even while the first process has it opened.

There is no workaround for this problem. This problem with the magtape driver will be corrected in a future release of Digital UNIX.

4.0.36 Support for Single-File Databases to be Dropped in a Future Release

Oracle Rdb currently supports both single-file and multifile databases on both OpenVMS and Digital UNIX. However, single-file databases will not be supported in a future release of Oracle Rdb. At that time, Oracle Rdb will provide the means to easily convert single-file databases to multifile databases.

Oracle recommends that users with single-file databases perform the following actions:

- Use the Oracle RMU commands, such as Backup and Restore, to make copies, back up, or move single-file databases. Do not use operating system commands to copy, back up, or move databases.
- Create new databases as multifile databases even though single-file databases are supported in Oracle Rdb release 6.1 and release 7.0.

4.0.37 DECdtm Log Stalls

Resource managers using the DECdtm services can sometimes suddenly stop being able to commit transactions. If Oracle Rdb7 is installed and transactions are being run, an RMU Show command on the affected database will show transactions as being "stalled, waiting to commit".

Refer to the DECdtm documentation and release notes for information on symptoms, fixes, and workarounds for this problem. One workaround, for OpenVMS V5.5-x, is provided here.

On the affected node while the log stall is in progress, type the following command from a privileged account:

```
$ MCR LMCP SET NOTIMEZONE
```

This should force the log to restart.

This stall occurs only when a particular bit in a pointer field becomes set. To see the value of the pointer field, enter the following command from a privileged account (where <nodename> is the SCS node name of the node in question).

```
$ MCR LMCP DUMP/ACTIVE/NOFORM SYSTEM$<nodename>
```

This command displays output similar to the following:

```
Dump of transaction log SYS$COMMON:[SYSEXE]SYSTEM$<nodename>.LM$JOURNAL;1
End of file block 4002 / Allocated 4002
Log Version 1.0
Transaction log UID:   29551FC0-CBB7-11CC-8001-AA000400B7A5
Penultimate Checkpoint: 000013FD4479 0079
Last Checkpoint:        000013FDFC84 0084

Total of 2 transactions active, 0 prepared and 2 committed.
```

The stall will occur when bit 31 of the checkpoint address becomes set, as this excerpt from the previous example shows:

```
        Last Checkpoint:        000013FDFC84 0084
                                    ^
                                    |
```

When the number indicated in the example becomes 8, the log will stall. Check this number and observe how quickly it grows. When it is at 7FFF, frequently use the following command:

```
$ MCR LMCP SHOW LOG /CURRENT
```

If this command shows a stall in progress, use the workaround to restart the log.

See your Compaq Computer Corporation representative for information about patches to DECdtm.

4.0.38 Cannot Run Distributed Transactions on Systems with DECnet/OSI and OpenVMS Alpha Version 6.1 or OpenVMS VAX Version 6.0

If you have DECnet/OSI installed on a system with OpenVMS Alpha Version 6.1 or OpenVMS VAX Version 6.0, you cannot run Oracle Rdb7 operations that require the two-phase commit protocol. The two-phase commit protocol guarantees that if one operation in a distributed transaction cannot be completed, none of the operations is completed.

If you have DECnet/OSI installed on a system running OpenVMS VAX Version 6.1 or higher or OpenVMS Alpha Version 6.2 or higher, you can run Oracle Rdb operations that require the two-phase commit protocol.

For more information about the two-phase commit protocol, see the *Oracle Rdb Guide to Distributed Transactions*.

4.0.39 Multiblock Page Writes May Require Restore Operation

If a node fails while a multiblock page is being written to disk, the page in the disk becomes inconsistent and is detected immediately during failover. (Failover is the recovery of an application by restarting it on another computer.) The problem is rare and occurs because only single-block I/O operations are guaranteed by OpenVMS to be written atomically. This problem has never been reported by any customer and was detected only during stress tests in our labs.

Correct the page by an area-level restore operation. Database integrity is not compromised, but the affected area will not be available until the restore operation completes.

A future release of Oracle Rdb will provide a solution that guarantees multiblock atomic write operations. Cluster failovers will automatically cause the recovery of multiblock pages, and no manual intervention will be required.

4.0.40 Oracle Rdb7 Network Link Failure Does Not Allow DISCONNECT to Clean Up Transactions

If a program attaches to a database on a remote node and it loses the connection before the COMMIT statement is issued, there is nothing you can do except exit the program and start again.

The problem occurs when a program is connected to a remote database and updates the database, but then just before it commits, the network fails. When the commit executes, SQL shows, as it normally should, that the program has lost the link. Assume that the user waits for a minute or two, then tries the transaction again. The problem is that when the start transaction is issued for the second time, it fails because old information still exists about the previous failed transaction. This occurs even if the user issues a DISCONNECT statement (in Release 4.1 and earlier, a FINISH statement), which also fails with an RDB-E-IO_ERROR error message.

4.0.41 Replication Option Copy Processes Do Not Process Database Pages Ahead of an Application

When a group of copy processes initiated by the Replication Option (formerly Data Distributor) begins running after an application has begun modifying the database, the copy processes will catch up to the application and will not be able to process database pages that are logically ahead of the application in the RDB$CHANGES system table. The copy processes all align waiting for the same database page and do not move on until the application has released it. The performance of each copy process degrades because it is being paced by the application.

When a copy process completes updates to its respective remote database, it updates the RDB$TRANSFERS system table and then tries to delete any RDB$CHANGES rows not needed by any transfers. During this process, the RDB$CHANGES table cannot be updated by any application process, holding up any database updates until the deletion process is complete. The application stalls while waiting for the RDB$CHANGES table. The resulting contention for RDB$CHANGES SPAM pages and data pages severely impacts performance throughput, requiring user intervention with normal processing.

This is a known restriction in Release 4.0 and higher. Oracle Rdb uses page locks as latches. These latches are held only for the duration of an action on the page and not to the end of transaction. The page locks also have blocking asynchronous system traps (ASTs) associated with them. Therefore, whenever a process requests a page lock, the process holding that page lock is sent a blocking AST (BLAST) by OpenVMS. The process that receives such a blocking AST queues the fact that the page lock should be released as soon as possible. However, the page lock cannot be released immediately.

Such work requests to release page locks are handled at verb commit time. An Oracle Rdb verb is an Oracle Rdb query that executes atomically, within a transaction. Therefore, verbs that require the scan of a large table, for example, can be quite long. An updating application does not release page locks until its verb has completed.

The reasons for holding on to the page locks until the end of the verb are fundamental to the database management system.

4.0.42 SQL Does Not Display Storage Map Definition After Cascading Delete of Storage Area

When you delete a storage area using the CASCADE keyword and that storage area is not the only area to which the storage map refers, the SHOW STORAGE MAP statement no longer shows the placement definition for that storage map.

The following example demonstrates this restriction:

```
SQL> SHOW STORAGE MAP DEGREES_MAP1
     DEGREES_MAP1
 For Table:             DEGREES1
 Compression is:        ENABLED
 Partitioning is:       NOT UPDATABLE
 Store clause:          STORE USING (EMPLOYEE_ID)
            IN DEG_AREA WITH LIMIT OF ('00250')
             OTHERWISE IN DEG_AREA2

SQL> DISCONNECT DEFAULT;
SQL> -- Drop the storage area, using the CASCADE keyword.
SQL> ALTER DATABASE FILENAME MF_PERSONNEL
cont> DROP STORAGE AREA DEG_AREA CASCADE;
SQL> --
SQL> -- Display the storage map definition.
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> SHOW STORAGE MAP DEGREES_MAP1
     DEGREES_MAP1
 For Table:             DEGREES1
 Compression is:        ENABLED
 Partitioning is:       NOT UPDATABLE

SQL>
```

The other storage area, DEG_AREA2, still exists, even though the SHOW STORAGE MAP statement does not display it.

A workaround is to use the RMU Extract command with the Items=Storage_Map qualifier to see the mapping.

4.0.43 ARITH_EXCEPT or Incorrect Results Using LIKE IGNORE CASE

When you use LIKE . . . IGNORE CASE, programs linked under Oracle Rdb Release 4.2 and Release 5.1, but run under higher versions of Oracle Rdb, may result in incorrect results or %RDB-E-ARITH_EXCEPT exceptions.

To work around the problem, avoid using IGNORE CASE with LIKE, or recompile and relink under a higher version (Release 6.0 or higher.)

4.0.44 Different Methods of Limiting Returned Rows from Queries

You can establish the query governor for rows returned from a query by using the SQL SET QUERY LIMIT statement, a logical name, or a configuration parameter. This note describes the differences between the mechanisms.

- If you define the RDMS$BIND_QG_REC_LIMIT logical name or RDB_BIND_QG_REC_LIMIT configuration parameter to a small value, the query will often fail with no rows returned. The following example demonstrates setting the limit to 10 rows and the resulting failure:

```
$ DEFINE RDMS$BIND_QG_REC_LIMIT 10
$ SQL$
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> SELECT EMPLOYEE_ID FROM EMPLOYEES;
%RDB-F-EXQUOTA, Oracle Rdb runtime quota exceeded
-RDMS-E-MAXRECLIM, query governor maximum limit of rows has been reached
```

Interactive SQL must load its metadata cache for the table before it can process the SELECT statement. In this example, interactive SQL loads its metadata cache to allow it to check that the column EMPLOYEE_ID really exists for the table. The queries on the Oracle Rdb system tables RDB$RELATIONS and RDB$RELATION_FIELDS exceed the limit of rows.

Oracle Rdb does not prepare the SELECT statement, let alone execute it. Raising the limit to a number less than 100 (the cardinality of EMPLOYEES) but more than the number of columns in EMPLOYEES (that is, the number of rows to read from the RDB$RELATION_FIELDS system table) is sufficient to read each column definition.

To see an indication of the queries executed against the system tables, define the RDMS$DEBUG_FLAGS logical name or the RDB_DEBUG_FLAGS configuration parameter as S or B.

- If you set the row limit using the SQL SET QUERY statement and run the same query, it returns the number of rows specified by the SQL SET QUERY statement before failing:

```
SQL> ATTACH 'FILENAME MF_PERSONNEL';
SQL> SET QUERY LIMIT ROWS 10;
SQL> SELECT EMPLOYEE_ID FROM EMPLOYEES;
EMPLOYEE_ID
00164
00165
   .
   .
   .
00173
%RDB-E-EXQUOTA, Oracle Rdb runtime quota exceeded
-RDMS-E-MAXRECLIM, query governor maximum limit of rows has been reached
```

The SET QUERY LIMIT specifies that only user queries be limited to 10 rows. Therefore, the queries used to load the metadata cache are not restricted in any way.

Like the SET QUERY LIMIT statement, the SQL precompiler and module processor command line qualifiers (QUERY_MAX_ROWS and SQLOPTIONS=QUERY_MAX_ROWS) only limit user queries.

Keep the differences in mind when limiting returned rows using the logical name RDMS$BIND_QG_REC_LIMIT or the configuration parameter RDB_BIND_QG_REC_LIMIT. They may limit more queries than are obvious. This is important when using 4GL tools, the SQL precompiler, the SQL module processor, and other interfaces that read the Oracle Rdb system tables as part of query processing.

4.0.45 Suggestions for Optimal Usage of the SHARED DATA DEFINITION Clause for Parallel Index Creation

The CREATE INDEX process involves the following steps:

1. Process the metadata.
2. Lock the index name.

Because new metadata (which includes the index name) is not written to disk until the end of the index process, Oracle Rdb must ensure index name uniqueness across the database during this time by taking a special lock on the provided index name.

3. Read the table for sorting by selected index columns and ordering.
4. Sort the key data.
5. Build the index (includes partitioning across storage areas).
6. Write new metadata to disk.

Step 6 is the point of conflict with other index definers because the system table and indexes are locked like any other updated table.

Multiple users can create indexes on the same table by using the RESERVING table_name FOR SHARED DATA DEFINITION clause of the SET TRANSACTION statement. For optimal usage of this capability, Oracle Rdb suggests the following guidelines:

- You should commit the transaction immediately after the CREATE INDEX statement so that locks on the table are released. This avoids lock conflicts with other index definers and improves overall concurrency.
- By assigning the location of the temporary sort work files SORTWORK0, SORTWORK1, . . . , SORTWORK9 to different disks for each parallel process that issues the SHARED DATA DEFINITION statement, you can increase the efficiency of sort operations. This minimizes any possible disk I/O bottlenecks and allows overlap of the SORT read/write cycle.
- If possible, enable global buffers and specify a buffer number large enough to hold a sufficient amount of table data. However, do not define global buffers larger than the available system physical memory. Global buffers allow sharing of database pages and thus result in disk I/O savings. That is, pages are read from disk by one of the processes and then shared by the other index definers for the same table, reducing the I/O load on the table.
- If global buffers are not used, ensure that enough local buffers exist to keep much of the index cached (use the RDM$BIND_BUFFERS logical name or RDB_BIND_BUFFERS configuration parameter or the NUMBER OF BUFFERS IS clause in SQL to change the number of buffers).
- To distribute the disk I/O load, place the storage areas for the indexes on separate disk drives. Note that using the same storage area for multiple indexes will result in contention during the index creation (Step 5) for SPAM pages.
- Consider placing the .ruj file for each parallel definer on its own disk or an infrequently used disk.
- Even though snapshot I/O should be minimal, consider disabling snapshots during parallel index creation.
- Refer to the *Oracle Rdb Guide to Performance and Tuning* to determine the appropriate working set values for each process to minimize excessive paging activity. In particular, avoid using working set parameters where the difference between WSQUOTA and WSEXTENT is large. The SORT utility uses the difference between these two values to allocate scratch virtual memory. A large difference (that is, the requested virtual memory grossly exceeds the available physical memory) may lead to excessive page faulting.

- The performance benefits of using SHARED DATA DEFINITION can best be observed when creating many indexes in parallel. The benefit is in the average elapsed time, not in CPU or I/O usage. For example, when two indexes are created in parallel using the SHARED DATA DEFINITION clause, the database must be attached twice, and the two attaches each use separate system resources.
- Using the SHARED DATA DEFINITION clause on a single-file database or for indexes defined in the RDB$SYSTEM storage area is not recommended.

The following table displays the elapsed time benefit when creating multiple indexes in parallel with the SHARED DATA DEFINITION clause. The table shows the elapsed time for 10 parallel process index creations (Index1, Index2, . . . Index10) and one process with 10 sequential index creations (All10). In this example, global buffers are enabled and the number of buffers is 500. The longest time for a parallel index creation is Index7 with an elapsed time of 00:02:34.64, compared to creating 10 indexes sequentially with an elapsed time of 00:03:26.66. The longest single parallel create index elapsed time is shorter than the elapsed time of creating all 10 of the indexes serially.

Index Create Job	Elapsed Time
Index1	00:02:22.50
Index2	00:01:57.94
Index3	00:02:06.27
Index4	00:01:34.53
Index5	00:01:51.96
Index6	00:01:27.57
Index7	00:02:34.64
Index8	00:01:40.56
Index9	00:01:34.43
Index10	00:01:47.44
All 10	00:03:26.66

4.0.46 Side Effect when Calling Stored Routines

When calling a stored routine, you must not use the same routine to calculate argument values by a stored function. For example, if the routine being called is also called by a stored function during the calculation of an argument value, passed arguments to the routine may be incorrect.

The following example shows a stored procedure P being called during the calculation of the arguments for another invocation of the stored procedure P:

```
SQL> CREATE MODULE M
cont>     LANG SQL
cont>
cont>     PROCEDURE P (IN :A INTEGER, IN :B INTEGER, OUT :C INTEGER);
cont>     BEGIN
cont>     SET :C = :A + :B;
cont>     END;
cont>
cont>     FUNCTION F () RETURNS INTEGER
cont>     COMMENT IS 'expect F to always return 2';
cont>     BEGIN
cont>     DECLARE :B INTEGER;
cont>     CALL P (1, 1, :B);
cont>     TRACE 'RETURNING ', :B;
cont>     RETURN :B;
cont>     END;
cont> END MODULE;
SQL>
SQL> SET FLAGS 'TRACE';
SQL> BEGIN
cont> DECLARE :CC INTEGER;
cont> CALL P (2, F(), :CC);
cont> TRACE 'Expected 4, got ', :CC;
cont> END;
~Xt: returning 2
~Xt: Expected 4, got 3
```

The result as shown above is incorrect. The routine argument values are written to the called routine's parameter area before complex expression values are calculated. These calculations may (as in the example) overwrite previously copied data.

The workaround is to assign the argument expression (in this example calling the stored function F) to a temporary variable and pass this variable as the input for the routine. The following example shows the workaround:

```
SQL> BEGIN
cont> DECLARE :BB, :CC INTEGER;
cont> SET :BB = F();
cont> CALL P (2, :BB, :CC);
cont> TRACE 'Expected 4, got ', :CC;
cont> END;
~Xt: returning 2
~Xt: Expected 4, got 4
```

This problem will be corrected in a future version of Oracle Rdb7.

4.0.47 Nested Correlated Subquery Outer References Incorrect

Outer references from aggregation subqueries contained within nested queries could receive incorrect values, causing the overall query to return incorrect results. The general symptom for an outer query that returned rows 1 to n was that the inner aggregation query would operate with the n^{th} - 1 row data (usually NULL for row 1) when it should have been using the n^{th} row data.

This problem has existed in various forms for all previous versions of Oracle Rdb7, but only appears in Release 6.1 and later when the inner of the nested queries contains an UPDATE statement.

The following example demonstrates the problem:

```
SQL> ATTACH 'FILENAME SHIPPING';
SQL> SELECT * FROM MANIFEST WHERE VOYAGE_NUM = 4904 OR
cont>                            VOYAGE_NUM = 4909;
  VOYAGE_NUM       EXP_NUM   MATERIAL              TONNAGE
        4904           311   CEDAR                  1200
        4904           311   FIR                     690
        4909           291   IRON ORE               3000
        4909           350   BAUXITE                1100
        4909           350   COPPER                 1200
        4909           355   MANGANESE               550
        4909           355   TIN                     500
7 rows selected

SQL> BEGIN
cont> FOR :A AS EACH ROW OF
cont>  SELECT * FROM VOYAGE V WHERE V.SHIP_NAME = 'SANDRA C.' OR
cont>                               V.SHIP_NAME = 'DAFFODIL' DO
cont>   FOR :B AS EACH ROW OF TABLE CURSOR MODCUR1 FOR
cont>    SELECT * FROM  MANIFEST M WHERE M.VOYAGE_NUM = :A.VOYAGE_NUM DO
cont>     UPDATE MANIFEST
cont>      SET TONNAGE = (SELECT (AVG (M1.EXP_NUM) *3) FROM MANIFEST M1
cont>                     WHERE M1.VOYAGE_NUM = :A.VOYAGE_NUM)
cont>      WHERE CURRENT OF MODCUR1;
cont>   END FOR;
cont> END FOR;
cont> END;
SQL> SELECT * FROM MANIFEST WHERE VOYAGE_NUM = 4904 OR
cont>                            VOYAGE_NUM = 4909;
  VOYAGE_NUM       EXP_NUM   MATERIAL              TONNAGE
        4904           311   CEDAR                  NULL
        4904           311   FIR                    NULL
        4909           291   IRON ORE                933
        4909           350   BAUXITE                 933
        4909           350   COPPER                  933
        4909           355   MANGANESE               933
        4909           355   TIN                     933
7 rows selected
```

The correct value for TONNAGE on both rows for VOYAGE_NUM 4904 (outer query row 1) is AVG (311+311)*3=933. However, Oracle Rdb7 calculates it as AVG (NULL+NULL)*3=NULL. In addition, the TONNAGE value for VOYAGE_NUM 4909 (outer query row 2) is actually the TONNAGE value for outer query row 1.

A workaround is to declare a variable of the same type as the outer reference data item, assign the outer reference data into the variable before the inner query that contains the correlated aggregation subquery, and reference the variable in the aggregation subquery. Keep in mind the restriction on the use of local variables in FOR cursor loops.

For example:

```
SQL> DECLARE :VN INTEGER;
SQL> BEGIN
cont> FOR :A AS EACH ROW OF
cont>  SELECT * FROM VOYAGE V WHERE V.SHIP_NAME = 'SANDRA C.' DO
cont>   SET :VN = :A.VOYAGE_NUM;
cont>   FOR :B AS EACH ROW OF TABLE CURSOR MODCUR1 FOR
cont>    SELECT * FROM MANIFEST M WHERE M.VOYAGE_NUM = :A.VOYAGE_NUM DO
cont>     UPDATE MANIFEST
cont>      SET TONNAGE = (SELECT (AVG (M1.EXP_NUM) *3) FROM MANIFEST M1
cont>                      WHERE M1.VOYAGE_NUM = :VN)
cont>      WHERE CURRENT OF MODCUR1;
cont>   END FOR;
cont> END FOR;
cont> END;
SQL> SELECT * FROM MANIFEST WHERE VOYAGE_NUM = 4904;
  VOYAGE_NUM       EXP_NUM   MATERIAL              TONNAGE
        4904           311   CEDAR                     933
        4904           311   FIR                       933
```

This problem will be corrected in a future release of Oracle Rdb.

4.0.48 Considerations when Using Holdable Cursors

If your applications use holdable cursors, be aware that after a COMMIT or ROLLBACK statement is executed, the result set selected by the cursor may not remain stable. That is, rows may be inserted, updated, and deleted by other users because no locks are held on the rows selected by the holdable cursor after a commit or rollback occurs. Moreover, depending on the access strategy, rows not yet fetched may change before Oracle Rdb actually fetches them.

As a result, you may see the following anomalies when using holdable cursors in a concurrent user environment:

- If the access strategy forces Oracle Rdb to take a data snapshot, the data read and cached may be inaccurate by the time the cursor fetches the data.

  For example, user 1 opens a cursor and commits the transaction. User 2 deletes rows read by user 1 (this is possible because the read locks are released). It is possible for user 1 to report data now deleted and committed.

- If the access strategy uses indexes that allow duplicates, updates to the duplicates chain may cause rows to be skipped, or even revisited.

  Oracle Rdb keeps track of the dbkey in the duplicate chain pointing to the data that was fetched. However, the duplicates chain could be revised by the time Oracle Rdb returns to using it.

Holdable cursors are a very powerful feature for read-only or predominantly read-only environments. However, in concurrent update environments, the instability of the cursor may not be acceptable. The stability of holdable cursors for update environments will be addressed in future versions of Oracle Rdb.

You can define the logical name RDMS$BIND_HOLD_CURSOR_SNAP or configuration parameter RDB_BIND_HOLD_CURSOR_SNAP to the value 1 to force all hold cursors to fetch the result set into a cached data area. (The cached data area appears as a "Temporary Relation" in the optimizer strategy displayed by the SET FLAGS STRATEGY statement or the RDMS$DEBUG_FLAGS S flag.) This logical name or configuration parameter helps to stabilize the cursor to some degree.

4.0.49 INCLUDE SQLDA2 Statement Is Not Supported for SQL Precompiler for PL/I in Oracle Rdb Release 5.0 or Higher

The SQL statement INCLUDE SQLDA2 is not supported for use with the PL/I precompiler in Oracle Rdb Release 5.0 or higher.

There is no workaround. This problem will be fixed in a future version of Oracle Rdb.

4.0.50 SQL Pascal Precompiler Processes ARRAY OF RECORD Declarations Incorrectly

The Pascal precompiler for SQL gives an incorrect %SQL-I-UNMATEND error when it parses a declaration of an array of records. The precompiler does not associate the END statement with the record definition, and the resulting confusion in host variable scoping causes a fatal error.

A workaround for the problem is to declare the record as a type and then define your array of that type. For example:

```
    main.spa:

        program main (input,output);

        type
        exec sql include 'bad_def.pin';    !gives error
        exec sql include 'good_def.pin';   !ok
        var
            a : char;

        begin
        end.

-----------------------------------------------------------------
    bad_def.pin

    x_record = record
    y  : char;
    variable_a:  array [1..50] of record
                a_fld1 : char;
                b_fld2  : record;
                             t : record
                                      v : integer;
                             end;
                end;
        end;
    end;
 ----------------------------------------------------------------

    good_def.pin

good_rec = record
        a_fld1 : char;
        b_fld2 : record
                t : record
                        v: integer;
                end;
        end;
end;

    x_record = record
        y  : char
        variable_a : array [1..50] of good_rec;
    end;
```

4.0.51 RMU Parallel Backup Command Not Supported for Use with SLS

The RMU Parallel Backup command is not supported for use with the Storage Library System (SLS) for OpenVMS.

4.0.52 Oracle RMU Commands Pause During Tape Rewind

Digital UNIX Systems

For Oracle Rdb Release 6.1 or higher on Digital UNIX, the Oracle RMU Backup and Restore commands pause under certain conditions.

If multiple tape drives are used for RMU Backup or RMU Restore commands and a tape needs to rewind, the Oracle RMU command pauses until the rewind is complete. This is different from behavior on OpenVMS systems where the command continues to write to tape drives that are not rewinding.

There is no workaround for this problem.

4.0.53 TA90 and TA92 Tape Drives Are Not Supported on Digital UNIX

Digital UNIX Systems

When rewinding or unloading tapes using either TA90 and TA92 drives, Digital UNIX intermittently returns an EIO error causing the Oracle RMU operation to abort. This problem occurs most often when Oracle RMU accesses multiple tape drives in parallel. However, the problem occurs even with single-tape drive access.

As a result of this problem, Oracle Rdb on Digital UNIX supports neither TA90 nor TA92 tape drives.

4.1 Oracle CDD/Repository Restrictions

This section describes known problems and restrictions in Oracle CDD/Repository Release 7.0 and earlier.

4.1.1 Oracle CDD/Repository Compatibility with Oracle Rdb Features

Some Oracle Rdb features are not fully supported by all versions of Oracle CDD/Repository. Table 4–1 shows which versions of Oracle CDD/Repository support Oracle Rdb features and the extent of support.

In Table 4–1, repository support for Oracle Rdb7 features can vary as follows:

- Explicit support—The repository recognizes and integrates the feature, and you can use the repository to manipulate the item.
- Implicit support—The repository recognizes and integrates the feature, but you cannot use any repository interface to manipulate the item.
- Pass-through support—The repository does not recognize or integrate the feature, but allows the Oracle Rdb7 operation to complete without aborting or overwriting metadata. With pass-through support, a CDD-I-MBLRSYNINFO informational message may be returned.

Table 4–1 Oracle CDD/Repository Compatibility for Oracle Rdb Features

Oracle Rdb Feature	Minimum Release of Oracle Rdb	Minimum Release of Oracle CDD/Repository	Support
CASE, NULLIF, and COALESCE expressions	6.0	6.1	Implicit
CAST function	4.1	7.0	Explicit
Character data types to support character sets	4.2	6.1	Implicit
Collating sequences	3.1	6.1	Explicit
Constraints (PRIMARY KEY, UNIQUE, NOT NULL, CHECK, FOREIGN KEY)	3.1	5.2	Explicit
CURRENT_DATE, CURRENT_ TIME, and CURRENT_ TIMESTAMP functions	4.1	7.0	Explicit
CURRENT_USER, SESSION_ USER, SYSTEM_USER functions	6.0	7.0	Explict
Date arithmetic	4.1	6.1	Pass-through
DATE ANSI, TIME, TIMESTAMP, and INTERVAL data types	4.1	6.1	Explicit
Delimited identifiers	4.2	6.1[1]	Explicit
External functions	6.0	6.1	Pass-through
External procedures	7.0	6.1	Pass-through
EXTRACT, CHAR_LENGTH, and OCTET_LENGTH functions	4.1	6.1	Explicit
GRANT/REVOKE privileges	4.0	5.0 accepts but does not store information	Pass-through
Indexes	1.0	5.2	Explicit
INTEGRATE DOMAIN	6.1	6.1	Explicit
INTEGRATE TABLE	6.1	6.1	Explicit
Logical area thresholds for storage maps and indexes	4.1	5.2	Pass-through
Multinational character set	3.1	4.0	Explicit
Multiversion environment (multiple Rdb versions)	4.1	5.1	Explicit
NULL keyword	2.2	7.0	Explicit
Oracle7 compatibility functions, such as CONCAT, CONVERT, DECODE, and SYSDATE	7.0	7.0	Explicit
Outer joins, derived tables	6.0	7.0	Pass-through
Query outlines	6.0	6.1	Pass-through

[1]The repository does not preserve the distinction between uppercase and lowercase identifiers. If you use delimited identifiers with Oracle Rdb, the repository ensures that the record definition does not include objects with names that are duplicates except for case.

(continued on next page)

Table 4–1 (Cont.) Oracle CDD/Repository Compatibility for Oracle Rdb Features

Oracle Rdb Feature	Minimum Release of Oracle Rdb	Minimum Release of Oracle CDD/Repository	Support
Storage map definitions correctly restored	3.0	5.1	Explicit
Stored functions	7.0	6.1	Pass-through
Stored procedures	6.0	6.1	Pass-through
SUBSTRING function	4.0	7.0 supports all features 5.0 supports all but 4.2 MIA features [2]	Explicit
Temporary tables	7.0	6.1	Pass-through
Triggers	3.1	5.2	Pass-through
TRUNCATE TABLE	7.0	6.1	Pass-through
TRIM and POSITION functions	6.1	7.0	Explicit
UPPER, LOWER, TRANSLATE functions	4.2	7.0	Explicit
USER function	2.2	7.0	Explict

[2]Multivendor Integration Architecture (MIA) features include the CHAR_LENGTH clause and the TRANSLATE function.

4.1.2 Multischema Databases and CDD/Repository

You cannot use multischema databases with CDD/Repository and Oracle Rdb release 7.0 and earlier. This problem will be corrected in a future release of Oracle Rdb.

4.1.3 Interaction of Oracle CDD/Repository Release 5.1 and Oracle RMU Privileges Access Control Lists

Oracle Rdb provides special Oracle RMU privileges that use the unused portion of the OpenVMS access control list (ACL) to manage access to Oracle RMU operations.

You can use the RMU Set Privilege and RMU Show Privilege commands to set and show the Oracle RMU privileges. The DCL SHOW ACL and DIRECTORY/ACL commands also show the added access control information; however, these tools cannot translate the names defined by Oracle Rdb.

Note

The RMU Convert command propagates the database internal ACL to the root file for access control entries (ACEs) that possess the SECURITY and DBADM (ADMINISTRATOR) privileges.

Oracle CDD/Repository protects its repository (dictionary) by placing the CDD$SYSTEM rights identifier on each file created within the anchor directory. CDD$SYSTEM is a special, reserved rights identifier created by Oracle CDD/Repository.

When Oracle CDD/Repository executes the DEFINE REPOSITORY command, it adds (or augments) an OpenVMS default ACL to the anchor directory. Typically, this ACL allows access to the repository files for CDD$SYSTEM and denies access to everyone else. All files created in the anchor directory inherit this default ACL, including the repository database.

Unfortunately, there is an interaction between the default ACL placed on the repository database by Oracle CDD/Repository and the Oracle RMU privileges ACL processing.

Within the ACL on the repository database, the default access control entries (ACEs) that were inherited from the anchor directory will precede the ACEs added by RMU Restore. As a result, the CDD$SYSTEM identifier will not have any Oracle RMU privileges granted to it. Without these privileges, if the user does not have the OpenVMS SYSPRV privilege enabled, Oracle RMU operations, such as Convert and Restore, will not be allowed on the repository database.

The following problems may be observed by users who do not have the SYSPRV privilege enabled:

- While executing a CDO DEFINE REPOSITORY or DEFINE DICTIONARY command:
  - If the CDD$TEMPLATEDB backup (.rbf) file was created by a previous version of Oracle Rdb7, the automatic RMU Convert operation that will be carried out on the .rbf file will fail because SYSPRV privilege is required.
  - If the CDD$TEMPLATEDB backup (.rbf) file was created by the current version of Oracle Rdb7, the restore of the repository database will fail because the default ACEs that already existed on the repository file that was backed up will take precedence, preventing RMU$CONVERT and RMU$RESTORE privileges from being granted to CDD$SYSTEM or the user.
  - If no CDD$TEMPLATEDB is available, the repository database will be created without a template, inheriting the default ACL from the parent directory. The ACE containing all the required Oracle RMU privileges will be added to the end of the ACL; however, the preexisting default ACEs will prevent any Oracle RMU privilege from being granted.
- You must use the RMU Convert command to upgrade the database disk format to Oracle Rdb7 after installing Release 7.0. This operation requires the SYSPRV privilege.

  During the conversion, RMU Convert adds the ACE containing the Oracle RMU privileges at the end of the ACL. Because the repository database already has the default Oracle CDD/Repository ACL associated with it, the Oracle CDD/Repository ACL will take precedence, preventing the granting of the Oracle RMU privileges.
- During a CDO MOVE REPOSITORY command, the Oracle RMU privilege checking may prevent the move, as the RMU$COPY privilege has not been granted on the repository database.
- When you execute the CDD template builder CDD_BUILD_TEMPLATE, the step involving RMU Backup privilege has not been granted.

Oracle CDD/Repository Releases 5.2 and higher correct this problem. A version of the Oracle CDD/Repository software that corrects this problem and allows new repositories to be created using Oracle Rdb7 is provided on the Oracle Rdb7 kit for use on OpenVMS VAX systems. See Section 4.1.3.1 for details.

4.1.3.1 Installing the Corrected CDDSHR Images

OpenVMS VAX Systems

Note

The following procedure must be carried out if you have installed or plan to install Oracle Rdb7 and have already installed CDD/Repository Release 5.1 software on your system.

Due to the enhanced security checking associated with Oracle RMU commands in Oracle Rdb on OpenVMS VAX, existing CDDSHR images for CDD/Repository Release 5.1 must be upgraded to ensure that the correct Oracle RMU privileges are applied to newly created or copied repository databases.

Included in the Oracle Rdb7 for OpenVMS VAX distribution kit is a CDD upgraded image kit, called CDDRDB042, that must be installed after you have installed the Oracle Rdb7 for OpenVMS VAX kit.

This upgrade kit should be installed by using VMSINSTAL. It automatically checks which version of CDDSHR you have installed and replaces the existing CDDSHR.EXE with the corrected image file. The existing CDDSHR.EXE will be renamed SYS$LIBRARY:OLD_CDDSHR.EXE.

The upgrade installation will also place a new CDD_BUILD_TEMPLATE.COM procedure in SYS$LIBRARY for use with CDD/Repository V5.1.

Note

If you upgrade your repository to CDD/Repository V5.1 after you install Oracle Rdb7 V7.0, you must install the corrected CDDSHR image again to ensure that the correct CDDSHR images have been made available.

The CDD/Repository upgrade kit determines which version of CDD/Repository is installed and replaces the existing CDDSHR.EXE with the appropriate version of the corrected image.

4.1.3.2 CDD Conversion Procedure

OpenVMS VAX Systems

Oracle Rdb7 provides RDB$CONVERT_CDD$DATABASE.COM, a command procedure that both corrects the anchor directory ACL and performs the RMU Convert operation. The command procedure is located in SYS$LIBRARY.

Note

You must have SYSPRV enabled before you execute the procedure RDB$CONVERT_CDD$DATABASE.COM because the procedure performs an RMU Convert operation.

Use the procedure RDB$CONVERT_CDD$DATABASE.COM to process the anchor directory and update the ACLs for both the directory and, if available, the repository database.

This procedure accepts one parameter: the name of the anchor directory that contains, or will contain, the repository files. For example:

```
$ @SYS$LIBRARY:DECRDB$CONVERT_CDD$DATABASE [PROJECT.CDD_REP]
```

If many repositories exist on a system, you may want to create a DCL command procedure to locate them, set the Oracle RMU privileges ACL, and convert the databases. Use DCL commands similar to the following:

```
$ LOOP:
$       REP_SPEC = F$SEARCH("[000000...]CDD$DATABASE.RDB")
$       IF REP_SPEC .NES. ""
$       THEN
$           @SYS$LIBRARY:DECRDB$CONVERT_CDD$DATABASE    -
                'F$PARSE(REP_SPEC,,,"DIRECTORY")'
$           GOTO LOOP
$       ENDIF
```

5
Enhancements

This chapter describes the enhancements that are introduced in Oracle Rdb7 Release 7.0.3.1.

5.1 Enhancements Provided in Oracle Rdb7 Release 7.0.3.1

5.1.1 Per-Process Monitoring for SHOW STATS

The purpose of the Per-Process Monitoring facility is to provide a powerful drill-down capability to allow the DBA to analyze process-specific information for a single process, class of processes, or all attached database processes. The facility also provides several screens that display a side-by-side comparison of individual process statistic information, such as I/O, transaction, and record statistics. The Per-Process Monitoring facility presents real-time information and, consequently, does not write its information to the binary output file. Therefore, the Per-Process Monitoring facility is not available during the replay of a binary input file.

The Per-Process Monitoring facility is not available if clusterwide statistic collection is active. Conversely, the clusterwide statistic collection facility is not available when the Per-Process Monitoring facility is active.

For a complete description of the various methods by which the Per-Process Monitoring facility can be activated, see the white paper, *RMU SHOW STATISTIC Utility Per-Process Monitoring Facility* on MetaLink.

5.1.2 New DOMAINS Option for RMU/EXTRACT

In Oracle Rdb7 Release 7.0.3.1, RMU Extract adds more control over the way domain definitions are handled. A new /OPTION=NODOMAINS can be used to eliminate the domain name from within metadata objects. When used the domain name is replaced by the underlying data type. This option is designed for use with tools that do not understand the SQL92 SQL language domain feature.

- Affect on /LANGUAGE=SQL output.

  The default is /OPTION=DOMAINS.

  A SQL script generated when using /OPTION=NODOMAINS no longer includes the domain name in the CREATE TABLE column definition, CREATE FUNCTION or CREATE PROCEDURE parameter definitions, and any other value expression which uses the CAST function to convert expression to a domain data type (such as a CREATE VIEW or CREATE TRIGGER).

  The output for CREATE MODULE is not affected by /OPTION=NODOMAINS because it is based on the original source SQL which is not edited by RMU Extract.

- Affect on the /LANGUAGE=ANSI_SQL output.

The default is /OPTION=NODOMAINS when /OPTION=NORMAL is specified or is the default.

In prior versions, ANSI_SQL would attempt to extract the definition using syntax from ANSI SQL89 if /OPTION=NORMAL was specified or was the default. This SQL language standard does not support domain definitions but RMU Extract would still output the definitions if /ITEM=ALL was specified. In this release of Oracle Rdb RMU Extract no longer generates a list of CREATE DOMAIN statements if /ITEM=DOMAINS or /ITEM=ALL is used.

To get the behavior of previous versions use the /OPTION=DOMAINS qualifier to have domains generated. For example:

```
$ RMU/EXTRACT/LANGUAGE=ANSI_SQL/OPTION=DOMAINS databasename
```

Use the /OPTION=FULL qualifier to have syntax generated for SQL92 to include the use of domains.

5.1.3 New NO REORGANIZE clause for ALTER STORAGE MAP

In Oracle Rdb V7.0.2, support was added to allow an existing storage map to be converted to a strictly partitioned storage map using the ALTER STORAGE MAP...PARTITIONING IS NOT UPDATABLE clause of the ALTER STORAGE MAP statement.

This statement implicitly performs a REORGANIZE on the base table, moving rows within the map if necessary, scanning the storage areas to make sure all the stored data conforms to the storage map definition. This allows the the Oracle Rdb optimizer to use this type of table efficiently when a sequential scan uses a subset of the storage areas.

In many cases the database administrator knows that a large table is already strictly partitioned but it is prohibitive to reorganize the table (the amount of I/O alone might last several hours). Therefore, in this release of Oracle Rdb, we have allowed the database administrator to bypass the automatic REORGANIZE performed by the ALTER STORAGE MAP ... PARTITIONING IS NOT UPDATABLE statement using a new NO REORGANIZE clause.

There is an associated risk because Oracle Rdb has not validated the table partitioning. It is possible that rows will be missed by sequential scans. The database administrator must take this risk into account when using this new clause. Oracle Corporation suggests that an ALTER STORAGE MAP...REORGANIZE be carried out as soon as practical.

When the NO REORGANIZE clause is used, Oracle Rdb records this information in the Oracle Rdb system tables. The SHOW STORAGE MAP statement will display informational text.

The following example shows an ALTER STORAGE MAP statement which disabled the area scan for PARTITIONING IS NOT UPDATABLE and the informational text from the SHOW STORAGE MAP statement:

```
SQL> set flags 'stomap_stats';
SQL> alter storage map EMPLOYEES_MAP
cont>     partitioning is not updatable
cont>     no reorganize
cont>     store
cont>         using (EMPLOYEE_ID)
cont>             in EMPIDS_LOW
cont>                 with limit of ('00200')
cont>             in EMPIDS_MID
cont>                 with limit of ('00400')
cont>             otherwise in EMPIDS_OVER;
~As: starting map restructure...
~As: REORGANIZE needed to preserve strict partitioning
~As: NO REORGANIZE was used to override scan
~As: reads: async 0 synch 21, writes: async 7 synch 3
SQL>
SQL> show storage map EMPLOYEES_MAP
     EMPLOYEES_MAP
 For Table:             EMPLOYEES
 Placement Via Index:   EMPLOYEES_HASH
 Partitioning is:       NOT UPDATABLE
 Strict partitioning was not validated for this table
 Comment:        employees partitioned by "00200" "00400"
 Store clause:          STORE
             using (EMPLOYEE_ID)
                 in EMPIDS_LOW
                     with limit of ('00200')
                 in EMPIDS_MID
                     with limit of ('00400')
                 otherwise in EMPIDS_OVER
 Compression is:        ENABLED
SQL>
```

Entering a subsequent ALTER STORAGE MAP...REORGANIZE will validate the partitioning:

```
SQL> alter storage map EMPLOYEES_MAP
cont>     partitioning is not updatable
cont>     reorganize;
~As: starting map restructure...
~As: starting REORGANIZE...
~As: reorganize AREAS...
~As: processing rows from area 69
~As: processing rows from area 70
~As: processing rows from area 71
~As: reads: async 408 synch 22, writes: async 3 synch 0
SQL>
```

Usage Notes

- The NO REORGANIZE clause is ignored unless used with PARTITIONING IS NOT UPDATABLE. This is because either no automatic reorganize is required, or a full rebuild of the table is needed to implement the new map structure.
- REORGANIZE and NO REORGANIZE may not appear in the same ALTER STORAGE MAP command.

```
SQL> alter storage map EMPLOYEES_MAP
cont>     partitioning is not updatable
cont>     no reorganize
cont>     reorganize areas
cont>     store
cont>         using (EMPLOYEE_ID)
cont>             in EMPIDS_LOW
cont>                 with limit of ('00200')
cont>             in EMPIDS_MID
cont>                 with limit of ('00400')
cont>             otherwise in EMPIDS_OVER;
%SQL-F-MULTSPECATR, Multiple specified attribute. "REORGANIZE" was specified
more than once
```

- The SET FLAGS option STOMAP_STATS will output an indication that NO REORGANIZE was used.
- The SHOW STORAGE MAP command will output an indication that NO REORGANIZE was used. For example:

```
SQL> show storage map EMPLOYEES_MAP
     EMPLOYEES_MAP
 For Table:             EMPLOYEES
 Placement Via Index:   EMPLOYEES_HASH
 Partitioning is:       NOT UPDATABLE
 Strict partitioning was not validated for this table ...
```

5.1.4 New Options for the GET DIAGNOSTICS Statement

For Oracle Rdb7 Release 7.0.3.1, two new options have been added to the GET DIAGNOSTICS statement:

- TRANSACTION_TIMESTAMP

  This option returns the date and time that the last transaction was started. If a transaction is not active, then this may be a prior transaction. The database server will start transactions when performing database operations and, therefore, this timestamp may reflect the time of an internal transaction.

  If the default date format is SQL92, this option returns a value with the data type TIMESTAMP(2); otherwise, it returns a DATE (VMS) data type. The default date format can be changed using either the SET DIALECT or SET DEFAULT DATE FORMAT statements, or one of the associated module attributes.

- TRANSACTION_SEQUENCE

  This is the transaction sequence number (TSN) assigned to the most recently started transaction. The TSN is a unique indicator of database transaction activity; however, note that the TSN may be reused in some cases. The TSN for a read-only transaction reflects the transaction state which is visible to the transaction, and therefore it could have been previously assigned to a read/write transaction. If a read/write transaction performs no database I/O, or was rolled back, then that TSN may be reused by a subsequent read/write transaction.

  This option returns a BIGINT data type.

The following example uses both these new options:

```
SQL> set transaction read write;
SQL> show transaction
Transaction information:
    Statement constraint evaluation is off
On the default alias
Transaction characteristics:
        Read Write
Transaction information returned by base system:
a read-write transaction is in progress
  - updates have not been performed
  - transaction sequence number (TSN) is 0:256
  - snapshot space for TSNs less than 0:256 can be reclaimed
  - recovery unit journal filename is USER2:[RDM$RUJ]SCRATCH$00018679B3AD.RUJ;1
  - session ID number is 8
SQL>
SQL> declare :x date vms;
SQL>
SQL> begin get diagnostics :x = transaction_timestamp; end;
SQL> print :x;
 X
 27-MAY-1999 22:39:17.02
SQL>
SQL> declare :y bigint;
SQL>
SQL> begin get diagnostics :y = transaction_sequence; end;
SQL> print :y;
                    Y
                  256
SQL>
SQL> select current_timestamp from rdb$database;

 27-MAY-1999 22:39:18.20
1 row selected
SQL>
SQL> commit;
```

5.1.5 RMU/SHOW Statistic OPCOM Message Tracking

The RMU/SHOW Statistic utility has been enhanced to provide tracking of database-related OPCOM messages. OPCOM messages are tracked using two independent methods:

1. New "OPCOM Messages" Screen. Located in the "Process Information" submenu, the "OPCOM Messages" screen identifies the last broadcast OPCOM message for each active process. If a single process is attached to the database multiple times, the OPCOM messages are displayed for the attach that issued the broadcast.
2. New /OPCOM_LOG qualifier. This qualifier specifies the file specification of the log file to record various OPCOM messages broadcast by attached database processes.

The following is an example of the "OPCOM Messages" screen:

```
 Oracle Rdb X7.1-00 Performance Monitor OPCOM Log
 Database KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS.RDB;2
 OPCOM Log created  7-JUN-1999 06:25:10.61
06:25:12.13 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS2.AIJ;1"
(missed 4)
06:25:12.13 2A526A05:1 AIJ Log Catch-Up Server activated (missed 4)
06:25:21.07 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS3.AIJ;1"
06:27:50.11 2A53D804:1 After-image journal 14 switch-over in progress (to 15)
06:27:51.26 2A53D804:1 After-image journal switch-over complete
06:28:53.44 2A47061D:1 AIJ backup operation started
06:28:59.00 2A53D804:1 Automatic backup utility cannot be invoked for TCS3
ABS AIJ backup
06:30:29.27 2A53D804:1 After-image journal 15 switch-over in progress (to 16)
06:30:30.41 2A53D804:1 After-image journal switch-over complete (missed 2)
06:31:40.23 2A53DE21:1 AIJ backup operation started
06:31:54.56 2A53D804:1 Automatic backup utility cannot be invoked for TCS4
ABS AIJ backup
06:32:20.88 2A53DE21:1 AIJ backup operation completed
06:34:17.52 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS1.AIJ;1"
06:36:56.55 2A51E222:1 AIJ backup operation started
06:37:30.46 2A51E222:1 AIJ backup operation completed
06:37:44.74 2A53D804:1 Replication server stalled waiting for MSN 5700 (AIJ
17:12526)
06:37:44.74 2A526A05:1 Replication server stalled waiting for MSN 5700 (AIJ
17:12526)
06:41:38.39 2A53D34B:1 After-image journal 17 switch-over in progress (to 18)
06:42:13.87 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS2.AIJ;1"
06:42:15.49 2A53D34B:1 After-image journal switch-over complete (missed 2)
06:42:19.96 2A53D34B:1 After-image journal 18 switch-over in progress (to 19)
06:42:23.31 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS3.AIJ;1"
06:42:23.31 2A53D34B:1 After-image journal switch-over complete
06:42:46.27 2A4D3423:1 AIJ backup operation started
06:44:26.61 2A53D804:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:32.13 2A53D804:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:33.27 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS4.AIJ;1"
(missed 2)
06:44:34.45 2A53D804:1 After-image journal switch-over complete
06:44:34.45 2A532216:31 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:35.66 2A4F9A1A:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:35.66 2A461220:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:36.88 2A505617:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:36.88 2A517218:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:36.88 2A462C1F:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:40.29 2A46EC1C:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:41.45 2A4EBA19:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:44:43.71 2A4F361E:1 Automatic backup utility cannot be invoked for TCS2
ABS AIJ backup
06:46:10.60 2A53E224:1 AIJ backup operation started
06:46:20.53 2A53D804:1 Automatic backup utility cannot be invoked for TCS3
ABS AIJ backup
06:46:42.36 2A53E224:1 AIJ backup operation completed
06:47:11.60 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS5.AIJ;1"
06:48:16.92 2A4D7825:1 AIJ backup operation started
06:48:47.36 2A4D7825:1 AIJ backup operation completed
06:49:40.60 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS1.AIJ;1"
```

```
(missed 2)
06:49:41.76 2A53D804:1 After-image journal switch-over complete
06:52:45.64 2A53D34B:1 After-image journal 22 switch-over in progress (to 23)
06:52:50.04 2A53D804:1 Opening "KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS2.AIJ;1"
06:52:50.04 2A53D34B:1 After-image journal switch-over complete
06:53:49.82 2A474E26:1 AIJ backup operation started
06:54:29.37 2A53D804:1 Automatic backup utility cannot be invoked for TCS1
ABS AIJ backup
06:55:14.93 2A53D804:1 Automatic backup utility cannot be invoked for TCS1
ABS AIJ backup
06:55:29.91 2A53D804:1 Automatic backup utility cannot be invoked for TCS1
ABS AIJ backup
```

The following is an example of the /OPCOM_LOG qualifier log file contents:

```
 Oracle Rdb X7.1-00 Performance Monitor OPCOM Log
 Database KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS.RDB;2
 OPCOM Log created  4-JUN-1999 14:52:16.81
14:52:26.87 2A506125:1 After-image journal 33 switch-over in progress (to 34)
14:52:28.44 2A506125:1 After-image journal switch-over complete
14:53:40.79 2A506125:1 After-image journal 34 switch-over in progress (to 35)
14:53:42.22 2A506125:1 After-image journal switch-over complete
15:03:21.67 2A506125:1 After-image journal 36 switch-over in progress (to 37)
15:03:23.16 2A506125:1 After-image journal switch-over complete
15:03:24.82 2A459220:1 AIJ backup operation started
15:03:27.82 2A459220:1 AIJ backup operation completed
```

When recording OPCOM messages, it is possible to occassionally miss a few messages for a specific process. When this occurs, the message "n missed" will be displayed in the log file.

It is possible to record specific operator classes of OPCOM messages if you specify the /OPTION=VERBOSE qualifier. This qualifier records only those messages that can be received by the process executing the RMU/SHOW Statistic utility. For example, if the process is enabled to receive operator class CENTRAL, then specifying /OPCOM_LOG=opcom.log/OPTION=VERBOSE will record all CENTRAL operator messages. Conversely, specifying only the /OPCOM_ LOG=opcom.log qualifier will record all database-specific OPCOM messages generated from this node.

The operator-specific log file output format is different from the database-specific contents, because the output is being captured directly from OpenVMS. The following is an example of the operator-specific log file contents for the CLUSTER and CENTRAL operator classes:

```
 Oracle Rdb X7.1-00 Performance Monitor OPCOM Log
 Database KODA_TEST:[R_ANDERSON.TCS_MASTER]TCS.RDB;2
 OPCOM Log created 11-JUN-1999 10:52:07.53
11-JUN-1999 10:52:23.85)  Message from user RDBVMS on ALPHA4  Oracle Rdb
X8.0-00 Event Notification for Database  _$111$DUA368:[BBENTON.TEST]
MF_PERSONNEL.RDB;1    AIJ Log Server terminated
11-JUN-1999 10:52:25.49)  Message from user RDBVMS on ALPHA4  Oracle Rdb
X8.0-00 Event Notification for Database  _$111$DUA368:[BBENTON.TEST]
MF_PERSONNEL.RDB;1    AIJ Log Roll-Forward Server started
11-JUN-1999 10:52:26.06)  Message from user RDBVMS on ALPHA4  Oracle Rdb
X8.0-00 Event Notification for Database  _$111$DUA368:[BBENTON.TEST]
MF_PERSONNEL.RDB;1    AIJ Log Roll-Forward Server failed
.
.
.
11-JUN-1999 10:54:16.05)  Message from user INTERnet on ALPHA4  TELNET Login
Request from Remote Host: 138.2.136.180    Port: 1252
11-JUN-1999 10:54:16.36)  Message from user RDBVMS on ALPHA4  Oracle Rdb
X8.0-00 Event Notification for Database  _$111$DUA368:[BBENTON.TEST]
MF_PERSONNEL.RDB;1    AIJ Log Catch-Up Server terminated
```

5.1.6 New Restricted_Access Qualifier for RMU/LOAD

RMU Load now supports the Restricted_Access qualifier when attaching to an Oracle Rdb database. This option allows a single process to load data and enables some optimizations available only when Restricted Access is in use.

If you are loading a table from an RMU Unload file which contains LIST OF BYTE VARYING data then the Restricted_Access qualifier will reserve the LIST areas for exclusive access. This reduces the virtual memory used by long transactions in RMU Load and also eliminates I/O to the snapshot files for the LIST storage areas.

The Restricted_Access and Parallel qualifiers are mutually exclusive and may not both be specified on the RMU Load command line, or within a plan file. While RMU Load is running with this option enabled, no other user may attach to the database. The default is Norestricted_Access.

5.1.7 RDO EDT Editor on OpenVMS Alpha Now Available

Previously, on OpenVMS Alpha, the RDO editor was restricted to the TPU editor. The EDT editor was not available via the RDO$EDIT logical name.

This problem has been corrected. The RDO$EDIT logical name controls the editor selection between EDT and TPU as it does on OpenVMS VAX.

5.1.8 New Options Added to SQL EXTRACT Function

In Oracle Rdb7 Release 7.0.3.1, the SQL EXTRACT function is being enhanced with two new options: WEEK_NUMBER and YEAR_WEEK. These options return the week number as defined by the International Standard ISO 8601:1988 "Data elements and interchange formats - Information interchange - Representation of dates and times".

Section 3.17 of this standard defines a week as "week, calendar: A seven day period within a calendar year, starting on a Monday and identified by its ordinal number within a year; the first calendar week of the year is the one that includes the first Thursday of that year. In the Gregorian calendar, this is equivalent to the week which includes 4 January."

WEEK_NUMBER is a number between 1 and 53 representing the week of the year (most years only have 52 weeks). A week starts on Monday and has most of its days falling in a specific year.

YEAR_WEEK is a variation of the WEEK_NUMBER that includes the year (including the century) in which the week logically falls. The values range from 185901 through 999952 (higher values are possible if dates are constructed with a year beyond 9999). The last two digits of the value are identical to the value returned by the WEEK_NUMBER option.

The following example shows the new function results:

```
SQL> select dt,
cont>       extract (week_number from dt),
cont>       extract (year_week from dt)
cont> from week_sample
cont> order by dt;
 DT
 1859-01-07             1        185901
 1999-01-01            53        199853
 1999-01-04             1        199901
 1999-01-10             1        199901
 1999-12-31            52        199952
 2000-01-01            52        199952
 2000-01-03             1        200001
 2000-02-28             9        200009
 2000-02-29             9        200009
 2000-03-01             9        200009
 9999-12-31            52        999952
11 rows selected
```

Usage Notes

- The source date/time expression must include a date component; DATE (ANSI), TIMESTAMP, or DATE (VMS). Attempts to use other data types will result in an error. For example:

  ```
  SQL> select extract (week_number from current_time) from ...;
  %RDB-E-CONVERT_ERROR, invalid or unsupported data conversion
  -RDMS-E-EXT_WEEKDAY_TS, invalid type for EXTRACT - must be DATE or TIMESTAMP
  ```

- Note that neither WEEK_NUMBER nor YEAR_WEEK can be calculated for the year 1858, or the first few days of 1859 because the Oracle Rdb date only supports part of the year 1858, and therefore the calculation cannot be made. For example:

  ```
  SQL> select extract (week_number from date'1859-1-1') from ...;
  %RDB-E-ARITH_EXCEPT, truncation of a numeric value at runtime
  -COSI-F-IVTIME, invalid date or time
  ```

- Note that the year in which the week falls may not be the same as the extracted year from the date/time value because days at the start of a calendar year may logically fall in the last week of the previous year, and days at the end of a calendar year may logically fall in the first week of the following year.