Oracle Rdb7™ for OpenVMS

Release Notes

Release 7.0.1

ORACLE®

Oracle Rdb Release Notes

Release 7.0.1

This document was prepared using VAX DOCUMENT Version 2.1.

Contents

3 Documentation Corrections

4 Known Problems and Restrictions

5 Enhancements

Examples

Tables

Preface

Purpose of This Manual

This manual contains release notes for Oracle Rdb Release 7.0.1. 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 Rdb for OpenVMS Alpha and Oracle Rdb for OpenVMS VAX, which are referred to by their abbreviated name, Oracle Rdb.

Intended Audience

This manual is intended for use by all Oracle Rdb users. Read this manual before you install, upgrade, or use Oracle Rdb Release 7.0.1.

Document Structure

This manual consists of five chapters:

Chapter 1	Describes how to install Oracle Rdb Release 7.0.1.
Chapter 2	Describes software errors corrected in Oracle Rdb Release 7.0.1.
Chapter 3	Provides information not currently available in the Oracle Rdb documentation set.
Chapter 4	Describes problems, restrictions, and workarounds known to exist in Oracle Rdb.
Chapter 5	Describes new features added in Oracle Rdb Release 7.0.1.

1
Installing Oracle Rdb Version 7.0.1

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:

- The installation requires approximately 125,000 free blocks on your system disk for OpenVMS VAX systems; 205,000 blocks for OpenVMS Alpha systems.
- Oracle Rdb Version 7.0 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 Rdb on all nodes in the cluster before proceeding.

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 Rdb Version 7.0.1 are:

- RDBSA070 for Oracle Rdb for OpenVMS VAX standard version.
- RDBASA070 for Oracle Rdb for OpenVMS Alpha standard version.
- RDBMMVA070 for Oracle Rdb for OpenVMS VAX multiversion.
- RDBAMVA070 for Oracle Rdb 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:[RDBSA070.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 this manual, which is the only documentation for this release.

The following example shows how to start the installation procedure on device MTA0: and print the documentation.

```
$ @SYS$UPDATE:VMSINSTAL RDBSA070 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 Rdb installed will probably not be usable.

1.4 After Installing Oracle Rdb

This update provides a new Oracle Rdb Oracle TRACE facility definition. Any Oracle TRACE selections that reference Oracle Rdb will need to be redefined to reflect the new facility version number for the updated Oracle Rdb facility definition, "RDBVMSV7.0-1".

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

The installation procedure inserts the Oracle Rdb facility definition into a library file called EPC$FACILITY.TLB. To be able to collect Oracle Rdb 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=RDBVMSV7.0-1 -
   _$ /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 Rdb, the process executing the INSERT DEFINITION command must use the version of Oracle Rdb that matches the version used to create the Oracle TRACE administration database or the INSERT DEFINITION command will fail.

2
Software Errors Fixed

This chapter describes software errors that are fixed by Oracle Rdb Version 7.0.1.

2.1 Software Errors Fixed in Oracle Rdb Version 7.0.1

2.1.1 Bugchecks at PIOFETCH$WITHIN_DB + 2DC

Bug 343349.

Oracle Rdb would bugcheck at PIOFETCH$WITHIN_DB + 2DC when the following conditions were true:

1. When a process attached to the database and a storage area was greater than one gigabyte in size,
2. The storage area had doubled in size,
3. The process attempted to extend the storage area to more than double its original size.

or, similarly:

1. When the process attached to the database and a storage area was less than one gigabyte in size,
2. The storage area had grown to more than two gigabytes in size,
3. The process attempted to extend the storage area to more than two gigabytes in size.

The bugcheck would contain the text similar to the following (this example is for V6.1 VAX):

```
***** Exception at 0020736C : PIOFETCH$WITHIN_DB + 000002DC
%COSI-F-BUGCHECK, internal consistency failure
```

There was a limitation in Oracle Rdb that prevented a process from extending a storage area when the extent would cause the storage area to extend to the greater of

1. 2 gigabytes

   or

2. twice its original size at the time the process attached to the database.

This restriction has been lifted for local buffer databases only. The restriction is still in place for databases that use global buffers. However, the limit for global buffer databases has been raised such that the storage area would have to more than double from an attach time size the greater of either 2 gigabytes, or twice its original size at the time the process attached to the database.

For local buffer databases, restarting the application would prevent the bugchecks. For global buffer databases, it was necessary to close and re-open the database if a storage area had more than doubled in size since the database was opened and the storage area had a size of two gigabytes when it was opened.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.2 RMU/RESTORE Bugchecks at FILE_CLEANUP + 72

Bug 373358.

A RMU/RESTORE or DBO/RESTORE command would bugcheck after restoring after image journal (AIJ) information in the restored database.

```
%DBO-I-AIJRECEND, after-image journal "state" recovery complete
%SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual
    address=7FFFFFF4, PC=00083373, PSL=03C00002
%DBO-I-BUGCHKDMP, generating bugcheck dump file ...
```

The bugcheck dump would contain an error message similar to the following:

```
***** Exception at 000850F3 : FILE_CLEANUP + 00000072
%SYSTEM-F-ACCVIO, access violation, reason mask=00,
    virtual address=7FFFFFF4, PC=000850F3, PSL=03C00002
```

The problem was due to an error in the AIJ recovery code that would sometimes corrupt various in-memory data structures.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.3 Read Only Transactions Returned Incorrect Results

Bug 397364.

If a transaction inserted a row in a table in the same physical database location (DBKEY) as a previously deleted row and then rolled back the insert, other read-only transactions executing at the time would not properly find the previous contents of the older row.

The following example demonstrates the problem:

```
-- Session 1:

SQL> create table foo (f1 int);
SQL> insert into foo values (1);
1 row inserted
SQL> insert into foo values (2);
1 row inserted
SQL> commit;
SQL> set transaction read only;

-- Session 2:

SQL> delete from foo where f1 = 2;
1 row deleted
SQL> commit;
SQL> insert into foo values (3);
1 row inserted
SQL> rollback;

-- Session 1:
-- The following query should have returned both rows
-- initially stored in the table (1 and 2) but
-- it only returned one row.
```

```
SQL> select * from foo;
          F1
           1
1 row selected
```

This problem would lead to various problems such as:

- Online verifies erroneously reporting corrupt databases
- Online backups not properly backing up some rows in the database
- Applications not always getting the correct results back from queries executed in a read-only transaction

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.4 Errors from RMU/VERIFY After an RMU/RESTORE Command

Under particular circumstances, RMU/RESTORE would create SPAM pages that had the incorrect threshold number for a data page. RMU/VERIFY would detect these errors during an off-line verify and display the PGSPAMENT message for those entries. The incorrect value did not cause any data corruption or incorrect results to queries, but it could result in free space on data pages not being used. The circumstances required for this to occur were:

- The page must have been in a storage area with a uniform page format.
- The logical area owning the page must have had thresholds defined for it.
- The correct value of the page fullness that should have been stored in the SPAM was 2.

Under these circumstances, RMU RESTORE used to store a SPAM threshold value of 3, instead of the correct value.

The following example shows a script that creates a database that meets the criteria described above. The database is backed up, deleted, then restored. After the restore operation, VERIFY returns a PGSPAMENT error.

```
$ sql
create database
    filename 'TESTCASE_DB.RDB'
    create storage area RDB$SYSTEM
        filename 'TESTCASE_SYSTEM_AREA.RDA'
    create storage area TEST_AREA1
        filename 'TEST_AREA1.RDA';

create table TABLE1 (
    FLD1
        CHAR (50));

create storage map TABLE1_MAP
    for TABLE1
    store in TEST_AREA1 (thresholds are (30, 60, 80));

insert into table1 values
('12345678901234567890123456789012345678901234567890');

<repeat insert 9 more times>

commit work;
```

```
exit
$ rmu/backup/nolog testcase_db testcase_db.rbf
$ sql
drop database filename testcase_db;
exit
$
$ rmu/restore/nocdd/nolog testcase_db.rbf
%RMU-I-AIJRSTAVL, 0 after-image journals available for use
%RMU-I-AIJISOFF, after-image journaling has been disabled
%RMU-W-USERECCOM, Use the RMU Recover command. The journals are not
available.
$
$ rmu/verify/all/nolog testcase_db
%RMU-W-PGSPAMENT, area TEST_AREA1, page 5
                  the fullness value for this data page does not match
                  the threshold value in the space management page
                  expected: 3, computed: 2
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.5 Error from Parallel Load

If you tried to use parallel load to load data from a delimited text file into a partitioned table and the field used for partitioning the table was not the first field in the data file, then LOAD would use incorrect data when it attempted to determine the correct partition for each row. In most cases, LOAD would abort with an INPCONERR error.

The following example shows the results of an attempt to load a database from a delimited text file when the partitioning column was in the 12th field in the data file.

```
ALPHA>RMU/LOAD/PLAN bench.par
%RMU-I-EXECUTORMAP, Executor EXECUTOR_1 (pid: 532066CA) will load
storage area BENCH01.
%RMU-I-EXECUTORMAP, Executor EXECUTOR_2 (pid: 532052CB) will load
storage area BENCH02.
%RMU-I-EXECUTORMAP, Executor EXECUTOR_3 (pid: 532066CC) will load
storage area BENCH03.
%RMU-I-EXECUTORMAP, Executor EXECUTOR_4 (pid: 532066CD) will load
storage area BENCH04.
%COSI-F-INPCONERR, input conversion error
%RMU-I-DATRECREAD,  1 data records read from input file.
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.6 RDO Global Field Matching Fails When SCALE is Negative

It is legal in RDO to fully specify field (aka column) attributes within a DEFINE RELATION statement. If the field name and attributes match an existing global field (aka domain) then that global field is used, otherwise a new global field is created by the DEFINE RELATION statement.

This behavior was not working in the following releases of Rdb: V5.0, V5.1, V6.0, V6.1, and V7.0. If the scale of a numeric value was negative (which is typical) then an error was generated. If the scale was zero, or positive then no error is generated.

```
RDO> define field global1 datatype is signed word scale -3.
RDO> commit;
RDO>  define relation gbl2_tbl.
cont>     global1 datatype is signed word scale -3.
cont> end gbl2_tbl.
%RDO-F-FIELD_EXISTS, field GLOBAL1 already exists in this database
```

In this case RDO should have detected that the new field matched exactly the existing global field.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.7 Bugcheck During Record Expansion

In some databases which used record compression, a bugcheck would occur during record expansion. The error within the bugcheck was:

```
Exception at RDMS$$ALPHA$DYN_EXPAND + 009C
```

Other offsets were also possible.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.8 RMU Errors Were not Always Directed to SYS$ERROR

RMU did not always direct error messages to the error stream. Instead, the error messages would show up in the output stream. Most users did not notice this, because the error stream and the output stream are usually the same. However, if you wanted to redirect the error stream to somplace different, there were some cases where error messages would not show up in the error stream.

The following example shows an attempt to use the RMU/DUMP command on a non-existent file. The output stream is redirected to file OUT.TXT and the error stream is redirected to file ERROR.TXT. Notice that the BADDBNAME error appears in the output file but not the error file.

```
$ DEFINE SYS$OUTPUT OUT.TXT
$ DEFINE SYS$ERROR ERROR.TXT
$
$ RMU/DUMP/HEADER FRED
$
$ DEASS SYS$OUTPUT
$ DEASSIGN SYS$ERROR
$
$ TYPE OUT.TXT
%RMU-W-BADDBNAME, can't find database root DS1_USER1:[DBA]FRED.RDB;
-RMS-E-FNF, file not found
%RMU-F-CANTOPNROO, cannot open root file "FRED"
$
$ TYPE ERROR.TXT
%RMU-F-CANTOPNROO, cannot open root file "FRED"
```

RMU will now display the error message in the error stream. The same commands now give the following results:

```
$ DEFINE SYS$OUTPUT OUT.TXT
$ DEFINE SYS$ERROR ERROR.TXT
$
$ RMU/DUMP/HEADER FRED
$
$ DEASS SYS$OUTPUT
$ DEASSIGN SYS$ERROR
$
$ TYPE OUT.TXT
%RMU-W-BADDBNAME, can't find database root DS1_USER1:[DBA]FRED.RDB;
-RMS-E-FNF, file not found
%RMU-F-CANTOPNROO, cannot open root file "FRED"
$
$ TYPE ERROR.TXT
%RMU-W-BADDBNAME, can't find database root DS1_USER1:[DBA]FRED.RDB;
-RMS-E-FNF, file not found
%RMU-F-CANTOPNROO, cannot open root file "FRED"
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.9 RMU/SHOW STATISTIC "Stall Messages" Could not be Internationalized

All RMU/SHOW STATISTIC "Stall Messages" were hard-coded and could not be internationalized. This has been corrected. All stall messages are now stored as messages in the standard message file.

2.1.10 Hash Index Incurs Extra I/Os.

Hash indexes created under Oracle Rdb could result in redundant I/Os when data was accessed by them. This behavior was seen using the RMU/SHOW STATISTICS utility. Dumping the hash index area showed that if collisions were present, they would be linked linearly by overflow pointers when this was not necessary.

The following example shows the problem:

```
                0038 2005  02E2  line 2: bucket for hash index 56
                           ....  total hash bucket size: 127
      FFFF FFFFFFFF FFFF   02E6  bucket overflow -1:-1:-1
                     00    02EE  flags 0
               00000001    02EF  duplicate count 1
      0032 00000011 000D   02F3   pointer 50:17:13
                     65    02FB   key len: 101 bytes
20202020202020202020706C656800  02FC   key: '.help           '
20202020202020202020202020202020  030C   key: '                '
                           ::::  (4 duplicate lines)
            2020202020     035C   key: '     '
                     00    0361  padding '.'

                0038 2005  0362  line 1: bucket for hash index 56
                           ....  total hash bucket size: 127
      0038 00000073 0002   0366  bucket overflow 56:115:2
                                                ^^^^^^^^
                     00    036E  flags 0
               00000001    036F  duplicate count 1
      0032 00000011 0005   0373   pointer 50:17:5
                     65    037B   key len: 101 bytes
20202020202020202020202074616300  037C   key: '.cat            '
20202020202020202020202020202020  038C   key: '                '
                           ::::  (4 duplicate lines)
            2020202020     03DC   key: '     '
                     00    03E1  padding '.'
```

Hash elements "cat" and "help" hash to the same hash value. They should have been in the same hash bucket instead they were in two different buckets linked by a bucket overflow pointer.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.11 A Race Condition Caused the Corruption of Indexes and Segmented Strings

Bug 352753.

Several database corruption problems have been discovered and were related to a race condition within Oracle Rdb.

Symptoms included:

- Corrupted B-tree indexes. This includes index keys out of sequence, index and table out of synchronization.
- Corrupted Hash indexes.
- Part of a segmented string structure missing.

This problem occurred in Rdb 6.1 and 7.0. When multiple users were doing select queries before inserting or deleting data there was a window when corruption of indexes and segmented strings could occur. A full verify of the database would identify the corruption.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.12 Excessive I/O During an Insert Into a Hashed Index

Bug 387602.

There were some cases where a single insert into a hash index incurred excessive I/O. This could be seen using the RMU/SHOW STATISTICS utility. This was the result of attempts at space reclamation within a hash bucket chain. Hash bucket fragments were marked in order to reclaim space and each hash bucket then had to be journaled. The longer the hash bucket chain the more I/Os that were incurred.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.13 Excessive SPAM Fetches During a Single Update, Insert and Delete

Bug 359926.

When using one large table to update, insert and delete data from other tables, excessive SPAM fetches occurred. In a uniform area, when the logical area became full, Oracle Rdb would walk the SPAM intervals again and again only to find out that another clump needed to be allocated. This resulted in excessive SPAM fetches.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.14 Random Arithmetic Trap Errors During RMU/LOAD Operations

Bug 418775.

During an RMU/LOAD command when a sorted ranked index was being built ARITH_EXCEPT errors could occur and the RMU/LOAD operation would be terminated.

```
$ RMU/LOAD /CORRESPONDING/COMMIT=1000/NOCONSTRAINTS/RECORD=(FILE=COT.RRD) -
/TRANSACTION_TYPE=EXCLUSIVE/PLACE FX CONFO_OUT_TRADE COT.DAT
%RMU-I-LOADERR, Error loading row 74.
%RDB-E-ARITH_EXCEPT, truncation of a numeric value at runtime
1SYSTEM-F-HPARITH, high performance arithmetic trap, Imask=00000000, Fmask=0000
000, summary=02, PC=0121AA90, PS=0000000B
-SYSTEM-F-FLTINV, floating invalid operation, PC=0121AA90, PS=0000000B
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.15 Monitor Log File Fragmentation

In the past the monitor log file has grown in small increments as the file was written. This growth behavior contributed to disk and file fragmentation and also caused additional I/O overhead for the disk where the log file was located.

The default monitor log allocation and extension quantity is now 512 blocks. Furthermore, a logical name RDM$MON_LOG_EXTEND_SIZE may now be used to increase the monitor log file allocation and extension quantity. Define the logical name RDM$MON_LOG_EXTEND_SIZE system-wide with an integer value to specify the monitor log file allocation and extension quantity.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.16 Rollback After a Delete Could Cause a Bugcheck

In certain combinations of locked and free space on a database page, a rollback of row deletions could result in bugchecks in the routine RUJUTL$ROLLBACK_LOOP and possibly cause process deletion.

The following script shows how this problem could occur:

```
$ @RDM$DEMO:PERSONNEL SQL M NOCDD "" SYS$DISK:[]
$ MCR SQL$
SQL> ATTACH 'FILE MF_PERSONNEL';
SQL> DELETE FROM EMPLOYEES;
100 rows deleted
SQL> ROLLBACK;
%RDMS-I-BUGCHKDMP, generating bugcheck dump
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

Additional information about the length of an erased row is now stored in the RUJ file in order to help the rollback operation be able to make better use of the locked space on a database page.

2.1.17 Errors from SELECT with GROUP BY When no Data is in the Underlying Table

Bug 418444.

In prior releases of Rdb it was possible to receive an error message such as RDB-E-CONVERT_ERROR or RDB-E-ARITH_EXCEPT when processing a SELECT that (directly or indirectly via a view) contained a GROUP BY clause. For this to occur, the GROUP BY clause had to reference an expression involving DATE data types or certain string operations such as SUBSTRING, and there had to be no data in the table contributing to the row data that was to be partitioned by the GROUP BY clause. In effect, certain actions related to the partitioning were still occurring even if there was no data.

The following example displays a view and the SELECT expression which results in a convert error.

```
create table ttest (d integer,i integer);

create view test_view as
 select sum(i) as sumi,
  cast(cast(cast(d as char(8)) as date vms) as date ansi)
     as d1
  from ttest group by d;
```

```
select sum(sumi) from test_view group by d1;
%RDB-E-CONVERT_ERROR, invalid or unsupported data conversion
-COSI-F-IVTIME, invalid date or time
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

If there is no data to be partitioned, none of the GROUP BY actions will occur, and the SELECT will return the expected result of '0 rows selected'.

2.1.18 Process Deletion After Several Date Expressions

In some databases certain SQL statements which used several date expressions could cause process deletion. An example of such an expression is:

```
select * from dem_trt_delai
  where cast(maj_date as date ansi) +
  cast (annee_numero as interval year) +
  cast(mois_numero as interval month) +
  cast(jour_numero as interval day)
    between cast(maj_date as date ansi)
 and cast(maj_date as date ansi);
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.19 Hot Standby Database Attributes Lost

Preconfigured Hot Standby database attributes were overwritten with default values when the /MASTER_ROOT or /STANDBY_ROOT qualifiers were specified when restarting database replication.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.20 ALTER INDEX Changes HASH ORDERED to HASH SCATTERED

When ALTER INDEX command was used to change the partitioning of a HASH ORDERED index, it incorrectly recreated the partitions as HASH SCATTERED. This could be verified using the RMU/DUMP LAREA=RDB$AIP command in Rdb V6.1 and later versions. The SQL SHOW INDEX, or SHOW TABLE (INDEX) commands still reported the index as HASH ORDERED.

The data was still intact, and only the access method was affected by this problem. The effect was that the hash bucket placement was not as expected by the database designer.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.21 ALTER STORAGE MAP Did Not Reorganize Data

If a storage map defined a table as PARTITIONING IS NOT UPDATABLE (also known as a strictly partitioned table) then the ALTER STORAGE MAP should have reorganized the data to maintain the strictly partitioned nature of the data. This was not done. This couls lead (after the ALTER STORAGE MAP) to range retrieval queries returning too little data when a sequential scan of the partitions was performed.

This problem has been corrected in Oracle Rdb Version 7.0.1.

When ALTER STORAGE MAP is executed on a strictly partitioned storage map and storage areas are added, dropped or changed then the REORGANIZE AREAS clause is implicit.

2.1.22 SQLCODE/SQLSTATE Incorrectly Propagated from SQL Functions and Procedures

When an SQL stored function or procedure was called it returned to the caller's environment warnings generated within the stored function. In Rdb7 SQLSTATE and SQLCODE were incorrectly being set to any END OF DATA conditions from within the called stored routine.

This could cause queries executing under the 'SQL92', or 'ORACLE LEVEL1' dialects which use SQL functions to terminate prematurely. The END OF DATA state from the called stored routine was incorrectly being used to terminate the query which called the function. For example, the query shown below should always return one row, not zero rows.

```
SQL> create module m_sqlcode
cont>     language SQL
cont>
cont>     function f_end_of_data () returns integer;
cont>     begin
cont>     declare :x integer;
cont>     select 1 into :x from employees where employee_id = 'xxxxx';
cont>     return -1;
cont>     end;
cont>
cont> end module;
SQL>
SQL> select 'value' from rdb$database;

 value
1 row selected
SQL>
SQL> select f_end_of_data () from rdb$database;
0 rows selected
SQL>
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

SQL stored functions and procedures no longer return the END OF DATA codes (SQLCODE 100 and SQLSTATE '02000') to the caller. Other warnings (SQLSTATE values starting with '01') such as 'String truncation during assignment' and 'NULL eliminated in aggregate' are still propagated to the caller.

2.1.23 ALTER INDEX Waiting Too Long to Validate Storage Area Name

When ALTER INDEX command processed a new index storage map it did not validate the storage area names until it had processed some of the data. This caused wasted time during execution and during rollback of the failed map.

This problem has been corrected in Oracle Rdb Version 7.0.1.

Oracle Rdb now validates the storage areas during the initial processing phase for the index, and before it starts performing I/O to alter the index.

2.1.24 RMU/EXTRACT Generated Incorrect Keyword for PARTITIONING Clause

Bugs 430928 and 422267.

The PARTITIONING IS NOT UPDATABLE clause was extracted by RMU Extract with a spelling error. The workaround was to manually correct the spelling of the word from 'updateable' to the correct spelling 'updatable'.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.25 Unexpected RDMS$_WISH_LIST Generated by ALTER DATABASE ... DROP STORAGE AREA CASCADE

When the clause DROP STORAGE AREA CASCADE was issued as part of the ALTER DATABASE statement it could generate an unexpected RDMS$_WISH_ LIST error with no explanation as to the cause of the problem. For example:

```
SQL> alter data file db$:vrp_test drop storage area area cascade;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-E-WISH_LIST, feature not implemented yet
```

This error occurred when an attempt was made to alter a vertically partitioned storage map (a STORAGE MAP which includes the STORE COLUMNS clause). This type of map can not be altered explicitly using ALTER STORAGE MAP (as documented in the Rdb7 documentation), nor can they be implicitly altered using the DROP STORAGE AREA CASCADE of the ALTER DATABASE statement. This last point was unfortunately not documented in the Rdb7 documentation.

For this update to Rdb7, an alternate message is raised which names the storage map causing the problem.

```
SQL> alter data file db$:vrp_test drop storage area area cascade;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-E-VRPINVALID, invalid operation for storage map "M6"
```

The description for this error states that the map is a vertically partitioned map and may not be altered. Oracle recommends that vertically partitioned tables not be mapped to storage areas shared by other tables or indexes as this will preclude the use of DROP STORAGE AREA CASCADE for those other tables.

2.1.26 Vertically Partitioned Tables Allocated Space for COMPUTED BY Columns

When a table was vertically partitioned (columns from the same table are stored in different storage areas), Rdb7 incorrectly allocated storage space which included space for COMPUTED BY columns. COMPUTED BY columns do not require disk space so this is wasted space. This problem does not exist for tables which are not vertically partitioned.

This problem is corrected in Oracle Rdb Version 7.0.1 . However, any existing tables which match these characteristics will need to be rebuilt. The old format of the data is incorrect and any partition which contains COMPUTED BY columns (usually the last) will not be read correctly after Oracle Rdb Version 7.0.1 is installed.

Any table which has COMPUTED BY columns and which is also vertically partitioned (has a storage map containing the STORE COLUMNS clause) will need to be unloaded, recreated and then reloaded. Please use RMU Unload (or another application) to unload the data **before** Oracle Rdb Version 7.0.1 is installed, then drop the table. After Oracle Rdb Version 7.0.1 is installed, you can then recreate the storage map and reload the saved data. The new on-disk image will now contain only those columns which contain data and should occupy less disk space.

2.1.27 VARCHAR Data in Vertically Partitioned Tables not Saved Correctly

When a table was vertically partitioned (columns from the same table are stored in different storage areas), Rdb was not copying data for VARCHAR columns correctly to the target partition. The last two bytes of the VARCHAR data may have been corrupted when read by a query.

This problem is corrected in Oracle Rdb Version 7.0.1 . However, any existing tables which match these characteristics (vertically partitioned and containing VARCHAR columns) will need to be reloaded. Note that in the case where the VARCHAR data is smaller than the declared VARCHAR length no corruption occurs.

This problem does not affect any other data type or COMPUTED BY columns of VARCHAR type.

2.1.28 ALTER TABLE not Fully Supported for Vertically Partitioned Tables

When a table was vertically partitioned (columns from the same table are stored in different storage areas) the ALTER TABLE statement could corrupt the on-disk stored data.

Specifically, if the ALTER TABLE command changed a data type, added or removed columns then the data from the vertical record partitions stored before the ALTER were not correctly processed.

This problem is corrected in Oracle Rdb Version 7.0.1 . However, any existing tables that have been altered should be reloaded (the table dropped, recreated and reloaded).

2.1.29 Error When Creating Storage Map with Many Columns in STORE COLUMNS Clause

Bug 429969.

In Oracle Rdb7, when creating a storage map that vertically partitions a table and stores many columns in a particular partition, virtual memory could be exhausted and the COSI-F-EXQUOTA error could be seen.

The following example shows this error on a storage map with many columns (75) being stored in the first partition.

```
create storage map TABLE_MAP for TABLE_A
  store columns
 (COLUMN1, COLUMN2, COLUMN3, COLUMN4, COLUMN5...COLUMN75)
  in AREA1
store columns
 (COLUMN76, COLUMN77)
  in AREA2;
%COSI-F-EXQUOTA, exceeded quota
-SYSTEM-F-VASFULL, virtual address space is full
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

The create storage map command now executes successfully for any number of columns.

2.1.30 Problems Corrected for Sequential Scan

In Rdb7 a sequential scan of a partitioned table (a table with a storage map across multiple areas) which was defined (or defaulted) as PARTITIONING IS UPDATABLE would sometimes not scan all partitions. This means that some data was not returned for the query.

The first execution of the query would correctly scan all storage areas in the map. However, subsequent executions of the query would not perform a complete scan. Examples of multiple executions include: multiple calls to a stored procedure, re-opening a cursor, and executing the same SQL module language or precompiler query multiple times.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.31 Problems Corrected for Strict Partitioning

Bug 430931.

When a table's storage map has the attribute PARTITIONING IS NOT UPDATABLE then mapping of data to a storage area is strictly enforced. This is known as **strict partitioning**. Oracle Rdb Version 7.0.1 corrects several problems with the strict partitioning functionality.

1. Table cardinality incorrectly updated after a partial sequential scan.

   When a full table scan was performed, Rdb collected and updated the table cardinality. However, when strict partitioning was used only a subset of the table data was scanned, and so an incorrect cardinality could be applied.

   This is now corrected. The cardinality is not updated for a partial sequential scan. However, RMU/COLLECT should be executed on the strictly partitioned tables to ensure that the cardinality is set correctly.

2. Incorrect partition selection when columns have type VARCHAR.

   If the query provided VARCHAR values (either literals, column values or host variables) the last two bytes of the value were not used correctly for partition selection. This was only a problem if the data value was equal or within one character of the VARCHAR declared length.

   This problem has been corrected. The full VARCHAR selection value is used. No other data types were affected by this problem. If VARCHAR columns were used for strict partitioning then the table should be reloaded.

3. Strict Partitioning not used for left, right or full outer joins.

   Bug 430931.

   When an outer join query was used, then strict partitioning was disabled for sequential scans. This meant that more I/O was required to solve these joins than was needed.

   This problem has been corrected. Where possible, strict partitioning will be used when both an outer join and a sequential retrieval is required. However, adding a SORTED index may result in less I/O and faster response time for such queries.

   Restriction

   When an outer join is executed with a USING clause, SQL generates hidden case expressions to select the non-NULL values for the USING columns. For example, the reference to EMPLOYEE_ID in the WHERE

clause references EMPLOYEE_ID from either the JOB_HISTORY or the EMPLOYEES table.

```
select count(*)
from employees full outer join job_history
    using (employee_id)
where employee_id < '00200';
```

These hidden conditional expressions currently disable the strict partitioning selection. Therefore, if you are using strict partitioning you may need to qualify the column name fully and use the ON clause to enable strict partitioning. The example query could be recoded as shown:

```
select count(*)
from employees e full outer join job_history jh
    on e.employee_id = jh.employee_id
where e.employee_id < '00200' or jh.employee_id < '00200';
```

This restriction will be documented in Oracle Rdb documentation in a future release.

---

4. Multiple table references interfere with partition selection.

   If a single query referenced a table multiple times, or multiple concurrent queries referenced the same table and sequential retrieval was used, an incorrect set of partitions could be scanned. This occurred because the various table references interfered with the correct partition selection.

   This problem has been corrected.

5. If a strictly partitioned table was defined with a storage map with no OTHERWISE clause then some queries would fail with an error, instead of executing correctly. This happened when a range query included an upper limit which was outside the maximum WITH LIMIT clause of the storage map.

```
SQL> select name
cont>  from T
cont>  where badge between 8 and 10;
Conjunct        Get     Retrieval sequentially of relation T
%RDMS-E-EXCMAPLIMIT, exceeded limit on last partition in storage map for T
```

   This query now works as expected.

```
SQL> select name
cont>  from T
cont>  where badge between 8 and 10;
Conjunct        Get     Retrieval sequentially of relation T
(partitioned scan#1)
 NAME
 i
 j
2 rows selected
SQL>
```

6. Changes to STRATEGY ("S") and EXECUTION ("E") debug flags output

   Either the SET FLAGS statement or the logical RDMS$DEBUG_FLAGS (RDB_DEBUG_FLAGS on Digital UNIX) could be used to enable the STRATEGY or EXECUTION tracing during query evaluation.

If a table is strictly partitioned and the query allows the use of this feature then the STRATEGY dump now includes the text "(partitioned scan#nn)" after the table name. The **#nn** indicates the leaf number for this sequential scan (there may be several within a single query). For instance,

```
select name
 from T
 where badge between 8 and 10;
~S#0045
Conjunct        Get     Retrieval sequentially of relation T
(partitioned scan#1)
~E#0045.1: Strict Partitioning using 2 areas
        partition 9 (larea=57)
        partition 10 (larea=58)
 NAME
 i
 j
2 rows selected
```

For this type of sequential retrieval the EXECUTION trace now includes a count and a list of the selected partitions each time the query is executed. Note that these examples use constant values for the partitioning columns, however, in an application these values are likely to vary and be supplied in host variables. Therefore, the selected partitions may change from one execution to the next.

The **~E#0045** in the example indicates an EXECUTION trace for the query numbered 0045—this corresponds to the **~S#0045** value printed by the STRATEGY dump. The **.1** indicates the leaf number and is used to associate the EXECUTION trace with a specific part of the query identified by the "(partitioned scan#1)" notation.

2.1.32 When Page Transfer via Memory was Enabled Read-Only Transactions Could Fail

Bugs 408761 and 432133.

When page transfer via memory was enabled and read-only transactions were running concurrently with read/write transactions, some read-only transactions could fail with the following error message :

```
%RDB-F-IO_ERROR, input or output error
-RDMS-F-CANTWRITEDBS, error writing pages x:x-x
-SYSTEM-F-NOPRIV, no privilege for attempted operation
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.33 Spam Pages Could be Incorrect After Recovery

After an RMU/RECOVER operation or a rollback operation when the Fast Commit option was enabled, it was possible that the SPAM page was not correctly updated to reflect a change of free space on a page.

The following example shows how this problem could be seen:

```
$RMU/RESTORE/NORECOVER MYRBF
$RMU/RECOVER/ROOT=MYDB.RDB MYAIJ.AIJ
$RMU/VERIFY/ALL MYDB.RDB
%RMU-W-PGSPMCLST, area MYAREA, page 2205
the 70% fullness value for this data page does not fall
within the 96-100% range found on the space management page
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.34 RMU/Load/Place Forces Constraint Evaluation

When the **PLACE** qualifier was used on the RMU/LOAD command, it forced constraints to be loaded and executed during loading, even if the **NOCONSTRAINTS** qualifier was used.

The **PLACE** option did a first pass through the data to calculate the probable target DBKEY for each data row when using a PLACEMENT VIA INDEX mapping. The data was then sorted in this DBKEY order to ensure a sequential updating of pages in the target area. The goal was to reduce random I/O during the load operation and reduce the load time.

The **NOCONSTRAINTS** option was only disabling constraints during the second load pass, unfortunately the first pass had already loaded and enabled the constraints.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The first pass to calculate the target DBKEY will no longer cause the constraints and triggers to be loaded and enabled.

2.1.35 Loss of Entries in Ranked Indexes with Duplicates

Bug 440823.

Rdb 7.0 sorted ranked index containing duplicates may lose duplicate entries when the index is built from existing records within a table. This means when CREATE INDEX is issued after there are records already in the table on which the index is being created.

In addition, incorrect results may be returned when the index is used for access.

RMU/ANALYZE may be used to determine if any entries have been lost after a ranked index build.

The following example shows how RMU/ANALYZE will show this error:

```
RMU-I-BTRDUPCAR, Inconsistent duplicate cardinality (C1) of 2
                 for entry 1 at dbkey 7:125:0.
                 Actual count of duplicates is 35                 .
%RMU-I-BTRERPATH, parent B-tree node of 7:125:0 is at 7:125:3
%RMU-I-BTRHEACAR, Sum of entry cardinalities given as 146   ; expected 12
%RMU-I-BTRNODDBK, Dbkey of B-tree node is 7:128:2
%RMU-I-BTRENTCAR, Inconsistent entry cardinality (C1) of 146 specified
                 for entry 13 at dbkey 7:125:3 using precision of 33.
                 Dbkey 7:128:2 at level 1 specified a cardinality of 12.
%RMU-W-IDXDATMIS, Index I_D1 does not point to a row in table D1.
                 Logical dbkey of the missing row is 95:2:3.
%RMU-W-IDXDATMIS, Index I_D1 does not point to a row in table D1.
                 Logical dbkey of the missing row is 95:3:7.
...
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.36 WAIT FOR CLOSE Clause Showed Incorrect Information and did not IMPORT Correctly

Bugs 428306 and 428308.

In Rdb 7.0, the OPEN IS AUTOMATIC (WAIT n MINUTES FOR CLOSE) clause of a CREATE or ALTER DATABASE statement did not SHOW the correct information. Specifically, a SHOW DATABASE displayed the WAIT period in seconds rather than minutes.

Additionally, the IMPORT operation did not correctly import the WAIT clause.

The following example shows a database created with the OPEN IS AUTOMATIC (WAIT 5 MINUTES FOR CLOSE) clause. A SHOW DATABASE shows the WAIT clause in minutes not seconds.

```
CREATE DATABASE FILENAME WAIT_CLAUSE
OPEN IS AUTOMATIC (WAIT 5 MINUTES FOR CLOSE);
SHOW DATABASE *
Default alias:
    Oracle Rdb database in file WAIT_CLAUSE
        Multischema mode is disabled
        Number of users:              50
.
.
 Mode is Open Automatic (Wait 300 minutes for close)
```

The following example shows the IMPORT operation does not work correctly for the WAIT CLAUSE. A SHOW DATABASE shows the WAIT CLAUSE as 0 minutes rather than 5 minutes:

```
ATT 'FILE WAIT_CLAUSE';
EXPORT DATA FILE WAIT_CLAUSE INTO WAIT_CLAUSE_TMP;
DISCONNECT ALL;
IMPORT DATA FROM WAIT_CLAUSE_TMP FILE WAIT_CLAUSE;
Exported by Oracle Rdb X7.0-00 Import/Export utility
A component of Oracle Rdb SQL X7.0-00
.
.
Open is Automatic, Wait period is 5 minutes
.
.
 Mode is Open Automatic (Wait 0 minutes for close)
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.37 Match Keys with Constant Equality Returned the Wrong Number of Rows

Bug 428374.

The wrong number of rows would be returned by a select query where the match keys had constant equality predicates. The following query should return 2000 for both columns, but returns NULL for the second column:

```
select (qty * price),
       (select sum (t2.qty * t2.price)
 from t2 t2
     where t2.f1 = t1.f1 and t2.f2 = t1.f2)
    from t1 t1
    where t1.f1 = 'K1001' AND t1.f2 = 1;

          2000                   NULL
1 row selected
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.38 RMU/SHOW STATISTIC Bugcheck Following Cancelled Sub-Menu Selection

Bug 450888.

It was possible for the **RMU/SHOW STATISTIC** utility to bugcheck when a sub-menu selection is cancelled.

Here is an example of the exception from the bugcheck:

```
Exception: Exception at 004290F0 : KUTDIS$SETUP_COMMON + 00000470

Reason   : %SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual
address=00000000, PC=0013D808, PSL=03C00000
```

The following example shows how to reproduce this problem:

```
1. select SUMMARY OBJECT STATISTICS
2. Select MENU
3. Select LOCKING ONE STAT FIELD
4. CTRL/Z from locking menu.
5. Press left arrow <-
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

Cancelling a sub-menu selection now works correctly.

2.1.39 RMU/RECOVER of Optimized AIJ Journal did not Preserve Sequence Number

The **RMU/RECOVER** utility did not properly preserve the "next" AIJ sequence number following recovery of an optimized AIJ journal. This problem occurred when an AIJ file containing multiple sequence numbers was optimized.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The **RMU/OPTIMIZE/AFTER_JOURNAL** and **RMU/RECOVER** utilities have been corrected to properly initialize the next AIJ sequence number following roll-forward completion.

2.1.40 Subquery of IN clause with Both DISTINCT and LIMIT TO Returned too few Rows

In some cases when the IN clause used a SELECT subquery the wrong number of rows were returned for the query. This occurred when both the DISTINCT and the LIMIT TO clause were applied in the subquery. This is shown in the following example:

```
SQL> attach 'file MF_PERSONNEL';
SQL>
SQL> -- first shown that the subquery should return 2 rows
SQL>
SQL> select distinct employee_id from job_history
cont> order by 1 limit to 2 rows;
 EMPLOYEE_ID
 00164
 00165
2 rows selected
SQL>
SQL> -- expect to get two rows from this query (matching
SQL> -- the two values reported above), but only get one
SQL>
SQL> select employee_id, last_name from employees where
cont> employee_id in (select distinct employee_id from job_history
cont> order by 1 limit to 2 rows);
 EMPLOYEE_ID   LAST_NAME
 00164         Toliver
1 row selected
SQL>
```

In general the IN clause does not need to perform a DISTINCT because existence in the set of returned values is sufficient for the query. Therefore, the Rdb optimizer removes the DISTINCT from these types of subqueries and so avoids a sorting step. However, when the source row set is restricted using the LIMIT TO clause then the DISTINCT is required to guarantee that the result is restricted correctly.

This problem was corrected in Oracle Rdb7. This release note was missing from the Oracle Rdb7 Release notes.

2.1.41 Database Recovery Processes Result in a Bugcheck

In certain combinations of locked and free space on a database page, a database recovery (DBR) process rollback operation could result in bugchecks in the routine DBR$RESOLVE. Because the DBR process failure results in a forced database shutdown, the workaround is to restore and recover the database.

This problem has been corrected in Oracle Rdb Version 7.0.1.

Additional information about the length of an erased row is now stored in the RUJ file in order to help the rollback operation be able to make better use of the locked space on a database page.

2.1.42 Minimum NODE SIZE for SORTED Indexes not Calculated Correctly

When UNIQUE index with a type of SORTED RANKED was created an error could be generated if specifying the minimum node size. The minimum node size can be calculated using the formula provided in the Oracle Rdb documentation.

```
SQL> create table T
cont>     (a integer,
cont>      b char(4));
SQL>
SQL> create unique index I2 on T (a)
cont>     type is SORTED RANKED
cont>     node size 101;
%RDB-E-NO_META_UPDATE, metadata update failed
-RDB-E-IMP_EXC, facility-specific limit exceeded
-RDMS-F-INDEX_S_MIN, user requested node size of 101 bytes for
index needing 104
```

In this case the specified NODE SIZE was correct but was rejected by Rdb7.

For non-UNIQUE indexes with a type of SORTED RANKED the minimum node size was not calculated correctly and so indexes could be created with a node size smaller than the minimum. This could lead to poor index performance and possible INSERT and UPDATE failures.

In addition, the NODE SIZE specified on an ALTER INDEX statement was not validated by Rdb.

This problem has been corrected in Oracle Rdb Version 7.0.1 .

2.1.43 RMU/BACKUP/AFTER_JOURNAL "File Already Exists" Failure

When a single extensible AIJ journal was created on OpenVMS using an *explicit* file version number, it was possible for subsequent AIJ backup operations to fail with a "file already exists" error.

This problem only occurred when using a single, extensible AIJ journal.

The following example shows the creation of an AIJ journal with an explicit file version number.

```
$ rmu/set after/enable/add=(name=tst, file=TST_AIJ.AIJ;1, alloc=512) tst
```

The following example shows the AIJ backup failure that can result following the creation of an AIJ journal with an explicit file version number.

```
$ rmu/back/after/log tst tst_aij.aij_bck            ! backup KO
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
    RDB$ROOT2:[70]TST.RDB;1 Event Notification
AIJ backup operation started
%RMU-I-AIJBCKSEQ, backing up after-image journal sequence number 1
%RMU-I-LOGBCKAIJ, backing up after-image journal TST at 09:44:26.66
%RMU-I-LOGCREBCK, created backup file RDB$ROOT2:[70]TST_AIJ.AIJ_BCK;2
%RMU-W-AIJDEVDIR, AIJ filename "TST_AIJ.AIJ;1" does not include a
device/directory
%RMU-I-AIJBCKSTOP, backup of after-image journal TST did not complete
%RMU-F-FILACCERR, error creating after-image journal file
_CHSR36$DKA500:[JSUBRI.70]TST_AIJ.AIJ;1
-RMS-E-FEX, file already exists, not superseded
%RMU-F-FTL_BCK, Fatal error for BACKUP operation at 19-FEB-1997 09:44:27.10
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

The AIJ backup operation correctly creates new versions of AIJ journals that were originally created with explicit file version numbers.

2.1.44 RMU/SHOW STATISTIC "Alarm Bell" Rings Excessively for Long Stalls

When a very long stall occurred, such as when an AIJ backup occurs, the **RMU/SHOW STATISTIC** utility alarm bell feature is not very useful, as it was constantly ringing.

This problem has been corrected in Oracle Rdb Version 7.0.1.

When the **RMU/SHOW STATISTIC** utility alarm bell feature is enabled, the alarm bell can be configured to ring continuously or selectively.

When in "continuous" mode, the default setting, the alarm bell rings whenever any stall exceeds the specified alarm threshold. This is equivalent to how the alarm bell previously worked.

When in "selective" mode, the alarm bell only rings when a *new* stall becomes the oldest stall to exceed the specified alarm threshold. Therefore, in the case of the long-running AIJ backup stall, the alarm bell will only ring once until the stall terminates and a new stall exceeds the alarm threshold.

The **RMU/SHOW STATISTIC** utility also now allows you to select one or more "classes" of stalls which control how the "Alarm Bell" facility and the "Alarm Notification" facility operate.

Currently, there are ten stall classes from which to choose:

1. Record Stalls
2. Page Stalls
3. Table Stalls
4. Storage Area Stalls
5. Database File Stalls
6. AIJ/RUJ Journal Stalls
7. Transaction Stalls
8. Hot Standby Stalls
9. Database Stalls
10. Miscellaneous Stalls

By default, all stall classes are selected.

Stall classes can be selected using the "Tools Menu", which is obtained using the "!" key. When either the "Alarm Bell" facility or the "Alarm Notification" facility is enabled, the "Tools Menu" will contain an "Alarm Classes" sub-menu option.

Selecting the "Alarm Classes" option will display a sub-menu containing all of the available stall classes. Stall classes that have been selected are identified by a "->" designation. Stall classes that have been de-selected do not have this designation.

To de-select a previously selected stall class, or to select a previously de-selected stall class, simply choose it from the sub-menu. The "Alarm Classes" sub-menu will continue to be displayed until you either type "CTRL-Z" to terminate it, or de-select all but the last stall class.

Note

You cannot de-select all stall classes. When you have de-selected all but one stall class, the "Alarm Classes" sub-menu will automatically cancel itself.

2.1.45 Floating Divide by Zero in Query with Constant Expression

Bug 450934.

The following query should select one record of employee with ID of '00169' but it generates a floating divide by zero error.

```
SELECT * FROM employees
 WHERE (
        EMPLOYEE_ID = '00169'
        AND (
             (
              1 = 1
             ) OR
             (
              1 <> 2
              AND LAST_NAME = 'Toliver'
             )
            )
       );
```

The problem was caused by the optimizer code where it computes the duplicity factor of the table by dividing the table cardinality by the distinct cardinality. The distinct cardinality becomes zero when the selectivity of the constant expression defaults to zero at the query compilation time, and thus generates the divide by zero error.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.46 Memory Leak with Sort when Cursor Closed Before end of Stream

Bug 420945.

Previously, if a cursor was closed before it had reached the end of stream and a sort occurred during its execution, sixteen byte chunks of virtual memory were not properly released.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.47 TRANSLATE() Function Returns Corrupted Data Instead of White Space

Bug 423056.

When using DEC_Kanji to DEC_MCS conversion, the Translate() function would return corrupted characters. This problem occurred if you passed an untranslatable character to the translate() function (ie:Japanese punctuation characters). This should return a single-octet of white space code instead.

The following is an example of the "bad" output.

```
SQL> select _dec_kanji'¡¦¡«¡¬',
cont>translate(_dec_kanji'¡¦¡«¡¬' using rdb$dec_mcs)
cont>from dual;

 ¡¦¡«¡¬   ¥Þß
1 row selected
```

The following demonstrates what the code now returns.

```
SQL> select _dec_kanji'¡¦¡«¡¬',
cont>translate(_dec_kanji'¡¦¡«¡¬' using rdb$dec_mcs)
cont>from dual;

 ¡¦¡«¡¬
1 row selected
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.48 Error During Remote Replication

Bug 441282.

An ACCVIO error could occur while performing a remote replication with the Replication Option for Oracle Rdb when the replicator detached from the database. The problem was only observed on ALPHA but could potentially happen on VAX as well.

Here is an example of the problem:

```
SQL> create trans_decnet type is replication move table emp
 to EXISTING filename chsr36"user pass"::W2:[JSUBRI.70]DDAL70.RDB
 log 'W2:[JSUBRI.70]DDAL70.LOG_decnet';
SQL> start trans trans_decnet now;
15:10:59  %DDAL-I-LOATABDAT, loading data for table EMP
15:10:59  %DDAL-I-TAB_STATS,                  TABLE  EMP
          %DDAL-I-ROWSFMSRC,         14 rows copied from RDB$ROOT2:[70]MF_PERSO
          %DDAL-I-ROWSTOTGT,         14 rows copied into chsr36"proxy"::W2:[JSU
15:10:59  %DDAL-I-CREATEIDX, creating index DDAL$DBKEY_INDEX1_2
15:10:59  %DDAL-I-LOATABDAT, loading data for table RDB$VINTAGE
----- 14-JAN-1997 15:11:00.20 ----- Error           ------------------------
%SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual
 address=0000000000000000, PC=0000000000000000, PS=0000001B
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.49 Row Cache Screen Shows Extraneous Characters Where Cache Name Exceeds 21 Characters

Bug 444581.

In the row cache (insert cache) RMU/SHOW STATISTICS screen (option U followed by G), where a cache name exceeded 21 characters, the last 3 characters of a cache name remained on the screen when the caches were paged through using the up and down arrow keys.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.50 Processes Stalling for Page Locks when ABW is Enabled

Bugs 460533, 460062, 442725, and 449685.

Processes could stall for extended periods, waiting for a page lock to be released by another process. The page lock would eventually be released when the blocking process voluntarily gave up that page: typically when finishing its transaction, or when reusing the buffer containing that page.

However, if the database operated in "fast commit" mode, the blocking process might not be able to give up the page, because it was waiting for user input or for other work to do.

Since the problem only happened for pages that were written by the "asynchronous batch write" mechanism, a temporary workaround was to disable this mechanism. This was achieved using the following syntax:

```
SQL> alter database filename my_database
cont> asynchronous batch writes are disabled;
```

This problem was introduced in Rdb7 and has been corrected in Oracle Rdb Version 7.0.1.

2.1.51 Database Hang When Using Multiple Attaches to Same Database

When using a large number of attaches to a database by the same process, or attaching to a large number of databases at the same time, it was possible to cause the database(s) to hang.

The problem was caused by various timing conditions, and only seemed to occur when using the AIJ Log Server ("ALS") server or when using the "Record Cache" feature.

The problem could occur when attaching multiple times to the same database, or attaching to multiple databases.

The problem did *NOT* occur when using only a single attach to the database. Also, because the problem was timing related, the problem seemed to occur when using three or more attaches, but the problem could occur when using at least two attaches.

This problem is fixed in Oracle Rdb Version 7.0.1.

2.1.52 Excess Query Compilation Time in ALTER TABLE with Complex COMPUTED BY Column

Excessive query compile time was observed when complex COMPUTED BY columns were added to a table using the ALTER TABLE command.

The problem was caused by DDL ALTER command which failed to mark the sub-query as "sub-select" within the computed by field. That, in turn, caused the optimizer to iterate through each join item, rather than start only from this sub-select query, and thus, increased the query compile time. The following example shows the problem:

```
create table T1 (
    T1_NUM INTEGER,
    T1_CODE CHAR(6));

create table T2 (
    T2_NUM INTEGER,
    T2_CODE INTEGER);

insert into t1 value (1, 'abcdef');
insert into t1 value (2, 'abcdef');
insert into t1 value (3, 'abcdef');
insert into t1 value (4, 'abcdef');
insert into t1 value (5, 'abcdef');

insert into t2 value (1, 1);
insert into t2 value (2, 1);
insert into t2 value (3, 1);
insert into t2 value (4, 1);
insert into t2 value (5, 1);
insert into t2 value (6, 1);
```

```
! now execute the  alter table command here
alter table T2
add column T2_NEW_COL
computed by
case
when
    ((not exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE
    ) and
                 (select count(*) from T2 C1
                  where C1.T2_CODE=T2.T2_CODE
    ) =
                 (select count(*) from T2 C1
                  where C1.T2_CODE=T2.T2_CODE
    and
                        C1.T2_CODE<>C1.T2_CODE
    ))
     or
     (
 not exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE)
    and
         exists (select 'Found' from T1
                  where T1_NUM=T2.T2_NUM)
    )
   )
    then 'J'
when
    ((exists (select 'Found' from T1
              where T1_NUM=T2.T2_CODE
       ))
     or
     (not exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE
    ) and
          exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE
    ))
     or
     (not exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE
    ) and
      not exists (select 'Found' from T1
                  where T1_NUM=T2.T2_CODE
    ) and
          exists (select 'Found' from T1
                        where T1_NUM=T2.T2_NUM
         )))
    then 'J'
else
    'N'
end;
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.53 Alpha Bugcheck Stack Dumps not Symbolized Correctly

OpenVMS Alpha platform.

If Oracle Rdb images were installed using the INSTALL utility /RESIDENT qualifier, Oracle Rdb bugcheck dump files might not contain the correct routine name symbols in the stack dump portion of the dump file.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.54 Database Cannot be Recovered Using RMU/BACKUP/AFTER_JOURNAL/RENAME Backup Files

The AIJ backup files created by the **RMU/BACKUP/AFTER_JOURNAL /RENAME** utility could not be used to recover the database. The sequence number information in the created backup files could not be used to recover the database in some cases.

The problem could be detected by dumping the header of the "new" AIJ journal following an AIJ backup using the **/RENAME** qualifier.

```
ALL> rmu/backup/after/rename/log mf_personnel ""
%RMU-W-DATACMIT, unjournaled changes made; database may not be recoverable
%RMU-I-AIJBCKBEG, beginning after-image journal backup operation
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
KODH$:[R_AN
DERSON.WORK.ALS]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation started

%RMU-I-AIJBCKSEQ, backing up after-image journal sequence number 0
%RMU-I-LOGBCKAIJ, backing up after-image journal RICK1 at 15:35:23.14
%RMU-I-QUIETPT, waiting for database quiet point
%RMU-W-AIJDEVDIR, AIJ filename "RICK1.AIJ" does not include a
device/directory
%RMU-I-LOGCREAIJ, created after-image journal file
_$111$DUA347:[R_ANDERSON.WORK.ALS]RICK1.AIJ;2
%RMU-I-OPERNOTIFY, system operator notification: Oracle Rdb Database
KODH$:[R_ANDERSON.WORK.ALS]MF_PERSONNEL.RDB;1 Event Notification
AIJ backup operation completed

%RMU-I-AIJBCKEND, after-image journal backup operation completed
successfully
%RMU-I-LOGAIJJRN, backed up 1 after-image journal at 15:35:24.85
%RMU-I-LOGAIJBLK, backed up 127 after-image journal blocks at 15:35:24.85
ALL> rmu/dump/after/end=1 rick1.aij;2
*-------------------------------------------------------------------------------
* Oracle Rdb X7.0-00                                        24-MAR-1997
15:35:42.87
*
* Dump of After Image Journal
*     Filename: KODH$:[R_ANDERSON.WORK.ALS]RICK1.AIJ;2
*
*-------------------------------------------------------------------------------

1/1             TYPE=O, LENGTH=510, TAD=24-MAR-1997 15:35:24.57, CSM=00
Database _$111$DUA347:[R_ANDERSON.WORK.ALS]MF_PERSONNEL.RDB;1
    Database timestamp is  1-DEC-1992 12:46:08.86
    Facility is "RDMSAIJ ", Version is 701.0
    AIJ Sequence Number is -1
    Last Commit TSN is 0:0
    Synchronization TSN is 0:0
    Journal created on VMS platform
    Type is Normal (unoptimized)
    Open mode is Initial
    Backup type is Latent
    I/O format is Record
    Commit-to-Journal
    optimization enabled
```

Note the "AIJ Sequence Number" was stored as "-1". This was incorrect and causes the AIJ journal to not be valid as the starting AIJ journal of an AIJ recovery operation.

Also notice that the "Last Commit TSN" is "0:0", which is an invalid TSN.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The **/RENAME** operation now creates a backup file that can be used to correctly recover a database.

Also the /RENAME qualifier can now be specified to the RMU/BACKUP/AFTER_JOURNAL utility when using extensible journals. Previously, the /RENAME qualifier was only valid when using fixed-size AIJ journals.

2.1.55 Application and Oracle Rdb Both Using SYS$HIBER

All OpenVMS Platforms.

In application processes that use Oracle Rdb and the $HIBER system service (possibly via RTL routines such as LIB$WAIT), it is important that the application ensures that the event being waited for has actually occurred. Oracle Rdb uses $HIBER/$WAKE sequences for interprocess communications particularly when the ALS (AIJ Log Server) feature is enabled.

Oracle Rdb's use of the $WAKE system service can interfere with other users of $HIBER (such as the routine LIB$WAIT) that do not check for event completion, possibly causing a $HIBER to be unexpectedly resumed without waiting at all.

To avoid these situations, consider altering the application to use a code sequence that avoids continuing without a check for the operation (such as a delay or a timer firing) being complete.

The following pseudo-code shows one example of how a flag can be used to indicate that a timed-wait has completed correctly. The wait does no complete until the timer has actually fired and set TIMER_FLAG to TRUE. This code relies on ASTs being enabled.

```
ROUTINE TIMER_WAIT:
    BEGIN
    ! Clear the timer flag
    TIMER_FLAG = FALSE

    ! Schedule an AST for sometime in the future
    STAT = SYS$SETIMR (TIMADR = DELTATIME, ASTRTN = TIMER_AST)
    IF STAT <> SS$_NORMAL
    THEN BEGIN
         LIB$SIGNAL (STAT)
         END

    ! Hibernate.  When the $HIBER completes, check to make
    ! sure that TIMER_FLAG is set indicating that the wait
    ! has finished.
    WHILE TIMER_FLAG = FALSE
    DO  BEGIN
        SYS$HIBER()
        END
    END

ROUTINE TIMER_AST:
    BEGIN
    ! Set the flag indicating that the timer has expired
    TIMER_FLAG = TRUE

    ! Wake the main-line code
    STAT = SYS$WAKE ()
    IF STAT <> SS$_NORMAL
    THEN BEGIN
         LIB$SIGNAL (STAT)
         END
    END
```

In OpenVMS V7.1, the LIB$WAIT routine has been enhanced via the FLAGS argument (with the LIB$K_NOWAKE flag set) to allow an alternate wait scheme (using the $SYNCH system service) that can avoid potential problems with multiple code sequences using the $HIBER system service.

2.1.56 Additional Bugcheck Information Added

Certain errors detected by Oracle Rdb when reading pages from the database result bugcheck dump files being created. The actual cause of the error was not always immediately obvious from the exception location and it was difficult to determine the root cause of the bugcheck.

For example, a bugcheck exception at PIO$FETCH_RET + 04F4 might mean that a database page stored on disk was corrupt and that this corruption was detected when the page was read from disk and the page number stored on the page did not match the page number that was requested. This information could be difficult to determine from the information in the bugcheck file.

Oracle Rdb now includes additional informational text in the bugcheck dump exception to identify the detected problem more easily. Two of the more common bugcheck conditions now include this additional information. This is similar to the information contained when a CHECKSUM exception occurs indicating a page checksum error.

When a page is read from the database, the page returned by the operating system is checked to ensure that it was the page requested. In particular each page contains a page number and a storage area number. If these do not match the page number and storage area number of the expected page, the exception BADPAGRED is listed in the bugcheck dump file. The following example shows this exception:

```
***** Exception at 00A7292A : PIO$FETCH_UPD + 0580
RDMS-F-CANTREADDBS, error reading pages 4:184-184
RDMS-F-PADPAGRED, read requesting physical page 4:184 returned
                  page 8:821
```

When a request is made to read a page from the database, the page number of the request is checked to make sure that it is not larger than the number of pages in the specified storage area. If an invalid DBKEY is specifies a bad page number, and the page number is greater than the number of pages in the storage area, the exception BADPAGNUM is listed in the bugcheck dump file. The following example shows this exception:

```
***** Exception at 00572320 : PIOFETCH$MAKE_PIB_ENT_VALID + 0074
RDMS-F-CANTREADDBS, error reading pages 4:1234567-1234567
RDMS-F-BADPAGNUM, page 1234567 is greater than the maximum page
                  number 58974 for physical area 4
```

2.1.57 Bugchecks at AIJUTL$SWITCH_FILE + 1430

If circular journaling was enabled and all journals became full any process that attempted to write to the current journal would bugcheck. The bugcheck dump would show the following exception:

```
***** Exception at 000AD068 : AIJUTL$SWITCH_FILE + 00001430
%SYSTEM-F-ACCVIO, access violation, reason mask=00,
    virtual address={some number}, PC={some number}, PS=0000001B
```

This problem was introduced in Oracle Rdb Version 7.0 ECO 1 but it does not exist in the original release of Oracle Rdb Version 7.0.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.58 Errors Corrected for RMU/EXTRACT

Bugs 427336 and 365883.

The following errors have been corrected in RMU/EXTRACT for this release:

- In prior versions RMU/EXTRACT may exhaust locks when extracting using the qualifiers /ITEM=VOLUME and /OPTION=VOLUME_SCAN and the database has snapshots disabled.

  This is because RMU/EXTRACT executes a series of "SELECT COUNT(*) FROM table" queries to extract the volume information. With snapshots disabled Oracle Rdb automatically changes the read-only transaction started by RMU/EXTRACT to a read/write transaction which means that row level locks are accumulated for many of the count operations.

  With this release of Rdb, RMU/EXTRACT will start a read/write ISOLATION LEVEL READ COMMITTED transaction if it detects that snapshots are disabled. This type of transaction will allow row read locks to be released once the row is read.

- The ALTER OPERATOR syntax for ALTER DATABASE extracted the option CLUSTER incorrectly. This has been corrected.

- The COMMENT IS clause for query outlines was not extracted by prior versions. This has been corrected for this release.

  The comments are displayed by the SQL SHOW OUTLINE command, so they can be edited into the script manually.

2.1.59 Privilege Check Relaxed for TRUNCATE TABLE

In prior releases of Rdb7 the user required the following privileges to execute TRUNCATE TABLE:

- DELETE on the table to allow the rows to be deleted,
- CREATE on the database,
- and DROP on the table

Effective with this release, the privilege requirements for TRUNCATE TABLE have been relaxed:

- DELETE on the table to allow the rows to be deleted,
- If there exists a BEFORE DELETE or an AFTER DELETE trigger for this table then the user is also required to have
  - CREATE on the table
  - and DROP on the table

  These privileges are required because these types of triggers are not executed during TRUNCATE TABLE. Therefore, to bypass the execution the user requires the same privileges that would be used to DROP and re-CREATE the trigger definitions.

These changes are available in Oracle Rdb 7.0.1.

2.1.60 RDO: GFLDNOEX Message Incorrectly Displays Column name Instead of Domain Name

Bug 482090.

This RDO script shows that when using an unknown DOMAIN name the diagnostic incorrectly displayed the name of the COLUMN.

```
RDO> data file mf_personnel
RDO> sho fields for jobs
 Fields for relation  JOBS
   Description:        information about different kinds of jobs
     JOB_CODE                          text size is  4
       based on global field  JOB_CODE_DOM
   Constraint:          JOBS_PRIMARY_JOB_CODE
                                JOBS.JOB_CODE PRIMARY KEY
     WAGE_CLASS                        text size is  1
       based on global field  WAGE_CLASS_DOM
     JOB_TITLE                         text size is  20
       based on global field  JOB_TITLE_DOM
     MINIMUM_SALARY                    signed longword scale  -2
       based on global field  SALARY_DOM
     MAXIMUM_SALARY                    signed longword scale  -2
       based on global field  SALARY_DOM
RDO> change relation jobs.
cont> change job_title based on unknownglobal.
cont> end.
%RDO-W-NOCDDUPDAT, database invoked by filename, the CDD will not be updated
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-GFLDNOEX, there is not a global field named JOB_TITLE in this database
```

This has been corrected in Oracle Rdb 7.0.1 to use the name of the DOMAIN instead.

```
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-GFLDNOEX, there is not a global field named UNKNOWNGLOBAL in this
database
```

2.1.61 RMU/RESTORE/TRANSACTION_MODE Missing a Value

In Rdb7 a /TRANSACTION_MODE qualifier was added to the RMU/RESTORE command. This is documented only in the Rdb7 Release Notes. The description there, as well as the implimentation, omitted an additional value for the qualifier, READ_WRITE. If this was specified, the result would be a database for which no access at all was permitted.

The following example shows an attempt to restore a database specifying that read/write transactions are to be allowed:

```
RMU/RESTORE/NOCDD/TRANSACTION_MODE=READ_WRITE BKUPFILE.RBF
```

The result would be a database on which no transactions are allowed at all.

This problem has been corrected in Oracle Rdb 7.0.1.

2.1.62 Nested Logical Names in Database Names

All OpenVMS platforms.

Bug 451582.

When two logical names were used to construct the complete database name, one inside the other, the inner logical name was not being expanded, resulting in BAD_DB_FORMAT errors. This often showed up when connecting to remote databases, but was not restricted to that case.

The following example shows two logical names, DBPLACE containing a node name and directory specification, and another, DBREF which uses the first one.

```
$ DEFINE DBPLACE NODE27::DISKB:[MARKETING]
$ DEFINE DBREF   DBPLACE:MREPORTS
```

This problem has been corrected in Oracle Rdb 7.0.1.

2.1.63 IMPORT Failed to Import Some Tables with COMPUTED BY Columns

Bug 481887.

In prior releases of Rdb the IMPORT command would fail for a table if it contained a COMPUTED BY column which referenced another table and this column was also referenced by other COMPUTED BY columns for that table. For example:

```
create table A
    (a integer);
create table B
    (b integer,
     c computed by (select count(*)
                    from A)
     d computed by c * 10);
```

When a COMPUTED BY column references a table, this COMPUTED BY column is normally deferred until all other tables are defined. Then it is added to the table using the equivalent of an ALTER TABLE ... ADD COLUMN statement. This ensures that all tables needed for these COMPUTED BY columns exist.

However, when the column is deferred, any column which references it also fails. This is shown in this example:

```
SQL> import database from SAMPLE_DB filename DB$:SAMPLE_DB;
Exported by Oracle Rdb V7.0-00 Import/Export utility
A component of Oracle Rdb SQL V7.0-00
Previous name was DB$:SAMPLE_DB
It was logically exported on  7-MAY-1997 20:42
...import output omitted...
IMPORTing table A
IMPORTing table B
%SQL-F-NORELRES, unable to import table B
%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 - C
%RDB-E-OBSOLETE_METADA, request references metadata objects that no longer exist
-RDMS-F-TABNOTDEF, relation B is not defined in database
%SQL-E-NOCMPBYRES, computed field C in relation B not imported
%RDB-E-NO_META_UPDATE, metadata update failed
-RDB-E-OBSOLETE_METADA, request references metadata objects that no longer exist
-RDMS-F-TABNOTDEF, relation B is not defined in database
```

In this example the column C was a COMPUTED BY column deferred for later processing.

In Oracle Rdb 7.0.1 COMPUTED BY columns are not deferred for later processing if all the referenced tables already exist.

However, please note that it is possible to use ALTER TABLE to define complex relationships between tables which can not be resolved during the IMPORT. IMPORT processes the tables in a serial manner and so assumes that the tables and all their columns were defined in that same order. The order is defined by the RDB$RELATION_ID column in the RDB$RELATIONS system table. The following table setup is not currently supported by IMPORT.

```
create table A (a integer);
create table B (b integer);
alter table A
    add column c computed by (select count(*)
                              from B)
    add column d computed by c * 10;
```

IMPORT tries to create table A first, and since the computed column C references table B which does not yet exist it will be deferred until after all the tables are created. However, IMPORT does not detect that column D is also indirectly dependent on table B. Therefore, the import of table A will fail.

The workaround to this problem is to DROP the columns affected (such as D) before the EXPORT is performed. Then use ALTER TABLE to add them to the imported database.

2.1.64 Monitor Bugchecks

The Oracle Rdb database monitor process would sometimes bugcheck with the following exception:

```
***** Exception at 0001900D : COSI$IO_PUT_OUTPUT + 0000018A
%COSI-F-FILACCERR, error formatting file
-COSI-F-BADPARAM, bad parameter value
```

The bugchecks would repeat over and over until the system disk was full or the monitor exceeded its stack space allocation.

This bugcheck would occur after an RMU/MONITOR REOPEN_LOG command was issued and only if there was a database user on the system that had set their process name to include an exclamation point "!". The database monitor had incorrectly interpreted the exclamation point as a message formatting directive. This problem would then be exacerbated by a loop condition in the exception handling for the error that caused the bugcheck to be written over and over again.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.65 Incorrect AIJ Data Following Transaction Rollback

Following a transaction rollback, during which buffers were modified but not flushed to disk, it was possible for AIJ records to be written for these modified buffers in subsequent transactions.

This problem *only* occurred when using the "AIJ Fast Commit" feature. When the **RDM$BIND_SNAP_QUIET_POINT** logical was defined to the value "0", the writing of the AIJ records could occur when starting the *next* transaction. Furthermore, if prestarted transactions were disabled (either by using the **RDMS$BIND_PRESTART_TXN** logical was defined to the value "0", or using the PRESTARTED TRANSACTION ARE OFF syntax in the SQL, RDBPRE or RDML application), the TSN recorded for these incorrect AIJ records could be "TSN=0:0".

This problem caused potential corruption of recovered databases. In all cases, the production database was correct and consistent.

The following shows an example of the incorrect AIJ record. The incorrect AIJ data was always recorded in "TYPE=D" AIJ records. Also, these incorrect records always occur following a commit record ("TYPE=C").

```
1680/3834        TYPE=C, LENGTH=14, TAD= 1-MAY-1997 08:32:00.95, CSM=00
    TID=6454, TSN=0:34758, AIJBL_START_FLG=1

1681/3835        TYPE=G, LENGTH=16, TAD= 1-MAY-1997 08:32:01.28, CSM=4A
    Group commit date is  1-MAY-1997 08:32:01.28
    Message sequence number is 48226
    Monitor ID is 1

1681/3836        TYPE=D, LENGTH=162, TAD= 1-MAY-1997 08:32:01.28, CSM=00
    TID=6454, TSN=0:0, AIJBL_START_FLG=1
 MODIFY: PDBK=1:1909:1, LDBID=5, PSN=2772, ABM_PNO=14,
 REC_LEN=136, LENGTH=116
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.66 CREATE INDEX on a Table that is Vertically Partitioned will Optimize the Number of Partitions to be Utilized During Index Creation.

When initially released in V7.0 vertical partitioning was not utilized during CREATE INDEX. When building an index all partitions of the table were used during index creation. In 7.0.1, Rdb will fetch only from the partitions necessary to retrieve the necessary key information.

2.1.67 AIJ Backup Could Fail when Opening Database Using Record Cache

When using the RMU/BACKUP/AFTER_JOURNAL on a database with record caching enabled, for which the record cache default cache file directory is invalid, and the RMU/BACKUP/AFTER_JOURNAL command is the first utility to open the database, the AIJ backup operation could fail with the message:

```
%RMU-F-AIJJRNBSY, journal XXX is busy and cannot be backed up
```

This problem only occurred when the record cache server could not be successfully started. Manually opening the database using the **RMU/OPEN** command was a good method to determine when this problem exists.

This problem has been corrected in Oracle Rdb 7.0.1.

The RMU/BACKUP/AFTER_JOURNAL will now correctly report that the problem lies with the record cache server, not the AIJ journal.

2.1.68 Possible Bugcheck when Using Replication Option

Bug 492613.

The following bugcheck would occur during INSERT, UPDATE or DELETE operations if the database was configured to use the Replication Option for Rdb with Oracle Rdb7.

```
***** Exception at 00F047B0 : DDAL$$WCF_MARK_ESTABLISH + 00000790
%SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual address=00000003,
PC=00F047B0, PS=0000000B
```

This only occurred if the logical DDAL$CHANGES_INDEX was defined during configuration of the target database. This logical name caused two indexes to be defined on the Replication Option table RDB$CHANGES, namely RDB$CHANGES_INDEX1 and RDB$CHANGES_INDEX2.

This problem is corrected in Oracle Rdb Version 7.0.1.

After this version is installed the indexes on RDB$CHANGES can be redefined.

2.1.69 Monitor Processing of OPCOM Messages

All OpenVMS platforms.

When an Oracle Rdb process needs to send an OPCOM message to the system operator (the notification of an AIJ switch, for example), the message may have to be queued for a delayed action by the sending process. This is the result of a restriction while calling the SYS$SNDOPR system service from executive mode while on an alternate stack.

To avoid the processing required to queue and later send the message when running in executive mode, processes now send the message to the Rdb monitor process (RDMMON) that will, in turn, call the SYS$SNDOPR system service to perform the operator notification. When in user mode, processes will call the SYS$SNDOPR system service directly (ie, the Rdb monitor process will not be requested to send the message).

When the Rdb monitor process is requested to perform the operator notification, it writes the message text to the monitor log file (RDMMON.LOG) as well as sending it to the system operator.

2.1.70 Storage Areas Extended Unnecessarily

It was possible for storage areas to sometimes extend even though there were plenty of pages with space available in the storage area. Sometimes pages would be marked full in the space area management (SPAM) pages but the pages would only contain locked line entries.

This problem was caused by an error in the SPAM calculations that would sometimes result in a page being erroneously marked as full when it was not.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.71 Nested Correlated Sub-query Outer References Incorrect

Outer references from aggregation sub-queries 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 nth-1 row data (usually NULL for row 1) when it should have been using the nth row data.

This problem has existed in various forms for all prior versions, but only appears in versions 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';
select * from manifest where voyage_num = 4904 or
                             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
begin
for :a as each row of
 select * from voyage v where v.ship_name = 'SANDRA C.' or
                              v.ship_name = 'DAFFODIL' do
  for :b as each row of table cursor modcur1 for
   select * from  manifest m where m.voyage_num = :a.voyage_num do
    update manifest
     set tonnage = (select (avg (m1.exp_num) *3) from manifest m1
                    where m1.voyage_num = :a.voyage_num)
     where current of modcur1;
  end for;
end for;
end;
select * from manifest where voyage_num = 4904 or
                             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 what is actually calculated is AVG (NULL + NULL) *3 = NULL. And the TONNAGE value for VOYAGE_NUM 4909 (outer query row 2) is actually the TONNAGE value for outer query row 1.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.72 SELECT ORDER BY from a View Returned Rows in the Wrong Order

Bug 476129.

The SELECT query using the ORDER BY clause from a view query with a DISTINCT clause returned rows in the wrong order.

```
! The following query returns rows in the wrong order:
select organ_id,op_last_name from v_orgpers order by organ_id,op_last_name;
     ORGAN_ID     OP_LAST_NAME
     333          Adamson
     409          Barnfield
     327          Johnson
     385          Smithfield
     392          Ziegler
     386          NULL
     428          NULL
    7 rows selected
```

The results are sorted by OP_LAST_NAME rather than ORGAN_ID. The SELECT statement used in the view with the ORDER BY added does work by itself.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.73 Problems Corrected for Query Outlines

Bug 392827.

The following corrections and enhancements have been made query outlines for this release of Rdb.

2.1.73.1 Change in Query Ordering for Multistatement Procedures

Prior versions of Rdb generated incorrect query outlines for multistatement procedures (this includes stored procedures, stored functions and anonymous compound statements) which contained nested queries. For example, the following procedure contains a FOR loop and an UPDATE statement nested within an outer FOR loop.

```
SQL> begin
cont> -- Find the employee and
cont> -- complete there current job, before being promoted
cont> for :cur as each row of cursor a
cont>     for select last_name
cont>         from EMPLOYEES
cont>         where employee_id = :emp_id
cont> do
cont>     begin
cont>     -- Display some details
cont>     trace 'Employee: ', :cur.last_name;
cont>
cont>     for :cur2 as each row of cursor b
cont>         for select cast(job_start as date ansi) as js,
cont>                    cast(job_end as date ansi) as je
cont>             from JOB_HISTORY
cont>             where employee_id = :emp_id
cont>             order by job_start
cont>     do
cont>         trace ' Job Duration: ',
cont>            (coalesce (:cur2.je, current_date) - :cur2.js) year to month;
cont>     end for;
cont>
cont>     -- Now complete the current job
cont>     update JOB_HISTORY
cont>         set job_end = cast(current_date as date vms)
cont>         where employee_id = :emp_id;
cont>
cont>     end;
cont> end for;
cont> end;
-- DEC Rdb Generated Outline : 29-MAY-1997 22:52
create outline QO_39BBA6C4E902AB2B_00000000
id '39BBA6C4E902AB2B6A252A71A1CFFB71'
mode 0
as (
  query (
-- For loop
    subquery (
      EMPLOYEES 0     access path index       EMPLOYEES_HASH
      )
    )
```

```
  query (
-- For loop
    subquery (
      JOB_HISTORY 0   access path index       JOB_HISTORY_HASH
      )
    )
  query (
-- Update
    subquery (
      JOB_HISTORY 0   access path index       JOB_HISTORY_HASH
      )
    )
  )
compliance optional     ;
```

The order of the queries in the query outline represents a flattened tree structure that represents the complex execution profile of the compound statement. When extracting this tree structure Rdb was generating an order related to the top down structure of the procedure, however, the optimizer requires a bottom up representation of the optimization phase.

This means that query outlines generated for any procedure with nested statements can not be used by the query optimizer. A work around is to edit the generated query outline and move the query within the outline so that the outer loop (in this example on the table EMPLOYEES) appears last in the outline instead of first.

```
-- DEC Rdb Generated Outline : 29-MAY-1997 22:52
create outline QO_39BBA6C4E902AB2B_00000000
id '39BBA6C4E902AB2B6A252A71A1CFFB71'
mode 0
as (
  query (
-- For loop
    subquery (
      JOB_HISTORY 0   access path index       JOB_HISTORY_HASH
      )
    )
  query (
-- Update
    subquery (
      JOB_HISTORY 0   access path index       JOB_HISTORY_HASH
      )
    )
  query (
-- For loop
    subquery (
      EMPLOYEES 0     access path index       EMPLOYEES_HASH
      )
    )
  )
compliance optional     ;
```

This problem has been corrected in Oracle Rdb Version 7.0.1. Query outlines now correctly represents the optimizer query tree for complex procedures.

2.1.73.2 TRACE Statement Containing Subqueries

If the TRACE statement is activated by the RDMS$DEBUG_FLAGS "Xt" (or by the SET FLAGS statement in Rdb7) then queries in the TRACE statement will be merged into the query outline for the procedure. For instance, the following query outline contains one query when the TRACE statement is disabled.

```
SQL> declare :ln char(40);
SQL>
SQL> begin
cont> trace 'Jobs Held: ',
cont>     (select count(*)
cont>        from job_history
cont>        where employee_id = '00201');
cont> select last_name
cont>     into :ln
cont>     from employees
cont>     where employee_id = '00201';
cont> end;
-- DEC Rdb Generated Outline : 28-MAY-1997 16:48
create outline QO_A17FA4B41EF1A68B_00000000
id 'A17FA4B41EF1A68B966C1A0B083BFDD4'
mode 0
as (
  query (
-- Select
    subquery (
      EMPLOYEES 0     access path index       EMPLOYEES_HASH
      )
    )
  )
compliance optional     ;
SQL>
```

If the query outline is generated with TRACE enabled, then two queries appear; the first is for the subquery in the TRACE statement and the other is for the singleton SELECT statement.

```
SQL> declare :ln char(40);
SQL>
SQL> begin
cont> trace 'Jobs Held: ',
cont>     (select count(*)
cont>        from job_history
cont>        where employee_id = '00201');
cont> select last_name
cont>     into :ln
cont>     from employees
cont>     where employee_id = '00201';
cont> end;
-- DEC Rdb Generated Outline : 28-MAY-1997 16:48
create outline QO_A17FA4B41EF1A68B_00000000
id 'A17FA4B41EF1A68B966C1A0B083BFDD4'
mode 0
as (
  query (
-- Trace
    subquery (
      JOB_HISTORY 0   access path index       JOB_HISTORY_HASH
      )
    )
  query (
-- Select
    subquery (
      EMPLOYEES 0     access path index       EMPLOYEES_HASH
```

```
      )
    )
  )
compliance optional     ;
~Xt: Jobs Held: 4
SQL>
```

If this second query outline is used at runtime with the TRACE statement disabled then it can not be not be applied to the query as shown below.

```
SQL> declare :ln char(40);
SQL>
SQL> begin
cont> trace 'Jobs Held: ',
cont>     (select count(*)
cont>         from job_history
cont>         where employee_id = '00201');
cont> select last_name
cont>     into :ln
cont>     from employees
cont>     where employee_id = '00201';
cont> end;
~S: Outline QO_A17FA4B41EF1A68B_00000000 used
~S: Outline/query mismatch; assuming JOB_HISTORY 0 renamed to EMPLOYEES 0
~S: Full compliance with the outline was not possible
Get     Retrieval by index of relation EMPLOYEES
  Index name  EMPLOYEES_HASH [1:1]       Direct lookup
```

The outline was created with compliance optional, so in this case the query outline is abandoned and a new strategy is calculated. If compliance was mandatory, then the query will fail.

If any TRACE statement contains a subquery, then Oracle recommends using two query outlines (if any are required at all), with different modes in order to run the query with and without TRACE enabled. That is, when TRACE is enabled define RDMS$BIND_OUTLINE_MODE to match the TRACE enabled query outlines.

```
$ DEFINE RDMS$DEBUG_FLAGS "Xt"
$ DEFINE RDMS$DEBUG_FLAGS_OUTPUT TRACE.DAT
$ DEFINE RDMS$BIND_OUTLINE_MODE 10
```

Alternatively, use the SET FLAGS statement (as described in Section 5.1.24, New MODE Keyword Added to SET FLAGS Statement) which allows the TRACE flag to be enabled and the MODE established from within an interactive session, or via dynamic SQL.

This scheme will allow the query to be run with TRACE enabled or disabled.

2.1.74 Restriction Lifted for CAST Numeric Value as INTERVAL

Bug 496702.

Various restrictions existed in prior versions of Rdb when using CAST to convert a numeric value to an INTERVAL For example, the following value causes an integer overflow:

```
SQL> select cast(21474836 as interval second(9,2)) from rdb$database;

  021474836.00
1 row selected
SQL> select cast(21474837 as interval second(9,2)) from rdb$database;
%RDB-E-ARITH_EXCEPT, truncation of a numeric value at runtime
-COSI-F-INTOVF, integer overflow
```

The value shown here represents only about 248 days. In fact the storage mechanism for day/time intervals will support values in excess of 999999999 seconds (11574 days) up to 24855 days in total.

In addition, the results on Alpha systems have been corrected. Where an integer overflow was expected, Rdb returned an incorrect result, for example:

```
SQL> select cast(999999999.99 as interval second (9,2)) from rdb$database;

  012157521.91
1 row selected
```

This error is corrected, and the other restrictions are removed for Rdb V6.1A ECO 2 and for Oracle Rdb Version 7.0.1. The full range of values can now be assigned , and the source numeric value can be a BIGINT value (values which are too large to be stored in an INTEGER data type).

2.1.75 Privilege Checking Lifted for Columns Referenced from FOR Loop

Bug 439485.

When an UPDATE statement was nested in a FOR loop in a compound statement and it also referenced columns fetched by that FOR loop, Rdb required that the user have the UPDATE privilege to those columns and the table to which they belonged. This requirement was unexpected because the table in the FOR loop was not being updated. For example:

```
begin
-- Find the employee who is changing jobs
for :emp as each row of cursor emp_cur for
    select last_name, employee_id
    from employees
    where employee_id = :eid
do
    -- now complete the current job
    update job_history
    set job_end = cast(current_date as date vms)
    where employee_id = :emp.employee_id
      and job_end is null;

    -- now create new job_history record for the new job
    ...remainder of procedure...

end for;
end;
```

In this example the UPDATE of JOB_HISTORY references the column :EMP.EMPLOYEE_ID from the outer FOR loop. Rdb would require UPDATE to the column EMPLOYEE_ID and the table EMPLOYEES even though they are not updated by these queries. When columns from the FOR loop are referenced only SELECT privilege should be required to reference the columns from an UPDATE statement.

This requirement was unnecessary and has been lifted in Oracle Rdb Version 7.0.1.

2.1.76 Logical Area Lock may not be Released After an Exception like a DEADLOCK Error

Bug 491599.

When using carry over locking, a logical area lock could not be released after an exception like DEADLOCK or lock time out.

The following example shows this problem:

```
Session 1:
---------
SQL> at 'file foo';
                                        Session 2:
                                        ---------
                                        SQL> at 'file foo';

SQL> set trans read write reserving t
     for shared write;

                                        SQL> set trans read write reserving t
                                             for shared write;

SQL>update t set f = 1 ;

                                        SQL>update t set f = 1 ;

%RDB-E-DEADLOCK, request failed due to resource deadlock
-RDMS-F-DEADLOCK, deadlock on logical area 47
SQL> roll;

                                        ** This process is still waiting with:

Process.ID Since..... T Stall.reason.......................... Lock.ID.
00000703:1 19:03:32.60 W waiting for logical area 47 (PW)         11001183
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.77 CREATE INDEX Could Bugcheck for SORTED RANKED Indexes

Bug 483581.

In Oracle Rdb7 it was possible when creating a SORTED RANKED index that a bugcheck would occur when attempting to build a duplicates bit map. This problem occurred only in non-UNIQUE indexes with a large number of duplicates per key value (i.e. more than 65535 duplicates). It occurred because there was insufficient space in the B-tree index node to expand the duplicates count to a 32 bit value.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.78 After dropping a Hash Index, Re-Creating the Same Index Bugchecks

Bug 478314.

In some cases, once a hashed index was dropped, a subsequent RMU/VERIFY /INDEX could display corrupted system records. If the same index was then created, a bugcheck could occur at PSIHASHBKT$SEARCH_BUCKET. This problem occurred only when there are two fragmented system records in a row that needed to be deleted.

This same behavior could be exhibited with TRUNCATE TABLE, DROP TABLE, ALTER INDEX, and DROP STORAGE AREA CASCADE which can all indirectly execute this same DROP INDEX functionality.

The following example illustrates the problem:

```
SQL> drop index hash_index;
SQL> commit;

$RMU/VERIFY/INDEX foo_db
- displays corrupted system records
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.79 Bugcheck Dump When Trying to Rollback After a Trigger Reported an Error

Bug 487430.

OpenVMS Alpha

This problem occurred when a SQL statement started a trigger which itself fired some other triggers and one of these triggers does an index scan which reports an error. As a result, a bugcheck dump could occur when a rollback is executed. The error is:

```
***** Exception at xxxxxxxx : KOD$ROLLBACK + 00000194
%COSI-F-BUGCHECK, internal consistency failure
```

The following example shows this problem:

```
SQL> delete from tab where fld='0890';
%RDB-E-TRIG_INV_UPD, invalid update; encountered error condition defined for tri
gger
-RDMS-E-TRIG_ERROR, trigger trg1 forced an error
-RDB-F-ON_DB, on database FOO.RDB;1
SQL> rollback;
%RDMS-I-BUGCHKDMP, generating bugcheck dump file DSK:[DIR]RDSBUGCHK.DMP;
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.80 LIKE Operator not Returning Rows with Index Only Retrieval

Bug 318983.

Rdb V6.1 introduced an optimization for the LIKE operator which enabled LIKE to be solved using an index scan which greatly reduced the I/O for the LIKE predicate. The optimization used the non-wildcard prefix to limit the upper and lower bounds of the index scan.

However, if the table was partitioned then LIKE would not return any rows except those found in the final partition of the index. This problem was caused by Rdb using the full LIKE pattern to select the starting index partition. These patterns contain "_" and "%" which sort higher than numeric and alphabetic data and so selected a starting partition beyond the required data values.

This problem has been corrected in Oracle Rdb Version 7.0.1.

Rdb now uses the non wildcard prefix for partition selection.

2.1.81 Recovery Operations with Record Cache Cause Corruption

When the record cache feature was enabled, subsequent recovery (roll-forward) operations from the after-image journal could result in missing rows from the database.

Oracle Rdb would sometimes be unable to determine that a row was modified or added and would then fail to add the row to the after-image journal file. A later recovery would then not insert the row into the database.

This problem has been corrected. Oracle Rdb now correctly journals rows when the record cache feature is enabled.

2.1.82 Float-Divide-by-Zero Error from a View Definition Query

Bug 498788.

The following view definition failed with the following arithmetic exception :

```
create view VIEW_MEDL_ARBGIVER_ALLE
    (A1,
     A2,
     A3,
     A4,
     A5,
     A6,
     A7,
     PS) as
    select
        case
            when (not (C1.A1 is null)) then C1.A1
            else C2.A1
        end,
        case
            when (not (C1.A1 is null)) then C1.A2
            else C2.A2
        end,
        case
            when (not (C1.A1 is null)) then C1.A3
            else C2.A3
        end,
        case
            when (not (C1.A1 is null)) then C1.A4
            else C2.A4
        end,
        case
            when (not (C1.A1 is null)) then C1.A5
            else C2.A5
        end,
        case
            when (not (C1.A1 is null)) then C1.A6
            else C2.A6
        end,
        case
            when (not (C1.A1 is null)) then C1.A7
            else C2.A7
        end,
        case
            when (not (C1.A1 is null)) then 'P'
            else 'S'
        end
    from VIEW_MEDLEM_NUV as C1
    full outer join
    VIEW_MEDL_ARBGIVER_SUPPL as C2 on (((C1.A2 = C2.A2y
    )
            and (C1.A3 = C2.A3))
            and (C1.A1 = C2.A1));
```

On OpenVMS VAX:

```
 %RDB-E-ARITH_EXCEPT, truncation of a numeric value at runtime
 -SYSTEM-F-FLTDIV_F, arithmetic fault, floating divide by zero at
 PC=002F1CB2, PSL=03C00004
```

On OpenVMS Alpha:

```
%RDB-E-ARITH_EXCEPT, truncation of a numeric value at runtime
%SYSTEM-F-HPARITH, high performance arithmetic trap, Imask=00000000,
Fmask=00000001, summary=02, PC=808D4B10, PS=0000001B
-SYSTEM-F-FLTINV, floating invalid operation, PC=808D4B10, PS=0000001B
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.83 Incorrect Number of Reserved Journals Upon Import

Bug 389781.

The number of unused journals was not correct after the export/import operation. If, for example, when creating/altering a database, the number of reserved journals was specified to be 10, then 3 journals were used, the number of unused journals would be 7. This is correct behavior. However. if the database is exported/imported, the count of unused journals should not be 7. Before the fix, the number of ununused journals was 7. This is incorrect because the journal information is not export/imported. Thus, the number of unused journals should be 10 as was specified when the user first created or altered the database.

```
-- The number of unused journals should be 10 not 7.
CREATE DATABASE FILE RESERVE_TEST
 RESERVE 10 JOURNALS;
-- Use 3 journals
ALTER DATA FILE RESERVE_TEST
 ADD JOURNAL JOU_1 FILENAME JOU_1
 ADD JOURNAL JOU_2 FILENAME JOU_2
 ADD JOURNAL JOU_3 FILENAME JOU_3
;
-- show that the database has 7 unused journals
SHOW DATABASE *
-- Export/import the database - the number of unused journals should be 10
DISCONNECT ALL;
EXPORT DATA FILE RESERVE_TEST INTO X;
DROP DATA FILE RESERVE_TEST;
IMPORT DATA FROM X FILE RESERVE_TEST;
-- Show the database to insure journals are 10
SHOW DATABASE *
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.84 Column Privileges for Views not Imported for Ansi-Style Protection

SQL was not importing column level ACL's for view columns for an ansi-style protection database. This problem only occurred with ANSI-style protection, not ACL-style protection.

The following example demonstrates that SQL is now importing the column level ACL's correctly:

```
set ver;
create data file foo protection is ansi;
!
create table tab1 (col1 char (5));
grant select on table tab1 to rdb_execute;
grant update on column tab1.col1 to rdb_execute;
!
create view view1 (col1) as select col1 from tab1;
grant select on table view1 to rdb_execute;
grant update on column view1.col1 to rdb_execute;
!
show prot on table view1;
show prot on column view1.col1;
commit;
disc all;
!
export data file foo into foo;
drop data file foo;
!
import data from foo file foo;
!
! These are all correct.
show prot on table tab1;
show prot on column tab1.col1;
show prot on table view1;
!
! The view column privs is no longer missing.
show prot on column view1.col1;
disc all;
```

2.1.85 Show Index for a Hashed Index with Mapping Values was Incorrect

SQL was not Showing the correct information for a hashed index with mapping values. Namely, SQL was displaying the index as HASH SCATTERED as well as SORTED. This is incorrect. SQL should not display TYPE IS SORTED for an hashed index.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The following example demonstrates that SQL is now showing index information correctly.

```
att 'file mf_personnel';
alter table employees add column emp_id_num1 int;
create index empid_map_h on employees
    (emp_id_num1 mapping values 160 to 500)
    type is hashed
    store in empids_low
;
commit;
sho table(index) employees;
Information for table EMPLOYEES

Indexes on table EMPLOYEES:
EMPLOYEES_HASH                  with column EMPLOYEE_ID
  No Duplicates allowed
  Type is Hashed Scattered
  Compression is DISABLED
 Store clause:   USING EMPLOYEE_ID
```

2.1.86 IMPORT was Showing Incorrect Information for OPEN Clause

Upon import, SQL was not displaying the correct information for the WAIT x MINUTES FOR CLOSE clause. Specifically, if a negative number was specified on the command line during import, import would display an incorrect number.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The following example demonstrates that SQL is now displaying the correct information during import:

```
! create a 'good' version for the alter's and import's to use...
create database filename 'auto_close_3'
  open automatic (wait 0 minutes for close);

disconnect all;

export database filename auto_close_3 into auto_close_3;

disconnect all;

!
! IMPORT section
!
-- setup for export/import sequences
drop database filename auto_close_3;

-- now do the import
import database from auto_close_3 filename auto_close_3
  open automatic (wait -10 minutes for close);
```

The example should fail with the following message. This is expected because the number -10 is not within the valid range:

```
%SQL-F-ERRCRESCH, Error creating database filename auto_close_3
-RDB-E-BAD_DPB_CONTENT, invalid database parameters in the database parameter block (DPB)
-RDMS-F-BADPARAM, close timer interval (-10) is out of valid range (0..1440)
```

2.1.87 Transaction Modes for CREATE/ALTER Database not EXPORTED/IMPORTED

SQL was not exporting/importing the transaction modes specified for a create /alter database.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The following example demonstrates that SQL now export and imports the transaction modes correctly:

```
CREATE DATA FILE X SET TRANSACTION MODE (NO BATCH UPDATE);
EXPORT DATA FILE X INTO X1;
DISCONNECT ALL;
IMPORT DATA FROM X1 FILE X2;
```

2.1.88 Incorrect Defaulting of THRESHOLDS in CREATE/ADD STORAGE AREA

When creating a MIXED page format storage area the THRESHOLDS clause can be used to control space utilization and adjust free space searches. The THRESHOLDS value list could contain one, two or three values. The Oracle Rdb documentation stated that if only one value was provided, such as in THRESHOLD IS (94), then this was equivalent to THRESHOLDS ARE (94,100,100). This means the missing threshold values were assumed to be the trailing thresholds which should default to 100.

In prior versions of RDb, the default of the missing trailing versions was not working correctly. For example, THRESHOLD IS (94) would default to THRESHOLDS ARE (0,0,94) which is incorrect.

This has been corrected in this release of Rdb. This problem did not occur for the THRESHOLDS clause specified on the CREATE/ALTER STORAGE MAP or CREATE/ALTER INDEX statements.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.89 SHOW TABLE now Orders Storage Area, Cache and Journal Names

With this release of Rdb the SQL SHOW STORAGE AREA, SHOW CACHE and SHOW JOURNAL commands will order names alphabetically by the ASCII collating sequence (these names do not have an associated character set).

The logical name RDMS$BIND_SORT_WORKFILES will be used to allocate work files if needed.

Note

If the IDENTIFIER CHARACTER SET for the database is other than MCS or ASCII then this option will be ignored. This is because the characters from other character sets do not sort appropriately under the ASCII collating sequence.

The following example shows the storage areas for MF_PERSONNEL ordered alphabetically:

```
att 'file mf_personnel';
show storage areas
Storage Areas in database with filename mf_personnel
     DEPARTMENTS
     EMPIDS_LOW
     EMPIDS_MID
     EMPIDS_OVER
     EMP_INFO
     JOBS
     MF_PERS_SEGSTR                  List storage area.
     RDB$SYSTEM                      Default storage area.
     SALARY_HISTORY
```

2.1.90 Use of STORE Clause Now Enforced with Vertical Record Partitioning Storage Maps

Bug 481766.

The STORE COLUMNS clause is now enforced with VRP (Vertical Record Partitioning) storage maps. Prior to this enforcement, a storage map created with just the COLUMNS keyword could result in looping behavior.

The following example demonstrates the syntax error now generated when the COLUMNS clause is not preceded by the keyword STORE:

```
SQL> create data file store_clause
cont> create storage area AREAA
cont> create storage area AREAB
cont> create storage area AREAC
cont> create table T6 (a integer, b integer, c integer);
QL> create storage map M6 for T6
cont>      enable compression
cont>   store columns (a)
cont>     disable compression
cont>     in AREAA
cont>   columns (b)
     columns (b)
     ^
%SQL-W-LOOK_FOR_STT, Syntax error, looking for:
%SQL-W-LOOK_FOR_CON,            (, OTHERWISE, STORE, ENABLE, DISABLE,
%SQL-W-LOOK_FOR_CON,            PLACEMENT, THRESHOLD, THRESHOLDS,
%SQL-W-LOOK_FOR_CON,            REORGANIZE, PARTITIONING, ;,
.
.
.
```

2.1.91 Error During Hot Standby Startup

During the startup phase of a Hot Standby "restart" operation, it was sometimes possible to receive the AIJSIGNATURE error when using multiple fixed-size AIJ journals. The problem could occur when the last block of a master database journal was completely full (every byte fully used).

The problem only occurred during hot standby restart. The problem seemed to occur more frequently when using the quiet-point ABS backup operation.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.92 Problems with the TRUNCATE Command

2.1.92.1 The RMU/VERIFY/ALL Command Returns Inconsistency Between SPAM Page and Area Bitmap After a TRUNCATE Command

Bug 483623

In some cases, after truncating a table, the RMU/VERIFY/ALL command reported that ABM and SPAM pages are not consistent. The problem existed in Oracle Rdb 7.0 and 7.0 ECO1.

The following example shows the reported error.

```
SQL> attach 'file MF_PERSONNEL';
SQL> truncate table EMPLOYEES;
SQL> commit;
```

```
$ rmu/verify/all MF_PERSONNEL
%RMU-W-ABMBITERR, inconsistency between spam page 4361 and bit 5 in area bitmap
in larea 121 page 4280
%RMU-E-BADABMPAG,       error verifying ABM pages
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.92.2 The TRUNCATE TABLE Command Returns RDMS-F-NODBK When Row Caching is Enabled

Bug 494941.

In some cases, truncating a table returned an error when row caching was enabled for that table. This problem existed in Oracle Rdb 7.0 and 7.0 ECO1.

The following example shows the reported error when row cache is enabled for the table and a truncate command issued.

```
SQL> truncate table employees;
%RDB-E-NO_RECORD, access by dbkey failed because dbkey is no longer associated
with a record
-RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NODBK, 51:4986:1 does not point to a data record
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.93 Query with "(SELECT COUNT(*) FROM ...) = 0" Expression Returns the Wrong Result

Bug 498674.

The following query with "(SELECT COUNT(*) FROM ...) = 0" expression in the WHERE clause, returned wrong number of rows in Rdb V7.0:

The following example shows the reported error:

```
! the following should return 3 rows
select p.cprnr
   from T1 p where
     (select count(*) from T2 d where
      d.id = p.id and
      d.data_type = 999) = 0 ;
0 rows selected
```

In the versions prior to Oracle Rdb7, a match solution with an aggregate subquery in its inner leg was always executed as a left outer join. That is, every row in the outer leg of the match was scanned. This means, no zigzag skip can be performed on the outer leg of the match.

In Oracle Rdb7, to enable better optimization, changes were made to differentiate the subqueries that required left outer joins from those that did not. This differentiation was based on the form of the subquery itself.

For example, NOT EXISTS subqueries and subqueries that are sub-selects require left outer join execution. However, EXISTS subqueries do not require left outer join execution.

Unfortunately, due to these changes the above query with "count=0" was not not recognized correctly by the optimizer.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.94 Query with Sub-Select in a Derived Table Returns the Wrong Result

Bug 503672

The following query, with sub-select in a derived table, returned incorrect results in Oracle Rdb7. It should return 100 rows.

```
select * from (
 select employee_Id,
  (select count(*) from degrees d
    where d.employee_id = e.employee_id
      and d.degree = 'PhD') x
 from employees e) z;
```

The following query without a derived table returns the correct results:

```
select employee_Id,
  (select count(*) from degrees d
    where d.employee_id = e.employee_id
      and d.degree = 'PhD') x
 from employees e;
```

Similarly a sub-select within a UNION returned incorrect results:

```
 select employee_Id,
  (select count(*) from degrees d
    where d.employee_id = e.employee_id
      and d.degree = 'PhD') x
 from employees e
 union all
 select employee_Id,
  (select count(*) from degrees d
    where d.employee_id = e.employee_id
      and d.degree = 'PhD') x
 from employees e ;
```

In the versions prior to Oracle Rdb7, a match solution with an aggregate subquery in its inner leg was always executed as a left outer join. That is, every row in the outer loop of the match was scanned. This means no zig-zag skip can be performed on the outer leg of the match.

In Oracle Rdb7, to enable better optimization, changes were made to differentiate the subqueries that required left outer joins from those that did not. This differentiation was based on the form of the subquery itself.

For example, NOT EXISTS subqueries and subqueries that are sub-selects require left outer join execution. However, EXISTS subqueries do not require left outer join execution.

Unfornately, due to these changes the above query with a sub-select statement from a derived table was not recognized correctly by the optimizer. The solution is to change the query compiler to detect the sub-select query of a derived table (and also a union).

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.95 Moved Storage Areas Fail in a Cluster

In certain cases, storage areas that were moved or renamed using the RMU/MOVE_AREA/ONLINE command might not be accessible from other nodes in a cluster.

For example, the following sequence resulted in the failure to open the storage area on NODE_2:

```
  ! Create a multifile database

  NODE_1: SQL> CREATE DATA FILE D CREATE STORAGE AREA D1;
  NODE_1: SQL> CREATE TABLE T1 (F1 INT);
  NODE_1: SQL> CREATE STORAGE MAP M1 FOR T1 STORE IN D1;
  NODE_1: SQL> INSERT INTO T1 VALUES (1);
  NODE_1: SQL> COMMIT;

  ! Open the database on two cluster nodes

  NODE_1: $ RMU/OPEN D
  NODE_2: $ RMU/OPEN D

  ! Rename an area on one cluster node

  NODE_1: $ RMU/MOVE_AREA/ONLINE D D1/FILE=D1_M
```

```
! Connect to the database on the other node &
! read the table stored in the area just moved:

NODE_2: SQL> ATTACH 'FILE D';
NODE_2: SQL> SELECT * FROM T1;

%RDB-F-SYS_REQUEST, error from system services request
-RDMS-F-FILACCERR, error opening storage area file DSK:[DIR]D1.RDA;1
-RMS-E-FNF, file not found
```

This problem occurred when the database filename was not correctly being moved to the user's process from the root file.

This problem has been corrected. Oracle Rdb now correctly maintains the storage area file name after a move operation.

2.1.96 Wrong Results when Using a Longword Literal

Bug 480013.

Zero rows were returned if a zig-zag match strategy was chosen and a small integer was compared to a longword integer. The following example illustrates the problem.

```
SQL> Select t1.col1, t2.col1
cont> from table1 t1, table2 t2
cont> where t2.col2 = t1.col1 + 99900;
Conjunct
Match
  Outer loop
    Sort    Get     Retrieval sequentially of relation TABLE1
  Inner loop      (zig-zag)
    Index only retrieval of relation TABLE2
      Index name  TABLE2_INDEX [0:0]
0 rows selected
```

If the query was changed so that the literal was on the left hand side of the equation, the optimizer's strategy changed and the query produced the correct results.

```
SQL> Select t1.col1, t2.col1
cont> from table1 t1, table2 t2
cont> where t2.col2 - 99900 = t1.col1;
Conjunct
Match
  Outer loop
    Sort    Get     Retrieval sequentially of relation TABLE1
  Inner loop
    Temporary relation      Sort
    Index only retrieval of relation TABLE2
      Index name  TABLE2_INDEX [0:0]
 T1.COL1      T2.COL1
      39        19291
1 row selected
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.97 Changes to AIP Format in RMU/DUMP

In prior releases the information in the area inventory page (AIP) dump output from RMU/DUMP/LAREA=RDB$AIP was not correct or clear when it described Hash indexes.

In this release of Oracle Rdb the dump format has been revised to better describe the information related to hash indexes.

```
                                entry #8
                       00000000  01FC  first area bitmap page 0
                      0005 0039  0200  logical area 57, physical area 5
                             0E  0204  area name length 14 bytes
00004B4F4F425F454E4F4850454C4554  0205  area name 'TELEPHONE_BOOK..'
  00000000000000000000000000000000  0215  area name '................'
                       00000004  0224  snaps enabled TSN 4
                           00D7  0228  record length 215 bytes
                       00000000  022A  MBZ '....'
                             01  022E  entry is in use
                           0000  022F  MBZ '..'
                         000000  0231  thresholds are (0,0,0)
                             02  0234  hash access is scattered ** junk **
                             03  0235  record type hash index
                             00  0236  MBZ '.'
```

This example shows that the logical area is used for a hash index (see offset 0235) and that the hash index of type scattered (offset 0234). If the index was of type ordered then the byte at offset 0234 would indicate the offset of the index segment within the index key which is used for page placement.

In this example the value should have been zero as it is not used by scattered style hash indexes. The dumper indicates this by displaying the string "** junk **". This problem is corrected in this of Oracle Rdb, however, databases created with older versions of Rdb (namely Rdb 6.0, or 6.1) may contain a non-zero value in this field. This is harmless and can be ignored.

2.1.98 Additional Types Supported for Optimized Mapping

The Oracle Rdb7 Release Notes section (3.1.23) entitled "Performance Enhancement for Storage Maps and Mapped Indexes" describes a CPU performance enhancement which allows Oracle Rdb to compare values more efficiently when the map is by a single column. That is one column in the USING clause and a data type which fits in a 64 bit value (such as SMALLINT, INTEGER, BIGINT, small CHAR values).

In this release of Oracle Rdb, columns of type DATE VMS, DATE ANSI, TIMESTAMP, TIME, and INTERVAL are also added to this performance optimization. No application or database changes are required to take advantage of these changes.

2.1.99 ALS Bugcheck Without Exception Following DBR Invocation

It was possible for the ALS to produce a bugcheck dump that does not contain an exception. This problem was caused by a race condition with a DBR recovery process running on the same node as the ALS process. Due to the race condition, it was possible for the DBR process to reset some information required and validated by the ALS process.

When the problem occurred, the ALS process produced a bugcheck dump that did not contain an exception. The bugcheck stack trace had the following "Saved PC" entries (the offsets may vary depending on platform and product version):

```
Saved PC = XXXXXXXX : AIJUTL$ABORT + 000002E4
Saved PC = XXXXXXXX : AIJUTL$WRITE_CACHE + 000002EC
Saved PC = XXXXXXXX : ALS$FLUSH_ONE_CACHE + 00000740
Saved PC = XXXXXXXX : ALS$FLUSH + 00000180
Saved PC = XXXXXXXX : ALS$MAIN + 00000A00
```

This problem could occur even when the "AIJ Fast Commit" feature is disabled. The problem occurred when the ALS process waits for the DBR process to complete recovery; typically, the DBR process waits for the ALS process to complete AIJ journal I/O.

Note

This problem was not exclusively related to the ALS process. Because the problem lies in the DBR process resetting the information, this problem could also occur when the ALS process was not used, resulting in application processes bugchecking without an exception.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.100 User Process Deleted During Processing of Complex Query

Bugs 491448, 497343, 485719, 479352 and 392597.

This symptom could be a result of two different categories of problems.

The first category: in some cases, when complex queries were processed, the default executive mode stack size can be exceeded. In earlier versions of Oracle Rdb this could cause the user process to be deleted. The following error message could be returned to the user when the executive mode stack was exceeded:

```
%RDB-F-IMP_EXC, facility-specific limit exceeded
-RDMS-F-XPR_STACK_OFLO, expression forces too many levels of recursion
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

The second category involves a query with complex nested CAST...AS INTERVAL expressions which require more executive mode stack space to process. This problem only occurred on OpenVMS Alpha.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.101 AIJ Backup Failure Leaves Tape Unusable

Bug 401707.

When performing an RMU Backup After_Journal operation to tape, RMU could abort the operation if a corruption exists in the after-image journal (.aij) file. In such a case, the tape was left in an unusable state. Any subsequent attempt to write past the current tape position resulted in a positioning error. Because the positioning error required operator intervention, this situation is particularly problematic when you used automated procedures that write other data to the tape after the .aij backup operation aborts.

The following example demonstrates the positioning error:

```
! Attempt to perform an .aij backup operation followed by a database
! backup operation.  Assume that the .aij file is corrupt.

$ MOUNT/FOREIGN ZARA$MKB500:
%MOUNT-I-OPRQST, Please mount device _ZARA$MKB500:
%MOUNT-I-MOUNTED, BACKUP mounted on _ZARA$MKB500:
%MOUNT-I-RQSTDON, operator request canceled - mount completed successfully
$ RMU/BACKUP/AFTER/REWIND/LABEL=BACKUP/FORMAT=NEW_TAPE DB.RDB ZARA$MKB500:DBAIJ
%RMU-F-AIJTERMINATE, inaccessible AIJ file forced image exit to protect
database
$ RMU/BACKUP/LABEL=BACKUP DB.RDB ZARA$MKB500:DBBCK
%RMU-E-POSITERR, error positioning ZARA$MKB500:[000000]DBBCK.RBF;
-SYSTEM-F-OPINCOMPL, operation is incomplete
%RMU-I-SPECIFY, specify option (QUIT or CONTINUE)
RMU> QUIT
%RMU-F-ABORT, operator requested abort on fatal error
%RMU-F-FATALERR, fatal error on BACKUP
%RMU-F-FTL_BCK, Fatal error for BACKUP operation at 19-JUN-1997 14:44:19.35
```

The only workaround required operator intervention. The operator had to rewind the tape before new data could be written. However, rewinding the tape resulted in the loss of all data previously written to the tape.

This problem has been corrected in Oracle Rdb Version 7.0.1. RMU writes the necessary end-of-volume and end-of-file marks to the tape, thus allowing a subsequent tape operation to position itself correctly.

2.1.102 Sometimes Ranked B-Tree Indexes not Pointing at Rows After Rollback of a Delete

All Platforms

Performing a RMU/VERIFY/INDEX on a database would show that rows are not being pointed to by a ranked b-tree index. Cardinality count was more than the actual number of entries in the leaf node. The difference was the exact number of the rows not pointed to by the index.

The problem only occurred when users deleted all the entries in the overflow leaf node (duplicates) and ROLLBACK the operation.

The following example shows this problem:

```
rmu/verify/all RDB_RANDOM_TU_12_CS.RDB;3
%RMU-I-BTRDUPCAR, Inconsistent duplicate cardinality (C1) of 482
  specified
                  for entry 1 at dbkey 88:48:0.
                  Actual count of duplicates is 474                .
%RMU-I-BTRROODBK, root dbkey of B-tree is 88:48:0
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:24.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:25.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:26.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:27.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:28.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:29.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:30.
%RMU-W-DATNOTIDX, Row in table YEAR_TO_DATE_REC is not in any indexes.
                  Logical dbkey is 126:241:31.
```

```
Note: 482 - 474 = 8 and there are exactly 8 rows in the table not pointed
by the index.
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.103 Read-Only Transactions Sometimes Bugcheck

Bugs 449000 and 432959.

All OpenVMS platforms.

Read-only transactions sometimes bugchecked in FETCH_SNAP_SEG under a multi-user environment.

The problem was introduced in 7.0 ECO1. The problem only showed up when a delete of a line happened to be located at the end of a page. Later, on a store operation, the line index of the page is trimmed off. This makes the TSN of the deleted line disappear and results in the snap shot page generating a wrong TSN.

The following example shows this error:

```
Exception: DIOFETCH$FETCH_SNAP_SEG + XXX
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.104 Bugcheck with Exception at RDMS$$FIND_MEMBER_EQV_CLASS + CC

Bug 490219.

In some cases, an exception would occur during a query with a join of several tables. The error observed was "exception at RDMS$$FIND_MEMBER_EQV_ CLASS + CC".

The following example shows a query that resulted in a bugcheck:

```
select employee_id from employees e
                        departments d1,
                        departments d2
         where  e.employed_by = d1.depart_id and
                e.cost_center = d2.depart_id and
                d1.depart_id  = d2.depart_id;
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.105 Query with Transitive Predicate in Match Strategy Returns Wrong Result

Bug 474343.

The following query should return 2 rows but only returns 1:

```
! this should return 2 rows, but find only 1 row
SELECT PRR FROM T1
    WHERE PO = 'A1234567' AND CO = 9 -
    AND EXISTS (SELECT T2.PRR FROM T2
      WHERE T2.CO = T1.CO  AND
     T2.PRR = T1.PRR);
Conjunct
Match
  Outer loop      (zig-zag)
    Conjunct        Index only retrieval of relation T1
      Index name  T1_NDX [2:2]
  Inner loop      (zig-zag)
    Aggregate-F1    Index only retrieval of relation T2
      Index name  T2_NDX [0:0]
 PRR
 CS002900040190
1 row selected
```

In the versions prior to Oracle Rdb7, a match solution with an aggregate subquery in its inner leg was always executed as a left outer join. That is, every row in the outer leg of the match was scanned. This means, no zig-zag skip can be performed on the outer leg of the match.

In Oracle Rdb7, to enable better optimization, changes were made to differentiate the subqueries that required left outer joins from those that did not.

For example, NOT EXISTS subqueries and subqueries that are sub-selects require left outer join execution. However, EXISTS subqueries do not require left outer join execution.

Due to these changes, the Oracle Rdb7 optimizer correctly generates the match strategy applying zig-zag on both legs on the above query, as compared to the version prior Oracle Rdb7, where outer join was applied with sort node in front of the index retrieval.

Unfortunately, this exposed a problem in the boolean node generation from transitive predicates for the match strategy. The match keys comparison (T2.CO = T1.CO AND T2.PRR = T1.PRR) without a transitive conjunct for "CO = 9" caused the query to return wrong results.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.106 EXTRACT Function Returning Incorrect Values from INTERVAL Types

Bug 500869.

In prior versions of Oracle Rdb the EXTRACT function for both VAX and Alpha platforms would return the incorrect value when large values for seconds and minutes were extracted from an INTERVAL data type.

```
SQL> declare :int_sec interval second(2);
SQL> declare :int_dt interval day(9) to second(2);
SQL> declare :sec_int bigint(2);
SQL> declare :sec_int2 bigint(2);
SQL> begin
cont>     set :int_sec = (timestamp'1997-06-02:13:40:52.67'
cont>               - timestamp'1970-01-01:00:00:00.00') second;
cont>     set :int_dt = cast(:int_sec as interval day(9) to second(2));
cont>     set :sec_int = extract(second from :int_sec);
cont>     set :sec_int2 = extract(second from :int_dt) +
cont>               60 * (extract(minute from :int_dt) +
cont>                     60 * (extract(hour from :int_dt) +
cont>                           24 * extract(day from :int_dt)));
cont> end;
SQL> print :sec_int, :int_dt, :sec_int2;
                SEC_INT   INT_DT                                    SEC_INT2
             6265393.47    000010014:13:40:52.67              865258852.67
```

This example calculates the number of seconds between the two TIMESTAMP values shown. One extracts each field of a DAY TO SECOND interval and multiplies the values to get the result. The other uses EXTRACT (SECOND) from the SECOND only interval. As can be seen in the example above, the result returned from EXTRACT is incorrect. The problem was that the resulting number of seconds requires a quadword (requiring 64 bits) intermediate variable.

In this release of Oracle Rdb, the EXTRACT function has been enhanced to use a quadword data type as an intermediate variable. However, as the documented return type for EXTRACT is INTEGER (or INTEGER(2) when SECOND is extracted), then the EXTRACT must be embedded in a CAST expression if you expect values large than will fit into an integer. For example, use CAST as a BIGINT or DOUBLE PRECISION value to extract a larger value.

```
SQL> begin
cont>     set :int_sec = (timestamp'1997-06-02:13:40:52.67'
cont>               - timestamp'1970-01-01:00:00:00.00') second;
cont>     set :sec_int = cast(Extract(second From :int_sec) as bigint(2));
cont>     set :sec_intd = cast(Extract(second From :int_sec) as double precision);
cont> end;
SQL> print :sec_int, :sec_intd;
                SEC_INT                 SEC_INTD
            865258852.67    8.652588526700001E+008
```

2.1.107 RMU/CONVERT Corrupts Metadata

Bug 507408.

In versions of RMU since v6.0, when converting a database from an earlier version, domain definitions for system domains other than those used by RDB's own system relations were being erroneously removed. This can cause failure of other options that have their own "system" domains, such as the Replication Option (Data Distributor).

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.108 Processes Stalling for Page Locks when ABW is Enabled

Bugs 460533, 460062, 442725, and 449685.

Processes could stall for extended periods, waiting for a page lock to be released by another process. The page lock would eventually be released when the blocking process voluntarily gave up that page: typically when finishing its transaction, or when reusing the buffer containing that page.

However, if the database operated in "fast commit" mode, the blocking process might not be able to give up the page, since it may be itself waiting for user input or for other work to do.

2.1.109 Query with a CAST Boolean Predicate Returns Wrong Result

Bug 498012.

The following query which a CAST boolean predicate returned wrong results:

```
select * from (select f1,cast(f2 as integer),f3 from t1) x (f1,f2,f3)
  where f2 = 1;
Merge of 1 entries
  Merge block entry 1
  Leaf#01 FFirst T1 Card=4
    BgrNdx1 I1 [0:0] Bool Fan=17
0 rows selected
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.110 Index Cardinality Problems

Bug 474805

With Rdb7 and with Rdb7 ECO1 there were reports of excessive locking, excessive stalls and incorrect index prefix cardinalities. Oracle believes these are corrected by the following changes.

2.1.110.1 AIJ File Corruption

In certain situations Rdb did not complete a transaction correctly. This meant that updates were entered into the AIJ file with a TSN of zero, which cannot be applied by a rollforward, nor by a hot standby.

If you are running a version prior to Oracle Rdb Version 7.0.1 and you are running with PRESTARTED TRANSACTIONS disabled (either explicitly using the SQL syntax or implicitly using the Rdb7 logical name), you must immediately disable CARDINALITY COLLECTION for the database.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.110.2 Stalls on the RDB$INDEX_SEGMENTS Table

Every 100 transactions Rdb would flush any cardinalities of active transactions to disk. This rate was often too high. Symptoms were stalls on rows in the RDB$INDEX_SEGMENTS table.

This problem has been corrected in Oracle Rdb Version 7.0.1.

We have changed the default value for RDMS$BIND_CARD_UPDATE_QUOTA from 100 to 2,147,483,646. This now means that normally the number of transactions will have no bearing on when index cardinalities are updated on disk. Now it will only come into play when you so choose by defining the logical name RDMS$BIND_CARD_UPDATE_QUOTA.

2.1.110.3 Excessive I/O for SORTED Indices

For SORTED indices prefix cardinalities were being written to disk more often than necessary thereby causing excessive I/O.

This problem has been corrected in Oracle Rdb Version 7.0.1.

A check is now made to determine if the cardinality of any of an index's prefixes has changed and meets or exceeds a difference threshold. If none of the prefixes for a given index meets this test, the prefix cardinalities will normally not be updated on disk. The exceptions are (1) when no cardinality updates have occurred and the number of transactions which have been recorded in the interim equals the value in RDMS$BIND_CARD_UPDATE_QUOTA or the Rdb default, or (2) the application disconnects from the database.

2.1.110.4 Query Optimization Affected By Index Cardinality Drift

Rdb maintains index and index prefix cardinalities as input for the Rdb query optimizer. These cardinality values are approximations and are adequate for most database usage. We did, however, note instances where the collection of index cardinality was not as accurate as it could be, in some cases causing excessive "cardinality drift".

One of the causes for cardinality drift was that accumulated changes were not being written out to disk (they were discarded) when an application disconnected from the database. This has been corrected.

In addition, storing of index prefix cardinalities has been separated from the updating of the index cardinality, since it is possible for one or the other to change by a significant amount without the other.

Even with these changes, however, cardinality drift can still occur when data rows are updated or deleted. To correct cardinality drift you can periodically use RMU/COLLECT to collect and update the cardinalities.

2.1.110.5 Excessive I/O with Compressed, Multisegment Indexes

An error was discovered in the way changes in index prefix cardinalities were being detected. A problem that happened with compressed indexes. With this error, Rdb had the potential of erroneously computing large changes in index prefix cardinality. This in turn could cause excessive I/O to update the cardinality changes and could affect query optimizer performance because of the resulting incorrect cardinalities.

It is suggested that you use RMU/COLLECT for those indices which are compressed to change the cardinalities to their true values.

This problem has been corrected in Oracle Rdb Version 7.0.1, but you must initially correct the cardinalities as described above.

2.1.111 RMU/SHOW Missing Output Records

Bug 511646.

All OpenVMS platforms.

On busy systems, it is possible for the Oracle Rdb monitor (RDMMON) processing of “RMU/SHOW USERS” or “RMU/SHOW SYSTEM” requests to overrun the output ability of the requesting process. This may result in missing output records and the monitor's logging of COSI-F-WRITERR and SYSTEM-F-MBFULL errors to the monitor log file.

This situation can arise because the monitor process priority is generally higher than user processes. If the user process is unable to keep up with the messages that the monitor sends to it, then output information may be lost. The monitor does not wait for the user process to read the mailbox in order to prevent a hung user process doing a "RMU/SHOW" operation from hanging all Oracle Rdb operations on the entire system.

This situation has been addressed by allowing the monitor to make several attempts to send a message to the user. If the communications mailbox is full, the monitor will delay for a fraction of a second to give the user process a chance to read the communications mailbox. The monitor will then try to send the message again.

However, in order to prevent serious slow downs, the monitor will only wait for a total of 25 times during the processing of one "RMU/SHOW" request. So it is still possible for a slow (perhaps due to a low priority) process performing a "RMU/SHOW" request to experience missing information and for the monitor to log COSI-F-WRITERR and SYSTEM-F-MBFULL errors to the monitor log file. If this occurs, consider increasing the priority of the process performing the "RMU/SHOW" command.

2.1.112 Read-Only Transcations Sometimes Bugcheck in FETCH_SNAP_SEG

All OpenVMS Platforms

Read-only transactions sometimes bugchecked in FETCH_SNAP_SEG under a multi-user environment. This problem occurred when a record was stored or modified on a live data page such that storing the previous record image on the snapshot page resulted in a new snapshot page being added to the snapshot chain. If the live page was then subsequently requested from the transaction and successfully written to disk and released, then the process was abnormally terminated without writing the snapshot page to disk. This problem is evident to snapshot transactions that started before the dead process' transaction started.

This problem only occurred when the "AIJ Fast Commit" feature is disabled.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.113 SYSTEM-F-BREAK when Using RMU/BACKUP/AFTER

Bug 402614.

When backing up an AIJ file to tape, you could encounter a System-F-Break error due to the number of bytes in the final block of the AIJ file not matching the count of bytes in the final block, as taken earlier.

The problem was that we were counting the number of bytes in the records that made up the last block in the AIJ file, but we were not returning that data to the module that actually processed the records as it backed them up.

While actually backing up these final records we would check to see if the total number of bytes backed up in the last block was less than or equal to our count from the previous step. We were checking the count against a variable that did not contain the valid count.

We would only encounter the System-F-Break when the invalid variable contained a higher value for the block count, otherwise we would follow a different code path that allowed us to complete the backup.

This would only occur with AIJ backups to tape, as backup to disk does block transfers and therefore never checks the number of bytes in the final record.

This problem has been corrected in Oracle Rdb Version 7.0.1.

This has now been corrected, so that the AIJ backup to tape correctly returns the number of bytes in the final record for comparison.

2.1.114 TSNs are Reused Following Process Failure

Bug 498800

Following an application process failure, it was possible for transaction sequence numbers ("TSNs") to appear to be re-used. This problem occurred when:

- A process failure resulted in the database recovery process being invoked to recover the failed process.
- The process failed while either starting a transaction or committing (or aborting) a transaction.
- The failed process was the only application process accessing the database and no other processes, including the database recovery process, have attached to the database since it was originally opened.

This problem has been corrected in Oracle Rdb Version 7.0.1.

The database recovery process now correctly preserves transaction sequence numbers upon recovery of an application process failure.

2.1.115 Problems in SQL Functions (SQL_FUNCTIONS.SQL) Have Been Fixed

The following problems have been fixed in Oracle Rdb Version 7.0.1.

- Bug 427178.

  REPLACE() function returns error, if the character set is 'DEC_KANJI'.

- Bug 427180.

  LTRIM function returns error, if the character set is 'DEC_KANJI'

- Bug 426921.

  Oracle Rdb built-in function 'SUBSTRING' returns error, if the character set is 'DEC_KANJI' and the character length is 'CHARACTER'.

- Bug number: 427174

  The procedure 'SYS$LIBRARY:SQL_FUNCITONS70.SQL' produces an access violation if the character set of the session is 'DEC_KANJI'.

In order for these functions to work correctly, the character set must be set properly if it is other than DEC_MCS (such as DEC_KANJI). This must be done before executing 'SQL_FUNCITONS70.SQL' and invoking the special SQL functions defined therein.

2.1.116 DEC_Kanji and Shift_JIS characters May be Specified as Table and Column Names

The following problems have been fixed in Oracle Rdb Version 7.0.1.

- Bug 427924.

  The User cannot specify Shift_JIS characters as metadata whose second octet is special ASCII symbol characters, which are symbols in the ASCII table code range between 0X40-0X7F.

  SQL now supports Shift_JIS characters as metadata.

- Bug 420939.

SQL did not support single-octet Katakana characters in DEC_KANJI character set as table name or column name.

SQL now supports Katakana characters as table and column names.

2.1.117 Internal Error in RMU/EXTRACT/ITEM=VIEW

Bugs 446855 and 427958.

A view definition in an AS SELECT clause with COUNT(FIELD) when performing a RMU/EXTRACT/ITEM=VIEW produced the following error:

```
%RMU-F-BLRINV, internal error - BLR string 23 for V2. is invalid
RDMS-E-BAD_CODE, corruption in the query string.
```

```
SQL> create data file test;
SQL> create table t1(f1 integer, f2 integer);
SQL> create view v1 (f1, cnt) as select f1,count(f2) from t1 group by f1;
SQL> create view v2 (f1, cnt) as select f1,count(distinct f2) from t1
group by f1;
commit;
```

Entering RMU/EXTRACT/ITEM=VIEW produced:

```
create view V1
    (F1,
     CNT) as
    select
%RMU-F-BLRINV, internal error - BLR string 23 for V1. is invalid
-RDMS-E-BAD_CODE, corruption in the query string
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

Now RMU/EXTRACT/ITEM=VIEW will produce the correct result:

```
create view V1
    (F1,
     CNT) as
    select
        C2.F1,
        count(C2.F2)
    from T1 C2
    group by C2.F1;

create view V2
    (F1,
     CNT) as
    select
        C2.F1,
        count(distinct C2.F2)
    from T1 C2
    group by C2.F1;

commit work;
```

2.1.118 RMU/SHOW STATISTIC Fails with Invalid Time After Multiple "Resets"

Bug 433963

It was possible for the **RMU/SHOW STATISTIC** utility to fail with an invalid-time error if the "Reset" key was pressed several times before the initial screen is displayed.

This problem did not occur once the first screen has been displayed.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.119 RMU/UNLOAD/RECORD=(FORMAT=DELIM) May Fail

Bug 505973.

If a table had a computed field defined the RMU/UNLOAD /RECORD=(FORMAT=DELIM) command might fail with an ACCVIO error and a bugcheck dump at RMUUNL$NAME_LEN.

The following example shows this error:

```
$ rmu/unload/record=(file=rec,format=delim,null) db tab rec.unl
%RMU-E-OUTFILDEL, Fatal error, output file deleted
-SYSTEM-F-ACCVIO, access violation, reason mask=05, virtual address=1286127C, PC
=001AAE99, PSL=03C00000
%RMU-I-BUGCHKDMP, generating bugcheck dump file DISK:[DIR]RMUBUGCHK.DMP;

***** Exception at 001AAE99 : RMUUNL$NAME_LEN + 0000008E
%SYSTEM-F-ACCVIO, access violation, reason mask=05,
virtual address=1286127C, PC=001AAE99, PSL=03C00000
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.120 Bugcheck Dump After an Attach or Open of a Database

Bug 490247.

Under certain timing conditions an ATTACH or OPEN of a database after a RMU/CLOSE/ABORT=DELPRC/NOWAIT command could cause a DBR bugcheck at DBR$RECOVER_RCACHE+6EE with the following error message:

```
***** Exception at 000185BC : DBR$RECOVER_RCACHE +000006EE
%RDMS-F-FILACCERR, error opening cache backing store file <rdc file>
-RMS-E-FNF, file not found
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.121 A RMU/EXTRACT/ITEM=VIEW Command Produces an Internal Error

A View Definition containing CASE statements produced the following error when the RMU/EXTRACT/ITEM=VIEW command was used:

```
%RMU-F-BLRINV, internal error - BLR string 26 for VIEW_1. is invalid
-RDMS-E-BAD_CODE, corruption in the query string
```

The following is an example:

```
SQL> sh view  VIEW_1
Columns for view VIEW_1:
Column Name   Data Type  Domain
-----------   ---------  ------
x1                              BIGINT
x2                              CHAR(2)
x3                              BIGINT(2)
x4                              BIGINT(2)
x5                              BIGINT(2)
x6                              BIGINT(2)
x7                              BIGINT(2)
SELECT DISTINCT
         x1,
         x2,
         SUM(y1) AS x3,
         SUM(y2) AS x4,
         SUM(y3) AS x5,
         SUM(y4) AS x6,
         SUM(y5) AS x7
     FROM
         (SELECT DISTINCT x1, x2, SUM(x8) AS y1,
         CASE x9 WHEN 0 THEN SUM(x8) ELSE 0 END AS y2,
         CASE x9 WHEN 3 THEN SUM(x8) ELSE 0 END AS y3,
         CASE x9 WHEN 6 THEN SUM(x8) ELSE 0 END AS y4,
         CASE x9 WHEN 9 THEN SUM(x8) ELSE 0 END AS y5
         FROM TABLE_1
         GROUP BY x1, x2, x9)
         AS NEW_GRP (x1, x2, y1, y2, y3, y4, y5)
         GROUP BY x1, x2
```

A RMU/EXTRACT/ITEM=VIEW BUG command produced the following error:

```
create view VIEW_1
    (x1,
     x2,
     x3,
     x4,
     x5,
     x6,
     x7) as
    select
%RMU-F-BLRINV, internal error - BLR string 26 for VIEW_1. is invalid
-RDMS-E-BAD_CODE, corruption in the query string
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

Now a RMU/EXTRACT/ITEM=VIEW command for this view produces:

```
create view VIEW_1
    (x1,
     x2,
     x3,
     x4,
     x5,
     x6,
     x7) as
    select
        C2.F1,
        C2.F2,
        sum(C2.F3) ,
        sum(C2.F4) ,
        sum(C2.F5) ,
        sum(C2.F6) ,
        sum(C2.F7)
    from
    (SELECT
        distinct C4.x1, C4.x2, sum(C4.x8) ,
        case
            when (C4.x9= 0) then sum(C4.x8)
            else 0
        end,
        case
            when (C4.x9= 3) then sum(C4.x8)
            else 0
        end,
        case
            when (C4.x9= 6) then sum(C4.x8)
            else 0
        end,
        case
            when (C4.x9= 9) then sum(C4.x8)
            else 0
        end
        from table_1 C4
        group by C4.x1, C4.x2, C4.x9) as C2 ( F1, F2,
        F3, F4, F5, F6, F7 )
    group by C2.F1, C2.F2;

commit work;
```

2.1.122 Virtual Pages Increase with Use of Temporary Tables

Bug 466313.

When using temporary tables, all the data is stored in virtual memory. In addition, each table needs recovery space to hold before images of deleted, and updated rows. Therefore, after a DELETE FROM temporary table there will be two copies of the data held in virtual memory until a user commits. This behavior is necessary in case the user decides to rollback the transaction and may result in a higher overall usage of virtual memory.

A problem was observed in Oracle Rdb 7.0 and Rdb 7.0 ECO1, where even after a commit was issued the virtual memory continued to grow. This problem occurred because deleted rows were not being correctly purged at commit time.

Increasing page file quota may be a workaround to this problem or creating the temporary table with "delete on commit" attributes.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.123 IMPORT fails with invalid value for ABW batch size

Bug 475932.

In prior versions of Oracle Rdb IMPORT may fail with an error such as:

```
-RDB-E-BAD_DPB_CONTENT, invalid database parameters in the database
parameter block (DPB)
-RDMS-F-BADPARAM, asynchronous batch-write maximum batch size (1) is out of
valid range (2..524288)
```

This problem occurs when the number of buffers for the database is set very small (such as 5). The number of buffers is used to calculate the default batch size for the asynchronous batch-write, and in prior versions of Rdb this resulted in a value smaller than allowed being saved in the interchange file (RBR) by the EXPORT command.

Oracle recommends that the NUMBER OF BUFFERS for all databases be set to a value greater than or equal to 10 to avoid this problem, or ensure that the MAX BUFFER values for the ASYNC BATCH WRITE clause be 2 or more. If the interchange file already has this problem then the ASYNC BATCH WRITE clause can be used on the IMPORT command to override the value recorded in the RBR file.

This problem has been corrected by this release of Oracle Rdb. IMPORT and CREATE/ALTER DATABASE now ensure that all values for buffer sizes are at least as large as the minimum allowed by Oracle Rdb.

2.1.124 CAST Setting the Wrong VARCHAR Length

Bug 399199.

In prior versions of Oracle Rdb it was possible to CAST a numeric value to a small VARCHAR string and have the length set incorrectly. This occurred when the numeric value contained more digits than would fit in the small VARCHAR string. For example, CAST(123 as VARCHAR(1)) would set the length to 3 even though it should have been limited to 1. The CHAR_LENGTH function can be used to confirm this problem.

This problem has been corrected in this release of Oracle Rdb. The length is now set correctly.

2.1.125 DBR Failure Trying to REDO Duplicate DBKEY Entries

In certain cases where DBR was attempting to REDO a failed process' transaction information, it was possible for DBR to fail because the first change of a duplicate DBKEY modification could not be successfully applied.

The problem occurred when there is concurrent contention for the same page by multiple processes.

The following example shows a dump of an AIJ journal containing multiple entries for the same DBKEY.

```
710/1858         TYPE=D, LENGTH=482, TAD=10-JUL-1997 13:18:12.38, CSM=00
    TID=9, TSN=0:292, AIJBL_START_FLG=0, SEQUENCE=148
    Appending to partial AIJBL

 MODIFY: PDBK=2:110:8, LDBID=0, PSN=7, LENGTH=133

[...unrelated AIJ information removed for clarity...]

714/1884         TYPE=D, LENGTH=334, TAD=10-JUL-1997 13:19:22.08, CSM=00
    TID=9, TSN=0:292, AIJBL_START_FLG=0, SEQUENCE=151
    Appending to partial AIJBL
```

```
MODIFY: PDBK=2:110:8, LDBID=0, PSN=15, LENGTH=29
```

The problem occurred when the DBR process is unable to apply the first change, typically because the page does not contain adequate freespace, but can apply the second change.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.126 Bugcheck Invoking Trigger or Constraint

Bugs 326619 and 313909

Invoking a trigger or constraint, by inserting a new row into a table, for example, would sometimes result in a bugcheck with an access violation exception at RDMS$$APPLY_RULES.

The following example shows a trigger definition that would result in a bugcheck, when the insert operation was executed.

```
create table table2 (cc char(6), da char(4), aa char(8),
                     cst computed by da||aa);
create table table3 (pt char(3), pp char(3), cst_1 char(6), cst_2 char(8),
                     cst computed by (cst_1||cst_2));
create table table4 (o1 char(6));
create trigger trig1 after insert on table3
    (insert into table4 (o1)
     values ((select distinct (select distinct cc from table2 c1
                               where c1.cst = table3.cst)
              from table3 i1 where i1.dbkey = table3.dbkey))
     for each row;
insert into table3 (pt, pp) values ('AAA','BBB');
```

In the bugcheck dump, the error reported was:

```
RDMS$$APPLY_RULES +189 SYSTEM-F-ACCVIO, access violation
```

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.127 Query with Zig-Zag Match Returns the Wrong Result

Bug 513298.

The following query with zig-zag match strategy generates incorrect results. The correct number of rows selected should be 1.

```
select * from TAB1 T1, TAB2 T2, TAB3 T3
where   T1.COL3 = T2.COL1
 and  T2.COL2 = T3.COL3
 and    T3.COL1 = 'ABCDE'
 and  T1.COL2 = 93901
 and  T1.COL1 = 4915 ;
Conjunct
Match
  Outer loop
    Sort
    Cross block of 2 entries
      Cross block entry 1
        Get     Retrieval by index of relation TAB1
          Index name  INX_TAB1 [2:2]    Direct lookup
      Cross block entry 2
        Get     Retrieval by index of relation TAB2
          Index name  INX_TAB2 [1:1]         Direct lookup
  Inner loop      (zig-zag)
    Index only retrieval of relation TAB3
      Index name  INX_TAB3 [1:1]
0 rows selected
```

In Oracle Rdb7, an enhancement was made in zig-zag match strategy to skip both the inner and outer loops, rather than the inner loop as was done in prior version of Rdb.

Along with this enhancement, changes were also made to enable better optimization, to rescan the duplicate keys in the inner loop via a temporary table, rather than rescanning the index. Due to this change a bug was introduced into the code which copies the duplicate keys to the temporary table, rather than rescanning the index.

This problem has been fixed in Oracle Rdb Version 7.0.1.

2.1.128 The SET TRANSACTION MODE Clause of a CREATE/ALTER DATABASE Is Not EXPORTED/IMPORTED.

Bug 428310.

In Rdb 7.0, the SET TRANSACTION MODE clause of a CREATE or ALTER DATABASE statement was not Exported or imported correctly. The same was true for the ALTER TRANSACTION MODE clause of a CREATE or ALTER DATABASE statement. Thus, in Rdb 7.0, the transaction mode clause could not be exported or imported. This was a restriction.

The following example shows a database created with the SET TRANSACTION MODE clause. If this database is exported or imported, the transaction mode information is lost. This is a restriction.

```
CREATE DATABASE FILENAME TRANSMODE_TEST
SET TRANSACTION MODE (NO BATCH UPDATE);
```

This restriction has been lifted in Oracle Rdb 7.0.1.

2.1.129 RMU/EXTRACT Problem when Generating RDO Scripts

When generating an RDO script invalid syntax could be generated for storage maps of segmented strings. A semicolon is missed from the script.

The workaround is to use a SQL script or to edit the RDO script and add the semicolon.

Here is an example of the problem:

```
RDO> ! RMU/EXTRACT for Oracle Rdb V7.0-01
RDO> !
RDO> !                              Storage Map Definitions
RDO> !
RDO>
.
.
RDO> define storage map LARGE_OBJECT_STORAGE
cont>     description is
cont>     /* Keep segmented strings in separate areas */
cont>     for SEGMENTED STRINGS
cont>         store within
cont>         MODEL_DB_1 for RELATION_1, EMPLOYEE.MANAGERS_COMMENTS
cont>         MODEL_DB_2 for EMPLOYEE.EMPLOYEE_PHOTO
        MODEL_DB_2 for EMPLOYEE.EMPLOYEE_PHOTO
        ^
%RDO-W-LOOK_FOR_STT, syntax error, looking for:
%RDO-W-LOOK_FOR_CON,            ,, THRESHOLDS, ;, NO, ENABLE, DEFAULT,
%RDO-W-LOOK_FOR_CON,            DISABLE, PLACEMENT, REORGANIZE, STORE,
%RDO-W-LOOK_FOR_CON,            END,
%RDO-F-LOOK_FOR_FIN,    found MODEL_DB_2 instead
RDO>         MODEL_DB_3;
        MODEL_DB_3;
        ^
%RDO-F-LOOK_FOR, syntax error, looking for a valid RDO statement, found MODEL_D
RDO>     end.
    end.
    ^
%RDO-F-LOOK_FOR, syntax error, looking for a valid RDO statement,
found END ins
```

The problem has been fixed in Oracle Rdb Version 7.0.1.

2.1.130 SQL Queries on the SHOW MODULE Command Causing Poor Performance

When a SHOW MODULE command was specified, SQL was fetching all routines, rather than fetching just the routines for the desired module. This has been corrected for performance reasons.

This problem has been corrected in Oracle Rdb Version 7.0.1.

2.1.131 SQL Module Language Does Not Align CDD Records

Bug 389762.

SQL module language would not align the fields of records that were defined using CDD. The records would be unaligned regardless of the setting of the command line qualifiers /ALIGN_RECORDS or -align.

Be aware that correcting this problem may present a new problem for SQL module language applications with language C using CDD records on Alpha VMS. The default for SQL module language with language C on Alpha VMS is /ALIGN_RECORDS, so applications using CDD records will now be aligned. If all records in the SQL module language file are defined by CDD, this can be worked around by compiling with the /NOALIGN_RECORDS qualifier.

This problem has been corrected in Oracle Rdb 7.0.1.

3
Documentation Corrections

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

3.1 Documentation Corrections

3.1.1 Accessing Master Index Information for the Oracle Rdb V7.0 Documentation Set

In V7.0, the entire Oracle Rdb documentation set was provided in Adobe Acrobat's Portable Document Format (PDF) on the Rdb Client kits CD–ROM in the directory RDBDOC. See the Oracle Rdb7 Release Notes for additional information.

Included with this set of PDF manuals was a master index built with Adobe Acrobat Catalog. You will need Adobe Acrobat Exchange to view the master index built with Adobe Acrobat Catalog.

3.1.2 Documentation Order Number Corrected

The order number for the library set for Oracle Rdb7 for OpenVMS is listed incorrectly in the V7.0 Release Notes. The correct number is A45807-1.

3.1.3 Documentation Corrections in the Oracle Rdb SQL Reference Manual for Rdb7

The Oracle Rdb SQL Reference Manual for Rdb7 contained an error in the syntax diagram when describing the SET FLAGS and the CREATE STORAGE MAP statements. This release note corrects the syntax as well as provides more examples for these statements.

3.1.3.1 Incorrect Syntax for SET FLAGS Statement

The SET FLAGS statement allows enabling and disabling of database system debug flags for the current session.

The literal or host variable passed to this command can contain a list keywords, or negated keywords separated by commas. Spaces are ignored. The keywords may be abbreviated to an unambiguous length.

Note

Oracle Corporation reserves the right to add new keywords to the SET FLAGS command in any release or update to Oracle Rdb which may change this unambiguous length. Therefore, it is recommended that the full keyword be used in applications.

The syntax for this SET statement is show below.

```
SET FLAGS --+--> literal --------+-->
            |                    |
            +--> host-variable --+

SET NOFLAGS ----------------------->
```

The SET NOFLAGS statement disables all currently enabled flags. It is equivalent to SET FLAGS 'NONE'.

The following examples demonstrate the use of host variables in interactive SQL, as well as literal strings with multiple options to enable and disable flags. The same SET FLAGS commands can also be used in dynamic SQL.

```
SQL> show flags

Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   PREFIX
SQL> -- declare a host variable to be used with SET FLAGS
SQL> declare :hv char(40);
SQL> -- assign a value to the variable
SQL> begin
cont> set :hv = 'strategy, outline';
cont> end;
SQL> -- use the host variable to enable or disable flags
SQL> set flags :hv;
SQL> show flags

Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   STRATEGY,PREFIX,OUTLINE
SQL> -- use a string literal directly with the SET FLAGS statement
SQL> set flags 'noprefix,execution(10)';
SQL> show flags

Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   STRATEGY,OUTLINE,EXECUTION(10)
```

Please refer to the Oracle Rdb SQL Reference Manual for further examples.

3.1.3.2 Incorrect Syntax for CREATE STORAGE MAP

The syntax diagram the **partition-clause** incorrectly indicated that the STORE keyword was not repeated. When creating a vertically partitioned table you must repeat the STORE keyword for each partition.

```
partition-clause =
-+-> STORE +-----------------+-+--------------------+-> store-clause -+->
 |         |                 | |                    |                 |
 |         +-> columns-clause-+ +-> store-attributes -+                 |
 |                                                                      |
 +----------------------------------------------------------------------+
```

The following storage map shows an alternate mapping for the EMPLOYEES table in the same MF_PERSONNEL database.

```
SQL> create storage map EMPLOYEES_MAP
cont>     for EMPLOYEES
cont>     placement via index EMPLOYEES_HASH
cont>     -- store the primary information horizontally partitioned
cont>     -- across the areas EMPIDS_LOW, EMPIDS_MID and EMPIDS_OVER
cont>     -- disable compress because these columns are accessed often
cont>     store
cont>         columns (EMPLOYEE_ID, LAST_NAME,
cont>                  FIRST_NAME, MIDDLE_INITIAL)
cont>         disable compression
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
cont>
cont>     -- place all the address information in EMP_INFO
cont>     -- make sure these character columns are compressed
cont>     -- to remove the trailing spaces
cont>     store
cont>         columns (ADDRESS_DATA_1, ADDRESS_DATA_2, CITY, STATE,
cont>                  POSTAL_CODE)
cont>         enable compression
cont>         in EMP_INFO
cont>
cont>     -- the remaining columns get
cont>     -- written randomly over these areas
cont>     store
cont>         enable compression
cont>         randomly across (SALARY_HISTORY, JOBS);
```

Please refer to the Oracle Rdb SQL Reference Manual for the full syntax of the CREATE STORAGE MAP statement.

3.1.4 Incorrect Description of THRESHOLDS Setting for Duplicate Indexes

The following is a revised section from the manual Oracle Rdb7 Guide to Database Performance and Tuning.

3.1.4.1 AIP Length Problems in Indexes that Allow Duplicates

When an index allows duplicates, the length stored in the AIP will be 215 bytes, regardless of the actual index node size. Because an index with duplicates can have variable node sizes, the 215-byte size is used as a median length to represent the length of rows in the index's logical area.

When the row size in the AIP is less than the actual row length, it is highly likely that SPAM entries will show space is available on pages when they have insufficient space to store another full size row. This is the most common cause of insert performance problems.

For example, consider a case where an index node size of 430 bytes (a common default value) is used; the page size for the storage area where the index is stored is 2 blocks. After deducting page overhead, the available space on a 2-block page is 982 bytes. Assume that the page in this example is initially empty.

1. A full size (430-byte) index node is stored. As 8 bytes of overhead are associated with each row stored on a page, that leaves 982—430—8 = 544 free bytes remaining on the page.

2. A duplicate key entry is made in that index node and thus a duplicate node is created on the same page. An initial duplicate node is 112 bytes long (duplicate nodes can have a variety of sizes depending on when they are created, but for this particular example, 112 bytes is used). Therefore, 544—112—8 = 424 free bytes remain on the page.

At this point, 424 bytes are left on the page. That is greater than the 215 bytes that the AIP shows as the row length for the logical area, so the SPAM page shows that the page has space available. However, an attempt to store a full size index node on the page will fail, because the remaining free space (424 bytes) is not enough to store a 430-byte node.

In this case, another candidate page must be selected via the SPAM page, and the process repeats until a page that truly has sufficient free space available is found. In a logical area that contains many duplicate nodes, a significant percentage of the pages in the logical area may fit the scenario just described. When that is the case, and a new full size index node needs to be stored, many pages may need to be read and checked before one is found that can be used to store the row.

It is possible to avoid the preceding scenario by using logical area thresholds. The goal is to set a threshold such that the SPAM page will show a page is full when space is insufficient to store a full size index node.

Using the previous example, here is how to properly set logical area thresholds to prevent excessive pages checked on an index with a 430-byte node size that is stored on a 2-block page. To calculate the proper threshold value to use, you must first determine how full the page can get before no more full size nodes will fit on the page. In this example, a database page can have up to 982—430—8 = 544 bytes in use before the page is too full. Therefore, if 544 or fewer bytes are in use, then enough space remains to store another full size node. The threshold is then 544 / 982 = .553971, or 55%.

In addition, you can determine how full a page must be before a duplicate node of size 112 will no longer fit. In this example, a database page can have up to 982—112—8 = 862 bytes in use before the page is too full. Therefore, if 862 or fewer bytes are in use, then enough space remains to store another small duplicates node. The threshold is then 862 / 982 = .8778, or 88%.

Here is an example of creating an index with the above characteristics:

```
SQL> CREATE INDEX TEST_INDEX ON EMPLOYEES (LAST_NAME)
cont>     STORE IN RDB$SYSTEM
cont>     (THRESHOLD IS (55, 55, 88));
```

These settings mean that any page at over 55% full will not be fetched when inserting a full index node, however, it may be fetched when inserting the smaller duplicates node. When the page is over 88% full then neither a full node nor a duplicate node can be stored, so the page is set as FULL. The lowest setting is not used and so can be set to any value less than or equal to the lowest used threshold.

Note that the compression algorithm used on regular tables that have compression enabled does not apply to index nodes. Index nodes are not compressed like data rows and will always utilize the number of bytes that is specified in the node size. Do not attempt to take into account a compression factor when calculating thresholds for indexes.

3.1.4.2 SET FLAGS Option IGNORE_OUTLINE not Available

Bug 510968

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

3.1.5 Documentation Corrections in the Oracle Rdb Guide to SQL Programming

3.1.5.1 Additional Information About Detached Processes

Oracle Rdb documentation omits necessary detail on running Oracle Rdb from a detached process.

Applications run from a detached process must ensure that the OpenVMS environment is established correctly before running Oracle Rdb. Otherwise, Oracle Rdb will not execute.

Attempts to attach to a database and execute an Oracle Rdb query from applications running as detached processes will result in an error similar to the following:

```
%RDB-F-SYS_REQUEST, error from system services request
-SORT-E-OPENOUT, error opening !AS as output
-RMS-F-DEV, error in device name or inappropriate device type for
operation
```

The problem occurs because a detached process does not normally have the logical names SYS$LOGIN or SYS$SCRATCH defined.

There are two methods that can be used to correct this:

1. Use the DCL command procedure RUN-PROCEDURE to run the ACCOUNTS application:

   ```
   $ RUN/DETACH/AUTHORIZE SYS$SYSTEM:LOGINOUT/INPUT=RUN-PROCEDURE
   $ RUN ACCOUNTS_REPORT
   ```

   This solution executes SYS$SYSTEM:LOGINOUT so that the default command language interface (CLI) is activated (this is usually DCL). This causes the logical names SYS$LOGIN and SYS$SCRATCH to be defined for the detached process. The /AUTHORIZE qualifier also ensures that the users' process quota limits (PQLs) are used from the system authorization file rather than relying on the default PQL system parameters, which are often insufficient for Oracle Rdb.

2. If DCL is not desired, and SYS$LOGIN and SYS$SCRATCH are not defined, then prior to executing any Oracle Rdb statement, you must define the following logical names:

   - RDMS$BIND_WORK_FILE

     Define this logical name to allow you to reduce the overhead of disk I/O operations for matching operations when used in conjunction with the RDMS$BIND_WORK_VM logical name.

     For more information on RDNS$BIND_WORK_FILE, see the Oracle Rdb Guide to Database Performance and Tuning.

- SORTWORK0, SORTWORK1, and so on

  The OpenVMS sort/merge utility (SORT/MERGE) attempts to create sort work files in SYS$SCRATCH. If the SORTWORK logical names exist, the utility will not require the SYS$SCRATCH logical. However, note that not all queries will require sorting, and that some sorts will be completed in memory and so will not necessarily require disk space.

  If you use the logical RDMS$BIND_SORT_WORKFILES, you will need to define further SORTWORK logical names as described in the Oracle Rdb Guide to Database Performance and Tuning.

  You should also verify that sufficient process quotas are specified on the RUN/DETACH command line, or defined as system PQL parameters to allow Oracle Rdb to execute.

3.1.6 Documentation: Clarification on Updates to the RDB$LAST_ALTERED Column for the RDB$DATABASE table

The ALTER DATABASE statement can be used to change many database attributes, however, only those listed below will cause the RDB$DATABASE table to be changed. The column RDB$LAST_UPDATED is used to record the date and time when the system table RDB$DATABASE is updated and so will change when any of the following clauses are used by ALTER DATABASE.

- DICTIONARY IS [NOT] REQUIRED
- DICTIONARY IS NOT USED
- MULTISCHEMA IS { ON | OFF }
- WORKLOAD COLLECTION IS { ENABLED | DISABLED }
- CARDINALITY COLLECTION IS { ENABLED | DISABLED }
- METADATA CHANGES ARE { ENABLED | DISABLED }

In addition any GRANT and REVOKE statements which use the ON DATABASE clause will cause the RDB$LAST_UPDATED column to be updated for RDB$DATABASE.

3.1.7 Missing Descriptions of RDB$FLAGS from HELP file

The HELP file for Oracle Rdb7 describes the system tables for Oracle Rdb and was missing these updated descriptions of the RDB$FLAGS column for several tables.

Table 3–1 Changed Columns for RDB$INDICES Table

Column Name	Data Type	Domain Name	Comments
RDB$FLAGS	integer	RDB$FLAGS	A bit mask where the bits have the following meaning when set: • Bit 0: This index is of type HASHED. • Bit 1: This index uses the MAPPING VALUES clause to compress integer value ranges. • Bit 2: If this is a HASHED index then it is of type ORDERED. If clear this indicates the index if of type SCATTERED. • Bit 3: Reserved for future use. • Bit 4: This index has run length compression enabled (ENABLE COMPRESSION). • Bit 5: This index is no longer used (MAINTENANCE IS DISABLED). • Bit 6 through 10: Reserved for future use. • Bit 11: This index has duplicates compressed (DUPLICATES ARE COMPRESSED). • Bit 12: This index is of type SORTED RANKED. • Bits 13 through 31: Reserved for future use.

Table 3–2 Changed Columns for RDB$RELATIONS Table

Column Name	Data Type	Domain Name	Comments
RDB$FLAGS	integer	RDB$FLAGS	A bit mask where the bits have the following meaning when set: • Bit 0: This relation is a view. • Bit 1: This relation is **not** compressed. • Bit 2: The SQL clause, WITH CHECK OPTION, is used in this view definition. • Bit 3: Indicates a special internal system relation. • Bit 4: This view is not an ANSI updatable view. • Bit 5: This is an imported table in the Distributed Option for Rdb catalog. • Bit 6: This is a passthru table in the Distributed Option for Rdb catalog. • Bit 7: This is a partitioned view in the Distributed Option for Rdb catalog. • Bit 8: This table has compression defined by the storage map. When set Bit 1 in this bit mask is ignored. • Bit 9: This is a temporary table. • Bit 10: When bit 9 is set this is a global temporary table, when clear it indicates a local temporary table. • Bit 11: When bit 9 is set this indicates that the rows in the temporary table should be deleted upon COMMIT. • Bit 12: Reserved for future use. • Bit 13: A table (via a computed by column) or view references a local temporary table. • Bit 14: Reserved for future use. • Bit 15: This is a system table with a special storage map. • Bits 16 through 31: Reserved for future use.

Table 3–3 Changed Columns for RDB$STORAGE_MAPS Table

Column Name	Data Type	Domain Name	Comments
RDB$FLAGS	integer	RDB$FLAGS	A bit mask where the bits have the following meaning when set: • Bit 0: This table or index is mapped to page format MIXED areas. • Bit 1: This partition is **not** compressed. • Bit 2: This is a strictly partitioned storage map, the partitioning columns become read only for UPDATE. • Bit 3 through 31: Reserved for future use.

3.1.8 Missing Tables Descriptions for the RDBEXPERT Collection Class

All OpenVMS platforms.

Appendix B in the Oracle Rdb7 Guide to Database Performance and Tuning describes the event-based data tables in the formatted database for the Oracle Rdb PERFORMANCE and RDBEXPERT collection classes. This section describes the missing tables for the RDBEXPERT collection class.

Table 3–4 shows the TRANS_TPB table.

Table 3–4 Columns for Table EPC$1_221_TRANS_TPB

Column Name	Data Type	Domain
COLLECTION_RECORD_ID	SMALLINT	COLLECTION_RECORD_ID_ DOMAIN
IMAGE_RECORD_ID	INTEGER	IMAGE_RECORD_ID_DOMAIN
CONTEXT_NUMBER	INTEGER	CONTEXT_NUMBER_DOMAIN
TIMESTAMP_POINT	DATE VMS	
CLIENT_PC	INTEGER	
STREAM_ID	INTEGER	
TRANS_ID	VARCHAR(16)	
TRANS_ID_STR_ID	INTEGER	STR_ID_DOMAIN
TPB	VARCHAR(127)	
TPB_STR_ID	INTEGER	STR_ID_DOMAIN

Table 3–5 shows the TRANS_TPB_ST table. An index is provided for this table. It is defined with column STR_ID, duplicates are allowed, and the type is sorted.

Table 3–5 Columns for Table EPC$1_221_TRANS_TPB_ST

Column Name	Data Type	Domain
STR_ID	INTEGER	STR_ID_DOMAIN
SEGMENT_NUMBER	SMALLINT	SEGMENT_NUMBER_DOMAIN
STR_SEGMENT	VARCHAR(128)	

3.1.9 Missing Columns Descriptions for Tables in the Formatted Database

Some of the columns were missing from the tables in Appendix B in the Oracle Rdb7 Guide to Database Performance and Tuning. The complete table definitions are described in this section.

Table 3–6 shows the DATABASE table.

Table 3–6 Columns for Table EPC$1_221_DATABASE

Column Name	Data Type	Domain
COLLECTION_RECORD_ID	SMALLINT	COLLECTION_RECORD_ID_ DOMAIN
IMAGE_RECORD_ID	INTEGER	IMAGE_RECORD_ID_DOMAIN
CONTEXT_NUMBER	INTEGER	CONTEXT_NUMBER_DOMAIN
TIMESTAMP_POINT	DATE VMS	
CLIENT_PC	INTEGER	
STREAM_ID	INTEGER	
DB_NAME	VARCHAR(255)	
DB_NAME_STR_ID	INTEGER	STR_ID_DOMAIN
IMAGE_FILE_NAME	VARCHAR(255)	
IMAGE_FILE_NAME_STR_ID	INTEGER	STR_ID_DOMAIN

Table 3–7 shows the REQUEST_ACTUAL table.

Table 3–7 Columns for Table EPC$1_221_REQUEST_ACTUAL

Column Name	Data Type	Domain
COLLECTION_RECORD_ID	SMALLINT	COLLECTION_RECORD_ID_ DOMAIN
IMAGE_RECORD_ID	INTEGER	IMAGE_RECORD_ID_DOMAIN
CONTEXT_NUMBER	INTEGER	CONTEXT_NUMBER_DOMAIN
TIMESTAMP_START	DATE VMS	
TIMESTAMP_END	DATE VMS	
DBS_READS_START	INTEGER	
DBS_WRITES_START	INTEGER	
RUJ_READS_START	INTEGER	
RUJ_WRITES_START	INTEGER	
AIJ_WRITES_START	INTEGER	
ROOT_READS_START	INTEGER	
ROOT_WRITES_START	INTEGER	
BUFFER_READS_START	INTEGER	
GET_VM_BYTES_START	INTEGER	
FREE_VM_BYTES_START	INTEGER	
LOCK_REQS_START	INTEGER	

(continued on next page)

Table 3–7 (Cont.) Columns for Table EPC$1_221_REQUEST_ACTUAL

Column Name	Data Type	Domain
REQ_NOT_QUEUED_START	INTEGER	
REQ_STALLS_START	INTEGER	
REQ_DEADLOCKS_START	INTEGER	
PROM_DEADLOCKS_START	INTEGER	
LOCK_RELS_START	INTEGER	
LOCK_STALL_TIME_START	INTEGER	
D_FETCH_RET_START	INTEGER	
D_FETCH_UPD_START	INTEGER	
D_LB_ALLOK_START	INTEGER	
D_LB_GBNEEDLOCK_START	INTEGER	
D_LB_NEEDLOCK_START	INTEGER	
D_LB_OLDVER_START	INTEGER	
D_GB_NEEDLOCK_START	INTEGER	
D_GB_OLDVER_START	INTEGER	
D_NOTFOUND_IO_START	INTEGER	
D_NOTFOUND_SYN_START	INTEGER	
S_FETCH_RET_START	INTEGER	
S_FETCH_UPD_START	INTEGER	
S_LB_ALLOK_START	INTEGER	
S_LB_GBNEEDLOCK_START	INTEGER	
S_LB_NEEDLOCK_START	INTEGER	
S_LB_OLDVER_START	INTEGER	
S_GB_NEEDLOCK_START	INTEGER	
S_GB_OLDVER_START	INTEGER	
S_NOTFOUND_IO_START	INTEGER	
S_NOTFOUND_SYN_START	INTEGER	
D_ASYNC_FETCH_START	INTEGER	
S_ASYNC_FETCH_START	INTEGER	
D_ASYNC_READIO_START	INTEGER	
S_ASYNC_READIO_START	INTEGER	
AS_READ_STALL_START	INTEGER	
AS_BATCH_WRITE_START	INTEGER	
AS_WRITE_STALL_START	INTEGER	
BIO_START	INTEGER	
DIO_START	INTEGER	
PAGEFAULTS_START	INTEGER	
PAGEFAULT_IO_START	INTEGER	
CPU_START	INTEGER	

(continued on next page)

Table 3–7 (Cont.) Columns for Table EPC$1_221_REQUEST_ACTUAL

Column Name	Data Type	Domain
CURRENT_PRIO_START	SMALLINT	
VIRTUAL_SIZE_START	INTEGER	
WS_SIZE_START	INTEGER	
WS_PRIVATE_START	INTEGER	
WS_GLOBAL_START	INTEGER	
CLIENT_PC_END	INTEGER	
STREAM_ID_END	INTEGER	
REQ_ID_END	INTEGER	
COMP_STATUS_END	INTEGER	
REQUEST_OPER_END	INTEGER	
TRANS_ID_END	VARCHAR(16)	
TRANS_ID_END_STR_ID	INTEGER	STR_ID_DOMAIN
DBS_READS_END	INTEGER	
DBS_WRITES_END	INTEGER	
RUJ_READS_END	INTEGER	
RUJ_WRITES_END	INTEGER	
AIJ_WRITES_END	INTEGER	
ROOT_READS_END	INTEGER	
ROOT_WRITES_END	INTEGER	
BUFFER_READS_END	INTEGER	
GET_VM_BYTES_END	INTEGER	
FREE_VM_BYTES_END	INTEGER	
LOCK_REQS_END	INTEGER	
REQ_NOT_QUEUED_END	INTEGER	
REQ_STALLS_END	INTEGER	
REQ_DEADLOCKS_END	INTEGER	
PROM_DEADLOCKS_END	INTEGER	
LOCK_RELS_END	INTEGER	
LOCK_STALL_TIME_END	INTEGER	
D_FETCH_RET_END	INTEGER	
D_FETCH_UPD_END	INTEGER	
D_LB_ALLOK_END	INTEGER	
D_LB_GBNEEDLOCK_END	INTEGER	
D_LB_NEEDLOCK_END	INTEGER	
D_LB_OLDVER_END	INTEGER	
D_GB_NEEDLOCK_END	INTEGER	
D_GB_OLDVER_END	INTEGER	
D_NOTFOUND_IO_END	INTEGER	

(continued on next page)

Table 3–7 (Cont.) Columns for Table EPC$1_221_REQUEST_ACTUAL

Column Name	Data Type	Domain
D_NOTFOUND_SYN_END	INTEGER	
S_FETCH_RET_END	INTEGER	
S_FETCH_UPD_END	INTEGER	
S_LB_ALLOK_END	INTEGER	
S_LB_GBNEEDLOCK_END	INTEGER	
S_LB_NEEDLOCK_END	INTEGER	
S_LB_OLDVER_END	INTEGER	
S_GB_NEEDLOCK_END	INTEGER	
S_GB_OLDVER_END	INTEGER	
S_NOTFOUND_IO_END	INTEGER	
S_NOTFOUND_SYN_END	INTEGER	
D_ASYNC_FETCH_END	INTEGER	
S_ASYNC_FETCH_END	INTEGER	
D_ASYNC_READIO_END	INTEGER	
S_ASYNC_READIO_END	INTEGER	
AS_READ_STALL_END	INTEGER	
AS_BATCH_WRITE_END	INTEGER	
AS_WRITE_STALL_END	INTEGER	
BIO_END	INTEGER	
DIO_END	INTEGER	
PAGEFAULTS_END	INTEGER	
PAGEFAULT_IO_END	INTEGER	
CPU_END	INTEGER	
CURRENT_PRIO_END	SMALLINT	
VIRTUAL_SIZE_END	INTEGER	
WS_SIZE_END	INTEGER	
WS_PRIVATE_END	INTEGER	
WS_GLOBAL_END	INTEGER	

Table 3–8 shows the TRANSACTION table.

Table 3–8 Columns for Table EPC$1_221_TRANSACTION

Column Name	Data Type	Domain
COLLECTION_RECORD_ID	SMALLINT	COLLECTION_RECORD_ID_ DOMAIN
IMAGE_RECORD_ID	INTEGER	IMAGE_RECORD_ID_DOMAIN
CONTEXT_NUMBER	INTEGER	CONTEXT_NUMBER_DOMAIN
TIMESTAMP_START	DATE VMS	

(continued on next page)

Table 3–8 (Cont.) Columns for Table EPC$1_221_TRANSACTION

Column Name	Data Type	Domain
TIMESTAMP_END	DATE VMS	
CLIENT_PC_START	INTEGER	
STREAM_ID_START	INTEGER	
LOCK_MODE_START	INTEGER	
TRANS_ID_START	VARCHAR(16)	
TRANS_ID_START_STR_ID	INTEGER	STR_ID_DOMAIN
GLOBAL_TID_START	VARCHAR(16)	
GLOBAL_TID_START_STR_ID	INTEGER	STR_ID_DOMAIN
DBS_READS_START	INTEGER	
DBS_WRITES_START	INTEGER	
RUJ_READS_START	INTEGER	
RUJ_WRITES_START	INTEGER	
AIJ_WRITES_START	INTEGER	
ROOT_READS_START	INTEGER	
ROOT_WRITES_START	INTEGER	
BUFFER_READS_START	INTEGER	
GET_VM_BYTES_START	INTEGER	
FREE_VM_BYTES_START	INTEGER	
LOCK_REQS_START	INTEGER	
REQ_NOT_QUEUED_START	INTEGER	
REQ_STALLS_START	INTEGER	
REQ_DEADLOCKS_START	INTEGER	
PROM_DEADLOCKS_START	INTEGER	
LOCK_RELS_START	INTEGER	
LOCK_STALL_TIME_START	INTEGER	
D_FETCH_RET_START	INTEGER	
D_FETCH_UPD_START	INTEGER	
D_LB_ALLOK_START	INTEGER	
D_LB_GBNEEDLOCK_START	INTEGER	
D_LB_NEEDLOCK_START	INTEGER	
D_LB_OLDVER_START	INTEGER	
D_GB_NEEDLOCK_START	INTEGER	
D_GB_OLDVER_START	INTEGER	
D_NOTFOUND_IO_START	INTEGER	
D_NOTFOUND_SYN_START	INTEGER	
S_FETCH_RET_START	INTEGER	
S_FETCH_UPD_START	INTEGER	
S_LB_ALLOK_START	INTEGER	

(continued on next page)

Table 3–8 (Cont.) Columns for Table EPC$1_221_TRANSACTION

Column Name	Data Type	Domain
S_LB_GBNEEDLOCK_START	INTEGER	
S_LB_NEEDLOCK_START	INTEGER	
S_LB_OLDVER_START	INTEGER	
S_GB_NEEDLOCK_START	INTEGER	
S_GB_OLDVER_START	INTEGER	
S_NOTFOUND_IO_START	INTEGER	
S_NOTFOUND_SYN_START	INTEGER	
D_ASYNC_FETCH_START	INTEGER	
S_ASYNC_FETCH_START	INTEGER	
D_ASYNC_READIO_START	INTEGER	
S_ASYNC_READIO_START	INTEGER	
AS_READ_STALL_START	INTEGER	
AS_BATCH_WRITE_START	INTEGER	
AS_WRITE_STALL_START	INTEGER	
AREA_ITEMS_START	VARCHAR(128)	
AREA_ITEMS_START_STR_ID	INTEGER	STR_ID_DOMAIN
BIO_START	INTEGER	
DIO_START	INTEGER	
PAGEFAULTS_START	INTEGER	
PAGEFAULT_IO_START	INTEGER	
CPU_START	INTEGER	
CURRENT_PRIO_START	SMALLINT	
VIRTUAL_SIZE_START	INTEGER	
WS_SIZE_START	INTEGER	
WS_PRIVATE_START	INTEGER	
WS_GLOBAL_START	INTEGER	
CROSS_FAC_2_START	INTEGER	
CROSS_FAC_3_START	INTEGER	
CROSS_FAC_7_START	INTEGER	
CROSS_FAC_14_START	INTEGER	
DBS_READS_END	INTEGER	
DBS_WRITES_END	INTEGER	
RUJ_READS_END	INTEGER	
RUJ_WRITES_END	INTEGER	
AIJ_WRITES_END	INTEGER	
ROOT_READS_END	INTEGER	
ROOT_WRITES_END	INTEGER	
BUFFER_READS_END	INTEGER	

(continued on next page)

Table 3–8 (Cont.) Columns for Table EPC$1_221_TRANSACTION

Column Name	Data Type	Domain
GET_VM_BYTES_END	INTEGER	
FREE_VM_BYTES_END	INTEGER	
LOCK_REQS_END	INTEGER	
REQ_NOT_QUEUED_END	INTEGER	
REQ_STALLS_END	INTEGER	
REQ_DEADLOCKS_END	INTEGER	
PROM_DEADLOCKS_END	INTEGER	
LOCK_RELS_END	INTEGER	
LOCK_STALL_TIME_END	INTEGER	
D_FETCH_RET_END	INTEGER	
D_FETCH_UPD_END	INTEGER	
D_LB_ALLOK_END	INTEGER	
D_LB_GBNEEDLOCK_END	INTEGER	
D_LB_NEEDLOCK_END	INTEGER	
D_LB_OLDVER_END	INTEGER	
D_GB_NEEDLOCK_END	INTEGER	
D_GB_OLDVER_END	INTEGER	
D_NOTFOUND_IO_END	INTEGER	
D_NOTFOUND_SYN_END	INTEGER	
S_FETCH_RET_END	INTEGER	
S_FETCH_UPD_END	INTEGER	
S_LB_ALLOK_END	INTEGER	
S_LB_GBNEEDLOCK_END	INTEGER	
S_LB_NEEDLOCK_END	INTEGER	
S_LB_OLDVER_END	INTEGER	
S_GB_NEEDLOCK_END	INTEGER	
S_GB_OLDVER_END	INTEGER	
S_NOTFOUND_IO_END	INTEGER	
S_NOTFOUND_SYN_END	INTEGER	
D_ASYNC_FETCH_END	INTEGER	
S_ASYNC_FETCH_END	INTEGER	
D_ASYNC_READIO_END	INTEGER	
S_ASYNC_READIO_END	INTEGER	
AS_READ_STALL_END	INTEGER	
AS_BATCH_WRITE_END	INTEGER	
AS_WRITE_STALL_END	INTEGER	
AREA_ITEMS_END	VARCHAR(128)	
AREA_ITEMS_END_STR_ID	INTEGER	STR_ID_DOMAIN

(continued on next page)

Table 3–8 (Cont.) Columns for Table EPC$1_221_TRANSACTION

Column Name	Data Type	Domain
BIO_END	INTEGER	
DIO_END	INTEGER	
PAGEFAULTS_END	INTEGER	
PAGEFAULT_IO_END	INTEGER	
CPU_END	INTEGER	
CURRENT_PRIO_END	SMALLINT	
VIRTUAL_SIZE_END	INTEGER	
WS_SIZE_END	INTEGER	
WS_PRIVATE_END	INTEGER	
WS_GLOBAL_END	INTEGER	
CROSS_FAC_2_END	INTEGER	
CROSS_FAC_3_END	INTEGER	
CROSS_FAC_7_END	INTEGER	
CROSS_FAC_14_END	INTEGER	

Table 3–9 shows the REQUEST_BLR table.

Table 3–9 Columns for Table EPC$1_221_REQUEST_BLR

Column Name	Data Type	Domain
COLLECTION_RECORD_ID	SMALLINT	COLLECTION_RECORD_ID_ DOMAIN
IMAGE_RECORD_ID	INTEGER	IMAGE_RECORD_ID_DOMAIN
CONTEXT_NUMBER	INTEGER	CONTEXT_NUMBER_DOMAIN
TIMESTAMP_POINT	DATE VMS	
CLIENT_PC	INTEGER	
STREAM_ID	INTEGER	
REQ_ID	INTEGER	
TRANS_ID	VARCHAR(16)	
TRANS_ID_STR_ID	INTEGER	STR_ID_DOMAIN
REQUEST_NAME	VARCHAR(31)	
REQUEST_NAME_STR_ID	INTEGER	STR_ID_DOMAIN
REQUEST_TYPE	INTEGER	
BLR	VARCHAR(127)	
BLR_STR_ID	INTEGER	STR_ID_DOMAIN

3.1.9.1 Examples using Oracle TRACE collected data

The following example shows how the OPTIMIZE AS clause is reflected in the Oracle TRACE database. When a trace collection is started the following SQL commands will record the request names.

```
SQL> attach 'file personnel';
SQL> select last_name, first_name
cont> from employees
cont> optimize as request_one;
   .
   .
   .
SQL> select employee_id
cont> from employees
cont> optimize as request_two;
   .
   .
   .
SQL> select employee_id, city, state
cont> from employees
cont> optimize as request_three;
   .
   .
   .
SQL> select last_name, first_name, employee_id, city, state
cont> from employees
cont> optimize as request_four;
   .
   .
   .
```

Once an Oracle TRACE database has been populated from the collection, a query such as the following can be used to display the request names and types. The type values are described in Table 3–10. The unnamed queries in this example correspond to the queries executed by interactive SQL to validate the names of the tables and columns referenced in the user supplied queries.

```
SQL> select REQUEST_NAME, REQUEST_TYPE, TIMESTAMP_POINT
cont> from EPC$1_221_REQUEST_BLR;
 REQUEST_NAME                    REQUEST_TYPE   TIMESTAMP_POINT
                                            1   15-JAN-1997 13:23:27.18
                                            1   15-JAN-1997 13:23:27.77
 REQUEST_ONE                                1   15-JAN-1997 13:23:28.21
 REQUEST_TWO                                1   15-JAN-1997 13:23:56.55
 REQUEST_THREE                              1   15-JAN-1997 13:24:57.27
 REQUEST_FOUR                               1   15-JAN-1997 13:25:25.44
6 rows selected
```

The next example shows the internal query format (BLR) converted to SQL strings after EPC$EXAMPLES:EPC_BLR_TOSQL_CONVERTER.COM has been run.

```
SQL> SELECT A.REQUEST_NAME, B.SQL_STRING FROM
cont> EPC$1_221_REQUEST_BLR A,
cont> EPC$SQL_QUERIES B
cont> WHERE A.CLIENT_PC = 0 AND A.SQL_ID = B.SQL_ID;
 A.REQUEST_NAME
   B.SQL_STRING
 REQUEST_ONE
       SELECT C1.LAST_NAME, C1.FIRST_NAME.        FROM EMPLOYEES C1
 . . .
 REQUEST_TWO
       SELECT C1.EMPLOYEE_ID.        FROM EMPLOYEES C1
 . . .
 REQUEST_THREE
       SELECT C1.EMPLOYEE_ID, C1.CITY, C1.STATE.        FROM EMPLOYEES C1
    .
    .
    .
4 rows selected
```

Table 3–10 Request Types

Symbolic Name	Value	Comment
RDB_K_REQTYPE_OTHER	0	A query executed internally by Oracle Rdb
RDB_K_REQTYPE_USER_ REQUEST	1	A non-stored SQL statement, which includes compound statements
RDB_K_REQTYPE_PROCEDURE	2	A stored procedure
RDB_K_REQTYPE_FUNCTION	3	A stored function
RDB_K_REQTYPE_TRIGGER	4	A trigger action
RDB_K_REQTYPE_ CONSTRAINT	5	A table or column constraint

3.1.10 Use of SQL_SQLCA Include File Intended for Host Language File

Use of the SQLCA include files such as the SQL_SQLCA.H file for C, are intended for use with the host language files only. That is, only *.C should be including that file. Precompiled files (*.SC files) should use the EXEC SQL INCLUDE SQLCA embedded SQL command in the declaration section of the module. In this way the precompiler can properly define the structure to be used by the related SQL generated code.

Remember that the SQLCA is always scoped at the module level, unlike the SQLCODE or SQLSTATE variables which may be routine specific.

The following example shows this error:

```
#include <stdio.h>
#include <sql_sqlca.h>
struct SQLCA SQLCA;

int main (void)
{
  EXEC SQL EXECUTE IMMEDIATE 'show version';
  printf ("SQLCODE=%d\n", SQLCA.SQLCODE);
}

$ SQLPRE/CC issues the following error against this program:

  %SQL-F-NOSQLCODE, Neither SQLCA, SQLCODE nor SQLSTATE were declared
```

The following example shows correct usage:

```
#include <stdio.h>
#include <sql_sqlca.h>
EXEC SQL INCLUDE SQLCA;

int main (void)
{
  EXEC SQL EXECUTE IMMEDIATE 'show version';
  printf ("SQLCODE=%d\n", SQLCA.SQLCODE);
}
```

3.1.11 Missing Information on Temporary Tables

The following information was inadvertently omitted from the Oracle Rdb V7.0 SQL Reference Manual. (should be in USAGE NOTES for CREATE temp table)

Data for a temporary table is stored in virtual memory, not in a storage area. For journaling purposes, when changes are made to the data in a temporary table such as updates or deletes, recovery space is required to hold before images of deleted and updated rows. This recovery space also requires virtual memory and may result in having to increase Page File Quota and Virtual Page Count on OpenVMS.

A recommended way to reduce memory usage when using temporary tables is to commit transactions which modify temporary table data as soon as possible. Upon commit the additional copies of data are released and available for reuse by Oracle Rdb. This eliminates extra copies of data and therefore reduces virtual memory usage.

See the Oracle Rdb7 Guide to Database Design and Definition for calculating memory usage for temporary tables.

4
Known Problems and Restrictions

This chapter describes problems, restrictions, and workarounds known to exist in Oracle Rdb 7.0.1.

4.1 Known Problems and Restrictions

4.1.1 Record Cache Use Should be Limited to Read-Only Caches

The record cache feature should be used exclusively with read-only caches. Write operations to the cache can put your database into an unrecoverable state since, in some cases, bad data may be written to the AIJ. To avoid this problem, caching of read/write data should not be attempted. There is no such problem with read-only caches since no cached data will be written to the AIJ. A fix for the read/write record cache problem is under way and will be included in an upcoming ECO to 7.0.1.0.

4.1.2 RMU/SHOW STATISTIC /INPUT qualifier causes bugcheck dump

Using the **RMU/SHOW STATISTIC** utility **/INPUT** qualifier causes a bugcheck dump to be generated.

The following example shows

```
$ rmu/show statistic/inp=stats.dat
%SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual
address=000001E0, P=00240D05, PSL=03C00000
%RMU-F-FATALOSI, Fatal error from the Operating System Interface.
%RMU-I-BUGCHKDMP, generating bugcheck dump file USER1:[KLEIN]RMUBUGCHK.DMP;
%RMU-F-FTL_SHOW, Fatal error for SHOW operation at 11-JUL-1997 09:45:39.70
```

There is no work-around to this problem.

This problem will be corrected in the next release of Oracle Rdb.

4.1.3 Using the SYS$LIBRARY:SQL_FUNCTIONS70.SQL Oracle Functions Script

All OpenVMS platforms.

If your environment isn't 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 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 logicals. This will be necessary for programs that use the functions as well.

In a standard environment you'll still need to do the following.

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

In a multi-version environment you'll need to do this.

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

4.1.4 DECC and Use of the /STANDARD Switch

Bug 394451.

All OpenVMS platforms.

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 DECC was put into the product with V6.0 and this note is meant to clarify that support, not to indicate a change. It's possible to use /STANDARD=RELAXED_ANSI89 or /STANDARD=VAXC correctly, but in general it's best to avoid it's use.

The following example shows both the right and wrong way to compile an Rdb SQL program. Assume a symbol SQL$PRE has been defined and DECC 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 besides RELAXED_ANSI89 and VAXC. Those are not supported.

4.1.5 Restriction Added for CREATE STORAGE MAP on Table with Data

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

- The storage map must be a simple map which 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 may not change THRESHOLDS or COMPRESSION for the table, nor can it use the PLACEMENT VIA INDEX clause. It may only contain one area and may not be vertically partitioned. This new map simply describes the mapping as it exists by default for the table.

This version 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
```

4.1.6 OTHERWISE Clause only Valid with STORE USING Clause of a Vertical Record Partition Storage Map.

In Rdb 7.0, SQL did not enforce the correct use of the OTHERWISE clause in a Vertical Record Partition storage map. Specifically, in a CREATE STORAGE MAP statement, if a STORE columns IN clause was specified, SQL allowed the OTHERWISE clause. This is incorrect syntax.

The following example shows incorrect SQL syntax. There is no STORE USING clause; therefore the OTHERWISE clause is invalid.

```
CREATE STORAGE MAP TABLE_1_MAP FOR TABLE_1
STORE columns (PKEY,IKEy) across (area_1,area_2)
STORE columns (SKEY,CKEY)
IN AREA_4 OTHERWISE AREA_14;
```

There is a workaround. The workaround is to use the correct SQL syntax when creating a Vertical Record Partition storage map. The following example shows the correct SQL syntax. The OTHERWISE clause is used with the STORE USING clause.

```
CREATE STORAGE MAP TABLE_1_MAP FOR TABLE_1
STORE columns (PKEY,IKEy) across (area_1,area_2)
STORE columns (SKEY,CKEY) USING (PKEY)
IN AREA_4 WITH LIMIT OF (100) OTHERWISE AREA_14;
```

This problem will be corrected in a future version of Oracle Rdb. SQL will then generate an error message if the syntax is not correct.

4.1.7 DROP TABLE CASCADE will Result in %RDB-E-NO_META_UPDATE

All platforms.

In Oracle Rdb 7.0, SQL passed incorrect MBLR to Oracle Rdb when a DROP TABLE CASCADE statement was issued. This occurred when the following conditions applied:

1. A table was created with an index defined on the table.
2. A storage map was created with a placement via index.
3. The storage map was a Vertical Record Partition Storage Map with 2 or more STORE COLUMNS clauses.

Thus, if such conditions apply, and one issues a DROP TABLE CASCADE, the result will be the error message:

%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. The example also show the resultant error message.

```
create table vrp_table ( id int, id2 int);
commit;
create unique index vrp_idx on vrp_table (id)
store in empids_low;
commit;
create storage map vrp_map
for vrp_table
placement via index vrp_idx
enable compression
store columns (id)
in empids_low
store columns (id2)
in empids_mid;
commit;
```

```
drop table vrp_table cascade;
Index VRP_IDX is also being dropped.
VIA clause on storage map VRP_MAP 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"
```

There is a workaround. The workaround is to first drop the storage map, and then drop the table cascade. The following example shows the workaround. One must DROP the STORAGE MAP first, then one may issue a DROP TABLE CASCADE. The SHOW statement illustrates that the table, index, and storage map were dropped.

```
drop storage map vrp_map;
drop table vrp_table cascade;
Index VRP_IDX is also being dropped.
commit;
show table vrp_table
No tables found
show index vrp_idx
No indexes found
show storage map vrp_map
No Storage Maps Found
```

This problem will be corrected in a future version of Oracle Rdb. SQL will then generate correct MBLR.

4.1.8 Do Not Use RMU/CONVERT/NOCOMMIT with Databases Used by Replication Transfers

Bug 507408.

RMU/CONVERT/NOCOMMIT fails when converting a database that is used as a source for replication transfers used by the Replication Option (Data Distributor):

```
%RMU-I-RMUTXT_000, Executing RMU for Oracle Rdb V7.0-01
Are you satisfied with your backup of DISK1:[USER]SOURCE_DB.RDB;1 and your
backup of any associated .aij files [N]? y
%RMU-I-LOGCONVRT, database root converted to current structure level
%RMU-S-CVTDBSUC, database DISK1:[USER]SOURCE_DB.RDB;1 successfully converted
from version V6.1 to V7.0
%RDB-E-NO_META_UPDATE, metadata update failed
-RDMS-F-NOMETAUPD, metadata updates are prohibited until CONVERT is COMMITTED
```

The error happens as CONVERT tries adding a new index on the system table RDB$TRANSFER_RELATIONS, used by the Replication Option. Because of the failure, the index is not created, but it is critical for the correct operation of the Replication Option. In its absence, the database stops logging changes to the tables subject to replication, and the target database(s) are therefore no longer kept up to date.

In addition, the initial execution of any newly defined replication transfer will fail with bugchecks such as: (on VAX)

```
***** Exception at 0057B70B: RDMS$$DML$READY+000000F4
%SYSTEM-F-ACCVIO, access violation, reason mask=00, virtual
address=00000094, PC=0057B70B, PSL=01400004
```

There is no known workaround once updates have been performed against any of the tables subject to replication. If no updates have been performed yet, then you should:

- Rollback the conversion using RMU/CONVERT/ROLLBACK
- Then, redo the conversion using RMU/CONVERT/COMMIT

If updates have been performed, then you need to drop the transfer and redefine it. The first execution of the replication transfer will then perform a full reinitialization of the target tables.

Note that there is no problem with the extraction transfers.

This problem shall be corrected in a future ECO.

4.1.9 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 World-Wide Support.

4.1.10 Hangs Possible Using Multistatement or Stored Procedures

Long running multistatement or stored procedures can cause other users in the database to hang if they obtain 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. Thus any long running multistatement or stored procedure can cause other processes to hang. 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 hangs permanently.

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.

4.1.11 Use of RDB from Shared Images

All OpenVMS platforms.

Bug 470946.

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

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

1. Do not make RDB calls from the initialization routines of shared images.

2. 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 RDB shared images **last** in the linker options file.

This is not a bug; it is a result of the way OpenVMS image activation works.

4.1.12 DBAPack for use on Windows 3.1 is Deprecated

This is the last release of Oracle Enterprise Manager DBAPack that will be supported for use on Windows 3.1.

4.1.13 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 simple rules shown below.

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.

```
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.

```
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.

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

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

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

```
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 hand side of the assignment in a SET statement or SET clause of an UPDATE statement.

```
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.

```
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.

```
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.

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

SQL only copies values from the client (i.e. 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/ELSE. For example, 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 up to 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.

4.1.14 Interactive SQL Command Line Editor Rejects Eight Bit Characters

Digital UNIX platform.

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.

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

To quote a character using the command line editor, type Ctrl/V before each character to be quoted.

5
Enhancements

This chapter describes the enhancements that are introduced in Oracle Rdb Version 7.0.1.

5.1 Enhancements Provided in Oracle Rdb Version 7.0.1

5.1.1 RMU/SHOW STATISTIC "Stall Class" Informational Screens

Often, when attempting to diagnose performance problems, the existing RMU/SHOW STATISTIC "Stall Messages" and "Active User Stall Messages" screens are extremely useful for identifying the cause of present stalls. However, these screens show only the "moment in time" view of the cause of the stalls. Two new screens have been added to the RMU/SHOW STATISTIC utility; each of these screens displays information about the types or "classes" of stalls that are either occurring or have occurred since statistics collection began.

There are currently 10 classes of stall messages.

- **records**: These are record related stalls, such as waiting for record locks to be granted.
- **pages**: These are page related stalls, such as waiting for storage area I/O to complete or page locks to be granted.
- **tables**: These are table related stalls, such as waiting for logical area locks to be granted.
- **storage areas**: These are storage area related stalls, such as waiting for storage areas to be created, deleted, truncated or opened.
- **database rootfile**: These are database rootfile related stalls, such as waiting for rootfile I/O to complete or “object” locks to be granted.
- **recovery journals**: These are journal related stalls, such as opening, initializing or extending journals, as well as waiting for journal locks to be granted.
- **transactions**: These are transaction related stalls, such as waiting for 2PC transactions to commit, or waiting for checkpoints to complete.
- **hot standby**: These are hot standby related stalls.
- **database**: These are database related stalls, such as waiting for the database freeze to complete.
- **miscellaneous**: These are generic stalls, such as waiting for a bugcheck dump to complete.

The new “Stall Statistics” screen identifies the number of stalls and their corresponding durations for a particular stall class. The screen can be configured (using the "Config" onscreen-menu option) to display either the “aggregate count” information or the “aggregate duration” information. The following example shows the “Stall Statistics” screen displaying aggregate count information.

```
Node: ALPHA3   Oracle Rdb X7.0-00 Performance Monitor 24-OCT-1996 15:55:30
Rate: 1.00 Second Stall Statistics (Aggregate counts) Elapsed: 01:44:29.27
Page: 1 of 1  KODA_TEST:[R_ANDERSON.OE_MASTER]OE_RDB.RDB;1    Mode: Online
------------------------------------------------------------------------------

statistic.......   rate.per.second.............. total....... average......
name............   max..... cur..... avg....... count....... per.trans....

miscellaneous            0        0        0.0            0          0.0
records                 15        0        0.0          160          0.3
pages                 2706        0       48.1       301991        579.6
tables                   0        0        0.0            0          0.0
storage areas           33        0        0.0          133          0.2
database rootfile      125        0        0.3         2162          4.1
recovery journals       66        0        0.5         3405          6.5
transactions            69        0        0.0          294          0.5
hot standby              0        0        0.0            0          0.0
database                 9        0        0.0           19          0.0


------------------------------------------------------------------------------
Config Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset
```

In this example, there were 301,991 total stalls for the "page" class.

The following example shows the “Stall Statistics” screen displaying aggregate duration information.

```
Node: ALPHA3   Oracle Rdb X7.0-00 Performance Monitor 24-OCT-1996 15:55:41
Rate: 1.00 Second Stall Statistics (Aggregate durations) Elapsed: 01:44:40
Page: 1 of 1 KODA_TEST:[R_ANDERSON.OE_MASTER]OE_RDB.RDB;1     Mode: Online
------------------------------------------------------------------------------

statistic.......  rate.per.second.............. total....... average......
name............  max..... cur..... avg....... count....... per.trans....

miscellaneous            0        0        0.0            0          0.0
records               8720        0        4.8        30237         58.0
pages                78453        0       88.3       554787       1064.8
tables                   0        0        0.0            0          0.0
storage areas         3400        0        0.7         4742          9.1
database rootfile     1876        0        1.5         9835         18.8
recovery journals     6652        0        9.9        62716        120.3
transactions         31192        0       10.0        63145        121.1
hot standby              0        0        0.0            0          0.0
database              2052        0        0.1         1226          2.3

------------------------------------------------------------------------------
Config Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset
```

The stall durations are collected as hundredths of seconds. Therefore, the total "page" stall duration of 554787 is really 5547.87 seconds.

Note

Because certain types of stalls can be nested, the total stall durations may be larger than actually occurred.

The new "Active Stall Counts" screen identifies the actual number of processes currently stalled in a particular stall class. Ideally, the number of stalled processes for each class should be "0", which indicates that there are no stalled processes.

The following example shows the "Active Stall Counts" screen:

```
Node: ALPHA3   Oracle Rdb X7.0-00 Performance Monitor 24-OCT-1996 15:41:05
Rate: 0.10 Seconds      Active Stall Counts          Elapsed: 01:30:04.69
Page: 1 of 1  KODA_TEST:[R_ANDERSON.OE_MASTER]OE_RDB.RDB;1    Mode: Online
------------------------------------------------------------------------
Stall Category Stall.Cnt   10   20   30   40   50   60   70   80   90  100
                          +----+----+----+----+----+----+----+----+----+----+
miscellaneous      0      |    |    |    |    |    |    |    |    |    |    |
records            0      |    |    |    |    |    |    |    |    |    |    |
pages              5      +-*  |    |    |    |    |    |    |    |    |    |
tables             0      |    |    |    |    |    |    |    |    |    |    |
storage areas      0      |    |    |    |    |    |    |    |    |    |    |
database rootfile  0      |    |    |    |    |    |    |    |    |    |    |
recovery journals  0      |    |    |    |    |    |    |    |    |    |    |
transactions       1      *    |    |    |    |    |    |    |    |    |    |
hot standby        0      |    |    |    |    |    |    |    |    |    |    |
database           0      |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
                          |    |    |    |    |    |    |    |    |    |    |
----------------------+----+----+----+----+----+----+----+----+----+----+----+
Exit Help Menu Set_rate Write !
```

5.1.2 RMU/SHOW STATISTIC "Stall Messages" Include Transaction State

RMU/SHOW STATISTIC "Stall Messages" screens now display the process current transaction state ("R" for Read-Only or "W" for Read-Write) along with the stall information.

5.1.3 RMU/SHOW STATISTIC "Logical Area" Screens

The RMU/SHOW STATISTIC utility has been enhanced to display "Logical Area" statistics. A "logical area" is a table, b-tree index or hash index. This enhancement essentially provides the means to "drill down" to a specific table's or index's statistic information.

The following screen is an example of the EMPLOYEES table in the MF_PERSONNEL database:

```
Node: ALPHA3   Oracle Rdb X7.0-00 Performance Monitor 15-NOV-1996 07:28:18
Rate: 1.00 Second     Logical Area Statistics         Elapsed: 00:55:54.35
Page: 1 of 1 KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1  Mode: Online
------------------------------------------------------------------------
                     Table EMPLOYEES in EMPIDS_OVER

statistic.......  rate.per.second.............. total....... average......
name............  max..... cur..... avg....... count....... per.trans....

record marked           14        0        0.0           18           0.0

record fetched          14        0        0.7         2645           0.9
    fragmented           0        0        0.0            0           0.0

  record stored          9        0        0.7         2621           0.9
     fragmented          0        0        0.0            0           0.0
  pages checked          9        0        0.7         2621           0.9
       saved IO          2        0        0.0          162           0.0
      discarded          0        0        0.0            0           0.0

record erased            5        0        0.0            6           0.0
   fragmented            0        0        0.0            0           0.0
```

```
------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset Write
```

In the above screen, the statistic information displayed pertains specifically to the EMPLOYEES table within the EMPIDS_OVER storage area.

The following screen is an example of the EMPLOYEES_HASH hash index in the MF_PERSONNEL database:

```
Node: ALPHA3  Oracle Rdb X7.0-00 Performance Monitor 15-NOV-1996 07:47:11
Rate: 1.00 Second     Logical Area Statistics        Elapsed: 01:14:47.77
Page: 1 of 1 KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1 Mode: Online
------------------------------------------------------------------------
                Hash EMPLOYEES_HASH in EMPIDS_OVER

statistic.......  rate.per.second.............. total....... average......
name............  max..... cur..... avg....... count....... per.trans....

hash insertions          0        0        0.0            0           0.0
     duplicates          0        0        0.0            0           0.0

hash deletions           0        0        0.0            0           0.0
    duplicates           0        0        0.0            0           0.0

hash scans               0        0        0.0            0           0.0

hash index fetches       0        0        0.0            0           0.0
  bucket fragments       0        0        0.0            0           0.0
   duplicate nodes       0        0        0.0            0           0.0

------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset Write
```

The following screen is an example of the RDB$NDX_REL_NAME_N btree index in the MF_PERSONNEL database:

```
Node: ALPHA3  Oracle Rdb X7.0-00 Performance Monitor 15-NOV-1996 07:49:21
Rate: 1.00 Second        Logical Area Statistics     Elapsed: 01:16:57.55
Page: 1 of 1 KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1 Mode: Online
------------------------------------------------------------------------
                Btree RDB$NDX_REL_NAME_N in RDB$SYSTEM

statistic.......  rate.per.second.............. total....... average......
name............  max..... cur..... avg....... count....... per.trans....

node fetches             1        0        1.5         7135           2.4
 leaf fetches            0        0        0.6         3039           1.0
 dup. fetches            0        0        0.2         1189           0.4
index lookups            0        0        0.1          792           0.2
index scans              0        0        0.4         2115           0.7
 primary entries         0        0        0.0            0           0.0
 dup. entries            0        0        0.0            0           0.0
node insertions          0        0        0.0            0           0.0
 root insertions         0        0        0.0            0           0.0
 leaf insertions         0        0        0.0            0           0.0
 dup. insertions         0        0        0.0            0           0.0
node creations           0        0        0.0            0           0.0
 root splits             0        0        0.0            0           0.0
 leaf creations          0        0        0.0            0           0.0
 dup. creations          0        0        0.0            0           0.0
index creations          0        0        0.0            0           0.0
node removals            0        0        0.0            0           0.0
 root removals           0        0        0.0            0           0.0
 leaf removals           0        0        0.0            0           0.0
 dup. removals           0        0        0.0            0           0.0
node deletions           0        0        0.0            0           0.0
 leaf deletions          0        0        0.0            0           0.0
 dup. deletions          0        0        0.0            0           0.0
index destructions       0        0        0.0            0           0.0

------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset Write
```

Note that the type of information displayed depends on the type of the logical area. Also, the "graph", "time plot" and "scatter plot" options are available for all fields in the display.

The "Logical Area" screens are selected using the "Logical Area Information" option of the main menu. Currently, there is only 1 logical area screen, the "Logical Area Statistics" screen.

Once the logical area screen has been selected, a menu of available logical areas will be displayed. The menu is sorted in alphabetical order. If the RMU/SHOW STATISTIC utility knows the type of a particular logical area, then the type will be displayed in the menu option, but does not affect the sort order. Note that all logical areas, including the system relations and indexes, are displayed in the menu.

When a logical area has been selected, the RMU/SHOW STATISTIC utility will identify the "type" of the selected logical area. If the type cannot be determined, you will be prompted to enter the logical area type.

The following restrictions and caveats apply:

- The "Logical Area" statistics are *not* written to the binary output file. Conversely, the "Logical Area" statistics screens are not available during binary input file replay.
- There currently is no mechanism to display aggregate information for a partitioned logical area. All statistics information displayed is on a per-storage-area basis.
- A "summary information" screen of all logical areas is not currently available.
- The number of lines of statistics information displayed depends on the size of the terminal or window. In particular, the b-tree index displays up to 24 lines of statistics information; on a standard 24x80 terminal, this means that only the first 15 lines of statistics would be displayed.
- Any table or index created prior to Rdb7 ECO 1 will require that the user enter the appropriate logical area type. WARNING: If you specify an incorrect logical area type (i.e. specify a table as being a hash index), inappropriate statistics fields will be displayed.

5.1.4 RMU/REPAIR /INITIALIZE=LAREA_PARAMETERS /TYPE Qualifier

Starting with Oracle Rdb7 ECO 1, the RMU/SHOW STATISTIC utility can display information on a "per-logical area". A "logical area" is a table, b-tree index, hash index, or any partition of one of those.

The RMU/SHOW STATISTIC utility use the on disk AIP pages to determine the appropriate "type" of each logical area, so that the appropriate statistics information can be displayed. The logical area "type" information in the AIP is "unknown" for logical areas created prior to Oracle Rdb7 ECO 1. If the RMU/SHOW STATISTIC utility cannot determine the logical area type from the AIP, it will prompt the user to manually enter the logical area type; however, this information is *not* updated in the database AIP pages.

Therefore, in order to update the on disk logical area "type" in the AIP, the RMU/REPAIR utility has been enhanced. The /INITIALIZE=LAREA_PARAMETERS qualifier option file support has been enhanced. You can now specify a `/TYPE=[TABLE|BTREE|HASH|SYSTEM|BLOB]` in the options file. For example, to repair the EMPLOYEES table of the MF_PERSONNEL database, you would create an options file that contains the following line:

```
EMPLOYEES /TYPE=TABLE
```

For partitioned logical areas, the **/AREA=name** qualifier can be used to identify the specific storage areas that are to be updated. For example, to repair the EMPLOYEES table of the MF_PERSONNEL database for the EMPID_OVER storage area only, you would create an options file that contains the following line:

```
EMPLOYEES /AREA=EMPID_OVER /TYPE=TABLE
```

The **/TYPE** qualifier specifies the type of a logical area. The following keywords are allowed:

- **TABLE**. Specifies that the logical area is a data table. This would be a table created using the SQL "CREATE TABLE" syntax.
- **BTREE**. Specifies that the logical area is a b-tree index. This would be an index created using the SQL "CREATE INDEX TYPE IS SORTED" syntax.
- **HASH**. Specifies that the logical area is a hash index. This would be an index created using the SQL "CREATE INDEX TYPE IS HASHED" syntax.
- **SYSTEM**. Specifies that the logical area is a system record which is used to identify hash buckets. Users cannot explicitly create these types of logical areas.

  Note

  This type should NOT be used for the RDB$SYSTEM logical areas. This type does NOT identify system relations.

- **BLOB**. Specifies that the logical area is a blob repository.

There is no error checking of the "type" specified for a logical area. The specified type does not affect the collection of statistics, nor does it affect the readying of the affected logical areas. However, an incorrect type *will* cause incorrect statistics to be reported by the RMU/SHOW STATISTIC utility.

5.1.5 Replication Governor Status Configurable

The status of the Hot Standby Replication Governor is now a database attribute. The REPLICATE AFTER_JOURNAL CONFIGURE utility supports the enabling and disabling of the Replication Governor. The DUMP/HEADER command can be used to view the status of the Replication Governor for a given replication standby database.

The following example shows the RMU CONFIGURE command.

```
$ RMU/REPLICATE AFTER_JOURNAL CONFIGURE -
     USER_DISK1:[ORACLE_RDB.STANDBY]MF_PERSONNEL -
    /MASTER_ROOT=REMOTE1::USER_DISK2:[ORACLE_RDB.MASTER]MF_PERSONNEL.RDB -
    /GOVERNOR=ENABLED
```

The following example shows the output of an RMU/DUMP/HEADER=HOT_ STANDBY command.

```
*------------------------------------------------------------------------
* Oracle Rdb V7.0-00                                3-DEC-1996 10:48:02.68
*
* Dump of Database header
*   Database: USER_DISK1:[ORACLE_RDB.STANDBY]MF_PERSONNEL.RDB;1
*
*------------------------------------------------------------------------

Database Parameters:
    Root filename is "USER_DISK1:[ORACLE_RDB.STANDBY]MF_PERSONNEL.RDB;1"
    Hot Standby...
     - Database has been configured as "Replication Standby"
         Master database is "USER_DISK2:[ORACLE_RDB.MASTER]MF_PERSONNEL.RDB;1"
         Remote node name is "REMOTE1"
         Replication last commenced on 21-NOV-1996 09:03:36.92
         Database replication is "online"
         Replication Governor is "enabled"
         Server checkpoint interval is 100 messages
         Server gap-timeout interval is 5 minutes
         Server buffer count is 256
         Server 2PC transaction resolution is "commit"
```

5.1.6 Stall Messages and RMU/SHOW STATISTICS

In previous releases of Oracle Rdb, stall messages (the messages that show on the "Stall Messages" and "Active User Stall Messages" screens) were written and formatted by each process attached to the database. The RMU/SHOW STATISTICS process simply displayed the messages on the screen.

This behavior has been changed. Starting with Oracle Rdb Version 7.0.1 ECO 1 RMU/SHOW STATISTICS process creates and formats the stall message strings. Individual processes attached to the database quickly record information about the stall but perform no message formatting.

Moving the work of formatting the stall messages results in a reduction of CPU usage for all processes attached to the database. Less CPU resources are consumed because stall message information is now never formatted by the processes attached to the database (whether RMU/SHOW STATISTICS is being run or not).

However, a process running RMU/SHOW STATISTICS may use additional CPU resources than in previous versions when formatting and displaying the "Stall Messages" and "Active User Stall Messages" screens. But, moving the formatting work to the process running RMU/SHOW STATISTICS from the processes attached to the database should result in an overall reduction in the CPU resources used.

5.1.7 RMU/SHOW USERS Now Identifies Nodes Where a Database is Open

The **RMU/SHOW USERS** utility has been enhanced to identify the various nodes in the VMScluster where the database is currently "open" and available for use. Note that “open” includes those nodes where the database is open for utility access only.

The following example shows the **RMU/SHOW USERS** utility output for a database open on three nodes.

```
ALL> rmu/show users
Oracle Rdb X7.0-00 on node ALPHA3  3-JAN-1997 13:30:17.54
    - monitor started  3-JAN-1997 09:38:25.73
    - monitor log filename is
 "$111$DUA366:[RDMMON_LOGS]RDMMON701_ALPHA3.LOG;634"
    - 255 monitor buffers available (256 maximum)
```

```
database _$111$DUA347:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1
    - First opened  3-JAN-1997 09:39:17.94
    * database is opened by an operator
    - current after-image journal file is
 _$111$DUA347:[R_ANDERSON.WORK.STATS]RICK1.AIJ;1
    - Database also open on these nodes:
      ALPHA5
      QUICKR
ALL>
```

Note that the "current" node (ALPHA3) is not displayed in the list output.

5.1.8 Disk Erase Control Logical Name RDM$BIND_FORCED_DISK_ERASE

In certain cases, problems with the operating system file erase function have caused problems with Oracle Rdb's ability to correctly initialize file contents with specific patterns.

To work around these problems, a new logical name, RDM$BIND_FORCED_ DISK_ERASE can be used to control how Oracle Rdb attempts to initialize files with specific patterns. If RDM$BIND_FORCED_DISK_ERASE is undefined or defined as zero, Oracle Rdb will use the operating system's erase functionality to initialize file contents. If RDM$BIND_FORCED_DISK_ERASE is defined as "1", then Oracle Rdb will initialize the file contents by writing a pattern into the file directly. Because writing to the file directly is slower than using the operating system's erase features, it is not the default.

To be effective, the logical name RDM$BIND_FORCED_DISK_ERASE must be defined /SYSTEM and must be defined before a database is opened or created.

5.1.9 RMU/SHOW STATISTIC Lock Timeout and Lock Deadlock Logging

Tools like the **RMU/SHOW STATISTIC** utility can help solve a current problem within the database. But most problems are solved within a limited timeframe. Because most customers are running Rdb in a 7x24 hour environment it is not always possible to have an expert readily available. This means most customers cannot trace a problem until after it has occurred and been solved.

This is especially the case when analyzing lock timeouts and deadlocks, because once the event has occurred all information regarding the lock event is gone. Frequently, a lock deadlock of "interest" is superseded by another lock deadlock.

Currently, the **RMU/SHOW STATISTIC** utility provides the "Lock Timeout History" and "Lock Deadlock History" screens. However, these screens only record the *last* timeout or deadlock for the processes. There is no method to record each lock timeout or deadlock as it occurs.

The **RMU/SHOW STATISTIC** utility has been enhanced to provide two new command qualifiers: **/LOCK_TIMEOUT_LOG=logfile_spec** and **/DEADLOCK_ LOG=logfile_spec**.

The **logfile_spec** is the name of the file to which all lock timeout and lock deadlock messages will be logged. The lock timeout and lock deadlock messages are written in human-readable format similar to the "Lock Timeout History" and "Lock Deadlock History" screens.

The header region of the lock timeout and lock deadlock log contains three lines. The first line indicates that the **RMU/SHOW STATISTIC** utility created the log file. The second line identifies the database. The third line identifies the date and time the log was created.

The main body of the stall log contains three columns. The first columns contains the process ID and stream ID which experienced the lock timeout or deadlock. The second column contains the time the timeout or deadlock occurred; the date is NOT displayed. The third column contains the timeout or deadlock message describing the affected resource; this message is similar to the originating stall message. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR)
```

Also, if any lock timeouts or lock deadlocks are missed for a particular process, usually because the recording interval is too large, the number of missed lock timeouts or deadlocks is displayed after the message, in brackets. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR) [1 missed]
```

Only one message per occurring lock timeout or deadlock is logged.

The lock timeout or deadlock messages are written at the specified screen refresh rate, determined using the /TIME command qualifier or online using the "Set_ rate" onscreen menu option. Obviously, using a larger refresh rate will minimize the size of the file but result in a large number of missed lock timeout or deadlock messages. Using a smaller refresh rate will produce a large log file, but will contain a much finer granularity of lock timeout or deadlock messages.

The affected **LockID** is not displayed, since this is meaningless information after the lock deadlock has completed.

Note that you do not need to be displaying the "Lock Timeout History" or "Lock Deadlock History" screens in order to record the stall messages to the stall log. The logs are maintained regardless of which screen, if any, is displayed.

Note that the **/LOCK_TIMEOUT_LOG** and **/DEADLOCK_LOG** qualifiers are separate and distinct from each other. They can be used together or separately, as necessary.

Using the **/LOCK_TIMEOUT_LOG** or **/DEADLOCK_LOG** command qualifiers, a "Lock Event Logging" server can be easily constructed. The following **OpenVMS DCL** script how to create a server that logs both lock timeout and lock deadlock events on the MF_PERSONNEL database for the next 15 minutes:

```
$ RMU/SHOW STATISTIC /NOHISTORY /TIME=1 /NOINTERACTIVE
    /LOCK_TIMEOUT_LOG=TIMEOUT.LOG /DEADLOCK_LOG=DEADLOCK.LOG -
    /NOBROADCAST /UNTIL="+15:00" -
    MF_PERSONNEL
```

Note

Using the **/TIME=1** or **/TIME=-50** qualifier appears to produce a reasonable log while minimizing the impact on the system.

The lock timeout and lock deadlock logging facility can also be enabled or disabled while the **RMU/SHOW STATISTIC** utility is running. Using the "Tools" menu (obtained by typing the "!" key at any screen).

The lock timeout log and lock deadlock log are not available during binary file replay.

The following sample "Lock Deadlock" log was produced using a refresh rate of "1.00" seconds:

```
            Oracle Rdb X7.0-00 Performance Monitor Lock Deadlock Log
             Database KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1
                Lock Deadlock Log created  7-FEB-1997 14:20:40.57
2EA00B52:33 14:20:46.22 - waiting for page 5:924 (PW)
2EA04F3A:34 14:20:45.09 - waiting for page 5:792 (PR)
2EA00B52:33 14:20:46.98 - waiting for page 5:1228 (PW)
2EA04F3A:34 14:20:48.19 - waiting for page 5:1086 (PW)
2EA00B52:33 14:20:48.20 - waiting for page 5:1244 (PR)
2EA00B52:33 14:20:58.34 - waiting for page 5:638 (PR)
2EA04F3A:34 14:20:59.17 - waiting for page 5:638 (PW)
2EA04F3A:34 14:21:00.47 - waiting for page 5:482 (PR)
2EA00B52:33 14:21:02.23 - waiting for page 5:661 (PR)
2EA00B52:33 14:21:08.56 - waiting for page 5:523 (PW)
2EA00B52:33 14:21:09.33 - waiting for page 5:918 (PW)
2EA00B52:33 14:21:10.05 - waiting for page 1:976 (PW)
2EA04F3A:34 14:21:15.91 - waiting for page 5:623 (PW)
2EA04F3A:34 14:21:18.63 - waiting for page 5:794 (PW)
2EA00B52:34 14:25:50.45 - waiting for page 5:1086 (PW)
   .
   .
   .
```

This enhancement is available in Oracle Rdb Version 7.0.1.

5.1.10 ALTER INDEX Now Performs Less Area Scans

When ALTER INDEX is used to change the partitioning of an index it scans each referenced storage area to initialize and create the index partitions.

With this version of Rdb, the amount of I/O has been reduced (especially for MIXED format areas) by avoiding one of the area scans. Prior to this change Rdb would scan each of the areas twice to remove the old index. With this version of Rdb7 only a single scan of each area is required to remove the old hash index.

This change makes use of the same optimizations introduced in Rdb7 for commands such as TRUNCATE TABLE, DROP TABLE, DROP INDEX and ALTER DATABASE ... DROP STORAGE AREA CASCADE.

5.1.11 RMU/DUMP/AFTER /START & /END Qualifiers are Difficult to Use

The **/START** and **/END** qualifiers for the **RMU/DUMP/AFTER_JOURNAL** utility are extremely difficult to use as users seldom know, nor can they determine, the AIJ record number in advance of using the utility.

There is no workaround to this problem.

This problem has been corrected in Oracle Rdb Version 7.0.1. The **RMU/DUMP /AFTER_JOURNAL** utility has been enhanced to provide more advanced selection criteria. Two new optional qualifiers, **/FIRST=select_list** and **/LAST=select_list** have been added.

The **select_list** of these qualifiers consists of a list of one or more of the following keywords:

- **RECORD=record#**: Specifies the first or last record in the AIJ journal. This is the same as the existing **/START** and **/END** qualifiers, which are still supported, but obsolete.
- **BLOCK=block#**: Specifies the first or last block in the AIJ journal.

- **TSN=tsn**: Specifies the first or last TSN in the AIJ journal, using the standard "[n:]m" TSN format.
- **TID=tid**: Specifies the first or last TID in the AIJ journal.
- **TIME=date_time**: Specifies the first or last date/time in the AIJ journal, using the standard date/time format.

Both of the **/FIRST** and **/LAST** qualifiers are optional. You may specify both, either or them, or neither of them.

The keywords specified for the **/FIRST** qualifier can differ from the keywords specified for the **/LAST** qualifier.

For example, to start the dump from the fifth block of the AIJ journal, you would use the following command:

```
RMU/DUMP/AFTER_JOURNAL /FIRST=(BLOCK=5) MF_PERSONNEL.AIJ
```

To start the dump from block 100 or TSN 52, whichever occurs first, you would use the following command:

```
RMU/DUMP/AFTER_JOURNAL /FIRST=(BLOCK=100,TSN=0:52) MF_PERSONNEL.AIJ
```

When multiple keywords are specified for a qualifier, the first condition being encountered activates the qualifier. In the above example, the dump will start when *either* block 100 or TSN 52 is encountered.

5.1.12 RMU Extract Can Now Order Storage Area, Cache and Journal Names

This release of Rdb adds the following new option to RMU Extract.

/OPTION=ORDER_BY_NAME

This option will cause the storage area, cache and journal names for the items DATABASE, ALTER_DATABASE (also known as CHANGE_DATABASE) and IMPORT to be ordered alphabetically by the ASCII collating sequence (these names do not have an associated character set).

The default, as in prior releases, is /OPTION=NOORDER_BY_NAME which displays the names in approximate definition order. The default ordering is approximate because a DROP STORAGE AREA, DROP CACHE or DROP JOURNAL will free a slot which may later be reused, this changing the order.

The logical name RDMS$BIND_SORT_WORKFILES will be used to allocate work files if needed.

Note

If the IDENTIFIER CHARACTER SET for the database is other than MCS or ASCII then this option will be ignored. This is because the characters from other character sets do not sort appropriately under the ASCII collating sequence.

5.1.13 SHOW STATISTIC Utility User-Defined Events

Through the configuration file, users are able to specify special “events” and specify what action is to be performed when an event occurs. An event is signalled when a statistic’s value exceeds a user-defined set of thresholds.

The “EVENT” variable’s value is a free-format description of the user-specified event. The format is defined as follows:

```
EVENT="operation statistic_name event_name [ attribute_list ]"
```

5.1.13.1 Operation

The “operation” identifies what action is to be performed. The “operation” keywords are the following:

- **ENABLE** - enables a new event or changes an existing event definition.
- **DISABLE** - disables an existing event, typically used when importing a new configuration file.

5.1.13.2 Statistic Name

The “statistic_name” identifies the particular valid statistic field for which the event is to be enabled or disabled. Note that some statistic names are only valid when certain database attributes are enabled, such as global buffers or record caching. The names of a particular statistic field can be found on the screen of interest. When statistic names contain multiple words, such as “process attaches”, it is required that the statistic name be either single- or double-quoted; failure to quote the statistic name may result in a parsing error.

5.1.13.3 Event Name

The “event_name” identifies the particular event threshold for the specified statistic to be enabled or disabled. Up to eight different thresholds can be specified for a particular statistic field. The “event_name” keywords are the following:

- **MAX_CUR_TOTAL** - the maximum "total" value yet collected.
- **MIN_CUR_RATE** - the lowest rate currently being sustained.
- **MAX_CUR_RATE** - the highest rate currently being sustained.
- **MAX_RATE** - the maximum "current" rate yet collected.
- **MIN_AVG_RATE** - the lowest average rate.
- **MAX_AVG_RATE** - the highest average rate.
- **MIN_PER_TX** - the lowest per-transaction rate.
- **MAX_PER_TX** - the highest per-transaction rate.

5.1.13.4 Attribute List

The optional “attribute_list” provides information about *enabled* events; these attributes are ignored when disabling an event. The “attribute_list” keywords are the following:

- **INITIAL value** - defines the initial value of the event threshold; the default value is “0” for **MAX_XXX** events and “very big number” for **MIN_XXX** events. The default value guarantees that at least one event will be signalled, thereby initializing the new “current” threshold value.

- **EVERY value** - defines the value by which the initial threshold will be incremented or decremented when an event is signalled. If this value is "0", the default value, then the event will be signalled just once.
- **LIMIT value** - defines the maximum number of times the event may be signaled. If the value is "0", the default value, events may be signalled indefinitely, *if* the **EVERY** keyword is specified with a non-zero value.
- **NOTIFY oper_class_list** - defines the *quoted* comma-separated list of operators to be notified for all events defined on the specified statistic.
- **INVOKE program** - defines the *quoted* user-supplied program to be invoked for all events defined on the specified statistic.

5.1.13.5 Event Semantics

In order for an event to be active, you must specify either one or both of the "NOTIFY" or "INVOKE" keywords. When using the "INVOKE" keyword, the program must be a symbol pointing to the DCL script or image to be invoked. Also note that the "INVOKE" and "NOTIFY" definitions apply to *all* events defined for the specified statistic field. Therefore, specifying either or both of these attributes for multiple event descriptions for the same statistics field means that only the last definition will be used.

Note

The **INVOKE** program and **NOTIFY** operator classes apply to *ALL* events defined for the statistic field. Therefore, these keywords only need to be defined *once* per statistic field, no matter how many events are defined. Specifying multiple programs or operator classes causes only the last specified attribute to be used.

Once an event has been signalled, it will only be re-signalled if the "EVERY" clause was specified with a non-zero value. The current threshold value, originally initialized to the "INITIAL" value, will be incremented for **MAX_XXX** thresholds and decremented for **MIN_XXX** thresholds. The **MIN_XXX** thresholds disable themselves once the "INITIAL" value reaches "0", while the **MAX_XXX** thresholds never disable themselves.

Once an event has been disabled, it can only be re-enabled by importing a new configuration file or manually using the "Statistics Event Information" screen "Re-enable all disabled events" configuration sub-menu option.

The user-defined events are analyzed at the specified screen refresh rate. Multiple events defined for the same statistic field may cause the specified program to be invoked multiple times, once for each affected event.

If a program is specified to be invoked when an event occurs, the program will be invoked with seven parameters. Some of the parameters contain multiple words; be sure to quote them if the parameters are passed to other utilities. The parameters are the following:

1. This parameter is the date and time the event occurred.
2. This parameter is the statistic field name.
3. This parameter is the event name.
4. This parameter is the current event value.

5. This parameter is the word “above” or “below”.

6. This parameter is the current event threshold.

7. This parameter is the event occurrence count.

5.1.13.6 Simple Event Configuration File Example

The following is an example of a possible configuration file containing four user-defined event entries:

```
# This file was generated by the SHOW STATISTIC utility.
DATABASE = "KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1"
# Generated on  8-APR-1997 15:08:33.72
EVENT_DESCRIPTION="ENABLE transactions MAX_RATE INITIAL 5 EVERY 2 INVOKE 'LOG_EVENT'";
EVENT_DESCRIPTION="DISABLE transactions MAX_CUR_RATE";
EVENT_DESCRIPTION="ENABLE transactions MIN_CUR_RATE INITIAL 5 EVERY 2 LIMIT 5";
EVENT_DESCRIPTION="ENABLE 'process failures' MAX_CUR_TOTAL INITIAL 1 EVERY 1 NOTIFY OPER12";
```

In the above example, an event is enabled for the “transactions” statistic, specifying a maximum rate threshold starting at “5” and incrementing by “2” whenever the event is signalled. When the event is signalled, the user-supplied program “LOG_EVENT” will be invoked.

In the above example, the program “LOG_EVENT” identifies a VMS symbol which defines the program to be executed. The symbol is defined on VMS as follows:

```
$ LOG_EVENT :== @SYS$SYSTEM:LOG_EVENT.COM
```

If the program fails for any reason, the event will be automatically disabled. The following is a simple example of the “LOG_EVENT.COM” DCL script:

```
$ set noon
$ open/append/share=read event_log sys$disk:[]event.log
$ write event_log "''p1' ''p2' ''p3' ''p4' ''p5' ''p6' (count is ''p7')"
$ close event_log
$ exit
```

In this example, the DCL script appends the event message to the disk file “event.log”.

Note that, in the above event description example, the second event to enable the MIN_CUR_RATE threshold for the “transactions” statistic specified neither an “INVOKE” nor “NOTIFY” attribute. Because either of these attributes was already specified for the preceeding “transactions” statistic event definition, you do not need to specify it again.

When the event is signalled, the designated program or operator class(es) are invoked with a description of the event. For example:

```
22-MAY-1997 13:41:35.17 transactions MIN_PER_TX 1.0 below 4294967295.0 (count is 1)
22-MAY-1997 13:42:06.73 transactions MAX_RATE 14.0 above 1.0 (count is 1)
22-MAY-1997 13:42:07.66 transactions MAX_CUR_TOTAL 17.0 above 2.0 (count is 1)
22-MAY-1997 13:42:08.61 transactions MAX_CUR_RATE 14.0 above 2.0 (count is 1)
22-MAY-1997 13:42:10.58 transactions MAX_RATE 15.0 above 14.0 (count is 2)
22-MAY-1997 13:42:11.45 transactions MAX_CUR_TOTAL 77.0 above 17.0 (count is 2)
22-MAY-1997 13:42:12.39 transactions MAX_CUR_RATE 15.0 above 14.0 (count is 2)
22-MAY-1997 13:42:14.34 transactions MAX_CUR_TOTAL 137.0 above 77.0 (count is 3)
22-MAY-1997 13:42:16.25 transactions MAX_CUR_TOTAL 169.0 above 137.0 (count is 4)
22-MAY-1997 13:42:18.12 transactions MAX_RATE 17.0 above 16.0 (count is 3)
22-MAY-1997 13:42:18.98 transactions MAX_CUR_TOTAL 201.0 above 169.0 (count is 5)
22-MAY-1997 13:42:19.88 transactions MAX_CUR_RATE 17.0 above 16.0 (count is 3)
22-MAY-1997 13:42:21.78 transactions MAX_CUR_TOTAL 260.0 above 201.0 (count is 6)
22-MAY-1997 13:42:23.65 transactions MAX_CUR_TOTAL 294.0 above 260.0 (count is 7)
22-MAY-1997 13:42:25.58 transactions MAX_CUR_TOTAL 326.0 above 294.0 (count is 8)
22-MAY-1997 13:42:27.43 transactions MAX_CUR_TOTAL 356.0 above 326.0 (count is 9)
22-MAY-1997 13:42:28.43 transactions MIN_AVG_RATE 0.1 below 4294967295.0 (count is 1)
22-MAY-1997 13:42:30.52 transactions MAX_CUR_TOTAL 407.0 above 356.0 (count is 10)
22-MAY-1997 13:42:32.36 transactions MAX_CUR_TOTAL 441.0 above 407.0 (count is 11)
22-MAY-1997 13:42:34.29 transactions MAX_CUR_TOTAL 474.0 above 441.0 (count is 12)
22-MAY-1997 13:42:36.23 transactions MAX_CUR_TOTAL 507.0 above 474.0 (count is 13)
22-MAY-1997 13:42:38.12 transactions MAX_CUR_TOTAL 542.0 above 507.0 (count is 14)
22-MAY-1997 13:42:40.13 transactions MAX_CUR_TOTAL 574.0 above 542.0 (count is 15)
```

The current runtime state of the user-defined events can be examined using the new "Statistics Event Information" screen, located in the "Database Parameters" sub-menu. Note that you do *not* have to be in the screen to signal events. Consider the following "Statistics Event Information" screen example:

```
Node: ALPHA3 (1/1/2)    Oracle Rdb X7.0-00 Perf. Monitor 22-MAY-1997 13:54:44.37
Rate: 1.00 Second         Statistics Event Information     Elapsed: 01:10:46.17
Page: 1 of 8    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
--------------------------------------------------------------------------------
Statistic..... Event..... State...     Threshold Every       Current  Cnt
transactions       MAX_RATE      enabled           1.0     2           1.0    0
transactions       MAX_CUR_TOTAL enabled        8194.0     2        8194.0    1
transactions       MIN_CUR_RATE  disabled          1.0     2           0.0    1
transactions       MAX_CUR_RATE  enabled           2.0     2           0.0    0
transactions       MIN_AVG_RATE  disabled          1.9     0           1.9    1
transactions       MAX_AVG_RATE  enabled           3.0     2           1.9    0
transactions       MIN_PER_TX    disabled          1.0     2           1.0    1
transactions       MAX_PER_TX    enabled           4.0     2           1.0    0

--------------------------------------------------------------------------------
Config Exit Full Help Menu >next_page <prev_page Options Set_rate Write !
```

Note that the screen contains the "Full" onscreen-menu option. Using this option displays two lines of information per event, including the **INVOKE** program filespec, if any.

5.1.13.7 Step-by-Step Event Example

Suppose you want to be sent email whenever an application process on the database terminates prematurely terminates. Using events, this is easy to accomplish. The following steps describe how this can be achieved:

1. Identify the operation.

   We are going to **ENABLE** a new event.

2. Identify the statistic name to which the event will be assigned.

   We will use the "process failures" statistic from the "Recovery Statistics" screen (located in the "AIJ Information" sub-menu). This statistic is available even if you are not using after-image journaling.

3. Identify the event name to use.

We will use the **MAX_CUR_TOTAL** event name, since this represents the current number of processes that have failed.

4. Identify the event attributes to use.

   This is probably the hardest part of defining an event. Since we want to be alerted to *any* process failure, then we have to set the **INITIAL** attribute to "0". Since we want to be alerted to *every* process failure, then we have to set the **EVERY** attribute to "1" and the **LIMIT** attribute to "0".

5. Define how we will be alerted about the event.

   Since we want to be sent mail, we will need to use the **INVOKE** keyword. Invoking a program on OpenVMS requires that we define a "DCL symbol" to identify the actual DCL script; for example:

   ```
   $ DBR_LOGGER :== @SYS$DISK:[]DBR_LOGGER.COM
   ```

6. Write the program to be invoked.

   Since we want to be sent mail with a nice description of what event actually occurred, we will use the following DCL script "DBR_LOGGER.COM" to do this:

   ```
   $ set noon
   $ create /nolog sys$scratch:dbr_logger.tmp
   EOD
   $ open /write dbr_logger sys$scratch:dbr_logger.tmp
   $ write dbr_logger "''p1' ''p2' ''p3' ''p4' ''p5' ''p6' count is ''p7'"
   $ close dbr_logger
   $ mail sys$scratch:dbr_logger.tmp r_anderson /subject="DBR notification"
   $ delete /nolog sys$scratch:dbr_logger.tmp;*
   $ exit
   ```

7. Combine all of this this information into the configuration file entry.

   ```
   EVENT_DESCRIPTION="ENABLE 'process failures' MAX_CUR_TOTAL INITIAL 0 EVERY 1 INVOKE DBR_LOGGER"
   ```

8. Invoke the SHOW STATISTIC utility using the configuration file.

   ```
   $ RMU/SHOW STATISTIC /CONFIG=CONFIG.CFG MF_PERSONNEL
   ```

5.1.14 Operator Notification of Storage Area Extension

All OpenVMS platforms.

The system operator classes CLUSTER and CENTRAL may be optionally notified for storage area extend events.

By default, notification of area extends is disabled. To enable this notification, define the logical name RDM$BIND_NOTIFY_STORAGE_AREA_EXTEND to 1. This will result in storage area extend events being notified. There is, however, no way to alter the operator classes that are notified.

The logical name RDM$BIND_NOTIFY_STORAGE_AREA_EXTEND should be defined system-wide so that any process extending an area will cause the notification to occur.

Following is an example operator message:

```
%%%%%%%%%%  OPCOM  16-JAN-1997 05:44:11.71  %%%%%%%%%%
Message from user SYSTEM on SMILES
Oracle Rdb Database DUA0:[DB]DB.RDB;1 Event Notification
Storage area DUA0:[DB]DB1.RDA;1 extending by 4321 disk blocks
```

5.1.15 Unique Identification in Bugcheck Dump Files

Oracle Rdb now includes a unique identifier in bugcheck dump files. This unique identifier is intended to allow bugcheck dumps to be more easily identified and possibly catalogued by users and by Oracle. This identifier is a 32 byte character string and is generated using the system boot time, the time of the bugcheck dump and the process identification of the process writing the dump.

Following is an example of the unique identifier in a bugcheck dump file:

```
=================================================================
          Bugcheck dump FSA0:[DB]RDSBUGCHK.DMP;
=================================================================

This file was generated by Oracle Rdb V7.0-1 upon detection of a
fatal, unexpected, error.  Please return this file, the query or
program that produced the bugcheck, the database, monitor log,
and any other pertinent information to your Oracle support
representative for assistance.

This is a AlphaServer 2100 4/275 running VMS V7.1
Current time is 16-MAY-1997 11:08:14.25
Bugcheck Dump Identification is: "PS6J3WHQF4GBAA2UD32L8BPVA23PBAAA"
```

5.1.16 Operator Notification of Bugchecks

All OpenVMS platforms.

When an Oracle Rdb process writes a bugcheck dump file the system operator classes CLUSTER and CENTRAL are now notified. The operator message includes the Oracle Rdb version number, the user name, process ID and the bugcheck dump file name.

To disable this notification, define the logical name RDM$BUGCHECK_IGNORE_FLAGS to include "O" in the translation. This will result in the current behavior (no operator notification). There is, however, no way to alter the operator classes that are notified.

Following is an example operator message:

```
%%%%%%%%%%  OPCOM  15-JAN-1997 04:55:11.78  %%%%%%%%%%
Message from user SYSTEM on RDBNT
Oracle Rdb Database DUA0:[DB]DB.RDB;1 Event Notification
Process 00000EFA generating bugcheck dump file DUA0:[DB]RDSBUGCHK.DMP;
```

5.1.17 New Request Options for RDO, RDBPRE and RDB$INTERPRET

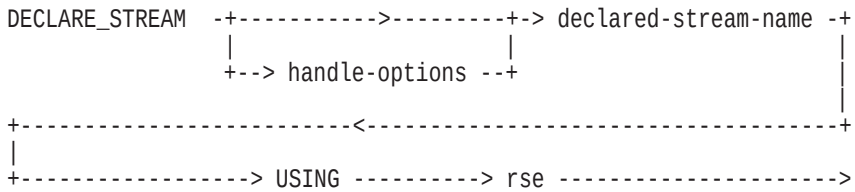
All OpenVMS platforms.

For this release of Rdb two new keywords have been added to the handle-options for the DECLARE_STREAM, the START_STREAM (undeclared format) and FOR loop statements. These changes have been made to RDBPRE, RDO and RDB$INTERPRET at the request of several RDO customers.

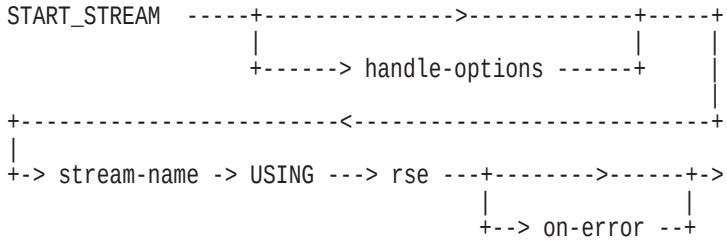
In prior releases the handle-options could not be specified in interactive RDO or RDB$INTERPRET. This has changed in Rdb7 but these allowed options will be limited to MODIFY and PROTECTED keywords. For RDBPRE all options listed will be supported. These option names were chosen to be existing keywords to avoid adding any new keywords to the RDO language.

The altered statements are shown in Example 5–1, Example 5–2 and Example 5–3.

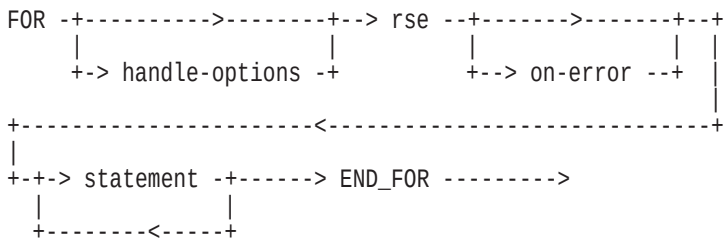
Example 5–1 DECLARE_STREAM Format



Example 5–2 START_STREAM Format

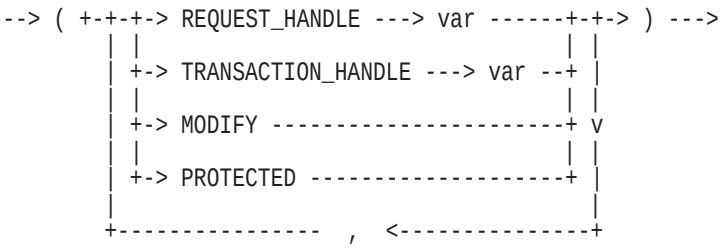


Example 5–3 FOR Format



Each of these statements references the syntax for the HANDLE-OPTIONS which has been revised and is shown below.

handle-options =



The following options are available for HANDLE-OPTIONS:

- REQUEST_HANDLE specifies the request handle for this request. This option is only valid for RDBPRE and RDML applications. It can not be used with RDB$INTERPRET, nor interactive RDO.

- TRANSACTION_HANDLE specifies the transaction handle under which this request executes. This option is only valid for RDBPRE and RDML applications. It can not be used with RDB$INTERPRET, nor interactive RDO.
- MODIFY specifies that the application will modify all (or most) records fetched from the stream or for loop. This option can be used to improve application performance by avoiding lock promotion from SHARED READ for the FETCH to PROTECTED WRITE access for the nested MODIFY or ERASE statement. It can also reduce DEADLOCK occurrence because lock promotions are avoided.

  This option is valid for RDBPRE, RDB$INTERPRET, and interactive RDO. This option is not currently available for RDML.

  For example:

```
RDO>   FOR (MODIFY) E IN EMPLOYEES WITH E.EMPLOYEE_ID = "00164"
cont>    MODIFY E USING E.MIDDLE_INITIAL = "M"
cont>    END_MODIFY
cont>  END_FOR
```

  This FOR loop uses the MODIFY option to indicate that the nested MODIFY is an unconditional statement and so aggressive locking can be undertaken during the fetch of the record in the FOR loop.
- PROTECTED specifies that the application may modify records fetched by this stream by a separate and independent MODIFY statement. Therefore, this stream should be protected from interference (aka Halloween affect). The optimizer will select a snapshot of the rows and store them in a temporary relation for processing, rather than traversing indexes at the time of the FETCH statement. In some cases this may result in poorer performance when the temporary relation is large and overflows from virtual memory to a temporary disk file, but the record stream will be protected from interference. The programmer is directed to the documentation for the Oracle Rdb logical names RDMS$BIND_WORK_VM and RDMS$BIND_WORK_FILE.

  This option is valid for RDBPRE, RDB$INTERPRET, and interactive RDO. This option is not currently available for RDML.

  The following example creates a record stream in a BASIC program using Callable RDO:

```
RDMS_STATUS = RDB$INTERPRET ('INVOKE DATABASE PATHNAME "PERSONNEL"')

RDMS_STATUS = RDB$INTERPRET ('START_STREAM (PROTECTED) EMP USING ' + &
                             'E IN EMPLOYEES')

RDMS_STATUS = RDB$INTERPRET ('FETCH EMP')

DML_STRING = 'GET ' +                                          &
                '!VAL = E.EMPLOYEE_ID;' +                      &
                '!VAL = E.LAST_NAME;' +                        &
                '!VAL = E.FIRST_NAME' +                        &
             'END_GET'

RDMS_STATUS = RDB$INTERPRET (DML_STRING, EMP_ID, LAST_NAME, FIRST_NAME)
```

  In this case the FETCH needs to be protected against MODIFY statements which execute in other parts of the application.

5.1.18 SQL*Net for Rdb Release 1.0

All OpenVMS platforms.

This release of Oracle Rdb includes the new SQL*Net for Rdb feature. SQL*Net for Rdb Release 1.0 is "middleware" that allows you access Rdb7 databases. Applications written using Developer/2000 (or those using OCI or PL/SQL interfaces) can directly access and manage data in an Rdb7 database.

SQL*Net for Rdb is installed as a part of Oracle SQL/Services Release 7.1.1. See the *Oracle Rdb7 Oracle SQL/Services Release Notes Release 7.1.1* for complete information and installation instructions.

5.1.19 Oracle Network Manager Software and GUI

To manage SQL*Net for Rdb install the Oracle Network Manager (located in the Netman31\Windows directory on the client CD). Acrobat help files are provided as part of the installation.

Caution

If your system is Windows 95 or Windows NT, you must install Oracle Network Manager in a 16-bit Oracle directory tree. For these systems, your Oracle directory tree most likely contains 32-bit applications. Therefore, because Oracle Network Manager is a 16-bit application, you must install it into an Oracle directory tree that contains other 16-bit applications.

5.1.20 RMU/SHOW STATISTIC Configuration File

The ability to import and export a configuration file has been added to the **RMU/SHOW STATISTIC** utility. The configuration file is a human-readable format file that can be created using any editor, or can be automatically generated from the **RMU/SHOW STATISTIC** utility using the current runtime configuration settings.

The configuration file is processed by the **RMU/SHOW STATISTIC** utility prior to opening the database or the binary input file, if the new optional **/CONFIGURE=config_filespec** qualifier is used. If you do not specify this qualifier, all of the variables are defaulted based on command-line qualifiers and logicals. The default configuration file type is **.CFG**.

The configuration file is processed in two passes. The first pass occurs before the database is opened and processes most of the configuration file entries. The second pass occurs after the database is open, and processes those variables that are database dependent, such as the "CUSTOMER_LINE_n" variables.

5.1.20.1 Configuration File Syntax

Each entry in the configuration file uses the general format:

variable=value;

The equal-sign ("=") separating the variable and value is required. Also note that each definition is terminated using the semicolon (";").

A comment is specified using either the "#" or "!" characters, and continues to the end of the current line. A comment can occur anywhere in a line, but always terminates that line.

Blank lines are ignored.

The "variable = value" is free-format. That is, spacing is not relevant to the parsing of the token. For example, the entry **STALL_LOG="STALL.LOG";** could be entered in the configuration file as:

```
STALL_LOG
=
"STALL.LOG"
;
```

However, for readability reasons, this type of formatting is not recommended.

Also, multiple entries can be put on a single line, although this is also not recommended, for readability reasons. For example, the following entry is permitted:

```
STALL_LOG="STALL.LOG"; DBKEY_LOG="DBKEY.LOG";
```

5.1.20.2 Creating Configuration Files

The configuration file can be created using the editor of your choice.

A new configuration file can also be automatically exported by selecting the "Tools" menu, using the "!" key, and then selecting the "Save current configuration" option. You will then be prompted to enter the name of the new configuration file. The default configuration file type is **.CFG**. For portability, comments are generated using the "#" character.

5.1.20.3 Importing Configuration Files

A new configuration file can be imported at any time by selecting the "Tools" menu, using the "!" key, and then selecting the "Import configuration settings" option. You will then be prompted to enter the name of the new configuration file. The default configuration file type is **.CFG**.

5.1.20.4 Nested Configuration Files

A nice feature of configuration files is that they can also be nested to any depth, using the special "INCLUDE" variable. This command variable will be described in detail below. The use of nested configuration files allows you to create hierarchies of configuration variable definitions.

Note

When using nested configuration files, remember that the *last* definition of a variable is the one that is used by the **RMU/SHOW STATISTIC** utility.

5.1.20.5 Variable Semantics

The "variable" is either a pre-defined configuration parameter known to the **RMU/SHOW STATISTIC** utility, or a user defined variable whose value will presumably be used later as the value of some other pre-defined configuration parameter. The complete list of pre-defined variables is defined later in this document.

All variable names must start with an alphabetic character. The "variable" name is case-insensitive.

All variables known to the **RMU/SHOW STATISTIC** utility have defaults; these defaults are described later in this document. It is not necessary to explicitly specify every variable. It is generally recommended that you only explicitly specify those variables whose default value is not acceptable to your application. Most configuration file variables replace existing command-line qualifiers or logicals.

5.1.20.6 Variable Types

Configuration variables have seven "types". These types dictate how the corresponding atomic value is used. The variable types are:

- **Numeric**. This is the most common type of configuration variable. Some, but not all, numeric variables have minimum and maximum values. Also, some, but not all, numeric variables have scale values, which dictate the number of fractional digits allowed. For example, a variable with a scale of "2" can be specified in terms of hundredths. A scale of "0" designates only a whole number is allowed (0 decimal positions).
- **Boolean**. These types of variables can use the "values" **TRUE** and **FALSE**, or **ENABLED** and **DISABLED** interchangeably. These values are *NOT* quoted. You can also use the numeric values "0" and "1".
- **String**. String variables specify a quoted "value", which are typically filespecs. Null string values are specified using an empty set of quotes ("").
- **Date**. Date variables specify a quoted "value" in the standard operating system date format.
- **Command**. Command variables cause an action specified by the quoted "value" to be performed. The most common variable is the "PRINT", "PROMPT" and "INQUIRE" variables. Command variables are *NOT* automatically exported.
- **Control**. Control variables are most commonly used to document *interesting* "values". They do not actually do anything, but they can be useful as the value of another variable. The most common variable is the "DATABASE" variable.
- **User Defined**. User defined variables are used as the values of other types of variables, and are treated as simple strings.

  User defined variables are not exported.

5.1.20.7 Value Semantics

The "value" is the numeric or quoted-string value to be assigned to the respective variable. The value, when quoted, is case-sensitive.

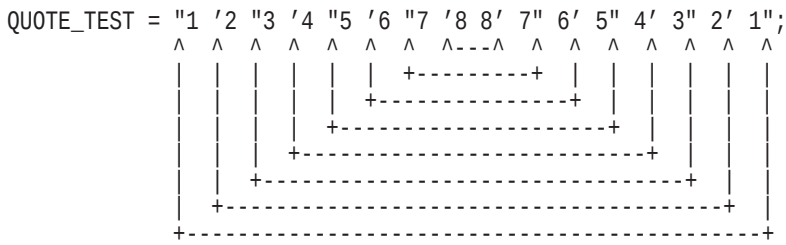
The values for numeric "variables" can be specified using either whole numbers or floating-point numbers, depending on the respective "scale" of the particular variable. The scale of a variable indicates the number of digits to the right of the decimal place.

Obviously, if you specify a value whose fractional portion exceeds the scale, then the value will be truncated. For example, if you specify "4.25" for a variable whose scale is "1" (meaning 1 digit to the right of the decimal point) then only "4.2" would be used.

When a variable's value is quoted using double-quotes ("), the evaluation of the value is treated as a hierarchy of values. The evaluation hierarchy is: variable name, logical name, atomic value. This hierarchy means that every value is first iteratively treated as another variable, then iteratively treated as a logical name, then finally as an atomic value.

Note that if you can also specify a single-word string value without the double-quotes, if so desired. For example, the value "FOOBAR" could be specified using either double-quotes or no quotes at all, since "FOOBAR" is a single-word string. However, the value "here and now" *must* be enclosed with either single- or double-quotes.

Quotes can also be nested, so long as they alternate. Note the "pairings" of the quotation marks in the following example:



The hierarchical evaluation semantics allows you to specify the value of one variable as the value of another. In the following example, the user defined variable **TEST** is defined with the value "0". The pre-defined numeric variable **CYCLE** is defined to the value "TEST", which because it is a variable will be evaluated as the value "0". Finally, the pre-defined command variable **PRINT** will be evaluated as "CYCLE", which because it is also a variable will be evaluated as the value "0".

```
TEST = 0
CYCLE = "TEST"
PRINT = "CYCLE"
```

It is interesting to note that, even though the **CYCLE** variable is of type "numeric", you can specify a quoted string value so long as it ultimately evaluates to a numeric value.

It is also interesting to note that the hierarchical evaluation of values prevents a quoted-value from being the same as the variable name. For instance, in the configuration file entry

INCLUDE="INCLUDE";

the value "INCLUDE" obviously cannot be evaluated as the variable name "INCLUDE", since this would be recursive.

Using logicals as the value of variables is very useful for certain variables, such as the **INCLUDE** variable (see section "Nested Configuration Files"). This allows users to customize their configuration files through the use of logicals while still utilizing a centralized configuration file hierarchy.

Some variables have special "keywords" for their values. These keywords must be specified in upper-case, exactly as they are described later in this document. Be especially careful when specifying user defined variables whose name conflicts with any of the valid keywords. When using atomic values for keywords, it is recommended to use single-quotes to avoid inadvertant evaluation of the keyword as another variable.

When a variable's value is quoted using single-quotes (’), the value is treated as a simple string. For more information and examples on the difference between single-quoted and double-quoted strings, refer to the “Print Variable” section below.

5.1.20.8 PRINT & PROMPT Command Variables

The **PRINT** and **PROMPT** variables are useful for debugging and tracing execution of the configuration file. For instance, the following example demonstrates how to display a variable's “default” value, and then the “new” value initialized from the configuration file.

```
PRINT="CYCLE";
CYCLE=5;
PRINT="CYCLE";
```

The output from the above example is:

```
# This file was generated by the SHOW STATISTIC utility.
# Generated on 16-APR-1997 10:35:05.48
line 3: variable "PRINT" value "CYCLE = 0"
line 5: variable "PRINT" value "CYCLE = 5"
```

The **PRINT** variable is also useful for tracing execution of the configuration file, which may be very useful when using nested configuration files. For example, the **PRINT** command can be used to display simple messages to the log file:

```
PRINT="Now initializing CYCLE variable";
CYCLE=5;
```

The output from the above example is:

```
# This file was generated by the SHOW STATISTIC utility.
# Generated on 16-APR-1997 10:35:05.48
line 1: variable "PRINT" value "Now initializing CYCLE variable"
```

Care should be taken when printing out single-word messages, as these may be evaluated as variables. Consider the following example:

```
HERE="there";
PRINT="here";
PRINT='here';
PRINT="here and there";
```

In this example, the value of the first **PRINT** variable is “there” not “here”, because “here” is a user defined variable. The value of the second **PRINT** variable “here” because it is a single-quoted simple string. Finally, the value of the third **PRINT** variable is “here and there” because it is an atomic string.

5.1.20.9 Variable Value Redirection

Redirecting is an interesting use of a variable, especially a variable whose value is input by the user. Consider the following configuration file example:

```
OVER="AND OUT";
THERE="OVER";
HELLO="THERE";
FOO="HELLO";
```

Setting the variable “FOO” to the value “HELLO” causes an implicit redirection of its value to be the final value of all of its own value(s). This results in the “FOO” variable being set to the value “AND OUT”, which is the final value of all values.

The **REDIRECT** command variable is extremely useful following an **INQUIRE** command, where the user entered the name of another variable to use. By redirecting the variable's value, you can essentially implement a menu mechanism. For example, consider the following configuration file example:

```
VALUE1="10";
VALUE2="20";
VALUE3="30";
PROMPT="Enter VALUE1, VALUE2 or VALUE3";
INQUIRE="VALUE";
REDIRECT="VALUE";
PRINT="VALUE";
```

Note how this differs from setting the "VALUE" variable to itself.

5.1.20.10 Configuration File Example

The following is an example of a possible configuration file:

```
# This file was generated by the SHOW STATISTIC utility.
DATABASE = "KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1"
# Generated on  8-APR-1997 15:08:33.72
EVENT_DESCRIPTION="ENABLE transactions MAX_RATE INITIAL 5 EVERY 2 INVOKE EVENT";
EVENT_DESCRIPTION="DISABLE transactions MAX_CUR_RATE";
EVENT_DESCRIPTION="ENABLE transactions MIN_CUR_RATE INITIAL 5 EVERY 2 LIMIT 5";

ALARM = 0;
BROADCAST = FALSE;
CYCLE = 0;
INTERACTIVE = TRUE;
HISTOGRAM = FALSE;
REOPEN_INTERVAL = 0;
REFRESH_INTERVAL = 1;
NOTIFY = "OPER1,OPER11";
STALL_LOG = "stall.log";
TIMEOUT_LOG = "timeout.log";
DEADLOCK_LOG = "deadlock.log";
DBKEY_LOG = "";
TX_DURATION = "TOTAL";
LOGICAL_AREA = "INDIVIDUAL";
STALL_MESSAGE = "ACTUAL";
ACTIVE_USER = "ACTUAL";
CHECKPOINT_TX = "ACTUAL";
CHECKPOINT_SORT = "OLDEST_CHECKPOINT";
AIJ_ARBS_PER_IO = 99.9;
AIJ_BKGRD_ARB_RATIO = 50.5;
AIJ_BLKS_PER_IO = 2.5;
AIJ_SEC_TO_EXTEND = 60.5;
BTR_FETCH_DUP_RATIO = 15.5;
BTR_LEF_FETCH_RATIO = 25.5;
DBR_RATIO = 15.5;
FULL_BACKUP_INTRVL = 6;
GB_IO_SAVED_RATIO = 85.5;
GB_POOL_HIT_RATIO = 85.5;
LB_PAGE_HIT_RATIO = 75.5;
MAX_HASH_QUE_LEN = 2;
MAX_LOCK_STALL = 2.5;
MAX_TX_DURATION = 15.5;
PAGES_CHECKED_RATIO = 10.5;
RECS_FETCHED_RATIO = 20.5;
RECS_STORED_RATIO = 20.5;
RUJ_SYNC_IO_RATIO = 10.5;
VERB_SUCCESS_RATIO = 25.5;
```

5.1.20.11 Log File

If any problem occurs processing the specified configuration file, a log file is automatically created. The log file's name is the same as the configuration filespec with the ".CFG" file type replaced with a ".LOG" filetype. For example, if the configuration filespec is "CONFIG.CFG" then the corresponding log filespec would be "CONFIG.LOG".

Logfile entries have the format: "line #: message". The following example shows a problem trying to include a nested configuration file; the offending command is on line 4 of the configuration file.

```
# This file was generated by the SHOW STATISTIC utility.
# Generated on  6-APR-1997 07:20:00.73
line 4: INCLUDE="config.cfg";
line 4: command "INCLUDE" failed
%COSI-E-FLK, file currently locked by another user
```

You can also specify the **/LOG** qualifier, which lists each processed variable to the corresponding log file. This qualifier is highly recommended until you become more familiar with using the configuration files.

For example, the following **CONFIG.LOG** was produced using the **/LOG** qualifier in conjunction with the **/CONFIGURE=CONFIG.CFG** qualifier. Note that, in some cases, the set of valid keywords is included as a comment for your convenience.

```
# This file was generated by the SHOW STATISTIC utility.
DATABASE = "KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1";
# Generated on 15-APR-1997 15:31:52.92
ALARM = 0;
BROADCAST = TRUE;
CYCLE = 0;
DASHBOARD_UPDATE = TRUE;
INTERACTIVE = TRUE;
HISTOGRAM = FALSE;
REOPEN_INTERVAL = 0;
RESET = FALSE;
REFRESH_INTERVAL = 1;
OUTPUT = "";
SCREEN = "Stall Messages";
STALL_LOG = "stall.log";
TIMEOUT_LOG = "timeout.log";
DEADLOCK_LOG = "deadlock.log";
DBKEY_LOG = "";
CLUSTER_NODES = "";
NOTIFY = "OPER1,OPER11"; # "CENTRAL" "DISKS" "CLUSTER" "SECURITY" "OPER2" "OPER3" "OPER4" "OPER5" "OPER6" "OPER7" "OPER8" "OPER9"
TX_DURATION = "TOTAL"; # "TOTAL" "READ_WRITE" "READ_ONLY"
LOGICAL_AREA = "INDIVIDUAL"; # "INDIVIDUAL" "AGGREGATE"
STALL_MESSAGE = "ACTUAL"; # "ACTUAL" "ELAPSED"
ACTIVE_USER = "ACTUAL"; # "ACTUAL" "ELAPSED"
CHECKPOINT_TX = "ACTUAL"; # "ACTUAL" "ELAPSED"
CHECKPOINT_SORT = "OLDEST_CHECKPOINT"; # "OLDEST_CHECKPOINT" "OLDEST_TRANSACTION" "OLDEST_QUIET_POINT"
AIJ_ARBS_PER_IO = 99.9;
AIJ_BKGRD_ARB_RATIO = 50.5;
AIJ_BLKS_PER_IO = 2.5;
AIJ_SEC_TO_EXTEND = 60.5;
BTR_FETCH_DUP_RATIO = 15.5;
BTR_LEF_FETCH_RATIO = 25.5;
DBR_RATIO = 15.5;
FULL_BACKUP_INTRVL = 6;
GB_IO_SAVED_RATIO = 85.5;
GB_POOL_HIT_RATIO = 85.5;
LB_PAGE_HIT_RATIO = 75.5;
MAX_HASH_QUE_LEN = 2;
MAX_LOCK_STALL = 2.5;
MAX_TX_DURATION = 15.5;
PAGES_CHECKED_RATIO = 10.5;
RECS_FETCHED_RATIO = 20.5;
```

```
RECS_STORED_RATIO = 20.5;
RUJ_SYNC_IO_RATIO = 10.5;
VERB_SUCCESS_RATIO = 25.5;
```

5.1.20.12 Pre-Defined Variable Definitions

This section describes the configuration "variables" that are known to the **RMU/SHOW STATISTIC** utility. The variables are listed in alphabetical order, but may not necessarily be exported in this order.

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
ACTIVE_USER	String					Specifies the "Active User Stall Messages" screen configuration options. The valid keywords are "ACTUAL" and "ELAPSED".
AIJ_ARBS_PER_IO	Numeric	2	0		1	Specifies the default online analysis value of AIJ request blocks per AIJ I/O. The RDM$BIND_STATS_AIJ_ARBS_PER_IO logical name and the RDB_BIND_STATS_AIJ_ARBS_PER_IO configuration parameter allow you to override the default value of AIJ request blocks per AIJ I/O. The default is 2 blocks. You can also set this threshold from the configuration submenu in the Performance Monitor AIJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
AIJ_BKGRD_ARB_RATIO	Numeric	50	0		1	Specifies the default online analysis value for the background AIJ request block threshold. The RDM$BIND_STATS_AIJ_BKGRD_ARB_RATIO logical name and the RDB_BIND_STATS_AIJ_BKGRD_ARB_RATIO configuration parameter allow you to override the default value for the background AIJ request block threshold. The default value is 50. You can also set this threshold from the configuration submenu in the Performance Monitor AIJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
AIJ_BLKS_PER_IO	Numeric	2	0		1	Specifies the default online analysis value of blocks per AIJ I/O. The RDM$BIND_STATS_AIJ_BLKS_PER_IO logical name and the RDB_BIND_STATS_AIJ_BLKS_PER_IO configuration parameter allow you to override the default value of blocks per AIJ I/O. The default value is 2. You can also set this threshold from the configuration submenu in the Performance Monitor AIJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
AIJ_SEC_TO_EXTEND	Numeric	60	0		1	Specifies the default online analysis value of seconds to AIJ extend. The RDM$BIND_STATS_AIJ_SEC_TO_EXTEND logical name and the RDB_BIND_STATS_AIJ_SEC_TO_EXTEND configuration parameter allow you to override the default value of seconds to AIJ extend. The default value is 60. You can also set this threshold from the configuration submenu in the Performance Monitor AIJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
ALARM	Numeric	0	0		0	Establishes an alarm interval (in seconds) for the Stall Messages screen from the command line. This is useful when you plan to submit the RMU Show Statistics command as a batch job. This variable supercedes the **/ALARM=seconds** qualifier.
AUTO_ACTIVE_DETECT	Boolean	F	F	T		When the AUTO_NODE_DETECT variable is set to the value TRUE, specifies whether or not to actively or passively detect new nodes joining the cluster. Active detection may incur an I/O operation per screen refresh. Passive detection relies on other users on the current node to passively refresh the node information.
AUTO_NODE_DETECT	Boolean	F	F	T		Specifies whether or not to automatically detect new nodes joining the cluster. When the utility detects a new node joining the cluster, the cluster statistics menu will be automatically displayed.
AUTO_RECONNECT	Boolean	F	F	T		Specifies whether or not to automatically reconnect to disconnected nodes.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
BROADCAST	Boolean	T	F	T		Specifies whether or not to broadcast messages. The Broadcast qualifier is the default, if broadcasting of certain messages has been enabled with DCL SET BROADCAST. If broadcasting has been disabled with the DCL SET BROADCAST=none command, broadcast messages are not displayed, even if you specify the RMU Show Statistics command with the Broadcast qualifier. This variable supercedes the **/[NO]BROADCAST** qualifier.
BTR_FETCH_DUP_ RATIO	Numeric	15	0		1	Specifies the default online analysis value of the B-tree duplicate fetch threshold. The RDM$BIND_STATS_BTR_FETCH_ DUP_RATIO logical name and the RDB_ BIND_STATS_BTR_FETCH_DUP_RATIO configuration parameter allow you to override the default value of the B-tree duplicate fetch threshold. The default threshold is 15. You can also set this threshold from the configuration submenu in the Performance Monitor Index Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
BTR_LEF_FETCH_ RATIO	Numeric	25	0		1	Specifies the default online analysis value of the B-tree leaf node fetch threshold. The RDM$BIND_STATS_BTR_LEF_ FETCH_RATIO logical name and the RDB_BIND_STATS_BTR_LEF_FETCH_ RATIO configuration parameter allow you to override the default value of the B-tree leaf node fetch threshold. The default threshold is 25. You can also set this threshold from the configuration submenu in the Performance Monitor Index Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
CHECKPOINT_TX	String					Specifies the “Checkpoint” screen transaction configuration options. The valid keywords are “ACTUAL” and “ELAPSED”.
CHECKPOINT_ALARM	Numeric	0	0		0	Establishes an alarm interval (in seconds) for the Checkpoint Information screen. This is useful when you plan to submit the RMU Show Statistics command as a batch job.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
CHECKPOINT_SORT	String					Specifies the "Checkpoint" screen sort configuration options. The valid keywords are "OLDEST_CHECKPOINT", "OLDEST_TRANSACTION" and "OLDEST_QUIET_POINT".
CLUSTER_NODES	String					Identifies the set of nodes that are to participate in statistics collection for the current session. The node name(s) should be comma separated. The keyword **ALL_OPEN** indicates that statistics should be collected from all nodes on which the database is currently open. Note that the **ALL_OPEN** keyword is never automatically generated. This variable supercedes the **/CLUSTER=node_list** qualifier.
CUSTOM_LINE_n	String			18		Specifies the name of the statistic field to the located on line "n" of the "Custom Statistics" screen. Statistics may be entered for lines "1" through "36", although the number of lines that can actually be displayed depends on your terminal. The statistic name must be specified *exactly* as it appears on its "home" screen, including leading spaces. Duplicate statistics as well as duplicated line numbers are detected. Note that the specified custom statistics fields are not evaluated until *after* the database has been opened. Opening the database activates the various screens, which determines the set of custom statistic fields that can be specified. Therefore, some custom statistics fields may not always be available, depending on which database attributes (for instance "global buffers") are active.
CYCLE	Numeric	0	0		0	Specifies the interval (in seconds) to automatically migrate to the next screen in the current sub-menu. This variable supercedes the **/CYCLE=seconds** qualifier.
DASHBOARD_UPDATE	Boolean	D	D	E		Specifies whether or not dashboard updates are permitted, if you have the proper privileges. This variable supercedes the **/OPTION=[NO]UPDATE** qualifier.
DATABASE	Control					This control variable identifies the database from which the configuration file was generated. This variable is for documentation purposes only, and is primarily useful only when the **/LOG** qualifier is specified.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
DBKEY_LOG	String			255		Logs the records accessed during a given processing period by the various attached processes. The file-spec is the name of the file to which all accessed dbkeys are logged. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supercedes the **/DBKEY_LOG=dbkey_log** qualifier.
DBR_RATIO	Numeric	15	0		1	Specifies the default online analysis value of the DBR invocation threshold. The RDM$BIND_STATS_DBR_RATIO logical name and the RDB_BIND_STATS_DBR_RATIO configuration parameter allow you to override the default value of the DBR invocation threshold. The default threshold is 15. You can also set this threshold from the configuration submenu in the Performance Monitor RUJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
DEADLOCK_FULL_DISPLAY	Boolean	D	D	E		Specifies whether or not the "Lock Deadlock History" screen is to display all processes or just those with deadlock messages.
DEADLOCK_LOG	String			255		Specifies that lock deadlock message are to be written to the specified file. This can be useful when you notice a great number of lock deadlock messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the Deadlock_Log qualifier can be reviewed later so that the problem can be traced and resolved. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supercedes the **/DEADLOCK_LOG=deadlock_log** qualifier.
EVENT_DESCRIPTION	Command					This command describes a SHOW STATISTICS “event” and either enables a new event or disables an active event. Please refer to the “User-Defined Events” section of the document for more information.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
FULL_BACKUP_ INTRVL	Numeric	6	0		0	Specifies the online analysis full database backup threshold. The RDM$BIND_STATS_FULL_BACKUP_ INTRVL logical name and the RDB_ BIND_STATS_FULL_BACKUP_INTRVL configuration parameter allow you to override the full database backup threshold. The default threshold is 6. You can also set this threshold from the configuration submenu in the Performance Monitor Recovery Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
GB_IO_SAVED_RATIO	Numeric	85	0		1	Specifies the online analysis GB IO-saved default threshold. The RDM$BIND_STATS_GB_IO_SAVED_ RATIO logical name and the RDB_ BIND_STATS_GB_IO_SAVED_RATIO configuration parameter allow you to override the GB IO-saved default threshold. The default threshold is 85. You can also set the global buffer IO-saved threshold from the configuration submenu in the Performance Monitor Buffer Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
GB_POOL_HIT_RATIO	Numeric	85	0		1	Specifies the online analysis GB pool hit default threshold. The RDM$BIND_STATS_GB_POOL_ HIT_RATIO logical name and the RDB_ BIND_STATS_GB_POOL_HIT_RATIO configuration parameter allow you to override the GB pool hit default threshold. The default threshold is 85. You can also set the global buffer pool hit threshold from the configuration submenu in the Performance Monitor Buffer Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
HISTOGRAM	Boolean	F	F	T		Directs Oracle RMU to display the initial statistics screen in the numbers display mode or the graph display mode. The Histogram qualifier specifies the graph display mode. The Nohistogram qualifier specifies the numbers display mode. This variable supercedes the **/[NO]HISTOGRAM** qualifier.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
INCLUDE	Command					This command temporarily switches processing to the quoted string value that defines a new configuration file. This configuration file, in turn, may also specify the **INCLUDE** command to switch to yet another configuration file. When processing of the command completes, processing of the current configuration file continues. Remember that variable definitions are *superceded* by subsequent references. The "INCLUDE" variable detects infinite recursion. The "INCLUDE" variable is not exported.
INQUIRE	Command					This command prompts the user to enter a value for the specified variable. This command is typically used after the **PROMPT** command to prompt the user to enter a value for certain variables. This command does not work when a configuration file is imported. The "INQUIRE" variable is not exported.
INTERACTIVE	Boolean	T	F	T		Displays the statistics dynamically to your terminal. The Interactive qualifier is the default when you execute the RMU Show Statistics command from a terminal. You can use the Nointeractive qualifier with the Output qualifier to generate a binary statistics file without generating a terminal display. The Nointeractive qualifier is the default when you execute the RMU Show Statistics command from a batch job. Note that most of these variable are not interesting when the **INTERACTIVE** variable is set to **FALSE**. This variable supercedes the **/[NO]INTERACTIVE** qualifier.
LB_PAGE_HIT_RATIO	Numeric	75	0		1	Specifies the online analysis LB/AS page hit default threshold. The RDM$BIND_STATS_LB_PAGE_ HIT_RATIO logical name and the RDB_ BIND_STATS_LB_PAGE_HIT_RATIO configuration parameter allow you to override the LB/AS page hit default threshold. The default is 75. You can also set the local buffer pool hit threshold from the configuration submenu in the Performance Monitor Buffer Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
LOGICAL_AREA	String					Specifies the "By Logical Area" screen configuration options. The valid keywords are "INDIVIDUAL" and "AGGREGATE".

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
MAX_HASH_QUE_LEN	Numeric	2	0		0	Specifies the online analysis hash table queue length default threshold. The RDM$BIND_STATS_MAX_HASH_QUE_LEN logical name and the RDB_BIND_STATS_MAX_HASH_QUE_LEN configuration parameter allow you to override the hash table queue length default threshold. The default threshold is 2 rows. You can also set the hash table queue length threshold from the configuration submenu in the Performance Monitor Transaction Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
MAX_LOCK_STALL	Numeric	2	0		1	Specifies the online analysis lock stall default threshold. The RDM$BIND_STATS_MAX_LOCK_STALL logical name and the RDB_BIND_STATS_MAX_LOCK_STALL configuration parameter allow you to override the lock stall default threshold. The default threshold is 2 seconds. You can also set this threshold from the configuration submenu in the Performance Monitor Locking Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
MAX_TX_DURATION	Numeric	15	0		1	Specifies the online analysis transaction duration default threshold. The RDM$BIND_STATS_MAX_TX_DURATION logical name and the RDB_BIND_STATS_MAX_TX_DURATION configuration parameter allow you to override the transaction duration default threshold. The default value is 15. You can also set the transaction duration threshold from the configuration submenu in the Performance Monitor Transaction Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
NOTIFY	String					Notifies the specified system operator or operators when a stall process exceeds the specified alarm interval by issuing a broadcast message and ringing a bell at the terminal receiving the message. The valid operator classes are: CENTRAL, CLUSTER, DISKS, SECURITY, and OPER1 through OPER12. Multiple operator classes can be comma-separated; for example "OPER11,OPER12". Be sure to use single-quotes for this variable. This variable supercedes the **/NOTIFY=oper_classes** qualifier.
OUTPUT	String			255		Specifies that the collected statistics are to be written to the specified binary output file. This variable supercedes the **/OUTPUT=binary_file** qualifier.
PAGES_CHECKED_ RATIO	Numeric	10	0		1	Specifies the online analysis pages checked default threshold. The RDM$BIND_STATS_PAGES_ CHECKED_RATIO logical name and the RDB_BIND_STATS_PAGES_CHECKED_ RATIO configuration parameter allow you to override the pages checked default threshold. The default threshold is 10 pages. You can also set this threshold from the configuration submenu in the Performance Monitor Record Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
PRINT	Command					This command prints the "value" to the log file. This variable is useful if you do not want to use the **/LOG** qualifier, but want to display the value of a variable. This variable is also useful for displaying the "initial" value of a variable before it is changed in the configuration file. Interesting uses of the "PRINT" command have already been described above in this document. The "PRINT" variable is not exported.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
PROMPT	Command					This command prints the "value" to the terminal. This variable is typically used in conjunction with the **INQUIRE** variable, but can also be useful if you do not want to use the **/LOG** qualifier, but want to display the value of a variable. This command does not work when a configuration file is imported. Interesting uses of the "PROMPT" command have already been described above in this document. The "PROMPT" variable is not exported.
RECS_FETCHED_ RATIO	Numeric	20	0		1	Specifies the online analysis records fetched default threshold. The RDM$BIND_STATS_RECS_ FETCHED_RATIO logical name and the RDB_BIND_STATS_RECS_FETCHED_ RATIO configuration parameter allow you to override the records fetched default threshold. The default threshold is 20 records. You can also set this threshold from the configuration submenu in the Performance Monitor Record Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
RECS_STORED_RATIO	Numeric	20	0		1	Specifies the online analysis records stored default threshold. The RDM$BIND_STATS_RECS_STORED_ RATIO logical name and the RDB_ BIND_STATS_RECS_STORED_RATIO configuration parameter allow you to override the records stored default threshold. The default threshold is 20 records. You can also set this threshold from the configuration submenu in the Performance Monitor Record Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
REDIRECT	Command					This command forces the redirection of another variable's value into itself. This is the means by which variable indirection can be achieved. Command variables and control variables cannot be redirected, but all other variables can be. For example, if the variable "HELLO" has the value "THERE", and the variable "FOO" has the value "HELLO", then the command "REDIRECT="FOO"" causes the variable "FOO" to have the value "THERE". This command variable is typically used after defining the variable's value using the **INQUIRE** command. The "REDIRECT" variable is not exported.
REFRESH_INTERVAL	Numeric	3	0		0	Specifies the statistics collection interval in seconds. If you omit this qualifier, a sample collection is made every 3 seconds. The integer has a *normal* range of 1 to 180 (1 second to 3 minutes). However, if you specify a negative number for the Time qualifier, the RMU Show Statistics command interprets the number as hundredths of a second. For example, Time=–20 specifies an interval of 20/100 or 1/5 of a second. This variable supercedes the **/TIME=seconds** qualifier.
REOPEN_INTERVAL	Numeric	0	0		0	After the specified interval, closes the current output file and opens a new output file without requiring you to exit from the Performance Monitor. The new output file has the same name as the previous output file, but the version number is incremented by 1. This variable supercedes the **/REOPEN_INTERVAL=seconds** qualifier.
RESET	Boolean	F	F	T		Specifies whether or not the statistics are to be automatically reset prior to being displayed. This variable is always exported with the value **FALSE** regardless of its initial value. This variable supercedes the **/[NO]RESET** qualifier.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
RUJ_SYNC_IO_RATIO	Numeric	10	0		1	Specifies the online analysis synchronous RUJ I/O default threshold. The RDM$BIND_STATS_RUJ_SYNC_IO_RATIO logical name and the RDB_BIND_STATS_RUJ_SYNC_IO_RATIO configuration parameter allow you to override the synchronous RUJ I/O default threshold. The default threshold is 10. You can also set this threshold from the configuration submenu in the Performance Monitor RUJ Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.
SCREEN	String			255		Specifies that name of the initial screen to be displayed. This variable supercedes the **/SCREEN=screen_name** qualifier.
STALL_LOG	String			255		Specifies that stall message are to be written to the specified file. This can be useful when you notice a great number of stall messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the Stall_Log qualifier can be reviewed later so that the problem can be traced and resolved. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supercedes the **/STALL_LOG=stall_log** qualifier.
STALL_MESSAGE	String					Specifies the “Stall Messages” screen configuration options. The valid keywords are “ACTUAL” and “ELAPSED”.
SYSTEM_LOGICAL_AREAS	Boolean	T	F	T		Specifies whether or not to display system relations on the “Logical Area” statistics screens. This variable only applies to **Rdb** databases; it does not apply to **DBMS** databases.
TIMEOUT_FULL_DISPLAY	Boolean	D	D	E		Specifies whether or not the "Lock Timeout History" screen is to display all processes or just those with deadlock messages.

Legend

T—True
F—False
E—Enabled
D—Disabled

Variable Name	Type	Def. Value	Min. Value	Max. Value	Scale	Description
TIMEOUT_LOG	String			255		Specifies that lock timeout message are to be written to the specified file. This can be useful when you notice a great number of lock timeout messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the Timeout_Log qualifier can be reviewed later so that the problem can be traced and resolved. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supercedes the **/TIMEOUT_LOG=timeout_log** qualifier.
TX_DURATION	String					Specifies the "Tx Duration" screen configuration options. The valid keywords are "TOTAL", "READ_WRITE" and "READ_ONLY".
UNSET	Command					This command removes a user defined variable from the symbol table. It is not necessary to unset user defined variables prior to changing their value. It is only necessary to unset a variable if you do not wish it to exist for some reason. Note that using the single-quote is not supported for this command because variables can be specified using double-quotes only. The "UNSET" variable is not exported.
VERB_SUCCESS_RATIO	Numeric	25	0		1	Specifies the online analysis verb success default threshold. The RDM$BIND_STATS_VERB_SUCCESS_RATIO logical name and the RDB_BIND_STATS_VERB_SUCCESS_RATIO configuration parameter allow you to override the verb success default threshold. The default threshold is 25. You can also set the verb success threshold from the configuration submenu in the Performance Monitor Transaction Analysis screen. See Performance Monitor Online Analysis Facility for more information on the Performance Monitor Online Analysis facility.

Legend

T—True
F—False
E—Enabled
D—Disabled

Note

The configuration file is not yet supported for **Digital Unix**.

5.1.20.13 Example of Poorly Constructed Configuration File

The following is an example of a really bad configuration file. It is presented here to provide examples of what *NOT* to do.

```
PRINT="DATABASE";               # display current database name
PRINT='DATABASE';               # should display "database"

UNSET="FOO";                    # warning: unknown variable "FOO"
FOO=5;
UNSET='FOO';                    # warning: unknown variable "FOO" (not variable)
UNSET="FOO";
PRINT="FOO";                    # should display "FOO" instead of "5"

123=456;    # warning: incorrect variable name "123"
PRINT="PRINT";                  # should display "PRINT"
INCLUDE="INCLUDE";              # %COSI-E-FNF, file not found
INCLUDE="CONFIG.CFG";           # %COSI-F-EEXIST, file exists

CUSTOM_LINE_5="transactions     # warning: unterminated string (expecting " or ')
;
CUSTOM_LINE_5="transactions";
CUSTOM_LINE_5="verb successes";  # warning: replacing defined variable "CUSTOM_LINE_5"
CUSTOM_LINE_6="verb successes";  # warning: replacing defined statistic "transactions"
CUSTOM_LINE_7="verb failures";
CUSTOM_LINE_8="verb confusion";  # warning: custom statistic field "verb confusion" not found

HERE="there";
PRINT="here and now";           # should display "here and now"
PRINT="here";                   # should display "there"
PRINT='here';                   # should display "here"

YES="yes";
PRINT="yes";                    # should display "YES = yes"

PRINT = "CYCLE";                # should display "CYCLE = 0"
CYCLE = "5";                    # this is reasonable
CYCLE = 5;                      # this also
PRINT = "CYCLE";                # should display "CYCLE = 5"
CYCLE = ALARM;
PRINT = "CYCLE";                # should display "CYCLE = 0"

NO="no";
PRINT="no";                     # should display "NO = no"

BROADCAST = BROADCAST;          # warning: invalid value "BROADCAST"
NOTIFY = "CLUSTER";             # should display "NOTIFY = ''"
```

5.1.21 Optimizer Zig-Zag Strategy now uses Temporary Relation

With the release of Oracle Rdb7, Rdb may use a temporary relation for storing intermediate results fetched from the inner loop of a join. This temporary relation avoids a rescan of the index when performing duplicate processing for the outer loop and results in lower I/O cost for zig-zag match strategy.

The side effect is that Rdb now uses virtual memory for the temporary table, which can be controlled using the logical name RDMS$BIND_WORK_VM. If there are many duplicate values in the inner loop of the join then the temporary relation may overflow into a temporary file on disk, the location of which is controlled by the logical name RDMS$BIND_WORK_FILE. These logical names are documented in the Oracle Rdb7 Guide to Database Performance and Tuning.

The overflow from virtual memory to disk file may be avoided by assigning a larger byte count to the logical name RDMS$BIND_WORK_VM before attaching to the database. Making this value larger may allow queries to keep these temporary results in memory and reduce the physical I/O to disk.

Increasing the working set for the process may be required if some queries use many zig-zag strategies, as each will create a temporary relation. For instance, this portion of a query strategy show that zig-zag is used for the Match strategy on both the inner and outer loops. The inner loop is what manages the temporary relation.

```
Reduce  Sort
Cross block of 8 entries
  Cross block entry 1
    Conjunct
    Match
      Outer loop      (zig-zag)
        Conjunct
        Leaf#01 Sorted TABLE_A Card=11924
          FgrNdx  INDEX_1 [3:3] Bool Fan=5
          BgrNdx1 INDEX_2 [2:2] Fan=5
      Inner loop      (zig-zag)
        Conjunct        Get     Retrieval by index of relation TABLE_B
          Index name  INDEX_3 [2:2]
  Cross block entry 2
    ...etc...
```

5.1.22 SHOW STATISTIC "Cluster Statistics Collection and Presentation" Feature

The **Rdb7** RMU/SHOW STATISTIC utility has been enhanced with the "Cluster Statistics Collection and Presentation" feature. The purpose of this feature is to provide for the realtime collection and presentation of database statistics from other nodes within the cluster where the database is currently being accessed.

The "Cluster Statistics Collection and Presentation" feature is invoked using the **RMU/SHOW STATISTIC** utility **/CLUSTER** command qualifier. This qualifier specifies the list of remote nodes from which statistics collection and presentation is to be performed. The collected statistics are merged with the current node's information and displayed using the normal statistics screens.

The following list shows the use of the /CLUSTER qualifier:

- If /CLUSTER is specified by itself, then remote statistics collection will be performed on all cluster nodes on which the database is currently open.
- If /CLUSTER=(node_list) is specified, then remote statistics collection will be performed on the specified nodes only, even if the database is not yet open on that node.
- If /CLUSTER is NOT specified, or /NOCLUSTER (the default) is specified, then cluster statistics collection will NOT be performed. However, the user may still enable cluster-wide statistics collection online using the "Tools" menu.

Up to 95 different cluster nodes can be specified using the **/CLUSTER** qualifier. There is a maximum number of 95 cluster nodes because **Rdb7** only supports a maximum number of 96 nodes per database. The "current" node is always included in the list of nodes from which statistics collection is to be performed.

The Cluster-wide "Cluster Statistics Collection and Presentation" feature is currently only available on **OpenVMS** platforms. The feature may be available on **UNIX** in a future release.

You do *not* have to have a SHOW STATISTIC utility running on the specified remote nodes. The database does not even have to be open on the remote node. These events are automatically handled by the feature.

The following example shows the use of the **/CLUSTER** qualifier to initiate statistics collection and presentation from two remote nodes:

```
$ RMU /SHOW STATISTIC /CLUSTER=(BONZAI, ALPHA4) MF_PERSONNEL
```

Remote nodes can also be added and removed online at runtime. Use the "Cluster Statistics" option located in the "Tools" menu. The "Tools" menu is obtained using the "!" on-screen menu option.

The "Cluster Statistics" menu contains up to six options, depending on the current state of remote statistics collection. These menu options are the following:

1. "Manually add remote node to collection". This option allows you to type in the name of a remote node for which you wish to start statistics collection. Up to 95 different remote nodes can be added using this option.

   This option is useful for entering a comma-separated list of cluster nodes to be connected. Alternately, the nodename specified can be a logical whose definition is a comma-separated list of cluster nodes to be connected. When defining the logical, be sure to place the comma-separated cluster list in quotes so that it is processed as a single logical and not a search-list logical. For example, the following defines a logical containing the names of three cluster nodes:

   ```
   $ DEFINE NODENAME "BONZAI,ALPHA4,ALPHA5"
   ```

   When a list of cluster nodes is specified, connection to the nodes is terminated upon encountering any error condition. Any previously connected nodes will remain connected. You will be notified when an error condition is encountered.

   Note that you cannot add the current node to the cluster statistics collection list because statistics are always collected on the current node.

2. "Select cluster node for collection". This option displays an alphabetical menu of all available cluster nodes in the cluster and lets you easily select the cluster node for which you wish to start statistics collection. Up to 95 different cluster nodes can be added using this option.

3. "Remove cluster node from collection". This option displays a menu of cluster nodes that are currently activated with the cluster statistics collection and presentation feature. Selecting one of these nodes will stop cluster statistics collection from that node. This option is only available when cluster statistics are being actively collected.

   Note that you cannot remove the current node from the cluster statistics collection list.

4. "Display active collection nodes". This option displays a menu of cluster nodes that are currently activated with the cluster statistics collection and presentation feature. This option is available for your information. This option always displays the current node first, in case you forget which node you are using.

   Each line of the display lists the node name, the date when the database was opened on that node, and whether cluster statistics collection is "active" or "inactive". A cluster node becomes inactive when a network transmission error occurs (see #6 below).

5. "Display available cluster nodes". This option displays an alphabetical list of all valid cluster nodes in your cluster. This option is available for your information.

6. "Reconnect to inactive cluster nodes". This option is displayed when there are one or more inactive cluster nodes in your cluster statistics collection (see #4 above). This option is used to manually attempt to re-connect to the cluster node; if unsuccessful, the cluster node is automatically removed from cluster statistics collection.

Note

Adding a new node or deleting an existing node from the cluster statistics collection automatically resets all of the statistics information so that you have a consistent point-in-time view of the statistics information. If desired, you may un-reset the statistics using the "Unreset" on-screen menu option.

The following example shows a sample "Active User Stall Messages" screen:

```
Node: Cluster (2)    Oracle Rdb X7.0-00 Performance Monitor 16-DEC-1996 13:03:06
Rate: 1.00 Second          Active User Stall Messages       Elapsed: 00:03:32.35
Page: 1 of 1    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
--------------------------------------------------------------------------------
Process.ID Elapsed.... T Stall.reason.............................. Lock.ID.
20600CB8:1u              reading pages 1:667 to 1:667
20600CB5:1*00:00:16.55 W waiting for record 51:60:2 (EX)
20600CB6:1*
232068CC:1u              reading ROOT file (RTUPB VBN 98)
20608403:1R              binding to database


--------------------------------------------------------------------------------
Config Exit Help LockID Menu >next_page <prev_page Set_rate Write Zoom !
```

Note that the "Node:" section of the header region indicates "cluster" and identifies the total number of nodes from which information is being collected, *including* the current node. On screens for which remote information is **not** collected, the node name indicates only the current node name.

On the "Stall Messages" and "Active User Stall Messages" screen, the processes from cluster nodes are highlighted to permit easy identification. Also, cluster application processes are suffixed with an asterisk ("*") for easy identification when the screen is written to disk.

The following example shows a sample "Summary IO Statistics" screen:

```
Node: Cluster (2)    Oracle Rdb X7.0-00 Performance Monitor 17-DEC-1996 06:49:23
Rate: 1.00 Second           Summary IO Statistics          Elapsed: 00:02:20.28
Page: 1 of 1    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
-------------------------------------------------------------------------------
statistic.........      rate.per.second.............. total....... average......
name..............      max..... cur..... avg....... count....... per.trans....
transactions                    0        0        0.6          98           1.0
verb successes                  0        0        9.9        1397          14.2
verb failures                   0        0        0.7         100           1.0

synch data reads                0        0       12.4        1751          17.8
synch data writes               0        0        4.7         669           6.8
asynch data reads               0        0        3.9         549           5.6
asynch data writes              0        0        0.3          44           0.4
RUJ file reads                  0        0        0.0           2           0.0
RUJ file writes                 0        0        0.0          10           0.1
AIJ file reads                  0        0        0.5          74           0.7
AIJ file writes                 0        0        0.7         111           1.1
ACE file reads                  0        0        0.0           0           0.0
ACE file writes                 0        0        0.0           0           0.0
root file reads                 0        0        1.6         227           2.3
root file writes                0        0        0.6          95           0.9
-------------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset Write X_plot
```

Note that, other than the "cluster" designation in the screen header region, ordinary numeric statistics screens appear exactly as they normally should.

While the "main" set of numeric statistics are always collected, such as the "summary" statistics screens, the remote collection of "by-cache", "by-area" and "by-logical-area" statistics is based upon the current screen being displayed. For example, the per-process stall information is only collected when one of the stall messages screen is being displayed. In addition, not all screens have remotely collected information; you will be notified when you migrate to a screen that does not support remote statistics collection. However, the majority of statistics screens do support remote statistic collection.

Note

The "per-screen" statistics collection feature is designed to minimize network utilization and improve overall system performance. However, this may affect the writing of "by-area" data to the binary output file when the **/OUTPUT** qualifier is specified. For example, because the by-area cluster statistics are only collected when a by-area screen is being displayed, by-area information written to the binary output file may be stale *if* the current screen is not a by-area statistics screen.

The following caveats apply to the "Cluster Statistics Collection and Presentation" feature:

- Up to 95 cluster nodes can be specified. However, it is recommended that cluater statistics collection be used prudently, as the system overhead in collecting the remote statistics may be substantial depending on the amount of information being transmitted on the network.
- Cluster statistics are collected at the specified display refresh rate. Therefore, it is recommended that the display refresh rate should be set to a reasonable rate based on the number of cluster nodes being collected. The default refresh rate of three seconds is reasonable for most remote collection loads.

- If the **/CLUSTER** qualifier is specified, the list of cluster nodes will apply to any database accessed during the SHOW STATISTIC session. When accessing additional databases using the "Switch Database" option, the same cluster nodes will be automatically accessed. However, any nodes manually added using the "Cluster Statistics" menu will *not* automatically be added to the new database's remote collection.

  In other words, manually adding and deleting cluster nodes only affects the "current" database, and does not apply to any other database that may have been accessed during the session. For example, when running the SHOW STATISTIC utility on node **ALPHA3** with manually added node **BONZAI**, subsequently switching to **BONZAI** as the "current" node will *not* display cluster statistics from node **ALPHA3** unless you manually add that node. Furthermore, switching back to node **ALPHA3** as the "current" node loses the previous collection of node **BONZAI** because it was manually added.

- Both **DECnet** and **TCP/IP** network protocols are supported. By default, the **DECnet** protocol is used. To explicitly specify which network protocol to use, define the **RDM$BIND_STT_NETWORK_TRANSPORT** to **DECNET** or **TCPIP** respectively. The **RDM$BIND_STT_NETWORK_TRANSPORT** logical must be defined to the *same* definition on both the local and cluster nodes. The **RDM$BIND_STT_NETWORK_TRANSPORT** logical can be specified in **LNM$FILE_DEV** on the local node, but must be specified in the **LNM$SYSTEM_TABLE** on all remote nodes.

  Note

  There is no command qualifier to specify the network protocol.

- The **/OUTPUT** qualifier continues to work as normal, but when in "cluster" mode writes the cluster statistics information to the binary output file.
- The **/CLUSTER** qualifier cannot be specified when using the **/INPUT** qualifier. Furthermore, the online selection of cluster nodes is not available when using the **/INPUT** qualifier.
- While the collection and presentation feature is active, all on-screen menu options continue to operate as normal. This include the time-plot, scatter-plot, screen pause and various other options.
- There is no way to exclude the current node from statistics collection. Log into another node if you want to accomplish this.
- The cluster collection of per-process stall information automatically detects the binding or unbinding of processes to cluster databases. There is no need to manually refresh the database information on the current node.
- If the database is not currently "open" on the specified node, cluster statistics collection will still be attempted. However, the remote database must still be opened normally prior to having regular process attaches.
- Attempting to add more than 95 cluster nodes will result in an error.
- When displaying any of the per-process screens that support cluster statistics collection, such as the "Stall Messages" screen, "zooming" on any of the displayed processes will display which node that process is using.

- The Cluster-wide “Statistics Collection and Presentation” feature is currently only available on OpenVMS platforms.

The “Cluster Statistics Collection and Presentation” feature is a separately installed product option. The separate installation is required to correctly setup the **DECnet** and **TCP/IP** protocols and login accounts. Attempting to use the cluster statistics collection when the feature has not been installed will result an error message being displayed.

Using the "Cluster Statistics" sub-menu from the tools menu, it is also possible to collect statistics from all "open" database nodes using the "Collect from open database nodes" option. This option simplifies the DBA's job of "remembering" where the database is currently open. However, subsequently opened nodes will NOT be automatically added to the collection; these would still have to be manually added.

The cluster statistics collection is an intra-cluster feature in that it only works on the SAME database, using the same device/directory specification originally used to run the initial RMU/SHOW STATISTICS (i.e. on a shared disk). The cluster statistics collection does NOT work across clusters (inter-cluster).

When replaying a binary output file, the screen header region accurately reflects the number of cluster nodes whose statistics are represented in the output file.

5.1.23 New Annotations on Query Outlines

As described in Section 2.1.73.1, Change in Query Ordering for Multistatement Procedures , the order of the queries in an outline match the order of optimization, not the order of execution. To make the reading of query outlines a little easier comments have been embedded in the outline to indicate the type of statement which generated the query component. For example,

```
SQL> begin
cont> declare :x integer;
cont>
cont> -- Assignment
cont> set :x = (select count(*) from TOUT_1);
cont>
cont> -- Delete statement
cont> delete from TOUT_1;
cont>
cont> -- Update statement
cont> update TOUT_1
cont>     set a = (select avg(a) from TOUT_2)
cont>     where a is null;
cont>
cont> -- Singleton Select
cont> select a into :x
cont>     from TOUT_1
cont>     where a = 1;
cont>
cont> -- Trace (nothing if TRACE is disabled)
cont> trace 'The first value: ', (select a from TOUT_1 limit to 1 row);
cont> end;
```

The query outline generated by the Rdb now appears with comments after the QUERY keyword in the outline. Currently, only the COMMENT IS clause is stored with the query outline by SQL.

```
-- Rdb Generated Outline : 29-MAY-1997 23:17
create outline QO_C11395E6020C6FFA_00000000
id 'C11395E6020C6FFA5A183A6CCE7C1F33'
mode 0
as (
  query (
-- Set
    subquery (
      TOUT_1 0        access path sequential
      )
    )
  query (
-- Delete
    subquery (
      TOUT_1 0        access path sequential
      )
    )
  query (
-- Update
    subquery (
      subquery (
        TOUT_2 1        access path sequential
        )
        join by cross to
      subquery (
        TOUT_1 0        access path sequential
        )
      )
    )
  query (
-- Select
    subquery (
      TOUT_1 0        access path sequential
      )
    )
  query (
-- Trace
    subquery (
      TOUT_1 0        access path sequential
      )
    )
  )
compliance optional     ;
```

It should be noted that the query outline generated by Rdb may not have a query corresponding to each statement within the procedure.

1. Not all statements require the query optimizer. For example, TRACE and SET statements which don't reference tables.
2. Subqueries in IF and CASE statements may be lifted into a previous statement by the optimizer to reduce the overheads associated with that query.
3. Subqueries within an INSERT statement are executed as though there were SET statements performed prior to the INSERT.
4. INSERT statements are not subject to query optimization.

5.1.24 New MODE Keyword Added to SET FLAGS Statement

The keyword MODE can be used to set the query outline mode from with in interactive and dynamic SQL session.

```
SQL> set flags 'mode(10),outline';
SQL> show flags

Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   PREFIX,OUTLINE,MODE(10)
SQL> select count(*) from employees;
-- Rdb Generated Outline : 30-MAY-1997 16:35
create outline QO_B3F54F772CC05435_0000000A
id 'B3F54F772CC054350B2B454D95537995'
mode 10
as (
  query (
-- For loop
    subquery (
      subquery (
        EMPLOYEES 0     access path index       EMP_EMPLOYEE_ID
        )
      )
    )
  )
compliance optional     ;

         100
1 row selected
```

The following options are accepted:

- NOMODE—this is the same as MODE(0) and disables the display of the mode in SHOW FLAGS.

  A mode of zero is a valid mode setting and is the default for Rdb generated query outlines.

- MODE—this is the same as MODE(1)

- MODE(n)—where n can be any numeric value (positive or negative).

  In the previous example the mode was set to 10 when generating the query outline. If the generated outline is added to the database it will only be used when the mode is set to 10, either by the SET FLAGS statement or by using the logical name RDMS$BIND_OUTLINE_MODE.

5.1.25 RMU/SHOW STATISTIC Screen Header Region Changes

With the introduction of cluster-wide statistics collection for a database, having the ability to know when cluster-wide statistics are being collected if very important. Equally important, however, is knowing whether statistics are being collected on all available cluster nodes on which the database is being accessed.

The following example shows the original **RMU/SHOW STATISTIC** utility screen, with the standard three-line header region.

```
Node: ALPHA3         Oracle Rdb X7.0-00 Performance Monit27-MAR-1997 07:25:57.96
Rate: 1.00 Second            Summary IO Statistics          Elapsed: 00:00:03.27
Page: 1 of 1    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
------------------------------------------------------------------------------

statistic.........      rate.per.second.............. total....... average......
name..............      max..... cur..... avg....... count....... per.trans....
transactions                   0        0        0.0            0           0.0
verb successes                 0        0        0.0            0           0.0
verb failures                  0        0        0.0            0           0.0
```

```
synch data reads              0       0       0.9           3          0.0
synch data writes             0       0       0.0           0          0.0
asynch data reads             0       0       0.0           0          0.0
asynch data writes            0       0       0.0           0          0.0
RUJ file reads                0       0       0.0           0          0.0
RUJ file writes               0       0       0.0           0          0.0
AIJ file reads                0       0       0.0           0          0.0
AIJ file writes               0       0       0.0           0          0.0
ACE file reads                0       0       0.0           0          0.0
ACE file writes               0       0       0.0           0          0.0
root file reads               5       5       5.8          19          0.0
root file writes              0       0       0.0           0          0.0

------------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Write X_plot Yank !
```

It is difficult to know on how many nodes the database is current active, as well as the maximum number of nodes on which the database *may* be active.

This problem has been corrected in Oracle Rdb Version 7.0.1. The **RMU/SHOW STATISTIC** screen header region has been enhanced to display the current node suffixed with a "(x/y/z)" identifier.

The value "x" indicates the number of nodes statistics are being actively collected on. The value "y" indicates the number of nodes on which the database is being accessed. The value "z" indicates the maximum number of nodes on which the database may be accessed.

For example, the indicator "(1/3/5)" specifies that statistics are being collected from 1 node, but that the database is currently open on 3 nodes, which a maximum open node count of 5.

The following example shows the above information.

```
Node: ALPHA3 (1/3/5)    Oracle Rdb X7.0-00 Perf. Monitor 27-MAR-1997 07:36:56.08
Rate: 1.00 Second           Summary IO Statistics         Elapsed: 00:00:02.20
Page: 1 of 1    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
------------------------------------------------------------------------------

statistic.........      rate.per.second.............. total....... average......
name..............      max..... cur..... avg....... count....... per.trans....
transactions                  0       0       0.0           0          0.0
verb successes                0       0       0.0           0          0.0
verb failures                 0       0       0.0           0          0.0

synch data reads              1       1       1.3           3          0.0
synch data writes             0       0       0.0           0          0.0
asynch data reads             0       0       0.0           0          0.0
asynch data writes            0       0       0.0           0          0.0
RUJ file reads                0       0       0.0           0          0.0
RUJ file writes               0       0       0.0           0          0.0
AIJ file reads                0       0       0.0           0          0.0
AIJ file writes               0       0       0.0           0          0.0
ACE file reads                0       0       0.0           0          0.0
ACE file writes               0       0       0.0           0          0.0
root file reads               9       9       9.0          20          0.0
root file writes              0       0       0.0           0          0.0

------------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Write X_plot Yank !
```

When statistics are being collected from more than 1 node, the "Node:" tag is changed to "Cluster:" to designate that cluster-wide statistics collection is active. For example, the following screen shows that statistics are being collected from 3 nodes:

```
Cluster: ALPHA3 (3/3/5) Oracle Rdb X7.0-00 Perf. Monitor 27-MAR-1997 07:37:13.33
Rate: 1.00 Second           Summary IO Statistics          Elapsed: 00:00:02.86
Page: 1 of 1    KODH$:[R_ANDERSON.WORK.STATS]MF_PERSONNEL.RDB;1     Mode: Online
--------------------------------------------------------------------------------

statistic.........      rate.per.second.............. total....... average......
name..............      max..... cur..... avg....... count....... per.trans....
transactions                  0        0        0.0            0           0.0
verb successes                0        0        0.0            0           0.0
verb failures                 0        0        0.0            0           0.0

synch data reads              0        0        0.0            0           0.0
synch data writes             0        0        0.0            0           0.0
asynch data reads             0        0        0.0            0           0.0
asynch data writes            0        0        0.0            0           0.0
RUJ file reads                0        0        0.0            0           0.0
RUJ file writes               0        0        0.0            0           0.0
AIJ file reads                0        0        0.0            0           0.0
AIJ file writes               0        0        0.0            0           0.0
ACE file reads                0        0        0.0            0           0.0
ACE file writes               0        0        0.0            0           0.0
root file reads              10        0        1.0            3           0.0
root file writes              0        0        0.0            0           0.0

--------------------------------------------------------------------------------
Exit Graph Help Menu Options Pause Reset Set_rate Time_plot Unreset Write X_plot
```

5.1.26 New WARN_INVALID debugging flag

Routines and query outlines can become invalid due to the following statements:

- When a table is dropped using the CASCADE option any procedure or function which references that table is marked invalid.
- When a table is dropped (using either the CASCADE or RESTRICT options) any query outline which references that table is marked as invalid.
- When a module is dropped using the CASCADE option any procedure, function or query outline which references that module is marked invalid. A query outline references a module when it uses a temporary table declared at the module level.
- When an index is dropped, or altered to have MAINTENANCE IS DISABLED any query outline which references that index is marked as invalid.

In this release of Oracle Rdb a new debugging flag has been added to report invalidated objects during the ALTER INDEX, DROP INDEX, DROP TABLE and DROP MODULE statements. Either the RDMS$DEBUG_FLAGS "Xw" (upper case X and lower case w) can be defined, or the SQL SET FLAGS statement can specify the keyword WARN_INVALID to enable this behavior. This is shown in the following example where lines prefixed with "~Xw" are displayed.

```
SQL> set flags 'warn_invalid';
SQL> show flags;
```

```
Alias RDB$DBHANDLE:
Flags currently set for Oracle Rdb:
   PREFIX,WARN_INVALID
SQL> -- warning because of dependencies
SQL> drop table T1 cascade;
~Xw: Routine "P3" marked invalid
~Xw: Routine "P2" marked invalid
~Xw: Routine "P1" marked invalid
SQL>
SQL> -- Create an outline with INDEX and
SQL> create table T1 (a integer, b integer);
SQL> create index I1 on T1 (a);
SQL> create outline QO1
cont> id '19412AB61A7FE1FA6053F43F8F01EE6D'
cont> mode 0
cont> as (
cont>   query (
cont>     subquery (
cont>       T1 0    access path index       I1
cont>       )
cont>     )
cont>   )
cont> compliance optional     ;
SQL>
SQL> -- warning because of disabled index
SQL> alter index I1
cont>     maintenance is disabled;
~Xw: Outline "QO1" marked invalid (index "I1" disabled)
SQL> show outline QO1;
     QO1
    Object has been marked INVALID
 Source:
create outline QO1
id '19412AB61A7FE1FA6053F43F8F01EE6D'
mode 0
as (
  query (
    subquery (
      T1 0    access path index       I1
      )
    )
  )
compliance optional     ;
SQL>
```

5.1.27 CONSTRAINT Clause for ALTER TABLE is now Syntactically Compatible with Oracle7

In Rdb, the syntax for adding a constraint definition to a table includes the clause ADD CONSTRAINT CONSTRAINT. In Oracle7, the syntax includes the clause ADD CONSTRAINT. Specifically, in Oracle7, CONSTRAINT is specified only once.

For compatibility, SQL now allows both ADD CONSTRAINT CONSTRAINT and ADD CONSTRAINT as valid syntax for Oracle Rdb. Additionally, this conforms with SQL92.

The following example demonstrates that SQL allows ADD CONSTRAINT <constraint-name> as valid syntax.

```
!The following 2 examples show correct syntax.
ALTER TABLE TABLA ADD CONSTRAINT CONSTRAINT CONSTR PRIMARY KEY (COL)
ALTER TABLE TABLA ADD CONSTRAINT CONSTR PRIMARY KEY (COL)
```

5.1.28 Process Image Name Written to Monitor Log on OpenVMS

All OpenVMS platforms.

For local (non-remote) database attach requests, the process's image name (the main image being run by the process) is now logged to the Oracle Rdb monitor log file.

The following is an example of the image name display:

```
5-JUN-1997 12:43:03.49 - Received user attach request from 2020412F:1
  - process name A_TOLLIVER_1, user TOLLIVER
  - image name "DKA0:[SYS0.SYSEXE]SQL$.EXE;1"
  - for database "_$1$DKA0:[DB]DB.RDB;1" [_$1$DKA0] (738,510,0)
  - database global section name is "RDM70T_B4I0ATO0"
  - database global section size is 99 pages (512 bytes per page)
  - database dashboard installed
  - cluster watcher is active
  - sending normal user attach reply to 2020412F:1
```

If the image name is not available for remote attaches or due to unexpected error conditions, a message of " image name not available " is written to the log file in place of the process's image name.