Oracle® CODASYL DBMS for OpenVMS

Release Notes

Release 7.2 for OpenVMS Alpha and
HP OpenVMS Industry Standard 64 for Integrity Servers

January 2006

ORACLE®

Oracle CODASYL DBMS Release Notes, Release 7.2 for OpenVMS Alpha and OpenVMS I64

Beta Draft Documentation

Contents

4 Known Problems and Restrictions

Tables

Preface

Purpose of This Manual

This manual contains release notes for Oracle CODASYL DBMS release 7.2. The notes describe changed and enhanced features, upgrade and compatibility information, new and existing software problems and restrictions, and software and documentation corrections.

Intended Audience

This manual is intended for use by all Oracle CODASYL DBMS users. Read this manual before you install, upgrade, or use Oracle CODASYL DBMS release 7.2.

Document Structure

This manual consists of the following chapters:

Chapter 1	Describes how to install Oracle CODASYL DBMS release 7.2.
Chapter 2	Describes new and changed features in Oracle CODASYL DBMS release 7.2.
Chapter 3	Describes problems fixed in Oracle CODASYL DBMS release 7.2.
Chapter 4	Describes problems, restrictions, and workarounds known to exist in Oracle CODASYL DBMS release 7.2.

Conventions

Oracle CODASYL DBMS is often referred to as DBMS in this manual..

HP OpenVMS Industry Standard 64 for Integrity Servers is often referred to as OpenVMS I64.

OpenVMS refers to both OpenVMS Alpha and OpenVMS I64.

1
Installing Oracle CODASYL DBMS

All Oracle CODASYL DBMS release 7.2 kits are full kits. There is no requirement to install any prior release of Oracle CODASYL DBMS prior to installing this release.

1.1 Oracle CODASYL DBMS on OpenVMS I64

In addition to the HP OpenVMS Alpha platform, Oracle CODASYL DBMS is now available on the HP OpenVMS Industry Standard 64 for Integrity Servers platform. In general, the Oracle CODASYL DBMS functionality is comparable between the two platforms.

This release provides a full set of Oracle CODASYL DBMS functionality for both platforms, including local and remote database access, as well as native DML and DDL operations. This means that users running on OpenVMS I64 can create Oracle CODASYL DBMS databases, compile, link, and run their database applications natively.

Because the Oracle CODASYL DBMS database format is the same across all supported platforms, you can, for example, back up an Oracle CODASYL DBMS database on an Alpha system, then restore it on an I64 system (the reverse is also true). If necessary, implicit forward conversions are performed to bring the database version to the currently installed level.

With remote access, you can bind to an Oracle CODASYL DBMS database on an Alpha system from an I64 system, or vice versa, as long as the appropriate Oracle CODASYL DBMS software is available on both platforms.

Additionally, if your environment consists of Alpha and I64 systems in a mixed cluster environment, you can access an Oracle CODASYL DBMS V7.2 database from either system, or both systems concurrently.

1.2 Database Format Changed

Starting with field test version T7.2-05 of Oracle CODASYL DBMS, the on-disk database format has been incremented to version 721. You must execute a DBO/CONVERT command on those databases created or accessed by prior DBMS 7.2-05 field test kits, as well as DBMS release 7.0 or 7.1.

Prior to upgrading to Oracle CODASYL DBMS release 7.2, and prior to converting an existing database to the 7.2 format, Oracle strongly recommends that you perform a full database verification (with the DBO/VERIFY command) along with a a full database backup to ensure a valid and protected database copy.

1.3 Using Databases from Releases Earlier Than V7.0

Starting with this release, you cannot convert or restore databases from versions earlier than 7.0 directly. The DBO CONVERT command for Oracle CODASYL DBMS V7.2 supports conversions from V7.0 and V7.1 only.

If you have a V3.3 through V6.1 database, you must convert it to at least V7.0 and then convert it to V7.2 in two steps. For example, if you have a V4.2 database, install the latest update to DBMS 7.0, convert the database to that version, install DBMS 7.2 then convert the v7.0 database to V7.2.

If you attempt to convert or restore a database version prior V7.0 directly to V7.2, Oracle DBO generates an error.

1.4 Installation

Install this software update using the standard OpenVMS Install Utility.

1.4.1 Requirements

One of the following conditions must be met in order to install this software:

- OpenVMS Alpha version 8.2 or later
- OpenVMS I64 version 8.2-1 or later.

The installation requires approximately 110,000 blocks for OpenVMS Alpha systems.

The installation requires approximately 280,000 blocks for OpenVMS I64.

1.4.2 Pre-installation Steps

If there are any pre-existing versions of Oracle CODASYL DBMS release 7.2 on the target system, you must shut down that Oracle CODASYL DBMS monitor before beginning the installation. If you are installing the multiversion kit, no other Oracle CODASYL DBMS monitors need to be stopped for this installation. If you are installing the standard version of DBMS release 7.2, you must stop any standard version monitor.
To stop the Oracle CODASYL DBMS 7.2 monitor, issue the command:

```
$ @SYS$STARTUP:MONSTOP72.COM
```

To stop the Oracle CODAYSYL DBMS standard monitor, issue the command:

```
$ @SYS$STARTUP:MONSTOP.COM
```

In an OpenVMS cluster environment, you must execute either procedure on each cluster member where the monitor is started.

1.4.3 Invoking VMSINSTAL

The installation procedure for Oracle CODASYL DBMS is the same for OpenVMS Alpha or OpenVMS I64 systems. To begin the installation of Oracle CODASYL DBMS for OpenVMS Alpha, issue the command:

```
@SYS$UPDATE:VMSINSTAL DBM0720A072 device-name
```

To begin the installation of Oracle CODASYL for OpenVMS I64, issue the command:

```
@SYS$UPDATE:VMSINSTAL DBM0720I072 device-name
```

Device-name is the name of the device on which the media is mounted. If the device is a disk drive, you also need to specify a directory. For example:

```
DKA400:[DBM.KIT]
```

1.4.4 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 the VMSINSTAL procedure 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 CODASYL DBMS installed will probably not be usable.

1.4.5 Post-installation Steps

After installing Oracle CODASYL DBMS release 7.2, you must convert any existing database that you wish to access with this software.

To convert an existing Oracle CODASYL DBMS database to release 7.2, perform one of the following steps:

- issue the DBO/CONVERT[/NOCOMMIT] command to perform an explicit convert, for example:

```
$ DBO/CONVERT PARTS
```

- back up and restore the database to perform an implicit convert:

```
$ DBO/BACKUP/MULTITHREAD PARTS PARTS.DBF
$ @SYS$LIBRARY:DBMSETVER 7.2
$ DBO/RESTORE/MULTITHREAD PARTS.DBF
```

The backup and restore method is a good way to migrate an existing Oracle CODASYL DBMS database from an Alpha system to an I64 system.

1.4.6 VMS$MEM_RESIDENT_USER Rights Identifier Required

Oracle CODASYL DBMS release 7.1 introduced additional privilege enforcement for the database or row cache qualifiers MEMORY_MAPPING=SYSTEM and LARGE_MEMORY. If a database utilizes any of these features then the user account that opens the database must be granted the VMS$MEM_RESIDENT_USER rights identifier. Also, any process attempting to change these attributes, to convert, or restore a database with these attributes enabled must also hold the right.

Oracle recommends that the DBO/OPEN command be used when utilizing these features.

2

Enhancements Provided in ORACLE CODASYL DBMS Release 7.2

This chapter describes new and changed features in Oracle CODASYL DBMS release 7.2.

2.1 Default Floating Point Format

The Itanium architecture has a 64-bit model and basic system functions similar to the Alpha chip. However, there are some implementation differences between the two platforms that might affect user-written applications.

One of the differences is the availability of hardware-supported floating-point formats. The Itanium architecture implements floating-point arithmetic in hardware using the IEEE floating-point formats, including IEEE single and IEEE double. The Alpha architecture supports both IEEE and VAX floating-point formats in hardware, and OpenVMS compilers generate code using the VAX formats by default, with options (on Alpha) to use IEEE formats. Irrespective of whether it was originally written for VAX or Alpha, an OpenVMS application that uses the default VAX floating-point formats needs to produce equivalent behavior on the Itanium architecture using IEEE formats at the lowest level.

- On OpenVMS VAX and OpenVMS Alpha, VAX float is the default. VAX format data is assumed and VAX floating instructions are used.
- On OpenVMS Alpha, you can specify the compiler option /FLOAT=IEEE. In this case, IEEE format data is assumed and IEEE floating instructions are used.
- On OpenVMS I64, IEEE float is the default. IEEE format data is assumed and IEEE floating instructions are used.
- On OpenVMS I64, you can specify the compiler option /FLOAT=D_FLOAT or /FLOAT=G_FLOAT.

When you compile an OpenVMS application that specifies an option to use VAX floating-point on the Itanium architecture, the compiler automatically generates code for converting floating-point formats. Whenever the application performs a sequence of arithmetic operations, this code does the following:

1. Converts VAX floating-point formats to either IEEE single or IEEE double floating-point formats.
2. Performs arithmetic operations in IEEE floating-point arithmetic.
3. Converts the resulting data from IEEE formats back to VAX formats.

Note that where no arithmetic operations are performed (VAX float fetches followed by stores), conversions will not occur. The code handles such situations as moves. VAX floating-point formats have the same number of bits and precision as their equivalent IEEE floating-point formats. For most applications, the conversion process will be transparent. In a few cases, arithmetic calculations might have different results because of the following differences between VAX and IEEE formats:

- Values of numbers represented
- Rounding rules
- Exception behavior

For more information, Oracle recommends reviewing the white paper "OpenVMS floating-point arithmetic on the Intel Itanium architecture" available from HP.

2.2 HP Compilers on OpenVMS I64

The following compilers on OpenVMS I64 have been used with this release. It is expected that later compiler versions (and perhaps earlier ones) will continue to function correctly. Please contact HP for additional information.

- HP C V7.1-012
- HP COBOL V2.8-1414
- HP Fortran V8.0-48071
- I64 BASIC V1.6-000
- HP Pascal I64 V5.9-98-50F9M

2.3 DBO/SHOW LOCKS Includes Time and Node Name

Bug 4761828

The output of the DBO/SHOW LOCKS command has been enhanced to include the current date and time and the system node name in the header line as shown in the following example:

```
$ DBO /SHOW LOCKS
================================================================
  SHOW LOCKS Information at 26-NOV-2005 09:29:01.21 on node RDBI64
================================================================

----------------------------------------------------------------
Resource Name: AIJ journal control
Granted Lock Count: 7,  Parent Lock ID: 180007FA,   Lock Access Mode:
Executive, Resource Type: Global, Lock Value Block: 00000013 00000000
00000000 00000000
     .
     .
     .
```

2.4 Database Server Process Priority Clarification

By default, the database servers (ABS, ALS, DBR, LCS, LRS, RCS) created by the DBMS monitor inherit their OpenVMS process scheduling base priority from the DBMS monitor process. The default priority for the DBMS monitor process is 15.

Individual server priorities can be explicitly controlled via system-wide logical names as described in Table 2–1.

Table 2–1 Server Process Priority Logical Names

Logical Name	Use
DBM$BIND_ABS_PRIORITY	Base Priority for the ABS Server process
DBM$BIND_ALS_PRIORITY	Base Priority for the ALS Server process
DBM$BIND_DBR_PRIORITY	Base Priority for the DBR Server process
DBM$BIND_LCS_PRIORITY	Base Priority for the LCS Server process
DBM$BIND_LRS_PRIORITY	Base Priority for the LRS Server process
DBM$BIND_RCS_PRIORITY	Base Priority for the RCS Server process

When the Hot Standby feature is installed, the DBMAIJSERVER account is created specifying an account priority of 15. The priority of AIJ server processes on your system can be restricted with the system-wide logical name DBM$BIND_AIJSRV_PRIORITY. If this logical name is defined to a value less than 15, an AIJ server process will adjust its base priority to the value specified when the AIJ server process starts. Values from 0 to 31 are allowed for DBM$BIND_AIJSRV_PRIORITY, but the process is not able to raise its priority above the DBMAIJSERVER account value.

For most applications and systems, Oracle discourages changing the server process priorities.

2.5 DBM$BIND_MAX_DBR_COUNT Documentation Clarification

The following is an updated description for the DBM$BIND_ MAX_DBR_COUNT logical.

When an entire database is abnormally shut down (for example, due to a system failure), the database must be recovered in a node failure recovery mode. This recovery is performed by another monitor in the cluster if the database is opened on another node or is performed the next time the database is opened.

The DBM$BIND_MAX_DBR_COUNT logical name and the DBB_BIND_MAX_DBR_COUNT configuration parameter define the maximum number of database recovery (DBR) processes to be simultaneously invoked by the database monitor for each database during a node failure recovery. This logical name and configuration parameter apply only to databases that do not have global buffers enabled. Databases that utilize global buffers have only one recovery process started at a time during a node failure recovery.

In a node failure recovery situation with the Row Cache feature enabled (regardless of the global buffer state), the database monitor starts a single database recovery (DBR) process to recover the Row Cache Server (RCS) process and all user processes from the oldest active checkpoint in the database.

Per-Database Value

The DBM$BIND_MAX_DBR_COUNT logical name specifies the maximum number of database recovery processes to run at once for each database. For example, if there are 10 databases being recovered and the value for the DBM$BIND_MAX_DBR_COUNT logical name is 8, up to 80 database recovery processes would be started by the monitor after a node failure.

The DBM$BIND_MAX_DBR_COUNT logical name is translated when the monitor process opens a database. Databases must be closed and reopened for a new value of the logical to become effective.

2.6 Maximum Page and Buffer Size Increases

In previous releases, the maximum allowed database buffer size was 64 blocks and the maximum allowed database page size was 32 blocks. These limits have been increased. The current maximum allowed database buffer size is 128 blocks and the maximum allowed database page size is 63 blocks.

Be aware that using larger database buffer sizes will require additional virtual memory.

2.7 No File-System Caching When Writing Database and AIJ Backup Files

It is expected that the disk-based output file from a database or after-image journal backup operation may be relatively large, sequentially accessed, and not read in the near future. In order to avoid polluting the file system cache and to streamline file write operations, caching by the operating system is now explicitly disabled when writing these files. There is no effect on caches implemented in storage devices or controllers.

2.8 Performance: Improved Rollback Performance

This release of Oracle CODASYL DBMS introduces optimizations for rolling back transactions. These improvements affect the performance of ROLLBACK statements issued by an application and also the database recovery (DBR) process. A summary of the most significant changes are listed below:

- When reading the recovery-unit (RUJ) file, I/O operations are now done using 256-block buffers instead of reading one block at a time as was done in previous versions.
- Multiple buffers are now used to read the journal. While the contents of one buffer are being processed, data is being read into the next buffer asynchronously.
- When writing to the journal, RUJ data is copied directly into the RUJ I/O buffer from the storage area data page instead of being copied into an intermediate buffer and then to the RUJ buffer.
- When reading from the journal, journal entries are processed directly from the RUJ I/O buffer instead of being copied to an intermediate buffer first.

- When rolling back a transaction, the content of the RUJ buffer is scanned to determine what data pages will be rolled back, and I/Os are started to those pages immediately. That is, asynchronous prefetches (APF) are issued for pages that will be rolled back. As journal entries are processed, new prefetches are started for subsequent journal entries as soon as buffers are available. This significantly reduces the time spent waiting for I/O completion.
- In previous releases, if a process failed and a DBR was started to recover the user, the DBR would scan the journal to locate the last entry in the journal. For large transactions, the scanning operation could take a considerable amount of time. In this release, the location of the last journal entry is maintained in shared memory. Now, when a DBR process is started it can immediately locate the last entry in the journal without having to scan the journal.

2.9 64-bit Statistics

In prior versions of Oracle CODASYL DBMS, statisics counters were maintained in 32-bit longword integers. This limited counters to a maximum value of 4,294,967,294. This limit could be exceeded and would cause counters to wrap back to zero.

Oracle CODASYL DBMS statistics counters have now been promoted to 64-bit quadword integers. This change affects the binary statistics output file format as well. Longword statitsics counters have been promoted to quadwords.

Most field displays within the DBO/SHOW STATISTICS utility have not been widened and may overflow if the internal counter value exceeds the decimal display width.

2.10 Maximum Global Buffer Count Increased

Prior versions of Oracle CODASYL DBMS limited the total number of global buffers per database to 524,288. This limit has been relaxed. The maximum global buffer count allowed for Oracle CODASYL DBMS Release 7.2 is 1,048,576.

2.11 MACRO-32 Compiler for OpenVMS I64

For OpenVMS I64 only, you must use a MACRO-32 Compiler for OpenVMS I64 to compile any DBMS application compiled through the DML interface.

When compiling a host language DBMS module, the DML command automatically generates and compiles VAX MACRO code and appends the object module to object module of the host language.

On OpenVMS I64, you can specify the DML /NODELETE qualifier to review macro code generated for a module.

2.12 DBO Operator Notification Syntax Change

The DBO syntax to enable or disable system notification for certain database events was changed in DBMS release 7.1.0. Due to an omission, this new syntax was never documented.

In versions of DBMS prior to 7.1.0, the DBO operator notification facility was used to provide notification of after-image journal changes or problems that may occur during normal database activity. (Refer to the *Oracle CODASYL DBMS Database Administration Reference Manual* for more information). The syntax for the DBO/CREATE and DBO/MODIFY commands reflected this association between notifications and journaling:

Old Syntax:

```
$ DBO/CREATE/JOURNAL_OPTIONS=([NO]NOTIFY=(operator-name) db-name
$ DBO/MODIFY/JOURNAL_OPTIONS=([NO]NOTIFY=(operator-name) db-name
```

where operator-name was a list of one or more standard OpenVMS operator classes:

- CENTRAL
- CLUSTER
- CONSOLE
- DISKS
- OPER1
- OPER2
- OPER3
- OPER4
- OPER5
- OPER6
- OPER7
- OPER8
- OPER9
- OPER10
- OPER11
- OPER12
- SECURITY

Starting with DBMS 7.1.0, operator notification has been expanded to include non-journal related events (refer to the DBMS release 7.1-0 Release Notes for more details), including:

- Bugcheck notification
- Corrupt Page Table Additions
- Storage Area Extensions
- AIJ Fullness
- Server startup and termination messages

The old syntax is now obsolete. The syntax for enabling or disabling system notification has been changed to correspond to this new database-wide behavior.

New Syntax:

```
$ DBO/CREATE/ALERT_OPERATOR=(ENABLED=operator-name)  db-name
$ DBO/CREATE/ALERT_OPERATOR=(DISABLED=operator-name) db-name

$ DBO/MOD/ALERT_OPERATOR=(ENABLED=operator-name)  db-name
$ DBO/MOD/ALERT_OPERATOR=(DISABLED=operator-name) db-name
```

The ENABLED and DISABLED keywords can be combined within the same DBO command. The list of valid operator-names remains unchanged.

2.13 Support for ACE (AIJ Cache on Electronic disk) Removed

Prior versions of Oracle CODASYL DBMS provided support for a file called an AIJ cache on an electronic disk (also known as ACE) to use as a temporary cache for AIJ write operations. At one point in time, these devices provided a performance benefit for some classes of applications that heavily used the after-image journal.

With changes in technologies (in particular, improved I/O interfaces and various write-back caching schemes), the benefits of the ACE feature have declined to the point where it is no longer an effective performance advantage. This support has been removed in Oracle CODASYL DBMS release 7.2.

The database attribute JOURNAL_OPT=(CACHE...) is now ignored by the DBO/CREATE and DBO/MODIFY commands.

2.14 Logical DBM$BIND_RW_TX_CHECKPOINT_ADVANCE Removed

BUG 1584167

Prior to DBMS release 7.1.2, if the logical DBM$BIND_RW_TX_CHECKPOINT_ADVANCE was not defined to be 1, read/write transactions that did not make any database modifications would not advance their fast commit checkpoint location. In release 7.1.2, in response to BUG 2439694, the checkpointing code was restructured such that checkpoints may advance at the end of any transaction, whether or not the transactions made any database modifications. That change made the logical DBM$BIND_RW_TX_CHECKPOINT_ADVANCE no longer necessary. It has been removed.

2.15 TRANSPORT Added to DBO/REPLICATE AFTER

BUG 4109344

The TRANSPORT qualifier has been added to the DBO/REPLICATE AFTER START and CONFIGURE commands. This new qualifier allows the network transport to be specified. The valid values are DECNET and TCPIP. The specified network transport is saved in the database. For example:

```
$ DBO/REPLICATE AFTER CONFIGURE -
  /TRANSPORT=TCPIP /STANDBY=REMNOD::DEV:[DIR]STANDBY_DB M_TESTDB
```

In previous releases, you had to define the system-wide logical DBO$BIND_HOT_NETWORK_TRANSPORT in order to use TCP/IP as the network transport for Hot Standby.

2.16 DBO/BACKUP/MULTITHREAD Support For /DENSITY = SDLT320

The DBO/BACKUP/MULTITHREAD/DENSITY command now supports the SDLT320 keyword for use with SuperDLT320 tape drives.

2.17 DBO SHOW LOCKS /RESOURCE_TYPE Qualifier

Previously, the DBO /SHOW LOCKS command would display all lock resource types. This could sometimes result in a significant amount of output and could make it cumbersome to locate locks for specific types of resources.

This situation has been improved with the /RESOURCE_TYPE=(restyp...) qualifier. When this qualifier is present on the command line, only the specific resource types will be displayed. This permits, for example, only PAGE or RECORD lock types to be selected. This functionality is intended primarily as a debugging tool. Knowledge of the lock types and functionality of Oracle CODASYL DBMS is assumed. Not all lock types will exist on all systems and versions of DBMS.

The following keywords are allowed with the /RESOURCE_TYPE qualifier.

Table 2–2 RESOURCE_TYPE keywords

Internal Lock Type Name	Keyword(s)
ACCESS	ACCESS
ACTIVE	ACTIVE
AIJDB	AIJDB
AIJFB	AIJFB
AIJHWM	AIJHWM, AIJ_HIGH_WATER_MARK
AIJLOGMSG	AIJ_LOG_MESSAGE
AIJLOGSHIP	AIJ_LOG_SHIPPING
AIJOPEN	AIJ_OPEN
AIJSWITCH	AIJ_SWITCH
AIJ	AIJ
ALS	ALS_ACTIVATION
BCKAIJ	AIJ_BACKUP, BCKAIJ
BCKAIJ_SPD	AIJ_BACKUP_SUSPEND
BUGCHK	BUGCHECK
CHAN	CHAN, FILE_CHANNEL
CLIENT	CLIENT
CLOSE	CLOSE
CLTSEQ	CLTSEQ
CPT	CORRUPT_PAGE_TABLE, CPT
DASHBOARD	DASHBOARD_NOTIFY
DBK_SCOPE	DBKEY_SCOPE

(continued on next page)

Table 2–2 (Cont.) RESOURCE_TYPE keywords

Internal Lock Type Name	Keyword(s)
DBR	DBR_SERIALIZATION
DB	DATABASE
FIB	FAST_INCREMENTAL_BACKUP, FIB
FILID	FILID
FRZ	FREEZE
GBL_CKPT	GLOBAL_CHECKPOINT
GBPT_SLOT	GLOBAL_BPT_SLOT
KROOT	KROOT
LOGFIL	LOGFIL
MEMBIT	MEMBIT
MONID	MONID, MONITOR_ID
MONITOR	MONITOR
NOWAIT	NOWAIT
PLN	DBKEY, RECORD, PLN
PNO	PAGE, PNO
QUIET	QUIET
RCACHE	RCACHE
RCSREQUEST	RCS_REQUEST
RCSWAITRQST	RCS_WAIT_REQUEST
REL_AREAS	RELEASE_AREAS
REL_GRIC_REQST	RELEASE_GRIC_REQUEST
ROOT_AREA	DUMMY_ROOT_AREA
RO_L1	L1_SNAP_TRUNCATION
RTUPB	RTUPB
RUJBLK	RUJBLK
RW_L2	L2_SNAP_TRUNCATION
SAC	SNAP_AREA_CURSOR
SEQBLK	SEQBLK
STAREA	STORAGE_AREA, PAREA
STATRQST	STATISTICS_REQUEST
TRM	TERMINATION
TSNBLK	TSNBLK
UTILITY	UTILITY

The RESOURCE_TYPE qualifier is incompatible with the MODE, LIMIT, LOCK and PROCESS qualifiers.

3
Problems Corrected

This chapter describes software errors corrected in Oracle CODASYL DBMS release 7.2.

3.1 DBO /SHOW STATISTICS Enhanced Navigation Between Row Caches

Bugs 4727723 and 3738511

Previously, when you used the DBO /SHOW STATISTICS "Row Cache Utilization", "Hot Row Information", "Row Cache Status", "Row Cache Queue Length", and "Row Length Distribution" displays, it was difficult to move between displays for multiple row caches.

This problem has been corrected. The end bracket (]) and begin bracket ([) keys can be used to scroll between row caches on these displays.

3.2 Bugcheck at KOD$UNBIND

Starting with Oracle CODASYL DBMS release 7.1.1.1, if a user attempted to UNBIND from an application or DBQ session without first terminating the transaction (COMMIT or ROLLBACK), a bugcheck would be generated and the process would be terminated.

This problem has now been fixed. Now, DBMS will raise a DBM-F-TRAN_IN_PROG exception, indicating that there is a transaction in progress, and allow the application to deal with the error.

3.3 DBO /SHOW LOCKS Limits Relaxed

Previously, the /LOCK= and /PROCESS= qualifiers of the DBO /SHOW LOCKS command were limited to 32 specified values.

This problem has been corrected. The /LOCK= and /PROCESS= qualifiers of the DBO /SHOW LOCKS command now accept up to 256 values each.

3.4 Support for Fortran-95 compiler

Until recently Oracle CODASYL DBMS only supported (and was supported by) HP's FORTRAN–77 compiler on OpenVMS Alpha. The DBMS FORTRAN precompiler did not function with either the later FORTRAN-90 (F90) or FORTRAN-95 (F95) compilers.

At first, when the F90 compiler started shipping, FORTRAN–77 was installed as the default and DBMS customers did not notice a problem. However, when the F95 compiler began shipping, it was installed as the default. As such, the FORTRAN command did not recognize the the /DML qualifier.

To get around this problem, you needed to specify the /OLD_F77 qualifer on the FORTRAN command line in order to compile DBMS FORTRAN applications (for example, FORTRAN/DML/OLD_F77...). The /OLD_F77 qualifier signified that you wanted to use the F77 compiler.

This problem has now been fixed. Starting with HP FORTRAN 8.0, DBMS 7.2 supports the F95 compiler and you no longer need to supply the /OLD_F77 qualifer.

The necessary changes to DBMS were also included in the DBMS V7.1-2 and V7.0-61 (and later) releases.

Note

NOTE: DBMS still only supports the fixed-form source model, not the free-form model available with the F90 compiler. Oracle will consider easing this restriction in a future release.

3.5 Restoring Non-Snapshot Database with Snap=Enabled

In Oracle CODASYL DBMS release 7.0 or 7.1, if you attempted to restore a non-snapshot database with the global qualifier SNAPSHOTS=ENABLED, you would receive the following warning:

```
$ DBO/RESTORE/SNAP=ENABLED PARTS
%DBO-W-NODBSNAPS, No snapshots allowed on database, /SNAP qualifiers ignored
```

However, internally, the database would be set to "Area does not have snapshots", but snapshots would be "enabled" (SNAPS_ALLOWED = 00 and SNAPS_ ENABLED = 01). This inconsistency caused a dbmbugchk at PIO$READY when trying a start a read/write transaction.

This problem only affected single-threaded restores and has been corrected in this release.

The only workaround would be to:

- restore the current database backup without the SNAPSHOT qualifier;
- back up the damaged database and restore explicitly disabling snapshots (DBO/RESTORE/SNAPSHOTS=NOENABLED).

3.6 Problems Mixing Stream and Non-Stream DML within the Same Image

BUG 4121994

In previous versions of Oracle CODASYL DBMS, problems could arise running applications comprised of stream and non-stream modules linked together into the same image. The applications could generate run-time wrong results.

Specifically, starting with release 7.0.5.1, a "%DBM-F-ID_MAP, ID number mapping" run-time error may be returned.

This problem has now been fixed. You can mix stream and non-stream DML modules in the same image.

Applies only to embedded DML applications

This fix only applies to modules containing embedded DML. There is still a long-standing restriction mixing stream and non-stream modules containing callable DBQ. Oracle will consider removing this restriction in a future release.

Example:

```
$ CREATE MAIN.FOR
$DECK
C       MAIN.FOR:

        PROGRAM MAIN

        external  START_DEFAULT
        external  FETCH_DEFAULT
        external  END_DEFAULT
        external  START_STREAM
        external  FETCH_STREAM
        external  END_STREAM


        CALL START_DEFAULT ()
        CALL START_STREAM ()

        CALL FETCH_DEFAULT ()
        CALL FETCH_STREAM ()
        CALL FETCH_DEFAULT ()

        CALL END_DEFAULT ()
        CALL END_STREAM ()

        END
$EOD
$ CREATE STREAM.FOR
$DECK
C       ------------------------------------------------------------------------
C       STREAM :
C       ------------------------------------------------------------------------

C       ------------------------------------------------------------------------
       SUBROUTINE START_STREAM ()

        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB2,
     2       DATABASE=PARTS,
     3       STREAM = 11)

        PRINT *, 'READY (STREAM)'
        READY (CONCURRENT, UPDATE)

        RETURN
        END

C       ------------------------------------------------------------------------
        SUBROUTINE FETCH_STREAM ()

        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB2,
     2       DATABASE=PARTS,
     3       STREAM = 11)

        PRINT *, 'FETCH NEXT EMPLOYEE (STREAM)'
        FETCH (NEXT, RECORD = EMPLOYEE)
        PRINT *,EMP_ID
```

```
        RETURN
        END
C       ------------------------------------------------------------------------
       SUBROUTINE END_STREAM ()
        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB2,
     2       DATABASE=PARTS,
     3       STREAM = 11)
        PRINT *, 'ROLLBACK (STREAM)'
        ROLLBACK(STREAM)
        RETURN
        END
C       ------------------------------------------------------------------------
$EOD
$ CREATE DEFAULT.FOR
$DECK
C       ------------------------------------------------------------------------
C       DEFAULT STREAM
C       ------------------------------------------------------------------------
        SUBROUTINE START_DEFAULT ()
        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB1,
     2       DATABASE=PARTS)
        PRINT *,'READY (DEFAULT)'
        READY (CONCURRENT, UPDATE)
        RETURN
        END
C       ------------------------------------------------------------------------
        SUBROUTINE FETCH_DEFAULT ()
        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB1,
     2       DATABASE=PARTS)
        PRINT *,'FETCH NEXT CLASS (DEFAULT)'
        FETCH (NEXT, RECORD = CLASS)
        PRINT *,CLASS_CODE
        RETURN
        END
C       ------------------------------------------------------------------------
        SUBROUTINE END_DEFAULT ()
        INVOKE (SCHEMA=PARTS,
     1       SUBSCHEMA=SUB1,
     2       DATABASE=PARTS)
        PRINT *,'ROLLBACK (DEFAULT)'
        ROLLBACK
        RETURN
        END
$EOD
$
$!-------------------------------------------------------------------------------
$ CREATE SUB1.DDL
$DECK
SUBSCHEMA NAME IS SUB1 FOR PARTS SCHEMA
REALM MAKE
    IS MAKE
```

```
REALM BUY
    IS BUY

RECORD NAME IS CLASS
    ITEM CLASS_CODE TYPE IS CHARACTER 2
    ITEM CLASS_DESC TYPE IS CHARACTER 20
    ITEM CLASS_STATUS TYPE IS CHARACTER 1

RECORD NAME IS PART
    ITEM PART_ID TYPE IS CHARACTER 8
    ITEM PART_DESC TYPE IS CHARACTER 50
    ITEM PART_STATUS TYPE IS CHARACTER 1
    ITEM PART_PRICE TYPE IS FLOATING
    ITEM PART_COST TYPE IS FLOATING
    ITEM PART_SUPPORT TYPE IS CHARACTER 2

SET NAME IS ALL_CLASS

SET NAME IS ALL_PARTS

SET NAME IS CLASS_PART

$EOD

$!-------------------------------------------------------------------------------
$ CREATE SUB2.DDL
$DECK

SUBSCHEMA NAME IS SUB2 FOR PARTS SCHEMA

REALM PERSONNEL
    IS PERSONNEL

RECORD NAME IS EMPLOYEE
    ITEM EMP_ID TYPE IS CHARACTER 5
    ITEM EMP_LAST_NAME TYPE IS CHARACTER 20
    ITEM EMP_FIRST_NAME TYPE IS CHARACTER 10
    ITEM EMP_PHONE TYPE IS CHARACTER 7
    ITEM EMP_LOC TYPE IS CHARACTER 5

RECORD NAME IS DIVISION
    ITEM DIV_NAME TYPE IS CHARACTER 20

SET NAME IS ALL_EMPLOYEES

SET NAME IS MANAGES

SET NAME IS CONSISTS_OF

$EOD

$!-------------------------------------------------------------------------------
$ DBO/RESTORE PARTS.DBB

$ DBO/EXPORT PARTS PARTS.DBM
$ DDL/COMPILE/EXPORT=PARTS.DBM SUB1,SUB2
$ DBO/MODIFY/IMPORT=PARTS.DBM/SUB=(SUB1,SUB2) PARTS
$
$ FORTRAN/LIST/NOOPT MAIN.FOR
$ FORTRAN/DML/LIS/NOOPT     STREAM.FOR
$ FORTRAN/DML/LIS/NOOPT     DEFAULT.FOR
$ LINK/MAP MAIN,DEFAULT,STREAM,SYS$LIBRARY:DBMDML/opt
$!-------------------------------------------------------------------------------
```

If you build and run MAIN.EXE, you should see the following incorrect results:

```
READY (DEFAULT)
READY (STREAM)
FETCH NEXT CLASS (DEFAULT)

FETCH NEXT EMPLOYEE (STREAM)
12333
FETCH NEXT CLASS (DEFAULT)
```

```
ROLLBACK (DEFAULT)
ROLLBACK (STREAM)
```

Note, that there is no CLASS record returned from the FETCH_DEFAULT module. In this example, if you were to switch the order of START_DEFAULT and START_STREAM calls in MAIN.FOR, you would get a totally different (and correct) answer.

```
READY (STREAM)
READY (DEFAULT)
FETCH NEXT CLASS (DEFAULT)
BU
FETCH NEXT EMPLOYEE (STREAM)
75624
FETCH NEXT CLASS (DEFAULT)
BT
ROLLBACK (DEFAULT)
ROLLBACK (STREAM)
```

4
Known Problems and Restrictions

This chapter describes problems and restrictions relating to Oracle COADASYL DBMS release 7.2 and includes workarounds where appropriate.

4.1 Features Not Yet Available for OpenVMS I64

The following features or capabilities or components are not currently available to run or are known to not run reliably on OpenVMS I64 with this Oracle CODASYL DBMS field test release.

- Ada compiler and Oracle Rdb ADA precompilers
- PL/I compiler and Oracle Rdb PL/I precompilers

4.2 Oracle CODASYL DBMS and IEEE Floating Point Support

Currently, Oracle CODASYL DBMS does not support floating point IEEE formats for either OpenVMS Alpha or OpenVMS I64. Because of the default float point behavior on OpenVMS IA64, if your Oracle CODASYL DBMS metadata contains floating point data items, you must compile your OpenVMS I64 applications with the FLOAT=G_FLOAT compiler switch.

Note

This restriction should not impact the storing or fetching float point items with the DBQ utility.

Oracle will look into lifting or easing this restriction for a future release.

For more information about IEEE floating point and OpenVMS I64, please refer to Section 2.1 in this document.

4.3 Expect Additional Memory Consumption

Due to the increased sizes of image files (especially on Integrity servers) and more aggressive buffering and caching schemes and larer I/O size defaults, you should expect to allocate additional page file quota, working set sizes and buffered I/O byte limit quota when using Oracle CODASYL DBMS release 7.2. In particular, when running on Integrity servers, a page file quota of perhaps three times larger may be required for some applications.

4.3.1 ILINK-E-INVOVRINI Error on I64

When linking an application with multiple modules, the following error message may be returned:

```
%ILINK-E-INVOVRINI, incompatible multiple initializations for overlaid section
        section: DBM$UWA_B
        module: M1
        file: DKA0:[BLD]M1.OBJ;1
        module: M2
        file: DKA0:[BLD]SYS.OLB;1
```

On I64 systems, you cannot have a program section that attempts to be initialized a subsequent time where the non-zero portions of the initializations do not match. This is a difference from OpenVMS Alpha and VAX systems where the linker permitted such initializations.

This can be seen when linking multiple FORTRAN DML modules, where some modules use the default (non-stream) UWA, and another uses a "naked" invoke, which only contributes an abbreviated contribution to the DBM$UWA_B psect.

For example, A.FOR contains:

```
      PROGRAM AFOR
      INVOKE (SUBSCHEMA = FORTRAN_SUBSCHEMA,
     1       SCHEMA = PARTS,
     2       DATABASE = PARTS)
      CALL BSUB()
      END
```

B.FOR contains:

```
      SUBROUTINE BSUB
      INVOKE
      RETURN
      END
```

On VAX or ALPHA, the above code will link and run correctly, However, on I64, the linker will generate the following:

```
%ILINK-E-INVOVRINI, incompatible multiple initializations for overlaid section
        section: DBM$UWA_B
        module: AFOR
        file: A.OBJ
        module: BSUB
        file: B.OBJ
```

4.4 SYSTEM-F-INSFMEM Fatal Error With SHARED SYSTEM MEMORY or LARGE MEMORY Enabled in Galaxy Environment

When GALAXY support is enabled in an OpenVMS Galaxy environment, a `%SYSTEM-F-INSFMEM, insufficient dynamic memory error` message may be returned when mapping row caches or opening the database. One source of this problem specific to a Galaxy configuration is running out of Galaxy Shared Memory regions. For Galaxy systems, GLX_SHM_REG is the number of shared memory region structures configured into the Galaxy Management Database (GMDB).

While the default value of 64 regions (for OpenVMS versions through at least V7.3-1) might be adequate for some installations, sites using a larger number of databases or row caches when the SHARED MEMORY IS SYSTEM or LARGE MEMORY IS ENABLED features are enabled may find the default insufficient.

If a %SYSTEM-F-INSFMEM, insufficient dynamic memory error is returned when mapping record caches or opening databases, Oracle Corporation recommends that you increase the GLX_SHM_REG parameter by two times the sum of the number of row caches and number of databases that might be accessed in the Galaxy at one time. As the Galaxy shared memory region structures are not very large, setting this parameter to a higher than required value does not consume a significant amount of physical memory. It also may avoid a later reboot of the Galaxy environment. This parameter must be set on all nodes in the Galaxy.

Galaxy Reboot Required

Changing the GLX_SHM_REG system parameter requires that the OpenVMS Galaxy environment be booted from scratch. That is, all nodes in the Galaxy must be shut down and then the Galaxy reformed by starting each instance.

To enable Galaxy support, issue the command:

```
$ DBO/SET GALAXY/ENABLED <db>
```

To enable SYSTEM SHARED MEMORY, issue to command;

```
$ DBO/MODIFY/MEMORY_MAPPING=SYSTEM <db>
```

To enable LARGE MEMORY for record cache, issue the command:

```
$ DBO/CACHE/MODIFY/LARGE_MEMORY <db> <cache>
```

4.5 Oracle CODASYL DBMS and OpenVMS ODS-5 Volumes

The OpenVMS Version 7.2 release introduced an Extended File Specifications feature, which consists of two major components:

- A new, optional, volume structure, ODS-5, which provides support for file names that are longer and have a greater range of legal characters than in previous versions of OpenVMS.
- Support for “deep” directory trees.

ODS-5 was introduced primarily to provide enhanced file sharing capabilities for users of Advanced Server for OpenVMS 7.2 (formerly known as PATHWORKS for OpenVMS), as well as DCOM and JAVA applications.

In some cases, Oracle CODASYL DBMS performs its own file and directory name parsing and explicitly requires ODS- 2 (the traditional OpenVMS volume structure) file and directory name conventions to be followed. Because of this knowledge, Oracle does not support any Oracle CODASYL DBMS database file components (including root files, storage area files, after-image journal files, record cache backing store files, database backup files, after-image journal backup files, and so forth) that utilize any non-ODS-2 file naming features. For this reason, Oracle recommends that Oracle CODASYL DBMS database components not be located on ODS-5 volumes.

Oracle CODASYL DBMS does support database file components on ODS-5 volumes provided that all of these files and directories strictly follow the ODS-2 file and directory name conventions. In particular, all file names must be specified entirely in uppercase and special characters in file or directory names are forbidden.

4.6 Carryover Locks and NOWAIT Transaction Clarification

In NOWAIT transactions, the BLAST (Blocking AST) mechanism cannot be used. For the blocking user to receive the BLAST signal, the requesting user must request the locked resource with WAIT (which a NOWAIT transaction does not do).

Oracle CODASYL DBMS defines a resource called NOWAIT, which is used to indicate that a NOWAIT transaction has been started. When a NOWAIT transaction starts, the user requests the NOWAIT resource. All other database users hold a lock on the NOWAIT resource so that when the NOWAIT transaction starts, all other users are notified with a NOWAIT BLAST.

The BLAST causes blocking users to release any carryover locks. There can be a delay before the transactions with carryover locks detect the presence of the NOWAIT transaction and release their carryover locks. You can detect this condition by examining the stall messages. If the "Waiting for NOWAIT signal (CW)" stall message appears frequently, the application is probably experiencing a decrease in performance, and you should consider disabling the carryover lock behavior.

4.7 Both Application and Oracle CODASYL DBMS Using SYS$HIBER

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

The Oracle CODASYL DBMS 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 not 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 V7.2, the LIB$WAIT routine has been enhanced through the FLAGS argument (with the LIB$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.

4.8 Row Cache Not Allowed While Hot Standby Replication is Active

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

A new command qualifier, /CACHE=NOENABLED, has been added to the DBO /OPEN command. To open the Hot Standby database prior to starting replication, use the /CACHE=NOENABLED qualifier on the DBO/OPEN command.

4.9 Exclusive Access Transactions May Deadlock with RCS Process

If a record is frequently accessed by long running transactions that request read/write access, reserving the record for exclusive update, and if the record has one or more indexes, you may experience deadlocks between the user process and the row cache server (RCS) process.

There are at least three suggested workarounds to this problem:

1. Reserve the record for CONCURRENT UPDATE.
2. Close the database and disable row cache for the duration of the exclusive transaction

3. Change the checkpoint interval for the RCS process to a time longer than the time required to complete the batch job and then trigger a checkpoint just before the batch job starts. Set the interval back to a smaller interval after the checkpoint completes.