Oracle® CODASYL DBMS

Release Notes

May 2006

Release 7.1.2.4

ORACLE®

Oracle CODASYL DBMS Release Notes, Release 7.1.2.4

Copyright © 1986, 2006 Oracle Corporation. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the Programs on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, back up, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

Oracle is a registered trademark of Oracle Corporation. All other company or product names mentioned are used for identification purposes only and may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or servcies; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Contents

Preface

Purpose of This Manual

The Oracle CODASYL DBMS release 7.1.2.4 release notes summarize new features, corrections to software, restrictions, workarounds, and problems. They also include new features and corrections provided in release 7.1.2, 7.1.2.1, 7.1.2.2, and 7.1.2.3. These release notes cover Oracle CODASYL DBMS for OpenVMS Alpha.

Intended Audience

This document is intended for users responsible for:

- System management
- Database administration
- Application programming

Document Structure

This document consists of five chapters:

Chapter 1	Describes installation requirements and location of documents
Chapter 2	Describes new features and technical changes
Chapter 3	Describes corrected software errors
Chapter 4	Describes known problems, restrictions, and workarounds, as well as documentation errors and omissions
Chapter 5	Describes new features and corrected software errors in releases 7.1.2, 7.1.2.1, 7.1.2.2, and 7.1.2.3.

Conventions

Oracle CODASYL DBMS is often referred to as DBMS.

The following conventions are used in this document:

word	A lowercase word in a format example indicates a syntax element that you supply.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
{ }	Braces enclose clauses from which you must choose one alternative.

. . .

A horizontal ellipsis means you can repeat the previous item.

.
.
.

A vertical ellipsis in an example means that information not directly related to the example has been omitted.

1
Installation and Documentation

This chapter contains installation and documentation information for Oracle CODASYL DBMS release 7.1.2.4.

1.1 Installation of Oracle CODASYL DBMS Software

Please refer to the *CODASYSL DBMS V7.1 Installation Guide* for detailed Oracle CODASYL DBMS installation instructions. Oracle strongly recommends that you read the installation guide before attempting an installation.

To extract either the PostScript (PS) or text (TXT) version of the installation guide from the kit, use one of the following commands:

```
$ BACKUP <device>:DBM07124A071.A/SAVE/SEL=DBM071_INSTALL_GDE.PS
```

or

```
$ BACKUP <device>:DBM07124A071.A/SAVE/SEL=DBM071_INSTALL_GDE.TXT
```

The release 7.1 installation guide is available on MetaLink and OTN in Adobe Acrobat PDF format.

1.2 Documentation in Adobe Acrobat Format

You can view the documentation in Adobe Acrobat format using the Acrobat Reader, which allows anyone to view, navigate, and print documents in the Adobe Portable Document Format (PDF). For information about obtaining a free copy of Acrobat Reader and for information on supported platforms, see the Adobe Web site at:

```
http://www.adobe.com
```

The Oracle CODASYL DBMS and Hot Standby documentation in Adobe Acrobat format is available on MetaLink and OTN.

2

Enhancements Provided in ORACLE CODASYL DBMS Release 7.1.2.4

This chapter describes new and changed features in Oracle CODASYL DBMS release 7.1.2.4.

2.1 Oracle Media Management V2.0 API for Oracle CODASYL DBMS

Starting with this release, Oracle CODASYL DBMS supports the Oracle Media Management release 2.0 API. This interface permits backing up to and restoring from data archiving software applications supporting this interface. Examples of such applications include:

- Archive Backup System for OpenVMS from Hewlett-Packard Corporation on the World Wide Web at `http://h71000.www7.hp.com/openvms/storage/abspage.html`
- LEGATO NetWorker(R) from LEGATO Systems, Inc. on the World Wide Web at `http://www.legato.com/`
- Archive Backup Client (ABC) for OpenVMS from STORServer Inc. on the World Wide Web at `http://www.storserver.com/`

More information on these products is available from the vendors.

2.1.1 New LIBRARIAN Qualifier

In oder to provide the interface to Oracle Media Management API, a new qualifier, /LIBRARIAN, has been added to following DBO commands:

- DBO /BACKUP /MULTITHREAD
- DBO /BACKUP /AFTER_JOURNAL
- DBO /OPTIMIZE /AFTER_JOURNAL
- DBO /RESTORE /MULTITHREAD
- DBO /DUMP /AFTER_JOURNAL
- DBO /DUMP /BACKUP /MULTITHREAD
- DBO /RECOVER

DBO supports the retrieval using the /LIBRARIAN qualifier only for data that has been previously stored by DBO using the /LIBRARIAN qualifier.

The /LIBRARIAN qualifier accepts the following parameters.

- WRITER_THREADS=n

Specifies *n* writer threads to write *n* backup data streams to the LIBRARIAN. The database storage areas will be partitioned among the database streams. The streams will be named BACKUP_FILENAME.EXT, BACKUP_FILENAME.EXT02, BACKUP_FILENAME.EXT03, up to BACKUP_FILENAME.EXT99. BACKUP_FILENAME.EXT is the backup file name specified in the DBO command, excluding any specified device, directory, or version number. The default extension name is .DBF. The WRITER_THREADS parameter can only be specified for database backups. The default is one writer thread. The minimum is one thread; the maximum is 99 threads. If the value exceeds 99, the actual number of writer threads will be set to a value equal to the number of database storage areas.

- READER_THREADS=n

  Specifies *n* reader threads to read all the backup data streams from the LIBRARIAN created for the backup filename. The streams will be named BACKUP_FILENAME.EXT, BACKUP_FILENAME.EXT02, BACKUP_FILENAME.EXT03, up to BACKUP_FILENAME.EXT99. BACKUP_FILENAME.EXT is the backup file name specified in the DBO command, excluding any specified device, directory, or version number. The default extension name is .DBF. The READER_THREADS parameter can only be specified for database restores and dumps of databases stored by DBO in the LIBRARIAN. The default reader thread value of 1 is used for all other DBO commands that read data from the LIBRARIAN. The minimum READER_THREADS value is one; the maximum is 99.

  The number of READER_THREADS specified for a restore should be equal to or less than the number of WRITER_THREADS specified for the database backup. If it is not, the number of reader threads will be set by DBO to be equal to the number of data streams actually stored in the LIBRARIAN by the backup. If the number of READER_THREADS specified is less than the number of WRITER_THREADS, DBO will partition the data streams among the specified reader threads so that all data streams representing the database are restored. Each reader thread may read more than one data stream.

- TRACE_FILE=file_specification

  Specifies that the LIBRARIAN application will write trace data to the named file, if specified.

- LEVEL_TRACE=#

  Specifies the level number of the trace data written by the LIBRARIAN application (levels 0 through 2) or a higher level as defined by the LIBRARIAN application. Level 0 (trace all error conditions) is the default.

- LOGICAL_NAMES=(logical_name=equivalence_value,...)

  This parameter allows the user to specify a list of process logical names which the LIBRARIAN application may use to specify particular catalogs or archives for storing or retrieving backup files, LIBRARIAN debug logical names, and so on. See the LIBRARIAN-specific documentation for the definition of these logical names. The list of process logical names will be defined by DBO prior to the start of the backup or restore operation.

2.1.2 New DBO /LIBRARIAN Command

In addition to the /LIBRARIAN qualifier used with existing DBO commands, there is a new DBO /LIBRARIAN command. This command lets you list or delete data streams stored in the LIBRARIAN implementation based on the backup file name used for the DBO backup. The LIST and REMOVE options cannot be used together in the same DBO/LIBRARIAN command.

```
DBO /LIBRARIAN /LIST=(OUTPUT=disk:[directory]listfile.ext) FILENAME.DBF
DBO /LIBRARIAN /REMOVE=([NO]CONFIRM) FILENAME.DBF
```

FILENAME.DBF is the backup filename. Any device, directory, or version number specified with the backup file name will be ignored. The backup file name must be the same name previously used for an DBO backup to the LIBRARIAN. A default file type of .DBF is assumed if none is specified.

The following command qualifiers are supported:

- /LIST=(OUTPUT=disk:[directory]listfile.ext)

  The LIST qualifier used alone displays output to the default output device. If the OUTPUT option is used, output will be displayed to the specified file. All data streams existing in the LIBRARIAN that were generated for the specified backup name will be listed. The information listed for each data stream name may include:

  - The backup stream name based on the backup file.
  - Any comment associated with the backup stream name.
  - The creation method associated with the backup stream name. This will always be STREAM to indicate creation by a backup operation.
  - The creation date and time when the stream was backed up to the LIBRARIAN.
  - Any expiration data and time specified for deletion of the stream by the LIBRARIAN.
  - The media sharing mode which indicates if the media can be accessed concurrently or not. This is usually the case for disks but not tapes.
  - The file ordering mode which indicates if files on the media can be accessed in random order or sequential order.
  - Any volume label(s) for the media which contain the backup stream.

  Implementation Specific

  Not all of these items will be listed depending on the particular LIBRARIAN implementation.

- /REMOVE=([NO]CONFIRM)

  Use this qualifier to delete all data streams existing in the LIBRARIAN that were generated for the specified backup name. This command should be used with caution. You should be sure that a more recent backup for the database exists in the LIBRARIAN under another name before using this command.

The CONFIRM option is the default. It will prompt you to confirm that you want to delete the backup from the LIBRARIAN. You can then reply Y(ES) to do the deletion or N(O) to exit the command without doing the deletion. Specifying NOCONFIRM will cause the deletion to be done without the confirmation prompt.

The following additional optional keywords can be specified with either the /LIST qualifier or the /REMOVE qualifier. They must be specified and have no defaults. These are the same options discussed earlier for the /LIBRARIAN qualifier used with other DBO commands such as /BACKUP/MULTITHREAD and /RESTORE/MULTITHREAD.

- TRACE_FILE=file_specification
- LEVEL_TRACE=n
- LOGICAL_NAMES=(logical_name=equivalence_value,...)

Oracle Media Manager Release 2.0 Interface

Only applications that conform to Oracle Media Manager V2.0 can be called using the /LIBRARIAN qualifier or the new DBO /LIBRARIAN commands.

2.1.3 Opaque Archive Application

The archive application is effectively an opaque black box for DBO commands; the backup file name is the identifier of the stream of data stored in the archive. The utilities and command procedures specific to the particular LIBRARIAN application must be used to associate devices with the stream of data sent to or retrieved from the archive by DBO. Device specific qualifiers such as /REWIND, /DENSITY or /LABEL cannot be used with this interface.

2.1.4 DBO Backup Streams

Each writer thread for a backup operation or reader thread for a restore operation manages its own stream of data. Therefore, each thread uses a unique backup file name generated from the backup file name specified on the command line. With the exception of the first file name, a number is incremented and added to the end of the backup file extension specified to the archive representing a unique data stream. This number is the equivalent of the volume number associated with non-LIBRARIAN DBO multithreaded backups and restores.

For example, if the following backup command is issued:

```
$DBO /BACKUP/MULTITHREAD /LIBRARIAN=(WRITER_THREADS=3) /LOG DB FILENAM.DBF
```

These backup file data stream names are specified to the archive:

```
FILENAME.DBF
FILENAME.DBF02
FILENAME.DBF03
```

The names identify the three streams of data stored in the archive by the three writer threads which together represent the stored database. Because each data stream must contain at least one database storage area and a single storage area must be completely contained in one data stream, if the number of writer threads specified is greater than the number of storage areas, it will be set equal to the number of storage areas.

If the following command is issued to restore the database:

```
$DBO /RESTORE/MULTITHREAD /LIBRARIAN=(READER_THREADS=3) /LOG FILENAM.DBF
```

These same three data stream backup file names, one name specified by each of the three reader threads, will be generated by DBO and sent to the archive application to retrieve all the data associated with the database.

In this example, the number of reader threads was equal to the number of writer threads specified on the backup. However, these values do not need to be the same. If the number of reader threads is fewer than the number of backup writer threads, one or more restore reader threads will restore more than one data stream. An algorithm is used that assigns the data streams so that each thread will have an approximately equal amount of work to do. If the number of reader threads specified is greater than the number of backup writer threads, the number of reader threads will be set equal to the number of backup writer threads.

2.1.5 Data Stream Naming Considerations

Data stream names representing the database are generated based on the backup file name specified for the DBO backup command. You must either use a different backup file name to store the next backup of the database to the LIBRARIAN application or first delete the existing data streams before the SAME backup file name can be reused for the next backup.

To delete the existing data streams stored in the LIBRARIAN implementation, use a LIBRARIAN management utility or the DBO /LIBRARIAN /REMOVE command with just the backup file name to delete all the data streams generated based on that name. If you want to avoid deleting a previous backup to the LIBRARIAN which used the same backup file name, you can incorporate the date or some other unique identifier in the backup file name when you do each backup to make it unique. Many LIBRARIAN implementations allow you to specify an automatic deletion date for each data stream stored in the archives.

2.1.6 Logical Names To Access LIBRARIAN Application

The following OpenVMS logical names are for use with a LIBRARIAN application. These logical names need to be defined before the DBO backup or restore command is executed and should not be included with the list of logical names specified with the /LIBRARIAN qualifier.

- DBO$LIBRARIAN_PATH

  This logical name must be defined to the file specification for the shareable LIBRARIAN image to be loaded and called by DBO backup and restore operations. The translation must include the file type (.EXE for example) and must not include a version number. The shareable LIBRARIAN shareable image referenced must be an installed (known) image. See the LIBRARIAN implementation documentation for the name and location of this image and how it should be installed.

  ```
  $ DEFINE /SYSTEM /EXECUTIVE_MODE -
      DBO$LIBRARIAN_PATH librarian_shareable_image.exe
  ```

- DBO$DEBUG_SBT

  This logical name is not required. If it is defined to any value, DBO will display debug tracing information messages from modules that make calls to the LIBRARIAN shareable image. This information may be helpful for support analysts from Oracle or your librarian vendor when analyzing problems. See the LIBRARIAN documentation for any other logical names or setup procedures specific to the particular LIBRARIAN implementation.

2.1.7 Limitations

DBO commands used with the /LIBRARIAN qualifier may not specify a list of tape or disk devices. The qualifier accepts a backup file (DBF file) name. Any disk or device specification and version number specified with the backup file name is ignored for the backup file name specified to the archive. For example, device:[directory]FILENAME.DBF;1 is truncated to FILENAME.DBF when the backup file data is stored in or retrieved from the archive.

The /VOLUMES qualifier cannot be used on the DBO/RESTORE/MULTITHREAD command if the /LIBRARIAN qualifier is used. DBO automatically determines the number of data streams stored in the LIBRARIAN implementation based on the backup file name specified for the restore command and sets the volume number to the actual number of stored data streams. This helps to ensure that all data streams which represent the database are retrieved.

3
Problems Corrected

This chapter describes software errors corrected in Oracle CODASYL DBMS release 7.1.2.4.

3.1 Inconsistent Snapshot Results Using COMMIT TO JOURNAL

BUG 5024150

When the COMMIT TO JOURNAL OPTIMIZATION feature was enabled, it was possible for processes executing READ ONLY transactions to get inconsistent results when reading from the database. The problem could be encountered when the following sequence of events occurred:

1. A READ ONLY transaction starts.
2. A READ WRITE transaction starts.
3. The READ WRITE transaction deletes a record and then commits.
4. The READ WRITE transaction inserts a new record reusing the space freed by the previous delete, and commits.
5. The READ ONLY transaction attempts to read the record just deleted from or inserted into the database.

In the above sequence of events, the READ ONLY transaction would conclude that the record was deleted instead of using the contents of the record that were current at the time that the READ ONLY transaction started.

Utilities that use READ ONLY transactions, such as online backups or verifies, could also encounter the problem. Online backups would back up an empty record instead of the old contents.

This problem can be avoided by disabling the COMMIT TO JOURNAL OPTIMIZATION option.

This problem has been corrected.

3.2 Active User Count Incorrect As ABS Starts And Stops

Bug 5134756

In prior versions of Oracle CODASYL DBMS, the statistics counter NUM_ACTIVE was not correctly decremented when the AIJ Backup Server (ABS) process completed a backup. This would lead to an ever-increasing value for the number of users as shown by the DBO /SHOW STATISTICS utility.

This problem has been corrected. The AIJ Backup Server (ABS) process correctly adjusts the active user counter when it exits.

3.3 Invalid Log File Logical Name Causes RCS to Terminate

Bug 5125792

In prior versions of Oracle CODASYL DBMS, the Row Cache Server (RCS) process could fail to correctly start if the RCS log file could not be created.

For example, if the DBM$BIND_RCS_LOG_FILE logical name was defined with an invalid or inaccessible device or directory specification, the RCS process could fail while starting. The monitor log file would contain an entry "%DBM-F-RCSABORTED, record cache server process terminated abnormally " and user processes would be terminated with the status "%DBM-F-TERMINATE, database recovery failed—access to database denied by monitor".

This problem has been corrected. The Row Cache Server (RCS) process now matches the behavior of the other database server processes (such as the database recovery service (DBR)) and will continue running without a log file if the log file cannot be created.

3.4 Bugchecks at KOD$START + 0000080C

BUG 5059527

Beginning in release 7.1.4.3 of Oracle CODASYL DBMS. it was possible to get bugchecks with the following exception:

```
***** Exception at 0193578C : KOD$START + 0000080C
%COSI-F-BUGCHECK, internal consistency failure
```

In release 7.1.4.3, the routine KOD$START was modified to verify that it was not assigned a TSN of zero. If a TSN of zero was assigned, then this bugcheck would occur.

This problem was caused by a small race condition in the code responsible for determining the oldest transaction sequence number (TSN) in the database. Occasionally, if multiple processes were accessing the global oldest TSN location at the same time, the code would incorrectly determine that the oldest TSN was zero.

The only way to completely avoid the problem is to use a previous version of Oracle CODASYL DBMS. The incidence can be reduced by setting the number of cluster nodes for the database to a value greater than one. Note that by doing so, performance features that rely on the number of nodes being one, such as row cache, are disabled. Also, if the system is not part of a cluster, then setting the number of cluster nodes to a value greater than one will have no effect.

This problem has been corrected. The race condition that led to a TSN of zero being assigned has been eliminated.

3.5 DBO/MOVE_AREA Enabled AIJ When It Had Been Disabled

In previous versions of Oracle CODASYL DBMS, the DBO/MOVE_AREA command would enable after image journaling of the moved database even when journaling was disabled prior to the move and there were no journal qualifiers specified on the move command. This has been fixed. Now the AIJ Enabled state of the original database will be preserved unless the AFTER_JOURNAL or AIJ_OPTIONS qualifiers have been used to change the AIJ Enabled state of the database.

A workaround for this problem is to explicitly specify journaling qualifiers on the DBO/MOVE_AREA command, such as DBO/MOVE_AREA/NOAFTER_JOURNAL to disable journaling or DBO/MOVE_AREA/AFTER_JOURNAL to enable it.

3.6 Memory Leak in Attach/Detach

BUG 4866466

Every time a process would attach and detach from a database without exiting the main image, at least 104 bytes of memory would be lost.

The only way to avoid the problem is to periodically rundown the main image and restart the application.

This problem has been corrected.

3.7 Monitor Bugchecks at MON$SEND_REPLY + 0000008C

BUG 4961487

If a database used the AIJ Log Server (ALS) or Row Cache Server (RCS) processes, and the database monitor encountered an error when attempting to start those servers, then the monitor could fail with a bugcheck similar to the following:

```
***** Exception at 000E0BBC : MON$SEND_REPLY + 0000007C
%COSI-F-BUGCHECK, internal consistency failure
```

Examination of the monitor logfile would show that the monitor could not start a server. For example:

```
- sending user attach reply to 0000D369:1
  - "%DBM-F-CANTCREALS, error creating AIJ Log Server process"
  - "-SYSTEM-F-NOSLOT, no PCB available"
```

After encountering the error, the next attempt to attach to the database would result in the bugcheck.

This problem occurred because the monitor neglected to delete the data structure representing the user that had the failed attach attempt. When the server startup failed, an error processing path was used that did not properly delete the structure. When the monitor again attempted to send messages to waiting users, it would attempt to send to the same user again, but that user was not in a state that would allow another message, resulting in the monitor bugcheck.

This is an exceptionally rare problem and would typically only be encountered when the system is low on resources.

This problem has been corrected.

3.8 Row Cache Latching Enhancements and Corrections

In prior releases of Oracle CODASYL DBMS, it was possible for row cache hash latches to be incorrectly held causing hangs. To help avoid these problems, the two latching mechanisms within the row cache feature have been corrected to help eliminate possible race conditions and errant latches without matching unlatches.

In addition, for those hash latches that experience higher levels of contention, the built-in stall timer used between polls of the latch has been reduced to allow more reponsive detection of the latch being released.

Customers are reminded that setting caches to DBO/CACHE/[MODIFY | ADD] /NOREPLACEMENT allows multiple processes to scan internal row cache hash chains simultaneously. This can improve cache search performance for heavily utilized caches.

Finally, a new show statistics screen, Cache Latch Information, may provide additional debugging information.

4

Known Problems, Workarounds, and Documentation Errors

This chapter describes known problems, restrictions, and workarounds, as well as documentation errors and omissions for Oracle CODASYL DBMS release 7.1.2.4.

4.1 AIJ Log Server Process May Loop or Bugcheck

Under unknown but extremely rare conditions, on busy databases where the After Image Journal (AIJ) Log Server process is enabled, the ALS process has been observed to enter a loop condition writing AIJ information to the .AIJ files.

In the worst case, this problem could cause all available journal files to be filled with repeating data. If no remedial action were taken, this condition could cause the database to be shut down, and the AIJ journals to be considered inaccessable.

The database is not corrupted by this problem.

Stopping and restarting the ALS process will clear the looping condition, even if the ALS process must be stopped using the STOP/ID command.

Stopping the ALS process will not impact production as AIJ writes automatically revert to the non-ALS behaviour.

In this release, the behavior has been changed so that if this problem is detected, the ALS process will automatically shut down, producing a bugcheck dump file. This will prevent any danger of filling all available journals and ensure that the dabase remains available.

ALS may be safely restarted immediately as the conditions that cause such a loop are resolved during recovery of the ALS process.

4.2 VMS$MEM_RESIDENT_USER Rights Identifier

Oracle CODASYL DBMS version 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, 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, or to convert or restore a database with these attributes enabled must also hold the same right.

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

4.3 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 DBM_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.

5

New Features and Corrections in Previous Releases

5.1 Corrections in Release 7.1.2.3

This section describes software errors corrected in Oracle CODASYL DBMS release 7.1.2.3.

5.1.1 User Attach Stalls Waiting for ALS Startup

BUG 4767826

It was possible for a user attach request to stall indefinitely when the following occurred:

1. A user had updated the database from one node.
2. A DBO/SHOW STATISTICS session had been started on another node. The database was not manually opened and no other users were attached to the database on that node.
3. The database was forced closed on the first node due to a node failure or by a DBO/CLOSE/ABORT=DELPRC command.
4. Before the failure of the first node could be recovered by the second node, a user attach request is made.

When the above sequence of events occurred, the monitor would show the attach request as pending and would never complete the attach request. For example:

```
$ DBO/SHOW SYSTEM
Oracle CODASYL DBMS X7.1-00 on node CLYPPR 29-NOV-2005 13:18:13.07
    - monitor started 29-NOV-2005 08:25:19.67 (uptime 0 04:52:53)
    - monitor log filename is "DEV:[DIR]DBMMON.LOG"

database DEV:[DIR]PARTS.ROO;1
    - opened 29-NOV-2005 11:57:31.54 (elapsed 0 01:20:41)
    - current after-image journal file is
      DEV:[DIR]AIJ_4.AIJ;1
    - 1 pending database user on this node
```

If the attach request was made while the database recovery process (DBR) was active in recovering the failed attach from the first node, then the attach would stall until another attach was made to the database.

This problem can be avoided by manually opening the database using the DBO/OPEN command on each node that accesses the database, or by having a normal user attach to the database prior to invoking the DBO/SHOW STATISTICS utility.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.3.

5.1.2 Various Bugchecks Due to Memory Corruption

BUG 4762619, 4742021

If an application attached to multiple databases, and those databases did not have an identical number of TSNBLKs allocated to each database, then bugchecks related to memory corruption could occur. Exceptions like the following were typically encountered:

```
***** Exception at 0105C084 : COSI_MEM_GET_VM + 000000E4
***** Exception at 01C96394 : COSI_MEM_GET_VM_2 + 000000A4
***** Exception at 014F7764 : COSI_MEM_FREE_VMLIST + 00000094
```

A TSNBLK is an internal datastructure allocated in the database root (.ROO) file. The number of TSNBLKs allocated is dependent on the number of users and the number of cluster nodes declared for the database. Thus a database that is configured for one cluster node will not have the same number of TSNBLKs allocated for as a database configured for 16 cluster nodes.

This problem was introduced in Oracle CODASYL DBMS releases 7.0.6 and 7.1.2.1.

The problem can be avoided by configuring the databases accessed by the application such that they have the same limits for the number of users and cluster nodes.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.3.

5.1.3 Some DBO/SHOW STATISTICS Screens Not Cleared Properly

BUG 4755183

In several DBO/SHOW STATISTICS multi-page displays for file information, the final page of the display would not be correctly cleared after the last valid item on the page.

This problem has been corrected. The display is now correctly cleared in the “File IO Overview” and “File Lock Overview” displays.

5.2 New Features for Release 7.1.2.2

This section contains new features and technical changes for Oracle CODASYL DBMS release 7.1.2.2.

5.2.1 READER_THREAD_RATIO and DBO/BACKUP/MULTITHREAD

Prior to Oracle CODASYL DBMS release 7.1, the DBO/BACKUP/MULTITHREAD command created one reader thread to read each storage area in the storage area list that was assigned to a writer thread. This could cause a decrease in performance due to reader thread overhead caused by a larger number of reader threads competing for system resources, especially if a large number of storage areas was assigned to a writer thread.

Starting with release 7.1, the DBO/BACKUP/MULTITHREAD command has used a reader thread pool to enhance performance. The pool lessens the reader thread overhead caused when a larger number of reader threads executed at the same time. By default, five reader threads are created and assigned to each writer thread to read the storage areas assigned to that thread for backup. If fewer than five areas are assigned to a writer thread, one reader thread is created to read each storage area.

If the default does not give the best backup performance for a particular database, you can use the /READER_THREAD_RATIO = *n* qualifier to specify the number of reader threads to be created and assigned to each writer thread. If *n* is set to 0, a thread pool is not used; the number of reader threads created and assigned to a writer thread is set to the number of storage areas that writer thread is writing to the backup media. The minimum value for using a thread pool is 1. There is no specific maximum number. If the specified number of reader threads exceeds the number of storage areas assigned to a writer thread, the number will be reduced to equal that number of storage areas. Therefore, no reader threads will be created without work to do.

5.2.2 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.

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.

For example:

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

5.3 Corrections in Release 7.1.2.2

This section describes software errors corrected in Oracle CODASYL DBMS release 7.1.2.2.

5.3.1 Restoring Non-Snapshot Database with Snap=Enabled

In previous versions of Oracle CODASYL DBMS release 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).

5.3.2 Various Problems When Using Lock Partitioning

In prior releases of Oracle CODASYL DBMS, various problems could be experienced when using the LOCK PARTITIONING feature (DBO/MODIFY /PLT). Problems may include excessive snapshot file growth, database corruption, and instability.

This problem has been corrected. Root area lock names are now correctly qualified with the storage area identified and the locks are correctly protected from having invalid lock value blocks by using system-owned null mode locks on each cluster node.

5.3.3 Recovered Database Missing Updates

BUG 4667857

If a database was manually recovered using the original (not backed up) journals, and the journals had been renamed or copied from their original location, or the database was restored with the /NEW option, and instead of listing all journals in a single recover command separate recover commands were used for each journal, then the resulting database could be missing updates.

This problem was caused by the DBO/RECOVER command mistakenly assuming that the copied journal would not have any subsequent journals in the sequence. After recovering the copied journal, the database was incorrectly marked as being consistent through the end of the last journal recovered, even if there were uncommitted transactions still active after the journal was processed.

This problem can be avoided by explicitly naming all journals in a single DBO/RECOVER command, for example:

```
$ DBO/RECOVER/LOG/NOTRACE COPY1, COPY2
```

This problem has been corrected. The database will no longer have the AIJ sequence number advanced if there are transactions still active after recovering a journal that is no longer in its original location.

5.3.4 DBR Bugchecks at PIO$BIND + 000003B8

BUG 4661618

The database recovery process (DBR) could fail with a bugcheck similar to the following when the Row Cache feature was being used in conjunction with Page Transfer Via Memory:

```
***** Exception at 00188958 : PIO$BIND + 000003B8
%COSI-F-BUGCHECK, internal consistency failure
```

The failure would occur when the DBR was attempting to do a “node failure recover all” recovery, but it was not allocated any global buffers by the monitor.

A node failure is considered to have occurred when all users from a node stop accessing a database without clearing their user entries in the database root file, and the monitor accessing the database ends its access without clearing its member ID (MEMBIT) entry in the database root file. When a database monitor detects that a node failure has occurred, special recovery algorithms are invoked if the Row Cache or Page Transfer Via Memory features are being used in the database. In that situation, the DBR will recover all failed users in the database in one pass, instead of having separate DBRs started for each failed user. This is referred to as the recover all algorithm. If a recover all recovery is being done, the monitor will temporarily disable global buffers so that the DBR process is

not constrained by the currently configured global buffer user limit. The DBR is expected to allocate local buffers for page I/O.

In the reported problem, there was a sequence of events that could cause the monitor to fail to disable global buffers, and fail to allocate any global buffers to the DBR process when the DBR attempted to do node failure recovery. For this problem to occur, the database had to have the following features enabled:

- Global Buffers
- Row Cache
- Page Transfer Via Memory
- AIJ Log Server (ALS) automatic

When the following sequence of events occurred, the DBR would fail because it did not have any buffers allocated.

1. The database was manually opened with a DBO/OPEN command.
2. The ALS started and attached to the database.
3. The Row Cache Server (RCS) started.
4. A DBO/CLOSE command was issued.
5. At the same time that the DBO/CLOSE command was issued, the RCS attempted to attach to the database. The attach request was refused by the monitor because the database was being closed.
6. The RCS failed because the monitor refused its attach request.
7. The monitor detected that an RCS process failed, which requires an immediate database shutdown.
8. The monitor terminated the ALS process using $DELPRC.
9. The database is then completely shutdown, with an ALS process needing to be recovered.
10. The database is again opened with the DBO/OPEN command.
11. The monitor cluster watcher detected that the database required node failure recovery.
12. Node failure recovery detected that row cache was enabled and searched for a failed RCS process.
13. No failed RCS process was found.
14. At that point, since no failed RCS process was found, the monitor should have also checked to see if page transfer via memory was enabled. Page transfer via memory requires that the DBR use the same node failure recovery algorithms used by row cache. However, the monitor did not check for page transfer via memory and instead treated the recovery as a regular recovery, not a recover all recovery. Since it was a regular recovery, the monitor did not disable global buffers.
15. The DBR started and sent an attach request to the monitor. Since it was doing a recover all recovery, and global buffers were enabled, the DBR expected that it would have buffers allocated to it by the monitor.

16. The monitor saw that the DBR was recovering an ALS process. The ALS does not use any page buffers so the monitor does not allocate any buffers to the DBR.
17. The DBR finds that it was not allocated any buffers and fails since it needs buffers to do a recover all recovery.

After the above occurred, any attempt to open the database would fail. The DBR would continue to terminate with the bugcheck described. The database had to be restored and the after image journals applied to recover it to its last consistent state.

This problem has been corrected. The monitor will now always check for page transfer via memory when determining if a recover all recovery is necessary.

5.3.5 DBO/REPLICATE AFTER Qualifiers Ignored

When a DBO/REPLICATE AFTER command was issued without a /STANDBY or /MASTER qualifier, all other qualifiers were accepted but ignored.

For example, in the following command the /ONLINE qualifier would be ignored and the previously configured or default setting would be used:

```
$ DBO/REPLICATE AFTER START /ONLINE STANDBY_DB.ROO
```

If the same command was issued with the /MASTER qualifier, then the /ONLINE qualifier would be obeyed.

```
$ DBO/REPLICATE AFTER START /ONLINE /MASTER=MASTER_DB.ROO STANDBY_DB.ROO
```

This problem has been corrected. If the database has previously had replication started, or a CONFIGURE command has been previously issued, then all qualifiers will be obeyed, even if /STANDBY and /MASTER are not specified.

5.3.6 Many Processes Stalling with "hibernating on AIJ submission"

BUG 4638908

If a database was open on multiple nodes, and the AIJ Log Server (ALS) process on one of the nodes failed, the ALS on the node that recovered the failed ALS would start running slowly, only processing AIJ writes every five seconds. This would result in users stalling for an unusually long time with the stall message "hibernating on AIJ submission".

This problem was introduced in release 7.1.2.1. In that release the Database Recovery process (DBR) was changed to clear global data structures related to the ALS when recovering a failed ALS. However, it would incorrectly clear the local node's structures when recovering a failed ALS from another node when it should have only cleared those structures when the local ALS failed. The problem has been fixed in release 7.1.2.2.

This problem can be avoided by not using the /ABORT=DELPRC option when closing a database that has an ALS running. It can also be avoided by manually stopping the ALS prior to forcing other users out of the database with the /ABORT=DELPRC qualfier. The ALS can be manually stopped by issuing the DBO/SERVER AFTER STOP command.

If an ALS process starts running slowly the problem can be resolved by manually stopping and restarting the ALS. For example:

```
$ DBO/SERVER AFTER STOP PARTS
$ DBO/SERVER AFTER START PARTS.
```

5.3.7 Cannot Start Hot Standby if Incremental Restore Done

BUG 4387482

During the initial creation of a standby database, if an incremental backup is applied to the standby database, then attempts to start replication will fail with the following error:

```
%DBM-F-CANTSTARTLRS, error starting AIJ Log Roll-Forward Server process
-DBM-F-CANTSTARTLRS, error starting AIJ Log Roll-Forward Server process
```

The Log Recover Server (LRS) would fail because it assumed that the journals currently available on the standby node would contain all changes that are in the current standby database. When it found that the journals did not contain all the transactions it would fail with the CANTSTARTLRS error.

This problem can be avoided by not applying incremental backups to the standby database.

This problem has been corrected. The LRS will no longer require that the journals on the standby contain all transactions. Instead, it will wait until it has synchronized with the master database before checking to ensure that all journal entries needed to roll forward the standby database to its current state are available.

5.3.8 ALS Would not Stay Stopped

BUG 3297321

If the AIJ Log Server (ALS) mode was set to AUTOMATIC and the ALS was manually stopped using the DBO/SERVER AFTER_JOURNAL STOP command, the ALS would automatically restart when another user attached to the database. The ALS should not have been restarted.

This problem has been corrected. The ALS process will not restart unless it is manually restarted.

5.3.9 AIJ Server Process Not Always Run Under DBMAIJ Account

BUG 2379799

The DBMAIJ (AIJ Server) process would not always run under the DBMAIJ account if a proxy was defined for the user name that started the DBMS monitor on the master node. For example, consider the following hot standby configuration:

- A privileged account on the master node is used to start the DBMS monitor process.
- An identically named user account also exists on the standby node.
- A proxy has been defined on the standby node for the privileged account on the master node that allows proxy network access to the standby node from the master node.

In a configuration like the one described above, it was possible for the AIJ Server process to run under the user name of the privileged account instead of under the DBMAIJ user name declared in the network object definition. Thus the AIJ Server process would inherit the default privileges defined for the privileged account instead of the privileges defined for the RDMAIJ account.

Typically this would not cause any problems, except in situations where the default privileges for the user were less than the DBMAIJ process. In that situation the AIJ Server process would not be able to perform some privileged functions that it would have been able to perform if it was running under the DBMAIJ user name. For example, the AIJ Server process might not be able to increase its process priority, or it might not be able to determine if the Log Recovery Server (LRS) was accumulating CPU.

This problem can be corrected by changing the DECnet network object to disallow proxy access. For DECnet phase IV nodes the following NCP command can be used:

```
$ MCR NCP SET OBJECT DBMAIJ{nn} PROXY NONE
$ MCR NCP DEFINE OBJECT DBMAIJ{nn} PROXY NONE
```

For DECnet-plus nodes the following NCL command can be used:

```
$ MCR NCL SET NODE 0 SESSION CONTROL APPLICATION DBMAIJ{nn} -
  INCOMING PROXY = FALSE
```

Note that in the above examples *nn* should be replaced with the appropriate version number. For example, the object name might be DBMAIJ71.

This problem has been corrected. The DBMAIJSERVER{nn}_NCP.COM and DBMAIJSERVER{nn}_NCL.COM files have been updated to disable proxy access to the DBMAIJ network object.

5.3.10 Standby Journal Disk No Longer Has to Have Same Cluster Size as Master

BUG 3092027

The Hot Standby feature requires that the journal configuration on the standby database be identical to the master database. If the configuration is not identical, attempts to start replication will fail with a %DBM-F-AIJSIGNATURE error. For example:

```
%DBO-I-HOTFAILURE, hot standby failure: check number, size of AIJ journals
 and corresponding device type
%DBM-F-AIJSIGNATURE, standby database AIJ signature does not match master database
```

In previous releases, to ensure that the journal configurations matched, it was necessary to use disks on the standby that had the same cluster size, or a multiple of the cluster size, as the disks used for the master database. In some environments this could be a difficult requirement to meet.

When Hot Standby replication is started, DBMS attempts to verify that the journal configurations between the master and the standby database are the same. This is done by comparing the number of journals and the logical end-of-file size for each of the journals. If those characteristics match, then the configuration is considered to be the same.

In previous releases of DBMS, when a fixed size (circular) journal was created, the logical end-of-file would be advanced to match the physical end-of-file. For example, if a journal was created with an allocation of 513 blocks, and the disk being used had a cluster size of eight blocks, the actual allocated size of the journal would be 520 blocks. Thus the declared allocation size for a journal would often be different from the actual size of the journal.

When the logical end-of-file was being rounded up for new journals, it was necessary to ensure that the disk cluster size on the standby journal disks was compatible with the cluster size on the master journal disks. Failing to do so would often result in %DBM-F-AIJSIGNATURE errors.

This restriction has been lifted. Now, when journals are created, the logical end-of-file is not advanced to the physical end-of-file. That is, the logical end-of-file will now always match the allocation specified when the journal is created. That eliminates the case where the declared allocation sizes of master and standby journals will match but the logical end-of-file sizes do not.

To take advantage of this change, the journals on the master database should be removed and recreated. The standby database can then be created using any available disk as long as the journal configuration (number and allocation size of journals) matches the master. Both the master and standby systems must be running the release of DBMS that has this change.

5.3.11 AIJSIGNATURE Error not Always Returned When Journals Different

BUG 3422435

The Hot Standby feature requires that the journal configuration on the standby database be identical to the master database. If the configuration is not identical, attempts to start replication would fail with an %DBM-F-AIJSIGNATURE error. For example:

```
%DBO-I-HOTFAILURE, hot standby failure: check number, size of AIJ journals
 and corresponding device type
%DBM-F-AIJSIGNATURE, standby database AIJ signature does not match master
 database
```

However, if the journals on the standby were created with allocation sizes that were different from the master, but happened to have the same physical size as the master, no AIJSIGNATURE error was returned. Replication would sometimes start successfully only to shut down later, or a bugcheck would occur.

In previous releases of DBMS, when a fixed size (circular) journal was created, the logical end-of-file would be advanced to match physical end-of-file. For example, if a journal was created with an allocation of 513 blocks, and the disk being used had a cluster size of eight blocks, the actual allocated size of the journal would be 520 blocks. Thus the declared allocation size for a journal would often be different from the actual size of the journal.

When Hot Standby replication is started, DBMS attempts to verify that the journal configurations between the master and the standby database are the same. This is done by comparing the number of journals and the logical end-of-file size for each of the journals. If those characteristics match, then the configuration is considered to be the same. After this validation is done, replication commences and journal entries from the master are copied to the standby database. Hot Standby will attempt to match the journal being copied from the master with the corresponding journal on the standby database by comparing the allocation sizes. However, if no journal on the standby has the same allocation as the journal being replicated from the master, then Hot Standby will fail.

This problem has been resolved. Now, when journals are created, the logical end-of-file is not advanced to the physical end-of-file. That is, the logical end-of-file will now always match the allocation specified when the journal is created. That eliminates the case where the physical journal allocations will match but the declared allocations do not. Attempts to start replication with

journals that don't have matching allocations will now properly fail with an %DBM-F-AIJSIGNATURE error.

This problem can be avoided by always using the matching allocation sizes when creating the standby journal configuration.

5.3.12 Monitor Fails with VASFULL Error

BUG 4548613

The DBMS database monitor process could fail with a COSI-F-VASFULL error after processing a large number of blocking ASTs for cluster membership (MEMBIT) locks. The monitor logfile may contain entries similar to the following:

```
10-AUG-2005 23:47:22.89 - received request from remote node to join
  - database name "DEV:[DIR]FILE.ROO;1"
  - starting forced-exit shutdown of database:
    - "%COSI-F-VASFULL, virtual address space full"
```

A bugcheck dump may be generated containing an exception similar to the following:

```
***** Exception at 000D7E78 : MONLCK$GET_UMAIL + 00000058
%COSI-F-VASFULL, virtual address space full
```

The cluster membership lock is receiving a blocking ASTs when messages similar to the following are seen in the monitor logfile:

```
10-AUG-2005 00:02:37.42 - received request from remote node to join
  - database name "DEV:[DIR]FILE.ROO;1"
  - cluster watcher waiting for MEMBIT lock

10-AUG-2005 00:02:37.42 - lock granted to cluster watcher
  - database name "DEV:[DIR]FILE.ROO;1"
  - cluster watcher is active

10-AUG-2005 00:02:56.24 - received notification of remote monitor deaccess
  - database name "DEV:[DIR]FILE.ROO;1"
  - monitor 2 (OTRNOD) is shutdown
  - cluster recovery completed successfully
  - cluster watcher is active
```

This problem can be avoided by manually opening databases that are often accessed within a cluster by using the DBO/OPEN command.

This problem has now been corrected.

5.3.13 DBM$BIND_HOT_NETWORK_RETRY_COUNT not Consistently Used

BUG 3000359

The Hot Standby feature will attempt to reconnect to the standby database if the network link is lost. By default it will retry one time. The number of retries can be set to a different number by defining the system-wide logical DBM$BIND_HOT_NETWORK_RETRY_COUNT to the desired value. However, depending on the type of network failure, sometimes the defined value would be used, other times it would not.

For example, if the database administrator did not want activity on the master database to stall waiting for a network timeout, he might set the number of retries to zero. In prior releases, even with the number of retries set to zero, Hot Standby would still attempt to reconnect to the standby database.

There is no workaround for this problem.

This problem has been corrected. The retry count specified by the DBM$BIND_HOT_NETWORK_RETRY_COUNT logical will be obeyed any time that a network link is lost.

5.4 Corrections in Release 7.1.2.1

This section describes software errors corrected in Oracle CODASYL DBMS release 7.1.2.1.

5.4.1 Transaction State Not Set in Root

If the COMMIT TO JOURNAL OPTIMIZATION feature was enabled, the transaction state for database users was not correctly shown by the DBO/DUMP /USERS command. That is, the DBO/DUMP/USERS output may have indicated that a process did not have a transaction active when in fact it did. This problem would occur because the database root file data structures that reflected the transaction state of a user were not always updated when the COMMIT TO JOURNAL OPTIMIZATION feature was enabled.

This problem has been corrected in this release of Oracle CODASYL DBMS. The database root file will now correctly show that a user has a transaction active. Note that when the COMMIT TO JOURNAL OPTIMIZATION feature is enabled, the transaction sequence number (TSN) displayed by the DBO/DUMP/USER command will be zero. The TSN is not maintained in the database root file when this optimization is being used.

5.4.2 Database Recovery Process May Fail When AIJs Are Full

When AIJ files were full, it was possible for the database recovery process to fail when trying to recover processes with active read-write transactions. The failure occurred when trying to undo the current uncommitted transaction. The DBR log shows:

```
<timestamp> - Starting transaction UNDO for TSN 0:128
<timestamp> - UNDO TSN 0:128 starts at RUJ JFA (2:0)
<timestamp> - Scanning AIJ for optimistic commit
<timestamp> - Starting AIJ scan at 1:513
%DBM-I-BUGCHKDMP, generating bugcheck dump file SYS$SYSROOT:[SYSEXE]DBMDBRBUG.DMP;
```

Note that the AIJ scan started at 1:513 while the AIJ file is only 512 blocks.

```
The exception in the dump is :
***** Exception at 001B3418 : UTIO$READ_BLOCK + 00000208
%DBM-F-FILACCERR, error reading disk file
-SYSTEM-W-ENDOFFILE, end of file
```

This problem has now been corrected.

5.4.3 Bugchecks in AIJUTL$FREE_DIRTY_ARBS When Journals Full

BUG 4088221

Various processes could fail with bugchecks in AIJUTL$FREE_DIRTY_ARBS if all journals became full, a user process was terminated, a journal was backed up, and further journaling activity in the database occurred.

The bugcheck exception was similar to the following:

```
***** Exception at 011E9E94 : AIJUTL$FREE_DIRTY_ARBS + 000005F4
%COSI-F-BUGCHECK, internal consistency failure
```

This problem can be avoided by ensuring that journals are backed up prior to all journals becoming full.

This problem has now been corrected.

5.4.4 Excessive CPU Consumed by LRS Process

BUG 4268610

When you enabled the Hot Standby feature the AIJ Log Recovery Server (LRS) process would sometimes consume inordinate amounts of CPU. This would especially be true when the following conditions were met:

- Many short duration transactions were being generated by the application
- Many journal slots were allocated in the database
- The current journal sequence number was relatively large, for example, in the tens of thousands

When the above conditions were met, a problem in the recovery code statistics collection was exposed. The recovery code was attempting to determine how many blocks of journal were spanned in each committed transaction. However, the recovery code did not record the journal starting point for the transaction, thus it would appear that the transaction started in journal sequence number 0. The statistics code would then first attempt to find journal sequence 0. When it did not find sequence 0, it would then try to find sequence 1 so that it could approximate the number of blocks in the transaction. That journal would not be found so it would continue searching for journals until it tried all possible journal sequence numbers up to the current journal. This could consume huge amounts of CPU time.

There is no workaround for this problem.

This problem has been corrected in this release. The recovery code will no longer attempt to gather statistics on the number of journal blocks per transaction because the number is not useful in the context of database recovery.

5.4.5 DBO /CLOSE /ABORT=DELPRC /WAIT Hang

Bug 4304166

Starting with Oracle CODASYL DBMS release 7.1, it was possible that when you enabled the row cache feature, closing a database with the /ABORT=DELPRC and /WAIT qualifiers could hang. The problem caused the database recovery processes and the record cache server process to become deadlocked. To actually close the database may require manual intervention to explicitly STOP/ID the database recovery processes.

With this combination of command line qualifiers, the DBMS monitor process was incorrectly issuing a $DELPRC command to the record cache server process. The /ABORT=DELPRC and /ABORT=FORCEX qualifiers are intended to apply only to database user processes and not to database servers when the /WAIT qualifier is not specified.

This problem has been corrected. The DBMS monitor process no longer attempts to issue a $DELPRC command to the record cache server process when the /WAIT qualifier is specified.

5.4.6 Failed User Processess Not Recovered if DBR Startup Fails

Failed user processes would not always be recovered if a database shutdown was forced due to an error that occurred when the database monitor attempted to create a database recovery (DBR) process.

For example, if there were insufficient process slots available on the system to create another process and the database monitor could not create a DBR, then the database would be shut down. The forced shutdown would leave behind user entries in the database for all the processes that were accessing the database from that node. However, the monitor would incorrectly clear the entry in the data structure used to determine if a node recovery was needed. This would prevent the failed processes from being recovered the next time that the database was accessed.

Since the failed processes were not recovered, it is possible that database corruption could be introduced because changes made by the failed processes were not rolled back before other processes accessed the data. It is possible that logical inconsistincies could have been introduced in the database that cannot be detected by the DBO/VERIFY command. If this problem is encountered, it is advised that the database be restored from the last backup and after-image journals be applied to recover the database up to the point of failure. Recovery past the point of failure may be possible, but could re-introduce corruption.

This problem can be avoided by ensuring that there are sufficient system resources available to start DBR processes when needed.

This problem has now been corrected.

5.4.7 DBO/SHOW AFTER/BACKUP May Stall When AIJs Are Full

Bug 4347617

In previous releases of Oracle CODASYL DBMS, if you used mutiple circular journals which had a state of Full, the DBO/SHOW AFTER/BACKUP command could stall with a "waiting for AIJ journal lock 0 (PW)" error.

This problem has now been corrected.

5.4.8 Bugcheck at KOD$UNBIND

Bug 4469657

Starting with Oracle CODASYL DBMS V7.1.1.1, if you 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.

5.4.9 Journals not Initialized After Backup If Backing Up to Tape Device

BUG 2808539

If after image-journal backups were being done to tape, the backed up journal would not get properly initialized. The could lead to various issues, such as journaling being shutdown with a “journal is not empty” error. When that occurred, the journal state displayed by the DBO/DUMP/HEADER=JOURNAL command would show the following lines of output:

```
File is inaccessible
 journal has been made inaccessible by system
 journal is not empty
```

Other symptoms were also possible. The database recovery process (DBR) could fail with a bugcheck in DBR$RECOVER_ALL. Attempts to recover a database using an existing journal that was not properly re-initialized could fail with AIJCORRUPT errors. For example:

```
%DBO-W-AIJCORRUPT, journal entry 1451795/3364123 contains a new AIJBL that doesn't
 have the start flag set
```

This problem was introduced in Oracle CODASYL DBMS release 7.1.1 and only occurs with circular AIJs. Extensible After Image Journaling is not affected.

The problem can be avoided by backing up the journals to a disk destination or using extensible journaling.

This problem has now been corrected.

5.4.10 Constant Snapshot File Growth

With Oracle CODASYL DBMS it was possible for snapshot files to continually extend. This would occur after an abnormal user termination was handled by a database recovery (DBR) process. The problem was seen after installing Oracle CODASYL DBMS release 7.1.2.

There are two data structures in the database root (.ROO) file that are used to represent the state of a database user:

- The RTUPB list
- The TSNBLKs

In release 7.1.2, when the DBR process would clear the failed user's entries in the root file, it would neglect to clear the TSNBLK entry. That would cause DBMS to include the old user's last transaction sequence number (TSN) when determining what TSN should be granted to new snapshot (read only) transactions. Typically, the TSNBLK entry would soon get reused by another user and no symptoms of the problem would be seen. Occasionally, it was possible for the old TSNBLK entry to linger for quite some time. As long as the old TSN was still in the TSNBLK, pages in the snapshot files with TSNs greater than the old user's TSN could not be reclaimed. This would cause the snapshots files to grow since pages in the file could not be reused.

Take the following steps to determine if this problem is affecting a database:

1. Determine the TSN being granted to all snapshot transactions:

```
$ DBO/DUMP/HEADER/OPTION=DEBUG/OUTPUT=TEMP.TXT DB
$ SEARCH TEMP.TXT /WINDOW=(0,3) "Snapshot transaction in progress"
    Snapshot transaction in progress
    Last Process quiet-point was AIJ sequence 0
    Transaction sequence number is 0:3392
```

   Note that in the above example the TSN is 3392.

2. See if there is a TSNBLK entry showing an active transaction with that TSN:

```
$ SEARCH TEMP.TXT 3392
    Transaction sequence number is 0:3392
  SLOT[1.] SIP TSN = 0:3392, COMMIT_TSN = 0:0.
  SLOT[2.] WIP TSN = 0:3392, COMMIT_TSN = 0:3390.
```

Note that in the above example there is a line that contains "WIP TSN = 0:3392". This indicates that there should be a user with a read-write transaction with sequence number 3392.

3. Confirm that there are no read-write transactions active with that TSN:

```
$ PIPE SEARCH /WINDOW=(0,3) TEMP.TXT "Read/write transaction in progress" | -
  SEARCH SYS$INPUT "0:3392"
%SEARCH-I-NOMATCHES, no strings matched
```

No read-write transactions exist with that TSN, thus the TSNBLK contains an invalid entry.

This problem can be corrected by forcing the TSNBLK entry to get reused. That can be done by adding multiple concurrent attaches to the database until the TSNBLK entry is reclaimed by one of the new users. For example:

```
DBQ> BIND DB ON STREAM 1;
DBQ> READY CONC UPDATE;
DBQ> BIND DB ON STREAM 2;
DBQ> READY CONC UPDATE;
...
```

Since each TSNBLK can only be used by one node in the cluster it may be necessary to do attaches from each node that currently has the database open. Continue to add attaches to the database until the "WIP TSN =" string noted above is no longer found in the DBO/DUMP/HEADER output.

This problem has been corrected in Oracle CODASYL DBMS Release 7.1.2.1. The TSNBLK is now properly cleared by the DBR when recovering a failed user.

5.5 New Features for Release 7.1.2

This section contains new features and technical changes for Oracle CODASYL DBMS release 7.1.2.

5.5.1 Support for OpenVMS Version 8.2

This release of Oracle CODASYL DBMS, release 7.1.2, is the first 7.1-x release that supports OpenVMS Version 8.2. Prior releases of Oracle CODASYL DBMS 7.1 will not run on OpenVMS V8.2.

The minimum version of OpenVMS supported by this release is Version 7.2.

5.5.2 DBO /BACKUP /MULTITHREAD New ALLOCATION_QUANTITY Qualifier

An ALLOCATION_QUANTITY=number-blocks qualifier has been added to the DBO /BACKUP /MULTITHREAD command. This qualifier specifies the size, in blocks, which is initially allocated to the backup file. The minimum value for the number-blocks parameter is 1; the maximum value allowed is 2147483647. If you do not specify the ALLOCATION_QUANTITY qualifier, the EXTEND_QUANTITY value effectively controls the initial allocation of the file.

The ALLOCATION_QUANTITY qualifier cannot be used with backup to tape operations.

5.5.3 DBM$BIND_SNAP_QUIET_POINT Logical Reinstated

Over the years, various problems have been reported related to quiet point backups. In particular, database backups and journal backups would sometimes fail with the following error:

```
%DBO-F-TIMEOUT, timeout on quiet
```

Quiet point backups are required so that recovery of a journal can be done without always requiring previous journals. Full database backups can avoid lock conflicts if they wait for the quiet point lock. See the documentation for the DBO/BACKUP commands for more information regarding quiet point backups.

Lock timeout errors for the quiet lock are intended to be returned when a long running update transaction has not completed within the timeout period. However, Oracle CODASYL DBMS Release 6.0 forced all transactions (read-only or read-write) to obtain the quiet point lock when starting. That greatly increased the incidence of timeout errors from backups. To remedy this situation the logical DBM$BIND_SNAP_QUIET_POINT was implemented to force processes starting read-only transactions to release the quiet point lock. If that logical was defined to the value 0, read-only transactions would not obtain the quiet point lock.

Defining the DBM$BIND_SNAP_QUIET_POINT logical to 0 would usually resolve problems with timeouts on the quiet lock, but it would effectively disable the fast commit performance feature for applications that often switched between read-only and read-write transactions. (See BUG 884004.) To remedy that situation the transaction start code was modified to retain the quiet lock during read-only transactions, but release the lock during the read-only transaction if a backup process started. With that change the DBM$BIND_SNAP_QUIET_POINT lock logical could be defined without impacting the performance of the fast commit feature. The change was introduced in releases 7.0.4.3 and 7.1.0.

That fix resolved performance problems, but the logical still could not be defined to 0 if the hot standby feature was being used. If the hot standby feature was being utilized then the logical had to be undefined, or defined to the default value of 1. Applications that utilized hot standby were still subject to undeserved timeouts from backup commands. In releases 7.0.5 and 7.1.1 the hot standby feature was modified so that it did not require read-only transactions to hold the quiet point lock. Also, because read-only processes would usually release the quiet point lock when a backup started, the DBM$BIND_SNAP_QUIET_POINT logical was completely removed.

After these changes, quiet lock timeouts were no longer an issue for most applications. However, a read-only transaction would only release the quiet point lock when Oracle CODASYL DBMS had returned control to the user application. Some complicated queries could execute for an exceptionally long period of time before returning a record to the user application and thus might not release the quiet lock in time to prevent timeout errors for the backup process. In addition, any update transaction that started after the backup process requested the quiet point lock would stall until the long running read-only request returned control to the user application. That means that it was possible for many database users to stall waiting for the read-only transaction to complete, even though they had released the quiet point lock.

This release of Oracle CODASYL DBMS reinstates the DBM$BIND_SNAP_QUIET_POINT logical so that read-only transactions can be forced to release the quiet point lock before starting. The logical now has a slightly different meaning than the original implementation. The default value is

still 1, but that value now signifies that all transactions will hold the quiet point lock until a backup process requests it. Read-only transactions will not obtain the quiet point lock; only read-write requests will obtain the quiet point lock. This is the behavior that was introduced in response to BUG 884004. If the logical is defined to be 0, read-only transactions will always release the quiet point lock at the beginning of the transaction, regardless of the existence of a backup process. That implies that all modified buffers in the buffer pool have to be written to disk before the transaction proceeds. Applications that utilize the fast commit feature and often switch between read-only and read-write transactions within a single attach may experience performance degradation if the logical is defined to 0. This is the behavior that was in place prior to releases 7.0.4.3 and 7.1.0.

Oracle Corporation recommends that you do not define the DBM$BIND_SNAP_QUIET_POINT logical for most applications. If you encounter the scenario described in BUG 3908414, you can define the logical to be 0 to force read-only transactions to always release the quiet point lock. Rather than define the logical system wide, this logical can be defined for specific jobs that are likely to execute for an extensive period of time before returning to the user application. This parameter may also be manipulated from the DBO /SHOW STATISTICS locking dashboard. That way most users will not have to sacrifice fast commit performance when switching between read-only and read-write transactions. As of releases 7.0.5 and 7.1.1, hot standby is no longer affected by this logical, so hot standby is not a factor when determining how to define the logical.

5.5.4 DBO Operator Notification Syntax Change

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

In versions of Oracle CODASYL 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 DBO/CREATE and DBO/MODIFY syntax 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
```

Operator-name was a list of one or more standard VMS operator classes:

- CENTRAL
- CLUSTER
- CONSOLE
- DISKS
- OPER1
- OPER2
- OPER3
- OPER4
- OPER5

- OPER6
- OPER7
- OPER8
- OPER9
- OPER10
- OPER11
- OPER12
- SECURITY

Starting with Oracle CODASYL DBMS release 7.1.0, operator notification has been expanded to include non-journal related events (refer to the *Oracle CODASYL DBMS Release Notes, Release 7.1* 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.

5.6 Corrections in Release 7.1.2

This section describes software errors corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.1 Logical Names Translated Twice

Starting with Oracle CODASYL DBMS release 7.1, many logical name translation requests within DBMS components were incorrectly performed two times.

This problem has been corrected. Logical names are no longer translated twice.

5.6.2 Incorrect Backup of Empty Single AIJ File

Starting with Oracle CODASYL DBMS release 7.1.1, if you backed up an empty single extendable AIJ file, that is, an AIJ with only an open record, the "last commit TSN" field in the open record was incorrectly reset to zero as shown below:

```
$ dbo/dump/after/nodata foo_aij.aij

1/1             TYPE=O, LENGTH=510, TAD=24-SEP-2004 04:48:43.32, CSM=00
    Database MYDISK:[MYDB]FOO.ROO;2
    Database timestamp is 24-SEP-2004 04:48:41.11
    Facility is "DBMAIJ ", Version is 711.1
    Database version is 71.0
    AIJ Sequence Number is 0
    Last Commit TSN is 0:0
    .
    .
    .
```

If you tried to recover the AIJ file, the recovery process ignored all transactions from the file as shown below :

```
$ dbo/recover foo_aij.aij
%DBO-I-LOGRECDB, recovering database file MYDISK:[MYDB]FOO.ROO;2
%DBO-I-LOGOPNAIJ, opened journal file MYDISK:[MYDB]FOO_AIJ.AIJ;1 at 24-SEP-2004 05:07:56.35
%DBO-I-LOGRECSTAT, transaction with TSN 0:128 ignored
%DBO-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations completed
%DBO-I-LOGRECOVR,  0 transactions committed
%DBO-I-LOGRECOVR,  0 transactions rolled back
%DBO-I-LOGRECOVR,  1 transaction ignored
   .
   .
   .
```

To avoid the problem, you can use multiple circular AIJs instead of a single extendable AIJ.

When you back up an empty single extendable AIJ file, no backup file is produced, and the AIJ open record is left unchanged. The AIJ roll forward works as expected.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.3 DBO/SHOW AFTER_JOURNAL/BACKUP_CONTEXT Reset DBM$AIJ_BACKUP_SEQNO

In previous releases the DBO/SHOW AFTER_JOURNAL/BACKUP_CONTEXT command reset the DBM$AIJ_BACKUP_SEQNO symbol to -1 while the DBO /BACKUP/AFTER command was setting it correctly:

```
$ dbo/backup/after/nolog db ""
$ sh symbol DBM$AIJ_BACKUP_SEQNO
  RDM$AIJ_BACKUP_SEQNO == "0"
$ dbo/show after/backup/out=tdb:x.x db
$ sh symbol DBM$AIJ_BACKUP_SEQNO
  DBM$AIJ_BACKUP_SEQNO == "-1"
```

The DBO/SHOW AFTER_JOURNAL/BACKUP_CONTEXT command now sets the DBM$AIJ_BACKUP_SEQNO symbol correctly:

```
$ dbo/backup/after/nolog db ""
$ sh symbol DBM$AIJ_BACKUP_SEQNO
  DBM$AIJ_BACKUP_SEQNO == "0"
$ dbo/show after/backup/out=tdb:x.x db
$ sh symbol DBM$AIJ_BACKUP_SEQNO
  DBM$AIJ_BACKUP_SEQNO == "0"
```

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.4 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.

5.6.5 NOREQIDT Error After Many Attach/Detaches

An application that utilized lock timeouts (for example, the WAIT clause of the READY statement) and often disconnected from a database could eventually encounter the following error:

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

This problem could be reproduced by creating two login sessions. The first session would lock a record. The second session would repeatedly attempt to access the same record. After each attempt it would disconnect from the database and attach again.

This problem can be avoided by not repeatedly detaching from and attaching to the database within a single invocation of a database application, or by not using the lock timeout feature.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.6 Processes Hang in HIB State

Oracle CODASYL DBMS applications that hibernate may occasionally hang in HIB state if after-image journaling (AIJ) is enabled. Applications utilizing the OpenVMS POSIX Threads Library will often hibernate, so threaded applications are especially vulnerable to this issue.

This problem could occur if a user application had executed the OpenVMS $HIBER system service and one of the following Oracle CODASYL DBMS events occurred while the process was hibernating:

- A Global Checkpoint request was issued by a journal switch or a DBO /CHECKPOINT command.
- A checkpoint timer expired.
- A two-phase commit (2PC or DECdtm) transaction that involved the process was ended.

If the OpenVMS POSIX Threads Library is being utilized, the OpenVMS remedial kit VMS732_SYS-V0600 may also be required to completely resolve this issue.

This problem can be avoided by disabling journaling. Oracle Corporation does not recommend disabling journaling unless another suitable database recovery method is available.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.7 DBR Bugchecks at DBR$DDTM_RESOLVE + 000005A8

If a process was involved in a two-phase commit (2PC or DECdtm) transaction, and it failed after the prepare phase of a transaction but before the commit phase, and the database did not have the FAST COMMIT feature enabled, it was possible for the database recovery process (DBR) to fail with the following exception:

```
***** Exception at 00071858 : DBR$DDTM_RESOLVE + 000005A8
%COSI-F-BUGCHECK, internal consistency failure
```

The problem was introduced in Oracle CODASYL DBMS releases 7.0.5.1 and 7.1.1.

Enabling the FAST COMMIT feature will prevent this problem.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.8 DBR Bugchecks at DBR$WAKE_ALL + 000000F4

Previously it was possible for a database recovery process (DBR) to fail with the following exceptions:

```
***** Exception at 00088C54 : DBR$WAKE_ALL + 000000F4
%COSI-F-UNEXPERR, unexpected system error
-SYSTEM-F-REMRSRC, insufficient system resources at remote node

***** Exception at 00088C54 : DBR$WAKE_ALL + 000000F4
%COSI-F-UNEXPERR, unexpected system error
-SYSTEM-F-NOSUCHTHREAD, specified kernel thread does not exist
```

The DBR did not anticipate the possibility of those errors occurring when it issued the OpenVMS $WAKE system service and would terminate when those error status codes were returned by the system service.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.9 Database Corruption When 2PC Transaction Fails

It was possible for a database to become corrupt when the following conditions were met:

- The FAST COMMIT feature was enabled
- A failing process was a participant in a two-phase commit (2PC) transaction
- The process failed after completing the prepare phase of a 2PC transaction but before completing the commit phase
- Other participants in the same transaction failed before completing the prepare phase

In this situation the database recovery process (DBR) would neglect to roll back the failed transaction.

This problem was introduced in Releases 7.0.5.1 and 7.1.1. There is no workaround for this problem.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.10 DBMS Hangs with No Stall Messages

When Oracle CODASYL DBMS called RMS for asynchronous operation, if the call to RMS returned an error, Oracle CODASYL DBMS would hang waiting for a completion AST from RMS. Often, no stall message would be displayed by DBO/SHOW STATISTICS.

For example, if Oracle CODASYL DBMS was using a temporary file on disk, and it did not have sufficient ASTLM quota, it could hang when attempting to read from or write to the temporary file. Oracle CODASYL DBMS may have issued many asynchronous disk writes (ABW) to flush modified page buffers to disk, consuming all available ASTLM quota. While that I/O was underway, if Oracle CODASYL DBMS attempted to access a temporary RMS file, it would not properly handle the error returned by RMS and would hang.

Oracle CODASYL DBMS will now properly trap RMS errors and report them to the application. For example, the following error may be returned if quota is exhausted:

```
%DBO-F-IO_ERROR, input or output error
-COSI-F-READERR, read error
-RMS-F-CDA, cannot deliver AST
```

This problem can be avoided by ensuring that users have sufficient DIOLM and ASTLM quota to concurrently flush all allocated page buffers, while also having enough additional quota to do I/O to other files that are used by Oracle CODASYL DBMS, such as journals, temporary files, etc. For example, if a process is using 500 page buffers, the minimum quota values for DIOLM and ASTLM would be greater than 500.

This problem has been corrected in Oracle CODASYL DBMS release 7.1.2.

5.6.11 %DBM-F-BADPROTOCOL Error on Remote Access

It was possible, although unlikely, that a remote operation would return a %DBM-F-BADPROTOCOL message from DBMSERVER.

This error would only occur if the amount of data being sent by the client application happened to be an exact multiple of the buffer size used to transfer information to the server.

This problem would only affect remote access and would be most likely to occur while trying to bind to a database using a very large subchema. There is no workaround, although using a different subschema would generally succeed.

This problem has been fixed in Oracle CODASYL DBMS release 7.1.2.

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

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 errors or produce 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 been fixed in Oracle CODASYL DBMS release 7.1.2. You can mix stream and non-stream DML modules in the same image.

Applies only to embedded DML applications

This fix applies only 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.

The following example illustrates the problem behavior which occurred in previous releases:

```
$ 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 would 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)
```

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)
```