Oracle® Exadata Storage Server Software

User's Guide

11*g* Release 2 (11.2)

E13861-11

April 2011

ORACLE®

Oracle Exadata Storage Server Software User's Guide, 11*g* Release 2 (11.2)

E13861-11

Primary Author: Caroline Johnston

Contributors: Andrew Babb, Bharat Baddepudi, Mark Bauer, John Berensniewicz, Nilesh Choudhury, Kevin Closson, Ravindra Dani, Greg Doherty, Boris Erlikhman, Roger Hansen, Shrikumar Hariharasubrahmanian, Bill Hodak, Kevin Jernigan, Poojan Kumar, Sue Lee, Juan Loaiza, Catherine Luu, Varun Malhotra, Valarie Moore, Adrian Ng, Michael Nowak, Tony Ouk, Umesh Panchaksharaiah, Sugam Pandey, Greg Pogossiants, Dmitry Potapov, Samitha Samaranayake, Mark Scardina, Akshay Shah, Kesavan Srinivasan, Mahesh Subramaniam, Michelle Sun, Alex Tsukerman, Kothanda Umamageswaran, Doug Utzig, Ron Weiss

Contents

4 Configuring Security for Oracle Exadata Storage Server Software

5 Maintaining Oracle Exadata Storage Server Software

8 Using the CellCLI Utility

List of Tables

List of Examples

Preface

Oracle Exadata Storage Server Software User's Guide describes how to initialize and administer Oracle Exadata Storage Server Software. This guide describes the Oracle Exadata Storage Server Software product and its components, as well as Oracle Exadata Storage Server Software administrative and deployment procedures. This preface contains the following topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

Oracle Exadata Storage Server Software User's Guide is intended for Oracle Database and storage administrators who perform the following tasks:

- Configure Oracle Exadata Storage Server Software
- Manage Oracle Exadata Storage Server Software
- Troubleshoot Oracle Exadata Storage Server Software

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/support/contact.html` or visit `http://www.oracle.com/accessibility/support.html` if you are hearing impaired.

Related Documents

For additional information, see the following Oracle resources:

- *Oracle Exadata Storage Server Software Release Notes*
- *Oracle Exadata Database Machine Owner's Guide*
- *Oracle Database 2 Day DBA*
- *Oracle Database Administrator's Guide*
- *Oracle Database Concepts*
- *Oracle Automatic Storage Management Administrator's Guide*
- *Oracle Database Error Messages*
- *Oracle Database 2 Day + Real Application Clusters Guide*
- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle Real Application Clusters Administration and Deployment Guide*
- Platform-specific Oracle Database, Oracle Clusterware, and Oracle Real Application Clusters installation guides

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, emphasis, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.
$ prompt	The dollar sign ($) prompt indicates a command run as the `oracle` user.
# prompt	The pound (#) prompt indicates a command that is run as the `root` user.

What's New in Oracle Exadata Storage Server Software

This chapter describes the new features included in Oracle Exadata Storage Server Software 11*g* Release 2 (11.2). The following topics are included in this chapter:

- Exadata Secure Erase
- Exadata Smart Flash Cache
- Exadata Hybrid Columnar Compression
- Storage Index
- Smart Scan of Encrypted Data
- Optimized Smart Scan
- Interleaved Grid Disks
- Data Mining
- Enhanced Manageability Features

Exadata Secure Erase

Oracle Exadata Storage Server Software includes a method to securely erase and clean physical disks before redeployment. The `ERASE` option overwrites the existing content on the disks with one pass, three passes or seven passes. The one pass option overwrites content with zeros. The three pass option follows recommendations from NNSA and the seven pass option follows recommendations from DOD.

The following table shows approximate time needed to securely erase a drive using the supported algorithms.

Type of Drive	One Pass	Three Pass	Seven Pass
600 GB drive	1 hour	3 hours	7 hours
2 TB drive	5 hours	15 hours	35 hours
Flash Drive (22.875 GB)	NA	NA	21 minutes

Notes:

- Oracle Exadata Storage Server Software secure data erase uses multiple over-writes of all accessible data. The over-writes use variations of data characters. This method of data erase is based on commonly known algorithms. Under rare conditions even a 7-pass erase may not remove all traces of data. For example, if a disk has internally remapped sectors, then some data may remain physically on the disk. This data will not be accessible using normal I/O interfaces.
- Using tablespace encryption is another way to secure data.

Exadata Smart Flash Cache

Exadata Smart Flash Cache provides a caching mechanism for frequently-accessed data on each Exadata Cell. It is a write-through cache which is useful for absorbing repeated random reads, and very beneficial to online transaction processing (OLTP). It provides a mechanism to cache data in `KEEP` mode using database-side storage clause hints. The Exadata Smart Flash Cache area on flash disks is automatically created on Exadata Cells during start up.

Oracle Exadata Storage Servers are equipped with high-performance flash disks in addition to traditional rotational hard disks. These high-performance flash disks can be used to create Exadata grid disks to store frequently accessed data. It requires the user to do accurate space planning, and to place the most active tablespaces on the premium disks. An alternative option is to dedicate all or part of flash disk space for Exadata Smart Flash Cache. In this case, the most frequently-accessed data on the spinning disks are automatically cached in the Exadata Smart Flash Cache area on high-performance flash disks. When the database needs to access this data Oracle Exadata Storage Server fetches the data from Exadata Smart Flash Cache instead of getting it from slower rotational disks.

When a partition or a table is scanned by the database, Oracle Exadata Storage Server can fetch the data being scanned from the Exadata Smart Flash Cache if the object has the `CELL_FLASH_CACHE` attribute set. In addition to serving data from the Exadata Flash Cache, Oracle Exadata Storage Server also has the capability to fetch the object being scanned from hard disks.

The performance delivered by Oracle Exadata Storage Server is additive when it fetches scanned data from the Exadata Smart Flash Cache and hard disks. Oracle Exadata Storage Server has the ability to utilize the maximum Exadata Smart Flash Cache bandwidth and the maximum hard disk bandwidth to scan an object, and give an additive maximum bandwidth while scanning concurrently from both.

Oracle Database and Exadata Smart Flash Cache software work closely with each other. When the database sends a read or write request to Oracle Exadata Storage Server, it includes additional information in the request about whether the data is likely to be accessed again, and should be cached. For example, when writing data to a log file or to a mirrored copy, the database sends a hint to skip caching. When reading a table index, the database sends a hint to cache the data. This cooperation allows optimized usage of Exadata Smart Flash Cache space to store only the most frequently-accessed data.

Users have additional control over which database objects, such as tablespace, tables, and so on, should be cached more aggressively than others, and which ones should not be cached at all. Control is provided by the new storage clause attribute, CELL_FLASH_CACHE, which can be assigned to a database object.

For example, to pin table CALLDETAIL in Exadata Smart Flash Cache one can use the following command:

```
ALTER TABLE calldetail STORAGE (CELL_FLASH_CACHE KEEP)
```

Oracle Exadata Storage Server caches data for the CALLDETAIL table more aggressively and tries to keep this data in Exadata Smart Flash Cache longer than cached data for other tables. If the CALLDETAIL table is spread across several Oracle Exadata Storage Servers, then each one caches its part of the table in its own Exadata Smart Flash Cache. If caches are sufficient size, then CALLDETAIL table is likely to be completely cached over time.

Exadata Hybrid Columnar Compression

Exadata Hybrid Columnar Compression offers higher compression levels for direct path loaded data. This new compression capability is recommended for data that is not updated frequently. You can specify Exadata Hybrid Columnar Compression at the partition, table, and tablespace level. You can also specify the desired level of compression, to achieve the proper trade-off between disk usage and CPU overhead. Included is a compression advisor that helps you determine the proper compression levels for your application.

This feature allows the database to reduce the number of I/Os to scan a table. For example, if you compress data 10 to 1, then the I/Os are reduced 10 to 1 as well. In addition, Exadata Hybrid Columnar Compression saves disk space by the same amount.

This feature also allows the database to offload Smart Scans for a column-compressed table to Exadata Cells. When a scan is done on a compressed table, Exadata Cell reads the compressed blocks from the disks for the scan. Oracle Exadata Storage Server Software then decompresses the data, does predicate evaluation of the data, and applies the filter. The cell then sends back qualifying data in an uncompressed format. Without this offload, data decompression would take place on the database host. Having Exadata Cell decompress the data results in significant CPU savings on the database host.

See Also: "About Exadata Hybrid Columnar Compression" on page 7-30

Storage Index

Storage indexes are a very powerful capability provided in Oracle Exadata Storage Server Software that help avoid I/O operations. Oracle Exadata Storage Server Software creates and maintains a storage index in Exadata memory. The storage index keeps track of minimum and maximum values of columns for tables stored on that cell. This functionality is done transparently, and does not require any administration by the user.

When a query specifies a WHERE clause, Oracle Exadata Storage Server Software examines the storage index to determine if rows with the specified column value exist in the cell by comparing the column value to the minimum and maximum values maintained in the storage index. If the column value is outside the minimum and maximum range, then scan I/O for that query is avoided. Many SQL operations run dramatically faster because large numbers of I/O operations are automatically replaced by a few in-memory lookups. To minimize operational overhead, storage indexes are created and maintained transparently and automatically by Oracle Exadata Storage Server Software.

Storage indexes provide benefits for encrypted tablespaces. However, storage indexes do not maintain minimum and maximum values for encrypted columns.

Smart Scan of Encrypted Data

Oracle Exadata Storage Server Software offloads decryption, and performs Smart Scans on encrypted tablespaces and encrypted columns. While the earlier release of Oracle Exadata Storage Server Software fully supported encrypted tablespaces and encrypted columns, it did not benefit from Exadata offload processing. For encrypted tablespaces, Oracle Exadata Storage Server Software can decrypt blocks and return the decrypted blocks to the Oracle Database, or it can perform column projection which returns rows and columns. When Oracle Exadata Storage Server Software performs the decryption instead of the database there is significant CPU savings because CPU usage is offloaded to Exadata Cells.

Optimized Smart Scan

Oracle Exadata Storage Server Software detects resource bottlenecks on Oracle Exadata Storage Servers by monitoring CPU utilization. When a bottleneck is found, work is relocated to improve performance. Each Exadata Cell maintains the following statistics:

- Exadata Cell CPU usage and push-back rate snapshots for the last 30 minutes.
- Total number of 1MB blocks that had a push-back decision on.
- Number of blocks that have been pushed back to the database servers.
- The statistic `Total cpu passthru output IO size` in KB.

Interleaved Grid Disks

Space for grid disks can be allocated in an interleaved manner. Grid disks that use this type of space allocation are referred to as interleaved grid disks. This method attempts to equalize the performance of the grid disks residing on the same cell disk rather than having the grid disks that occupy the outer tracks getting better performance at the expense of the grid disks on the inner tracks.

A cell disk is divided into two equal parts, the outer half (upper portion) and the inner half (lower portion). When a new grid disk is created, half of the grid disk space is allocated on the outer half of the cell disk, and the other half of the grid disk space is allocated the inner half of the cell disk. The upper portion of the grid disk starts on the first available outermost offset in the outer half depending on free or used space in the outer half. The lower portion of the grid disk starts on the first available outermost offset in the inner half.

For example, if cell disk, `CD_01_cell01` is completely empty and has 100 GB of space, and a grid disk, `data_CD_01_cell01`, is created and sized to 50 GB on the cell disk, then the cell disk would have the following layout:

```
- Outer portion of data_CD_01_cell01 - 25GB
- Free space - 25GB
----------- Middle Point -----------------
- Inner portion of data_CD_cell01 - 25GB
```

```
- Free space - 25GB
```

See Also:

- "ALTER GRIDDISK" on page 8-22
- "CREATE CELLDISK" on page 8-39
- "LIST GRIDDISK" on page 8-107

Data Mining

Oracle Exadata Storage Server Software now offloads data mining model scoring. This makes the deployment of your data warehouse on Exadata Cells a better and more performant data analysis platform. All data mining scoring functions, such as `PREDICTION_PROBABILITY`, are offloaded to Exadata Cells for processing. This accelerates warehouse analysis while it reduces database server CPU consumption and the I/O load between the database server and Exadata storage.

Enhanced Manageability Features

Oracle Exadata Storage Server Software now includes the following manageability features:

- Automatic addition of replacement disk to the disk group: All the required Exadata operations to re-create the disk groups, and add the grid disks back to the original disk group are now performed automatically when a replacement disk is inserted after a physical disk failure.
- Automatic cell restart: Grid disks are automatically changed to online when a cell recovers from a failure, or after a restart.
- Support for OCR and voting disks on ASM disk groups: In Oracle Database 11*g* Release 2 (11.2), Oracle Cluster Registry (OCR) and voting disks are supported on ASM disk groups, and the iSCSI partitions are no longer needed.
- Support for up to four dual-port InfiniBand Host Channel Adapters in the database server. This feature enables larger servers to be used as database servers using Oracle Exadata Storage Server Software.

1

Introducing Oracle Exadata Storage Server Software

This chapter introduces Oracle Exadata Storage Server Software (Exadata Cell). This chapter contains the following topics:

Overview of Oracle Exadata Storage Server Software

Oracle Exadata Storage Server is a highly optimized storage server that runs Oracle Exadata Storage Server Software, also referred to as Exadata Cell, to store and access Oracle Database data. With traditional storage, data is transferred to the database server for processing. In contrast, Exadata Cell provides database-aware storage services, such as the ability to offload SQL and other database processing from the database server, while remaining transparent to the SQL processing and database applications. Exadata Cells process data at the storage level, and pass only what is needed to the database server. Oracle Exadata Storage Server can also be used in addition to traditional storage arrays and products.

Exadata Cell offloads some SQL processing from the database server to the Oracle Exadata Storage Servers. Exadata Cell enables function shipping between the database instance and the underlying storage, in addition to traditional data shipping. Function shipping greatly reduces the amount of data processing that must be done by the database server. Eliminating data transfers and database server workload can greatly benefit query processing operations that often become bandwidth constrained. Eliminating data transfers can also provide a significant benefit to online transaction processing (OLTP) systems that include large batch and report processing operations.

The hardware components of Oracle Exadata Storage Server are carefully chosen to match the needs of high performance processing. The cell software is optimized to maximize the advantage of the hardware components. Each cell delivers outstanding processing bandwidth for data stored on disk, often several times better than traditional solutions.

Oracle Exadata Storage Server uses a state-of-the-art InfiniBand interconnection between servers and storage. Each InfiniBand link provides 32 gigabits per second of bandwidth, many times higher than traditional storage or server networks. Additionally, Oracle interconnection protocol uses direct data placement, also referred to as direct memory access (DMA), to ensure low CPU overhead by directly moving data from the wire to database buffers with no extra copies. The InfiniBand network has the flexibility of a LAN network with the efficiency of a storage area network (SAN). With an InfiniBand network, Oracle eliminates network bottlenecks that could reduce performance. This InfiniBand network also provides a high performance cluster interconnection for the Oracle Real Application Clusters (Oracle RAC) servers.

The Exadata Cell architecture scales to any level of performance. To achieve higher performance or greater storage capacity, you add more storage cells to the configuration. As more cells are added, capacity and performance increase linearly. Data is mirrored across cells to ensure that the failure of a cell does not cause loss of data or availability. The scale-out architecture achieves near infinite scalability, while lowering costs by allowing storage to be purchased incrementally on demand.

Note: Oracle Exadata Storage Server Software must be used with Oracle Exadata Storage Server hardware, and only supports databases on the database servers of Oracle Exadata Database Machines. Information is available on My Oracle Support at

`http://support.oracle.com/`

and on the Products page of Oracle Technology Network at

`http://www.oracle.com/technology/index.html`

Key Features of Oracle Exadata Storage Server Software

The key features of Exadata Cell include the following:

- Reliability, Modularity, and Cost-Effectiveness
- Compatibility with Oracle Database
- Centralized Storage
- I/O Resource Management
- Offloading of Data Search and Retrieval Processing
- Offloading of Incremental Backup Processing
- Protection Against Data Corruption
- Fast File Creation
- Storage Index

Reliability, Modularity, and Cost-Effectiveness

Exadata Cell enables cost-effective modular storage hardware to be used in a scale-out architecture while providing a high level of availability and reliability. All single points of failure are eliminated in the Exadata Cell architecture by data mirroring, fault isolation technology, and protection against disk and other storage hardware failure.

In the Exadata Cell architecture, one or more storage cells can support one or more databases. The placement of data is transparent to database users and applications. Storage cells use **Oracle Automatic Storage Management (Oracle ASM)** to distribute data evenly across the cells. Because Exadata Cell supports dynamic disk insertion and removal, the online dynamic data redistribution feature of Oracle ASM ensures that data is appropriately balanced across the newly added, or remaining, disks without interrupting database processing. Exadata Cell also provides data protection from disk and cell failures.

Compatibility with Oracle Database

Oracle Exadata Storage Server Software 11*g* Release 2 (11.2) requires Oracle Database and Oracle ASM 11*g* Release 2 (11.2). All Oracle Database features are fully supported with Exadata Cell. Exadata Cell works equally well with single-instance or Oracle RAC deployments of Oracle Database. Oracle Data Guard, Oracle Recovery Manager (RMAN), Oracle Streams, and other database administration tools are managed the same with Oracle Exadata Storage Server as with traditional storage. This enables database administrators to use the same tools with which they are familiar.

Notes:

- Oracle Exadata Storage Server Software must be used with Oracle Exadata Storage Server hardware, and only supports databases on the database servers of Oracle Exadata Database Machines. Information is available on My Oracle Support at

  `http://support.oracle.com/`

  and on the Products page of Oracle Technology Network at

  `http://www.oracle.com/technology/index.html`

- Oracle Solaris Cluster is not supported on Oracle Database servers on Oracle Exadata Database Machine.

Centralized Storage

You can use Oracle Exadata Storage Server to consolidate your storage requirements into a central pool that can be used by multiple databases. Exadata Cell with Oracle ASM evenly distributes the data and I/O load for every database across available disks in the storage pool. Every database can use all of the available disks to achieve superior I/O rates. Exadata Cell can provide higher efficiency and performance at a lower cost while also lowering your storage administration overhead.

I/O Resource Management

I/O Resource Management (IORM) and the Database Resource Manager process enable multiple databases to share the same storage while ensuring that I/O resources are allocated across the various databases. Exadata Cell works with IORM and Database Resource Manager to ensure that customer-defined policies are met, even when multiple databases share the Exadata Cell grid. As a result, one database cannot monopolize the I/O bandwidth and degrade the performance of the other databases.

IORM enables storage cells to service disk I/O resources among multiple applications and users across all databases in accordance with sharing and prioritization levels established by the administrator. This improves the coexistence of online transaction processing (OLTP) and reporting workloads, because latency-sensitive OLTP applications can be given a larger share of disk I/O bandwidth than throughput-sensitive batch applications. Database Resource Manager enables the administrator to control processor utilization on the database host on a per-application basis. Combining IORM and Database Resource Manager enables the administrator to establish more accurate policies.

IORM for a database is implemented and managed from the Database Resource Manager. Database Resource Manager in the database instance communicates with the IORM software in the storage cell to manage user-defined service-level targets.

Database resource plans are administered from the database, while interdatabase plans are administered on the storage cell.

> **See Also:** Chapter 6, "Managing I/O Resources" for additional information about IORM

Offloading of Data Search and Retrieval Processing

One of the most powerful features of Exadata Cell is that it offloads the data search and retrieval processing to the storage cell. Exadata Cell does this by performing predicate filtering, which entails evaluating database predicates to optimize the performance of certain classes of bulk data processing.

Oracle Database can optimize the performance of queries that perform table and index scans to evaluate selective predicates in Exadata Cell. The database can complete these queries faster by pushing the database expression evaluations to the storage cell. These expressions include simple SQL command predicates, such as `amount > 200`, and column projections, such as `SELECT customer_name`. For example:

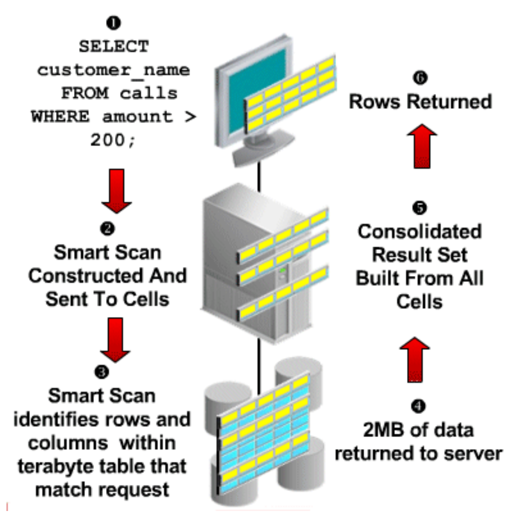
```
SQL> SELECT customer_name FROM calls WHERE amount > 200;
```

In the preceding example, only rows satisfying the predicate, specified columns, and predicated columns are returned to the database server, eliminating unproductive data transfer to the database server.

Exadata Cell uses storage-side predicate evaluation that transfers simplified, predicate evaluation operations for table and index scans to the storage cell. This brings the table scan closer to the disk to enable a higher bandwidth, and prevents sending unmatched rows to hosts.

Figure 1–1 shows an example of the query process.

Figure 1–1 Offloading Data Search and Retrieval



Offloading of Incremental Backup Processing

To optimize the performance of incremental backups, the database can offload block filtering to Oracle Exadata Storage Server. This optimization is only possible when taking backups using Oracle Recovery Manager (RMAN). The offload processing is done transparently without user intervention. During offload processing, Exadata Cell filters out the blocks that are not required for the incremental backup in progress.

Therefore, only the blocks that are required for the backup are sent to the database, making backups significantly faster.

See Also:

- "Using V$BACKUP_DATAFILE with Exadata Cell" on page 7-21
- *Oracle Database Backup and Recovery User's Guide*

Protection Against Data Corruption

Exadata Cell is compliant with the Oracle Hardware Assisted Resilient Data (HARD) initiative, a joint initiative between Oracle and hardware vendors to prevent data corruptions from being written to disks. Data corruptions, while rare, can have a catastrophic effect on a database, and therefore on a business. Exadata Cell takes data protection to the next level by protecting business data, not just the physical bits.

The key approach to detecting and preventing corrupted data is block checking in which the storage subsystem validates the Oracle block contents. Oracle Database validates and adds protection information to the database blocks, while Exadata Cell detects corruptions introduced into the I/O path between the database and storage. It stops corrupted data from being written to disk, and validates data when reading the disk. This eliminates a large class of failures that the database industry had previously been unable to prevent.

Exadata Cell implements all the HARD checks, and because of its tight integration with Oracle Database, additional checks are implemented that are specific to Exadata Cell. Unlike other implementations of HARD checking, HARD checks with Exadata Cell operate completely transparently. No parameters need to be set at the database or storage tier. The HARD checks transparently handle all cases, including Oracle ASM disk rebalance operations and disk failures.

Storing the Server Parameter File on HARD-enabled Storage

The server parameter file (SPFILE) is compliant with the HARD specifications implemented by Oracle Exadata Database Machine. To fully enable HARD protection for the data in the SPFILE, the SPFILE must reside on Oracle Exadata Storage Server.

The HARD-compliant SPFILE can be stored on non-HARD-enabled storage. In this case, the SPFILE format supports only detection of corrupt SPFILE data. Storing the SPFILE on HARD-enabled storage prevents corrupt data from being written to storage.

See Also:

- *Oracle Database High Availability Overview* for additional information about the Hardware Assisted Resilient Data (HARD) Initiative
- Oracle Maximum Availability Architecture (MAA) Web site for additional information about the Hardware Assisted Resilient Data (HARD) initiative at

  `http://www.oracle.com/technology/deploy/availability/htdocs/HARD.html`

Fast File Creation

File creation operations are offloaded to Exadata Cells. Operations such as `CREATE TABLESPACE`, which can create one or more files, have a significant increase in speed due to file creation offload.

Storage Index

Exadata Cells maintain a storage index which contains a summary of the data distribution on the disk. The storage index is maintained automatically, and is transparent to Oracle Database. It is a collection of in-memory region indexes, and each region index stores summaries for up to eight columns. There is one region index for each 1 MB of disk space.

The content stored in a region index is independent of other region indexes. This makes them highly scalable, and avoids latch contention. For each region index, the storage index maintains the minimum and maximum values of the columns of the table. The minimum and maximum values are used to eliminate unnecessary I/O, also known as I/O filtering. The cell physical IO bytes saved by storage index statistic, available in the `V$SYS_STAT` view, shows the number of bytes of I/O saved using storage index.

Queries using the following comparisons are improved by the storage index:

- Equality (=)
- Inequality (<, !=, or >)
- Less than or equal (<=)
- Greater than or equal (>=)
- IS NULL
- IS NOT NULL

Example 1–1 Queries on Range-partitioned Table

In this example, there is a table named `Orders` with the columns Order_Number, Order_Date, Ship_Date, or Order_Item. The table is range partitioned by Order_Date column. The following query looks for orders placed since January 1, 2010:

```
SELECT count (*) FROM Orders WHERE Order_Date >= to_date ('2010-01-01', \
'YYY-MM-DD')
```

Because the table is partitioned on the Order_Date column, the preceding query avoids scanning unnecessary partitions of the table.

Note: The storage index is maintained during write operations to uncompressed blocks and OLTP compressed blocks. Write operations to Exadata Hybrid Columnar Compression compressed blocks or encrypted tablespaces will invalidate a region index, but not the storage index.

See Also: "Using V$SYSSTAT with Exadata Cell" on page 7-22

Oracle Exadata Storage Server Software Components

This section provides a summary of the following Exadata Cell components:

About Oracle Exadata Storage Server Software

Oracle Exadata Storage Server is a network-accessible storage device with Oracle Exadata Storage Server Software installed on it. The software communicates with the database using a specialized *i*DB protocol, and provides both simple I/O functionality, such as block-oriented reads and writes, and advanced I/O functionality, including predicate offload and I/O resource management. Each storage cell has a physical disk. The physical disk is an actual device within the storage cell that constitutes a single disk drive spindle.

Within the storage cells, a logical unit number (LUN) defines a logical storage resource from which a single cell disk can be created. The LUN refers to the access point for storage resources presented by the underlying hardware to the upper software layers. The precise attributes of a LUN are configuration-specific. For example, a LUN could be striped, mirrored, or both striped and mirrored.

A cell disk is an Exadata Cell abstraction built on the top of a LUN. After a cell disk is created from the LUN, it is managed by Exadata Cell and can be further subdivided into grid disks, which are directly exposed to the database and Oracle ASM instances. Each grid disk is a potentially noncontiguous partition of the cell disk that is directly exposed to Oracle ASM to be used for the Oracle ASM disk group creations and expansions.

This level of virtualization enables multiple Oracle ASM clusters and multiple databases to share the same physical disk. This sharing provides optimal use of disk capacity and bandwidth. Various metrics and statistics collected on the cell disk level enable you to evaluate the performance and capacity of Exadata Cell. I/O Resource Management schedules the cell disk access in accordance with user-defined policies.

Figure 1–2 illustrates how the components of a cell are related to grid disks.

- A LUN is created from a physical disk.
- A cell disk is created on a LUN. A segment of cell disk storage is used by the Exadata Cell system.
- Multiple grid disks can be created on a cell disk.

Figure 1–2 Exadata Cell Storage Components

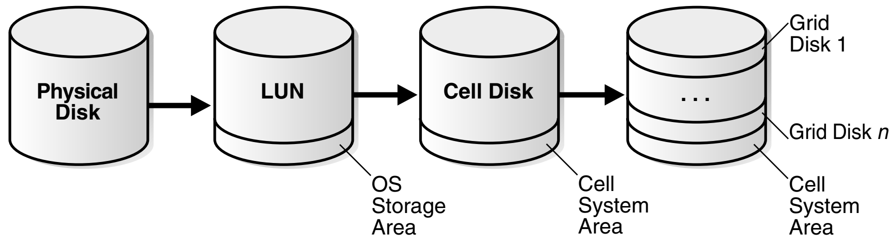
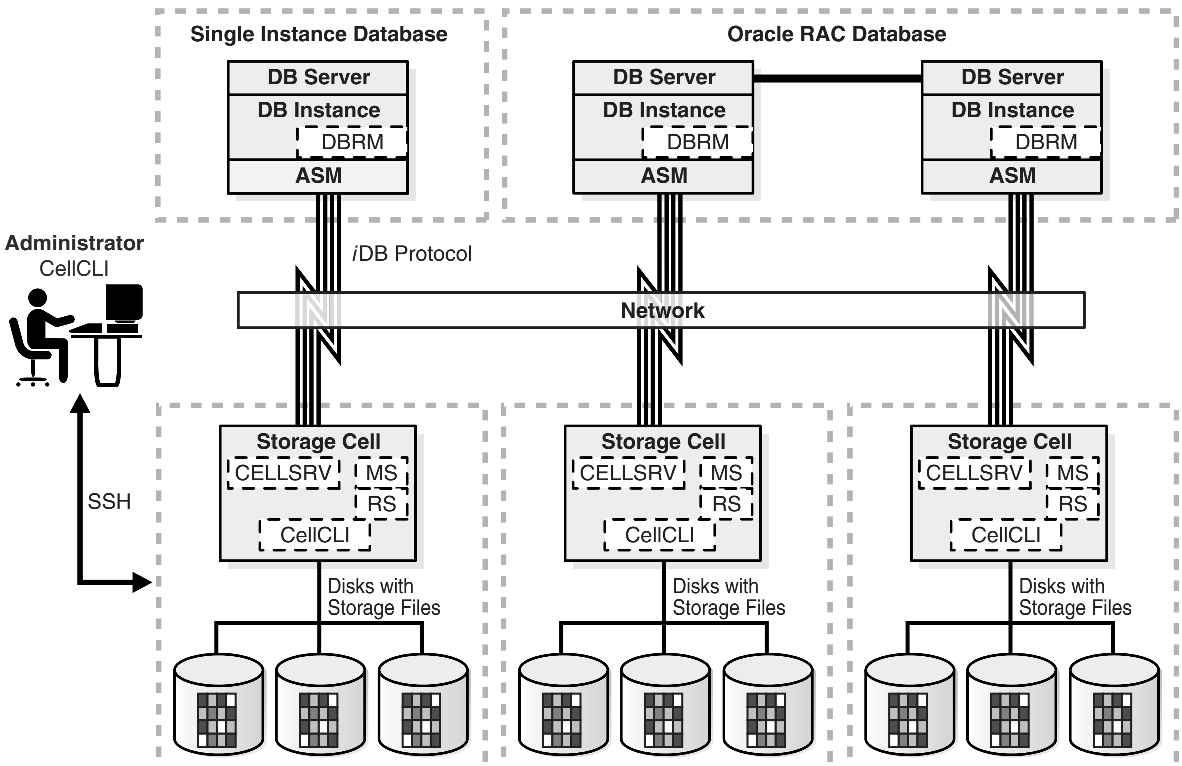


Figure 1–3 illustrates software components in the Exadata Cell environment.

Figure 1–3 Software Components in the Oracle Exadata Storage Server Environment



The figure illustrates the following environment:

- Single-instance or Oracle RAC databases access storage cells using the *i*DB protocol over an InfiniBand network.
- The database server software includes Exadata Cell functionality.
- Storage cells contain cell-based software.
- Storage cells are configured on the network, and are managed by the Exadata Cell CellCLI utility.

About Oracle Automatic Storage Management

Oracle Automatic Storage Management (Oracle ASM) is the cluster volume manager and file system used to manage Exadata Cell resources. Oracle ASM provides enhanced storage management by:

- Striping database files evenly across all available storage cells and disks for optimal performance.
- Using mirroring and failure groups to avoid any single point of failure.
- Enabling dynamic add and drop capability for nonintrusive cell and disk allocation, deallocation, and reallocation.
- Enabling multiple databases to share storage cells and disks.

> **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for additional information about Oracle ASM

Automatic Storage Management Disk Group

An Oracle ASM disk group is the primary storage abstraction within Oracle ASM, and is composed of one or more disks. Oracle Exadata Storage Server grid disks appear to Oracle ASM as individual disks available for membership in Oracle ASM disk groups. Whenever possible, grid disk names should correspond closely with Oracle ASM disk group names to assist in problem diagnosis between Oracle ASM and Exadata Cell.

To take advantage of Exadata Cell features, such as predicate processing offload, the disk groups must contain only Exadata Cell grid disks, and the tables must be fully inside these disk groups.

> **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for additional information about Oracle ASM

Automatic Storage Management Failure Group

An Oracle ASM failure group is a subset of disks in an Oracle ASM disk group that can fail together because they share the same hardware. Oracle ASM considers failure groups when making redundancy decisions.

For Exadata Cell, all grid disks, which consist of the Oracle ASM disk group members and candidates, can effectively fail together if the storage cell fails. Because of this scenario, all Oracle ASM grid disks sourced from a given storage cell should be assigned to a single failure group representing the cell.

For example, if all grid disks from two storage cells, A and B, are added to a single Oracle ASM disk group with normal redundancy, then all grid disks on storage cell A are designated as one failure group, and all grid disks on storage cell B are designated as another failure group. This enables Exadata Cell and Oracle ASM to tolerate the failure of either storage cell.

Failure groups for Exadata Cell grid disks are set by default so that the disks on a single cell are in the same failure group, making correct failure group configuration simple for Exadata Cell.

You can define the redundancy level for an Oracle ASM disk group when creating a disk group. An Oracle ASM disk group can be specified with normal or high redundancy. Normal redundancy double mirrors the extents, and high redundancy triple mirrors the extents. Oracle ASM normal redundancy tolerates the failure of a single cell or any set of disks in a single cell. Oracle ASM high redundancy tolerates the failure of two cells or any set of disks in two cells. Base your redundancy setting on your desired protection level. When choosing the redundancy level, ensure the post-failure I/O capacity is sufficient to meet the redundancy requirements and performance service levels. Oracle recommends using three cells for normal redundancy. This ensures the ability to restore full redundancy after cell failure. Consider the following:

- If a cell or disk fails, then Oracle ASM automatically redistributes the cell or disk contents across the remaining disks in the disk group as long as there is enough space to hold the data. For an existing disk group using Oracle ASM redundancy, the `USABLE_FILE_MB` and `REQUIRED_FREE_MIRROR_MB` columns in the `V$ASM_DISGKROUP` view give the amount of usable space and space for redundancy, respectively.
- If a cell or disk fails, then the remaining disks should be able to generate the MBPS and IOPS necessary to sustain the performance service level agreement.

After a disk group is created, the redundancy level of the disk group cannot be changed. To change the redundancy of a disk group, you must create another disk group with the appropriate redundancy, and then move the files.

Each Exadata Cell is a failure group. A normal redundancy disk group must contain at least two failure groups. Oracle ASM automatically stores two copies of the file extents, with the mirrored extents placed in different failure groups. A high redundancy disk group must contain at least three failure groups. Oracle ASM automatically stores three copies of the file extents, with each file extent in separate failure groups.

System reliability can diminish if your environment has an insufficient number of failure groups. A small number of failure groups, or failure groups of uneven capacity, can lead to allocation problems that prevent full use of all available storage.

> **See Also:** Example 3–1, "Creating Oracle ASM Disk Groups for Exadata Cell" on page 3-5 for an example of creating an Oracle ASM disk group

About Oracle ASM for Maximum Availability

Oracle recommends high redundancy Oracle ASM disk groups, and file placement configuration which can be automatically deployed using the Oracle OneCommand utility. Maximum availability architecture (MAA) best practice uses three Oracle ASM disk groups, DATA, RECO, and Oracle Database File System (DBFS). The disk groups are located as follows:

- The disk groups are striped across all disks and Oracle Exadata Storage Servers to maximize I/O bandwidth and performance, and simplify management.
- The DATA disk group is located on the outer section of all disks.
- The RECO disk group is located on the outer/inner section of all disks.
- The DBFS disk group is located on the inner section of all disks.
- The DATA and RECO disk groups are configured for high redundancy.

The preceding attributes ensure optimal file placement in the different Oracle ASM disk groups. In addition, all operations have access to full I/O bandwidth, when needed. To avoid excessive resource consumption, use I/O Resource Management, Oracle Database Resource Manager, and instance caging.

The following are some of the benefits of a high redundancy disk group, also known as 3-way mirroring:

- Protection against loss of the cluster and Oracle ASM disk group due to concurrent disk failures containing the primary and secondary extents, such as two partner disk failures.

- Protection against loss of the cluster and Oracle ASM disk group when Oracle Exadata Storage Server is taken offline during planned maintenance and a concurrent partner disk failure occurs.
- Protection against data loss when latent block corruptions exist and a partner storage disk is unavailable either due to planned maintenance or disk failure.

Having both the DATA and RECO disk groups high redundancy provides maximum application availability against storage failures. When only the DATA disk group is high redundancy, then the applications remain available only if an alternative archive destination is configured. When only the RECO disk group is high redundancy, then the database can be restored and recovered with zero data loss, but recovery time can be very long. Other than better storage protection, the major difference between high redundancy and normal redundancy is the amount of usable storage, and write I/Os. High redundancy requires more space, and has three write I/Os instead of two for each physical write.

Oracle recommends creating at least one high redundancy disk group, and locate database files there to meet high availability requirements. The following table describes the disk group configurations.

Redundancy Option	Description	Recommendation
High Redundancy for ALL (DATA and RECO)	Zero application downtime for disk failures and latent block corruption.	Best storage protection for mission-critical applications. Requires more space. This is considered MAA. To increase usable space for DATA, use external backups only. External backups consist of tape-based backups or disk backups residing on an external system.
High Redundancy for DATA	DATA is high redundancy, RECO and DBFS are normal redundancy. Zero application downtime for disk-level failures, but requires an alternative archive destination.	Internal disk-based backups are to preconfigured RECO disk group. If only external backup is used, then use High Redundancy for ALL because the usable space difference is minimal.
High Redundancy for LOG and RECO	DATA and DBFS_DG are normal redundancy. RECO is high redundancy. Zero data loss.	This option provides the most usable space for DATA disk group while preserving the potential of zero data loss recovery. This is recommended for external backups only.

The following table describes the optimal file placement for setup for MAA:

File Type	Location
Oracle Database files	DATA disk group
Flashback log files, archived redo files, and backup files	RECO disk group
Redo log files, and control files	First high redundancy disk group

File Type	Location
Server parameter files (SPFILE)	First high redundancy disk group
Oracle Cluster Registry (OCR) and voting disks for Oracle Exadata Database Machine Full Rack and Oracle Exadata Database Machine Half Rack	First high redundancy disk group
OCR and voting disks for Oracle Exadata Database Machine Quarter Rack	Normal redundancy disk group[1]
Temporary files	First normal redundancy disk group, but not DBFS_DG
Staging and non-database files	DBFS disk group

[1] To place voting disks and Oracle ASM metadata files in a high redundancy Oracle ASM disk group, a minimum of five Exadata Cells are needed.

See Also:

- *Oracle Exadata Database Machine Owner's Guide*
- *Oracle Exadata Database Machine Configuration Worksheets*

About Grid RAID

A grid Redundant Array of Independent Disks (RAID) configuration uses Oracle ASM mirroring capabilities. To use grid RAID, you place grid disks in an Oracle ASM disk group with a normal or high redundancy level, and set all grid disks in the same cell to be in the same Oracle ASM failure group. This ensures that Oracle ASM does not mirror data extents using disks within the cell. Using disks from different cells ensures that an individual cell failure does not cause the data to be unavailable.

Grid RAID also provides simplified creation of cell disks. With grid RAID, LUNs are automatically created from available physical disks because Oracle software automatically creates the required LUNs.

About Cell Security

Security for Exadata Cell is enforced by identifying which clients can access cells and grid disks. Clients include Oracle ASM instances, database instances, and clusters. When creating or modifying grid disks, you can configure the Oracle ASM owner and the database clients that are allowed to use those grid disks.

About iDB Protocol

The *i*DB protocol is a unique Oracle data transfer protocol that serves as the communications protocol among Oracle ASM, database instances, and storage cells. General-purpose data transfer protocols operate only on the low-level blocks of a disk. In contrast, the *i*DB protocol is aware of the Oracle internal data representation and is the necessary complement to Exadata Cell-specific features, such as predicate processing offload.

In addition, the *i*DB protocol provides interconnection bandwidth aggregation and failover.

About Cell Software Processes

Exadata Cell includes the following software processes:

- **Cell Server (CELLSRV)** services *i*DB requests for disk I/O and advanced Exadata Cell services, such as predicate processing offload. CELLSRV is implemented as a multithread process and should be expected to use the largest portion of processor cycles on a storage cell.
- **Management Server (MS)** provides standalone storage cell management and configuration.
- **Restart Server (RS)** monitors the Cell Server and Management Server processes and restarts them, if necessary.

About Cell Management

Each cell in the Exadata Cell grid is individually managed with Cell Control Command-Line Interface (CellCLI).The CellCLI utility provides a command-line interface to the cell management functions, such as cell initial configuration, cell disk and grid disk creation, and performance monitoring. The CellCLI utility runs on the cell, and is accessible from a client computer that has network access to the storage cell or is directly connected to the cell. The CellCLI utility communicates with Management Server to administer the storage cell.

To access the cell, you should either use Secure Shell (**SSH**) access, or local access, for example, through a **KVM switch** (keyboard, video or visual display unit, mouse) switch. SSH allows remote access, but local access might be necessary during the initial configuration when the cell is not yet configured for the network. With local access, you have access to the cell operating system shell prompt and use various tools, such as the CellCLI utility, to administer the cell.

You can run the same CellCLI commands remotely on multiple cells with the dcli utility.

See Also:

- Chapter 8, "Using the CellCLI Utility" for additional information about CellCLI commands
- Chapter 9, "Using the dcli Utility" for additional information about managing multiple cells with a centralized management tool

About Database Server Software

Exadata Cell works seamlessly with Oracle Database. The software on the database server includes:

- Oracle Database instance, which contains the set of Oracle Database background processes that operate on the stored data and the shared allocated memory that those processes use to do their work.
- Oracle Automatic Storage Management (Oracle ASM), which provides storage management optimized for the database and Exadata Cell. Oracle ASM is part of Oracle Grid Infrastructure.

  The Oracle ASM instance handles placement of data files on disks, operating as a metadata manager. The Oracle ASM instance is primarily active during file creation and extension, or during disk rebalancing following a configuration change. Run-time I/O operations are sent directly from the database to storage cells without passing through an Oracle ASM instance.
- The Database Resource Manager (DBRM), which ensures that I/O resources are properly allocated within a database.

- The *i*DB protocol is used by the database instance to communicate with cells, and is implemented in an Oracle-supplied library statically linked with the database server.

See Also:

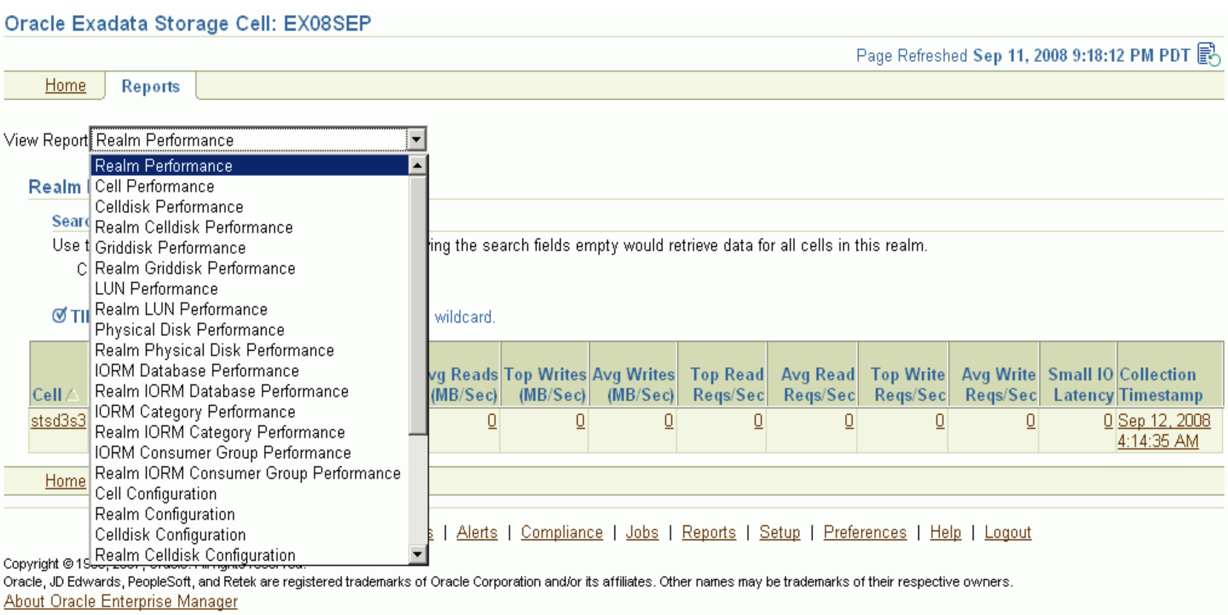
- Chapter 6, "Managing I/O Resources" for additional information about DBRM
- "About iDB Protocol" on page 1-12 for additional information about iDB protocol

About Oracle Enterprise Manager Oracle Exadata Storage Server Plug-in

Oracle Enterprise Manager provides a plug-in that enables you to monitor Exadata Cell configuration and performance in a GUI display.

Figure 1–4 shows the Exadata Cell Performance report with the menu of configuration and performance reports displayed. With the menu, you can select both configuration and performance reports for Exadata Cell objects such as cell, realm, cell disk, grid disk, LUN, physical disk, and IORM. The format of the report varies according to the type of report selected.

Figure 1–4 Exadata Cell Plug-in Cell Performance Report



In addition to reports, Oracle Enterprise Manager plug-in for Exadata Cell enables you to set metric thresholds for alerts and monitor metric values to determine the health of a storage cell.

Oracle Enterprise Manager plug-in for Exadata Cell is available to download from the Oracle Technology Network (OTN) at

`http://www.oracle.com/technology/software/products/oem/index.html`

You must register online before using OTN. Registration is free and can be done at

`http://www.oracle.com/technology/membership/`

2

Configuring Oracle Exadata Storage Server Software

This chapter describes the major steps to configure a small Oracle Exadata Storage Server Software (Exadata Cell) grid. The steps are the same for a larger grid. You determine the number of disks and cells needed in your Exadata Cell grid based on your requirements for capacity, performance, and redundancy.

Hardware and software have already been installed for the cells. The procedures in this chapter describe how to configure a storage cell for use with the Oracle Database and Oracle Automatic Storage Management (Oracle ASM) instances.

Note: Modifications to the Oracle Exadata Storage Server hardware or software are not supported. Only the documented network interfaces on the Oracle Exadata Storage Server should be used for all connectivity including management and storage traffic. Additional network interfaces should not be used.

This chapter contains the following topics:

- Understanding Oracle Exadata Storage Server Software Release Numbering
- Understanding Oracle Exadata Storage Server Configuration
- Network Configuration and IP Addresses Recommendations
- Assigning IP Addresses for Oracle Exadata Storage Server Software
- Configuring Oracle Exadata Storage Server Software for Your Location
- Configuring Cells, Cell Disks, and Grid Disks with CellCLI
- Creating Flash Cache and Flash Grid Disks
- Understanding Automated Cell Maintenance

See Also:

- *Oracle Exadata Storage Server Software Release Notes* for the latest information about installation and configuration
- *Oracle Exadata Database Machine Owner's Guide* for information about the Oracle Exadata Database Machine Configurator Spreadsheet
- "Optimizing Performance" on page 7-29 for additional information about configuring for performance
- Appendix C, "Installation Information" for additional information about installing Exadata Cell hardware and software

Understanding Oracle Exadata Storage Server Software Release Numbering

The Oracle Exadata Storage Server Software release number is related to the Oracle Database release number, as follows:

- The first two digits of the Oracle Exadata Storage Server Software release number represent the major Oracle Database release number, such as Oracle Database 11*g* Release 2 (11.2). Oracle Exadata Storage Server Software release 11.2 is compatible with all Oracle Database 11*g* Release 2 (11.2) releases.
- The third digit represents the component-specific Oracle Database release number. This digit matches the fourth digit of the complete release number, such as 11.2.0.1.0 for the current release of Oracle Database.
- The fourth digit represents the major Oracle Exadata Storage Server Software release number.
- The fifth digit is the Oracle Exadata Storage Server Software patch set release.

See Also: *Oracle Database Administrator's Guide* for information about the database release number

Understanding Oracle Exadata Storage Server Configuration

Oracle Exadata Storage Server ships with all hardware and software pre-installed, however it is necessary to configure Exadata Cell for your environment. This section provides a general overview of the configuration tasks. Subsequent sections describe the actual procedures. This section includes the following topics:

- Task 1, "Assign IP Addresses for the Storage Cell"
- Task 2, "Configure the Storage Cell for Your Location"
- Task 3, "Configure the Cell"
- Task 4, "Verify Cell Attributes"
- Task 5, "Create the Cell Disks"
- Task 6, "Create the Grid Disks"
- Task 7, "Create the Flash Disks and Flash Cache"
- Task 8, "Configure Auto Service Request"

Task 1 Assign IP Addresses for the Storage Cell

Assign IP addresses for the storage cell for the following ports:

- Network access port
- Remote management port
- InfiniBand port

Task 2 Configure the Storage Cell for Your Location

Power on the storage cell and configure it for your location, such as setting the time zone and passwords.

Task 3 Configure the Cell

Use the `ALTER CELL` command to configure the cell. Example 2–1 shows an example of configuring a new cell.

Example 2–1 Configuring a New Cell

```
CellCLI> ALTER CELL realmName=my_realm,                                      -
         smtpServer='my_mail.example.com',                                   -
         smtpFromAddr='john.doe@example.com',                                -
         smtpFromPwd=email_address_password,                                 -
         smtpToAddr='jane.smith@example.com',                                -
         notificationPolicy='critical,warning,clear',                        -
         notificationMethod='mail,snmp'
```

In Example 2–1, e-mail notification is configured to send e-mail messages to the administrator of the storage cell. The hyphen (-) at the end of each line of the `ALTER CELL` command allows the command to continue to additional lines before pressing Enter. As an alternative, you can run the command using a text file.

Task 4 Verify Cell Attributes

Use the `LIST CELL DETAIL` command to verify the cell attributes. Example 2–2 shows an example of viewing the cell attributes.

Example 2–2 Viewing Storage Cell Details

```
CellCLI> LIST CELL DETAIL
        name:                   cell01
        bmcType:                IPMI
        cellVersion:            OSS_11.2.1.1.0_LINUX.X64_090815
        interconnectCount:      3
        iormBoost               0.0
        ipaddress1:             192.168.50.27/24
        cpuCount:               16
        fanCount:               12/12
        fanStatus:              normal
        kernelVersion:          2.6.18-128.1.16.0.1.el5
        makeModel:              SUN MICROSYSTEMS SUN FIRE X4275 SERVER SAS
        metricHistoryDays:      7
        offloadEfficiency:      1.0
        powerCount:             2/2
        powerStatus:            normal
        status:                 online
        smtpFrom:               "John Doe"
        smtpFromAddr:           john.doe@example.com
        smtpServer:             my_mail.example.com
        smtpToAddr:             jane.smith@example.com
```

```
snmpSubscriber:         host=host1,port=162,community=public
                        host=host2,port=162,community=public
temperatureReading:     24.0
temperatureStatus:      normal
upTime:                 0 days, 11:14
cellsrvStatus:          running
rsStatus:               running
msStatus:               running
```

Task 5 Create the Cell Disks

Use the CREATE CELLDISK command to create the cell disks. In Example 2–3, the ALL option creates all the cell disks using the default names.

Example 2–3 Creating Cell Disks

```
CellCLI> CREATE CELLDISK ALL
CellDisk CD_00_cell01 successfully created
CellDisk CD_01_cell01 successfully created
CellDisk CD_02_cell01 successfully created
...
CellDisk CD_10_cell01 successfully created
CellDisk CD_11_cell01 successfully created
```

The cell disks are created with names in the form CD_*lunID_cellname*. The lunID and cell name values correspond to the id attribute of the LUN and name attribute of the cell. You can specify other disk names if you create single cell disks.

On Oracle Exadata Storage Server with flash disks, the CREATE CELLDISK ALL command also creates cell disks on the flash disks, as shown in the following:

```
CellCLI> CREATE CELLDISK ALL
CellDisk FD_01_cell01 successfully created
CellDisk FD_02_cell01 successfully created
CellDisk FD_03_cell01 successfully created
CellDisk FD_04_cell01 successfully created
CellDisk FD_05_cell01 successfully created
CellDisk FD_06_cell01 successfully created
CellDisk FD_07_cell01 successfully created
CellDisk FD_08_cell01 successfully created
CellDisk FD_09_cell01 successfully created
CellDisk FD_10_cell01 successfully created
CellDisk FD_11_cell01 successfully created
CellDisk FD_12_cell01 successfully created
CellDisk FD_13_cell01 successfully created
CellDisk FD_14_cell01 successfully created
CellDisk FD_15_cell01 successfully created
```

Note: The CREATE CELLDISK command creates cell disks on flash disks if they do not currently exist. If there are cell disks on the flash disks, then they are not created again.

Task 6 Create the Grid Disks

Use the CREATE GRIDDISK command to create the grid disks. In the following example, the ALL HARDDISK PREFIX option creates one grid disk on each cell disk of the storage cell. The Oracle ASM disk group name is used with PREFIX to identify which grid disk belongs to the disk group. Example 2–4 shows an example of creating grid disks. The size of the disks depends on customer requirements.

Example 2–4 Creating Grid Disks

```
CellCLI> CREATE GRIDDISK ALL HARDDISK PREFIX=data, size=300G
GridDisk data_CD_00_cell01 successfully created
GridDisk data_CD_01_cell01 successfully created
GridDisk data_CD_02_cell01 successfully created
...
GridDisk data_CD_11_cell01 successfully created

CellCLI> CREATE GRIDDISK ALL HARDDISK PREFIX=reco, size=600G
GridDisk reco_CD_00_cell01 successfully created
GridDisk reco_CD_01_cell01 successfully created
GridDisk reco_CD_02_cell01 successfully created
...
GridDisk reco_CD_11_cell01 successfully created

CellCLI> LIST GRIDDISK
         data_CD_00_cell01       active
         data_CD_01_cell01       active
         data_CD_02_cell01       active
         ...
         data_CD_11_cell01       active

         reco_CD_00_cell01      active
         reco_CD_01_cell01      active
         reco_CD_02_cell01      active
         ...
         reco_CD_11_cell01      active
```

In Example 2–4, prefix values `data` and `reco` are the names of the Oracle ASM disk groups that are created. The `LIST GRIDDISK` command shows the grid disks that are created. When the `ALL PREFIX` option is used, the generated grid disk names are composed of the grid disk prefix followed by an underscore (_) and then the cell disk name.

Grid disk names must be unique across all cells within a single deployment. By following the recommended naming conventions for naming the grid and cell disks you automatically get unique names. If you do not use the default generated name when creating grid disks, then you must ensure that the grid disk name is unique across all storage cells. If the disk name is not unique, then it might not be possible to add the grid disk to an Oracle ASM disk group.

When creating a grid disk:

- You do not have to specify the `size` attribute. The maximum size possible is automatically chosen if the `size` attribute is omitted.
- Offset determines the position on the disk where the grid disk is allocated. The outermost tracks have lower offset values, and these tracks have greater speed and higher bandwidth. Offset can be explicitly specified to create grid disks that are relatively higher performing than other grid disks. If offset is not specified, then the best (warmest) available offset is chosen automatically in chronological order of grid disk creation. You should first create those grid disks expected to contain the most frequently accessed (hottest) data, and then create the grid disks that will contain the relatively colder data.

  When using the interleave option, the creation of a grid disk with an offset creates half of the grid disk at the start of the outermost tracks at the offset specified, and the other half of the grid disk at the same offset from the start of the innermost tracks.

Task 7 Create the Flash Disks and Flash Cache

Use the `CREATE GRIDDISK ALL FLASHDISK PREFIX='FLASH'` and `CREATE FLASHCACHE` commands to create the flash disks and flash cache.

By default, the `CREATE CELL` command creates flash cell disks on all flash disks, and then creates Exadata Smart Flash Cache on the flash cell disks. To change the size of the Exadata Smart Flash Cache or create flash grid disks it is necessary to remove the flash cache, and then create the flash cache with a different size, or create the flash grid disks.

Task 8 Configure Auto Service Request

Auto Service Request (ASR) support for Oracle Exadata Database Machine automatically creates service requests by detecting common hardware faults. This support covers selected components, such as disks and flash cards, in Exadata Cells and Oracle Database servers.

> **See Also:** *Oracle Exadata Database Machine Owner's Guide* for information on configuring ASR

Network Configuration and IP Addresses Recommendations

The following are recommendations for the network configuration and IP addresses.

- If your network is not already configured, then set up a fault-tolerant, private network subnet for Exadata Cell and database server hosts with multiple switches to eliminate the switch as a single point of failure. If all the interconnections in the Exadata Cell network are connected through a single switch, then that switch can be a single point of failure.

  If you are using a managed switch, then ensure that the switch VLAN configuration isolates Exadata Cell network traffic from all other network traffic.

- Allocate a block of IP addresses for the Oracle Exadata Storage Server general administration and the Lights Out (LO) remote management interfaces. Typically, these interfaces are on the same subnet, and may share the subnet with other hosts. For example, on the 192.168.200.0/24 subnet, you could assign the block of IP addresses between 192.168.200.31 and 192.168.200.100 for the Oracle Exadata Storage Server general administration and LO remote management interfaces. Other hosts sharing the subnet would be allocated IP addresses outside the block. The general administration and LO remote management interfaces can be placed on separate subnets, but this is not required.

  Do not allocate addresses that end in .0, .1, or .255, or those that would be used as broadcast addresses for the specific netmask that you have selected. For example, avoid addresses such as 192.168.200.0, 192.168.200.1, and 192.168.200.255.

  The following is a sample of four non-overlapping blocks of addresses. One set of addresses should be assigned to the normal Gigabit Ethernet interface/port for cells. The other may be assigned for the LO remote management port for the cells. The third set can be used for the database server Gigabit Ethernet port, and the fourth for the database server LO remote management port.

  ```
  192.168.200.0/21 (netmask 255.255.248.0)
  192.168.208.0/21 (netmask 255.255.248.0)
  192.168.216.0/21 (netmask 255.255.248.0)
  192.168.224.0/21 (netmask 255.255.248.0)
  ```

The InfiniBand network should be a private network for use by the database server hosts and Exadata Cells, and can have private local network addresses. These addresses must also be allocated in non-overlapping blocks.

The following example has 2 blocks of local InfiniBand addresses. Both the database server InfiniBand and the cell InfiniBand must be on the same subnet in order to communicate with each other. With bonding, only one subnet is necessary for InfiniBand addresses.

```
192.168.50.0/24 (netmask 255.255.255.0)
192.168.51.0/24 (netmask 255.255.255.0)
```

The preceding subnet blocks do not conflict with each other, and do not conflict with the current allocation to any of the hosts. It is a good practice to allocate the subnet blocks so that they have the identical netmask, which helps to simplify network administration.

Note: For Exadata Cell, the maximum allowed number of hosts in an InfiniBand network is 4096. Therefore, the network prefix value for the Infiniband network must be equal to or greater than 20. This means the netmask must be between 255.255.240.0 and 255.255.255.254 both inclusive.

You can determine the network prefix value for a given host IP address and its netmask using the ipcalc utility on any Linux machine, as follows:

```
ipcalc <host ip address such as 192.168.50.10 > -m
       <netmask for the host ip address such as 255.255.240.0>  -p
```

Ensure the network allows for future expansion. For example, 255.255.255.254 is valid network (prefix /31) but it only allows 1 host.

- If a domain name system (DNS) is required, then set up your DNS to help reference cells and interconnections. Exadata Cells do not require DNS. However, if DNS is required, then set up your DNS with the appropriate IP address and host name of Exadata Cell.
- The InfiniBand network should be used for network and storage communication when using Oracle Clusterware. Use the following command to verify the private network for Oracle Clusterware communication is using InfiniBand:

  ```
  oifcfg getif -type cluster_interconnect
  ```

- The Reliable Data Socket (RDS) protocol should be used over the InfiniBand network for database server to cell communication and Oracle Real Application Clusters (Oracle RAC) communication. Check the alert log to verify the private network for Oracle RAC is running the RDS protocol over the InfiniBand network. The following message should be in the log:

  ```
  cluster interconnect IPC version: Oracle RDS/IP (generic)
  ```

  If the RDS protocol is not being used over the InfiniBand network, then perform the following procedure:

  1. Shut down any processes that are using the Oracle binary.
  2. Change to the `ORACLE_HOME/rdbms/lib` directory.
  3. Run the following command:

```
make -f ins_rdbms.mk ipc_rds ioracle
```

Note: If a separate Oracle home is used for Oracle ASM and the database, then RDS should be enabled for both of them.

Assigning IP Addresses for Oracle Exadata Storage Server Software

This section summarizes the Exadata Cell network preparation before installing the new storage cell.

Each storage cell contains the following network ports:

- One dual-port InfiniBand card

  Exadata Cell is designed to be connected to two separate InfiniBand switches for high availability. The dual port card is only for availability. Each port of the InfiniBand card is capable of transferring the full data bandwidth generated by the storage cell. The loss of one network connection does not impact the performance of the storage cell.

- Gigabit Ethernet ports for normal network access, depending on the platform
  - Oracle Exadata Storage Server comes with four Gigabit Ethernet ports. However, only connect one port to a switch, and configure it for network access.
  - HP Oracle Exadata Storage Server comes with one Gigabit Ethernet port.
- One Gigabit Ethernet port is exposed by the Baseboard Management Controller (BMC), or Management Controller (MC) on Exadata Cell. This port is used for Lights Out (LO) remote management.
  - Oracle Exadata Storage Server uses Sun Integrated Lights Out Manager (ILOM) for remote management.
  - HP Oracle Exadata Storage Server uses HP LO100 for remote management.

To prepare the Exadata Cell network, you must perform the following procedure:

1. Assign one address to the bonded InfiniBand port. When you first set up the cell, you are prompted for the `BONDIB0` configuration information. This information is used automatically during the `CREATE CELL` command on first boot, and provides the data path for communication between the cell and the database servers.

   Note: To change the `BONDIB0` address after initial configuration, use the following command:

   ```
   CREATE CELL interconnect1=BONDIB0, [ipaddress1="new_ipaddress"]
   ```

   In the preceding command, use the `ipaddress` syntax to modify IP address.

   Oracle recommends that this InfiniBand network be a private network.

2. Assign an IP address to the cell for network access.
3. Assign an IP address to the cell for LO remote management.

You can access the remote management functionality with a Java-enabled Web browser at the assigned IP address.

See Also:

- `http://docs.sun.com/app/docs/coll/ilom3.0?l=en` for additional information about SUN ILOM
- `http://bizsupport.austin.hp.com/bc/docs/support/SupportManual/c01103956/c01103956.pdf` for additional information about HP LO100

Configuring Oracle Exadata Storage Server Software for Your Location

This section describes the storage cell configuration, and contains the following topics:

- Configuring LO Remote Management With Static IP for Oracle Exadata Storage Servers
- Preparing the Servers

Configuring LO Remote Management With Static IP for Oracle Exadata Storage Servers

This section describes how to configure lights out (LO) remote management with static IP for Exadata Cell. It contains the following topics:

- Configuring LO Remote Management for Exadata Cell on Oracle Exadata Storage Server
- Configuring LO Remote Management for Exadata Cell on HP Oracle Exadata Storage Server

Configuring LO Remote Management for Exadata Cell on Oracle Exadata Storage Server

Basic LO remote management configuration is done during the first boot. Refer to "Preparing the Servers" on page 2-10 for LO remote management configuration information.

Caution: Do not enable the sideband management available in ILOM. Doing so will disable all the SNMP agent reporting and monitoring functionality for the server.

Configuring LO Remote Management for Exadata Cell on HP Oracle Exadata Storage Server

The following procedure describes how to configure static IP addresses for LO remote management on an Exadata Cell on HP Oracle Exadata Storage Server:

1. Press **F10** during the power-on self test (POST) to enter the BIOS setup utility.
2. Press the right arrow to go to the Advanced menu.
3. Press the down arrow to scroll to IPMI.
4. Press **Enter**.
5. Perform the following steps to set the network BIOS settings on the cell:
   a. Press the down arrow to scroll to the LAN Configuration menu.

b. Press **Enter**.

c. Select **Disabled** on DHCP IP Source.

d. Press the down arrow to scroll, and enter a valid IP address, subnet mask, and gateway address. Press tab or period (.) to move between address fields.

6. Press **F10** to save and exit.

LO remote management functionality on the database server hosts on the HP Oracle Exadata Storage Server is provided by the iLO2 bundle. See the instructions at the following URL:

`http://h20000.www2.hp.com/bc/docs/support/SupportManual/c00553302/c00553302.pdf`

Preparing the Servers

The following procedure describes how to prepare the database servers and Oracle Exadata Storage Servers for use:

1. Configure LO remote management as described in "Configuring LO Remote Management With Static IP for Oracle Exadata Storage Servers" on page 2-9.
2. Power on the storage cell to boot its operating system.
3. Respond to the prompts to configure the system, after the storage cell boots.

   Press `y` to confirm, or `n` to retry or terminate when you are prompted for a yes or no response during the configuration steps. The yes or no prompt shows the default choice in brackets. If you do not enter a response, then the default choice is selected when you press Enter.
4. Check the network connections.

   The list of all discovered interfaces displays, and you are prompted to check the cables for those interfaces that do not have an active network cable connection. You can retry the configuration steps after connecting the cables, or ignore the unconnected interfaces. Only connected interfaces can be configured.
5. Enter the DNS server IP addresses, if needed. A DNS is not needed for a standalone, private storage environment.
6. Enter the time preference.
   - Choose the local time region number from the displayed list of available time regions.
   - Choose the location within the time region number from the displayed list of locations.
7. Enter the Network Time Protocol (NTP) servers.

   These servers are required to maintain the time on the system correctly, and are synchronized to your local time source.
8. Enter the Ethernet addresses, InfiniBand IP addresses and interfaces. A list of all Ethernet and InfiniBand interfaces that have an active network connection is displayed with the name of the interface on the extreme left. The InfiniBand interface is named `BONDIB0` and uses bonding between physical InfiniBand interfaces ib0 and ib1. Bonding provides the ability to transparently fail over from ib0 to ib1 or from ib1 to ib0 if connectivity is lost to ib0 or ib1, respectively.

   For each Ethernet and InfiniBand interface you configure, you are prompted for the following that apply to the interface:

- IP address
- Netmask
- Gateway IP address
- Fully-qualified domain name

If you choose not to configure each interface in the list, then that interface is not configured, and it does not start at system startup. After the configuration of the IP addresses, the system completes the startup process. At the end of the process, additional packages are installed, and then the installation of Exadata Cell is complete.

9. Select the canonical, fully-qualified domain name from the list. This host name is the primary public host name for the server, and is part of the `/etc/sysconfig/network` file.

   If more than one Ethernet interface was configured with the gateway, then select the line number for the default gateway. This gateway is in the `/etc/sysconfig/network` file, and is used as the default gateway.

10. Provide the following information when prompted for it:

    - ILOM full, domain-qualified host name
    - ILOM IP address
    - ILOM netmask
    - ILOM gateway
    - ILOM NTP servers
    - (Optional) ILOM DNS server

    See Also: Example 5–1, "Using the ipconf Utility to Set the Sun ILOM Interface"

11. (Oracle Exadata Storage Server only) Change the initial passwords for the `root`, `celladmin`, and `cellmonitor` users to more secure passwords. The initial password for the `root` user is `welcome1`. The initial password for the `cellmonitor` and `celladmin` users is `welcome`.

    To change the passwords, log in as the `root` user with password `welcome1`, then use the `passwd` command to change the passwords, such as the following:

    ```
    $ passwd
    $ passwd celladmin
    $ passwd cellmonitor
    ```

    To verify the changed passwords, log in and out using each of the user names.

    Note: The `cellmonitor` user is set up with privileges that enable you to view Exadata Cell objects only. You must be logged in as the `celladmin` user to perform administrative tasks.

12. Check for any failures reported in the `/var/log/cellos/vldrun.first_boot.log` file after the first boot configuration. For each failed validation, perform the following procedure:

   a. Look for `/var/log/cellos/validations/`*`failed_validation_name`*`.SuggestedRemedy` file. The file exists only if the validation process has identified some corrective action. Follow the suggestions in the file to correct the cause of the failure.

   b. If the `SuggestedRemedy` file does not exist, then examine the log file for the failed validation in `/var/log/cellos/validations` to track down the cause, and correct it as needed.

13. (Oracle Exadata Storage Server only) Use the following commands to verify acceptable performance levels:

```
cellcli -e "alter cell shutdown services cellsrv"
cellcli -e "calibrate"
```

Configuring Cells, Cell Disks, and Grid Disks with CellCLI

After you complete the tasks described in "Preparing the Servers", you must configure the cells, cell disks and grid disks for each new storage cell.

During the procedure, you can display help using the `HELP` command, and object attributes using the `DESCRIBE` command. Example 2–5 shows how to display help and a list of attributes for Exadata Cell objects.

Example 2–5 Displaying Help Information

```
CellCLI> HELP
CellCLI> HELP CREATE CELL
CellCLI> HELP ALTER CELL
CellCLI> DESCRIBE CELL
```

Use the following procedure to create the cells, cell disks, and grid disks for Exadata Cell:

1. Log in as the `celladmin` user.

2. Use the `cellcli` command to start Cell Control Command-Line Interface (CellCLI) to connect to the storage cell.

   The required cell services, Restart Server (RS) and Management Server (MS), should be running after the binary has been installed. If not, then an error message displays when using the CellCLI utility. If an error message displays, then run the following commands to start Exadata Cell RS and MS services:

```
CellCLI> ALTER CELL STARTUP SERVICES RS
CellCLI> ALTER CELL STARTUP SERVICES MS
```

3. Configure the cell using the CellCLI `ALTER CELL` command. During first boot, the cell is created, and the flash cell disks and flash cache defined automatically.

```
CellCLI> ALTER CELL name=cell_name,                                       -
           realmname=realm_name,                                          -
           smtpServer='my_mail.example.com',                              -
           smtpFromAddr='john.doe@example.com',                           -
           smtpFromPwd=email_address_password,                            -
           smtpToAddr='jane.smith@example.com',                           -
           notificationPolicy='critical,warning,clear',                   -
           notificationMethod='mail,snmp'
```

See Also:

- "ALTER CELL" on page 8-15
- "Creating Flash Cache and Flash Grid Disks" on page 2-13

4. Use the following command to check the storage cell attributes, and to verify the current configuration:

```
CellCLI> LIST CELL DETAIL
```

5. Create the cell disks, using the `CREATE CELLDISK` command. In most cases, you can use the default cell disk names and LUN IDs. Use the following command to create cell disks and LUN IDs with the default values.

```
CellCLI> CREATE CELLDISK ALL
```

6. Create grid disks on each cell disk of the storage cell, using the `CREATE GRIDDISK` command.

   See Also: "CREATE GRIDDISK" on page 8-42

7. Exit the CellCLI utility after setting up the storage cell using the following command:

```
CellCLI> EXIT
```

8. Repeat the configuration process for each new storage cell. This procedure must be done on each new cell before configuring the Exadata Cell realm, the database server hosts, or the database and Oracle ASM instances.

After you complete the cell configuration, you can perform the following optional steps on the storage cell:

- Add the storage cell to the Exadata Cell realm.
- Configure security on the Exadata Cell grid disks, as described in Chapter 4, "Configuring Security for Oracle Exadata Storage Server Software".
- Configure an interdatabase plan for a cell rather than using the default plans, as described in Chapter 6, "Managing I/O Resources".

For database server hosts other than those in Oracle Exadata Database Machine, refer to release notes for enabling them to work with Exadata Cell.

Creating Flash Cache and Flash Grid Disks

Oracle Exadata Storage Servers are equipped with flash disks. These flash disks can be used to create flash grid disks to store frequently accessed data. Alternatively, all or part of the flash disk space can be dedicated to Exadata Smart Flash Cache. In this case, the most frequently-accessed data is cached in Exadata Smart Flash Cache.

By default, the `CREATE CELL` command creates flash cell disks on all flash disks, and then creates Exadata Smart Flash Cache on the flash cell disks. To change the size of the Exadata Smart Flash Cache or create flash grid disks it is necessary to remove the flash cache, and then create the flash cache with a different size, or create the flash grid disks.

The amount of flash cache allocated can be set using the `flashcache` attribute with the `CREATE CELL` command. If the `flashcache` attribute is not specified, then all available flash space is allocated for flash cache.

Flash cache can also be created explicitly using the `CREATE FLASHCACHE` command. This command uses the `celldisk` attribute to specify which flash cell disks contain cache. Alternatively, `ALL` can be specified to indicate that all flash cell disks are used. The `size` attribute is used to specify the total size of the flash cache to be allocated, and the allocation is evenly distributed across all flash cell disks.

Example 2–6 shows how to create flash cache. In the example, the entire size of the flash cell disk is not used because the `size` attribute has been set.

Example 2–6 Using the CREATE FLASHCACHE Command

```
CellCLI> CREATE FLASHCACHE ALL size=100g
Flash cache cell01_FLASHCACHE successfully created
```

Example 2–7 shows how to use the remaining space on the flash cell disks to create flash grid disks.

Example 2–7 Using the CREATE GRIDDISK Command to Create Flash Grid Disks

```
CellCLI> CREATE GRIDDISK ALL FLASHDISK PREFIX='FLASH'
GridDisk FLASH_FD_00_cell01 successfully created
GridDisk FLASH_FD_01_cell01 successfully created
GridDisk FLASH_FD_02_cell01 successfully created
GridDisk FLASH_FD_03_cell01 successfully created
GridDisk FLASH_FD_04_cell01 successfully created
GridDisk FLASH_FD_05_cell01 successfully created
GridDisk FLASH_FD_06_cell01 successfully created
GridDisk FLASH_FD_07_cell01 successfully created
GridDisk FLASH_FD_08_cell01 successfully created
GridDisk FLASH_FD_09_cell01 successfully created
GridDisk FLASH_FD_10_cell01 successfully created
GridDisk FLASH_FD_11_cell01 successfully created
GridDisk FLASH_FD_12_cell01 successfully created
GridDisk FLASH_FD_13_cell01 successfully created
GridDisk FLASH_FD_14_cell01 successfully created
GridDisk FLASH_FD_15_cell01 successfully created

CellCLI> LIST GRIDDISK
         FLASH_FD_00_cell01      active
         FLASH_FD_01_cell01      active
         FLASH_FD_02_cell01      active
         FLASH_FD_03_cell01      active
         FLASH_FD_04_cell01      active
         FLASH_FD_05_cell01      active
         FLASH_FD_06_cell01      active
         FLASH_FD_07_cell01      active
         FLASH_FD_08_cell01      active
         FLASH_FD_09_cell01      active
         FLASH_FD_10_cell01      active
         FLASH_FD_11_cell01      active
         FLASH_FD_12_cell01      active
         FLASH_FD_13_cell01      active
         FLASH_FD_14_cell01      active
         FLASH_FD_15_cell01      active
```

Use the `LIST FLASHCACHE` command to display the flash cache configuration for the cell, as shown in Example 2–8.

Example 2–8 Displaying the Flash Cache Configuration for a Cell

```
CellCLI> LIST FLASHCACHE DETAIL
         name:                   cell01_FLASHCACHE
         cellDisk:               FD_00_cell01, FD_01_cell01,FD_02_cell01,
                                 FD_03_cell01, FD_04_cell01, FD_05_cell01,
                                 FD_06_cell01, FD_07_cell01, FD_08_cell01,
                                 FD_09_cell01, FD_10_cell01, FD_11_cell01,
                                 FD_12_cell01, FD_13_cell01, FD_14_cell01,
                                 FD_15_cell01
         creationTime:           2009-10-19T17:18:35-07:00
         id:                     b79b3376-7b89-4de8-8051-6eefc442c2fa
         size:                   365.25G
         status:                 normal
```

To remove flash cache from a cell, use the `DROP FLASHCACHE` command, as shown in Example 2–9.

Example 2–9 Dropping Flash Cache from a Cell

```
CellCLI> DROP FLASHCACHE
Flash cache cell01_FLASHCACHE successfully dropped
```

See Also:

- "CREATE CELL" on page 8-37
- "CREATE GRIDDISK" on page 8-42
- "LIST GRIDDISK" on page 8-107
- "LIST FLASHCACHE" on page 8-104
- "DROP FLASHCACHE" on page 8-87

Setting Up Configuration Files for a Database Server Host

After Exadata Cell is configured, the database server host must be configured with the `cellinit.ora` and the `cellip.ora` files to use the cell.

- The `cellinit.ora` file contains the database IP addresses.
- The `cellip.ora` file contains the storage cell IP addresses.

Both files are located on the database server host. These configuration files contain IP addresses, not host names.

The `cellinit.ora` file is host-specific, and contains all database IP addresses that connect to the storage network used by Exadata Cell. This file must exist for each database that connect to Exadata Cell. The IP addresses are specified in Classless Inter-Domain Routing (CIDR) format.

For example:

```
ipaddress1=192.168.50.23/24
```

These IP addresses should not be changed after this file is created.

Note: At boot time on an 8-socket system, each database server generates a `cellaffinity.ora` configuration file. The `cellaffinity.ora` file resides in the `/etc/oracle/cell/network-config` directory, and must be readable by the Oracle database.

The file contains a mapping between the NUMA node numbers and the IP address of the network interface card closest to each server. Oracle Database uses the file to select the closest network interface card when communicating with Exadata Cells, thereby optimizing performance.

This file is only generated and used on an 8-socket system. On a 2-socket system, there is no performance to be gained in this manner, and no `cellaffinity.ora` file. The file is not intended to be directly edited with a text editor.

To configure a database server host for use with a cell, refer to *Oracle Exadata Database Machine Owner's Guide*.

Understanding Automated Cell Maintenance

Management Server (MS) includes a file deletion policy. Every hour, MS deletes the following files:

- All files in the Automatic Diagnostic Repository (ADR) base directory older than 7 days.
- All files in the `LOG_HOME` directory older than 7 days.
- All metric history files older than 7 days.

In addition, the `alert.log` file is renamed if it is larger than 10 MB, and versions of the file that are older than 7 days are deleted if their total size is greater than 50 MB.

Note: The retention period of seven days is the default. The retention period can be modified using the `metricHistoryDays` attribute with the `ALTER CELL` command.

MS includes a file deletion policy that is triggered when file system utilization is high. Deletion of files is triggered when file utilization reaches 80 percent, and an alert is sent before the deletion begins. In particular, the deletion policy is as follows:

- For the `/var/log/oracle` and `/opt/oracle` file systems, files in the ADR base directory, metric history directory, and `LOG_HOME` directory are deleted using a policy based on the file modification time stamp. Files older than the number of days set by the `metricHistoryDays` attribute value are deleted first, then successive deletions occur for earlier files, down to files with modification time stamps older than or equal to 10 minutes, or until file system utilization is less than 75 percent. The renamed `alert.log` files and `ms-odl` generation files that are over 5 MB, and older than the successively-shorter age intervals are also deleted. Crash files in the `/var/log/oracle/crashfiles` directory over 5 MB and older than one day are deleted.

- For the / file system, files in the home directories (`cellmonitor` and `celladmin`), `/tmp`, `/var/crash`, and `/var/spool` directories that are over 5 MB and older than one day are deleted.

  Note: Any directories or files with `SAVE` in the name are not deleted.

  See Also:

  - "ALTER CELL" on page 8-15 and "DESCRIBE CELL" on page 8-54 for additional information about the `metricHistoryDays` attribute
  - Appendix B, "Alert and Error Messages" for additional information about ADR and alert files

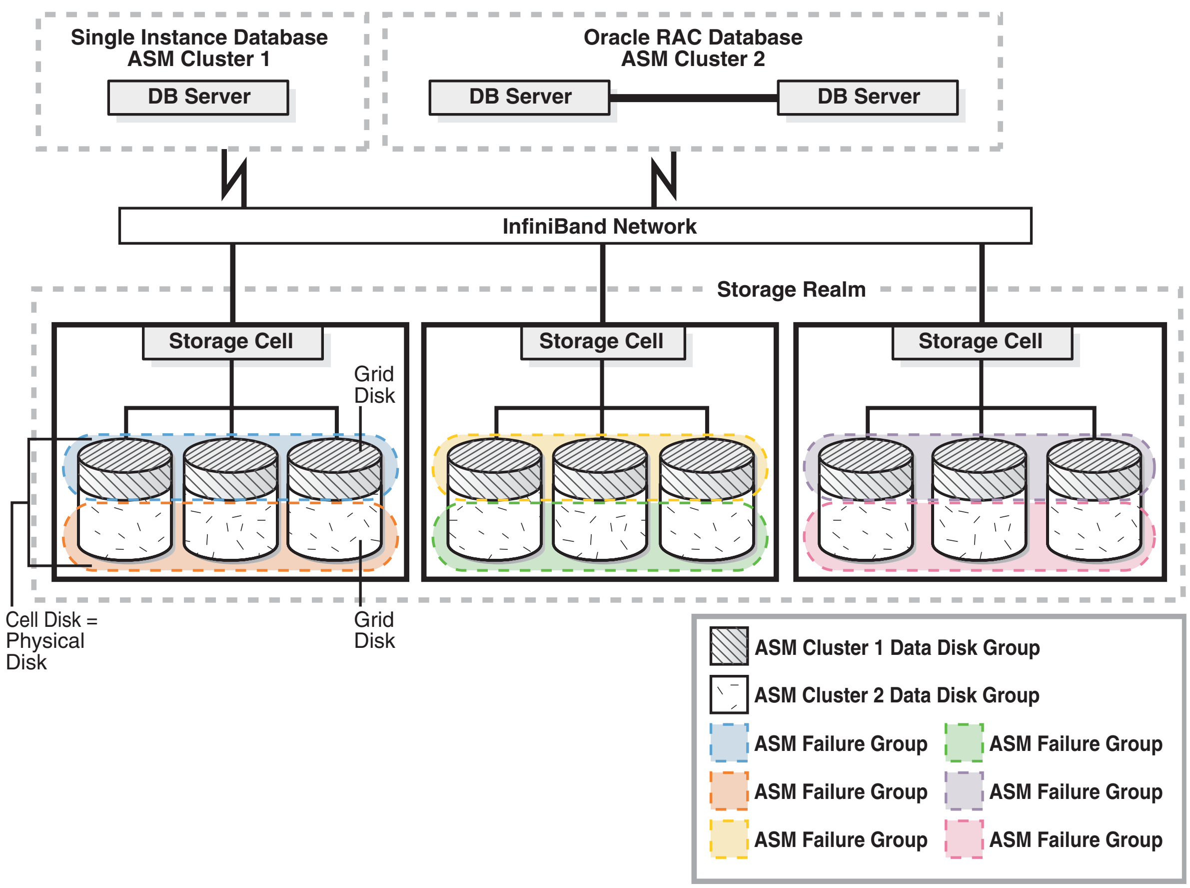
3

Administering Oracle ASM Disk Groups on Oracle Exadata Storage Servers

This chapter explains how to administer Oracle Automatic Storage Management (Oracle ASM) disk groups with Oracle Exadata Storage Server grid disks. Figure 3–1 shows Oracle ASM disk groups with Exadata Cell grid disks. It represents a typical, but simplified configuration, that can be used as a model for building larger Exadata Cell grids with additional storage cells and disks.

clever

Figure 3–1 Sample Exadata Cell Grid



This Exadata Cell grid illustrates the following:

- The storage cells in the grid are connected to the database servers that have a single-instance database and Oracle Real Application Clusters (Oracle RAC) database installation over an InfiniBand network.
- Each storage cell is composed of physical disks.
- Each cell disk represents a physical disk and a LUN.
- Each cell disk is partitioned into grid disks.
- Oracle ASM disk groups are set up to include the grid disks.

Oracle ASM failure groups are created to ensure that files are not mirrored on the same storage cell to tolerate the failure of a single storage cell. The number of failure groups equals the number of Exadata Cells. Exadata Cell Each failure group is composed of a subset of grid disks in the Oracle ASM disk group that belong to a single storage cell.

This chapter contains the following topics:

- Administering Oracle ASM Disk Groups Using Oracle Exadata Storage Servers
- Administering Oracle Exadata Storage Server Grid Disks with Oracle ASM

See Also:

- *Oracle Database 2 Day DBA*
- *Oracle Database Administrator's Guide*
- *Oracle Database Performance Tuning Guide*

Administering Oracle ASM Disk Groups Using Oracle Exadata Storage Servers

This section describes the basic Oracle ASM tasks needed to use Exadata Cell. This section contains the following topics:

- Creating an Oracle ASM Disk Group for Oracle Exadata Storage Servers
- Adding a Failure Group to an Oracle ASM Disk Group
- Mounting or Dismounting an Oracle ASM Disk Group
- Changing a Disk to Offline or Online
- Dropping a Disk from an Oracle ASM Disk Group
- Dropping an Oracle ASM Disk Group

See Also:

- *Oracle Automatic Storage Management Administrator's Guide* for additional information about the following:

  - Oracle ASM features and functions

  - starting and connecting to an Oracle ASM instance

- *Oracle Database SQL Language Reference* for additional information about the SQL disk group commands, such as `CREATE DISKGROUP` and `ALTER DISKGROUP`

Creating an Oracle ASM Disk Group for Oracle Exadata Storage Servers

This section describes how to create an Oracle ASM disk group for Exadata Cell using the `CREATE DISKGROUP` SQL command. Before creating an Oracle ASM disk group, determine which grid disks belong to the Oracle ASM disk group. It is recommended that you choose similar names for the Oracle ASM disk group and its grid disks whenever possible.

The Exadata Cell grid disks are specified with the following pattern:

```
o/cell_IPaddress/griddisk-name
```

In the preceding syntax, *`cell_IPaddress`* is the IP address of Oracle Exadata Storage Server.

The cell discovery strings begin with the `o/` prefix.

When specifying the grid disks to be added to the disk group, consider the following:

- The default Oracle ASM disk name is the grid disk name. Oracle recommends using the default.
- The default failure group name is the cell name. Oracle recommends using the default.

When a failure group is not specified, Oracle ASM adds each disk within its own failure group. However, when the disks are stored on Oracle Exadata Storage Server Software cells and a failure group is not specified, Oracle ASM adds a disk to the failure group for that cell. The failure group name is the cell name.

> **Note:** If a cell is renamed, and a disk from that cell is added to an existing disk group that has disks from that cell, then Oracle ASM adds the new disk to a failure group using the new cell name. To ensure all the disks from the cell are in one failure group, add the disk to the disk group and specify the original failure group name.

To enable Smart Scan predicate offload processing, all disks in a disk group must be Exadata Cell grid disks. You cannot include conventional disks with Exadata Cell grid disks.

See Also:

- "Naming Conventions for Oracle Exadata Storage Server Grid Disks" on page 3-8
- "Automatic Storage Management Failure Group" on page 1-9 for additional information about Oracle ASM and Exadata Cell

To create an Oracle ASM disk group to use Exadata Cell grid disks, perform the following procedure:

1. Connect to the Oracle ASM instance.
2. Ensure that the `ORACLE_SID` environment variable is set to the Oracle ASM instance using a command similar to the following:

   ```
   $ setenv ORACLE_SID ASM_instance_SID
   ```

3. Start SQL*Plus on the Oracle ASM instance, and log in as a user with `SYSASM` administrative privileges.

   ```
   $ sqlplus / AS SYSASM
   ```

4. Determine which Exadata Cell grid disks are available by querying the `V$ASM_DISK` view on the Oracle ASM instance, using the following syntax:

```
SQL> SELECT PATH, header_status STATUS FROM V$ASM_DISK WHERE path LIKE 'o/%';
```

5. Create an Oracle ASM disk group to include disks on the cells using a command similar to the following:

```
SQL> CREATE DISKGROUP data NORMAL REDUNDANCY
DISK 'o/*/DATA*'
ATTRIBUTE 'AU_SIZE' = '4M',
          'cell.smart_scan_capable'='TRUE',
          'compatible.rdbms'='11.2.0.2',
          'compatible.asm'='11.2.0.2';

SQL> CREATE DISKGROUP reco NORMAL REDUNDANCY
DISK 'o/*/RECO*'
ATTRIBUTE 'AU_SIZE' = '4M',
          'cell.smart_scan_capable'='TRUE',
          'compatible.rdbms'='11.2.0.2',
          'compatible.asm'='11.2.0.2';

SQL> ALTER DISKGROUP DBFS_OCRVOTE SET ATTRIBUTE
     'compatible.rdbms' = '11.2.0.2';
```

In the preceding commands, the `ALTER` command is needed to change `compatibile.rdbms` for the disk group created during installation to hold the OCR and voting disks.

> **Note:** The Oracle ASM disk group `compatible` attributes take precedence over the COMPATIBLE initialization parameter for the Oracle ASM instance.

6. View the Oracle ASM disk groups and associated attributes with a SQL query on `V$ASM` dynamic views. For example, the following SQL command lists the Oracle ASM disk groups and the attributes:

```
SQL> SELECT dg.name AS diskgroup, SUBSTR(a.name,1,24) AS name,
     SUBSTR(a.value,1,24) AS value FROM V$ASM_DISKGROUP dg, V$ASM_ATTRIBUTE a
     WHERE dg.group_number = a.group_number;

DISKGROUP                    NAME                       VALUE
---------------------------- ------------------------ ------------------------
DATA                         compatible.rdbms           11.2.0.2
DATA                         compatible.asm             11.2.0.2
DATA                         au_size                    4194304
DATA                         disk_repair_time           3.6h
DATA                         cell.smart_scan_capable    TRUE
...
```

7. Create a tablespace in the disk group to take advantage of Exadata Cell features, such as offload processing. This tablespace should contain the tables that you want to query with offload processing. The following is an example of the syntax:

```
SQL> CREATE TABLESPACE tablespace_name DATAFILE '+DATA';
```

In the preceding command, `+DATA` is the name of the Oracle ASM disk group.

8. Verify that the tablespace is in an Exadata Cell disk group. The `PREDICATE_EVALUATION` column of the `DBA_TABLESPACES` view indicates whether predicates are evaluated by host (`HOST`) or by storage (`STORAGE`). For example, the following SQL command verifies the tablespace is entirely within the cells:

```
SQL> SELECT tablespace_name, predicate_evaluation FROM dba_tablespaces
     WHERE tablespace_name = 'DATA_TB';

TABLESPACE_NAME                PREDICA
------------------------------ -------
DATA_TB                        STORAGE
```

See Also:

- *Oracle Automatic Storage Management Administrator's Guide* for additional information about the following:

  - Oracle ASM features and functions

  - managing Oracle ASM disk groups

- *Oracle Database SQL Language Reference* for additional information about the SQL disk group commands, such as CREATE DISKGROUP, ALTER DISKGROUP, and CREATE TABLESPACE

Example 3–1 shows the use of a CREATE DISKGROUP command to create a disk group. The names of the disk groups are shown in bold.

Example 3–1 Creating Oracle ASM Disk Groups for Exadata Cell

```
SQL> CREATE DISKGROUP data NORMAL REDUNDANCY

-- These grid disks are on cell01
   DISK
   'o/*/data_CD_00_cell01',
   'o/*/data_CD_01_cell01',
   'o/*/data_CD_02_cell01',

-- These grid disks are on cell02
   DISK
   'o/*/data_CD_00_cell02',
   'o/*/data_CD_01_cell02',
   'o/*/data_CD_02_cell02',

-- These disk group attributes must be set for cell access
-- Note that this disk group is set for cell only
   ATTRIBUTE 'compatible.rdbms' = '11.2.0.2',
             'compatible.asm' = '11.2.0.2',
             'au_size' = '4M',
             'cell.smart_scan_capable' = 'TRUE';

SQL> CREATE DISKGROUP reco NORMAL REDUNDANCY

-- These grid disks are on cell01
   DISK
   'o/*/reco_CD_00_cell01',
   'o/*/reco_CD_01_cell01',
   'o/*/reco_CD_02_cell01'

-- These grid disks are on cell02
   DISK
   'o/*/reco_CD_00_cell02',
   'o/*/reco_CD_01_cell02',
   'o/*/reco_CD_02_cell02'
```

```
-- These disk group attributes must be set for cell access
-- Note that this disk group is set for cell only
   ATTRIBUTE 'compatible.rdbms' = '11.2.0.2',
             'compatible.asm' = '11.2.0.2',
             'au_size' = '4M',
             'cell.smart_scan_capable' = 'TRUE';
```

Adding a Failure Group to an Oracle ASM Disk Group

To add a failure group to an Oracle ASM disk group, perform the following procedure:

1. Determine which disks are available by querying the `V$ASM_DISK` view on the Oracle ASM instance. If the header status is set to `CANDIDATE`, then the disk is a candidate for a disk group.

   Do not add Exadata Cell grid disks to a non-Exadata Cell ASM disk group unless you are planning to migrate the disk group to an Exadata Cell disk group.

2. Use the SQL `ALTER DISKGROUP` command with the `ADD DISK` clause to add the disk to the Oracle ASM disk group using syntax similar to the following:

   ```
   SQL> ALTER DISKGROUP disk_group_name ADD DISK 'o/cell_IPaddress/data*';
   ```

   When the disk is added, Oracle ASM rebalances the disk group. You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

   See Also: "Naming Conventions for Oracle Exadata Storage Server Grid Disks" on page 3-8

Mounting or Dismounting an Oracle ASM Disk Group

A disk group must be mounted by an Oracle ASM instance before database instances can access the files in the disk group. Mounting the disk group requires discovering all of the disks and locating the files in the disk group that is being mounted.

To mount or dismount a disk group, use the SQL `ALTER DISKGROUP` command with the `MOUNT` or `DISMOUNT` option.

You can use the `FORCE` option of the `ALTER DISKGROUP` command `MOUNT` clause to mount disk groups if their components are unavailable, which results in a loss of full redundancy.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for additional information about mounting disk groups

Changing a Disk to Offline or Online

To change an Oracle ASM disk to `INACTIVE` or `ACTIVE`, perform the following procedure:

1. Determine which disk you want offline or online in the Oracle ASM disk group by querying the `V$ASM_DISK` and `V$ASM_DISKGROUP` views on the Oracle ASM instance.

2. Use one of the following commands:

   - To make a disk `INACTIVE`, use the following command:

     ```
     CellCLI> ALTER GRIDDISK gdisk_name INACTIVE
     ```

   - To make a disk ACTIVE, use the following command:

```
CellCLI> ALTER GRIDDISK gdisk_name ACTIVE
```

As soon as the disk is online, Oracle ASM rebalances the disk group. You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

See Also:

- Example 3–5, "Determining Grid Disks in an Oracle ASM Disk Group" on page 3-10
- "ALTER GRIDDISK" on page 8-22 for additional information about active or inactive modes for grid disks
- "Making an Oracle Exadata Storage Server Grid Disk Active or Inactive" on page 3-9

Dropping a Disk from an Oracle ASM Disk Group

To drop a disk from a disk group, perform the following procedure:

1. Determine which disks you want to drop from the Oracle ASM disk group by querying the `V$ASM_DISK` and `V$ASM_DISKGROUP` views on the Oracle ASM instance.

   If you are removing an Exadata Cell grid disk, then ensure that you identify the grid disks that are mapped to each Oracle ASM disk group.

2. Use the SQL `ALTER DISKGROUP` command with the `DROP DISK` clause to drop the disks from the Oracle ASM disk group. For example:

   ```
   SQL> ALTER DISKGROUP disk_group_name DROP DISK data_CD_11_cell01;
   ```

   When the disk is dropped from the Oracle ASM disk group, Oracle ASM rebalances the disk group. You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

After an Exadata Cell grid disk is dropped from the Oracle ASM disk group, you can drop the grid disk from a cell.

See Also:

- Example 3–5, "Determining Grid Disks in an Oracle ASM Disk Group" on page 3-10
- "Determining Which Oracle Exadata Storage Server Grid Disks Belong to an Oracle ASM Disk Group" on page 3-11
- "Determining Which Oracle ASM Disk Group Contains an Oracle Exadata Storage Server Grid Disk" on page 3-10
- "Dropping an Oracle Exadata Storage Server Grid Disk" on page 3-9

Dropping an Oracle ASM Disk Group

To drop an Oracle ASM disk group, perform the following procedure:

1. Determine the disk group that you want to drop by querying the `V$ASM_DISKGROUP` view on the Oracle ASM instance.
2. Use the SQL `DROP DISKGROUP` command to drop the Oracle ASM disk group.

If you cannot mount a disk group but must drop it, then use the `FORCE` option with the `DROP DISKGROUP` command.

Administering Oracle Exadata Storage Server Grid Disks with Oracle ASM

This section describes the following tasks:

- Naming Conventions for Oracle Exadata Storage Server Grid Disks
- Changing an Oracle Exadata Storage Server Grid Disk That Belongs to an Oracle ASM Disk Group
- Determining Which Oracle ASM Disk Group Contains an Oracle Exadata Storage Server Grid Disk
- Determining Which Oracle Exadata Storage Server Grid Disks Belong to an Oracle ASM Disk Group

Naming Conventions for Oracle Exadata Storage Server Grid Disks

The name of the grid disk should contain the cell disk name to make it easier to determine which grid disks belong to a cell disk. To help determine which grid disks belong to an Oracle ASM disk group, a subset of the grid disk name should match all or part of the name of the Oracle ASM disk group to which the grid disk will belong.

For example, if a grid disk is created on the cell disk `CD_03_cell01`, and that grid disk belongs to an Oracle ASM disk group named `data0`, then the grid disk name should be `data0_CD_03_cell01`.

When you use the `ALL PREFIX` option with `CREATE GRIDDISK`, a unique grid disk name is automatically generated that includes the prefix and cell name. If you do not use the default generated name when creating grid disks, then you must ensure that the grid disk name is unique across all cells. If the disk name is not unique, then it might not be possible to add the grid disk to an Oracle ASM disk group.

See Also:

- "Determining Which Oracle ASM Disk Group Contains an Oracle Exadata Storage Server Grid Disk" on page 3-10
- "Determining Which Oracle Exadata Storage Server Grid Disks Belong to an Oracle ASM Disk Group" on page 3-11
- Example 2–3, "Creating Cell Disks" on page 2-4
- Example 2–4, "Creating Grid Disks" on page 2-5

Changing an Oracle Exadata Storage Server Grid Disk That Belongs to an Oracle ASM Disk Group

When you change a grid disk that belongs to an Oracle ASM disk group, you must consider how the change might affect the Oracle ASM disk group to which the grid disk belongs.

This section contains the following topics:

- Changing an Oracle Exadata Storage Server Grid Disk Name
- Making an Oracle Exadata Storage Server Grid Disk Active or Inactive
- Dropping an Oracle Exadata Storage Server Grid Disk

Changing an Oracle Exadata Storage Server Grid Disk Name

To change attributes of a grid disk, use the CellCLI `ALTER GRIDDISK` command. Use the `DESCRIBE GRIDDISK` command to determine which Exadata Cell grid disk attributes can be modified.

> **Caution:** Before changing the name of a grid disk that belongs to an Oracle ASM disk group, ensure that the corresponding Oracle ASM disk is offline. See "Changing a Disk to Offline or Online" on page 3-6

Example 3–2 shows how to rename a grid disk.

Example 3–2 Changing an Oracle Exadata Storage Server Grid Disk Name

```
CellCLI> ALTER GRIDDISK data011 name='data0_CD_03_cell04'
```

> **See Also:** "ALTER GRIDDISK" on page 8-22

Making an Oracle Exadata Storage Server Grid Disk Active or Inactive

To make an Exadata Cell grid disk active or inactive, use the CellCLI `ALTER GRIDDISK` command. A grid disk should be made inactive before dropping the grid disk to ensure that the grid disk is not in use.

Example 3–3 shows how to make a grid disk active or inactive.

Example 3–3 Setting an Oracle Exadata Storage Server Grid Disk to Active or Inactive Status

```
CellCLI> ALTER GRIDDISK data0_CD_03_cell04 ACTIVE

CellCLI> ALTER GRIDDISK data0_CD_03_cell04 INACTIVE
```

When you deactivate a grid disk that is in use, the cell automatically requests the Oracle ASM instance to offline the Oracle ASM disk before deactivating the disk. In order to make the disk usable again, make the grid disk active in the cell. The corresponding Oracle ASM disk is automatically brought back online.

There is usually no need to change the grid disk state from active to inactive, or from inactive to active. Oracle ASM discovers all active disks. However, if a grid disk is not available to Oracle ASM, then the grid disk should be changed to inactive.

> **Note:** If a disk group is created with external redundancy, and a grid disk in that group is made inactive, then the disk group cannot be mounted.

> **See Also:**
>
> - "ALTER GRIDDISK" on page 8-22
> - "Changing a Disk to Offline or Online" on page 3-6

Dropping an Oracle Exadata Storage Server Grid Disk

To drop an Exadata Cell grid disk, use the CellCLI `DROP GRIDDISK` command. Make the grid disk inactive before dropping the grid disk to ensure that the grid disk is not in use. The FORCE option can be used to force the grid disk that is in use to be dropped.

Caution:

- Before dropping a grid disk that belongs to an Oracle ASM disk group, ensure that the corresponding Oracle ASM disk was dropped from the disk group.
- Before dropping a grid disk using the FORCE option, ensure that the Oracle ASM disk was dropped from the disk group.

To drop a grid disk, perform the following procedure.

1. Drop the Oracle ASM disk from the disk group using the following command:

   ```
   SQL> ALTER DISKGROUP disk_group_name DROP DISK disk_name;
   ```

2. Make the corresponding grid disk inactive using the following command:

   ```
   CellCLI> ALTER GRIDDISK disk_name INACTIVE
   ```

3. Drop the grid disk using the following command:

   ```
   CellCLI> DROP GRIDDISK disk_name
   ```

Example 3–4 shows how to drop a specified grid disk or multiple grid disks.

Example 3–4 Dropping Grid Disks

```
CellCLI> ALTER GRIDDISK data0_CD_03_cell04 INACTIVE
CellCLI> DROP GRIDDISK data0_CD_03_cell04

CellCLI> ALTER GRIDDISK ALL PREFIX=data0 INACTIVE
CellCLI> DROP GRIDDISK ALL PREFIX=data0

CellCLI> DROP GRIDDISK data02_CD_04_cell01 FORCE
```

See Also: "DROP GRIDDISK" on page 8-88

Determining Which Oracle ASM Disk Group Contains an Oracle Exadata Storage Server Grid Disk

If a grid disk name matches the Oracle ASM disk name, and the name contains the Oracle ASM disk group name, then you can determine the Oracle ASM disk group to which the grid disk belongs. You can also use SQL commands on the Oracle ASM instance to find the Oracle ASM disk group that matches part of the specific grid disk name. This can help you to determine which Oracle ASM disk group contains a specific grid disk.

Example 3–5 shows how you can find the Oracle ASM disk group that contains grid disks that begin with DATA0, for example DATA0_CD_03_CELL04.

Example 3–5 Determining Grid Disks in an Oracle ASM Disk Group

```
SQL> SELECT d.label as asmdisk, dg.name as diskgroup
     FROM V$ASM_DISK d, V$ASM_DISKGROUP dg
     WHERE dg.name LIKE 'DATA0%'
           AND d.group_number = dg.group_number;
```

```
ASMDISK                DISKGROUP
---------------------- -------------
DATA0_CD_00_CELL04      DATA0
DATA0_CD_01_CELL04      DATA0
DATA0_CD_02_CELL04      DATA0
DATA0_CD_03_CELL04      DATA0
```

Determining Which Oracle Exadata Storage Server Grid Disks Belong to an Oracle ASM Disk Group

If a grid disk name contains the Oracle ASM disk group name, then you can use SQL commands on the Oracle ASM instance to list the Oracle ASM disk group names, and use the CellCLI utility to search for specific grid disk names.

Example 3–6 shows how to use a SQL command to display the Oracle ASM disk group names on the Oracle ASM instance.

Example 3–6 Displaying Oracle ASM Disk Group Names

```
SQL> SELECT name FROM V$ASM_DISKGROUP;

NAME
------------------------------
CONTROL
DATA0
DATA1
DATA2
LOG
STANDBY
```

Example 3–7 shows how to display similar grid disk group names on the cell using the dcli utility.

Example 3–7 Searching for Grid Disks by Name

```
$ ./dcli -l celladmin "cellcli -e list griddisk where name like \'.*data0.*\'"

data0_CD_01_cell04
data0_CD_02_cell04
data0_CD_03_cell04
...
```

See Also: "Naming Conventions for Oracle Exadata Storage Server Grid Disks" on page 3-8

4

Configuring Security for Oracle Exadata Storage Server Software

This chapter explains how to configure security for Oracle Exadata Storage Server Software (Exadata Cell).

This chapter contains the following topics:

Understanding Data Security for Oracle Exadata Storage Server Software

Exadata Cell data security is implemented by controlling which Oracle Automatic Storage Management (Oracle ASM) clusters and database clients can access specific grid disks on storage cells. A collection of grid disks in the same security domain is referred to as a storage realm. By default, all database and Oracle ASM instances have access to all storage cell grid disks.

- To set up security so that all database clients of an Oracle ASM cluster have access to specific grid disks, configure Oracle ASM-scoped security.
- To set up security so that specific database clients of an Oracle ASM cluster have access to specific grid disks, configure database-scoped security.

To have consistent Exadata Cell security, ensure the following:

- All grid disks that belong to the same Oracle ASM disk group have the same cell-side grid disk security defined to avoid confusion and errors.
- All Oracle Real Application Clusters (Oracle RAC) servers in an Oracle ASM cluster have the same content, ownership, and security for the Oracle ASM `cellkey.ora` file.
- All Oracle RAC servers in a database cluster have the same content, ownership, and security for the database `cellkey.ora` file.

- If database-scoped security is implemented, then ensure it is implemented for all databases accessing the grid disks. Do not mix Oracle ASM-scoped security and database-scoped security.

While setting up security, it is imperative that the configuration is the same across cells. Using the dcli utility to make configuration changes ensures consistency by eliminating the potential of user errors.

Exadata Cell security allows open security, Oracle ASM-scoped security or database security. The following sections describe each mode:

- About Open Security Mode
- About Oracle ASM-Scoped Security Mode
- About Database-Scoped Security Mode

To implement security, you must establish a security key.

> **See Also:** "About Security Keys" on page 4-6 for additional information about security keys

About Open Security Mode

Open security mode enables access by any database client to a grid disk. Open security mode is useful for test or development databases where there are no security requirements. This is the default security mode after creating a new storage cell.

To use this security mode, you do not set up any security functionality for an Oracle ASM cluster or a database client for the grid disk. You do not set up any security key files.

About Oracle ASM-Scoped Security Mode

Oracle ASM-scoped security mode enables access by all the database clients of an Oracle ASM cluster to grid disks on cells. Oracle ASM-scoped security is appropriate when you want all databases on a host cluster to have access to cell grid disks that compose the Oracle ASM disk groups managed by the Oracle ASM cluster. This includes the case when there is only one database in an Oracle ASM cluster.

When Oracle ASM-scoped security is set up for an Oracle ASM cluster and grid disks, the grid disks are available only to the databases on the Oracle ASM cluster.

About Database-Scoped Security Mode

Database-scoped security mode configures access to specific grid disks on cells for specific database clients of an Oracle ASM cluster. This security mode is appropriate when multiple databases are accessing cells, and you want to control which databases can access specific grid disks that compose Oracle ASM disk groups. Set up Oracle ASM-scoped security for your initial security mode, then set up database-scoped security for specific database clients and grid disks. After setting up database-scoped security among the database clients and grid disks, only those specific grid disks are available to the specified database clients.

When using database-scoped security, there is one key file per database per host, and one access control list (ACL) entry per database on each cell.

See Also:

- "Setting Up Oracle ASM-Scoped Security on Oracle Exadata Storage Servers" on page 4-7
- "Setting Up Database-Scoped Security on Oracle Exadata Storage Servers" on page 4-8
- "Understanding the cellkey.ora File" on page 4-5

Understanding Operating System Security of Oracle Exadata Storage Servers

The security of the operating system on Oracle Exadata Storage Servers consists of the following:

- Enforcing security policies
- Protecting network access paths to the cells
- Monitoring operating system-level activities

Oracle Exadata Storage Server Software includes security features to ensure the operating system and network access to the Oracle Exadata Storage Servers are secure.

Security Policies for Oracle Exadata Storage Servers

User access to the operating system can be secured by the use of secure, hardened passwords. The passwords for users who administer Oracle Exadata Storage Server Software adhere to the following security guidelines:

- Passwords must be changed every 90 days.
- Passwords cannot be changed within 24 hours of the last change.
- Passwords must use three character classes. Character classes for passwords are digits, lowercase letters, uppercase letters, and other characters.
- Minimum password length is 12 characters when using three character classes. Minimum password length is 8 characters when using four character classes.

> **Note:** Uppercase letters at the beginning of the password, and digits at the end of the password are not counted when calculating the number of character classes.

- Maximum password length is 40 characters.
- A new password cannot be similar to the existing password.
- A user account is locked after five failed attempts to log in or use SSH.
- A log in session or SSH session will time out after 15000 seconds when left idle.

Changing a Password

Users are notified of the need to change their passwords 7 days before the expiration date. To change a password, use the following command:

```
passwd username
```

In the preceding command, *username* is the user name. The following is an example of the command:

```
passwd celladmin
```

Enabling the Security Policies

The `/opt/oracle.cellos/RESECURED_NODE` file enables the security policies. If the file does not exist, then do the following:

1. Shut down the cell services using the following command:

   ```
   service celld stop
   ```

2. Shut down the Oracle Grid Infrastructure services on all database servers.
3. Run the following script to set the security policies:

   ```
   /opt/oracle.SupportTools/harden_passwords_reset_root_ssh
   ```

   The preceding command restarts the cell. After the cell is up, you need to set a new password.

4. Set a new password.

Viewing Failed Password Attempts

To view failed password attempts, use the following command:

```
/sbin/pam_tally2
```

Resetting a Locked Password

If a user account has 5 failed attempts to log in or use SSH, or the session has been idle for more than 15000 seconds, then the account is locked.

To reset an account, use the following command:

```
/sbin/pam_tally2 --user username --reset
```

In the preceding command, *username* is the name of the user that has the locked account.

To reset after an idle session timeout interval, do the following:

1. Change or remove the TMOUT parameter in the `/etc/profile` file.
2. Change or comment out the ClientAliveCountMax and ClientAliveInterval parameters in the `/etc/ssh/sshd_config` file.

Modifying Password Policies on the Database Servers

The password policies cannot be modified for Oracle Exadata Storage Servers. They can be modified for the database servers. The following procedure describes how to modify the database server password policies:

1. Modify the settings in the `/etc/login.defs` file to change the aging policies. The following is an example of the policies:

   ```
   PASS_MAX_DAYS 90
   PASS_MIN_DAYS 1
   PASS_MIN_LEN 8
   PASS_WARN_AGE 7
   ```

2. Modify the character class restrictions by changing the values for the min parameter in the `/etc/pam.d/system-auth` file. The current min values are `min=disabled,disabled,16,12 8`. A setting of `5,5,5,5,5` allows passwords to be as short as five characters, and removes character class restrictions.
3. Reboot the database servers.

Network Access to Oracle Exadata Storage Servers

Oracle Exadata Storage Server Software includes the cellwall service that implements a firewall on each cell. The service is located in the `/etc/init.d/cellwall` directory, and implements iptables firewall on the cell. In addition, the SSH server is configured to respond to connection requests only on the management network (NET0), and the InfiniBand network (BONDIB0).

To review the firewall rules, run the following command as the `root` user:

```
iptables --list
```

> **Note:** There is no firewall automatically configured for the database servers. Implement a set of iptables on the database servers to meet your network requirements for Oracle Exadata Database Machine.

Operating System Activity Monitoring on Oracle Exadata Storage Servers

Each Oracle Exadata Storage Server is configured with `auditid` to audit system-level activity. To manage audits and generate reports use the `auditctl` command. The audit rules are in the `/etc/audit/audit.rules` file. Any changes are not preserved when applying a patch set.

Understanding the cellkey.ora File

To set up security on the host computers that contain clients that access an Oracle Exadata Storage Server, you must configure the `cellkey.ora` file, and place the file on all the computers.

One `cellkey.ora` file is required for Oracle ASM-scoped security, and another `cellkey.ora` file is required for each database client using database-scoped security. The key file for Oracle ASM-scoped security has the same contents on all database servers. However, the contents of the key file for database-scoped security are different for each database client.

The `cellkey.ora` file contains entries that configure security among Oracle ASM, database clients and cells. The `key` and `asm` values in the `cellkey.ora` files on the Oracle ASM and database host computers must match the values assigned to the clients on the cells.

The following table shows the `cellkey.ora` fields:

Field	Description
key	For Oracle ASM-scoped security, this key must match the value of the key assigned to the Oracle ASM cluster with the CellCLI `ASSIGN KEY` command. For database-scoped security, this key must match the value of the key assigned to the database client with the CellCLI `ASSIGN KEY` command. See Example 4–3 on page 4-9. This field is required.
asm	This field must match the value of the Oracle ASM cluster unique name (DB_UNIQUE_NAME of the Oracle ASM cluster). This is the name used when configuring grid disks for security with the CellCLI `CREATE GRIDDISK` or `ALTER GRIDDISK` command. See Example 4–4 on page 4-9. This field is required.
realm	If entered, then it must match the value of the `realmName` attribute for the cells in the realm. See Example 2–1 on page 2-3. This field is optional.

The entries for the `cellkey.ora` file are shown in Example 4–1:

Example 4–1 cellkey.ora File Entries

```
key=66e12adb996805358bf82258587f5050
asm=+asm
realm=my_realm
```

About Security Keys

A security key is used to authenticate and protect messages between cells and the clients of the cells. The CellCLI `CREATE KEY` command generates a random hexadecimal string for use as a security key.

Example 4–2 Creating a Security Key for Oracle ASM-Scoped Security

```
CellCLI> CREATE KEY

         66e12adb996805358bf82258587f5050
```

The `CREATE KEY` command can be run on any cell and only must be run when you need to create a new unique key.

- For Oracle ASM-scoped security, you only run `CREATE KEY` once to create the single key necessary in that security mode.
- For database-scoped security, create one security key for Oracle ASM and one security key for each database that use database-scoped security.

See Also:

- "CREATE KEY" on page 8-44 for additional information about generating a key,
- "LIST KEY" on page 8-113 for additional information listing keys assigned to clients
- "Understanding the cellkey.ora File" on page 4-5 for additional information about setting up the `cellkey.ora` file

Setting Up Oracle ASM-Scoped Security on Oracle Exadata Storage Servers

This section describes how to configure Oracle ASM-scoped security for grid disks on Oracle Exadata Storage Servers.

To set up Oracle ASM-scoped security, perform the following procedure:

1. Shut down the database and Oracle ASM instances that will have their security configuration changed.
2. Use the CellCLI `CREATE KEY` command to generate a random hexadecimal string. The command can be run on any cell. After running the command, the system displays the new key.

   ```
   CellCLI> CREATE KEY

   66e12adb996805358bf82258587f5050
   ```

   This key is used for the Oracle ASM cluster client and for the key entry in the `cellkey.ora` file.

   See Also:

   - "About Security Keys" on page 4-6
   - "CREATE KEY" on page 8-44

3. Copy the key to the `cellkey.ora` file using the format shown in Example 4–1.
4. Use the `ASSIGN KEY` command to assign the security key to the Oracle ASM cluster client on all the cells that you want the Oracle ASM cluster to access. The following is an example of the command:

   ```
   CellCLI> ASSIGN KEY FOR 'cluster_name='66e12adb996805358bf82258587f5050'
   ```

   In the preceding command, *cluster_name* is the Oracle ASM or Oracle RAC cluster unique name (DB_UNIQUE_NAME). The cluster unique name must be used when assigning a key.

   Note: You can use either of the following SQL commands to display the DB_UNIQUE_NAME parameter:

   ```
   SQL> SHOW PARAMETER db_unique_name

   SQL> SELECT name, value FROM V$PARAMETER WHERE name = 'db_unique_
   name';
   ```

   See Also:

   - "ASSIGN KEY" on page 8-33
   - "LIST KEY" on page 8-113

5. Enter the Oracle ASM cluster name in the `availableTo` attribute with the `CREATE GRIDDISK` or `ALTER GRIDDISK` command to configure security on the grid disks on all the cells that you want the Oracle ASM cluster to access. The following are examples of the commands to create a new grid disk with security and to change security on existing grid disks:

- Create new grid disk with security:

```
CellCLI> CREATE GRIDDISK ALL PREFIX=sales, size=75G, -
         availableTo='cluster_name'
```

- Change security on existing grid disks:

```
CellCLI> ALTER GRIDDISK sales_CD_01_cell01, sales_CD_02_cell01,           -
         sales_CD_03_cell01,  sales_CD_04_cell01, sales_CD_05_cell01,     -
         sales_CD_06_cell01                                               -
         availableTo='cluster_name'
```

In the preceding commands, the *cluster_name* is the Oracle ASM or Oracle RAC cluster unique name (DB_UNIQUE_NAME).

See Also:

- "ALTER GRIDDISK" on page 8-22
- "CREATE GRIDDISK" on page 8-42
- "LIST CELLDISK" on page 8-102
- "LIST GRIDDISK" on page 8-107

6. Restart the database and Oracle ASM instances.

Setting Up Database-Scoped Security on Oracle Exadata Storage Servers

This section describes the database-scoped security configuration required for Oracle Exadata Storage Servers.

Before making updates to the security setups on cells, you must shut down the database and Oracle ASM instances. After you complete the steps in this section, you must ensure the key is formatted correctly, as described in "Understanding the cellkey.ora File" on page 4-5. After the `cellkey.ora` file configuration is complete, start the database and Oracle ASM instances.

To set up database-scoped security, perform the following procedure:

Note: You should only set up database-scoped security after configuring and testing Oracle ASM-scoped security.

---

1. Shut down the database and Oracle ASM instances that will have their security configuration changed.
2. Use the CellCLI `CREATE KEY` command to create security keys for those database clients that you want to access specific grid disks. The command can be run on any cell. After running the command, the system displays the new key.

   ```
   CellCLI> CREATE KEY

   66e12adb996805358bf82258587f5050
   ```

3. Copy the key to the `cellkey.ora` file using the format shown in Example 4–1.
4. Use the `ASSIGN KEY` command to assign the keys to database clients on the cells that contain the grid disks, as shown in Example 4–3.

Example 4–3 Assigning Keys to Database Clients

```
CellCLI> ASSIGN KEY FOR  db1='51a826646ebe1f29e33c6ed7c4965c9a',  -
                         db2='bd0843beeed5e18e6664576cf9805b69',  -
                         db3='6679ef9ec02fa664582c3464d4b0191f'
```

You must use the database unique name (DB_UNIQUE_NAME) for the database clients when assigning keys.

Note: You can use either of the following SQL commands to display the DB_UNIQUE_NAME parameter:

```
SQL> SHOW PARAMETER db_unique_name

SQL> SELECT name, value FROM V$PARAMETER WHERE name = 'db_unique_
name';
```

---

5. Use the `availableTo` attribute with the `CREATE GRIDDISK` or `ALTER GRIDIDISK` command to configure security on grid disks. You must include the Oracle ASM cluster name with the database clients names when setting the value of the `availableTo` attribute. The names that you enter must be the Oracle ASM, database, or Oracle RAC cluster unique names (DB_UNIQUE_NAME).

   In Example 4–4, database-scoped security is configured when the grid disks are created with the use of the `availableTo` attribute. This attribute determines which specific clients are configured to access the grid disks.

Example 4–4 Creating Grid Disks with Database-Scoped Security

```
CellCLI> CREATE GRIDDISK sales_CD_00_cell01, sales_CD_01_cell01 size=75G, -
         availableTo='+asm,db1'

CellCLI> CREATE GRIDDISK sales_CD_02_cell01, sales_CD_03_cell01 size=75G, -
         availableTo='+asm,db2'

CellCLI> CREATE GRIDDISK sales_CD_04_cell01, sales_CD_05_cell01 size=75G, -
         availableTo='+asm,db3'
```

You can change security on existing grid disks with the `ALTER GRIDDISK` command, as shown in Example 4–5.

Example 4–5 Altering Grid Disks to Configure Database-Scoped Security

```
CellCLI> ALTER GRIDDISK sales_CD_01_cell01, sales_CD_02_cell01 -
         availableTo='+asm,db1'

CellCLI> ALTER GRIDDISK sales_CD_03_cell01, sales_CD_04_cell01 -
         availableTo='+asm,db2'

CellCLI> ALTER GRIDDISK sales_CD_05_cell01, sales_CD_06_cell01 -
         availableTo='+asm,db3'
```

Note: The `availableTo='+asm'` argument is mandatory.

---

6. Restart the database and Oracle ASM instances only after the `cellkey.ora` file configuration is complete for all computers. Refer to "Setting Up the cellkey.ora

File on the Client Computers" on page 4-10 for information about configuring the `cellkey.ora` file on the client computers.

See Also:

- "ALTER GRIDDISK" on page 8-22, and "CREATE GRIDDISK" on page 8-42 for additional information about altering and creating grid disks
- "LIST CELLDISK" on page 8-102 and "LIST GRIDDISK" on page 8-107 for additional information about displaying attributes of cell disks and grid disks

Setting Up the cellkey.ora File on the Client Computers

To set up the `cellkey.ora` file on client computers, perform the following procedure:

Caution: If file permissions are not set correctly for the `cellkey.ora` file, then access to the file is open and anyone can read it.

---

1. Set up the `cellkey.ora` file on the Oracle ASM cluster for ASM-scoped security as follows:

   a. Put the `cellkey.ora` file in the `/etc/oracle/cell/network-config/` directory.

   b. Set the permissions on the file to be read-only by the owner using the following command:

   ```
   chmod 600 /etc/oracle/cell/network-config/cellkey.ora
   ```

   c. Set the group to the Oracle ASM administration group using the following command:

   ```
   chown grid:asmadmin /etc/oracle/cell/network-config/cellkey.ora
   ```

2. Set up the `cellkey.ora` file for database-based security as follows:

   a. Put the cellkey.ora file in the `ORACLE_HOME/admin/db_unique_name/pfile` directory.

   b. Set the ownership of the file to the local operating system group. The file should not be owned by the oinstall group, or any other global operating system group.

3. Shut down the database and Oracle ASM instances.

4. Create the `cellkey.ora` file on the client computers. A database client `cellkey.ora` must be located in the `ORACLE_HOME/admin/db_unique_name/pfile/` directory of the database client that you want to restrict access.

   The key value in this file must match the value of the key assigned to the database client.

   The `cellkey.ora` key file must be only readable by the database client that owns the Oracle home.

5. Restart the instances after you have created and edited the `cellkey.ora` files.

Removing Security

This section discusses the removal of security from Oracle Exadata Storage Servers.

- Removing Database-Scoped Security
- Removing Oracle ASM-Scoped Security

> **Note:** You must remove database-scoped security on a grid disk before removing Oracle ASM-scoped security.

Before making updates to the security on cells, you must shut down the database and Oracle ASM instances. After all of the changes to security configuration are complete, start the database and Oracle ASM instances.

Removing Database-Scoped Security

To remove database-scoped security on grid disks, perform the following procedure:

1. Shut down the database and Oracle ASM instances.
2. Remove any database clients named in the `availableTo` attribute of the grid disks for which you want to remove database-scoped security. Example 4–6 shows how to do this for a group of grid disks.

Example 4–6 Removing Database Clients from Grid Disks

```
The following removes all database clients from a group of grid disks
CellCLI> ALTER GRIDDISK sales_CD_01_cell01, sales_CD_02_cell01,                  -
         sales_CD_03_cell01, sales_CD_04_cell01, sales_CD_05_cell01,             -
         sales_CD_06_cell01                                                      -
         availableTo='+asm'

The following removes specific database clients from a group of grid disks
CellCLI> ALTER GRIDDISK sales_CD_04_cell01, sales_CD_05_cell01,                  -
         sales_CD_06_cell01, sales_CD_07_cell01, sales_CD_08_cell01,             -
         sales_CD_09_cell01,                                                     -
         availableTo='+asm,db2,db3'
```

3. If a database client is not configured for security with any other grid disks, then you can remove the key assigned to a database client with the CellCLI `ASSIGN KEY` command as follows:

   ```
   CellCLI> ASSIGN KEY FOR db_client=''
   ```

   In the preceding command, *db_client* is the name of the database client.
4. Remove the `cellkey.ora` file located in the `ORACLE_HOME/admin/db_unique_name/pfile/` directory for the database client.
5. Remove the database from the from the `availableTo` attribute using the following command:

   ```
   ALTER GRIDDISK ALL availableTo='+asm'
   ```

6. Restart the database and Oracle ASM instances.

> **Note:** If you want open security for the grid disks, then you must remove Oracle ASM-scoped security after removing the database-scoped security,

Removing Oracle ASM-Scoped Security

After you have removed database-scoped security, you can remove Oracle ASM-scoped security if you want open security for grid disks on cells. To remove ASM-scoped security, perform the following procedure:

1. Shut down the database and Oracle ASM instances.
2. Remove the Oracle ASM cluster client named in the `availableTo` attribute of the grid disks on cells, as shown in Example 4–7.

Example 4–7 Removing the Oracle ASM Cluster Client from Grid Disks

```
The following removes the ASM cluster client from all grid disks on a cell
CellCLI> ALTER GRIDDISK ALL availableTo=''

The following removes the ASM cluster client from a group of grid disks
CellCLI> ALTER GRIDDISK sales_CD_01_cell01, sales_CD_02_cell01,              -
         sales_CD_03_cell01, sales_CD_04_cell01, sales_CD_05_cell01,         -
         sales_CD_06_cell01                                                  -
         availableTo=''
```

3. If the Oracle ASM cluster client is not configured for security with any other grid disks, then you can remove the key assigned to the Oracle ASM cluster client with the CellCLI `ASSIGN KEY` command, as follows:

   ```
   CellCLI> ASSIGN KEY FOR asm_cluster=''
   ```

   In the preceding command, *asm_cluster* is the name of the Oracle ASM cluster client.
4. Remove the `cellkey.ora` file located in the `/etc/oracle/cell/network-config/` directory on each computer host in the Oracle ASM cluster.
5. Remove Oracle ASM from the availableTo attribute using the following command:

   ```
   ALTER GRIDDISK ALL availableTo=’’
   ```

6. Restart the database and Oracle ASM instances.

   See Also:

   - "ALTER GRIDDISK" on page 8-22
   - "ASSIGN KEY" on page 8-33

5

Maintaining Oracle Exadata Storage Server Software

This chapter explains how to maintain Oracle Exadata Storage Server Software. When changing the fundamental configuration of a cell, such as changing the IP address, host name, and InfiniBand address, note the following:

- Before changing the cell configuration, ensure that all Oracle Automatic Storage Management (Oracle ASM), Oracle Real Application Clusters (Oracle RAC) and database instances that use the cell will not access the cell while you are changing the IP address.
- After changing the cell configuration, ensure that consumers of cell services are correctly reconfigured to use the new connect information of the cell. If Auto Service Request is being used, then deactivate the asset from ASR Manager, and activate the asset with the new IP address.

  See Also: *Oracle Exadata Database Machine Owner's Guide* for information about activating assets

- When changing a cell configuration, change only one cell at a time to ensure that Oracle ASM and Oracle RAC work properly during the changes.

This chapter contains the following topics:

- Using the ipconf Utility
- Changing IP Addresses
- Changing Grid Disk Size
- Oracle Exadata Storage Server Software Validation Tests and Utilities

Caution: All operations in this chapter must be performed with extreme caution and only after you have ensured you have complete backups of the data. If not, then you may experience irrecoverable data loss.

Using the ipconf Utility

The ipconf utility is used to set and change the following on Oracle Exadata Storage Servers. During initial configuration of Oracle Exadata Database Machine, the utility also configures the database servers.

- IP address

- Host name
- NTP server
- Time zone
- DNS name servers
- InfiniBand addresses

The ipconf utility makes a back up copy of the files it modifies. When you rerun the utility, it overwrites the existing backup file. The log file maintains the complete history of every ipconf operation performed.

Table 5–1 lists the ipconf utility options.

Table 5–1 ipconf Options

Option	Description
no option	Utility starts in main editing mode.
`-ignoremismatch`	Starts utility in main editing mode when there is a mismatch between the stored cell configuration and the running configuration.
`-ilom print`	Prints basic ILOM settings.
`-ilom set`	Sets basic ILOM settings.
`-verify [-verbose]`	Verifies the consistency between the stored cell configuration and the running configuration. The `-verbose` option will show all details. If `-verbose` is not used, then only errors are displayed. Success returns zero errors.
`- verify -semantic [-verbose]`	Verifies consistency and checks for access to all DNS and NTP servers. The `-verbose` option will show all details. If `-verbose` is not used, then only errors are displayed. Success returns zero errors.
`-verify -semantic-min [-verbose]`	Verifies the consistency and checks for access to at least one NTP and one DNS server. The `-verbose` option will show all details. If `-verbose` is not used, then only errors are displayed. Success returns zero errors.

Example 5–1 shows the display for the ipconf utility when setting the Sun ILOM interface.

Example 5–1 Using the ipconf Utility to Set the Sun ILOM Interface

```
# ipconf
Logging started to /var/log/cellos/ipconf.log
Interface ib0 is Linked. hca: mxx4_0
Interface ib1 is Linked. mxx4_0
Interface eth0 is Linked. driver/mac: igb/00:00:00:01:cd:01
Interface eth1 is ... Unlinked. driver/mac: igb/00:00:00:01:cd:02
Interface eth2 is ... Unlinked. driver/mac: igb/00:00:00:01:cd:03
Interface eth3 is ... Unlinked. driver/mac: igb/00:00:00:01:cd:04

Network interfaces
Name  State      IP address      Netmask         Gateway         Hostname
ib0   Linked
ib1   Linked
eth0  Linked
```

```
eth1  Unlinked
eth2  Unlinked
eth3  Unlinked
Warning. Some network interface(s) are disconnected. Check cables and switches
         and retry
Do you want to retry (y/n) [y]: n

The current nameserver(s): 10.10.10.10 10.10.10.12 10.10.10.13
Do you want to change it (y/n) [n]:
The current timezone: America/Los_Angeles
Do you want to change it (y/n) [n]:
The current ntp server(s): 10.10.12.10 10.10.12.12 10.10.12.13
Do you want to change it (y/n) [n]:

Network interfaces
Name  State   IP address      Netmask       Gateway        Hostname
eth0  Linked  10.10.13.101  255.255.252.0 10.10.13.1  Managment myg.example.com
bond0 ib0,ib1    192.168.13.101  255.255.252.0        Private myg.example.com

Select interface name to configure or press Enter to continue:

Select canonical fully qualified domain name from the list below
1: myg.example.com
Canonical fully qualified domain name [1]:

Canonical hostname: myg.example.com
Nameservers: 10.10.10.10 10.10.10.12 10.10.10.13
Timezone: America/Los_Angeles
NTP servers: 10.10.12.10 10.10.12.12 10.10.12.13
Network interfaces
Name  State  IP address      Netmask        Gateway        Hostname
eth0  Linked 10.10.13.101 255.255.252.0  10.10.13.1  myg.example.com
bond0 ib0,ib1    192.168.13.101 255.255.252.0
Is this correct (y/n) [y]:

Do you want to configure basic ILOM settings (y/n) [y]: y
Loading configuration settings from ILOM ...s
ILOM Hostname [SUNSP00144FCAE76E]: myg_ilom.example.com
ILOM IP address [10.10.13.201]:
ILOM Netmask [255.255.252.0]:
ILOM Gateway [10.10.13.1]:
ILOM Nameserver or none: 10.10.10.10
ILOM Use NTP Servers (enabled/disabled) [disabled]: enabled
ILOM First NTP server. Fully qualified hostname or ip address or none [0.0.0.0]:
10.10.12.10
ILOM Second NTP server. Fully qualified hostname or ip address or none [0.0.0.0]:
10.10.12.12
Is this correct (y/n) [y]: y
```

Changing IP Addresses

This section discusses changing IP addresses for Exadata Cells and database servers. This activity is required if you reconfigure the Exadata Cell network.

This section includes the following topics:

- Changing the InfiniBand IP Address on Oracle Exadata Storage Server
- Changing the Management IP Address on Exadata Cell

- Changing IP Addresses on Oracle ASM and Oracle Database Servers

---

Note: If you are using SSH to change IP addresses on the InfiniBand interface, then ensure that you connect to SSH using the Ethernet interface.

Changing the InfiniBand IP Address on Oracle Exadata Storage Server

To change the InfiniBand IP address on Oracle Exadata Storage Server, perform the following procedure. During the procedure, it is necessary to shut down the cell services on the server.

1. Verify the grid disks can be taken offline using the following command:

   ```
   cellcli -e list griddisk attributes name,asmdeactivationoutcome
   ```

   All grid disks should show `yes`. If any grid disks are `no`, then determine the reason. Do not continue until all grid disks report `asmdeactivationoutcome='Yes.'`

2. Take the grid disks offline using the following command when all grid disks show `no`, as described in step 1:

   ```
   cellcli -e alter griddisk all inactive
   ```

3. Verify all grid disks are offline using the following command:

   ```
   cellcli -e list griddisk attributes name,asmmodestatus
   ```

   The asmmodestatus should show `OFFLINE` for all grid disks. Only continue if all show `OFFLINE`.

4. Log in as the `root` user, and run the ipconf utility located in the `/usr/local/bin` directory.

   You must shutdown all cell services, and the ocrvottargetd service before using the ipconf utility, as follows:

   a. Log in as the `root` user.

   b. Run the following commands:

      ```
      service celld stop
      service ocrvottargetd stop
      ```

5. Log in to the cell as the `celladmin` user.

6. Restart all cell services using the following command:

   ```
   CellCLI> ALTER CELL RESTART SERVICES ALL
   ```

7. Verify the services are started, and up.

8. Update the `cellip.ora` file with changed address information on each database server that uses the cell.

9. If the cell has iSCSI target devices used for OCR and voting disks, then perform following procedure:

Note: This step is only for environments that use iSCSI target devices. If your environment uses an Oracle ASM disk group for OCR and voting disks, then proceed to step 11.

---

a. For each database server, perform the following procedure:

- Shut down the database
- Shut down Oracle ASM
- Shut down Oracle Clusterware Cluster Ready Services (Oracle Clusterware CRS)
- Repeat shutdown steps for next database server

Steps b through i configure the shared storage to use the IP address.

b. Update the InitiatorName parameter in the `/etc/iscsi/initiatorname.iscsi` file.

Set the InitiatorName parameter using the following format:

```
InitiatorName=iqn.client_name
```

c. Discover the iSCSI devices on the cells using the following command:

```
iscsiadm -m discovery -t sendtargets -p IB_ipaddress
```

In the preceding command, *IB_ipaddress* identifies the private IP address of the target on the InfiniBand interface (BONDIB0). This ensures that all data remains on the private network.

For an Oracle RAC installation you need to use the mirroring mode for OCR and the voting disks, and the command to discover the iSCSI disks provided by three storage cells.

If you have configured both InfiniBand interfaces, then use `BONDIB0` for cells. This ensures availability in the case of an InfiniBand switch failure for OCR and the voting disks.

d. Identify the iSCSI device that you entered as follows:

```
iscsiadm -m node
```

The output is in the format:

```
IB_ipaddress:default_iscsi_port#,1 iqn.yyyy-mm.com.target.domain.com-disk0
```

For example:

```
192.168.50.29:3260,1 iqn.2001-04.com.myg.example.com-disk0
```

`IB_ipaddress` identifies the private IP address of the target on the InfiniBand interface (`ib0` or `ib1`).

e. Run the `service iscsi restart` command to ensure the devices appear in the `/dev/disk/*` directory.

f. Log in to the new iSCSI device. Use the `service iscsi restart` command to automatically log in to all discovered iSCSI devices.

g. Make the new device bindings persistent before assigning them during the Oracle RAC installation.

Add a line in the following format to the `/etc/udev/rules.d/60-persistent-storage.rules` file. If the `60-persistent-storage.rules` file does not exist, then create it.

```
ENV{ID_PATH}==
"ip-192.168.50.29:3260-iscsi-iqn.2001-04.com.sgbeta1s03.us.example.
com-disk0", OWNER="oracle", GROUP="dba", MODE="0640", SYMLINK+="ocrvota%n"
```

Add additional lines for each cell. Call the `symlink ocrvot[a|b|c]` for the three cells and update `ENV{ID_PATH}` accordingly.

h. Run `start_udev` to ensure that the `/dev` devices are available.

i. During the Oracle RAC installation, assign one partition to OCR, and the other to voting disks.

   For OCR disks, use two OCR device names (`/dev/ocrvota1` and `/dev/ocrvotb1`) and set the redundancy mechanism to mirror redundancy.

   For voting disks, use three of the voting disk names (`/dev/ocrvota2`, `/dev/ocrvotb2` and `/dev/ocrvotc2`) to add three voting disks.

j. For each database server, perform the following procedure:

   – Start Oracle Clusterware CRS
   – Start Oracle ASM
   – Start the database
   – Repeat startup steps for the next database server

10. Verify the grid disks are offline using the following command:

```
cellcli -e list griddisk attributes name,asmmodestatus
```

All grid disks should show `OFFLINE`.

11. Activate the grid disks using the following command:

```
cellcli -e alter griddisk all active
```

This command may take time to complete. Monitor the progress of the command using the following command:

```
cellcli -e list griddisk attributes name,asmmodestatus
```

The status of grid disks change from `OFFLINE` to `SYNCING` to `ONLINE`. When all disks are online, the Exadata Cell grid disks have received all pending updates and, if necessary, you may continue with maintenance on other cells.

See Also:

- Example 8–5, "Altering Cell Name, IP Addresses, IP Block, and Cell Number Attributes" on page 8-19
- "Assigning IP Addresses for Oracle Exadata Storage Server Software" on page 2-8
- For additional information about `ALTER DISKGROUP`, see *Oracle Database SQL Language Reference*

Changing the Management IP Address on Exadata Cell

To change the management IP address on Exadata Cell, perform the following procedure:

1. Log in as the `root` user.
2. Shut down any running cell services using the following command:

   ```
   CellCLI> ALTER CELL SHUTDOWN SERVICES ALL
   ```

3. Run the ipconf utility located in the `/usr/local/bin` directory.
4. Enter the Ethernet name when prompted by the ipconf utility.
5. Enter the new IP address when prompted by the ipconf utility.
6. Log in to the cell as the `celladmin` user.
7. Restart all cell services using the following command:

   ```
   CellCLI> ALTER CELL RESTART SERVICES ALL
   ```

8. Verify the services are started, and up.
9. Update the `cellip.ora` file with changed address information on each Oracle ASM node or instance that uses the cell.
10. Verify the grid disks are offline using the following command:

    ```
    cellcli -e list griddisk attributes name,asmmodestatus
    ```

    All grid disks should show `OFFLINE`.
11. Activate the grid disks using the following command:

    ```
    cellcli -e alter griddisk all active
    ```

    This command may take time to complete. Monitor the progress of the command using the following command:

    ```
    cellcli -e list griddisk attributes name,asmmodestatus
    ```

    The status of grid disks change from `OFFLINE` to `SYNCING` to `ONLINE`. When all disks are online, the Exadata Cell grid disks have received all pending updates and, if necessary, you may continue with maintenance on other cells.

See Also:

- Example 8–5, "Altering Cell Name, IP Addresses, IP Block, and Cell Number Attributes" on page 8-19
- "Assigning IP Addresses for Oracle Exadata Storage Server Software" on page 2-8
- For additional information about `ALTER DISKGROUP`, see *Oracle Database SQL Language Reference*

Changing IP Addresses on Oracle ASM and Oracle Database Servers

This section describes how to change the IP addresses on the Oracle ASM and database servers for the network that is used to communicate with Exadata Cell.

1. Shut down the database instances on the database servers where you are changing IP addresses.
2. Shut down the Oracle ASM instances on the database servers that will change IP addresses.
3. Shut down Oracle Clusterware CRS.
4. Change the IP addresses in the `cellinit.ora` file.
5. Restart the database servers that changed IP addresses. This restarts the instances, and Oracle Clusterware CRS, as well as cell affinity.

See Also:

- "Assigning IP Addresses for Oracle Exadata Storage Server Software" on page 2-8
- "Setting Up Configuration Files for a Database Server Host" on page 2-15

Changing Grid Disk Size

You can change the size of a grid disk after configuration. Before increasing the grid disk size, verify the free disk space on the corresponding cell disk. When reducing grid disk size, you reduce Oracle ASM, let the system rebalance, and then reduce storage. When increasing grid disk size, you increase the storage and then Oracle ASM. The following sections contain information on changing grid disk size:

- Reducing Grid Disk Size
- Increasing Grid Disk Size

Reducing Grid Disk Size

The following procedure describes how to reduce the size of the grid disks. In the procedure, the disk groups are `DATA`, `RECO`, `RECO2`, and `SYSTEMDG`.

1. Check the current size of all disk groups using the following command:

```
SQL> SELECT NAME, os_mb, total_mb, free_mb, total_mb-free_mb used_mb FROM \
     V$ASM_DISK ORDER BY name, used_mb;
```

2. Determine the maximum free space in the disk group using the following command:

```
SQL> SELECT MAX(total_mb-free_mb) free_mb FROM V$ASM_DISK WHERE name \
     LIKE 'RECO%' ;
```

3. Check the current size of all grid disks using the following command:

```
dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
offset, size
```

4. Check the current size of the `RECO` grid disks using the following command:

```
dcli -g cell_group  -l root cellcli -e "list griddisk attributes name, \
offset, size where name like \'RECO.*\' "
```

5. Reduce the size of the `RECO` disk group in the Oracle ASM disk group using a command similar to the following:

```
SQL> ALTER DISKGROUP reco RESIZE ALL SIZE new_size REBALANCE POWER 11
```

In the preceding command, *new_size* is the new size for the grid disk.

6. Reduce the size of the DATA disk group so there is space for a high-redundancy disk group name RECO2 using the following command:

   ```
   SQL> ALTER DISKGROUP data RESIZE ALL SIZE new_size REBALANCE POWER 11
   ```

   In the preceding command, *new_size* is the new size for the grid disk.

7. Allow time for Oracle ASM to rebalance the disk group.

   You can query the V$ASM_OPERATION view for the status of the rebalance operation using the following command:

   ```
   SQL> SELECT * FROM V$ASM_OPERATION
   ```

   If no rows are selected, then rebalance is done.

8. Reduce the size of the DATA grid disk to the new size using a command similar to the following:

   ```
   cellcli -e alter griddisk DATA_CD_01_cellname,DATA_CD_02_cellname, \
   DATA_CD_03_cellname, ..., DATA_CD_12_cellname size=new_size
   ```

   In the preceding command, *cellname* is the name of the Exadata Cell, and *new_size* is the new size for the grid disk, such as 140G.

9. Repeat the preceding step for each Exadata Cell, one at a time.

10. Check the grid disks using the following command:

    ```
    dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
    offset, size
    ```

11. Create new grid disks at an offset such that they are above the SYSTEMDG grid disks. The new grid disks should be sized large enough to meet expected needs, but small enough to allow the DATA grid disks to be resized. In this procedure, the new grid disks are RECO2.

    a. Get the offset of the SYSTEMDG grid disks using the following command:

       ```
       dcli -g cell_group  -l root cellcli -e "list griddisk attributes offset \
       where name like \'SYSTEMDG.*\' "
       ```

       The following is an example of the output from the preceding command:

       ```
       cellname: 250.234375G
       ```

    b. Subtract the planned size for RECO2 from the offset, and use the following command to set RECO2:

       ```
       dcli -c cellname -l root "cellcli -e create griddisk all harddisk \
       prefix=RECO2, offset=planned_offset"
       ```

       In the preceding command, *planned_offset* is the offset for the grid disks, such as 140.046875G.

    c. Repeat the preceding step for each Exadata Cell, one at a time.

    d. Check the grid disks using the following command:

       ```
       dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
       ```

```
offset, size
```

12. Create a new disk group, DATA2 using the following command:

```
SQL> CREATE DISKGROUP DATA2 HIGH REDUNDANCY DISK 'o/*/RECO2*' ATTRIBUTE \
'au_size' = '4M', \
'cell.smart_scan_capable' = 'TRUE', \
'compatible.rdbms' = '11.2.0.0', \
'compatible.asm' = '11.2.0.0'
```

13. Resize the DATA grid disks to include the space recovered from the RECO grid disks as follows:

    a. Check the amount of free space and offsets using the following commands:

    ```
    dcli -g cell_group  -l root "su - celladmin -c \"cellcli -e list griddisk \
    where name like \'DATA.*\'\ detail \""

    dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
    offset, size

    dcli -g cell_group  -l root "su - celladmin -c \"cellcli -e list griddisk \
    where name like \'RECO2.*\'\ attributes name,offset,size ""
    ```

    b. Check the cell disk in case some free space is not contiguous using the following commands:

    ```
    dcli -g cell_group -l root "su - celladmin -c \"cellcli -e list celldisk \
    attributes name,freespacemap "\"

    dcli -g cell_group -l root "su - celladmin -c \"cellcli -e list celldisk \
    attributes name,freespace "\"
    ```

    The size of the RECO2 offset is the maximum size for the DATA grid disks.

    c. Set the new size for the DATA grid disks using the following command:

    ```
    dcli -c cellname -l root "su - celladmin -c \"cellcli -e alter griddisk \
    DATA_CD_01_cellname,DATA_CD_02_cellname,... size=210G \""
    ```

    d. Repeat for each Exadata Cell.

14. Resize the DATA disk group in Oracle ASM to add the additional space to the DATA grid disks using the following command.

```
SQL> ALTER DISKGROUP data RESIZE ALL SIZE REBALANCE POWER 11
```

15. Monitor the process using the following command:

```
SQL> SELECT * FROM V$ASM_OPERATION;
```

Increasing Grid Disk Size

The following procedure describes how to increase the size of the grid disks. In the procedure, the disk groups are DATA, RECO, RECO2, and SYSTEMDG.

1. Check the current size of all disk groups using the following command:

```
SQL> SELECT NAME, os_mb, total_mb, free_mb, total_mb-free_mb used_mb FROM \
     V$ASM_DISK ORDER BY name, used_mb;
```

2. Determine the maximum free space in the disk group using the following command:

```
SQL> SELECT MAX(total_mb-free_mb) free_mb FROM V$ASM_DISK WHERE name \
     LIKE 'RECO%' ;
```

3. Check the current size of all grid disks using the following command:

   ```
   dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
   offset, size
   ```

4. Check the current size of the `RECO` grid disks using the following command:

   ```
   dcli -g cell_group  -l root cellcli -e "list griddisk attributes name, \
   offset, size where name like \'RECO.*\' "
   ```

5. Increase the size of the `DATA` grid disk to the new size using a command similar to the following:

   ```
   cellcli -e alter griddisk DATA_CD_01_cellname,DATA_CD_02_cellname, \
   DATA_CD_03_cellname, ..., DATA_CD_12_cellname size=new_size
   ```

   In the preceding command, *cellname* is the name of the Exadata Cell, and *new_size* is the new size for the grid disk, such as `200G`.

6. Repeat the preceding step for each Exadata Cell, one at a time.

7. Increase the size of the `RECO` disk group in the Oracle ASM disk group using a command similar to the following:

   ```
   SQL> ALTER DISKGROUP reco RESIZE ALL SIZE new_size REBALANCE POWER 11
   ```

   In the preceding command, *new_size* is the size is the new size for the grid disk.

8. Reduce the size of the `DATA` disk group so there is space for a high-redundancy disk group name `RECO2` using the following command:

   ```
   SQL> ALTER DISKGROUP data RESIZE ALL SIZE new_size REBALANCE POWER 11
   ```

   In the preceding command, *new_size* is the size is the new size for the grid disk.

9. Allow time for Oracle ASM to rebalance the disk group.

   You can query the `V$ASM_OPERATION` view for the status of the rebalance operation using the following command:

   ```
   SQL> SELECT * FROM V$ASM_OPERATION
   ```

   If no rows are selected, then rebalance is done.

10. Check the grid disks using the following command:

    ```
    dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
    offset, size
    ```

11. Create new grid disks at an offset such that they are above the `SYSTEMDG` grid disks. The new grid disks should be sized large enough to meet expected needs, but small enough to allow the `DATA` grid disks to be resized. In this procedure, the new grid disks are `RECO2`.

    a. Get the offset of the `SYSTEMDG` grid disks using the following command:

       ```
       dcli -g cell_group  -l root cellcli -e "list griddisk attributes offset \
       where name like \'SYSTEMDG.*\' "
       ```

       The following is an example of the output from the preceding command:

       ```
       cellname: 250.234375G
       ```

b. Subtract the planned size for RECO2 from the offset, and use the following command to set RECO2:

```
dcli -c cellname -l root "cellcli -e create griddisk all harddisk \
prefix=RECO2, offset=planned_offset"
```

In the preceding command, *planned_offset* is the offset for the grid disks, such as 140.046875G.

c. Repeat the preceding step for each Exadata Cell, one at a time.

d. Check the grid disks using the following command:

```
dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
offset, size
```

12. Create a new disk group, DATA2 using the following command:

```
SQL> CREATE DISKGROUP DATA2 HIGH REDUNDANCY DISK 'o/*/RECO2*' ATTRIBUTE \
'au_size' = '4M', \
'cell.smart_scan_capable' = 'TRUE', \
'compatible.rdbms' = '11.2.0.0', \
'compatible.asm' = '11.2.0.0'
```

13. Resize the DATA grid disks to include the space recovered from the RECO grid disks as follows:

a. Check the amount of free space and offsets using the following commands:

```
dcli -g cell_group  -l root "su - celladmin -c \"cellcli -e list griddisk \
where name like \'DATA.*\'\ detail \""

dcli -g cell_group -l root cellcli -e list griddisk attributes name, \
offset, size

dcli -g cell_group  -l root "su - celladmin -c \"cellcli -e list griddisk \
where name like \'RECO2.*\'\ attributes name,offset,size ""
```

b. Check the cell disk in case some free space is not contiguous using the following commands:

```
dcli -g cell_group -l root "su - celladmin -c \"cellcli -e list celldisk \
attributes name,freespacemap "\"

dcli -g cell_group -l root "su - celladmin -c \"cellcli -e list celldisk \
attributes name,freespace "\"
```

The size of the RECO2 offset is the maximum size for the DATA grid disks.

c. Set the new size for the DATA grid disks using the following command:

```
dcli -c cellname -l root "su - celladmin -c \"cellcli -e alter griddisk \
DATA_CD_01_cellname,DATA_CD_02_cellname,... size=210G \""
```

d. Repeat for each Exadata Cell.

14. Resize the DATA disk group in Oracle ASM to add the additional space to the DATA grid disks using the following command.

```
SQL> ALTER DISKGROUP data RESIZE ALL SIZE REBALANCE POWER 11
```

15. Monitor the process using the following command:

```
SQL> SELECT * FROM V$ASM_OPERATION;
```

Oracle Exadata Storage Server Software Validation Tests and Utilities

Oracle Exadata Storage Server Software includes the following validation tests that run at boot time:

- Summary of Software and Firmware Components on Exadata Cell
- Exadata Cell Image History
- Validation of the State and Health of the System
- Serial Numbers for System Components
- Diagnostics and Repairs Utilities
- System Diagnostics Data Gathering with sosreports and Oracle OSwatcher
- Serial Console Support
- Linux Kernel Crash Core Files

Summary of Software and Firmware Components on Exadata Cell

The `imageinfo` command located in the `/usr/local/bin/` directory provides a summary of release and status of the software and firmware components on Exadata Cell. The software and firmware components make the cell image. The release and status information is required when working with Oracle Support Services.

Table 5–2 lists the output fields from the `imageinfo` command.

Table 5–2 Description of imageinfo Command Output

Field	Description
Kernel version	Operating system kernel version of the cell.
Cell version	Release version as reported by the CellCLI utility.
Cell rpm version	Cell software version or cell rpm version as reported by the CellCLI utility.
Active image version	Main release version of the overall cell image indicating a specific combination of releases of operating system, core Oracle Exadata Storage Server Software (the cell rpm), and the firmware levels for most key components of the cell. A cell patch usually updates this information. The first five separated fields of the version match the standard way Oracle product releases are identified. The last field is the exact build number of the release. It corresponds to YYMMDD format of the build date.
Active image activated	Date stamp in UTC format when the image on the cell was considered completed, either successfully or unsuccessfully. A cell patch updates the time stamp to indicate the time the cell was patched.
Active image status	Status of the cell image based on the success or failure of a set of self-tests and configuration actions, collectively known as validations. Validations are explained in "Validation of the State and Health of the System" on page 5-15. When this status is undefined, empty or failure, then examine the different validation logs in the `/var/log/cellos` directory to determine the cause for the status.

Table 5–2 (Cont.) Description of imageinfo Command Output

Field	Description
Active system partition on device	Cell operating system root (/) partition device. A typical successful cell patch switches the cell from its active partitions to inactive partitions. Each successful cell patch keeps the cell switching between the active and inactive partitions. There are few occasions when the cell patch does not switch partitions. These are rare, and are known as in-partition patches.
In partition rollback	Some cell patches do not switch the partitions. These are in-partition patches. This field indicates whether there is enough information to roll back such patch.
Cell boot usb partition	Exadata Cell boot and rescue USB partition.
Cell boot usb version	Version of the software on the boot USB. On a healthy cell this release must be identical to the value of the Active image version line.
Inactive image version	Version of the cell before the most-recent patch was applied. If the cell is brand new, then this is
Inactive image activated	Time stamp for activation of the inactive image. This field is similar to Active image activated field.
Inactive image status	Status of the inactive image. This field is similar to the status of the active image.
Inactive system partition on device	The root (/) file system partition for the inactive image.
Inactive software partition on device	Oracle Exadata Storage Server Software file system partition, `/opt/oracle`, for the inactive image.
Boot area has rollback archive for version	For a patched cell using non in-partition cell patch, this indicates whether there is a suitable back up archive that can be used to roll the cell back to the inactive image version. Existence of this archive is necessary but not sufficient for rolling back to inactive version of the cell image.
Rollback to inactive partition	Summary indicator for a non-in-partition patched cell indicating whether rollback can be run on the cell to take it back to inactive version of the cell image. On a new cell, this field is empty or has the value `undefined`.

The following is an example of the output from the `imageinfo` command:

```
Kernel version: 2.6.18-194.3.1.0.3.el5 #1 SMP Tue Aug 31 22:41:13 EDT 2010 x86_64
Cell version: OSS_MAIN_LINUX.X64_101105
Cell rpm version: cell-11.2.2.1.1_LINUX.X64_101105-1

Active image version: 11.2.2.1.1.101105
Active image activated: 2010-11-06 21:52:08 -0700
Active image status: success
Active system partition on device: /dev/md5
Active software partition on device: /dev/md7

In partition rollback: Impossible

Cell boot usb partition: /dev/sdm1
Cell boot usb version: 11.2.2.1.1.101105

Inactive image version: 11.2.1.3.1
Inactive image activated: 2010-08-28 20:01:30 -0700
```

```
Inactive image status: success
Inactive system partition on device: /dev/md6
Inactive software partition on device: /dev/md8

Boot area has rollback archive for the version: 11.2.1.3.1
Rollback to the inactive partitions: Possible
```

Exadata Cell Image History

The `imagehistory` command lists the version history for Exadata Cell. For example, if a cell was updated from release 11.2.1.2.6 to release 11.2.1.3.1, and then updated to release 11.2.2.2.0, the `imagehistory` command will display this history. The following is an example of the output:

```
# imagehistory
Version                 : 11.2.2.1.0
Image activation date   : 2010-08-18 14:16:37 -0700
Imaging mode            : fresh
Imaging status          : success

Version                 : 11.2.2.1.0.100913
Image activation date   : 2010-09-13 20:20:18 -0700
Imaging mode            : out of partition upgrade
Imaging status          : failure
```

Validation of the State and Health of the System

Validation framework is a set of validation tests that run at boot time at the rc.local level. The logs for the tests are available in the `/var/log/cellos/validations` directory. Validation framework also runs different tests under certain conditions, such as on first boot after recovery of an Exadata Cell using the rescue and recovery functionality of the CELLBOOT USB flash drive, or when patching an Exadata Cell.

In addition, health check validations are a set of quick health checks on the system on each boot, such as basic health of the disks, and report the status. If a validation fails, then you should examine the log file for the cause as it may indicate potential problem requiring your attention.

Automatic patch rollback occurs if one or more validation checks fail after patch application. Refer to the documentation for the specific patch.

Check for any failures reported in the `/var/log/cellos/vldrun.first_boot.log` file after the first boot configuration. For all subsequent boots, the `/var/log/cellos/validations.log` file contains information about failed validations. For each failed validation, perform the following procedure:

1. Look for `/var/log/cellos/validations/`*`failed_validation_name`*`.SuggestedRemedy` file. The file exists only if the validation process has identified some corrective action. Follow the suggestions in the file to correct the cause of the failure.
2. If the `SuggestedRemedy` file does not exist, then examine the log file for the failed validation in `/var/log/cellos/validations` to track down the cause, and correct it as needed.

Serial Numbers for System Components

You may need to provide the serial numbers for the system components when contacting Oracle Support Services. Serial numbers for system components can be determined by using the following procedure:

1. Log in as the `root` user.
2. Enter the following command:

```
/opt/oracle.SupportTools/CheckHWnFWProfile -S
```

In addition, each time the system is booted, the serial numbers are written to the `/var/log/cellos/validations/SerialNumbers` file. This file can be used as a historical record of the serial numbers. The file also contains configuration information for some components.

Note: Use the CheckSWProfile.sh utility to determine the serial numbers for the InfiniBand switches. Refer to "The CheckSWProfile.sh Utility" on page 5-17 for additional information about the utility.

Diagnostics and Repairs Utilities

Oracle Exadata Storage Server Software includes utilities for diagnostics and repair of Exadata Cell. The utilities help diagnose and repair problems that may occur during the normal life cycle of Exadata Cells. The utilities are in the `/opt/oracle.SupportTools` directory.

Note: All utilities must be run as the `root` user from the `/opt/oracle.SupportTools` directory.

The CheckHWnFWProfile Utility

This utility checks that the system meets the required hardware and firmware specifications, and reports any mismatches.

Table 5–3 CheckHWnFWProfile Utility Commands

Command	Description
`./CheckHWnFWProfile -c strict` `./CheckHWnFWProfile -c loose`	Check the hardware. The checks are run as strict or loose. The `strict` option flags an error when all disks do not have an identical model number or an identical firmware version. The `loose` option does not consider mixed disks as an error condition.
`./CheckHWnFWProfile -d`	View the existing hardware and firmware versions on the system.
`./CheckHWnFWProfile -h`	View help and utility usage.
`./CheckHWnFWProfile -S`	List serial numbers. The list includes the following serial numbers: ▪ System ▪ Disk controller ▪ Each disk ▪ Infiniband HCA Depending on the system, serial numbers for all the memory (RAM) modules may be included. To obtain serial numbers for the InfiniBand switches, refer to the CheckSWProfile.sh utility in "The CheckSWProfile.sh Utility".
`./CheckHWnFWProfile -s`	View the expected hardware and firmware.

The CheckSWProfile.sh Utility

The CheckSWProfile.sh utility checks that the system has the correct InfiniBand software, and reports any mismatch. It can also check the software and firmware versions on the InfiniBand switches, and report whether they meet the specifications.

Table 5–4 CheckSWProfile.sh Utility Commands

Command	Description
`./CheckSWProfile.sh -c`	Run the check.
`./CheckSWProfile.sh -h`	View help and usage.
`./CheckSWProfile.sh -I` *`list_of_switches`*	Check the InfiniBand switch software and firmware version. The *list_of_switches* value is a comma-delimited list of InfiniBand switch management IP addresses or server names. It does not include spaces.
`./CheckSWProfile.sh -S list_of_switches`	List the serial numbers for the InfiniBand switches. You need the `admin` password for Voltaire switches, and the `root` password for the Sun switches.

The diagnostics.iso Utility

The diagnostics.iso utility may be used to boot the server to diagnose serious problems when no other way exists to analyze the system due to damage to the system, and its CELLBOOT USB flash drive. You must use this utility only with Oracle Support Services guidance. The `root` password should be available to Oracle Support Services, as needed.

The ibdiagtools Utilities

The most useful utilities are the verify-topology, checkbadlinks.pl, and infinicheck utilities. The verify-topology utility checks the correctness and health of InfiniBand connections. For example, it can determine if both cables from the server go to the same switch in the Oracle Exadata Database Machine. When both cables go to the same switch, the server loses the ability to fail over to another switch when the first InfiniBand switch fails.

The checkbadlinks.pl utility reports the links that are operating 5 Gbps. This is usually an indication that the cables are loose, and need to be reseated.

The infinicheck utility reports the base InfiniBand performance between servers in Oracle Exadata Database Machine, such as expected minimum throughput between the database server and cell, cell and cell, and database server and another database server. This utility can help identify potential issues in the InfiniBand fabric.

See Also:

- For detailed information about the ibdiagtools utilities, refer to the `README.txt` file in the `/opt/oracle.SupportTools/ibdiagtools/` directory.
- Sample outputs from each utility are included in the `/opt/oracle.SupportTools/ibdiagtools/SampleOutputs.txt` file.

The make_cellboot_usb Utility

The make_cellboot_usb utility allows you to rebuild a damaged CELLBOOT USB flash drive. Do not have more than one USB flash drive connected to the system when running this utility. It builds on the first discovered USB flash drive on the system.

> **Note:** This utility can only be used on Exadata Cell.

- To see what is done before rebuilding the USB flash drive, run the following command:

```
./make_cellboot_usb -dry -verbose
```

- To rebuild the USB flash drive, run the following command:

```
./make_cellboot_usb -verbose
```

System Diagnostics Data Gathering with sosreports and Oracle OSwatcher

On every boot, systemwide configuration information is collected by the sosreport utility, and stored in the `/var/log/cellos/sosreports` directory. The information can be used to help diagnose problems. You can generate a new sosreport by running the following command as the `root` user. The script starts collecting the report 30 minutes after entering the command.

```
/opt/oracle.cellos/vldrun -script sosreport
```

In addition, the `/opt/oracle.oswatcher/osw` directory contains the Oracle OSWatcher system data gathering and reporting utilities. Gathered data is then stored in archive subdirectories. The following data is gathered at different intervals. Refer to the actual data files for OSWatcher system data.

Data is retained for 168 hours (seven days) in bzip2-compressed files in the appropriate archive subdirectory.

The `osw` directory has a readme file, and utilities to report and graph some types of collected data.

Starting and Stopping the OSWatcher Utility

The OSWatcher utility can be run at any time. To start the OSWatcher utility, perform the following procedure:

1. Log in as the `root` user.
2. Run the following command:

```
/opt/oracle.cellos/vldrun -script oswatcher
```

To stop the OSWatcher utility, perform the following procedure:

1. Log in as the `root` user.
2. Change to the `/opt/oracle.oswatcher/osw` directory.
3. Run the following command:

```
./stopOSW.sh
```

Serial Console Support

The cells and database server of Oracle Exadata Database Machine are configured to provide serial console access. The serial console is useful when taking Linux kernel traces or creating crash dump files to help diagnose severe malfunctions. To access the serial console, perform the following procedure:

- For Sun Oracle Exadata Storage Server: Use the `start /SP/console` command. To stop using the console, use the `stop /SP/console` command.
- For HP Oracle Exadata Storage Server: Run the `VSP` command.

Linux Kernel Crash Core Files

The cells and database servers of Oracle Exadata Database Machine are configured to generate Linux kernel crash core files in the `/var/crash` directory, when there is a Linux crash. The crash utility can be used to analyze the crash files. The crash files are automatically removed by the OSWatcher utility so that the files do not occupy more than 10 percent of the free disk space on the file system. Older crash files are removed first.

6

Managing I/O Resources

I/O Resource Management is a tool for managing how multiple databases and the workloads within the databases share the I/O resources of Oracle Exadata Storage Server Software (Exadata Cell). To manage workloads within a database, the Database Resource Manager (DBRM) has been enhanced to work with I/O Resource Management to provide database resource management.

This chapter contains the following topics:

> **See Also:** *Oracle Database Administrator's Guide* for information about managing resource allocation

Overview of Resource Management

Storage is often shared by multiple types of workloads and databases. Shared storage has several benefits over dedicated storage. One benefit is that sharing lowers administration costs by reducing the number of storage systems that need to be maintained. Another benefit is that sharing leads to much more efficient use of storage, both from a space and a bandwidth standpoint. When a storage system is dedicated to a single database, the administrator must size the storage system based on the database's peak anticipated load and size. This practice leads to unused I/O bandwidth and space for some databases and, if the estimate was not large enough, then there is insufficient bandwidth and space for other databases. The correct balance of storage resources across the multiple databases is seldom achieved because real world workloads are very dynamic.

On the other hand, running multiple types of workloads and databases on shared storage often leads to performance problems. For example, large parallel queries on one production data warehouse can impact the performance of critical queries on another production data warehouse. Also, a data load on a data warehouse can impact the performance of critical queries also running on it. You can mitigate these problems by over-provisioning the storage system, but this diminishes the cost-savings benefit of shared storage. You can also schedule non-critical tasks at off-peak hours, but this manual process is laborious. It becomes impossible when databases have different administrators who do not coordinate their activities.

I/O Resource Management allows workloads and databases to share Exadata Cell according to user-defined policies. To manage workloads within a database, you can define database resource plans, using Database Resource Manager which has been enhanced to manage Exadata Cell I/O resources. To manage multiple databases, you can define interdatabase plans.

Understanding I/O Resource Management

I/O Resource Management (IORM) manages the Exadata Cell I/O resources on a per-cell basis. Whenever the I/O requests start to saturate a cell's capacity, IORM schedules incoming I/O requests according to the configured resource plans. IORM schedules I/O by immediately issuing some I/O requests and queuing others. The I/O requests that are immediately issued are from workloads that have not exceeded their resource allocation, according to the resource plans. The I/O requests that are queued are from workloads that have exceeded their resource allocation. They are issued when their workload no longer exceeds its resource allocation or when the cell is operating below capacity. When the cell is operating below capacity, IORM does not queue I/O requests.

For example, if a production database and test database are sharing Exadata Cell, you can configure resource plans that give priority to the production database. In this case, whenever the test database load would affect the production database performance, IORM schedules the I/O requests such that the production database I/O performance is not impacted. This means that the test database I/O requests are queued until they can be issued without disturbing the production database I/O performance.

IORM provides many features for managing resource allocations. Each feature can be used independently or in conjunction with other features.

Database resource management enables you to manage workloads within a database. Database resource management is configured at the database level, using Database Resource Manager to create database resource plans. You should use this feature if you have multiple types of workloads within a database and you need to define a policy for specifying how these workloads share the database resource allocation. If only one database is using the Exadata Cell, then this is the only IORM feature that you need.

Interdatabase resource management enables you to manage multiple databases. Interdatabase resource management is configured by using the CellCLI utility to create interdatabase plans. The interdatabase plans specify a resource allocation for each database. You should use this feature if you have multiple databases using Exadata Cell.

If an interdatabase plan has been configured, then each database can have a database plan and a resource allocation. A database resource plan specifies how the database resource allocation should be divided across its workloads, or consumer groups. If a database does not have a database resource plan enabled, then the database resource allocation is not be divided, and all I/O requests from the database are treated as a single workload.

Category resource management is an advanced feature. It is useful when Exadata Cell is hosting multiple databases, and you want to allocate resources primarily by the category of the work being done. For example, suppose all databases have three categories of workloads: OLTP, reports, and maintenance. To allocate the I/O resources based on these workload categories you would use category resource management.

The concept of maximum utilization limit (`limit`) is supported by I/O Resource Management. In addition to specifying the resource allocation values, you can also provide a maximum utilization limit for a given database.This directive ensures that the database never utilizes I/O resources beyond the specified limits. For example, if a production and test database are sharing the Exadata cell, then set a maximum utilization limit for the test database to limit the I/O utilization for the test database.

If a maximum utilization limit is specified, then excess capacity is never used by the databases. It is possible that the disks are running below full capacity when maximum utilization limits are specified.

I/O Resource Management supports flash cache management. The `ALTER IORMPLAN flashcache` attribute can be set to `off` to prevent databases from using the flash cache. This allows flash cache to be reserved for mission-critical databases, especially in consolidated environments.

About Database Resource Management

For each database, you can use the Database Resource Manager for the following tasks:

- Create resource consumer groups

  A database often has many types of workloads. These workloads may differ in their performance requirements and the amount of I/O that they issue. Resource consumer groups provide a way to group sessions that comprise a particular workload. For example, if your database is running four different applications, then you can create four consumer groups, one for each application. If your data warehouse has three types of workloads, such as critical queries, normal queries, and ETL (extraction, transformation, and loading), then you can create a consumer group for each type of workload.

- Map user sessions to consumer groups

  Once you have created the consumer groups, you must create rules that specify how sessions are mapped to consumer groups. Database Resource Manager allows you to create mapping rules based on session attributes such as the Oracle user name, the service that the session used to connect to the database, client machine, client program name, client user name, and so on. If you are creating consumer groups for each application and each application has a dedicated service, then you should create mapping rules based on service names. If you want to dedicate a consumer group to a particular set of users, then you should create mapping rules based on their user names. Sessions that are not explicitly assigned to a consumer group are placed in the OTHER_GROUPS consumer group.

- Create resource plans

  The database resource plan, also known as an intradatabase resource plan, specifies how CPU and I/O resources are allocated among consumer groups in its database. The resource plan is created using Database Resource Manager. It contains a resource allocation directive for each consumer group, which consists of a percentage and a level. You can specify up to eight levels. Consumer groups at level 2 get resources that were not allocated at level 1 or were not consumed by a consumer group at level 1. Similarly, consumer groups at level 3 are allocated resources only when some allocation remains from levels 1 and 2. The same rules apply to levels 4 through 8. Multiple levels not only provide a way of prioritizing, they also provide a way of explicitly specifying how all primary and leftover resources are to be used. You can construct resource plans that allocate resources across consumer groups using percentages, priorities, or a combination of the two.

  You can also specify a maximum utilization limit for a consumer group. This works in the same way as a maximum utilization limit for a database, and limits the I/O utilization for the consumer group to the specified value.

- Enable a resource plan

  A database resource plan can be manually enabled with the RESOURCE_MANAGER_PLAN initialization parameter or automatically enabled with the job scheduler window.

When you set a database resource plan on the database, a description of the plan is automatically sent to each cell. When an Oracle Real Applications Clusters (Oracle RAC) database uses Exadata Cell, all instances in the Oracle RAC cluster must be set to the same resource plan. When a new cell is added or an existing cell is restarted, the current plan of the database is automatically sent to the cell. The resource plan is used to manage resources on both the database server and cells.

Background I/Os are scheduled based on their priority relative to the user I/Os. For example, redo writes, and control file reads and writes are critical to performance and are always prioritized above all user I/Os. The database writer process (DBWR) writes are scheduled at the same priority level as the user I/Os. If a resource plan is not enabled for a database, then all user I/Os are treated equally, and background I/Os are treated as described in this paragraph.

Oracle provides several predefined plans. The most commonly used are `mixed_workload_plan`, `dss_plan`, and `default_maintenance_plan`.

> **See Also:** *Oracle Database Administrator's Guide* for additional information about consumer groups, assigning user sessions to consumer groups, and managing resource plans

About Interdatabase Resource Management

Interdatabase resource management is managed with an interdatabase plan. An interdatabase plan specifies how resources are allocated among multiple databases for each cell. The directives in an interdatabase plan specify allocations to databases, rather than consumer groups. The interdatabase plan is configured and enabled with the CellCLI utility at each cell.

For example, you might specify that 70 percent of the I/O resources are allocated to database SALES, 30 percent to HR, and any of their unused allocations are reallocated to the TEST_SALES database. The interdatabase plan is similar to a database resource plan, in that each directive consists of an allocation amount and a level from 1 to 8. For a given plan, the total allocations at any level must be less than or equal to 100 percent. An interdatabase plan differs from a database resource plan in that it cannot contain subplans and only contains I/O resource directives. Only one interdatabase plan can be active on a cell at any given time.

Exadata Cell uses the IORM and database resource plans together to allocate I/O resources.

- First, the interdatabase plan allocates the I/O resources to individual databases. Any unused resources are reallocated to other databases, as specified by the plan. This is similar to database resource plans.
- Next, the database resource plan for each database allocates the I/O resources to consumer groups. If a database does not have an active database resource plan, all user I/Os are treated the same. Background I/Os are automatically prioritized relative to the user I/Os based on their importance.

As a best practice, you should create a directive for each database that is using the same cell storage. To ensure that any database without an explicit directive can be managed, create an allocation named OTHER. Databases without explicit directives are managed using the allocation of the OTHER group directive.

Another use case for interdatabase resource management is for consolidation. For example, consolidation of four different applications on the same Oracle Exadata Storage Servers. All the applications have similar priority requirements, and are equally allocated 25% of the I/O resources.

However, you can isolate each application in case another application sees a sudden increase in its workload. Use the maximum utilization limit to isolate the applications, such as specifying a maximum utilization limit of 40% for each application. In such a scenario, each application can utilize a maximum of 40% of the I/O resources, and does not completely monopolize the system. Maximum utilization limit is useful in such consolidation scenarios.

Note the following when managing resource plans:

- If Exadata Cell is only hosting one database, then an interdatabase plan is not needed.
- If an interdatabase plan is not specified, then all databases receive an equal allocation.
- If only one database is mapped to the OTHER directive and all other databases have an explicit directive, then Exadata Cell uses the database resource plan of that database to determine how the allocation of the OTHER database is redistributed among the consumer groups in that database.
- If multiple databases are mapped to the OTHER directive, then Exadata Cell does not use database resource management for these databases. All of the I/O requests are treated the same.

About Category Resource Management

Database Resource Manager enables you to specify a category for every consumer group. While consumer groups represent collections of users within a database, categories represent collections of consumer groups across all databases. You can manage I/O resources based on categories by creating a category plan. For example, you can specify precedence to consumer groups in the interactive category over consumer groups in the batch category for all databases sharing Exadata Cell. The predefined categories provided in Oracle Database are described in Table 6–1, along with sample percentages.

You can add any number of categories, or modify the predefined categories. You should map consumer groups to the appropriate category for all databases that use the same cell storage. Any consumer group without an explicitly specified category defaults to the OTHER category.

When a category plan is enabled, the category plan is first used to allocate resources among the categories. For each category that is selected, the interdatabase plan is used to select those databases that have consumer groups with the selected category. Finally, the database resource plan of the selected database is used to select one of its consumer groups.

Category plans are configured and enabled using the CellCLI utility on the cell. Only one category plan can be enabled at a time. A sample category plan is shown in Table 6–1.

Table 6–1 Sample Category Plan

Category Name	Category Description	Level 1 (%)	Level 2 (%)	Level 3 (%)
ADMINISTRATIVE	For extremely high-priority work, such as urgent administrative tasks. This category is required.	80		
INTERACTIVE	For high-priority, performance-sensitive work, such as OLTP transactions.		70	
BATCH	For low-priority work, such as noncritical reports and backup.			70
MAINTENANCE	For low-priority work, such as automated tasks.			10
OTHER	For all consumer groups that do not have a category label or reference a category that is not in the current category plan. This category is required.			20

The sample plan shown in Table 6–1 prioritizes administrative activity across all databases. It also prioritizes interactive activity over batch, maintenance, and other activities. In the sample plan, the following are the resource allocations:

- Level 1 is given 80 percent of the I/O resources. The ADMINISTRATIVE category is the only category in level 1.
- Level 2 is given all resources that were unallocated or unused by level 1. In this example, level 2 is given 20 percent of the I/O resources and any resources unused by the ADMINISTRATIVE category. The INTERACTIVE category gets 70 percent of the level 2 amount.
- Level 3 categories are given the remaining resources, including those not used by the INTERACTIVE category. Of the remaining resources, the BATCH category gets 70 percent, the OTHER category gets 20 percent, and the MAINTENANCE category gets 10 percent.

All administrative consumer groups in all databases should be mapped to the ADMINISTRATIVE category. All high-priority user activity, such as consumer groups for important OLTP transactions and time-critical reports, should be mapped to the INTERACTIVE category. All low-priority user activity, such as reports, maintenance, and low-priority OLTP transactions, should be mapped to the BATCH, MAINTENANCE, and OTHER categories.

See Also: "Administering Database Resource Management" on page 6-8.

About Consumer Groups and Resource Plans

Oracle provides out-of-the-box consumer groups and resource plans specifically designed for data warehouses that use Oracle Exadata Storage Server Software. These plans can be modified to suit the needs of your environment.

The following are the consumer groups for data warehouses:

- ETL_GROUP: Consumer group for ETL (extract, transform, and load) jobs.

- DSS_GROUP: Consumer group for non-critical decision support system (DSS) queries.
- DSS_CRITICAL_GROUP: Consumer group for critical DSS queries.

The following are the resource plans for data warehouses:

- DSS_PLAN Resource Plan
- ETL_CRITCAL_PLAN Resource Plan

DSS_PLAN Resource Plan

The DSS_PLAN resource plan is designed for data warehouses that prioritize critical DSS queries over non-critical DSS queries and ETL jobs. In this plan, SYS_GROUP has the highest priority, followed by DSS_CRITICAL_GROUP, DSS_GROUP, and then a combination of ETL_GROUP and BATCH_GROUP. No consumer group is allowed to consume all of the bandwidth.

Table 6–2 DSS_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
SYS_GROUP	75			
DSS_CRITICAL_GROUP		75		
DSS_GROUP			75	
ETL_GROUP				45
BATCH_GROUP				45
ORA$DIAGNOSTICS		5		
ORA$AUTOTASK_SUB_PLAN		5		
OTHER_GROUPS				10

As shown in Table 6–2, the DSS_CRITICAL_GROUP group is only allocated 75 percent at level 2. Any unused allocation goes to the next level, not to other consumer groups at the same level. That means that if the DSS_CRITICAL_GROUP group does not completely consume its allocation, then the allocation is not given to the ORA$DIAGNOSTICS or ORA$AUTOTASK_SUBPLAN groups at the same level. According to the plan definition, the allocation is given to the DSS_GROUP group at level 3.

ETL_CRITCAL_PLAN Resource Plan

The ETL_CRITICAL_PLAN prioritizes ETL over DSS queries. In this plan, the SYS_GROUP group is given 75 percent of the bandwidth. The remaining bandwidth is divided between the other consumer groups in the ratios specified by the level 2 allocations. The ETL_GROUP and DSS_CRITICAL_GROUP groups have a higher allocation (35 percent) than the DSS_GROUP and BATCH_GROUP groups (10 percent).

Table 6–3 ETL_CRITICAL_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
SYS_GROUP	75			
DSS_CRITICAL_GROUP		35		

Table 6–3 (Cont.) ETL_CRITICAL_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
DSS_GROUP		10		
ETL_GROUP		35		
BATCH_GROUP		10		
ORA$DIAGNOSTICS		3		
ORA$AUTOTASK_SUB_PLAN		3		
OTHER_GROUPS		3		

Administering IORM

This section describes the tasks for I/O Resource Management (IORM). To perform the tasks, use the `DBMS_RESOURCE_MANAGER` package to define database resource plans on the database hosts, and the CellCLI utility to specify the IORM and category plans for each cell.

This section contains the following topics:

- Activating and Deactivating IORM
- Administering Database Resource Management
- Administering Interdatabase Resource Management
- Listing an I/O Resource Management Plan
- Verifying the Configuration of I/O Resource Management
- Example of Using the Plans

> **See Also:** Chapter 8, "Using the CellCLI Utility" for additional information about the CellCLI utility

Activating and Deactivating IORM

IORM is active by default. In the default state, IORM can only be used to manage flash cache. To manage I/O resources, set the `objective` option, which specifies the optimization mode for IORM.

Once the objective has been set, IORM manages the I/O resources in one of two ways. First, it manages I/O if any database has set a database resource plan. Second, it manages I/O if an interdatabase or category plan is configured.

IORM is deactivated by issuing the `ALTER IORMPLAN INACTIVE` command on each cell. When IORM is deactivated, it does not manage I/O resources or flash cache, even if a database resource plan is set or an interdatabase plan is configured. To reactivate IORM use the CellCLI `ALTER IORMPLAN ACTIVE` command on each cell.

Administering Database Resource Management

To set up database resource management, you must use Database Resource Manager to configure the consumer groups, assign sessions to consumer groups, create a database resource plan, and enable it.

This section contains the following topics:

- Setting Up Consumer Groups and Categories

- Assigning Sessions to Consumer Groups
- Creating a Database Plan
- Enabling a Database Resource Plan
- Managing Fast File Creation
- Managing Data Import
- Managing Oracle Recovery Manager Backups and Copies

Setting Up Consumer Groups and Categories

Consumer groups and categories are set up with the procedures in the PL/SQL DBMS_RESOURCE_MANAGER package. You can create new consumer groups and categories, or use one of the predefined consumer groups or categories. You do not need to set up categories if you are not planning on using a category plan.

> **Note:** Consumer groups and categories are created in the database and cannot be created explicitly on a cell.

Before running the DBMS_RESOURCE_MANAGER procedures for administering consumer groups and categories, you must first create a pending area. You must have the system privilege ADMINISTER_RESOURCE_MANAGER to run the procedures in the DBMS_RESOURCE_MANAGER PL/SQL package.

The following PL/SQL commands are used with consumer groups and categories:

- To manage categories: CREATE_CATEGORY(), DELETE_CATEGORY(), and UPDATE_CATEGORY()
- To manage consumers groups: CREATE_CONSUMER_GROUP() and UPDATE_CONSUMER_GROUP()
- To assign consumer groups to categories: CREATE_CONSUMER_GROUP() or UPDATE_CONSUMER_GROUP()

Example 6–1 shows how to set up consumer groups and categories in a database. In the example, the MAINTENANCE category is predefined, and is not created in the example.

Example 6–1 Setting Up Consumer Groups and Categories with PL/SQL in the Database

```
BEGIN
  DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();

  DBMS_RESOURCE_MANAGER.CREATE_CATEGORY(
     CATEGORY => 'dss',
     COMMENT => 'DSS consumer groups');

  DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP(
     CONSUMER_GROUP => 'critical_dss',
     CATEGORY => 'dss',
     COMMENT => 'performance-critical DSS queries');

  DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP(
     CONSUMER_GROUP => 'normal_dss',
     CATEGORY => 'dss',
     COMMENT => 'non performance-critical DSS queries');
```

```
  DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP(
     CONSUMER_GROUP => 'etl',
     CATEGORY => 'maintenance',
     COMMENT => 'data import operations');

  DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

In addition to the consumer groups that you set up, the database contains predefined consumer groups. The DBA_RSRC_CONSUMER_GROUPS view displays information about consumer groups, and the DBA_RSRC_CATEGORIES view displays information about categories in the database. Example 6–2 shows a query on the DBA_RSRC_CONSUMER_GROUPS view.

Example 6–2 Consumer Groups and Categories in an Oracle Database

```
SQL> SELECT consumer_group, category FROM DBA_RSRC_CONSUMER_GROUPS where
     consumer_group not like 'ORA%' ORDER BY category;

CONSUMER_GROUP                 CATEGORY
------------------------------ ------------------------------
SYS_GROUP                      ADMINISTRATIVE
ETL_GROUP                      BATCH
BATCH_GROUP                    BATCH
DSS_GROUP                      BATCH
CRITICAL_DSS                   DSS
NORMAL_DSS                     DSS
DSS_CRITICAL_GROUP             INTERACTIVE
INTERACTIVE_GROUP              INTERACTIVE
ETL                            MAINTENANCE
LOW_GROUP                      OTHER
OTHER_GROUPS                   OTHER

CONSUMER_GROUP                 CATEGORY
------------------------------ ------------------------------
AUTO_TASK_CONSUMER_GROUP       OTHER
DEFAULT_CONSUMER_GROUP         OTHER

13 rows selected
```

See Also:

- *Oracle Database Administrator's Guide* for additional information about managing consumer groups and categories
- *Oracle Database Reference* for additional information about database views

Assigning Sessions to Consumer Groups

You can assign a session to a resource consumer group manually, or automatically using consumer group mapping rules. For both approaches, you must give explicit permission for a user to switch to a consumer group. In order to control which consumer groups a user can switch to, use the PL/SQL procedure DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP().

The consumer group mapping rules are based on session attributes such as the user name, the name of the service that the session used to connect to the database, and the name of the client program. To create a consumer group mapping rule, use the SET_

CONSUMER_GROUP_MAPPING procedure, as shown in Example 6–3. Before running the SET_CONSUMER_GROUP_MAPPING procedure, you must first create a pending area.

Example 6–3 Creating Consumer Group Mapping Rules, Based on Service and User Name

```
BEGIN
DBMS_SERVICE.CREATE_SERVICE('SALES', 'SALES');
DBMS_SERVICE.CREATE_SERVICE('AD_HOC', 'AD_HOC');

DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING
     (DBMS_RESOURCE_MANAGER.ORACLE_USER, 'SYS', 'CRITICAL_DSS');
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING
     (DBMS_RESOURCE_MANAGER.SERVICE_NAME, 'SALES', 'CRITICAL_DSS');
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING
     (DBMS_RESOURCE_MANAGER.SERVICE_NAME, 'AD_HOC', 'NORMAL_DSS');

DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();

DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP (
   GRANTEE_NAME   => 'PUBLIC',
   CONSUMER_GROUP => 'CRITICAL_DSS',
   GRANT_OPTION   =>  FALSE);
DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP (
   GRANTEE_NAME   => 'PUBLIC',
   CONSUMER_GROUP => 'NORMAL_DSS',
   GRANT_OPTION   =>  FALSE);
END;
/
```

You can also manually switch a session to a particular consumer group, using the PL/SQL DBMS_RESOURCE_MANAGER.SWITCH_CONSUMER_GROUP_FOR_USER() or SWITCH_CONSUMER_GROUP_FOR_SESS() procedures.

> **See Also:** *Oracle Database Administrator's Guide* for additional information about the following:
>
> - Assigning sessions to consumer groups
> - Complete list of session attributes

Creating a Database Plan

Database resource plans, also known as intradatabase plans, are created using the PL/SQL procedures DBMS_RESOURCE_MANAGER.CREATE_PLAN() and CREATE_PLAN_DIRECTIVE(). You must have the system privilege ADMINISTER_RESOURCE_MANAGER to run the procedures in the DBMS_RESOURCE_MANAGER PL/SQL package. This resource plan manages both CPU resources on database instances and I/O resources on the cells.

One scenario is for multiple applications sharing a database where the I/O resources should be divided across the applications using a particular ratio. For example, there are three applications named SALES, FINANCE, and MARKETING. You would like the I/O resources to be allocated as 60 percent, 25 percent, and 10 percent, respectively, with the remaining 5 percent allocated to any sessions that do not map into these consumer groups. In this scenario, you would create a consumer group for each application, and then create a single-level resource plan and specify the percentage of I/O resources for each consumer group. This allocation is actually the minimum I/O resources that the consumer group can use. If a consumer group does not use its allocation, then it is redistributed to the other consumer groups in the ratio specified by the plan. You can specify the allocations using the MGMT_P1 parameter, as shown in Example 6–4.

Example 6–4 Sharing Resources Across Applications

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.CREATE_PLAN('DAYTIME_PLAN', 'Resource plan for managing all
 applications between 9 am and 5 pm');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('SALES', 'Sales App');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('FINANCE', 'Finance App');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('MARKETING', 'Marketing App');
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'SALES', 'Allocation
for SALES', MGMT_P1 => 60);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'FINANCE', 'Allocation
for FINANCE', MGMT_P1 => 25);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'MARKETING',
'Allocation for MARKETING', MGMT_P1 => 10);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'OTHER_GROUPS',
'Allocation for default group', MGMT_P1 => 5);
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

Another scenario is to prioritize one workload over another. For example, suppose that you load data into your data warehouse while servicing queries, and you want to always prioritize the queries over the data load. For this scenario, you would create two consumer groups for queries and one consumer group for data load. You would like to share the I/O resources between the two query consumer groups using a 75/25 ratio. In addition, you would like to issue I/Os for data load only if these consumer groups do not use all of their allocation. You can use resource plan levels to specify the allocation priorities as shown in Example 6–5:

Example 6–5 Sharing Resources Across Workloads

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.CREATE_PLAN('DAYTIME_PLAN', 'Resource plan for prioritizing
queries between 9 am and 5 pm');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('REPORT_QUERIES', 'Report Queries');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('AD-HOC_QUERIES', 'Ad-Hoc Queries');
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('DATA_LOAD', 'Data Load');

DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'REPORT_QUERIES',
'Allocation for REPORT_QUERIES', MGMT_P1 => 75);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'AD-HOC_QUERIES',
'Allocation for AD-HOC_QUERIES', MGMT_P1 => 25);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'DATA_LOAD',
'Allocation for DATA_LOAD', MGMT_P2 => 100);
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE('DAYTIME_PLAN', 'OTHER_GROUPS',
'Allocation for default group', MGMT_P3 => 100);
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

As shown by the preceding examples, you must always begin resource plan creations or updates with the PL/SQL procedure `CREATE_PENDING_AREA()` and complete them with the PL/SQL procedure `SUBMIT_PENDING_AREA()`. You must also include a directive for OTHER_GROUPS, which includes all sessions that are not explicitly mapped to a consumer group.

> **See Also:** *Oracle Database Administrator's Guide* for additional information about managing resource plans

Enabling a Database Resource Plan

You can manually enable database resource plans by setting the RESOURCE_MANAGER_PLAN parameter. You can automatically enable resource plans by defining a scheduler window with a resource plan. When the scheduler window opens, the resource plan is enabled. When the scheduler window closes, the resource plan is disabled.

When a resource plan is enabled, the database alerts all cells about this event and provides the resource plan. The database also alerts all cells when a resource plan is disabled. Because only one resource plan can be active for any database, you are required to enable the same resource plan on all instances of a database. If no database resource plan is enabled for a database, then all I/O requests are treated equally.

> **See Also:** *Oracle Database Administrator's Guide* for additional information about managing resource allocations

Managing Fast File Creation

Exadata Cell features fast file creation, allowing accelerated initialization of data files. This feature automatically runs whenever you create a new tablespace, add a datafile to an existing tablespace, or autoextend an existing tablespace. Exadata Cell can initialize files very quickly because it issues many concurrent I/O requests. However, these concurrent I/O requests create a heavy load that can interfere with performance-critical queries.

Using I/O Resource Management, you can control the priority of fast file creations for creating a new tablespace or adding a datafile to an existing tablespace. These operations are run under the `FASTFILECRE` function. By default, the `FASTFILECRE` function is mapped to a hidden consumer group that has lower priority than all consumer group and background I/Os. If you choose to increase the priority, and thereby performance, of file creations, add a mapping rule based on the mapping attribute `DBMS_RESOUCRE_MANAGER.ORACLE_FUNCTION`, and mapping value `FASTFILECRE`. For example, to run fast file creations in the MAINTENANCE_GROUP consumer group, you would run the following commands:

Example 6–6 Managing Fast File Creation

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP('MAINTENANCE_GROUP', 'Maintenance
activity');
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING(DBMS_RESOURCE_MANAGER.ORACLE_
FUNCTION, 'FASTFILECRE', 'MAINTENANCE_GROUP');
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

Because autoextending an existing tablespace is a brief and often time-critical operation, you cannot modify its priority using IORM.

Managing Data Import

Data import, or ETL, is an important part of maintaining a data warehouse. In some cases, ETL is extremely critical to performance because reports or queries cannot be run until the data has been loaded. In these cases, ETL should be prioritized above all other queries. In other cases, ETL is a low-priority background activity that only needs to be prioritized in the rare event that it does not complete by a certain time. You can control the priority of ETL as well as the amount of I/O resources that ETL consumes using IORM.

To manage ETL, you should map the ETL sessions to the ETL_GROUP consumer group and include the ETL_GROUP group in your resource plans. The mapping rules for ETL are typically based on user name or client program name. Data pump is run under the `DATALOAD` function. By default, the `DATALOAD` function is mapped to the `ETL_GROUP` consumer group.

Example 6–7 shows how to map a program to the ETL_GROUP consumer group.

Example 6–7 Mapping a Program to the ETL_GROUP Consumer Group

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING
  (DBMS_RESOURCE_MANAGER.CLIENT_PROGRAM, 'SQLLDR', 'ETL_GROUP');
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

Importing Non-compressed Data as Compressed Data Non-compressed data can be imported as compressed data when using the `TRANSFORM:SEGMENT_ATTRIBUTES=n` option, and the target tablespace has been configured to create new tables as Exadata Hybrid Columnar Compression tables by default.

> **See Also:** *Oracle Database Utilities* for additional information about the `TRANSFORM` option

Managing Oracle Recovery Manager Backups and Copies

Backups are an I/O intensive operation. You can control the rate of Oracle Recovery Manager (RMAN) I/Os by configuring the number of channels. In addition, you can use IORM to control the resource consumption and priority of RMAN I/Os to a much greater degree. For example, you can map RMAN to a low priority consumer group. If Exadata Cell is busy, then the RMAN operations run very slowly and not interfere with the other database operations. However, whenever the Exadata Cell is not fully utilized, then IORM schedules the RMAN I/Os, allowing it to consume the unutilized bandwidth.

RMAN backups run under the `BACKUP` function. RMAN copies run under the `COPY` function. By default, both the `BACKUP` and `COPY` functions are mapped to the BATCH_GROUP consumer group. You can remap these functions to any other consumer group, as shown in Example 6–8:

Example 6–8 Using Consumer Groups to Manage Resources

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING(DBMS_RESOURCE_MANAGER.ORACLE_
FUNCTION, 'BACKUP', 'BATCH_GROUP');
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING(DBMS_RESOURCE_MANAGER.ORACLE_
```

```
FUNCTION, 'COPY', 'MAINTENANCE_GROUP');
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

Administering Interdatabase Resource Management

You can configure an interdatabase or category plan for Exadata Cell using the CellCLI `ALTER IORMPLAN` command. The catPlan parameter specifies the category plan. The dbPlan parameter specifies the interdatabase plan. In order to manage the I/O resources using a category plan, interdatabase plan or database resource plan, set the objective.

The default for the `objective` option is off, and IORM does not manage I/O resources. The `objective` option can be set to `auto`, `low_latency`, `balanced`, or `high_throughput`. Refer to "ALTER IORMPLAN" on page 8-25 for information about the objective options.

Example 6–9 shows how to configure an interdatabase plan for a cell.

Example 6–9 Configuring an Interdatabase Plan

```
CellCLI> ALTER IORMPLAN                                                       -
 catPlan=((name=administrative, level=1, allocation=80),                      -
          (name=interactive,    level=2, allocation=90),                      -
          (name=batch,          level=3, allocation=80),                      -
          (name=maintenance,    level=4, allocation=50),                      -
          (name=other,          level=4, allocation=50)                       -
          ),                                                                  -
 dbPlan=((name='sales1,sales2',level=1, allocation=30, role=primary),  -
         (name=sales1,         level=1, allocation=20, role=standby),  -
         (name=sales2,         level=1, allocation=20, role=standby),  -
         (name=sales_test,     limit=40),              -
         (name=other,          limit=10))
```

The plan name is automatically set to `*cellname*_IORMPLAN`.

For dbPlan directives, you must provide a level and an allocation, or a maximum utilization limit, or both. A directive without a maximum utilization limit or allocation is invalid.

When configuring an interdatabase plan, catPlan and dbPlan are optional parameters. If catPlan is not specified, then I/O resource management between categories is not enabled. Similarly, if dbPlan is not specified, then I/O resource management between databases is not enabled. When specifying catPlan or dbPlan, a directive with `name=other` is required. For category plans, the `other` directive provides an allocation for all active consumer groups whose category is not specified in the category plan. For interdatabase plans, the `other` directive provides an allocation for all databases that are using the Exadata Cell, but are not explicitly specified in the interdatabase plan. If the `other` directive is not specified, then the CellCLI utility returns an error.

The `role` attribute is used for Oracle Data Guard. It allows a different allocation to be specified, based on whether the database is the `primary` or `standby` role. By default, all interdatabase plan allocations apply when the database is in either role. If you want the allocation to apply only when the database is in the `primary` role, then set `role=primary`. Similarly, if you want the allocation to apply only when the database is in the `standby` role, then set `role=standby`.

To reset an interdatabase plan attribute to the default value, set the attribute to an empty string, as shown in Example 6–10. You can reset the entire plan, or separately reset the catPlan or dbPlan.

Example 6–10 Resetting Default Values in an Interdatabase Plan

```
CellCLI> ALTER IORMPLAN dbPlan="", catPlan=""
CellCLI> ALTER IORMPLAN dbPlan=""
CellCLI> ALTER IORMPLAN catPlan=""
```

You can put the `ALTER IORMPLAN` commands in a text file named `alter_iorm`, then run the commands using the text file with the `START alter_iorm` command.

See Also:

- "Activating and Deactivating IORM" on page 6-8 for additional information about deactivating the interdatabase plan on a cell
- "ALTER IORMPLAN" on page 8-25 for additional information about using `ALTER IORMPLAN`
- Example 8–48, "Describing the IORMPLAN Object" on page 8-66 for a list of the `IORMPLAN` attributes
- "START and @" on page 8-7 for additional information about `START`

Listing an I/O Resource Management Plan

You can view the current interdatabase plan for a cell using the CellCLI `LIST IORMPLAN` command on the cell. Example 6–11 shows a detailed list of the interdatabase plan attributes.

Example 6–11 Displaying Interdatabase Plan Details

```
CellCLI> LIST IORMPLAN DETAIL
   name:                   cell01_IORMPLAN
   status:                 active
   catPlan:                name=administrative,level=1,allocation=80
                           name=interactive,level=2,allocation=90
                           name=batch,level=3,allocation=80
                           name=maintenance,level=4,allocation=50
                           name=other,level=4,allocation=50
   dbPlan:                 name='sales1,sales2',level=1,allocation=30,role=primary
                           name=sales1,level=1,allocation=20,role=standby
                           name=sales2,level=1,allocation=20,role=standby
                           name=sales_test,limit=40
                           name=other,limit=10
   objective:              balanced
```

See Also: "LIST IORMPLAN" on page 8-112 for additional information about using `LIST IORMPLAN`

Verifying the Configuration of I/O Resource Management

The following checklist can be used to verify that IORM is configured properly:

- The following criteria must be met when using IORM to manage I/O resources within a database:
  – A resource plan has been enabled.

– The same resource plan has been enabled on all database instances.

Note:

- If Database Resource Manager is enabled using Scheduler Window, then the same plan is always enabled on all database instances.
- If Database Resource Manager is enabled using the RESOURCE_MANAGER_PLAN parameter, then use `sid='*'` to set the parameter for all database instances.

– The resource plan includes MGMT_P[1-8] directives for each consumer group in the resource plan.

The following query can be used to verify the preceding criteria have been met:

```
SELECT DECODE(count(*), 0, 'Intra-Instance IORM Plan Enabled',
'No Intra-Instance IORM Plan Enabled') status FROM gv$instance WHERE
inst_id not in (SELECT inst_id FROM gv$rsrc_plan WHERE cpu_managed = 'ON');
```

- The following command is used to verify that the interdatabase plan has been configured properly when IORM is used to manage I/O resources from multiple databases:

```
CellCLI> LIST IORMPLAN DETAIL
```

If no interdatabase plan has been configured, then use the CellCLI `ALTER IORMPLAN` command to configure a plan. Each active database should have its own directive in dbPlan.

- The following CellCLI command is used to verify that IORM has been enabled on the cells:

```
CellCLI> LIST IORMPLAN
```

If IORM is enabled, then the command returns the current interdatabase plan and the string `active`. If IORM is not enabled, then use the following CellCLI command to enable it:

```
CellCLI> ALTER IORMPLAN ACTIVE
```

- The following query is used to verify that sessions are mapped to the correct consumer group. The command must be run while a workload is running.

```
SELECT r.sid,
       c.consumer_group current_consumer_group
  FROM v$rsrc_session_info r, dba_rsrc_consumer_groups c
  WHERE r.current_consumer_group_id = c.consumer_group_id
union
SELECT sid, 'OTHER_GROUPS' from v$rsrc_session_info
  WHERE current_consumer_group_id = 0;
```

A session may not be in the expected consumer group due to the following configuration errors:

– Missing privilege: In order for a session to switch into a consumer group, its user or role must have permission to switch into that consumer group. The following query shows the permissions for all consumer groups.

```
SELECT grantee, granted_group
```

```
FROM DBA_RSRC_CONSUMER_GROUP_PRIVS
ORDER BY granted_group;
```

The following command is an example of the SQL command that grants permission for any session to switch into a consumer group.

```
EXEC dbms_resource_manager_privs.grant_switch_consumer_group -
  ('public', 'BATCH_GROUP', FALSE);
```

In the preceding command, the consumer group was BATCH_GROUP.

– Inactive consumer group: If a session maps to or is manually switched to a consumer group that is not part of the current resource plan, then the session is switched into the default consumer group, OTHER_GROUPS.

  If sessions are being assigned to consumer groups using mapping rules, then the following query can be used to determine the consumer group that the mapping rules selected, the mapping attribute that was used, and the consumer group that the session started in originally.

```
SELECT r.sid,
       r.mapped_consumer_group,
       r.mapping_attribute,
       c.consumer_group original_consumer_group
FROM v$rsrc_session_info r, dba_rsrc_consumer_groups c
WHERE r.orig_consumer_group_id = c.consumer_group_id;
```

  If the mapped consumer group differs from the original consumer group, then the mapped consumer group was not part of the resource plan.

- While a workload is running, verify that I/O loads are being managed in the correct consumer groups. The following CellCLI command lists the number of small and large I/O requests that were issued for each consumer group across all databases:

```
CellCLI> LIST METRICCURRENT CG_IO_RQ_LG, CG_IO_RQ_SM   ATTRIBUTES name, -
         metricObjectName, metricValue, collectionTime;
```

  Each consumer group that has an active I/O workload should generate small or large I/O requests according to these metrics.

- While the workload is running, query the actual I/O utilization for each category, database and consumer group. The following CellCLI command lists the small and large I/O utilization for each database running on Oracle Exadata Storage Server:

```
CellCLI> LIST METRICCURRENT DB_IO_UTIL_LG, DB_IO_UTIL_SM ATTRIBUTES name, -
metricObjectName, metricValue, collectionTime;
```

  The output shows the percentage of disk resources utilized by small and large requests from the databases.

See Also: "Monitoring IORM Utilization" on page 7-16

Example of Using the Plans

Consider the example of four databases sharing the same Exadata Cell storage. The four databases are:

- An OLTP production database, named PROD
- A test database, named PROD_TEST

- A development database, named PROD_DEV
- A data warehouse database, named DW

An OLTP production database typically issues small I/O requests, and low latency for these requests is the critical requirement. A data warehouse issues large numbers of large I/O requests and is more sensitive to the I/O throughput than the latency of each individual I/O request. Without any I/O resource management, the large number of I/O requests issued by the DW database could overwhelm the storage subsystem and increase the latency of the I/O requests issued by the OLTP database. Additionally, the I/O requests issued by the test and development databases, PROD_TEST and PROD_DEV, could adversely affect the performance of the OLTP and the DW databases.

You can prioritize the I/O requests from these four databases by specifying an interdatabase plan, as follows:

- The OLTP database PROD gets 80 percent of the I/O resources with the highest priority level.
- The DW database gets 20 percent of the remaining I/O resources and 80 percent of PROD's unused allocation.
- The PROD_TEST, PROD_DEV and OTHER databases get any unused I/O in the amount of 50 percent, 40 percent, and 10 percent respectively.

This interdatabase plan can be specified at the cell using the CellCLI utility as shown in Example 6–12.

Example 6–12 Sample Interdatabase Plan

```
CellCLI> ALTER IORMPLAN                                            -
         dbPlan=(                                                  -
                 (name=prod, level=1,allocation=80),               -
                 (name=dw, level=2, allocation=80),                -
                 (name=prod_test,  level=3, allocation=50),        -
                 (name=prod_dev, level=3, allocation=40),          -
                 (name=other, level=3, allocation=10))
```

An example of an interdatabase plan is shown in Table 6–4.

Table 6–4 Sample Interdatabase Plan

Database Name	Level 1 (%)	Level 2 (%)	Level 3 (%)
PROD	80		
DW		80	
PROD_TEST			50
PROD_DEV			40
OTHER			10

If the PROD_TEST and PROD_DEV databases put out an inordinately large I/O load, then the performance of databases PROD and DW are not affected. Also, if database DW issues large amounts of I/O, then the performance of the PROD database are not adversely affected.

7

Monitoring and Tuning Oracle Exadata Storage Server Software

This chapter discusses monitoring and tuning Oracle Exadata Storage Server Software (Exadata Cell). Oracle provides utilities for most monitoring tasks in the Exadata Cell environment.

This chapter contains the following topics:

See Also:

- *Oracle Database 2 Day DBA*
- *Oracle Database Administrator's Guide*
- *Oracle Database Performance Tuning Guide*
- *Oracle Database Real Application Testing User's Guide*
- Oracle Enterprise Manager documentation

Understanding Metrics and Alerts

Metrics and alerts help you monitor Oracle Exadata Storage Server Software. Metrics are associated with objects such as cells and cell disks, and can be cumulative, rate, or instantaneous. By reviewing metrics and setting thresholds for alerts, you can tune and optimize resource management.

Metrics

Metrics are recorded observations of important run-time properties or internal instrumentation values of the storage cell and its components, such as cell disks or grid disks. Metrics are a series of measurements that are computed and retained in memory for an interval of time, and stored on a disk for a more permanent history.

Metric values typically record either the absolute value of a cumulative counter or a rate of change in a cumulative counter over an observed time period. Some metrics are used to record the time of state transitions as well.

Metric values can cause alerts to signal by comparing those values against threshold boundaries. Metrics can be associated with warning and critical thresholds when extreme values in the metric might indicate a problem or other event of interest to an administrator.

See Also: "Monitoring Exadata Cell with Metrics" on page 7-2

Alerts

Alerts represent events of importance occurring within the storage cell, typically indicating that storage cell functionality is either compromised or in danger of failure. An administrator should investigate these alerts, because they might require corrective or preventive action.

There are three types of alerts, informational, warning or critical. Alerts are typically propagated to a monitoring infrastructure, such as Oracle Enterprise Manager, for notification to storage administrators. Examples of possible alerts that can be used are physical disk failure, disk read/write errors, cell temperature exceeding recommended value, Exadata Cell software failure, and excessive I/O latency. Metrics can be used to signal alerts using warning or critical threshold values. When the metric value exceeds the threshold value, an alert can be signaled.

Alerts are either stateful or stateless. Stateful alerts represent observable cell states that can be subsequently retested to detect whether the state has changed, so that a previously observed alert condition is no longer a problem. Stateless alerts represent point-in-time events that do not represent a persistent condition; they simply show that something has occurred.

See Also: "Monitoring Requests and Alerts for Exadata Cell" on page 7-17

Monitoring Exadata Cell with Metrics

You can monitor a cell by viewing the Exadata Cell metrics. Metrics are of the following type:

- Cumulative: Statistics since the metric was created.
- Instantaneous: Values at the time the metric is collected.
- Rate: Computed value for metric by averaging statistics over a period of time.

This section contains the following topics:

- Displaying Metrics
- Monitoring Cell Metrics
- Monitoring Cell Disk Metrics
- Monitoring Flash Cache Metrics
- Monitoring Grid Disk Metrics
- Monitoring Host Interconnection Metrics

Displaying Metrics

You can use the CellCLI `LIST` command to display and monitor metrics for cell objects. You can display metric definitions, current metrics, and metric history using the `LIST` command. The following sections describe how to use the `LIST` command with metrics.

Displaying Metric Definitions

Use the `LIST METRICDEFINITION` command to display the metric definitions for the cell. A metric definition listing shows the configuration of a metric. Example 7–1 shows how to display attributes for the `METRICDEFINITION` object.

Example 7–1 Displaying Metric Definitions

```
CellCLI> LIST METRICDEFINITION CL_CPUT DETAIL

CellCLI> LIST METRICDEFINITION WHERE objectType = 'GRIDDISK'

CellCLI> LIST METRICDEFINITION WHERE name LIKE 'CD_IO_RQ.*' -
         ATTRIBUTES name, metricType, description
```

See Also:

- "DESCRIBE METRICDEFINITION" on page 8-71
- "LIST METRICDEFINITION" on page 8-117

Displaying Current Metrics

Use the `LIST METRICCURRENT` command to display the current metric values for the cell. A current metric listing shows a set of observations on the current value of an individual metric. Example 7–2 shows how to display attributes for the `METRICCURRENT` object.

Example 7–2 Displaying Current Metric Values

```
CellCLI> LIST METRICCURRENT CL_TEMP DETAIL

         name:                   CL_TEMP
         alertState:             normal
         collectionTime:         2009-12-17T15:32:25-08:00
         metricObjectName:       stsd2s3
         metricType:             Instantaneous
         metricValue:            48.0 C
         objectType:             CELL

CellCLI> LIST METRICCURRENT WHERE objectType = 'CELLDISK' AND             -
         metricValue != 0 ATTRIBUTES name, metricObjectName,              -
         metricValue, collectionTime

         CD_IO_BY_R_LG   CD_00_stsd2s3    1.9 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_01_stsd2s3    1.0 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_02_stsd2s3    4.1 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_03_stsd2s3    9.5 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_04_stsd2s3    0.1 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_05_stsd2s3    0.4 MB  2009-12-17T15:46:52-08:00
         ...
```

See Also:

- "DESCRIBE METRICCURRENT" on page 8-70
- "LIST METRICCURRENT" on page 8-116

Displaying Metric History

Use the `LIST METRICHISTORY` command to display the metric history for the cell. A metric history listing shows a collection of past individual observations of all metric values. Example 7–3 shows how to display attributes for the `METRICHISTORY` object.

Example 7–3 Displaying Metric History Values

```
CellCLI> LIST METRICHISTORY CD_IO_RQ_R_LG WHERE alertState='critical' DETAIL

CellCLI> LIST METRICHISTORY WHERE objectType = 'CELLDISK' AND metricValue != 0   -
         AND collectionTime > '2009-08-12T09:10:51-07:00' -ATTRIBUTES name,      -
         metricObjectName, metricValue, collectionTime
```

The retention period for metric history files is specified by the `metricHistoryDays` cell attribute. You can modify this setting with the CellCLI `ALTER CELL` command.

See Also:

- "DESCRIBE METRICHISTORY" on page 8-74
- "LIST METRICHISTORY" on page 8-118
- "ALTER CELL" on page 8-15
- "DESCRIBE CELL" on page 8-54

Monitoring Cell Metrics

Cell metrics provide information about the cell, such as CPU utilization or network interconnections. To display cell metrics, use an object type attribute equal to `CELL` with the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects. Example 7–4 shows how to display cell metrics.

Example 7–4 Displaying Cell Metrics

```
CellCLI> LIST METRICCURRENT CL_CPUT DETAIL

         name:                    CL_CPUT
         alertState:              normal
         collectionTime:          2009-12-17T15:54:25-08:00
         metricObjectName:        stsd2s3
         metricType:              Instantaneous
         metricValue:             4.3 %
         objectType:              CELL
```

Table 7–1 lists the metric name and description for cell metrics. The cell for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects.

Table 7–1 Cell Metrics and Descriptions

Metric	Description
`CL_BBU_CHARGE`	Disk controller battery charge. This metric is not applicable to Exadata Cell on HP Oracle Exadata Storage Server.

Table 7–1 (Cont.) Cell Metrics and Descriptions

Metric	Description
CL_BBU_TEMP	Temperature of disk controller battery. This metric is not applicable to Exadata Cell on HP Oracle Exadata Storage Server.
CL_CPUT	The cell CPU utilization which is the instantaneous percentage of time over the previous minute that the system CPUs were not idle (from /proc/stat).
CL_CPUT_CS	The percentage of CPU time used by CELLSRV.
CL_CPUT_MS	The percentage of CPU time used by MS.
CL_FANS	The instantaneous number of working fans on the cell.
CL_FSUT	The percentage of total space utilized on the file system that is currently in use. This metric shows the space utilization in the various files systems on the cell. Oracle recommends setting the threshold for CL_FSUT to a warning value of 85, and a critical value of 97.
CL_MEMUT_CS	The percentage of physical memory used by CELLSRV.
CL_MEMUT_MS	The percentage of physical memory used by MS.
CL_RUNQ	The instantaneous average number (over the preceding minute) of processes in the Linux run queue marked running or uninterruptible (from /proc/loadavg).
CL_SWAP_IN_BY_SEC	The number of swap pages read in KB per second.
CL_SWAP_OUT_BY_SEC	The number of swap pages written in KB per second.
CL_SWAP_USAGE	The percentage of swap space used.
CL_TEMP	The instantaneous temperature (Celsius) of the server, provided by the Baseboard Management Controller (BMC).
CL_VIRTMEM_CS	The amount of virtual memory used by CELLSRV in MB.
CL_VIRTMEM_MS	The amount of virtual memory used by MS in MB.
CL-MEMUT	The percentage of total physical memory used on the cell.
IORM_MODE	I/O Resource Management objective for the cell.
N_NIC_NW	The instantaneous number of non-working interconnections.
N_NIC_RCV_SEC	The rate which is the total number of I/O packets received by interconnections per second.
N_NIC_TRANS_SEC	The rate which is the total number of I/O packets transmitted by interconnections per second.

Monitoring Cell Disk Metrics

Cell disk metrics provide information about the I/O load for cell disks, such as the number of large blocks read from a cell disk. To display cell disk metrics, use an object type attribute equal to CELLDISK with the METRICCURRENT, METRICDEFINITION, and METRICHISTORY objects.

Table 7–2 lists the metric name and description for cell disk metrics. The cell disk for the metric is specified by the metricObjectName attribute of the METRICCURRENT and METRICHISTORY objects. For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different collectionTime periods. For rate metrics, the time period for the metric value is over the previous minute.

Table 7–2 Cell Disk Metrics and Descriptions

Metric	Description
CD_IO_BY_R_LG	The cumulative number of MB read in large blocks from a cell disk.
CD_IO_BY_R_LG_SEC	The rate which is the number of MB read in large blocks per second from a cell disk.
CD_IO_BY_R_SM	The cumulative number of MB read in small blocks from a cell disk.
CD_IO_BY_R_SM_SEC	The rate which is the number of MB read in small blocks per second from a cell disk.
CD_IO_BY_W_LG	The cumulative number of MB written in large blocks on a cell disk.
CD_IO_BY_W_LG_SEC	The rate which is the number of MB written in large blocks per second on a cell disk.
CD_IO_BY_W_SM	The cumulative number of MB written in small blocks on a cell disk.
CD_IO_BY_W_SM_SEC	The rate which is the number of MB written in small blocks per second on a cell disk.
CD_IO_ERRS	The cumulative number of I/O errors on a cell disk.
CD_IO_ERRS_MIN	The rate of I/O errors on a cell disk per minute.
CD_IO_LOAD	This metric shows the average I/O load from cell disks.
CD_IO_RQ_R_LG	The cumulative number of requests to read large blocks from a cell disk.
CD_IO_RQ_R_LG_SEC	The rate which is the number of requests to read large blocks per second from a cell disk.
CD_IO_RQ_R_SM	The cumulative number of requests to read small blocks from a cell disk.
CD_IO_RQ_R_SM_SEC	The rate which is the number of requests to read small blocks per second from a cell disk.
CD_IO_RQ_W_LG	The cumulative number of requests to write large blocks to a cell disk.
CD_IO_RQ_W_LG_SEC	The rate which is the number of requests to write large blocks per second to a cell disk.
CD_IO_RQ_W_SM	The cumulative number of requests to write small blocks to a cell disk.
CD_IO_RQ_W_SM_SEC	The rate which is the number of requests to write small blocks per second to a cell disk.
CD_IO_ST_RQ	Average service time per request for small I/O requests to a cell disk.
CD_IO_TM_R_LG	The cumulative latency of reading large blocks from a cell disk.
CD_IO_TM_R_LG_RQ	The rate which is the average latency of reading large blocks per request to a cell disk.
CD_IO_TM_R_SM	The cumulative latency of reading small blocks from a cell disk.
CD_IO_TM_R_SM_RQ	The rate which is the average latency of reading small blocks per request from a cell disk.
CD_IO_TM_W_LG	The cumulative latency of writing large blocks to a cell disk.

Table 7–2 (Cont.) Cell Disk Metrics and Descriptions

Metric	Description
CD_IO_TM_W_LG_RQ	The rate which is the average latency of writing large blocks per request to a cell disk.
CD_IO_TM_W_SM	The cumulative latency of writing small blocks to a cell disk.
CD_IO_TM_W_SM_RQ	The rate which is the average latency of writing small blocks per request to a cell disk.

Monitoring Flash Cache Metrics

Flash cache metrics provide information about the utilization of Flash Cache, such as the number of MB read per second from Flash Cache. To display Flash Cache metrics, use an object type attribute equal to FLASHCACHE with the METRICCURRENT, METRICDEFINITION, and METRICHISTORY objects.

Table 7–3 lists the metric name and description for Flash Cache metrics. The Flash Cache for the metric is specified by the metricObjectName attribute of the METRICCURRENT and METRICHISTORY objects.

Table 7–3 Flash Cache Metrics and Descriptions

Metric	Description
FC_BYKEEP_OVERWR	The number of MB pushed out of Flash Cache because of the space limit for keep objects.
FC_BYKEEP_OVERWR_SEC	The number of MB per second pushed out of Flash Cache because of space limit for keep objects.
FC_BYKEEP_USED	The number of MB used for keep objects on Flash Cache.
FC_BY_USED	The number of MB used on Flash Cache.
FC_IO_BYKEEP_R	The number of MB read from Flash Cache for keep objects.
FC_IO_BYKEEP_R_SEC	The rate which is the number of MB read per second from Flash Cache for keep objects
FC_IO_BYKEEP_W	The number of MB written to Flash Cache for keep objects.
FC_IO_BYKEEP_W_SEC	The rate which is the number of MB per second written to Flash Cache for keep objects.
FC_IO_BY_R	The number of MB read from Flash Cache.
FC_IO_BY_R_MISS	The number of MB read from disks because not all requested data was in Flash Cache.
FC_IO_BY_R_MISS_SEC	The rate which is the number of MB read from disks per second because not all requested data was in Flash Cache.
FC_IO_BY_R_SEC	The rate which is the number of MB read per second from Flash Cache.
FC_IO_BY_R_SKIP	The number of MB read from disks for I/O requests that bypass Flash Cache.[1]
FC_IO_BY_R_SKIP_SEC	The rate which is the number of MB read from disks per second for I/O requests that bypass Flash Cache.[1]
FC_IO_BY_W	The number of MB written to Flash Cache.
FC_IO_BY_W_SEC	The rate which is the number of MB per second written to Flash Cache.
FC_IO_ERRS	The number of I/O errors on Flash Cache.

Table 7–3 (Cont.) Flash Cache Metrics and Descriptions

Metric	Description
FC_IO_RQKEEP_R	The number of read I/O requests for keep objects satisfied from Flash Cache.
FC_IO_RQKEEP_R_MISS	The number of read I/O requests for keep objects which did not find all data in Flash Cache.
FC_IO_RQKEEP_R_MISS_SEC	The number of read I/O requests per second for keep objects which did not find all data in Flash Cache.
FC_IO_RQKEEP_R_SEC	The rate which is the number of read I/O requests for keep objects per second satisfied from Flash Cache.
FC_IO_RQKEEP_R_SKIP	The number of read I/O requests for keep objects that bypass Flash Cache.[1]
FC_IO_RQKEEP_R_SKIP_SEC	The rate which is the number of read I/O requests per second for keep objects that bypass Flash Cache.[1]
FC_IO_RQKEEP_W	The number of I/O requests for keep objects which resulted in Flash Cache being populated with data.
FC_IO_RQKEEP_W_SEC	The rate which is the number of I/O requests per second for keep objects which resulted in Flash Cache being populated with data.
FC_IO_RQ_R	The number of read I/O requests satisfied from Flash Cache.
FC_IO_RQ_R_MISS	The number of read I/O requests which did not find all data in Flash Cache.
FC_IO_RQ_R_MISS_SEC	The rate which is the number of read I/O requests per second which did not find all data in Flash Cache.
FC_IO_RQ_R_SEC	The rate which is the number of read I/O requests satisfied per second from Flash Cache.
FC_IO_RQ_R_SKIP	The number of read I/O requests that bypass Flash Cache.[1]
FC_IO_RQ_R_SKIP_SEC	The rate which is the number of read I/O requests per second that bypass Flash Cache.[1]
FC_IO_RQ_W	The number of I/O requests which resulted in Flash Cache being populated with data.
FC_IO_RQ_W_SEC	The rate which is the number of I/O requests per second which resulted in Flash Cache being populated with data.

[1] Read I/O requests that bypass Flash Cache go directly to disks. These requests do not populate Flash Cache after reading the requested data. For example: reads for database objects with CELL_FLASH_CACHE=NONE, or Smart Scan I/Os for database objects with CELL_FLASH_CACHE=DEFAULT.

Monitoring Grid Disk Metrics

Grid disk metrics provide information about the I/O load for grid disks, such as the number of large blocks read from a grid disk. To display grid disk metrics, use an object type attribute equal to GRIDDISK with the METRICCURRENT, METRICDEFINITION, and METRICHISTORY objects.

Table 7–4 lists the metric name and description for grid disk metrics. The grid disk for the metric is specified by the metricObjectName attribute of the METRICCURRENT and METRICHISTORY objects. For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different collectionTime periods. For rate metrics, the time period for the metric value is over the previous minute.

Table 7–4 Grid Disk Metrics and Descriptions

Metric	Description
GD_IO_BY_R_LG	The cumulative number of MB read in large blocks from a grid disk.
GD_IO_BY_R_LG_SEC	The rate which is the number of MB read in large blocks per second from a grid disk.
GD_IO_BY_R_SM	The cumulative number of MB read in small blocks from a grid disk.
GD_IO_BY_R_SM_SEC	The rate which is the number of MB read in small blocks per second from a grid disk.
GD_IO_BY_W_LG	The cumulative number of MB written in large blocks on a grid disk.
GD_IO_BY_W_LG_SEC	The rate which is the number of MB written in large blocks per second on a grid disk.
GD_IO_BY_W_SM	The cumulative number of MB written in small blocks on a grid disk.
GD_IO_BY_W_SM_SEC	The rate which is the number of MB written in small blocks per second on a grid disk.
GD_IO_ERRS	The cumulative number of I/O errors on a grid disk.
GD_IO_ERRS_MIN	The rate of I/O errors on a grid disk per minute.
GD_IO_RQ_R_LG	The cumulative number of requests to read large blocks from a grid disk.
GD_IO_RQ_R_LG_SEC	The rate which is the number of requests to read large blocks per second from a grid disk.
GD_IO_RQ_R_SM	The cumulative number of requests to read small blocks from a grid disk.
GD_IO_RQ_R_SM_SEC	The rate which is the number of requests to read small blocks per second from a grid disk.
GD_IO_RQ_W_LG	The cumulative number of requests to write large blocks to a grid disk.
GD_IO_RQ_W_LG_SEC	The rate which is the number of requests to write large blocks per second to a grid disk.
GD_IO_RQ_W_SM	The cumulative number of requests to write small blocks to a grid disk.
GD_IO_RQ_W_SM_SEC	The rate which is the number of requests to write small blocks per second to a grid disk.

Monitoring Host Interconnection Metrics

Host interconnection metrics provide information about the I/O transmission for hosts that access cell storage. To display host interconnection metrics, use an object type attribute equal to `HOST_INTERCONNECT` with the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects.

Table 7–5 lists the metric name and description for host interconnection metrics. The host interconnection for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects. For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute.

Table 7–5 Host Interconnection Metrics and Descriptions

Metric	Description
N_MB_SENT	The cumulative number of MB transmitted to a particular host.
N_MB_SENT_SEC	The rate which is the number of MB transmitted per second to a particular host.
N_MB_DROP	The cumulative number of MB dropped during transmission to a particular host.
N_MB_DROP_SEC	The rate which is the number of MB dropped per second during transmission to a particular host.
N_MB_RESENT	The cumulative number of MB retransmitted to a particular host.
N_MB_RESENT_SEC	The rate which is the number of MB retransmitted per second to a particular host.
N_MB_RECEIVED	The cumulative number of MB received from a particular host.
N_MB_RECEIVED_SEC	The rate which is the number of MB received per second from a particular host.
N_MB_RDMA_DROP	The cumulative number of MB dropped during remote direct memory access (RDMA) transmission to a particular host.
N_MB_RDMA_DROP_SEC	The rate which is the number of MB dropped per second during RDMA transmission to a particular host.
N_RDMA_RETRY_TM	The cumulative latency of the retry action during RDMA transmission to a particular host.

Monitoring IORM with Metrics

I/O Resource Management (IORM) can be monitored using Exadata Cell metrics. This section contains the following topics:

- Monitoring IORM with Category Metrics
- Monitoring IORM with Database Metrics
- Monitoring IORM with Consumer Group Metrics
- Monitoring IORM Utilization
- Tuning Interdatabase Plans with Metrics

Monitoring IORM with Category Metrics

Category metrics provide information about the size of the I/O load from each category specified in the current IORM category plan. The `objectType` attribute of the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects is equal to `IORM_CATEGORY` for category metrics.

Table 7–6 lists the metric name and description for category metrics. The category for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects. For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute. In the table descriptions, small means less than or equal to 128 KB, and large means more than 128 KB.

Table 7–6 Category Metrics and Descriptions

Metric	Description
CT_FC_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this category to flash cache.
CT_FC_IO_RQ	This metric shows the number of I/O requests issued by an IORM category to flash cache.
CT_FC_IO_RQ_SEC	This metric shows the number of I/O requests issued by an IORM category to flash cache per second.
CT_FD_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this category to flash disks.
CT_FD_IO_LOAD	This metric shows the average I/O load from this category for flash disks.
CT_FD_IO_RQ_LG	This metric shows the number of large I/O requests issued by an IORM category to flash disks.
CT_FD_IO_RQ_LG_SEC	This metric shows the number of large I/O requests issued by an IORM category to flash disks per second.
CT_FD_IO_RQ_SM	This metric shows the number of small I/O requests issued by an IORM category to flash disks.
CT_FD_IO_RQ_SM_SEC	This metric shows the number of small I/O requests issued by an IORM category to flash disks per second.
CT_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this category to hard disks.
CT_IO_LOAD	This metric shows the average I/O load from this category for hard disks.
CT_IO_RQ_LG	The cumulative number of large I/O requests issued by the category. A large value indicates a heavy I/O workload from this category.
CT_IO_RQ_LG_SEC	This metric is derived from `CT_IO_RQ_LG`. It specifies the rate of large I/O requests issued by the category. Its units are number of I/O requests per second. A large value indicates a heavy I/O workload from this category in the past minute.
CT_IO_RQ_SM	The cumulative number of small I/O requests issued by the category. A large value indicates a heavy I/O workload from this category.
CT_IO_RQ_SM_SEC	This metric is derived from `CT_IO_RQ_SM`. It specifies the rate of small I/O requests issued by the category. Its units are number of I/O requests per second. A large value indicates a heavy I/O workload from this category in the past minute.
CT_IO_UTIL_LG	This metric shows the percentage of disk resources utilized by large requests from this category.
CT_IO_UTIL_SM	This metric shows the percentage of disk resources utilized by small requests from this category.
CT_IO_WT_LG	This metric specifies the cumulative number of milliseconds that large I/O requests issued by the category have waited to be scheduled by IORM. A large value indicates that the I/O workload from this category is exceeding the allocation specified for it in the category plan.

Table 7–6 (Cont.) Category Metrics and Descriptions

Metric	Description
CT_IO_WT_LG_RQ	This metric is derived from CT_IO_WT_LG. It specifies the average number of milliseconds that large I/O requests issued by the category have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this category is exceeding the allocation specified for it in the category plan.
CT_IO_WT_SM	This metric specifies the cumulative number of milliseconds that small I/O requests issued by the category have waited to be scheduled by IORM. A large value indicates that the I/O workload from this category is exceeding the allocation specified for it in the category plan.
CT_IO_WT_SM_RQ	This metric is derived from CT_IO_WT_SM. This metric specifies the average number of milliseconds that small I/O requests issued by the category have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this category is exceeding the allocation specified for it in the category plan.

All category cumulative metrics are reset to zero whenever a category, IORM, or any database resource plan is modified. To list the category metric history for an interdatabase plan, use the following CellCLI command:

```
CellCLI> LIST METRICHISTORY WHERE objectType = 'IORM_CATEGORY' AND          -
         metricValue != 0 ATTRIBUTES name, metricObjectName, metricValue,     -
         collectionTime
```

In addition, category metrics are provided for the following internally-generated and automatically-managed categories:

- _ASM_: Oracle ASM-related I/Os
- _ORACLE_BG_CATEGORY_: High-priority I/Os issued by Oracle Database background processes
- _ORACLE_MEDPRIBG_CATEGORY_: Medium-priority I/Os issued by Oracle Database background processes
- _ORACLE_LOWPRIBG_CATEGORY_: Low-priority I/Os issued by Oracle Database background processes

Monitoring IORM with Database Metrics

Database metrics provide information about the size of the I/O load from each database specified in the interdatabase plan. The `objectType` attribute of the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects is equal to `IORM_DATABASE` for database metrics.

Table 7–7 lists the metric name and description for database metrics. The database for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects. For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute. In the table descriptions, small means less than or equal to 128 KB, and large means more than 128 KB.

Table 7–7 Database Metrics and Descriptions

Metric	Description
DB_FC_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this database to flash cache.
DB_FC_IO_RQ	This metric shows the number of I/O requests issued by a database to flash cache.
DB_FC_IO_RQ_SEC	This metric shows the number of I/O requests issued by a database to flash cache per second.
DB_FD_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this database to flash disks.
DB_FD_IO_LOAD	This metric shows the average I/O load from this database for flash disks.
DB_FD_IO_RQ_LG	This metric shows the number of large I/O requests issued by a database to flash disks.
DB_FD_IO_RQ_LG_SEC	This metric shows the number of large I/O requests issued by a database to flash disks per second.
DB_FD_IO_RQ_SM	This metric shows the number of small I/O requests issued by a database to flash disks.
DB_FD_IO_RQ_SM_SEC	This metric shows the number of small I/O requests issued by a database to flash disks per second.
DB_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this database to hard disks.
DB_IO_LOAD	This metric shows the average I/O load from this database for hard disks.
DB_IO_RQ_LG	The cumulative number of large I/O requests issued by the database. A large value indicates a heavy large I/O workload from this database.
DB_IO_RQ_LG_SEC	This metric specifies the rate of large I/O requests issued by a consumer group per second over the past minute. A large value indicates a heavy large I/O workload from this database in the past minute.
DB_IO_RQ_SM	The cumulative number of small I/O requests issued by the database. A large value indicates a heavy small I/O workload from this database.
DB_IO_RQ_SM_SEC	This metric specifies the rate of small I/O requests issued by a consumer group per second over the past minute. A large value indicates a heavy small I/O workload issued by this database in the past minute.
DB_IO_UTIL_LG	This metric shows the percentage of disk resources utilized by large requests from this database.
DB_IO_UTIL_SM	This metric shows the percentage of disk resources utilized by small requests from this database.
DB_IO_WT_LG	This metric specifies the cumulative number of milliseconds that large I/O requests issued by the database have waited to be scheduled by IORM. A large value indicates that the I/O workload from this database is exceeding the allocation specified for it in the interdatabase plan.
DB_IO_WT_LG_RQ	This metric specifies the average number of milliseconds that large I/O requests issued by the database have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this database is exceeding the allocation specified for it in the interdatabase plan.

Table 7–7 (Cont.) Database Metrics and Descriptions

Metric	Description
DB_IO_WT_SM	This metric specifies the cumulative number of milliseconds that small I/O requests issued by the database have waited to be scheduled by IORM. A large value indicates that the I/O workload from this database is exceeding the allocation specified for it in the interdatabase plan.
DB_IO_WT_SM_RQ	This metric specifies the average number of milliseconds that small I/O requests issued by the database have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this database is exceeding the allocation specified for it in the interdatabase plan.

All database cumulative metrics are reset to zero whenever a category, IORM, or any database resource plan is modified.

To list the database metric history for an interdatabase plan, use the following CellCLI command:

```
CellCLI> LIST METRICHISTORY WHERE objectType = 'IORM_DATABASE' AND -
metricValue != 0 ATTRIBUTES name, metricObjectName, metricValue, collectionTime
```

In addition, database metrics are provided for Oracle ASM and all other databases in a metric named `_OTHER_DATABASE_`.

Monitoring IORM with Consumer Group Metrics

Consumer group metrics provide information about the size of the I/O load from each consumer group specified in a database resource plan. Each database in the interdatabase plan has metrics for each of its consumer groups. The `objectType` attribute of the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects is equal to `IORM_CONSUMER_GROUP` for consumer group metrics.

Table 7–8 lists the metric name and description for consumer group metrics. The consumer group and database for the metric are specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects. The name is formed by the database name followed by a period (`.`) and the consumer group name. For example, for a database named PRODUCTIONDB and a consumer group named OLTP, the `metricObjectName` would be PRODUCTIONDB.OLTP.

For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute. In the table descriptions, small means less than or equal to 128 KB, and large means more than 128 KB.

Table 7–8 Consumer Group Metrics and Descriptions

Metric	Description
CG_FC_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this consumer group to flash cache.
CG_FC_IO_RQ	This metric shows the number of I/O requests issued by a consumer group to flash cache.
CG_FC_IO_RQ_SEC	This metric shows the number of I/O requests issued by a consumer group to flash cache per second.
CG_FD_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this consumer group to flash disks.

Table 7–8 (Cont.) Consumer Group Metrics and Descriptions

Metric	Description
CG_FD_IO_LOAD	The metric shows the average I/O load from this consumer group for flash disks.
CG_FD_IO_RQ_LG	This metric shows the number of large I/O requests issued by a consumer group to flash disks.
CG_FD_IO_RQ_LG_SEC	This metric shows the number of large I/O requests issued by a consumer group to flash disks per second.
CG_FD_IO_RQ_SM	This metric shows the number of small I/O requests issued by a consumer group to flash disks.
CG_FD_IO_RQ_SM_SEC	This metric shows the number of small I/O requests issued by a consumer group to flash disks per second.
CG_IO_BY_SEC	This metric shows the number of megabytes of I/O per second for this consumer group to hard disks.
CG_IO_LOAD	The metric shows the average I/O load from this consumer group for hard disks.
CG_IO_RQ_LG	The cumulative number of large I/O requests issued by the consumer group. A large value indicates a heavy I/O workload from this consumer group.
CG_IO_RQ_LG_SEC	This metric is derived from CG_IO_RQ_LG. It specifies the rate of large I/O requests issued by a consumer group per second over the past minute. A large value indicates a heavy I/O workload from this consumer group in the past minute.
CG_IO_RQ_SM	The cumulative number of small I/O requests issued by the consumer group. A large value indicates a heavy I/O workload from this consumer group.
CG_IO_RQ_SM_SEC	This metric is derived from CG_IO_RQ_SM. It specifies the rate of small I/O requests issued by a consumer group per second over the past minute. A large value indicates a heavy I/O workload from this consumer group in the past minute.
CG_IO_UTIL_LG	This metric shows the percentage of disk resources utilized by large requests from this consumer group.
CG_IO_UTIL_SM	This metric shows the percentage of disk resources utilized by small requests from this consumer group.
CG_IO_WT_LG	This metric specifies the cumulative number of milliseconds that large I/O requests issued by the consumer group have waited to be scheduled by IORM. A large value indicates that the I/O workload from this consumer group is exceeding the allocation specified for it in the database resource plan.
CG_IO_WT_LG_RQ	This metric is derived from CG_IO_WT_LG. It specifies the average number of milliseconds that large I/O requests issued by the consumer group have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this consumer group is exceeding the allocation specified for it in the database resource plan.
CG_IO_WT_SM	This metric specifies the cumulative number of milliseconds that small I/O requests issued by the consumer group have waited to be scheduled by IORM. A large value indicates that the I/O workload from this consumer group is exceeding the allocation specified for it in the database resource plan.

Table 7–8 (Cont.) Consumer Group Metrics and Descriptions

Metric	Description
CG_IO_WT_SM_RQ	This metric is derived from CG_IO_WT_SM. It specifies the average number of milliseconds that small I/O requests issued by the consumer group have waited to be scheduled by IORM in the past minute. A large value indicates that the I/O workload from this consumer group is exceeding the allocation specified for it in the database resource plan.

All consumer group cumulative metrics are reset to zero whenever a category, IORM, or any database resource plan is modified.

To list the current metrics for consumer groups, use the following CellCLI command:

```
CellCLI> LIST METRICCURRENT WHERE objectType = 'IORM_CONSUMER_GROUP' AND       -
         metricValue != 0 ATTRIBUTES name, metricObjectName, metricValue,      -
         collectionTime
```

For Oracle ASM and all other databases, metrics are only provided for the BACKGROUND and OTHER consumer groups. The BACKGROUND consumer groups are:

- _ORACLE_BACKGROUND_GROUP_: High-priority I/O requests from Oracle Database background processes
- _ORACLE_MEDPRIBG_GROUP_: Medium-priority I/O requests from Oracle Database background processes
- _ORACLE_LOWPRIBG_GROUP_: Low-priority I/O requests from Oracle Database background processes

See Also:

- "LIST METRICHISTORY" on page 8-118
- "LIST METRICCURRENT" on page 8-116

Monitoring IORM Utilization

When OLTP and DSS workloads share Exadata Cells, IORM determines whether to optimize for low latency or high throughput. To optimize for low latency, large I/O requests should be distributed so the disk is not fully utilized. To optimize for high throughput, each Exadata Cell must handle many concurrent large I/O requests, allowing the cell to be fully utilized while applying optimization algorithms. However, when a cell has many concurrent large I/O requests, I/O latency is high because each I/O is queued behind many other I/Os.

The utilitization metrics for I/O requests from database and consumer groups correspond to the amount of time a database or consumer group utilized a cell. Large I/O requests utilize more of a cell than small I/O requests. The following are the utilization metrics for determining IORM optimization:

- CG_IO_UTIL_LG
- CG_IO_UTIL_SM
- CT_IO_UTIL_LG
- CT_IO_UTIL_SM
- DB_IO_UTIL_LG

- DB_IO_UTIL_SM

By comparing the amount of I/O resources consumed with the I/O resource allocations, the database administrator can determine if IORM should be tuned for latency or throughput. The IORM metric, IORM_MODE, shows the mode for IORM. The metric value ranges between 1 and 3. The following are the definitions for the values:

- 1 means the cell IORM objective was set to low_latency.
- 2 means the cell IORM objective was set to balanced.
- 3 means the cell IORM objective was set to high_throughput.

A value in between 1-2 or 2-3 indicates the IORM objective was not the same throughout the metric period, and the value indicates proximity to a given objective. It is also indicative of a constantly-changing mix of workloads.

See Also:

- "ALTER IORMPLAN" on page 8-25
- "DESCRIBE IORMPLAN" on page 8-66

Tuning Interdatabase Plans with Metrics

The Exadata Cell metrics relevant to I/O Resource Management (IORM) can be used in multiple ways. Metrics can be used to understand the size of the I/O workload as a whole and which portion of it corresponds to each category, database, or consumer group. For example, the metrics might show that a particular database is issuing a higher I/O workload than expected.

In addition, metrics can be used to understand which category, database, or consumer group is not using its resource allocation and which one is exceeding its resource allocation. For example:

- If the wait times (CT_IO_WT_RQ, DB_IO_WT_SM_RQ, DB_IO_WT_LG_RQ, and CG_IO_WT_RQ) are typically small or zero, then the plan allocation is probably sufficient.
- If the wait times are large, then the plan allocation is insufficient.
- If the wait times cause an unacceptable drop in performance, then the plans should be adjusted to give a larger allocation, or more cells or disks should be added to increase the total I/O resources.

Monitoring Requests and Alerts for Exadata Cell

An active request provides a client-centric or application-centric view of client I/O requests that are currently being processed by a cell. Alerts represent events of importance occurring within the cell. Typically, alerts indicate that Exadata Cell functionality is compromised or in danger of failure. You can monitor the active requests and alerts for a cell. In addition, you can receive e-mail notifications for alerts. To receive notifications, use the ALTER CELL command.

This section contains the following topics:

- Displaying Active Requests
- Displaying Alert Definitions
- Receiving Alert Notifications
- Monitoring syslog Messages Remotely

- Displaying Alert History
- Modifying Alert History

See Also:

- "ALTER CELL" on page 8-15 for additional information about receiving e-mail notifications for alerts
- "Alerts" on page 7-2 for an overview of alerts

Displaying Active Requests

Use the `LIST ACTIVEREQUEST` command to display the active requests for the cell. Example 7–5 shows how to display a detailed list of attributes for a specified request I/O type.

Example 7–5 Listing Active Request Attributes

```
CellCLI> LIST ACTIVEREQUEST WHERE IoType = 'predicate pushing' DETAIL
```

To view the `ACTIVEREQUEST` attributes, use the `DESCRIBE ACTIVEREQUEST` command.

See Also:

- "LIST ACTIVEREQUEST" on page 8-97
- "DESCRIBE ACTIVEREQUEST" on page 8-48

Displaying Alert Definitions

Use the `LIST ALERTDEFINITION` command to display the alert definitions for the cell. An alert definition provides a definition for every alert that can be produced on a cell.

Example 7–6 shows how to display a detailed list of attributes for an alert definition.

Example 7–6 Listing Alert Definition Attributes

```
CellCLI> LIST ALERTDEFINITION ADRAlert DETAIL
         name:                  ADRAlert
         alertShortName:        ADR
         alertSource:           "Automatic Diagnostic Repository"
         alertType:             Stateless
         description:           "CELLSRV Incident Error"
         metricName:
```

Example 7–7 shows how to display a list of specified attributes for an alert definition.

Example 7–7 Listing Alert Definition Name and Description Attributes

```
CellCLI> LIST ALERTDEFINITION ATTRIBUTES name, metricName, description
    ADRAlert                                        "Incident Alert"
    HardwareAlert                                   "Hardware Alert"
    StatefulAlert_CD_IO_ERRS_MIN    CD_IO_ERRS_MIN  "Threshold Alert"
    StatefulAlert_CG_IO_RQ_LG       CG_IO_RQ_LG     "Threshold Alert"
    StatefulAlert_CG_IO_RQ_LG_SEC   CG_IO_RQ_LG_SEC "Threshold Alert"
    StatefulAlert_CG_IO_RQ_SM       CG_IO_RQ_SM     "Threshold Alert"
    StatefulAlert_CG_IO_RQ_SM_SEC   CG_IO_RQ_SM_SEC "Threshold Alert"
    StatefulAlert_CG_IO_WT_LG       CG_IO_WT_LG     "Threshold Alert"
```

```
StatefulAlert_CG_IO_WT_LG_RQ    CG_IO_WT_LG_RQ  "Threshold Alert"
StatefulAlert_CG_IO_WT_SM       CG_IO_WT_SM     "Threshold Alert"
StatefulAlert_CG_IO_WT_SM_RQ    CG_IO_WT_SM_RQ  "Threshold Alert"
StatefulAlert_CL_FSUT           CL_FSUT         "Threshold Alert"
StatefulAlert_CL_MEMUT          CL_MEMUT        "Threshold Alert"
StatefulAlert_CT_IO_RQ_LG       CT_IO_RQ_LG     "Threshold Alert"
StatefulAlert_CT_IO_RQ_LG_SEC   CT_IO_RQ_LG_SEC "Threshold Alert"
StatefulAlert_CT_IO_RQ_SM       CT_IO_RQ_SM     "Threshold Alert"
StatefulAlert_CT_IO_RQ_SM_SEC   CT_IO_RQ_SM_SEC "Threshold Alert"
StatefulAlert_CT_IO_WT_LG       CT_IO_WT_LG     "Threshold Alert"
StatefulAlert_CT_IO_WT_LG_RQ    CT_IO_WT_LG_RQ  "Threshold Alert"
StatefulAlert_CT_IO_WT_SM       CT_IO_WT_SM     "Threshold Alert"
StatefulAlert_CT_IO_WT_SM_RQ    CT_IO_WT_SM_RQ  "Threshold Alert"
StatefulAlert_DB_IO_RQ_LG       DB_IO_RQ_LG     "Threshold Alert"
StatefulAlert_DB_IO_RQ_LG_SEC   DB_IO_RQ_LG_SEC "Threshold Alert"
StatefulAlert_DB_IO_RQ_SM       DB_IO_RQ_SM     "Threshold Alert"
StatefulAlert_DB_IO_RQ_SM_SEC   DB_IO_RQ_SM_SEC "Threshold Alert"
StatefulAlert_DB_IO_WT_LG       DB_IO_WT_LG     "Threshold Alert"
StatefulAlert_DB_IO_WT_LG_RQ    DB_IO_WT_LG_RQ  "Threshold Alert"
StatefulAlert_DB_IO_WT_SM       DB_IO_WT_SM     "Threshold Alert"
StatefulAlert_DB_IO_WT_SM_RQ    DB_IO_WT_SM_RQ  "Threshold Alert"
StatefulAlert_GD_IO_ERRS_MIN    GD_IO_ERRS_MIN  "Threshold Alert"
Stateful_HardwareAlert                          "Hardware Stateful Alert"
Stateful_SoftwareAlert                          "Software Stateful Alert"
```

The preceding example displays the alert name, metric name, and description. The metric name identifies the metric on which the alert is based. `ADRAlert`, `HardwareAlert`, `Stateful_HardwareAlert`, and `Stateful_SoftwareAlert` are not based on a metric, and therefore do not have metric names.

See Also:

- "Displaying Metric Definitions" on page 7-3 for additional information about metric names
- "DESCRIBE ALERTDEFINITION" on page 8-51
- "LIST ALERTDEFINITION" on page 8-98

Receiving Alert Notifications

Administrators for Oracle Exadata Storage Server Software can receive alert notifications by e-mail or by Simple Network Management Protocol (SNMP) trap alerts. Use of SNMP alerts allows Exadata Cells to be monitored by a management application, such as Oracle Enterprise Manager. To configure Exadata Cell to send e-mail messages or SNMP trap alerts, use the `ALTER CELL` command.

> **Note:** The SNMP alerts conform to a MIB (management information base) which is included with each installation of Oracle Exadata Storage Server Software. The MIB file on Exadata Cell is available at `/opt/oracle/cell/cellsrv/deploy/config/cell_alert.mib`. The SNMP alerts and MIB conform to SNMP version 1 (SNMPv1).

See Also: "Format of Alert Messages for SNMP Notification" on page B-39

Monitoring syslog Messages Remotely

By default, storage server syslog messages are written to local log files. A separate management server, known as a loghost server, can receive syslog messages from Exadata Cells. To monitor the syslog messages remotely, configure the syslog service on the loghost server to listen for incoming syslog messages by setting `SYSLOGD_OPTIONS -r` in the loghost server `/etc/sysconfig/syslog` file. Configure each Exadata Cell to forward specified syslog messages to the loghost server using the `ALTER CELL` command. The configuration on Exadata Cells is maintained across restarts and updates. The `ALTER CELL VALIDATE SYSLOGCONF` command can be used to test that messages are sent by the cells and received by the loghost server.

> **See Also:** "ALTER CELL" on page 8-15 for additional information about syslog messages

Displaying Alert History

Use the `LIST ALERTHISTORY` command to display the alert history that has occurred on a cell.

Example 7–8 shows how to display a detailed list of attributes for alert history entries where the `severity` attribute is set to `critical` and the `examinedBy` attribute has not been set.

Example 7–8 Listing Alert History Attributes

```
CellCLI> LIST ALERTHISTORY WHERE severity = 'critical' AND examinedBy = '' DETAIL
```

Alert history entries are retained for a maximum of 100 days. If the number of alert history entries exceeds 500, then the alert history entries are only retained for 7 days. When stateful alerts are cleared, meaning that the underlying metric, hardware or software condition has returned to normal, then the stateful alert is retained either 100 or 7 days, depending on the number of alert history entries. Stateful alerts that are not cleared are retained, regardless of their age.

> **See Also:**
>
> - "DESCRIBE ALERTHISTORY" on page 8-52
> - "LIST ALERTHISTORY" on page 8-99
> - "ALTER CELL" on page 8-15
> - "DESCRIBE CELL" on page 8-54

Modifying Alert History

Use the `ALTER ALERTHISTORY` command to update the alert history for the cell.

Example 7–9 shows how to set the `examinedBy` attribute to the user ID of the administrator that examined the alert. The `examinedBy` attribute is the only `ALERTHISTORY` attribute that can be modified.

Example 7–9 Altering Alert History Attributes

```
CellCLI> ALTER ALERTHISTORY 1671443714 examinedBy="jdoe"
```

See Also:

- "ALTER ALERTHISTORY" on page 8-14
- "DESCRIBE ALERTHISTORY" on page 8-52

Monitoring Exadata Cell using Views

Several dynamic V$ views can help monitor the Exadata Cell environment. This section contains the following topics:

Using the V$CELL and GV$CELL Views to Display Exadata Cell Identification

The V$CELL view provides identifying information about cells. Table 7–9 lists the view columns.

Table 7–9 V$CELL View Columns and Descriptions

Column	Description
CELL_HASHVAL	A numeric hash value for the cell. For example: 138889696 **Note:** This value is useful because the P1 argument is often the cell hash name for cell-related wait events in the V$SESSION_WAIT and V$ACTIVE_SESSION_HISTORY views. Using this value, you can use a join with the V$CELL view on cell_hash to determine the corresponding cell name.
CELL_PATH	A character string (maximum 400) that specifies the IP addresses of the cell. These are the IP addresses specified in the cellip.ora file.

The GV$CELL view contains the same columns as the V$CELL view, and includes the INST_ID column. The INST_ID column displays the instance number from which the associated V$ view information was obtained. Querying a GV$ view retrieves the V$ view information from all qualified instances.

Using V$BACKUP_DATAFILE with Exadata Cell

The V$BACKUP_DATAFILE view contains columns relevant to Exadata Cell during Oracle Recovery Manager (RMAN) incremental backups. Table 7–10 lists the view columns.

Table 7–10 V$BACKUP_DATAFILE Columns and Descriptions

Column	Description
BLOCKS	Size of the backup data file in blocks.
BLOCKS_READ	Number of blocks that were scanned while taking this backup. If this is an incremental backup, and block change tracking was used to optimize the backup, then the value of this column is smaller than DATAFILE_BLOCKS. Otherwise, the value of this column is the same as DATAFILE_BLOCKS.
BLOCKS_SKIPPED_IN_CELL	The number of blocks that were read and filtered at the Exadata Cell to optimize the RMAN incremental backup.
DATAFILE_BLOCKS	Size of the data file in blocks at backup time. This value is also the number of blocks taken by the data file restarted from this backup.

The percentage of blocks skipped by Exadata Cell is calculated as follows:

```
(BLOCKS_SKIPPED_IN_CELL / BLOCKS_READ) * 100
```

This number changes significantly based on block change tracking.

If block change tracking for fast incremental backups is used, then most of the filtering is done at the database using the change tracking file, and the blocks are skipped before making an I/O request to the cell. If block change tracking is not used, then all of the blocks are filtered at the cell.

See Also:

- *Oracle Database Reference* for additional information about the V$BACKUP_DATAFILE view
- *Oracle Database Backup and Recovery User's Guide* for additional information about making and updating incremental backups, and block change tracking

Using V$SYSSTAT with Exadata Cell

You can query the V$SYSSTAT view for statistics that can be used to compute Exadata Cell effectiveness. Table 7–11 lists the statistics.

Table 7–11 V$SYSSTAT View Key Statistics

Statistic	Description
cell flash cache read hits	The number of read requests that were a cache hit on exadata flash cache.
cell IO uncompressed bytes	The total size of uncompressed data that is processed on the cell. For scan on hybrid-columnar-compressed tables, this statistic is the size of data after decompression.
cell physical IO interconnect bytes returned by smart scan	The number of bytes that are returned by the cell for Smart Scan only, and does not include bytes for other database I/O.
cell physical IO bytes saved by storage index	The number of bytes saved by storage index.

Table 7–11 (Cont.) V$SYSSTAT View Key Statistics

Statistic	Description
`cell physical IO bytes eligible for predicate offload`	The total number of I/O bytes processed with physical disks when processing was offloaded to the cell.
`cell physical IO bytes pushed back due to excessive CPU`	The number of I/O bytes sent back to the database server for processing due to CPU usage on Exadata Cell.
`cell physical IO bytes saved during optimized file creation`	The number of I/O bytes saved by the database host by offloading the file creation operation to cells. This statistic shows the Exadata Cell benefit due to optimized file creation operations.
`cell physical IO bytes saved during optimized RMAN file restore`	The number of I/O bytes saved by the database host by offloading the RMAN file restore operation to cells. This statistic shows the Exadata Cell benefit due to optimized RMAN file restore operations.
`cell physical IO interconnect bytes`	The number of I/O bytes exchanged over the interconnection between the database host and cells.
`physical read requests optimized`	Total number of read requests satisfied either by using Exadata Smart Flash Cache or storage index.
`physical read total bytes`	Total amount of I/O bytes for reads processed with physical disks. This includes when processing was offloaded to the cell and when processing was not offloaded.
`physical read total bytes optimized`	Total number of bytes read from Exadata Smart Flash Cache or storage index.
`physical write total bytes`	Total amount of I/O bytes for writes processed with physical disks. This includes when processing was offloaded to the cell and when processing was not offloaded.

Example 7–10 shows the `V$SYSSTAT` statistics.

Example 7–10 Querying Cell Statistics in the V$SYSSTAT View

```
SQL> SELECT name, value/1024/1024 MB from v$sysstat a WHERE
        a.name = 'physical read total bytes' OR
        a.name = 'physical write total bytes' OR
        a.name = 'cell physical IO interconnect bytes' OR
        a.name = 'cell physical IO bytes eligible for predicate offload' OR
        a.name = 'cell physical IO bytes saved during optimized file creation' OR
        a.name = 'cell physical IO bytes saved during optimized RMAN file restore'
        OR
        a.name = 'cell IO uncompressed bytes' OR
        a.name = 'cell physical IO interconnect bytes returned by smart scan' OR
        a.name = 'cell physical IO bytes saved by storage index';


NAME                                                                          MB
---------------------------------------------------------------------------------
physical read total bytes                                                5425.11
physical write total bytes                                                 26.24
cell physical IO interconnect bytes                                       550.94
cell physical IO bytes saved during optimized file creation                    0
cell physical IO bytes saved during optimized RMAN file restore                0
cell physical IO bytes eligible for predicate offload                    5372.53
cell physical IO bytes saved by storage index                                  0
```

```
cell physical IO interconnect bytes returned by smart scan              472.12
cell IO uncompressed bytes                                             5372.53

9 rows selected.
```

You can determine the amount of disk I/O performed by doing the following:

```
physical read total bytes  + physical write total bytes - cell physical IO bytes
saved by storage index
```

> **See Also:** *Oracle Database Reference* for additional information about `V$SYSSTAT`

Using V$SEGMENT_STATISTICS with Exadata Cell

The V$SEGMENT_STATISTICS view lists statistics on a per segment basis. The segment-level statistics can be used to detect specific objects, such as tables or indexes, that are performing optimized reads from the cell. The `optimized physical read` segment statistic provides the number of read requests for an objects that were read from Exadata Smart Flash Cache or from the storage index.

Example 7–11 shows a query that returns all objects that have performed more than 1000 optimized reads from a cell. A similar query can be used to determine which objects have performed very few optimized reads.

Example 7–11 Using the V$SEGMENT_STATISTICS View

```
SELECT object_name, value FROM V$segment_statistics WHERE \
statistic_name='optimized physical reads' AND value>1000 ORDER BY value;
```

Using V$SQL with Exadata Cell

The `V$SQL` view lists statistics on shared SQL areas and contains one row for each child of the original SQL text entered. You can query the following columns in the `V$SQL` view for physical IO read disk bytes, physical IO write disk bytes, cell physical IO interconnect bytes, cell physical IO bytes eligible for predicate offload, cell uncompressed bytes, cell interconnect bytes returned for Smart Scan, and the number of physical read requests returned by Exadata Smart Flash Cache or storage index:

- PHYSICAL_READ_BYTES
- PHYSICAL_WRITE_BYTES
- IO_INTERCONNECT_BYTES
- IO_CELL_OFFLOAD_ELIGIBLE_BYTES
- IO_CELL_UNCOMPRESSED_BYTES
- IO_CELL_OFFLOAD_RETURNED_BYTES
- OPTIMIZED_PHY_READ_REQUESTS

The values returned in the view are the number of bytes read by the parallel queries for the particular instance.

Note: The preceding columns are also available in the following views:

- V$SQLAREA
- V$SQLAREA_PLAN_HASH
- V$SQLSTATS
- V$SQLSTATS_PLAN_HASH

Example 7–12 shows a query returning Exadata Cell offload processing performance data. A WHERE predicate was used to select only the V$SQL command of interest, which was the query that scanned the SALES table. The performance data stored in V$SQL for the query shows that of the approximately 5 GB of data in the SALES table, all of it was eligible for offload processing. Because of the effectiveness of Exadata Cell offload processing, only 417 MB of data was delivered over the network to the database host.

Example 7–12 Querying the V$SQL View

```
SELECT sql_text,
       io_cell_offload_eligible_bytes/1024/1024 cell_offload_eligible_mb,
       io_cell_uncompressed_bytes/1024/1024 io_uncompressed_mb,
       io_interconnect_bytes/1024/1024 io_interconnect_mb,
       io_cell_offload_returned_bytes/1024/1024 cell_return_bytes_mb,
       (physical_read_bytes + physical_write_bytes)/1024/1024 io_disk_mb
       FROM v$sql WHERE
       sql_text LIKE '%from sales%';

SQL_TEXT CELL_OFFLOAD_ELIGIBLE_MB IO_UNCOMPRESSED_MB IO_INTERCONNECT_MB CELL_RETURN_BYTES_MB
IO_DISK_MB
---------------------------------------------------------------------------------------------
select count(*) from sales
5283.06   5283.06  520.34  417.65  5385.75
```

See Also:

- "Using V$SYSSTAT with Exadata Cell" on page 7-22
- *Oracle Database Reference* for additional information about V$SQL view

Using V$CELL_STATE to Display Exadata Cell Performance Statistics

The V$CELL_STATE view describes the state of all the cells accessible from the database client. The state includes key performance statistics, such as **Cell Server (CELLSRV)** network layer information, scheduling layer information, and Cell Server configuration information. This view is used by Oracle Support Services. Table 7–12 lists the view columns.

Table 7–12 V$CELL_STATE Columns and Descriptions

Column	Description
`CELL_NAME`	A character string (maximum 400) that specifies the IP addresses of the cell. These are the IP addresses specified in the `cellip.ora` file. For example: `172.16.50.28`
`STATISTICS_TYPE`	The statistics type, such as `thread_stats`.
`OBJECT_NAME`	Key for a specific statistics type, such as the thread ID if `STATISTICS_TYPE` is `thread_stats`.
`STATISTICS_VALUE`	The actual statistic values in an XML document with attribute and value pairs.

Example 7–13 shows how to retrieve the information in the `STATISTICS_VALUE` column in a readable format.

Example 7–13 Displaying Statistics from the V$CELL_STATE View

```
SQL> SPOOL /tmp/cell_state_080619.log
SQL> SET PAGESIZE 10000
SQL> SET LONG 500000
SQL> SELECT  statistics_type, XMLTYPE(statistics_value) FROM v$cell_state;
SQL> SPOOL OFF
```

Using V$CELL_THREAD_HISTORY to Display Exadata Cell Threads

The `V$CELL_THREAD_HISTORY` view contains the samples that Cell Server takes of the threads in the cells visible to the database client. This view is used by Oracle Support Services. Table 7–13 lists the view columns.

Table 7–13 V$CELL_THREAD_HISTORY Columns and Descriptions

Column	Description
`CELL_NAME`	A character string (maximum 400) that specifies the IP addresses of the cell. These are the IP addresses specified in the `cellip.ora` file. For example: `172.16.50.28`
`SNAPSHOT_ID`	The ID of the snapshot (NUMBER).
`SNAPSHOT_TIME`	The date and time of the snapshot.
`THREAD_ID`	The thread ID (NUMBER).
`JOB_TYPE`	The job that the thread was running when the snapshot was taken.
`WAIT_STATE`	A unique state that identifies the location of the wait, if any exists.
`WAIT_OBJECT_NAME`	Object being waited on, if any exists. For example, the mutex name.
`SQL_ID`	The identifier of the SQL command that the client was processing for the job that is running.
`DATABASE_ID`	The ID of the database (NUMBER).

Table 7–13 (Cont.) V$CELL_THREAD_HISTORY Columns and Descriptions

Column	Description
INSTANCE_ID	The ID of the instance (NUMBER).
SESSION_ID	The ID of the session (NUMBER).
SESSION_SERIAL_NUM	The session serial number (NUMBER).

Using V$CELL_REQUEST_TOTALS with Exadata Cell

The V$CELL_REQUEST_TOTALS view contains a historical view of the types and frequencies of the requests being run by a cell. Cell Server samples these requests periodically and stores them for use in this view. This view is used by Oracle Support Services.

Table 7–14 lists the view columns.

Table 7–14 V$CELL_REQUEST_TOTALS Columns and Descriptions

Column	Description
CELL_NAME	A character string (maximum 400) that specifies the IP addresses of the cell. These are the IP addresses specified in the cellip.ora file. For example: 172.16.51.28
SNAPSHOT_ID	The ID of the snapshot (NUMBER).
SNAPSHOT_TIME	The date and time of the snapshot.
STATISTICS_NAME	The name of the statistic.
STATISTICS_VALUE	The value of the statistic.

Understanding Exadata Cell Wait Events

Oracle uses a different set of wait events for disk I/O to Exadata Cell than disk I/O to conventional storage because the wait events that are designed for Exadata Cell directly show the cell and grid disk being accessed. This information is more useful for performance and diagnostics purposes than the database file# and block# information provided by wait events for conventional storage. Information about wait events is displayed in V$ dynamic performance views.

Note: The V$SESSION_ACTIVE_HISTORY view can be used to analyze wait events. This view shows what has happened, when a query was run, and how it ran. It also shows what events the query had to wait on.

This section contains these topics:

- Monitoring Wait Events for Exadata Cell
- Using V$SESSION_WAIT to Monitor Sessions
- Using V$SYSTEM_EVENT to Monitor Wait Events
- Using V$SESSION_EVENT to Monitor Events by Sessions

See Also: *Oracle Database Reference* for additional information about V$SESSION_WAIT, V$SYSTEM_EVENT, and V$SESSION_EVENT

Monitoring Wait Events for Exadata Cell

Table 7–15 lists the wait events useful for monitoring a cell.

Table 7–15 Wait Events Useful for Cell Monitoring

Wait Event	Description
`cell interconnect retransmit during physical read`	This wait event appears during retransmission for an I/O of a single-block or multiblock read. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view is the same cell identified for `cell single block physical read` and `cell multiblock physical read`. The `P2` column contains the subnet number to the cell, and the `P3` column contains the number of bytes processed during the I/O read operation.
`cell list of blocks physical read`	This wait event is equivalent to db file parallel read for a cell. The `P1`, `P2`, and `P3` columns in `V$SESSION_WAIT` view for this event identify the cell hash number, disk hash number, and the number of blocks processed during the I/O read operation.
`cell multiblock physical read`	This wait event is equivalent to `db file scattered read` for a cell. The `P1`, `P2`, and `P3` columns in the `V$SESSION_WAIT` view for this event identify the cell hash number, disk hash number, and the total number of bytes processed during the I/O read operation.
`cell single block physical read`	This wait event is equivalent to `db file sequential read` for a cell. The `P1`, `P2`, and `P3` columns in the `V$SESSION_WAIT` view for this event identify the cell hash number, disk hash number, and the number of bytes processed during the I/O read operation.
`cell smart file creation`	This wait event appears when the database is waiting for the completion of a file creation on a cell. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view for this event should help identify a slow cell compared to the rest of the cells.
`cell smart incremental backup`	This wait event appears when the database is waiting for the completion of an incremental backup on a cell. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view for this event should help identify a slow cell when compared to the rest of the cells.
`cell smart index scan`	This wait event appears when the database is waiting for index or index-organized table (IOT) fast full scans. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view for this event should help identify a slow cell when compared to the rest of the cells.
`cell smart restore from backup`	This wait event appears when the database is waiting for the completion of a file initialization for restore from backup on a cell. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view for this event should help identify a slow cell when compared to the rest of the cells.
`cell smart table scan`	This wait event appears when the database is waiting for table scans to complete on a cell. The cell hash number in the `P1` column in the `V$SESSION_WAIT` view for this event should help identify a slow cell when compared to the rest of the cells.

Table 7–15 (Cont.) Wait Events Useful for Cell Monitoring

Wait Event	Description
cell statistics gather	This wait event appears when a select is done on the V$CELL_STATE, V$CELL_THREAD_HISTORY, or V$CELL_REQUEST_TOTALS tables. During the select, data from the cells and any wait events are shown in this wait event.

If a cell hash number or disk hash number is associated with these wait events, then the value can be joined with the CELL_HASHVAL column of V$CELL and the HASH_VALUE column of V$ASM_DISK to help identify slow cells or disks.

Using V$SESSION_WAIT to Monitor Sessions

The V$SESSION_WAIT view displays the current or last wait for each session. Example 7–14 shows how to query the V$SESSION_WAIT view. The second SELECT query displays the cell path and disk name.

Example 7–14 Using the V$SESSION_WAIT View

```
SELECT w.event, w.p1, w.p2, w.p3 FROM V$SESSION_WAIT w, V$EVENT_NAME e
       WHERE e.name LIKE 'cell%' AND e.wait_class_id = w.wait_class_id;

SELECT w.event, c.cell_path, d.name, w.p3 FROM V$SESSION_WAIT w,
       V$EVENT_NAME e, V$ASM_DISK d, V$CELL c
       WHERE e.name LIKE 'cell%' AND e.wait_class_id = w.wait_class_id
       AND w.p1 = c.cell_hashval AND w.p2 = d.hash_value;
```

Using V$SYSTEM_EVENT to Monitor Wait Events

The V$SYSTEM_EVENT view displays information about the number of total waits for an event. Example 7–15 shows how to query the V$SYSTEM_EVENT view.

Example 7–15 Using the V$SYSTEM_EVENT View

```
SELECT s.event FROM V$SYSTEM_EVENT s, V$EVENT_NAME e WHERE e.name LIKE 'cell%'
       AND e.event_id = s.event_id;
```

Using V$SESSION_EVENT to Monitor Events by Sessions

The V$SESSION_EVENT view displays information about waits for an event by a session. Example 7–16 shows how to query the V$SESSION_EVENT view.

Example 7–16 Using the V$SESSION_EVENT View

```
SELECT s.event FROM V$SESSION_EVENT s, V$EVENT_NAME e WHERE e.name LIKE 'cell%'
       AND e.event_id = s.event_id;
```

Optimizing Performance

This section describes settings for optimizing the performance of Exadata Cell. This section includes the following topics:

- About Exadata Hybrid Columnar Compression
- About SQL Processing Offload
- About SQL Tuning Advisor

- About SQL Monitor
- About Quality of Service Management
- About Fault Isolation
- About Fast Disk Scan Rates
- About Indexes and Scan Rates
- About Automatic Extent Management
- About Minimum Extent Size
- About SQL*Loader in Parallel Mode
- About Fragmentation
- About Disk Group Balance
- About Oracle ASM Disk Repair Timer

See Also: My Oracle Support Note 757552.1 at `http://support.oracle.com/`

About Exadata Hybrid Columnar Compression

With Exadata Hybrid Columnar Compression, the database stores the same column for a group of rows together. Storing the database column values together improves the effectiveness of compression algorithms. The data block does not store the rows in the row-major format. The database compresses data manipulated by any SQL operation, although compression levels are higher for direct path loads.

Exadata Hybrid Columnar Compression provides a very high degree of size reduction. Database operations work transparently against compressed objects, so no application changes are required.

You can specify the following types of Exadata Hybrid Columnar Compression, depending on your requirements:

- Warehouse compression: This type of compression is optimized for query performance, and is intended for data warehouse applications.
- Online archival compression: This type of compression is optimized for maximum compression levels, and is intended for historical data and data that does not change.

Assume that you apply Exadata Hybrid Columnar Compression to a `daily_sales` table. At the end of every day, the table is populated with items and the number sold, with the item ID and date forming a composite primary key. A row subset is shown in Table 7–16.

Table 7–16 Sample Table daily_sales

Item_ID	Date	Num_Sold	Shipped_From	Restock
1000	01-JUN-07	2	WAREHOUSE1	Y
1001	01-JUN-07	0	WAREHOUSE3	N
1002	01-JUN-07	1	WAREHOUSE3	N
1003	01-JUN-07	0	WAREHOUSE2	N
1004	01-JUN-07	2	WAREHOUSE1	N

Table 7–16 (Cont.) Sample Table daily_sales

Item_ID	Date	Num_Sold	Shipped_From	Restock
1005	01-JUN-07	1	WAREHOUSE2	N

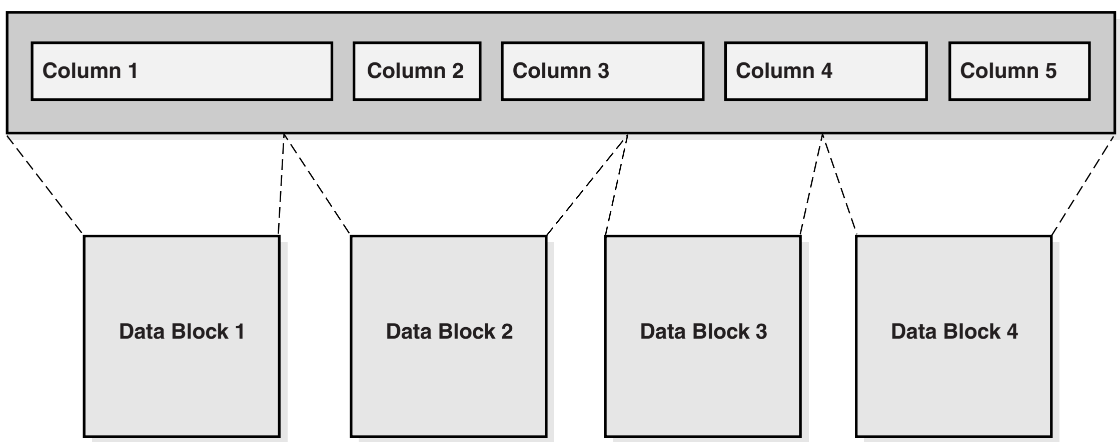
The database stores a set of rows in an internal structure called a compression unit. For example, assume that the rows in Table 7–16 are stored in one unit. Exadata Hybrid Columnar Compression stores each unique value from column 4 with metadata that maps the values to the rows. Conceptually, the compressed value can be represented as follows:

```
WAREHOUSE1WAREHOUSE3WAREHOUSE2
```

The database then compresses the repeated word `WAREHOUSE` in this value by storing it once and replacing each occurrence with a reference. If the reference is smaller than the original word, then the database achieves compression. The compression benefit is particularly evident for the `Date` column, which contains only one unique value.

As shown in Figure 7–1, each compression unit can span multiple data blocks. The values for a particular column may or may not span multiple blocks.

Figure 7–1 Compression Unit



Exadata Hybrid Columnar Compression has implications for row locking. When an update occurs for a row in an uncompressed data block, only the updated row is locked. In contrast, the database must lock all rows in the compression unit if an update is made to any row in the unit. Updates to rows using Exadata Hybrid Columnar Compression cause rowids to change.

Note: When tables use Exadata Hybrid Columnar Compression, Oracle DML locks larger blocks of data (compression units) which may reduce concurrency.

Oracle Database supports four methods of table compression. The methods are summarized in Table 7–17.

Table 7–17 Table Compression Methods

Table Compression Method	Compression Level	CPU Overhead	Applications
Basic compression	High	Minimal	DSS
OLTP compression	High	Minimal	OLTP, DSS
Warehouse compression	Higher[1]	Higher[2]	DSS
Online archival compression	Highest[1]	Highest[2]	Archiving

[1] Compression level depends on compression level specified (LOW or HIGH)
[2] CPU overhead depends on compression level specified (LOW or HIGH)

Warehouse compression and online archival compression achieve the highest compression levels because they use Exadata Hybrid Columnar Compression technology. Exadata Hybrid Columnar Compression technology uses a modified form of columnar storage instead of row-major storage. This enables the database to store similar data together, which improves the effectiveness of compression algorithms. Because Exadata Hybrid Columnar Compression requires high CPU overhead for DML, use it only for data that is updated infrequently.

The higher compression levels of Exadata Hybrid Columnar Compression are achieved only with data that is direct-path inserted. Conventional inserts and updates are supported, but result in a less compressed format, and reduced compression level.

Table 7–18 lists characteristics of each table compression method.

Table 7–18 Table Compression Characteristics

Table Compression Method	CREATE/ALTER TABLE Syntax	Direct-Path Insert	DML
Basic compression	`COMPRESS [BASIC]`[1]	Yes	Yes[2]
OLTP compression	`COMPRESS FOR OLTP`	Yes	Yes
Warehouse compression	`COMPRESS FOR QUERY [LOW|HIGH]`	Yes	Yes[3,4]
Online archival compression	`COMPRESS FOR ARCHIVE [LOW|HIGH]`	Yes	Yes[2,4]

[1] `COMPRESS` and `COMPRESS BASIC` are equivalent
[2] Inserted and updated rows are uncompressed
[3] High CPU overhead
[4] Inserted and updated rows go to a block with a less compressed format and have lower compression level

The `COMPRESS FOR QUERY HIGH` option is the default data warehouse compression mode. It provides good compression and performance. The `COMPRESS FOR QUERY LOW` option should be used in environments where load performance is critical. It loads faster than data compressed with the `COMPRESS FOR QUERY HIGH` option.

The `COMPRESS FOR ARCHIVE LOW` option is the default online archival compression mode. It provides a high compression level and good query performance. It is ideal for infrequently-accessed data. The `COMPRESS FOR ARCHIVE HIGH` option should be used for data that is rarely accessed.

A compression advisor, provided by the `DBMS_COMPRESSION` package, helps you determine the expected compression level for a particular table with a particular compression method.

You specify table compression with the `COMPRESS` clause of the `CREATE TABLE` command. You can enable compression for an existing table by using these clauses in an ALTER TABLE statement. In this case, only data that is inserted or updated is compressed after compression is enabled. Similarly, you can disable table compression for an existing compressed table with the ALTER TABLE...NOCOMPRESS command. In this case, all data that was already compressed remains compressed, and new data is inserted uncompressed.

Determining If a Table Is Compressed

In the *_TABLES data dictionary views, compressed tables have ENABLED in the COMPRESSION column. For partitioned tables, this column is null, and the COMPRESSION column of the *_TAB_PARTITIONS views indicates the partitions that are compressed. In addition, the COMPRESS_FOR column indicates the compression method in use for the table or partition.

```
SQL> SELECT table_name, compression, compress_for FROM user_tables;

TABLE_NAME       COMPRESSION   COMPRESS_FOR
---------------- ------------  -----------------
T1               DISABLED
T2               ENABLED       BASIC
T3               ENABLED       OLTP
T4               ENABLED       QUERY HIGH
T5               ENABLED       ARCHIVE LOW


SQL> SELECT table_name, partition_name, compression, compress_for
  FROM user_tab_partitions;

TABLE_NAME  PARTITION_NAME   COMPRESSION   COMPRESS_FOR
----------- ---------------- -----------   ------------------------------
SALES       Q4_2004          ENABLED       ARCHIVE HIGH
  ...
SALES       Q3_2008          ENABLED       QUERY HIGH
SALES       Q4_2008          ENABLED       QUERY HIGH
SALES       Q1_2009          ENABLED       OLTP
SALES       Q2_2009          ENABLED       OLTP
```

Determining Which Rows are Compressed

When Exadata Hybrid Columnar Compression tables are updated, the rows change to a lower level of compression, such as from COMP_FOR_QUERY_HIGH to COMP_FOR_OLTP or COMP_NOCOMPRESS. To determine the compression level of a row, use the following query:

```
DBMS_COMPRESSION.GET_COMPRESSION_TYPE (
   ownname    IN    VARCHAR2,
   tabname    IN    VARCHAR2,
   row_id     IN    ROWID)
  RETURN NUMBER;
```

By sampling the table rows, you can determine the percentage of rows that are no longer at the higher compression level. You can use ALTER TABLE or MOVE PARTITION to set the rows to a higher compression level. For example, if 10 percent of the rows are no longer at the highest compression level, then you might alter or move the rows to a higher compression level.

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for additional information about GET_COMPRESSION_TYPE

Changing Compression Level

The compression level can be changed for a partition, table, or tablespace. For example, a company uses warehouse compression for its sales data, but sales data older than six months is rarely accessed. If the sales data is stored in a table that is partitioned based on the age of the data, then the compression level for the older data can be changed to online archival compression to free up disk space.

If a table is partitioned, then the `DBMS_REDEFINITION` package can be used to change the compression level of the table. This package performs online redefinition of a table by creating a temporary copy of the table which holds the table data while it is being redefined. The table being redefined remains available for queries and DML statements during the redefinition. The amount of free space for online table redefinition depends on the relative compression level for the existing table, and the new table. Ensure you have enough hard disk space on your system before using the `DBMS_REDEFINITION` package.

If a table is not partitioned, then the `ALTER TABLE...MOVE...COMPRESS FOR...` command can be used to change the compression level. The `ALTER TABLE...MOVE` command does not permit DML statements against the table while the command is running.

To change the compression level for a partition, use the `ALTER TABLE...MODIFY PARTITION` command. To change the compression level for a tablespace, use the `ALTER TABLESPACE` command.

See Also:

- *Oracle Database Administrator's Guide* for additional information about the `ALTER TABLE` command
- *Oracle Database PL/SQL Packages and Types Reference* for additional information about the `DBMS_REDEFINITION` package

Importing and Exporting Exadata Hybrid Columnar Compression Tables

Exadata Hybrid Columnar Compression tables can be imported using the `impdp` command of the Data Pump Import utility. By default, the `impdp` command preserves the table properties and the imported table is Exadata Hybrid Columnar Compression table. The tables can also be exported using the `expdp` command.

On tablespaces not supporting Exadata Hybrid Columnar Compression, the `impdp` command fails with the following error:

```
ORA-6430: hybrid columnar compression is only supported in tablespaces residing on
Exadata storage
```

You can import the Exadata Hybrid Columnar Compression table as an uncompressed table using the `TRANSFORM:SEGMENT_ATTRIBUTES=n` option clause of the `impdp` command.

An uncompressed or OLTP-compressed table can be converted to Exadata Hybrid Columnar Compression format during import. To convert a non-Exadata Hybrid Columnar Compression table to an Exadata Hybrid Columnar Compression table, do the following:

1. Specify default compression for the tablespace using the `ALTER TABLESPACE ... SET DEFAULT COMPRESS` command.
2. Override the `SEGMENT_ATTRIBUTES` option of the imported table during import.

See Also:

- *Oracle Database Utilities* for additional information about the Data Pump Import utility
- *Oracle Database SQL Language Reference* for additional information about the `ALTER TABLESPACE` command

Restoring an Exadata Hybrid Columnar Compression Table

There may be times when an Exadata Hybrid Columnar Compression table needs to be restored from a backup. The table can be restored to a system that supports Exadata Hybrid Columnar Compression, or to a system that does not support Exadata Hybrid Columnar Compression. When restoring a table with Exadata Hybrid Columnar Compression to a system that supports Exadata Hybrid Columnar Compression, restore the file using Oracle Recovery Manager (RMAN) as usual.

When an Exadata Hybrid Columnar Compression table is restored to a system that does not support Exadata Hybrid Columnar Compression, it is necessary to convert the table from Exadata Hybrid Columnar Compression to OLTP compression or a uncompressed format. To restore the table, do the following:

1. Ensure there is sufficient storage in the non-Exadata Cell environment to hold the data in uncompressed or OLTP compression format.
2. Use RMAN to restore the Exadata Hybrid Columnar Compression tablespace.
3. Use the following command to change the data compression from Exadata Hybrid Columnar Compression to `NOCOMPRESS`:

   ```
   ALTER TABLE table_name MOVE NOCOMPRESS
   ```

4. Use the following command to change each partition:

   ```
   ALTER TABLE table_name MOVE PARTITION partition_name NOCOMPRESS
   ```

   Change each partition separately.
5. Use the following command to move the data in parallel:

   ```
   ALTER TABLE table_name MOVE NOCOMPRESS PARALLEL
   ```

6. Use the following command to change the data compression from Exadata Hybrid Columnar Compression to `OLTP COMPRESS`:

   ```
   ALTER TABLE table_name MOVE COMPRESS for OLTP
   ```

See ALso:

- *Oracle Database Backup and Recovery User's Guide* for additional information about RMAN
- *Oracle Database SQL Language Reference* for additional information about the `ALTER TABLE` command

About SQL Processing Offload

To optimize the performance of queries that do table and index scans, the database can offload data search and retrieval processing to the storage cell. This feature is managed by the CELL_OFFLOAD_PROCESSING and CELL_OFFLOAD_PLAN_DISPLAY initialization parameters.

- CELL_OFFLOAD_PROCESSING

- CELL_OFFLOAD_PLAN_DISPLAY

CELL_OFFLOAD_PROCESSING

The CELL_OFFLOAD_PROCESSING initialization parameter enables SQL processing offload to Exadata Cell. When the value of the parameter is set to TRUE, predicate evaluation can be offloaded to cells. The default value of the parameter is TRUE. If the parameter is set to FALSE at the session or system level, then the database performs all the predicate evaluation with cells serving blocks. You can set CELL_OFFLOAD_PROCESSING dynamically with the SQL ALTER SYSTEM or ALTER SESSION commands. For example:

```
SQL> ALTER SESSION SET CELL_OFFLOAD_PROCESSING = TRUE;
```

The CELL_OFFLOAD_PROCESSING parameter can also be set with the OPT_PARAM optimizer hint to enable or disable predicate filtering for a specific SQL command. For example:

```
-- to disable cell_offload_processing for a SQL command
SELECT /*+ OPT_PARAM('cell_offload_processing' 'false') */ COUNT(*) FROM
EMPLOYEES;

-- to enable cell_offload_processing for a SQL command
SELECT /*+ OPT_PARAM('cell_offload_processing' 'true') */ COUNT(*) FROM EMPLOYEES;
```

Note: The CELL_OFFLOAD_PROCESSING initialization parameter cannot be used to compare the performance of Exadata Cell with conventional storage. Even when CELL_OFFLOAD_PROCESSING set to FALSE, Exadata Cell has many advantages over conventional storage. Exadata Cell is highly optimized for fast processing of large queries. It has no bottlenecks at the controller or other levels inside the cell. Exadata Cell uses a modern scale-out architecture and a state-of-the-art InfiniBand network that has much higher throughput than conventional storage networks. Exadata Cell is tightly integrated with the Oracle Database, and has unique capabilities for setup, execution, monitoring, diagnostics, resource management, and corruption prevention.

See Also: *Oracle Database SQL Language Reference* for additional information about using hints

CELL_OFFLOAD_PLAN_DISPLAY

The database parameter CELL_OFFLOAD_PLAN_DISPLAY determines whether the SQL EXPLAIN PLAN command displays the predicates that can be evaluated by Exadata Cell as STORAGE predicates for a given SQL command.

The values for the CELL_OFFLOAD_PLAN_DISPLAY parameter are AUTO, ALWAYS, or NEVER. The default value is AUTO.

- AUTO instructs the SQL EXPLAIN PLAN command to display the predicates that can be evaluated as STORAGE only if a cell is present and if a table is on the cell.
- ALWAYS produces changes to the SQL EXPLAIN PLAN command based on Exadata Cell, whether or not Exadata Cell is present or the table is on the cell. You can use this setting to see what can be offloaded to Exadata Cell before migrating to Exadata Cell.

- NEVER produces no changes to the SQL EXPLAIN PLAN command for Exadata Cell.

You can set the CELL_OFFLOAD_PLAN_DISPLAY parameter dynamically with the SQL ALTER SYSTEM or ALTER SESSION commands. For example:

```
SQL> ALTER SESSION SET CELL_OFFLOAD_PLAN_DISPLAY = ALWAYS;
```

See Also: "Using the SQL EXPLAIN PLAN Command with Exadata Cell" on page 7-41 for additional information about using the EXPLAIN PLAN command

About SQL Tuning Advisor

SQL Tuning Advisor takes one or more SQL statements as input and uses the Automatic Tuning Optimizer to perform SQL tuning on the statements. The output of SQL Tuning Advisor is in the form of advice or recommendations, along with a rationale for each recommendation and its expected benefit. SQL Tuning Advisor provides information about the following:

- Missing and stale statistics
- Better execution plans
- Better access paths and objects
- Better SQL statements

See Also:

- *Oracle Database 2 Day DBA* for additional information about running SQL Tuning Advisor
- *Oracle Database Performance Tuning Guide* for additional information about SQL Tuning Advisor

About SQL Monitor

Oracle Database has the SQL monitor which allows you to monitor SQL statements as they are run. The SQL monitor provides extensive statistics for queries, and works with serial and parallel queries. By default, SQL monitoring is automatically started when a SQL statement runs parallel queries, or when the statement has consumed at least 5 seconds of CPU or I/O time in a single execution. To monitor shorter queries, use the MONITOR hint.

The REPORT_SQL_MONITOR function builds a report in text, HTML or XML with the monitoring information collected about the execution of the SQL statement.

See Also:

- *Oracle Database Performance Tuning Guide* for additional information about the SQL monitor
- *Oracle Database PL/SQL Packages and Types Reference* for additional information about the REPORT_SQL_MONITOR function

About Quality of Service Management

Oracle Exadata Quality of Service Management (QoS Management) allows system administrators to manage application service levels hosted on Oracle Exadata Database Machines. Using a policy-based architecture, QoS Management correlates accurate run-time performance and resource metrics, analyzes the data with its expert system to identify bottlenecks, and produces recommended resource adjustments to meet and maintain performance objectives under dynamic load conditions. When sufficient resources are not available, QoS Management preserves the most business critical objectives at the expense of the less critical ones.

In conjunction with Oracle Clusterware Cluster Health Monitor, QoS Management Memory Guard detects servers that are at risk of failure due to memory over-commitment. It responds to such conditions by automatically preventing new connections, and preserving existing workloads. QoS Management restores connectivity when sufficient memory is available.

See Also: *Oracle Database Quality of Service Management User's Guide*

About Fault Isolation

Oracle Exadata Storage Server Software has the ability to learn from the past events to avoid a potential fatal error. For example, when a faulty SQL statement caused a crash of the server in the past, Oracle Exadata Storage Server Software quarantines the SQL statement so that when the faulty SQL occurs again, Oracle Exadata Storage Server Software does not allow the SQL statement to perform Smart Scan. This reduces the chance of server software crashes, and improves storage availability. The following types of quarantine are available:

- SQL Plan: Created when Oracle Exadata Storage Server Software crashes while performing Smart Scan for a SQL statement. As a result, the SQL Plan for the SQL statement is quarantined, and Smart Scan is disabled for the SQL plan.
- Disk Region: Created when Oracle Exadata Storage Server Software crashes while performing Smart Scan of a disk region. As a result, the 1 MB disk region being scanned is quarantined and Smart Scan is disabled for the disk region.
- Database: Created when Oracle Exadata Storage Server Software detects that a particular database causes instability to a cell. Instability detection is based on the number of SQL Plan Quarantines for a database. Smart Scan is disabled for the database.
- Cell Offload: Created when Oracle Exadata Storage Server Software detects some offload feature has caused instability to a cell. Instability detection is based on the number of Database Quarantines for a cell. Smart Scan is disabled for all databases.

When a quarantine is created, alerts notify administrators of what was quarantined, why the quarantine was created, when and how the quarantine can be dropped manually, and when the quarantine is dropped automatically. All quarantines are automatically removed when a cell is patched or upgraded.

CellCLI commands are used to manually manipulate quarantines. For instance, the administrator can manually create a quarantine, drop a quarantine, change attributes of a quarantine, and list quarantines.

See Also:

- Chapter 8, "Using the CellCLI Utility"
- "ADR Alert Messages" on page B-40
- *Oracle Database Administrator's Guide* for additional information about the Automatic Diagnostic Repository (ADR)

About Fast Disk Scan Rates

To achieve fast disk scan rates, it is important to lay out segments with at least 4 MB of contiguous space. This allows disk scans to read 4 MB of data before performing another seek at a different location on disk. To ensure segments are laid out with 4 MB of contiguous space, set the Oracle ASM allocation unit size to 4 MB, and ensure data file extents are also at least 4 MB. The allocation unit can be set with the disk group attribute `AU_SIZE` when creating the disk group.

The following SQL command creates a disk group with the allocation unit set to 4 MB.

```
SQL> CREATE DISKGROUP data NORMAL REDUNDANCY
     DISK 'o/*/data_CD*'
     ATTRIBUTE 'compatible.rdbms' = '11.2.0.2',
               'compatible.asm' = '11.2.0.2',
               'cell.smart_scan_capable' = 'TRUE',
               'au_size' = '4M';
```

> **See Also:** *Oracle Database SQL Language Reference* for additional information about `CREATE_DISKGROUP`

About Indexes and Scan Rates

In the past, storage required indexes for good performance. Exadata Cell can have superior scan rates without using indexes. Review the application execution plans that use indexes to determine if they would run faster with Oracle Exadata Storage Server Software scans. To determine if a scan would be faster when there is no index, make the index invisible to the optimizer. An invisible index is maintained by DML operations, but it is not used by the optimizer. To make the index invisible, use the following command:

```
ALTER INDEX index_name INVISIBLE;
```

In the preceding command, *index_name* is the name of the index.

About Automatic Extent Management

Table extent size is managed automatically in locally-managed tablespaces using the `AUTOALLOCATE` option. This option automatically increases the size of the extent depending on segment size, available free space in the tablespace, and other factors. The extent size starts at 64 KB and increases to 1 MB for the segment. The extent can increase to 8 MB if the segment size exceeds 64 MB. For a large table, the extent size automatically increases to accommodate the table.

> **See Also:** *Oracle Database SQL Language Reference* for additional information about `storage_clause`

About Minimum Extent Size

Extent size is based on the size of the segment, not the table. A large table that has many partitions can have many small segments. For example, a 100 GB table that is partitioned into 1000 partitions has an average segment size of 100 MB. Therefore, it is good practice to ensure that large tables with many partitions use large extents. To do this, you can provide a hint to the database when the table is created or use an instance level initialization parameter. The following option can be used to hint to the database to allocate large extents for a table:

- The INITIAL storage parameter sets the starting extent size for tables in locally-managed tablespace. The database makes best effort to size every extent in the segment to be at least the extent size defined by the INITIAL storage parameter. For tables that are large, set INITIAL to 8 MB. This storage parameter affects newly-created segments. It is not possible to alter the INITIAL parameter for objects that have already been created.

> **See Also:** *Oracle Database SQL Language Reference* for additional information about `storage_clause`

About SQL*Loader in Parallel Mode

When SQL*Loader is used to load data in parallel, the server uses temporary segments to load data, and then merge the segments into the base segment during transaction commit. For heavily partitioned tables, do not use SQL*Loader to load data. Instead, put the data in a flat file, and use parallel `INSERT` direct load to load from the flat file using the external tables feature of the database.

> **See Also:** *Oracle Database SQL Language Reference* for additional information about `INSERT`

About Fragmentation

A table is considered to be fragmented when the number of small extents in the table is high relative to the size of the table. When the number of small extents is high, then the metadata used to map the segment space is bloated, and the user data is not optimally laid out on disk. This impacts the performance of most of the operations including subsequent load, scan, DDL and DML operations. If the extent size are significantly less than 8 MB, then the segment must be reorganized. Large objects must have at least 8 MB extent sizes. There are several methods to reorganize objects. During reorganization, the storage parameter INITIAL may be set to at least 8 MB for the target object. If the table is not partitioned and very large, then consider partitioning it.

> **See Also:** *Oracle Database Administrator's Guide* for additional information about the following:
>
> - Reclaiming space
> - Redefinition of tablespaces

About Disk Group Balance

Files should be equally balanced across all disks. The following queries and script can be used to check disk group balance:

- To check I/O balance, query the `V$ASM_DISK_IOSTAT` view before and after running a large SQL statement. For example, if a large query has a lot of reads, then the read column and the read_bytes column should be approximately the same for all disks in the disk group.
- To check all mounted disk groups, run the script available in My Oracle Support Note 367445.1. My Oracle Support is available at

  `http://support.oracle.com/`

About Oracle ASM Disk Repair Timer

The Oracle ASM disk repair timer represents the amount of time a disk can remain offline before it is dropped by Oracle ASM. While the disk is offline, Oracle ASM tracks the changed extents so the disk can be resynchronized when it comes back online. The default disk repair time is 3.6 hours. If the default is inadequate, then the attribute value can be changed to the maximum amount of time it might take to detect and repair a temporary disk failure. The following command is an example of changing the disk repair timer value to 8.5 hours for the `DATA` disk group:

```
ALTER DISKGROUP data SET ATTRIBUTE 'disk_repair_time' = '8.5h'
```

The `disk_repair_time` attribute does not change the repair timer for disks currently offline. The repair timer for those offline disks is either the default repair timer or the repair timer specified on the command line when the disks were manually set to offline. To change the repair timer for currently offline disks, use the `OFFLINE` command and specify a repair timer value. The following command is an example of changing the disk repair timer value for disks that are offline:

```
ALTER DISKGROUP data OFFLINE DISK data_CD_06_cell11 DROP AFTER 20h;
```

Note: When the disk repair time value is increased, the vulnerability of a double failure is increased.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for additional information about Oracle ASM disk repair timer

Using the SQL EXPLAIN PLAN Command with Exadata Cell

The SQL `EXPLAIN PLAN` command was enhanced to display the predicates that are evaluated on Exadata Cell. The database parameter CELL_OFFLOAD_PLAN_DISPLAY must be set to `AUTO` or `ALWAYS` for `EXPLAIN PLAN` to display predicates pushed to Exadata Cell.

Predicate evaluation is not offloaded to Exadata Cell in the following cases:

- The CELL_OFFLOAD_PROCESSING parameter is set to `FALSE`.
- The table or partition being scanned is small.
- The optimizer does not use direct path read.
- A scan is performed on a clustered table.
- A scan is performed on an index-organized table.
- A fast full scan is performed on compressed indexes.
- A fast full scan is performed on reverse key indexes.
- The table has row dependencies enabled or the `rowscn` is being fetched.
- The optimizer wants the scan to return rows in `ROWID` order.
- The command is `CREATE INDEX` using `nosort`.
- A `LOB` or `LONG` column is being selected or queried.
- A `SELECT ... VERSIONS` query is done on a table.
- A query that has more than 255 columns referenced and heap table is uncompressed, or Basic or OLTP compressed. However such queries on Exadata Hybrid Columnar Compression-compressed tables are offloaded.
- The tablespace is encrypted, and the CELL_OFFLOAD_DECRYPTION parameter is set to `FALSE`. In order for Exadata Cell to perform decryption, Oracle Database needs to send the decryption key to Exadata Cell. If there are security concerns

about keys being shipped across the network to Exadata Cell, then disable the decryption feature.

- The tablespace is not completely stored on Exadata Cell.
- The predicate evaluation is on a virtual column.

In Example 7–17, the `storage` predicate in the plan output indicates the predicates evaluated by Exadata Cell. The storage predicate is shown in bold.

Example 7–17 Using the EXPLAIN PLAN Command

```
SQL> EXPLAIN PLAN FOR
  SELECT t.prod_id, v.exp1, t2_prod_id, t2_amount_sold
  FROM   sales t, v1 v
  WHERE  t.prod_id = v.prod_id and t.cust_id = v.cust_id
    AND  t.prod_id != 45
    AND  v.amount_sold * v.quantity_sold > 10000;

Explained.

SQL>
SQL> SELECT PLAN_TABLE_OUTPUT FROM TABLE(DBMS_XPLAN.DISPLAY());

PLAN_TABLE_OUTPUT
----------------------------------------------------------------------------------
-----------------
Plan hash value: 2267424675

--------------------------------------------------
| Id  | Operation                   | Name        |
--------------------------------------------------
|   0 | SELECT STATEMENT            |             |
|*  1 |  HASH JOIN                  |             |
|*  2 |   HASH JOIN                 |             |
|*  3 |    TABLE ACCESS STORAGE FULL| SALES       |
|*  4 |    TABLE ACCESS STORAGE FULL| SALES       |
|*  5 |   TABLE ACCESS STORAGE FULL | SALES       |
--------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - access("T"."CUST_ID"="T2"."CUST_ID" AND
              "T1"."PROD_ID"="T2"."PROD_ID" AND "T1"."CUST_ID"="T2"."CUST_ID")
   2 - access("T"."PROD_ID"="T1"."PROD_ID")
   3 - storage("T1"."PROD_ID"<200 AND
              "T1"."AMOUNT_SOLD"*"T1"."QUANTITY_SOLD">10000 AND "T1"."PROD_
ID"<>45)
       filter("T1"."PROD_ID"<200 AND
              "T1"."AMOUNT_SOLD"*"T1"."QUANTITY_SOLD">10000 AND "T1"."PROD_
ID"<>45)
   4 - storage("T"."PROD_ID"<200 AND "T"."PROD_ID"<>45)
       filter("T"."PROD_ID"<200 AND "T"."PROD_ID"<>45)
   5 - storage("T2"."PROD_ID"<200 AND "T2"."PROD_ID"<>45)
       filter("T2"."PROD_ID"<200 AND "T2"."PROD_ID"<>45)
```

Example 7–18 shows the use of `UTLXPLP` to display the `EXPLAIN PLAN` command output when `CELL_OFFLOAD_PLAN_DISPLAY` is enabled.

Example 7–18 Using UTLXPLP to Display the EXPLAIN PLAN Command

```
SQL> ALTER SESSION SET CELL_OFFLOAD_PLAN_DISPLAY = ALWAYS;
Session altered.

SQL> EXPLAIN PLAN FOR SELECT COUNT(*),SUM(creditlim) FROM s_customer1, s_region1
     WHERE s_customer1.zip = s_region1.zip AND region_code = 'NE' ;
Explained.

SQL> @?/rdbms/admin/utlxplp

...
Predicate Information (identified by operation id):
---------------------------------------------------

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
   5 - access("S_CUSTOMER1"."ZIP"="S_REGION1"."ZIP")
  10 - storage("REGION_CODE"='NE')
       filter("REGION_CODE"='NE')
  15 - storage(SYS_OP_BLOOM_FILTER(:BF0000,"S_CUSTOMER1"."ZIP"))
       filter(SYS_OP_BLOOM_FILTER(:BF0000,"S_CUSTOMER1"."ZIP"))
```

See Also:

- *Oracle Database Performance Tuning Guide* for additional information about using and interpreting EXPLAIN PLAN commands
- *Oracle Database SQL Language Reference* for additional information about the SQL EXPLAIN PLAN command
- *Oracle Database Reference* and *Oracle Database Performance Tuning Guide* for additional information about OPTIMZER_MODE

SQL Storage Clause CELL_FLASH_CACHE for Exadata Smart Flash Cache

The CELL_FLASH_CACHE storage clause attribute allows the override of the automatic caching policy when the COMPATIBLE parameter is greater than or equal to 11.2.0.2. In addition, this storage clause attribute can be used to specify how specific database objects are cached in Exadata Smart Flash Cache. The CELL_FLASH_CACHE *storage_clause* option supports the following values:

- NONE: The value ensures that the database object buffers are never cached in Exadata Smart Flash Cache. This allows the user to reserve the flash cache space for useful and frequently-accessed objects.
- DEFAULT: This value specifies that database objects are cached with the default LRU (least recently used) of Exadata Smart Flash Cache. This is the default value for CELL_FLASH_CACHE when the storage clause is omitted from the command.
- KEEP: This value specifies the database object buffers remain cached in the flash cache as long as the flash cache is large enough.

The storage clause can be provided during the CREATE and ALTER commands for a table or other objects. Example 7–19 shows the CREATE TABLE command for the CELL_FLASH_CACHE use with the storage clause.

Example 7–19 Using CREATE TABLE with CELL_FLASH_CACHE

```
CREATE TABLE ptable (c1 number, c2 clob) TABLESPACE TBS_1
          PARTITION BY RANGE(c1) ( PARTITION p1 VALUES LESS THAN (100)
```

```
          TABLESPACE TBS_2 STORAGE (CELL_FLASH_CACHE DEFAULT),
       PARTITION p2 VALUES LESS THAN (200) TABLESPACE TBS_3
          STORAGE (CELL_FLASH_CACHE KEEP));

CREATE TABLE tkbcsrbc (c1 number, l1 clob)
           lob (l1) STORE AS securefile
              (cache nologging STORAGE (CELL_FLASH_CACHE NONE))
           PCTFREE 0 TABLESPACE tbs_93 STORAGE
              (initial 128K next 128K pctincrease 0);
```

For objects where altering the storage clause is allowed, the ALTER command can be used with CELL_FLASH_CACHE, as shown in Example 7–20:

Example 7–20 Using ALTER TABLE with CELL_FLASH_CACHE

```
ALTER TABLE tkbcsrbc STORAGE( CELL_FLASH_CACHE DEFAULT);

ALTER TABLE tkbcsrbc MODIFY LOB (l1) (STORAGE (CELL_FLASH_CACHE KEEP));
```

This CELL_FLASH_CACHE storage clause attribute can be queried using database views based on the object involved, as shown in Example 7–21.

Example 7–21 Using Views to Query CELL_FLASH_CACHE Storage Clause

```
SELECT TABLESPACE_NAME, TABLE_NAME, CELL_FLASH_CACHE FROM user_tables WHERE table_
name='TKBCSRBC';

SELECT CELL_FLASH_CACHE FROM ALL_INDEXES WHERE index_name='TKBCIDX';
```

See Also: *Oracle Database SQL Language Reference* for additional information about storage_clause.

Determining which SQL Operators and Functions are Supported by Oracle Exadata Storage Server Software

One of the primary factors in determining whether Oracle Exadata Storage Server Software performs predicate evaluation is the SQL operator or function referenced in the predicate. Oracle Exadata Storage Server Software supports many SQL operators and functions. However, there are also some operators which are not supported. The dynamic performance view V$SQLFN_METADATA has been enhanced to indicate whether a given SQL operator or function is supported by Oracle Exadata Storage Server Software. The OFFLOADABLE column in the view has been added to provide this information. A YES value indicates that the operator is supported, while a NO value indicates the operator is not supported.

See Also: *Oracle Database Reference* for additional information about the V$SQLFN_METADATA view

8

Using the CellCLI Utility

This chapter describes the Cell Control Command-Line Interface (CellCLI) utility for managing Oracle Exadata Storage Server Software (Exadata Cell). CellCLI provides many of the features that are provided with SQL*Plus, including the use of script files. This chapter contains the following topics:

- Overview of the CellCLI Utility
- About CellCLI Administration Commands
- About CellCLI Object Commands
- About CellCLI Object Types
- CellCLI Object Attributes

> **See Also:** Chapter 9, "Using the dcli Utility" for additional information about managing multiple cells with a centralized management tool

Overview of the CellCLI Utility

The CellCLI utility is the command-line administration tool for Exadata Cell. CellCLI runs on each cell to enable you to manage an individual cell. You use CellCLI to start and stop the cell, to manage cell configuration information, to enable or disable cells, and to manage objects in the cell environment. The command-line utility is already installed when Exadata Cell is shipped. This section contains the following topics:

- Starting CellCLI
- Understanding Command Syntax and Options for CellCLI
- Reserved Words
- CellCLI Command-Line Editing
- CellCLI Input and Output Options
- Comments in CellCLI Scripts
- Line Continuation in CellCLI Commands

Starting CellCLI

You can start CellCLI from the operating system command line on the cell that you want to manage or remotely from a network-attached client using Secure Shell (SSH). The command-line syntax is as follows:

```
cellcli [port_number] [-n] [-m] [-xml] [-v | -vv | -vvv] [-x] [-e command]
```

The *port_number* in the preceding command specifies the HTTP port number of the **Management Server (MS)** for the cell. If the `port_number` argument is omitted, then the CellCLI utility uses the value assigned to the `HTTP_PORT` variable in the `cellinit.ora` file on the cell. If the `HTTP_PORT` variable in the `cellinit.ora` file is not set, then the port number defaults to `8888`.

The following table lists the options that can be used with the CellCLI command:

Option	Description
`-n`	Runs the CellCLI utility in noninteractive mode. This option suppresses the command prompt and disables the command-line editing features.
`-m`	Runs CellCLI monitor (read-only) mode.
`-xml`	Causes output to be displayed in XML format for the Oracle Enterprise Manager plug-in.
`-v`, `-vv`, and `-vvv`	Sets the log level. The `-v` option is for fine, `-vv` is for finer, and `-vvv` is for the finest level.
`-x`	Suppresses the banner.

`-e` *command*	Runs the specified CellCLI command. CellCLI exits after running the command. For example:
	`$ cellcli -e list cell detail`
	`$ cellcli -e "list celldisk attributes name where name   -`
	`  like '.*cell01'"`

CellCLI does not have a login parameter or a connect command. CellCLI uses the cell operating system authentication. The directory from which CellCLI is invoked is the default directory for unqualified file access in CellCLI `SPOOL` and `START` commands.

When starting CellCLI, a banner is displayed showing the cell efficiency ratio. Cell efficiency ratio is the ratio between data accessed by the storage cell and data processed by the database for the cell. Larger values mean more offloading to the cell. The cell efficiency ratio is the same as the value displayed for the offloadEfficiency attribute in the CellCLI `LIST CELL` command.

> **See Also:** "DESCRIBE CELL" on page 8-54 for additional information about the offloadEfficiency attribute

Understanding Command Syntax and Options for CellCLI

CellCLI syntax is as follows:

```
{admin-command | object-command object} [options] ;
```

In the preceding syntax, the following arguments are used:

- *admin-command* is an administrative action.
- *object-command* is an action performed on an object.
- *object* is an object or target on which a command performs an action.
- *options* extend the use of a command combination to include additional parameters for the command.

When using the CellCLI utility, the following rules apply:

- Commands, objects, and options are not case-sensitive except where explicitly stated, such as in string patterns used in filtering strings with the `LIKE` operator.
- Use single quotation marks or double quotation marks around the name of an object that includes spaces or punctuation. The use of quotation marks should match. For example, `"this is incorrect'` is incorrect because the first mark is double quotation marks, and the second is a single quotation mark.
- The current, local cell is the cell to which all CellCLI commands apply.
- A semicolon (;) is optional at the end of a CellCLI command.
- A hyphen (-) is used at the end of a line to continue a long command onto the next line.

See Also:

- "About CellCLI Administration Commands" on page 8-4
- "About CellCLI Object Commands" on page 8-7
- "About CellCLI Object Types" on page 8-7
- Example 8–88 on page 8-114
- Example 8–93 on page 8-120
- "HELP" on page 8-5 for additional information about syntax and usage information for commands
- "DESCRIBE" on page 8-47 for additional information about the attributes for each object type

Reserved Words

The following are CellCLI reserved words:

ABORT	ACTIVE
ACTIVEREQUEST	ALERTDEFINITION
ALERTHISTORY	ALL
ALTER	ASSIGN
BMC	CALIBRATE
CELL	CATPLAN
CELLSRV	CELLDISK
CONFIGUREBMC	CREATE
DBPLAN	DESCRIBE
DETAIL	DROP
EXPORT	FLASHCACHE
FLASHCACHECONTENT	FORCE
GRIDDISK	IMPORT
INACTIVE	IORMPLAN
KEY	LED
LIST	LUN
MAIL	MEMORY

METRICDEFINITION	METRICCURRENT
METRICHISTORY	MS
NULL	OFF
ON	PHYSICALDISK
REALM	RESTART
RS	SHUTDOWN
SNMP	STARTUP
THRESHOLD	VALIDATE

If these keywords are used as values in commands, then they must be enclosed in quotation marks.

CellCLI Command-Line Editing

The CellCLI utility supports command-line history and editing, similar to BSD editline and GNU readline functionality. Most of the command editing features of CellCLI are similar to modern shells, such as `bash` and `tcsh`.

CellCLI Input and Output Options

Exadata Cell command-line utilities read commands from standard input and write output to standard output. You can use the host operating system options for redirecting input and output to compose and process command scripts. For example, you can perform the following redirection:

```
$ cellcli < command-script-in  > results-out
```

In this example, the output from CellCLI commands in the *`command-script-in`* file are written to the *`results-out`* file.

Comments in CellCLI Scripts

You can add single-line comments to CellCLI scripts using several formats. You can begin the comment line with `REMARK`, `REM` or `--` (two hyphens).

For example, the following are valid syntax for comments:

```
REMARK This is a comment
REM This is a comment
-- This is a comment
```

Line Continuation in CellCLI Commands

To continue a long command on to the next line, insert a hyphen (-) at the end of the line. Then, press Enter, and continue typing the command.

For example:

```
CellCLI> LIST CELLDISK WHERE name LIKE 'CD_04.*' -
         ATTRIBUTES name, status, comment
```

About CellCLI Administration Commands

The following CellCLI administration commands are described in this section:

- EXIT

- HELP
- QUIT
- SET
- SPOOL
- START and @

EXIT

Purpose

The `EXIT` command exits from the CellCLI utility, and returns control to the operating system prompt.

Syntax

```
EXIT
```

`EXIT` has the same functionality as the `QUIT` command.

HELP

Purpose

The `HELP` command displays syntax and usage descriptions for all CellCLI commands.

Syntax

```
HELP [help_topic]
```

If no topic argument is provided, `HELP` displays the name of all available topics. If a topic is specified, then detailed help text is displayed for that topic.

Example 8–1 shows examples of the `HELP` command.

Example 8–1 Display Help Text with the HELP Command

```
CellCLI> HELP
CellCLI> HELP ALTER
CellCLI> HELP ALTER CELL
```

QUIT

Purpose

The `QUIT` command exits from the CellCLI utility, and returns control to the operating system prompt.

Syntax

```
QUIT
```

`QUIT` has the same functionality as the `EXIT` command.

SET

Purpose

The SET command sets parameter options in the CellCLI environment.

Syntax

```
SET DATEFORMAT {LOCAL | STANDARD}
SET ECHO [ON | OFF]
```

The SET DATEFORMAT command controls the format of displayed dates. For commands that accept dates, the standard date-time format is recommended. The local format is also accepted. The standard format is recommended for scripts because that format is less sensitive to the time zone, region, and locale changes that might occur when running a script.

The SET ECHO command controls whether to echo commands in a script that is run with @ or START. The ON option displays the commands on screen. The OFF option suppresses the display. The SET ECHO command does not affect the display of commands entered interactively or redirected from the operating system.

Example 8–2 shows an example of the SET command.

Example 8–2 Setting the Date Format with the SET Command

```
SET DATEFORMAT STANDARD
```

SPOOL

Purpose

The SPOOL command writes (spools) the results of commands to the specified file on the cell file system.

Syntax

```
SPO[OL] [file_name [ CRE[ATE] | REP[LACE] | APP[END] ] | OFF]
```

If you issue SPOOL *file_name* with no option, then the output is spooled to that file whether or not the file already exists. The REPLACE option is the default behavior.

The SPOOL options are described in Table 8–1.

Table 8–1 SPOOL Options

Option	Description
APPEND	Adds the results to the end of the file specified.
CREATE	Creates a new file with the name specified, and raises an error if the file exists.
file_name	Names the file to which the results are written. It can be specified with a fully-qualified path name, or with a partially-qualified path name relative to the current directory.
no option	Displays the name of the current spool target file, if any.
OFF	Stops writing (spooling) output to the file.
REPLACE	Replaces the contents of an existing specified file. If the file does not exist, then REPLACE creates the file. This is the default behavior.

START and @

Purpose

The START or @ command runs the CellCLI commands in the specified script file.

Syntax

```
STA[RT] file_name
@file_name
```

The START and @ option is *file_name*. It is the name of the script file that contains the CellCLI commands. If the file name does not include a fully-qualified path, then the CellCLI utility searches for the file relative to the current directory.

The START or @ command is useful when entering long or multiple CellCLI commands. For example, all the commands in Example 8–5 on page 8-19 or Example 8–6 on page 8-19 can be entered in a text file named alter_cell, then run with START alter_cell, assuming that the alter_cell file is in the current directory.

About CellCLI Object Commands

This section describes the CellCLI object commands, object types, and object attributes. The following CellCLI commands operate on Exadata Cell objects:

- ALTER
- ASSIGN KEY
- CALIBRATE
- CREATE
- DESCRIBE
- DROP
- EXPORT CELLDISK
- IMPORT CELLDISK
- LIST

The preceding commands are described in detail later in this chapter.

About CellCLI Object Types

Table 8–2 lists the Exadata Cell object types that can be used with CellCLI object commands:

Table 8–2 Exadata Cell Object Types

Object Type	Description
ACTIVEREQUEST	An active request provides a client-centric or application-centric view of client I/O requests that are currently being processed by a cell. The active request object can be used only with the LIST command.
ALERTDEFINITION	An alert definition provides a definition for every alert that can be produced on the cell. Alerts are defined on metrics and other sources of alerts.

Table 8–2 (Cont.) Exadata Cell Object Types

Object Type	Description
ALERTHISTORY	An alert history provides a list of alerts that have occurred on the cell.
CELL	Cell refers to the current or local cell. A cell is the server to which disks are attached and on which the CellCLI utility runs.
CELLDISK	Each cell disk is associated with a logical unit number (LUN). One physical disk is associated with each cell disk.
FLASHCACHE	The portion of flash storage allocated for use as a cache.
FLASHCACHECONTENT	List of all objects currently cached in the flash cache.
GRIDDISK	A grid disk is a logical partition of a cell disk. It is exposed on the Exadata Cell network to the database hosts, where it becomes an Oracle ASM disk that can be used to create an Oracle ASM disk group.
IBPORT	The InfiniBand ports for Oracle Exadata Storage Server.
IORMPLAN	An interdatabase plan is a set of directives that determines allocation of I/O resources to database clients. There is one plan for the cell.
KEY	A key is a unique hexadecimal string that identifies clients for security purposes.
LUN	Logical unit number (LUN) is the address for an individual physical disk device (a single-disk LUN). LUNs are automatically discovered when the cell is started. They are assigned to the corresponding cell disk when the cell disk is first created or when cell disks are discovered after the system is restarted. LUNs that are not yet assigned to a cell disk have a NULL value for the `cellDisk` attribute.
METRICCURRENT	A current metric describes a set of observations on the current value of an individual metric.
METRICDEFINITION	A metric definition describes the configuration of a metric.
METRICHISTORY	A metric history describes a collection of past individual observations of all metric values.
PHYSICALDISK	A disk is called a physical disk on the cell. Physical disks can be listed, but they are not managed directly by CellCLI. Physical disks are automatically discovered and assigned to the corresponding cell disk when the cell disk is first created or when cell disks are discovered after the system is restarted.
QUARANTINE	A quarantine stops faulty SQL statements from perform a Smart Scan. This reduces software crashes, and improves storage availability.
THRESHOLD	A threshold describes the rules for generating stateful alerts based on a specific metric. The rules include boundary (threshold) values and how long the metric values can violate these boundaries before an alert is generated.

Not all possible command-object combinations are valid. For valid command-object combinations, review the syntax for the specific object command.

See Also:

- "Alerts" on page 7-2
- "About Oracle Exadata Storage Server Software" on page 1-7
- "I/O Resource Management" on page 1-3
- "Monitoring Exadata Cell with Metrics" on page 7-2
- "About CellCLI Object Commands" on page 8-7 for additional information about valid combinations
- "Oracle Exadata Storage Server Software Components" on page 1-6 for additional information about the objects that are Exadata Cell components

CellCLI Object Attributes

Each CellCLI object has a set of attributes that are assigned when the object is created or altered. Attribute filters and lists are used to specify which attributes and objects are displayed in the output of the `LIST` command.

All attributes can be displayed, but only some can be modified directly by the user. To display a list of attributes and determine which ones can be modified, use the `DESCRIBE` command.

See Also:

- "LIST" on page 8-95 for additional information about the `LIST` command
- "DESCRIBE" on page 8-47 for additional information about the `DESCRIBE` command

Restrictions on Values of Common Attributes

The following restrictions apply to the values of attributes common to multiple CellCLI objects.

- The value of the `name` attribute must be less than 256 characters and composed only of the following ASCII characters (alphanumeric and underscore only):
  - Lowercase alphabetic characters (a to z)
  - Uppercase alphabetic characters (A to Z)
  - Numbers (0 to 9)
  - Underscore (_)
- The value of the `comment` attribute must be less than 256 characters.

See the syntax of each CellCLI command for any additional restrictions on attribute values.

Attribute Lists in LIST Command

You can specify which attributes to display for the `LIST` command with the following optional clause:

```
ATTRIBUTES { ALL | attribute1 [, attribute2] ... }
```

`ALL` displays all possible object attributes for the `LIST` object combination.

Example 8–3 shows the `LIST METRICHISTORY` command with the `name` and `metrictype` attributes specified, and the output.

Example 8–3 Listing METRICHISTORY for Specific Attributes

```
LIST METRICHISTORY ATTRIBUTES name, metrictype
         CL_CPUT          Instantaneous
         CL_FANS          Instantaneous
         CL_RUNQ          Instantaneous
         CL_TEMP          Instantaneous
         N_NIC_NW         Instantaneous
         N_NIC_RCV_SEC    Rate
         N_NIC_TRANS_SEC  Rate
         CL_CPUT          Instantaneous
         CL_FANS          Instantaneous
         CL_RUNQ          Instantaneous
         CL_TEMP          Instantaneous
         N_NIC_NW         Instantaneous
         N_NIC_RCV_SEC    Rate
...
```

Attribute Filters in LIST Commands

You can specify which objects to display with the `LIST` command using the following optional clause:

```
WHERE attribute-filter1 [AND attribute-filter2] ...
```

Each *`attribute-filterN`* has the following syntax:

```
attribute [ NOT | !] operator comparison_value
```

In the preceding syntax, the following arguments are used:

- *operator*: The supported operators are listed in Table 8–3. These operators can be combined with `NOT` or `!`.

Table 8–3 Supported Operators in Attribute Filters

Operator	Description
=	Tests for equality between string, status, or numeric attributes. For example: `status NOT = normal`
>	Tests for values greater than the numeric attributes. For example: `size > 139920M`
<	Tests for values less than the numeric attributes. For example: `freeSpace !< 100M`
`LIKE`	Tests for a regular expression match with a string attribute using case-sensitive matching. For example: `LIKE 'GD_IO_RQ.*'`

- *comparison_value*: When used with the supported operators, comparison_value is one of the following value types:
  - Numeric
  - Literal: Value such as `active` or `normal`

– Datetime: Time value supported only for `ALERTHISTORY`
– String: Value delimited by single quotation marks (`''`) or double quotation marks (`" "`)
– `NULL`: Unassigned strings or empty lists

ALTER

Purpose

The ALTER command performs an action on or changes attributes of a single cell object or multiple Exadata Cell objects. The ALTER command can be used to change an attribute or to take an action upon the object.

Syntax

```
ALTER { object_type object_name [, object_name]... operation
      | attribute_name = attribute_value
        [, attribute_name = attribute_value]...
     }
```

Usage Notes

The following arguments can be used with the command:

- *object_type* can be the following:
  - ALERTHISTORY
  - CELL
  - CELLDISK
  - GRIDDISK
  - IBPORT
  - IORMPLAN
  - LUN
  - PHYSICALDISK
  - QUARANTINE
  - THRESHOLD
- *object_name* is the name or identifier of an alert history, cell, cell disk, grid disk, interdatabase plan, LUN, or threshold.
- *operation* is a valid action for the specified *object_type*.

When multiple objects are the target of an ALTER command, there is the possibility of partial success. If an error occurs, then the command is interrupted, and the remaining objects are not changed.

See Also:

- "ALTER ALERTHISTORY" on page 8-14
- "ALTER CELL" on page 8-15
- "ALTER CELLDISK" on page 8-21
- "ALTER GRIDDISK" on page 8-22
- "ALTER IBPORT" on page 8-24
- "ALTER IORMPLAN" on page 8-25
- "ALTER LUN" on page 8-29
- "ALTER PHYSICALDISK" on page 8-30
- "ALTER QUARANTINE" on page 8-31
- "ALTER THRESHOLD" on page 8-32
- "About CellCLI Object Types" on page 8-7 for a description of the object types

ALTER ALERTHISTORY

Purpose

The ALTER ALERTHISTORY command changes the attributes of all or specified alert histories.

Syntax

```
ALTER ALERTHISTORY { ALL | alertid1  [,alertid2 ...]}
       examinedBy=user_name
```

Usage Notes

The following arguments can be used with the command:

- *alertidn*: The identifier of the alerts to be changed.
- *user_name*: The name of the user who acknowledged the alert.

Examples

Example 8–4 shows the ALTER command used with the ALERTHISTORY object to update the examinedBy attribute. The examinedBy attribute is the only ALERTHISTORY attribute that can be modified.

Example 8–4 Altering ALERTHISTORY Attributes

```
CellCLI> ALTER ALERTHISTORY 1671443714 -
                            examinedBy="jdoe"

CellCLI> ALTER ALERTHISTORY ALL examinedBy="jdoe"
```

ALTER CELL

Purpose

The ALTER CELL command changes the attributes of the cell.

Syntax

```
ALTER CELL  {
  | SHUTDOWN SERVICES { RS | MS | CELLSRV | ALL}
  | RESTART SERVICES  { RS | MS | CELLSRV | ALL}
  | RESTART BMC
  | STARTUP SERVICES  { RS | MS | CELLSRV | ALL}
  | LED {ON | OFF}
  | VALIDATE { MAIL | SNMP }
  | VALIDATE SYSLOGCONF selector.node
  | CONFIGUREBMC
  | attribute_name = attribute_value
        [, attribute_name = attribute_value]...
  }
```

Usage Notes

The following table lists the arguments and options for the ALTER CELL command:

Argument	Options	Description
SHUTDOWN SERVICES	RS (**Restart Server (RS)**) MS (**Management Server (MS)**) CELLSRV (**Cell Server (CELLSRV)**) ALL	All services are shutdown, or the specified service is shutdown. When the SHUTDOWN SERVICES ALL or SHUTDOWN SERVICES CELLSRV option is specified, then the following occurs: ■ All grid disks are made inactive and stop accepting user I/O. ■ Oracle ASM takes disks offline. Communication between the cell, the database instance, and Oracle ASM instances is disrupted.
RESTART SERVICES	RS MS CELLSRV ALL	Restart Server must be running before restarting individual services. If Restart Server is not running, then the only possible commands are STARTUP SERVICES {RS | ALL} or RESTART SERVICES {RS | ALL}.
RESTART BMC	none	Restarts the Baseboard Management Controller (BMC).
STARTUP SERVICES	RS MS CELLSRV ALL	All services are started or the specified service in the command is started. RS must be running before you can start other individual services. If RS is not running, then the only possible commands are STARTUP SERVICES {RS | ALL} or RESTART SERVICES {RS | ALL}.

Argument	Options	Description
`LED`	`ON` `OFF`	`LED ON` and `LED OFF` operations turn on and off the chassis LED. You can manually set the LED to light to indicate that a cell requires maintenance. The LED is also set to light automatically if a component fails.

`VALIDATE`	`ASR`
	`MAIL`
	`SNMP`
	`CONFIGURATION`
		The `VALIDATE ASR` operation sends a test from the storage server host to the Auto Service Request Manager.
		The `VALIDATE MAIL` operation sends a test message using the e-mail attributes configured for the cell.
		The `VALIDATE SNMP` operation sends a test message using the SNMP attributes configured for the cell.
		The `VALIDATE CONFIGURATION` operation validates the configuration. When the validation is complete and correct, the system responds with `Cell` *`cell_disk`* `successfully altered`. If there is a problem, then the system responds with an error message.

`VALIDATE SYSLOGCONF`	*`facility.priority`*	The `VALIDATE SYSLOGCONF` *`facility.priority`* sends a test message for the specified facility and priority.
`CONFIGUREBMC`	none	Configures the BMC for hardware alerts to the local cell so that Management Server can pick up the alerts.
attributes		The attributes that can be changed using the `ALTER CELL` command are shown as `modifiable` in Example 8–42, "Describing the CELL Object" on page 8-56.

The following are additional usage notes for the `ALTER CELL` command:

- You might need to restart, shut down, or start up a cell for the following reasons:
  - Software upgrades
  - Service outages that include any condition under which a cell is not responding to service requests
- To set up the cell to send notifications about alerts, you can configure the following cell attributes:
  - `smtpServer`
  - `smtpPort`
  - `smtpUser`
  - `smtpPwd`
  - `smtpUseSSL`
  - `smtpFrom`
  - `smtpFromAddr`
  - `smtpToAddr`

– `snmpSubscriber`
– `notificationMethod`
– `notificationPolicy`

The `smtpToAddr` attribute can be set to a list of comma-delimited e-mail addresses that are the recipients of the alert notification. The list must be enclosed in quotation marks.

The `snmpSubscriber` attribute can be set to a list of SNMP targets to which the SNMP alert notification is sent. These targets are specified as follows:

```
snmpSubscriber=
  ((host=host [,port=port] [,community=community][,type=ASR])
    [,(host=host[,port=port][,community=community][,type=ASR])…)
```

Notes:

- The `snmpSubscriber` attribute is used to configure Auto Service Request.
- The SNMP alerts conform to a MIB (management information base) which is included with each installation of Oracle Exadata Storage Server Software. The MIB file on Exadata Cell is available at `/opt/oracle/cell/cellsrv/deploy/config/cell_alert.mib`. The SNMP alerts and MIB conform to SNMP version 1 (SNMPv1). However, the Auto Service Request alerts conform to SNMP version 2c (SNMPv2c).

The default value for `port` is `162`. The default value for `community` is `public`.

The `notificationMethod` attribute value can be a combination of `mail` and `snmp`, such as `notificationMethod='mail,snmp'`. The default value is unassigned, it is neither `mail` or `snmp`.

The `smtpUseSSL` attribute enables Secure Socket Layer (SSL) encryption on the e-mail notifications when the attribute is set to `true`.

The `notificationPolicy` attribute value can be `none` or a combination of `critical`, `warning`, and `clear`, such as `notificationPolicy='warning,critical'`

– The `critical` value refers to hardware-generated alerts or alerts generated by Automatic Diagnostic Repository (ADR) or BMC. The `critical` value also refers to a metric alert when the value exceeds the critical threshold specified in the metric definition.
– The `warning` value refers to a metric alert when the value exceeds the warning threshold specified in the metric definition.
– The `clear` value refers to a metric alert when the value is below the threshold boundary after having previously exceeded a warning or critical threshold.

- The `ALTER CELL snmpSubscriber...type=asr` command sets the Auto Service Request destination for Exadata Cell, and its ILOM. Removing all `snmpSubscriber` entries with `type=asr` from the SNMP subscriber list disables the trap mechanism for Exadata Cell and its ILOM.
- For each subscriber, the host must be specified as either a domain name or an IP address. Enclose the host name or IP address in quotation marks if it contains

non-alphanumeric characters. Port and community values are optional. The default port value is `162`. The default community value is `public`. The type value is optional. The default value for type is `NULL`. Currently, `type=asr` is the only supported non-NULL value.

- After startup of the Management Server (MS), the `snmpSubscriber` list entries with `type=asr` are added to the ILOM for the cell. This ensures that when an ILOM is replaced, the entries are automatically set for the new ILOM.
- The `snmpSubscriber` with `type=asr` should only be configured to point to ASR Manager.
- To validate that e-mail messages are successfully sent for cell alerts or events, use the `ALTER` command with the `VALIDATE MAIL` option. The validation process sends a test e-mail message to the configured recipient. If that test e-mail message is not received, then an e-mail configuration setting is not valid.
- The `emailFormat` attribute can be `html` or `text`. By default, e-mail notifications are sent in HTML format. Change the value to `text` to receive plain text e-mail notifications.
- If you change the cell name, then you must choose a unique cell name.
- The `bbuLearnCycleTime` attribute is used to set the start time for the battery learn cycle. After the learn cycle has completed, the attribute reverts to its default quarterly cycle.
- If `interconnectN` is defined, then specifying only `ipaddressN` is valid. However, if `interconnectN` is not specified in the cell, then `interconnectN` should be specified at the same time as the `ipaddressN="`*`address`*`"` command.
- The `ALTER CELL interconnectN=""` or `ALTER CELL ipaddressN=""` commands delete information about both `interconnectN` and `ipaddressN` from the system. You cannot have one have a value and the other be empty.
- If an `ipaddressN` attribute is specified and not empty, then the network startup scripts are modified. The user must issue a `service network restart` command as the `root` user for the changes to the IP address to take effect.
- If the IP address to an InfiniBand interface is changed, then the command `service openibd restart` must be run as the `root` user before the `service network restart` command.
- After changing an IP address, you must restart all services using the `ALTER CELL RESTART SERVICES ALL` command.
- The `syslogconf` attribute extends syslog rules for a cell. The attribute can be used to designate that syslog messages be forwarded to a specified management server. On the management server, the forwarded messages are directed to a file, console, or management application, depending on the syslog configuration on the management server. The following shows the syntax for the attribute:

  ```
  syslogconf = ('selector @node' [, 'selector @node']... )
  ```

  In the preceding syntax, *selector* is the message type, and *node* is the specified server. Both variables follow `syslog.conf` standard syntax rules.
- The `ALTER CELL VALIDATE syslogconf` *`selector`* command sends a test log message. The test message is directed as specified by rules in the `/etc/syslog.conf` file. If the syslogconf assignment extends the syslog rules, then a test message is forwarded to the specified management servers.

Examples

Example 8–5 shows the ALTER command with the CELL object.

Example 8–5 Altering Cell Name, IP Addresses, IP Block, and Cell Number Attributes

```
CellCLI> ALTER CELL name='cell05', ipaddress1='10.142.0.221/24',  -
                                   interconnect1 = 'bondib0'

CellCLI> ALTER CELL name=cell02, ipBlock='10.10.128/15', cellNumber=2
```

In the preceding command, the IP address is in compressed format.

Example 8–6 shows how to set up e-mail notifications for the cell.

Example 8–6 Configuring E-Mail Notifications for a Cell

```
CellCLI> ALTER CELL smtpServer='my_mail.example.com',            -
                    smtpFromAddr='john.doe@example.com',         -
                    smtpFrom='John Doe',                         -
                    smtpToAddr='jane.smith@example.com',         -
                    snmpSubscriber=((host=host1),(host=host2)),  -
                    notificationPolicy='critical,warning,clear', -
                    notificationMethod='mail,snmp'
```

Example 8–7 shows how to validate the e-mail setup on a cell.

Example 8–7 Validating E-mail on a Cell

```
CellCLI> ALTER CELL VALIDATE MAIL
```

Example 8–8 shows how to validate the Auto Service Request e-mail setup on a cell.

Example 8–8 Validating Auto Service Request E-mail on a Cell

```
CellCLI> ALTER CELL VALIDATE SNMP type=asr
```

Example 8–9 shows how to change the format of e-mail messages.

Example 8–9 Changing the Format of E-mail Messages

```
CellCLI> ALTER CELL emailFormat='text'
CellCLI> ALTER CELL emailFormat='html'
```

Example 8–10 shows how to validate the SNMP setup on a cell.

Example 8–10 Validating SNMP on a Cell

```
CellCLI> ALTER CELL VALIDATE SNMP
```

Example 8–11 shows how to validate the configuration on a cell.

Example 8–11 Validating Configuration on a Cell

```
CellCLI> ALTER CELL VALIDATE CONFIGURATION

Cell CD_01_cell01 successfully altered
```

Example 8–12 shows a sample error message when configuration on a cell is incorrect.

Example 8–12 Checking an Incorrect Configuration on a Cell

```
CellCLI> ALTER CELL VALIDATE CONFIGURATION

[ILOMVersion]
Requires:
3.0.6.10.a r49240
Found:
3.0.6.10.a r49385
The hardware and firmware are not supported. See details above.
The suggested action is: Correct the configuration problems. Then run cellcli
command:ALTER CELL VALIDATE CONFIGURATION.Verify that the new configuration is
correct.
```

Example 8–13 shows how to start up and shut down cell services.

Example 8–13 Starting Up and Shutting Down Cell Services

```
CellCLI> ALTER CELL STARTUP SERVICES ALL
CellCLI> ALTER CELL SHUTDOWN SERVICES MS
CellCLI> ALTER CELL RESTART SERVICES CELLSRV
```

Example 8–14 shows how to set the LED on the cell.

Example 8–14 Setting the Cell LED Off and On

```
CellCLI> ALTER CELL LED OFF
CellCLI> ALTER CELL LED ON
```

Example 8–15 shows how to add a rule using the syslogconf attribute.

Example 8–15 Using the syslogconf Attribute

```
CellCLI> ALTER CELL syslogconf=('*.err;authpriv.none @loghost', -
         '*.emerg @loghost')
```

Example 8–16 shows how to add and validate a rule with test message.

Example 8–16 Adding and Validating a Rule

```
CellCLI> ALTER CELL syslogconf=('kern.crit @loghost')
CellCLI> ALTER CELL VALIDATE syslogconf   'kern.crit'
```

Example 8–17 shows how to remove the syslog.conf rule:

Example 8–17 Removing All syslog.conf Rules

```
CellCLI> ALTER CELL syslogconf=''
```

See Also:

- "CREATE CELL" on page 8-37 for additional information about setting cell attributes
- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions on the cell `name` and `comment` attribute values

ALTER CELLDISK

Purpose

The ALTER CELLDISK command changes the attributes of all cell disks or the specified cell disks.

Syntax

```
ALTER CELLDISK { ALL [FLASHDISK | HARDDISK] | cdisk_name [, cdisk_name]... }
   { attribute_name = attribute_value
        [, attribute_name = attribute_value]...
   }
```

Usage Notes

The attributes that can be changed with the ALTER command are shown as modifiable in Example 8–43, "Describing the CELLDISK Object" on page 8-58.

The FLASHDISK option limits the ALTER CELLDISK command to cell disks that are flash disks.

The HARDDISK option limits the ALTER CELLDISK command to cell disks that are hard disks.

Examples

Example 8–18 shows how to change cell disk attributes.

Example 8–18 Altering Cell Disk Attributes

```
CellCLI> ALTER CELLDISK cdiska name = CD_01_cell01, -
               comment = 'cdiska is now CD_01_cell01'

CellCLI> ALTER CELLDISK ALL -
               comment = 'This cell disk is on cell cell01'
```

See Also:

- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions on the name or comment attribute value
- "CREATE CELLDISK" on page 8-39 for additional information about setting cell disk attributes

ALTER GRIDDISK

Purpose

The ALTER GRIDDISK command changes the attributes of all grid disks or specified grid disks.

> **Caution:** Before changing the name of a grid disk that belongs to an Oracle ASM disk group, ensure that the Oracle ASM disk group is offline.

Syntax

```
ALTER GRIDDISK { ALL  [FLASHDISK | HARDDISK] |gdisk_name [, gdisk_name]... }
   { ACTIVE
   |INACTIVE
   | attribute_name = attribute_value
        [, attribute_name = attribute_value]...
   [NOWAIT]
   [FORCE]
   }
```

Usage Notes

The attributes that can be changed with the ALTER GRIDDISK command are shown as modifiable in Example 8–46, "Describing the GRIDDISK Object" on page 8-62.

- The FLASHDISK option limits the ALTER GRIDDISK command to grid disks that are flash disks.
- The HARDDISK option limits the ALTER GRIDDISK command to grid disks that are hard disks.
- ACTIVE mode notifies Cell Server to accept I/O as normal for the specified grid disks. The grid disks are visible to the database clients.
- INACTIVE mode makes the grid disks visible to the cell administrator, but not visible to the database clients. This mode allows management operations on the grid disks. You can do upgrading and testing on the grid disks before making the grid disks visible to database users. This functionality is similar to starting up a database in RESTRICTED mode.

  > **Note:** When a grid disk that is currently in use by a database client is made INACTIVE, Oracle ASM takes the corresponding Oracle ASM disk offline when I/Os to the disk fail. To make the disk usable again, make the grid disk ACTIVE in the cell, and then bring the corresponding Oracle ASM disk back online in Oracle ASM.

- The length of a grid disk name is limited to 30 characters.
- The size attribute can be specified to expand or reduce space allocated to a grid disk. The corresponding Oracle ASM disk must be resized separately.

  The size attribute is specified as a number in bytes, unless the suffix M (megabytes) or G (gigabytes) is included with the number value. Grid disk space is allocated in 16 MB units, referred to as allocation units. The actual size allocated is the size of the largest multiple of allocation units less than or equal to the specified size. The minimum value is 16 MB. Values less than 16 MB are rounded up to 16 MB.

- A grid disk should not be renamed while it is part of an active Oracle ASM disk group because the Oracle ASM instance may not recognize the renamed disk. If you try to rename an active grid disk, then the operation fails. If you choose to rename an active grid disk, then use the `FORCE` option in the command.
- When an interleaved grid disk is resized, the contents of the grid disk are moved to achieve the interleaved space allocation across the cell disk. The resizing operation can take a few minutes. You can choose to have the data movement proceed as a background process by using the `NOWAIT` option. Use the `LIST GRIDDISK` command to check the status.

Examples

Example 8–19 shows the `ALTER` command with the `GRIDDISK` object.

Example 8–19 Altering Grid Disk Attributes

```
CellCLI> ALTER GRIDDISK data1_CD_01_cell01, data2_CD_01_cell01
         comment = "This grid disk is on cell01"

CellCLI> ALTER GRIDDISK ALL INACTIVE
```

Example 8–20 shows the `NOWAIT` option being used to have data movement done as a background process.

Example 8–20 Using the NOWAIT Option

```
CellCLI> ALTER GRIDDISK gd0 SIZE=40M NOWAIT
GridDisk gd0 alter in progress
```

See Also:

- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions on the `name` or `comment` attribute values
- "LIST CELLDISK" on page 8-102 for additional information about listing cell disk attributes
- "CREATE GRIDDISK" on page 8-42 for additional information about grid disk attributes
- Chapter 4, "Configuring Security for Oracle Exadata Storage Server Software" for additional information about Exadata Cell security for grid disks

ALTER IBPORT

Purpose

The ALTER IBPORT command performs an action on all InfiniBand ports, or specified InfiniBand ports.

Syntax

```
ALTER IBPORT {ALL | ibport_name [, ibport_name] ...} RESET COUNTERS
```

Usage Notes

The RESET COUNTERS option resets all counters on the InfiniBand port.

Examples

Example 8–21 shows the ALTER command with the IBPORT object.

Example 8–21 Altering IBPORT Attributes

```
CellCLI> ALTER IBPORT ALL RESET COUNTERS

         InfiniBand Port HCA-1:1 successfully altered.
         InfiniBand Port HCA-1:2 successfully altered.

CellCLI> ALTER IBPORT "HCA-1:1" RESET COUNTERS

         InfiniBand Port HCA-1:1 successfully altered.
```

ALTER IORMPLAN

Purpose

The ALTER IORMPLAN command updates the interdatabase plan for the cell. The directives determine the allocation of I/O resources to database clients.

Syntax

```
ALTER IORMPLAN { ACTIVE  | INACTIVE
               [objective='iorm_objective']
               [dbPlan=(directive1[, directive2]...),]
               [catPlan=( directive1[, directive2]...)]}
```

Usage Notes

The attributes that can be changed with the ALTER command are shown as modifiable in Example 8–48, "Describing the IORMPLAN Object" on page 8-66.

- You can activate the interdatabase plan with the ACTIVE option, and deactivate the interdatabase plan with the INACTIVE option. INACTIVE is the default value.
- If you change an inactive interdatabase plan, then the plan does not automatically become active. You must explicitly set the plan to active status using the ALTER IORMPLAN ACTIVE command.
- The directive*N* for a database plan (dbPlan) or category plan (catPlan) is of the following format:

  ```
  ( name=db_or_category_name, [level=number] [, allocation=number]
  [, role={ primary | standby }] [, limit=number]) [, flashcache={on|off}]
  ```

  – The name value cannot start with an underscore (_).

  – At least one attribute for level, allocation or limit must be specified with the name value. For example, name and limit or name and level.

  – A comma-delimited list in single quotation marks (' ') can be used for the name value in a dbPlan directive. A comma-delimited list implies that all of the names are the values of the DB_UNIQUE_NAME initialization parameter for the databases that are part of the same Oracle Data Guard cluster. The role value must be set to primary when using a comma-delimited list. Because only one database can be in the primary role at any time, it never shares the allocation value with another database.

  > **Note:** You can set up a dbPlan directive for a database that was not configured for Exadata Cell or specified as a client when the availableTo attribute was set. In these cases, the allocation of the database is unused until the appropriate configuration is complete.

  – The role attribute indicates that the directive is applied only when the databases are in that database role. If the role attribute is not specified, then the directive is applied regardless of the database role. The role attribute cannot be used on catPlan directives or dbPlan other directives. The role attribute must be the last attribute specified in the directive.

  – The limit attribute is used with the dbPlan directive. This attribute specifies the I/O utilization limit for databases. It is a value greater than zero, and less than or equal to 100.

Note: The `MAX_UTILIZATION_LIMIT` attribute specifies the I/O utilization limit for consumer groups. Refer to *Oracle Database Administrator's Guide* and *Oracle Database PL/SQL Packages and Types Reference* for additional information about this attribute.

  - The `flashcache` attribute is used to prevent a database from using the flash cache. This ensures that flash cache space is reserved for mission-critical databases.
  - Up to 28 directives are allowed for each `catPlan`. These directives include the mandatory `other` category directive.
  - Up to 32 directives are allowed for each `dbPlan`. These directives include the mandatory `other` category directive.
  - The level value is 1 to 8.
  - For each level and each role, the total `allocation` value cannot exceed 100.
  - The `other` value is a distinguished database and category name. It refers to all clients that are not specifically named in other directives. At least one directive with `name=other` is required for both `dbPlan` and `catPlan`, or the CellCLI utility returns an error.

- If the `dbPlan` directive is not set, then the default is that each database has an equal percentage or share of the resources.
- If the `catPlan` directive is not set, then the default is that no category plan is used. The resources are divided among databases.
- The `objective` option specifies the optimization mode for IORM. The objective option can be as follows:
  - `low_latency`: Use this setting for critical OLTP workloads that require extremely good disk latency. This setting provides the lowest possible latency by significantly limiting disk utilization.
  - `balanced`: Use this setting for critical OLTP and DSS workloads. This setting balances low disk latency and high throughput. This setting limits disk utilization of large I/Os to a lesser extent than `low_latency` to achieve a balance between good latency and good throughput.
  - `high_throughput`: Use this setting for critical DSS workloads that require high throughput.
  - `auto`: Use this setting to have IORM determine the optimization objective. IORM continuously and dynamically determines the optimization objective, based on the workloads observed, and resource plans enabled.
  - `off`: Use this setting to stop IORM from managing I/O resources. This the default value for the `objective` option.
- To reset an interdatabase plan attribute to the default value, set the attribute to an empty string. Single quotation marks or double quotation marks can be used to set an empty string. The marks must match, for example `" "` is correct, but `" '` is incorrect.
- Consider running `ALTER IORMPLAN` commands as scripts because of the length of the commands.

Examples

Example 8–22 shows the ALTER command with the IORMPLAN object.

Example 8–22 Altering IORMPLAN Attributes

```
CellCLI> ALTER IORMPLAN                                               -
        catPlan=(                                                     -
                 (name=administrative, level=1, allocation=80),-
                 (name=interactive, level=2, allocation=90),   -
                 (name=batch, level=3, allocation=80),         -
                 (name=maintenance, level=4, allocation=50),   -
                 (name=other, level=4, allocation=50)          -
                ),                                             -
        dbPlan=((name=sales_prod, level=1, allocation=80),     -
                (name=finance_prod, level=1, allocation=20),   -
                (name=sales_dev, level=2, allocation=100),     -
                (name=sales_test, level=3, allocation=50),     -
                (name=other, level=3, allocation=50))

CellCLI> ALTER IORMPLAN                                                          -
         dbPlan=((name='sales1,sales2', level=1, allocation=30, role=primary), -
                 (name=sales1, level=1, allocation=20, role=standby),          -
                 (name=sales2, level=1, allocation=20, role=standby),          -
                 (name=other, level=3, allocation = 50))

CellCLI> ALTER IORMPLAN                                                          -
         dbplan=((name=db1, level=1, allocation=50, limit=75),                 -
                 (name=db2, level=1, allocation=30, limit=75),                 -
                 (name=db3, level=1, allocation=20, limit=75),                 -
                 (name=other, level=2, allocation=100))

CellCLI> ALTER IORMPLAN                                         -
         catPlan=((name=interactive, level=1, allocation=90),   -
                  (name=batch, level=2, allocation=80),         -
                  (name=maintenance, level=3, allocation=50),   -
                  (name=other, level=3, allocation=50)          -

CellCLI> ALTER IORMPLAN objective=’low_latency’

CellCLI> ALTER IORMPLAN objective=’auto’

CellCLI> ALTER IORMPLAN                                        -
         dbplan=((name=db1, limit=50),                         -
                 (name=db2, limit=50),                         -
                 (name=other, level=1, allocation=25))
```

Example 8–23 shows how to set flash cache use with the IORMPLAN attributes.

Example 8–23 Setting Flash Cache Use with IORMPLAN Attributes

```
CellCLI> ALTER IORMPLAN                                        -
         dbPlan=((name=sales_prod, flashCache=on),             -
                 (name=sales_dev, flashCache=on),              -
                 (name=sales_test, flashCache=off),            -
                 (name=other, flashCache=off))

CellCLI> ALTER IORMPLAN                                                     -
         dbPlan=((name=oltp, level=1, allocation=80, flashCache=on),        -
                 (name=dss, level=1, allocation=20, limit=50, flashCache=off), -
                 (name=other, level=2, allocation=100, flashCache=off))
```

Example 8–24 shows how to reset the `IORMPLAN` attributes.

Example 8–24 Resetting IORMPLAN Attributes

```
CellCLI> ALTER IORMPLAN dbPlan="", catPlan=""
CellCLI> ALTER IORMPLAN dbPlan=""
CellCLI> ALTER IORMPLAN catPlan=""
```

See Also:

- Chapter 6, "Managing I/O Resources" for additional information about IORM
- "Administering IORM" on page 6-8 for additional information about managing interdatabase plans
- "START and @" on page 8-7 for additional information about running commands as scripts

ALTER LUN

Purpose

The `ALTER LUN` command reenables all LUNs or specified LUNs.

Syntax

```
ALTER LUN { ALL  | lun1 [ , lun2] ...  }
  REENABLE FORCE
```

Usage Notes

This command creates the cell disk and grid disk metadata on a replacement disk.

This command rebuilds redundancy for the system area of the system disks even when the system LUN is in a normal state.

Caution: Data might be lost when using this command.

Examples

Example 8–25 shows the `ALTER` command with the `LUN` object.

Example 8–25 Reenabling a LUN

```
CellCLI> ALTER LUN 'x:7' REENABLE FORCE

CellCLI> ALTER LUN ALL REENABLE FORCE
```

ALTER PHYSICALDISK

Purpose

The ALTER PHYSICALDISK command sets the Service Action Required LED on or off.

Syntax

```
ALTER PHYSICALDISK { ALL [harddisk]  | disk_id1 [,disk_id2]  ...  }
  SERVICELED {ON | OFF}
```

Usage Notes

Turning on the Service Action Required LED on can be used to ensure work is done on the proper disk. Oracle recommends the LED be turned off after service.

The Service Action Required LED is turned off after the disk has been replaced or put back in the server.

Examples

Example 8–26 shows the ALTER command with the PHYSICALDISK object.

Example 8–26   Turning On a LED

```
CELLCLI> ALTER PHYSICALDISK 12:3 SERVICELED on
```

ALTER QUARANTINE

Purpose

The ALTER QUARANTINE command changes the attributes for a quarantine.

Syntax

```
ALTER QUARANTINE { ALL  | quarantine1 [,quarantine2]  ...  }
   attribute_name = attribute_value
   [, attribute_name = attribute_value]...
```

Usage Notes

Only modifiable fields can be changed.

Examples

Example 8–27 shows the ALTER command with the QUARANTINE object.

Example 8–27 Altering a Quarantine

```
CELLCLI> ALTER QUARANTINE 12 comment='bugX'
```

ALTER THRESHOLD

Purpose

The `ALTER THRESHOLD` command updates the attribute values of all thresholds or the specified thresholds.

Syntax

```
ALTER THRESHOLD { ALL |threshold_name [, threshold_name ...] }
   attribute_name = attribute_value
   [, attribute_name = attribute_value]...
```

Usage Notes

The attributes that can be changed with the `ALTER` command are shown as `modifiable` in Example 8–57, "Describing the THRESHOLD Object" on page 8-81.

Examples

Example 8–28 shows how to alter threshold attributes.

Example 8–28 Altering Threshold Attributes

```
CellCLI> ALTER THRESHOLD ct_io_wt_rq.interactive warning=10, critical=20, -
                comparison='=', occurrences=2, observation=10

CellCLI> ALTER THRESHOLD ALL occurrences=3
```

> **See Also:** "CREATE THRESHOLD" on page 8-46 for additional information about setting threshold attributes

ASSIGN KEY

Purpose

The `ASSIGN KEY` command assigns or removes a security key to or from a client.

Syntax

```
ASSIGN KEY FOR client1 = key-value1 [, client2 = key-value2]...
```

Usage Notes

- *`clientN`* is the unique name (DB_UNIQUE_NAME) for a client.
- *`key-valueN`* is the hexadecimal string key that is assigned to the client as a security key.
  - The key value can be the same for multiple clients.
  - An empty string for the *`key-valueN`* removes a previously assigned key.

For database-scoped security, the clients must be entered in the `availableTo` attribute of the `GRIDDISK` object.

The key value is generated with the `CREATE KEY` command.

The key values assigned with the `ASSIGN` command must match the keys in the `cellkey.ora` files on the Oracle ASM and database host computers.

Examples

Example 8–29 shows the `ASSIGN KEY` command.

Example 8–29 Assigning and Removing Keys to and from Clients

```
CellCLI> ASSIGN KEY FOR db0 ='b67d5587fe728118af47c57ab8da650a'

CellCLI> ASSIGN KEY FOR +asm='7c57ab8da650ab118587feaf467d5728'

CellCLI> ASSIGN KEY FOR  '+asm'='ed63f41779c262ddd34a00c0d83590b8',  -
                         'db1' ='118af47c57ab8da650ab67d5587fe728',  -
                         'db2' ='8a65313e8de6cd8bcbab7f4bdddb0498',  -
                         'db3' ='9140c767bd92d1b45783e7fe6520e6d'

CellCLI> ASSIGN KEY FOR db1='', db2='', db3='', +asm=''
```

See Also:

- "CREATE GRIDDISK" on page 8-42
- "CREATE KEY" on page 8-44
- Chapter 4, "Configuring Security for Oracle Exadata Storage Server Software" for additional information about Exadata Cell security
- "LIST KEY" on page 8-113 for additional information about displaying the key value for a client

CALIBRATE

Purpose

The `CALIBRATE` command runs raw performance tests on cell disks, enabling you to verify the disk performance before the cell is put online.

Syntax

```
CALIBRATE [FLASHDISK | HARDDISK] [FORCE]
```

Usage Notes

You must be logged on to the cell as the `root` user to run `CALIBRATE`.

The `FORCE` option enables you to run the tests when Cell Server is running. If you do not use the `FORCE` option, then Cell Server must be shut down. Running `CALIBRATE` at the same time as a Cell Server process impacts performance.

Use the `FLASHDISK` option to specify that only flash LUNs be calibrated.

Use the `HARDDISK` option to specify that only hard disk LUNs be calibrated.

Examples

Example 8–30 shows how to check disk performance.

Example 8–30 Checking Disk Performance with CALIBRATE on HP Oracle Exadata Storage Server

```
CellCLI> CALIBRATE
Calibration will take a few minutes...
Aggregate random read throughput across all luns: 994.19 MBPS
Aggregate random read IOs per second (IOPS) across all luns: 3926
Controller read throughput: 1056.31 MBPS
Lun 6_1 on drive [1I:6_1:1] random read throughput: 90.96 MBPS, and 332 IOPS
...
Lun 6_9 on drive [1I:6_1:6] random read throughput: 92.64 MBPS, and 333 IOPS
CALIBRATE results are within an acceptable range.
CALIBRATE stress test is now running...
Calibration has finished.
```

In Example 8–30, the results were acceptable compared to standard properties. A message is displayed if the performance is substandard.

Example 8–31 shows the output when using `CALIBRATE` with `FORCE` option on Oracle Exadata Storage Server.

Example 8–31 Output from CALIBRATE Command with FORCE Option on Oracle Exadata Storage Server

```
CellCLI> CALIBRATE FORCE
Calibration will take a few minutes...
Aggregate random read throughput across all hard disk luns: 1604 MBPS
Aggregate random read throughput across all flash disk luns: 4242.9 MBPS
Aggregate random read IOs per second (IOPS) across all hard disk luns: 4927
Aggregate random read IOs per second (IOPS) across all flash disk luns: 148695
Controller read throughput: 1608.05 MBPS
Calibrating hard disks (read only) ...
Lun 0_0  on drive [20:0      ] random read throughput: 153.41 MBPS, and 412 IOPS
```

```
Lun 0_1  on drive [20:1      ] random read throughput: 155.38 MBPS, and 407 IOPS
Lun 0_10 on drive [20:10     ] random read throughput: 155.32 MBPS, and 423 IOPS
Lun 0_11 on drive [20:11     ] random read throughput: 151.24 MBPS, and 427 IOPS
Lun 0_2  on drive [20:2      ] random read throughput: 152.70 MBPS, and 422 IOPS
Lun 0_3  on drive [20:3      ] random read throughput: 155.42 MBPS, and 423 IOPS
Lun 0_4  on drive [20:4      ] random read throughput: 153.14 MBPS, and 428 IOPS
Lun 0_5  on drive [20:5      ] random read throughput: 154.06 MBPS, and 424 IOPS
Lun 0_6  on drive [20:6      ] random read throughput: 150.82 MBPS, and 409 IOPS
Lun 0_7  on drive [20:7      ] random read throughput: 154.61 MBPS, and 426 IOPS
Lun 0_8  on drive [20:8      ] random read throughput: 154.46 MBPS, and 424 IOPS
Lun 0_9  on drive [20:9      ] random read throughput: 154.63 MBPS, and 426 IOPS
Calibrating flash disks (read only, note that writes will be significantly slower)
...
Lun 1_0  on drive [[10:0:0:0]] random read throughput: 269.11 MBPS, and 19635 IOPS
Lun 1_1  on drive [[10:0:1:0]] random read throughput: 268.86 MBPS, and 19648 IOPS
Lun 1_2  on drive [[10:0:2:0]] random read throughput: 268.68 MBPS, and 19645 IOPS
Lun 1_3  on drive [[10:0:3:0]] random read throughput: 268.92 MBPS, and 19640 IOPS
Lun 2_0  on drive [[12:0:0:0]] random read throughput: 269.78 MBPS, and 20436 IOPS
Lun 2_1  on drive [[12:0:1:0]] random read throughput: 269.69 MBPS, and 20394 IOPS
Lun 2_2  on drive [[12:0:2:0]] random read throughput: 269.04 MBPS, and 20439 IOPS
Lun 2_3  on drive [[12:0:3:0]] random read throughput: 269.51 MBPS, and 20420 IOPS
Lun 4_0  on drive [[9:0:0:0] ] random read throughput: 269.07 MBPS, and 19668 IOPS
Lun 4_1  on drive [[9:0:1:0] ] random read throughput: 269.24 MBPS, and 19697 IOPS
Lun 4_2  on drive [[9:0:2:0] ] random read throughput: 269.09 MBPS, and 19676 IOPS
Lun 4_3  on drive [[9:0:3:0] ] random read throughput: 269.03 MBPS, and 19681 IOPS
Lun 5_0  on drive [[11:0:0:0]] random read throughput: 268.06 MBPS, and 19714 IOPS
Lun 5_1  on drive [[11:0:1:0]] random read throughput: 268.24 MBPS, and 19696 IOPS
Lun 5_2  on drive [[11:0:2:0]] random read throughput: 268.33 MBPS, and 19717 IOPS
Lun 5_3  on drive [[11:0:3:0]] random read throughput: 268.14 MBPS, and 19693 IOPS
CALIBRATE results are within an acceptable range.

CALIBRATE stress test is now running...
Calibration has finished.
```

See Also: "ALTER CELL" on page 8-15 for additional information about shutting down cell services

CREATE

Purpose

The `CREATE` command creates a new object and assigns initial attributes to the object.

Syntax

```
CREATE object_type [name] ...
   [attribute_name = attribute_value [, attribute_name = attribute_value]...]
```

Usage Notes

- *`object_type`* can be as follows:
  - `CELL`
  - `CELLDISK`
  - `FLASHCACHE`
  - `GRIDDISK`
  - `KEY`
  - `QUARANTINE`
  - `THRESHOLD`
- *`name`* is the name of the new object.
- *`attribute_name`* is the name of a modifiable attribute for the *`object_type`*.
- *`attribute_value`* is the initial setting for the attribute.
- When multiple objects are valid as the target of a `CREATE` command, there is the possibility of partial success. If an error occurs, then the command is interrupted, and the remaining objects are not created.

  See Also:

  - "CREATE CELL" on page 8-37
  - "CREATE CELLDISK" on page 8-39
  - "CREATE FLASHCACHE" on page 8-41
  - "CREATE GRIDDISK" on page 8-42
  - "CREATE KEY" on page 8-44
  - "CREATE QUARANTINE" on page 8-45
  - "CREATE THRESHOLD" on page 8-46
  - "About CellCLI Object Types" on page 8-7 for additional information about object types and their descriptions

CREATE CELL

Purpose

The `CREATE CELL` command creates a cell object and assigns initial attributes to the object.

Syntax

```
CREATE CELL [name]
   interconnect1=intValue1 [, interconnect2=intValue2 ...]
   [  { , ipaddress1=ipValue1 [, ipaddress2=ipValue2 ...]  }  |
   { , ipBlock=ipBlkValue, cellNumber=number }   ]
   [, attributeName = attributeValue  ...]
```

Usage Notes

The attributes that can be set are shown as `modifiable` in Example 8–42, "Describing the CELL Object" on page 8-56.

- This command can be used to assign the `ASR` value to the `snmpSubscriber` attribute. See "ALTER CELL" on page 8-15 for additional details.
- The default cell name is set to the network host name of the cell with hyphens in the network name replaced with underscores. You can display the network name with the `uname -n` command. If you change the cell name, then you must choose a unique cell name.
- One to four interconnects can be specified. The `interconnect1` attribute must be specified if the `interconnect2` attribute is specified. The `interconnect1` and `interconnect2` attributes must be specified if `interconnect3` is specified, and so on.
- Interconnects are InfiniBand (`ibN`).
- If the `ipBlock` attribute is specified, then four network interconnects must be configured. In addition, if the `ipBlock` attribute is specified, then do not specify the `ipaddress` attributes.
- By default, the `CREATE CELL` command creates Exadata Smart Flash Cache cell disks on all flash LUNs, and creates Exadata Smart Flash Cache on the cell disks. Users can specify `FLASHCACHE=0` to bypass creation of flash cell disks and Exadata Smart Flash Cache on the cell disks. A non-zero value for Exadata Smart Flash Cache is taken as the total size for creating Exadata Smart Flash Cache. The size is divided evenly across the flash LUNs.
- The `ipBlock` and `cellNumber` attributes are alternate ways of setting all IP addresses and netmasks individually. The `ipBlock` attribute denotes the IP address block or prefix. It is a bit string that consists of a number of initial bits of an address, written as an address followed by a slash (/), and the number of initial bits. Examples of IP blocks are 10.5.0.0/16 and 101.15.20.0/24. An IP block can be abbreviated by omitting less significant zero fields, but there should be enough fields to contain the indicated number of initial bits. An example of an abbreviated IP block is 10.5/16.

  The network administrator usually determines the `ipBlock` value based on the range of addresses available. The `ipBlock` attribute is combined with the `cellNumber` attribute to compute the `ipaddress` for each interconnection used by the cell. The `cellNumber` becomes the least significant bits of the address. Each of the four cell interconnections is assigned an index (0, 1, 2, 3), and this becomes the high-order 2 bits within the range of addresses defined by the `ipBlock` attribute.

IP addresses can be set using `ipBlock` and `cellNumber` or by setting the addresses individually. Each set overrides previous set values. The IP addresses can be displayed by `LIST` commands, but they are not used to configure interconnections.

> **Note:** After a cell is created, the network configuration rarely needs to be changed.

- You cannot specify an `ipaddress` attribute if an `interconnect` attribute is not specified. A maximum of four `ipaddress` attributes can be specified.

Examples

Example 8–32 shows how to create a cell. In the example, the interconnections are set to existing InfiniBand connections.

Example 8–32 Creating a Cell

```
CellCLI> CREATE CELL cell02 interconnect1=bond0, -
           ipaddress1='10.142.0.221/24'

CellCLI> CREATE CELL cell22 interconnect1=bond0

CellCLI> CREATE CELL cell22          -
           interconnect1=bond0, -
           ipBlock='10.142.0/22', cellNumber=221
```

See Also:

- "ALTER CELL" on page 8-15 for information about changing the attributes
- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions for cells
- Example 2–1, "Configuring a New Cell" on page 2-3

CREATE CELLDISK

Purpose

The CREATE CELLDISK command creates a cell disk object and assigns initial attributes to the object. You can use the ALL option to automatically create cell disks.

Syntax

```
CREATE CELLDISK  {
   ALL [FLASHDISK | HARDDISK]
  |cdisk1 attribute_name=value,[attribute_name=value]... [FORCE]
  |((name=cdisk2,attribute_name=value,[attribute_name=value]...)
  [,(name=cdisk3,attributename=value,[attributename=value]...)]...)
  }
```

Usage Notes

The attributes that can be specified during creation are the cell disk name (*cdiskN*), comment, lun, and physicalDisk.

- Either lun or physicalDisk is required when adding a specifically-named cell disk.
  - When a physical disk is provided, a single-disk LUN is created, and that LUN is used to create the cell disk. The LUN is flagged as automatically-created.
  - When a LUN is provided, that device is used to create the cell disk.
- CREATE CELLDISK ALL is a shortcut command to create all candidate cell disks for the cell. This operation occurs in two steps:
  - All physical disks that are not already configured as LUNs are configured as single-disk LUNs (SDLs). These LUNs are flagged as automatically-created LUNs.
  - All LUNs that are not configured as cell disks are used to create cell disks. These cell disks are initially named according to the template CD_*lunname_cellname*. This name can later be changed using the ALTER CELLDISK command.

    Note: LUNs with a second or third extended file system (ext2/ext3) are ignored during the CREATE CELLDISK ALL operation.

- The FLASHDISK option limits the CREATE CELLDISK command to cell disks that are flash disks.
- The HARDDISK option limits the CREATE CELLDISK command to cell disks that are hard disks.
- The list form of CREATE CELLDISK enables you to add a series of cell disks in a single command.
- The FORCE keyword overrides the following error conditions:
  - The physical disk provided is already part of an existing LUN.
  - The LUN provided is already associated with a cell disk.

`FORCE` causes the LUN to be reused to create the new cell disk. Any preexisting configuration is lost. `FORCE` is not an option for `CREATE CELLDISK ALL` or for the list form of `CREATE CELLDISK`.

- The `INTERLEAVING` option should be used when you want to use interleaved grid disks. Setting this parameter to `normal_redundancy` changes the manner in which the cell disk sectors are allocated to grid disks. When using `normal_redundancy`, the cell disk sectors are allocated such that the outer portion of the grid disks consist of sectors from the first half of the cell disk. The inner portion of the grid disk is allocated sectors from the second half of the cell disk.

Cell disks are created automatically using the `CREATE CELLDISK ALL` command. This command creates single-disk LUNs from all available physical disks, and then creates cell disks from all available LUNs.

When a cell disk is created, metadata describing the cell disk is written to the cell disk itself and to the cell configuration files. Approximately 48 MB of the cell disk is allocated for the celldisk metadata partition. On a subsequent restart, Cell Server attempts to rediscover the created cell disk by reading configuration data on the disk. Any hardware changes in the cell might cause a change in the LUN and device name for a cell disk. The rediscovery mechanism that runs during the cell restart process changes the cell disk configuration accordingly.

Examples

Example 8–33 shows how to create a cell disk.

Example 8–33 Creating a Cell Disk

```
CellCLI> CREATE CELLDISK ALL

CellCLI> CREATE CELLDISK cdisk03 lun=03

CellCLI> CREATE CELLDISK cdisk04 physicalDisk='I2:1:2'

CellCLI> CREATE CELLDISK CD_03_cell04 lun=03

CellCLI> CREATE CELLDISK CD_05_cell09 physicalDisk='2I:1:2'

CellCLI> CREATE CELLDISK CD_06_cell10 INTERLEAVING='normal_redundancy'
```

See Also:

- "ALTER CELLDISK" on page 8-21 for information about changing the attributes
- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions
- Example 2–3, "Creating Cell Disks" on page 2-4

CREATE FLASHCACHE

Purpose

The CREATE FLASHCACHE command creates Exadata Smart Flash Cache on a cell for IO requests.

Syntax

```
CREATE FLASHCACHE [ALL [FLASHDISK]] [attribute_name = attribute_value
   [, attribute_name = attribute_value] ...]
```

Usage Notes

Cell disks defined on Exadata Smart Flash Cache cannot be exported.

The CREATE FLASHCACHE command accepts a list of comma-delimited flash cell disks. If a size is specified in the command, then that size is divided evenly across the cell disks, and totals the specified size. If a size is not specified, then all available space on each cell disk in the list is used for Exadata Smart Flash Cache.

The ALL FLASHDISK argument creates Exadata Smart Flash Cache on all flash cell disks. If the ALL argument is not specified, then the cell disk attribute must be specified. The FLASHDISK argument is not required.

Examples

Example 8–34 shows how to create Exadata Smart Flash Cache on a cell.

Example 8–34 Creating Exadata Smart Flash Cache

```
CellCLI> CREATE FLASHCACHE ALL

CellCLI> CREATE FLASHCACHE ALL SIZE=250g

CellCLI> CREATE FLASHCACHE ALL FLASHDISK

CellCLI> CREATE FLASHCACHE CELLDISK='fd1,fd2,fd3,fd4'
```

CREATE GRIDDISK

Purpose

The `CREATE GRIDDISK` command creates a grid disk object on a specified cell disk or creates one grid disk on each cell disk on the cell. The command also assigns initial attributes to the new grid disks.

Syntax

```
CREATE GRIDDISK  {  ALL [FLASHDISK | HARDDISK] PREFIX=gdisk_name_prefix
   | gdisk cellDisk = attribute_value  }
   [, attribute_name = attribute_value]...
```

Usage Notes

The attributes that can be specified are the grid disk name (*`gdisk`*), `cellDisk`, `size`, and `comment`. The `cellDisk` attribute is required when a single grid disk is created.

- If an individual name is entered with the command, then the grid disk is created on the cell disk specified by the `cellDisk` attribute. You must ensure that the grid disk name is unique across all cells. If the disk name is not unique, then it might not be possible to add the grid disk to an Oracle ASM disk group.
- The length of a grid disk name is limited to 30 characters.
- The `FLASHDISK` option limits the `CREATE GRIDDISK` command to cell disks that are flash disks.
- The `HARDDISK` option limits the `CREATE GRIDDISK` command to cell disks that are hard disks.
- If the `ALL PREFIX` option is entered with the command, then one grid disk is created on each cell disk on the cell. `PREFIX` is required when `ALL` is used.

  The `PREFIX` option specifies the prefix assigned to the names of the created grid disks. The generated grid disk names are composed of the grid disk prefix followed by an underscore (_) and then the cell disk name in the following form:

  ```
  gdisk_name_prefix_cdisk_name
  ```

  Choose a prefix for the grid disk name that matches the Oracle ASM disk group to which the grid disk belongs to help you identify which disks belong to a disk group. The generated grid disk (*`gdisk_name_prefix_cdisk_name`*) must follow the restrictions on the `name` value.

  For example, if the Oracle ASM disk group name is `data01`, then `data01` is used as the prefix for the grid disk names. If `CREATE GRIDDISK ALL PREFIX=data01` is run on a cell with cell disks `CD_01_cell01`, `CD_02_cell01`, and `CD_03_cell01`, then grid disks named `data01_CD_01_cell01`, `data01_CD_02_cell01`, and `data01_CD_03_cell01` are created on each cell disk respectively.
- The `CREATE GRIDDISK ALL` command skips disks which do not have enough free space for a minimum grid disk. A message stating which disks did not have enough free space appears, and the command continues.
- The `size` and `offset` attributes are optional attributes specified as a number in bytes, unless the suffix `M` (megabytes) or `G` (gigabytes) is included with the number, such as `size=300M`, or `size=150G`.

Grid disk space is allocated in 16 MB units, referred to as allocation units. If the size attribute is specified when creating a grid disk, then the size allocated is the size of the largest multiple of allocation units less than or equal to the specified size. For example, if `300M` is specified for the size attribute, then 288 MB (16x18) is allocated because 288 is the largest multiple of 16 that is less than or equal to 300.

A minimum of 1 allocation unit is always allocated, so the minimum size for a grid disk is 16 MB. Any size value less than 16 MB is rounded up to 16 MB.

Before specifying the `size` attribute, ensure that you have first determined the available free space on each target cell disk with the `LIST CELLDISK` command. For example, `LIST CELLDISK` *`cdisk`* `ATTRIBUTES freespace`. If the `size` attribute is not specified, then the maximum size is allocated.

Offset determines the position on the disk where the grid disk is allocated. The outermost tracks have lower offset values, and these tracks have greater speed and higher bandwidth. Offset can be explicitly specified to create grid disks that are relatively higher performing than other grid disks. If offset is not specified, then the best (warmest) available offset is chosen automatically in chronological order of grid disk creation. You should first create the grid disks that are expected to contain the most frequently accessed (hottest) data, and then create the grid disks that contain the relatively colder data.

- The value of the `availableTo` attribute is set to the names of the clients that you want to set up for database-scoped security. These clients were assigned security keys that match keys in the configuration files on the hosts.

  The value entered for a client name is the unique name (DB_UNIQUE_NAME). The specified clients are those that are allowed to access the grid disk. If a value is entered for `availableTo`, then only the specified clients have access to the grid disk; otherwise, any client can have access.

Examples

Example 8–35 shows how to create a grid disk.

Example 8–35 Creating a Grid Disk

```
CellCLI> CREATE GRIDDISK data1_CD_01_cell01 cellDisk=CD_01_cell01, size=200G

CellCLI> CREATE GRIDDISK data2_CD_02_cell01 cellDisk=CD_02_cell01, size=200G

CellCLI> CREATE GRIDDISK data3_CD_03_cell01 cellDisk=CD_03_cell01

CellCLI> CREATE GRIDDISK ALL PREFIX=data01, -
         availableTo='+asm,db1,db2'

CellCLI> CREATE GRIDDISK hr7_CD_07_cell01 cellDisk=CD_07_cell01, -
         availableTo='asm_hr,hrdb0'
```

See Also:

- "Restrictions on Values of Common Attributes" on page 8-9 for additional information about restrictions on the grid disk name
- "ALTER GRIDDISK" on page 8-22 on page 8-62 for information about changing the attributes

CREATE KEY

Purpose

The `CREATE KEY` command creates and displays a random hexadecimal string to assign client keys. The use of `CREATE KEY` ensures that the security key is in the correct format. This command provides a way to generate a key in the correct format, and it can be run on any cell.

Syntax

```
CREATE KEY
```

Usage Notes

The security key must be entered in the `cellkey.ora` configuration file on the computer hosts that contain clients for which you want to authorize access to a cell.

The key is also assigned to clients that access grid disk storage.

The key must be copied manually to the hosts and cells.

Examples

Example 8–36 shows the `CREATE` command with the `KEY` object.

Example 8–36 Creating a Key

```
CellCLI> CREATE KEY
         3452c64fec9a5800bbe48d4093269400
```

See Also:

- Chapter 4, "Configuring Security for Oracle Exadata Storage Server Software" for additional information about Exadata Cell security
- "ASSIGN KEY" on page 8-33

CREATE QUARANTINE

Purpose

The `CREATE QUARANTINE` command allows a quarantine to be created manually.

Syntax

```
CREATE QUARANTINE quarantineType=value[,attributename=value]...
```

Usage Notes

Manual creation of quarantines should be done in coordination with Oracle Support Services. In general, manual quarantines are created to proactively isolate SQL statements which are known to cause problems.

Examples

Example 8–37 shows the `CREATE` command with the `QUARANTINE` object.

Example 8–37 Creating a QUARANTINE

```
CELLCLI> CREATE QUARANTINE  quarantineType="SQLID", sqlid="5xnjp4cutc1s8"
```

See Also: "DESCRIBE QUARANTINE" on page 8-79

CREATE THRESHOLD

Purpose

The `CREATE THRESHOLD` command creates a threshold object that specifies the conditions for generation of a metric alert.

Syntax

```
CREATE THRESHOLD name attributename=value [, attributename=value]...
```

Usage Notes

The attributes that can be specified are `comparison`, `critical`, `occurrences`, `observation`, and `warning`.

- The *name* argument is required. The name is comprised of a metric name and an object name with the format `metricName.objectName`, such as `db_io_rq_sm_sec.db123` or `ct_io_wt_rq.interactive`.
- The `comparison` attribute is required with a condition value. The value must be `'<'`, `'<='`, `'='`, `'>='`, or `'>'`.
- The `occurrences` attribute specifies the number of consecutive measurements over the threshold value that trigger a state change.
- The `observation` attribute is the number of measurements over which measured values are averaged.
- A state change to the value set in `warning` or `critical` causes a stateful alert to be generated.

Examples

Example 8–38 shows how to create a threshold.

Example 8–38 Creating a Threshold

```
REM the following threshold is created on a database named db123
CellCLI> CREATE THRESHOLD db_io_rq_sm_sec.db123 comparison='>', critical=120

REM the following threshold is created on a cateogory named interactive
CellCLI> CREATE THRESHOLD ct_io_wt_sm.interactive warning=10, critical=20, -
         comparison='=', occurrences=2, observation=5
```

See Also:

- Chapter 7, "Monitoring and Tuning Oracle Exadata Storage Server Software" for information about metrics
- "DESCRIBE THRESHOLD" on page 8-81 for a description of the attributes
- "DESCRIBE METRICDEFINITION" on page 8-71 for a description of the `name` attribute
- "LIST ALERTDEFINITION" on page 8-98 for information about metrics and thresholds
- *Oracle Database Administrator's Guide* for additional information about categories

DESCRIBE

Purpose

The DESCRIBE command displays a list of attributes for the object type that is provided as an argument. The list of attributes indicates whether each attribute can be modified.

Syntax

```
DESCRIBE object_type
```

Usage Notes

- The *object_type* is one of the supported object types described in "About CellCLI Object Types" on page 8-7.
- The list of attributes can be used as arguments in the LIST command.
- DESCRIBE does not display all of the attributes for the objects.

See Also:

- "DESCRIBE ACTIVEREQUEST" on page 8-48
- "DESCRIBE ALERTDEFINITION" on page 8-51
- "DESCRIBE ALERTHISTORY" on page 8-52
- "DESCRIBE CELL" on page 8-54
- "DESCRIBE CELLDISK" on page 8-58
- "DESCRIBE FLASHCACHE" on page 8-60
- "DESCRIBE FLASHCACHECONTENT" on page 8-61
- "DESCRIBE GRIDDISK" on page 8-62
- "DESCRIBE IBPORT" on page 8-64
- "DESCRIBE IORMPLAN" on page 8-66
- "DESCRIBE KEY" on page 8-67
- "DESCRIBE LUN" on page 8-68
- "DESCRIBE METRICCURRENT" on page 8-70
- "DESCRIBE METRICDEFINITION" on page 8-71
- "DESCRIBE METRICHISTORY" on page 8-74
- "DESCRIBE PHYSICALDISK" on page 8-76
- "DESCRIBE QUARANTINE" on page 8-79
- "DESCRIBE THRESHOLD" on page 8-81
- "LIST" on page 8-95

DESCRIBE ACTIVEREQUEST

Purpose

The `DESCRIBE ACTIVEREQUEST` command displays a list of attributes for the `ACTIVEREQUEST` object type.

Syntax

```
DESCRIBE ACTIVEREQUEST
```

Usage Notes

Table 8–4 lists the attributes for the `DESCRIBE ACTIVEREQUEST` command.

Table 8–4 DESCRIBE ACTIVEREQUEST Attributes

Attribute	Description
asmDiskGroupNumber	Number of the Oracle ASM disk group.
asmFileIncarnation	Incarnation number of the Oracle ASM file.
asmFileNumber	Number of the Oracle ASM file.
consumerGroupID	Identifier of the consumer group.
consumerGroupName	Name of the consumer group.
dbID	Database unique name.
dbName	Database name.
dbRequestID	Identifier of the database request.
fileType	File type associated with the request.
id	Unique identifier of the active request.
instanceNumber	Instance number associated with the request.
ioBytes	Number of bytes of I/O against the grid disk in the current session.
ioBytesSoFar	Number of total bytes of I/O.
ioGridDisk	Grid disk used by a request.
ioOffset	Measure of the offset on the grid disk.
ioReason	Reason for I/O activity, such as a control-file read.
ioType	Type of active request, such as `file initialization`, `read`, `write`, `predicate pushing`, `filtered backup read`, or `predicate push read`.
name	Unique name of the active request.
objectNumber	Object number associated with the request.
parentID	Identifier of the parent request.

Table 8–4 (Cont.) DESCRIBE ACTIVEREQUEST Attributes

Attribute	Description
requestState	State of the active request, such as the following: ■ Accessing Disk ■ Computing Result ■ Network Receive ■ Network Send ■ Queued Extent ■ Queued for Disk ■ Queued for File Initialization ■ Queued for Filtered Backup Read ■ Queued for Network Send ■ Queued for Predicate Pushing ■ Queued for Read ■ Queued for Write ■ Queued in Resource Manager
sessionID	Identifier of the session.
sessionSerNumber	Serial number of the database session.
sqlID	Identifier of the SQL command associated with the request.
tableSpaceNumber	Tablespace number associated with the request.

Examples

Example 8–39 shows the `DESCRIBE` command with the `ACTIVEREQUEST` object.

Example 8–39 Describing the ACTIVEREQUEST Object

```
CellCLI> DESCRIBE ACTIVEREQUEST

         name
         asmDiskGroupNumber
         asmFileIncarnation
         asmFileNumber
         consumerGroupID
         consumerGroupName
         dbID
         dbName
         dbRequestID
         fileType
         id
         instanceNumber
         ioBytes
         ioBytesSofar
         ioGridDisk
         ioOffset
         ioReason
         ioType
         objectNumber
         parentID
         requestState
         sessionID
         sessionSerNumber
```

```
sqlID
tableSpaceNumber
```

DESCRIBE ALERTDEFINITION

Purpose

The `DESCRIBE ALERTDEFINITION` command displays a list of attributes for the `ALERTDEFINITION` object type.

Syntax

```
DESCRIBE ALERTDEFINITION
```

Usage Notes

Table 8–5 lists the attributes for the `DESCRIBE ALERTDEFINITION` command.

Table 8–5 DESCRIBE ALERTDEFINITION Attributes

Attribute	Description
alertShortName	Abbreviated name for the alert. If the alert is based on a metric, then the short name is the same as the corresponding metric `name` attribute.
alertSource	Source of the alert, such as `BMC` or `ADR`.
alertType	Type of the alert. Values are stateful or stateless. ■ Stateful alerts are automatically cleared on transition to `normal`. ■ Stateless alerts are never cleared. You can change the alert by setting the `examinedBy` attribute.
description	Description for the alert.
metricName	Metric name if the alert is based on a metric.
name	Identifier for the alert.

Examples

Example 8–40 shows the `DESCRIBE` command with the `ALERTDEFINITION` object.

Example 8–40 Describing the ALERTDEFINITION Object

```
CellCLI> DESCRIBE ALERTDEFINITION

         name
         alertShortName
         alertSource
         alertType
         description
         metricName
```

See Also: "Displaying Alert Definitions" on page 7-18 for additional information about alerts

DESCRIBE ALERTHISTORY

Purpose

The `DESCRIBE ALERTHISTORY` command displays a list of attributes for the `ALERTHISTORY` object type.

Syntax

```
DESCRIBE ALERTHISTORY
```

Usage Notes

Table 8–6 lists the attributes for the `DESCRIBE ALERTHISTORY` command.

Table 8–6 DESCRIBE ALERTHISTORY Attributes

Attribute	Description
alertAction	Recommended action to perform for this alert.
alertMessage	Brief explanation of the alert.
alertSequenceID	Unique sequence ID for the alert. When an alert changes its state, such as `warning` to `critical` or `critical` to `clear`, another occurrence of the alert is created with the same sequence number and a time stamp of the transition.
alertShortName	Abbreviated name for the alert. If the alert is based on a metric, then the short name is the same as the corresponding metric `name` attribute.
alertType	Type of the alert. Values are stateful or stateless. ■ Stateful alerts are automatically cleared on transition to `normal`. ■ Stateless alerts are never cleared. You can change the alert by setting the `examinedBy` attribute.
beginTime	Time stamp when an alert changes its state.
endTime	Time stamp for the end of the period when an alert changes its state.
examinedBy	Administrator who reviewed the alert.
failedMail	Intended e-mail recipient when a notification failed.
failedSNMP	Intended SNMP subscriber when a notification failed.
metricObjectName	Object, such as cell disk or grid disk, for which a metric threshold has caused an alert.
metricValue	Value of the metric that caused the alert.
name	Unique identifier for the alert.
notificationState	Number indicating progress in notifying subscribers to alert messages: ■ 0: never tried ■ 1: sent successfully ■ 2: retrying, up to 5 times ■ 3: 5 failed retries
sequenceBeginTime	Time stamp when an alert sequence ID is first created.

Table 8–6 (Cont.) DESCRIBE ALERTHISTORY Attributes

Attribute	Description
severity	Severity level. Values are `clear`, `info`, `warning`, or `critical`.

Examples

Example 8–41 shows the `DESCRIBE` command with the `ALERTHISTORY` object.

Example 8–41 Describing the ALERTHISTORY Object

```
CellCLI> DESCRIBE ALERTHISTORY

         name
         alertAction
         alertMessage
         alertSequenceID
         alertShortName
         alertType
         beginTime
         endTime
         examinedBy                modifiable
         failedMail
         failedSNMP
         metricObjectName
         metricValue
         notificationState
         sequenceBeginTime
         severity
```

See Also: "DESCRIBE METRICDEFINITION" on page 8-71

DESCRIBE CELL

Purpose

The DESCRIBE CELL command displays a list of attributes for the CELL object type.

Syntax

```
DESCRIBE CELL
```

Usage Notes

Table 8–7 lists the attributes for the DESCRIBE CELL command.

Table 8–7 DESCRIBE CELL Attributes

Attribute	Description
bmcType	Name and version of the BMC.
cellNumber	Number of the cell. It is used with ipBlock. Each cell is assigned a number within an ipBlock. This number becomes the low-order bits of the IP addresses within the ipBlock.
cellsrvStatus	Status of Cell Server.
cellVersion	Release number of the cell software.
cl_bbu_chargE	Charge of disk controller battery.
cl_bbu_temp	Temperature of disk controller battery.
comment	User-supplied text string.
cpuCount	Number of CPUs on the cell.
emailFormat	File format for e-mail messages. The value can be html or text.
events	String for *events++* that is passed to Cell Server for debugging and trace information purposes.
fanCount	Count of working fans and total fans, displayed as working/total.
fanStatus	Status of the fan. The value can be normal, warning, or critical.
id	Global unique identifier (GUID) supplied by the hardware vendor.
interconnect1-4	Interconnect 1 to 4 for the cell. For example: bondeth0.
interconnectCount	Number of network interconnects.
iorm_mode	I/O Resource Management objective for the cell.
iormBoost	Ratio of the cumulative number of positions in the I/O queue that were skipped because of IORM scheduling to the number of I/Os that where scheduled. This ratio is calculated by sampling the changes each minute in the two numbers.
ipaddress1-4	IP address 1 to 4 for the cell.
ipBlock	Block of IP addresses.
iorm_mode	
kernelVersion	Version of the host kernel software.
locatorLEDStatus	Status of cell LOCATE LED. The value can be on or off.

Table 8–7 (Cont.) DESCRIBE CELL Attributes

Attribute	Description
location	Physical location of the cell hardware supplied by the user.
makeModel	Make and model of the cell hardware supplied by the vendor.
metricCollection	Indicator for whether Management Server performs metric collection. Values are `TRUE` or `FALSE`. If set to `FALSE`, then all collection and alert mining is stopped. The default setting is `TRUE`.
metricHistoryDays	Number of days that metric history, and ADR files are retained. The default is 7 days.
msStatus	Status of Management Server.
name	Unique name for the cell.
notificationMethod	Notification method for alerts. The value should be `mail`, `snmp`, or `both`.
notificationPolicy	Indicator for severity alerts to be sent to subscribers. The value for `notificationPolicy` should be `none` or any combination of `critical`, `warning`, and `clear`.
offloadEfficiency	Ratio of data accessed by the storage cell to the data that is processed by the database. Larger values mean more offloading to the cell. Also known as cell efficiency ratio. The cell efficiency ratio starts at 1, and does not change until Cell Server (CELLSRV) is running and accessing disks. Then the ratio of the total MB accessed on the cell to the MB accessed on the host (for the cell) is computed every minute that CELLSRV is active by Management Server (MS). The ratio is averaged over the previous 24 hours, or less if MS has been running less than 24 hours. Hourly averages are stored so they are not lost during MS and CELLSRV outages.
powerCount	Count of power supplies, displayed as working/total.
powerStatus	Status of the power. The value can be `normal`, `warning`, or `critical`.
realmName	Name assigned as the cell realm by the user.
releaseVersion	Release number for the cell software, such as `11.2.2.3.0`.
rsStatus	Status of Restart Server.
smtpFrom	SMTP e-mail user name that appears in alert notifications.
smtpFromAddr	SMTP e-mail address from which alert notifications are sent.
smtpPort	SMTP e-mail server port used to send alert notifications.
smtpPwd	Password of the SMTP e-mail user that sends alert notifications.
smtpServer	SMTP e-mail server used to send alert notifications.
smtpToAddr	Address to which e-mail is sent. It can be a comma-delimited list in quotation marks to allow multiple subscribers to alerts.
smtpUser	SMTP e-mail user name that is displayed in alert notifications.
smtpUseSSL	Specification to use Secure Socket Layer (SSL) authentication.
snmpSubscriber	List of hosts that subscribe to the SNMP alert notifications.
status	Status of the cell.
temperatureReading	Current temperature (Celsius) of the cell obtained from the BMC.

Table 8–7 (Cont.) DESCRIBE CELL Attributes

Attribute	Description
temperatureStatus	Status of the temperature. The value can be `normal`, `warning`, or `critical`.
traceLevel	Level for which trace messages are written. This can be a valid Java logging level (`SEVERE`, `WARNING`, `INFO`, `CONFIG`, `FINE`, `FINER`, `FINEST`) or a valid Oracle Diagnostic Logging (ODL) logging level (`INCIDENT_ERROR:1`, `ERROR:1`, `WARNING:1`, `NOTIFICATION:1`, `NOTIFICATION:16`, `TRACE:1`, `TRACE:16`, `TRACE:32`).
upTime	Time (*`days`*, *`hours:minutes`*) since the system was restarted.

Examples

Example 8–42 shows the `DESCRIBE` command with the `CELL` object.

Example 8–42 Describing the CELL Object

```
CellCLI> DESCRIBE CELL

         name                    modifiable
         bmcType
         cellNumber              modifiable
         cellVersion
         comment                 modifiable
         cpuCount
         emailFormat             modifiable
         events                  modifiable
         fanCount
         fanStatus
         id
         interconnectCount
         interconnect1           modifiable
         interconnect2           modifiable
         interconnect3           modifiable
         interconnect4           modifiable
         iormBoost
         ipBlock                 modifiable
         ipaddress1              modifiable
         ipaddress2              modifiable
         ipaddress3              modifiable
         ipaddress4              modifiable
         kernelVersion
         locatorLEDStatus
         location                modifiable
         makeModel
         metricCollection        modifiable
         metricHistoryDays       modifiable
         notificationMethod      modifiable
         notificationPolicy      modifiable
         offloadEfficiency       modifiable
         powerCount
         powerStatus
         realmName               modifiable
         releaseVersion
         smtpFrom                modifiable
         smtpFromAddr            modifiable
         smtpPort                modifiable
```

```
smtpPwd                 modifiable
smtpServer              modifiable
smtpToAddr              modifiable
smtpUser                modifiable
smtpUseSSL              modifiable
snmpSubscriber          modifiable
status
temperatureReading
temperatureStatus
traceLevel              modifiable
upTime
cellsrvStatus
msStatus
rsStatus
```

See Also:

"ALTER CELL" on page 8-15 and "CREATE CELL" on page 8-37 for additional information about which attributes can be changed

DESCRIBE CELLDISK

Purpose

The `DESCRIBE CELLDISK` command displays a list of attributes for the `CELLDISK` object type.

Syntax

```
DESCRIBE CELLDISK
```

Usage Notes

Table 8–8 lists the attributes for the `DESCRIBE CELLDISK` command.

Table 8–8 DESCRIBE CELLDISK Attributes

Attribute	Description
comment	User comment for the cell disk.
creationTime	Time stamp when the cell disk was created.
deviceName	Operating system device name of the LUN used by the cell disk.
devicePartition	Operating system device name of the partition that is used by the cell disk.
diskType	The type of disk.
errorCount	Number of errors that occurred on the cell disk.
freeSpace	Amount of unused space available on the cell disk.
freeSpaceMap	Location and size of unused space on the cell disk.
id	Global unique identifier (GUID) that is generated when the cell disk is created.
interleaving	Set to `normal_redundancy` to indicate that grid disks should be allocated using the interleaving policy.
lun	Identifier of the LUN on which the cell disk is located.
name	Unique name of the cell disk.
physicalDisk	Name of the physical disk on which the cell disk is located.
raidLevel	Value of the RAID level that is currently used on the LUN. For example, `RAID 0`.
size	Total size of the cell disk.
status	Current status of the cell disk, such as `normal` or `importRequired`.

Examples

Example 8–43 shows the `DESCRIBE` command with the `CELLDISK` object.

Example 8–43 Describing the CELLDISK Object

```
CellCLI> DESCRIBE CELLDISK

         name                    modifiable
         comment                 modifiable
         creationTime
         deviceName
```

```
devicePartition
diskType
errorCount
freeSpace
freeSpaceMap
id
interleaving
lun
physicalDisk
raidLevel
size
status
```

See Also: "ALTER CELLDISK" on page 8-21 and "CREATE CELLDISK" on page 8-39 for additional information about the attributes that can be changed

DESCRIBE FLASHCACHE

Purpose

The DESCRIBE FLASHCACHE command displays a list of attributes for the FLASHCACHE object type.

Syntax

```
DESCRIBE FLASHCACHE
```

Usage Notes

Table 8–9 lists the attributes for DESCRIBE FLASHCACHE command.

Table 8–9 DESCRIBE FLASHCACHE Attributes

Attribute	Description
cellDisk	Cell disk names that contain Exadata Smart Flash Cache.
creationTime	Time stamp when the Exadata Smart Flash Cache was created.
degradedCelldisks	List of cell disks configured for cache but not currently available.
effectiveCacheSize	Usable cache size after deducting space on unavailable celldisks.
id	Global unique identifier (GUID) that is generated when the Exadata Smart Flash Cache is created.
name	Unique name of the Exadata Smart Flash Cache.
size	Total size of the Exadata Smart Flash Cache.
status	Current status of the Exadata Smart Flash Cache, such as `normal`, `warning` or `critical`.

Examples

Example 8–44 shows the DESCRIBE command with the FLASHCACHE object.

Example 8–44 Describing the FLASHCACHE Object

```
CellCLI> DESCRIBE FLASHCACHE

         name
         cellDisk
         creationTime
         degradedCelldisks
         effectiveCacheSize
         id
         size
         status
```

DESCRIBE FLASHCACHECONTENT

Purpose

The `DESCRIBE FLASHCACHECONTENT` command displays a list of attributes for the `FLASHCACHE` object type.

Syntax

```
DESCRIBE FLASHCACHECONTENT
```

Usage Notes

Table 8–10 lists the attributes for `DESCRIBE FLASHCACHECONTENT` command.

Table 8–10 DESCRIBE FLASHCACHECONTENT Attributes

Attribute	Description
cachedSize	Size, in bytes, cached for this object.
cacheKeepSize	Size, in bytes, cached in `keep` mode for this object.
dbID	Database unique name identifier.
dbUniqueName	Database unique name.
hitCount	Number of I/Os which read data from flash cache for this object.
hoursToExpiration	Time before this object is downgraded from `keep` section, if not accessed again.
missCount	Number of I/Os which read data from disk for this object.
objectNumber	Dictionary object number of the segment which contains the object in the database.
tableSpaceNumber	Tablespace number for the object number.

Examples

Example 8–45 shows the `DESCRIBE` command with the `FLASHCACHECONTENT` object.

Example 8–45 Describing the FLASHCACHECONTENT Object

```
CellCLI> DESCRIBE FLASHCACHECONTENT

         cachedKeepSize
         cachedSize
         dbID
         dbUniqueName
         hitcount
         hoursToExpiration
         missCount
         objectNumber
         tableSpaceNumber
```

DESCRIBE GRIDDISK

Purpose

The `DESCRIBE GRIDDISK` command displays a list of attributes for the `GRIDDISK` object type.

Syntax

```
DESCRIBE GRIDDISK
```

Usage Notes

Table 8–11 lists the attributes for the `DESCRIBE GRIDDISK` command.

Table 8–11 DESCRIBE GRIDDISK Attributes

Attribute	Description
ASMDeactivationOutcome	Indicator whether a grid disk can be deactivated without loss of data.
ASMModeStatus	Indicator shows the current Oracle ASM usage of a grid disk.
availableTo	Names of the clients that can access this grid disk.
cellDisk	Name of the cell disk that contains the grid disk.
comment	User-supplied text string.
creationTime	Time stamp when the grid disk was created.
diskType	The type of disk.
errorCount	Count of hardware errors detected by the cell disk containing this grid disk.
id	Global unique identifier (GUID) that is generated when the grid disk is created.
name	Unique name of the grid disk.
offset	Lowest byte offset of the grid disk on the physical disk.
size	Total size of the grid disk.
status	Current status of the grid disk, such as `active`, `inactive`, `not present` or `importRequired`.

Note: The ASMDeactivationOutcome and ASMModeStatus attributes are not listed when using the `DESCRIBE GRIDDISK` command. These attributes must be explicitly specified when using the `LIST GRIDDISK` command.

Examples

Example 8–46 shows the `DESCRIBE` command with the `GRIDDISK` object.

Example 8–46 Describing the GRIDDISK Object

```
CellCLI> DESCRIBE GRIDDISK

         name                    modifiable
```

```
availableTo             modifiable
cellDisk
comment                 modifiable
creationTime
diskType
errorCount
id
offset
size                    modifiable
status
```

See Also: "ALTER GRIDDISK" on page 8-22 and "CREATE GRIDDISK" on page 8-42 for additional information about the attributes that can be changed

DESCRIBE IBPORT

Purpose

The `DESCRIBE IBPORT` command displays a list of attributes for the IBPORT object type.

Syntax

```
DESCRIBE IBPORT
```

Usage Notes

Table 8–12 lists the attributes for the `DESCRIBE IBPORT` command.

Table 8–12 DESCRIBE IBPORT Attributes

Attribute	Description
activeState	Indicator whether the port is currently the active port for the bonded IP.
dataRate	The data rate of the InfiniBand port.
hcaFWVersion	The version of the host channel adapter firmware.
id	The Global unique identifier (GUID) of the InfiniBand port.
lid	The local identifier of the InfiniBand port. It is unique within the subnet, and the 16-bit identifiers are used within a network by switches for routing.
linkDowned	The number of times the port training state machine has failed the link error recovery process, and halted the link.
linkIntegrityErrs	The number of link integrity errors.
linkRecovers	The number of times the port training state machine has successfully completed the link error recovery process.
name	The name of the InfiniBand port.
physLinkState	The physical link state.
portNumber	The port number of the InfiniBand port.
rcvConstraintErrs	The number of received constraint errors experienced by the InfiniBand port.
rcvData	The number of 32-bit data words received by the InfiniBand port.
rcvErrs	The number of packets received at the InfiniBand port containing an error.
rcvRemotePhysErrs	The number of physical errors experienced at the InfiniBand port.
status	The link status.
symbolErrs	The number of minor link errors experienced at the InfiniBand port.
vl15Dropped	The number of incoming VL15 packets dropped at the InfiniBand port due to resource limitations, such as lack of buffers.
xmtConstraintErrs	The number of transmitted constraint errors experienced at the InfiniBand port.

Table 8–12 (Cont.) DESCRIBE IBPORT Attributes

Attribute	Description
xmtData	The number of 32-bit data words transmitted on the InfiniBand port.
xmtDiscards	The number of outbound packets discarded by the InfiniBand port because the port was down or congested.

Examples

Example 8–47 shows the `DESCRIBE` command with the `IBPORT` object.

Example 8–47 Describing the IBPORT Object

```
CellCLI> DESCRIBE IBPORT

        name
        activeSlave
        dataRate
        hcaFWVersion
        id
        lid
        linkDowned
        linkIntegrityErrs
        linkRecovers
        physLinkState
        portNumber
        rcvConstraintErrs
        rcvData
        rcvErrs
        rcvRemotePhysErrs
        status
        symbolErrs
        vl15Dropped
        xmtConstraintErrs
        xmtData
        xmtDiscards
```

DESCRIBE IORMPLAN

Purpose

The DESCRIBE IORMPLAN command displays a list of attributes for the IORMPLAN object type.

Syntax

```
DESCRIBE IORMPLAN
```

Usage Notes

Table 8–13 lists the attributes for the DESCRIBE IORMPLAN command.

Table 8–13 DESCRIBE IORMPLAN Attributes

Attribute	Description
catPlan	Allocation plan for the categories set up in the databases using the cell.
dbPlan	Allocation plan for the databases using the cell.
name	Unique name of the interdatabase plan. The name value is automatically set to *cellname*_IORMPLAN.
objective	Optimization mode for IORM.
status	Current status of the interdatabase plan, either active or inactive.

Examples

Example 8–48 shows the DESCRIBE command with the IORMPLAN object.

Example 8–48 Describing the IORMPLAN Object

```
CellCLI> DESCRIBE IORMPLAN

         name
         catPlan                 modifiable
         dbPlan                  modifiable
         objective               modifiable
         status
```

See Also: "ALTER IORMPLAN" on page 8-25 for additional information about the attributes that can be changed

DESCRIBE KEY

Purpose

The `DESCRIBE KEY` command displays a list of attributes for the `KEY` object type.

Syntax

```
DESCRIBE KEY
```

Usage Notes

Table 8–14 lists the attributes for the `DESCRIBE KEY` command.

Table 8–14 DESCRIBE KEY Attributes

Attribute	Description
key	Random hexadecimal string used to assign client keys.
name	Name of the key. The value of this field is not displayed with `LIST`.

Examples

Example 8–49 shows the `DESCRIBE` command with the `KEY` object.

Example 8–49 Describing the KEY Object

```
CellCLI> DESCRIBE KEY

         name
         key                       modifiable
```

See Also: "CREATE KEY" on page 8-44

DESCRIBE LUN

Purpose

The `DESCRIBE LUN` command displays a list of attributes for the `LUN` object type.

Syntax

```
DESCRIBE LUN
```

Usage Notes

Table 8–15 lists the attributes for the `DESCRIBE LUN` command.

Table 8–15 DESCRIBE LUN Attributes

Attribute	Description
cellDisk	Name of the cell disk associated with the LUN. This is empty if the LUN is not associated with a cell disk.
deviceName	Operating system device name for the LUN. For example, `/dev/c1d5`
diskType	The type of disk.
errorCount	Number of errors on this LUN.
id	Identifier assigned by the system.
isSystemLun	Indicator whether the disk is a system disk. If value is `TRUE`, then the disk is a system disk. If the value is `FALSE`, then the disk is not a system disk, and only has data on it.
lunAutoCreate	Indicator whether the LUN was automatically created. Values are `TRUE` or `FALSE`. Single-disk LUNs are automatically created when disks without LUNs are converted to cell disks. For example, `CREATE DISK ALL` creates single-disk LUNs for all physical disks without LUNs. These single-disk LUNs have this attribute set to `TRUE`.
lunSize	Raw size of the LUN before being converted to a cell disk.
lunUID	Unique identifier assigned by the system.
lunWriteCacheMode	Status of LUN write cache. The status can be in `Write Through Mode` or `Write Back Mode`.
name	Unique name assigned to the LUN. This might be different (or extended from) the LUN ID if the ID is not unique.
overProvisioning	Indicator of the percentage of overprovisioned blocks in flash storage that are still available for a particular LUN. This attribute is only used for flash disks.
physicalDrives	Physical disk names that form the LUN.
raidLevel	Value of the RAID level that is used on the LUN. For example: `RAID 0`.
status	Status of the LUN, which can be `normal`, `warning`, or `critical`.

Examples

Example 8–50 shows the `DESCRIBE` command with the `LUN` object.

Example 8–50 Describing the LUN Object

```
CellCLI> DESCRIBE LUN

         name
         cellDisk
         deviceName
         diskType
         errorCount
         id
         lunAutoCreate
         isSystemLun
         lunSize
         lunUID
         lunWriteCacheMode
         overProvisioning
         physicalDrives
         raidLevel
         status
```

DESCRIBE METRICCURRENT

Purpose

The `DESCRIBE METRICCURRENT` command displays a list of attributes for the `METRICCURRENT` object type.

Syntax

```
DESCRIBE METRICCURRENT
```

Usage Notes

Table 8–16 lists the attributes for the `DESCRIBE METRICCURRENT` command.

Table 8–16 DESCRIBE METRICCURRENT Attributes

Attribute	Description
alertState	Indicator of the alert state. Values are `normal`, `warning`, or `critical`.
collectionTime	Time stamp when the metric value was collected.
metricObjectName	Name of the object, such as cell disk, grid disk, and consumer group, being measured.
metricType	Specification for how the statistic was created or defined.
metricValue	Value of the metric when it was collected.
name	Unique name of the current metric.
objectType	Type of object being measured. Values are `CELL`, `CELLDISK`, `FLASHCACHE`, `GRIDDISK`, `IORM_CATEGORY`, `IORM_DATABASE`, `IORM_CONSUMER_GROUP`, or `HOST_INTERCONNECT`.

Examples

Example 8–51 shows the `DESCRIBE` command with the `METRICCURRENT` object.

Example 8–51 Describing the METRICCURRENT Object

```
CellCLI> DESCRIBE METRICCURRENT

         name
         alertState
         collectionTime
         metricObjectName
         metricType
         metricValue
         objectType
```

See Also: "DESCRIBE METRICDEFINITION" on page 8-71 for additional information about the attributes

DESCRIBE METRICDEFINITION

Purpose

The `DESCRIBE METRICDEFINITION` command displays a list of attributes for the `METRICDEFINITION` object type.

Syntax

```
DESCRIBE METRICDEFINITION
```

Usage Notes

Table 8–17 lists the attributes for the `DESCRIBE METRICDEFINITION` command.

Table 8–17 DESCRIBE METRICDEFINITION Attributes

Attribute	Description
description	Description of the metric.
metricType	Indicator of how the statistic was created or defined. The options are as follows: ■ `cumulative`: Cumulative statistics since the metric was created. ■ `instantaneous`: Value at the time that the metric is collected. ■ `rate`: Rates computed by averaging statistics over observation periods. ■ `transition`: Transition metrics are collected at the time their value has changed and typically capture important transitions in hardware status.

Table 8–17 (Cont.) DESCRIBE METRICDEFINITION Attributes

Attribute	Description
name	Unique name of the metric definition. The value of the `name` attribute is a composite of abbreviations. The attribute value starts with an abbreviation of the object type on which the metric is defined: ■ `CL_` (cell) ■ `CD_` (cell disk) ■ `GD_` (grid disk) ■ `FC_` (flash cache) ■ `DB_` (database-level) ■ `CG_` (consumer group, database-qualified) ■ `CT_` (category) ■ `N_` (network) After the abbreviation of the object type, most of the `name` attributes contain one of the following combinations to identify the operation: ■ `IO_RQ` (number of requests) ■ `IO_BY` (number of MB) ■ `IO_TM` (I/O latency) ■ `IO_WT` (I/O wait time) Next, in the name could be `_R` or `_W` for read or write. Following that in the `name` attribute value there might be `_SM` or `_LG` to identify small or large blocks, respectively. At the end of the name, there could be `_SEC` to signify per seconds or `_RQ` to signify per request. For consumer group and category metrics, read or write details are omitted. For example: ■ `CD_IO_RQ_R_SM` is the number of requests to read small blocks on a cell disk. ■ `GD_IO_TM_W_LG` is the microseconds of I/O latency writing large blocks on a grid disk.
objectType	Options are `CELL`, `CELLDISK`, `CELL_FILESYSTEM`, `GRIDDISK`, `IORM_CATEGORY`, `IORM_DATABASE`, `IORM_CONSUMER_GROUP`, or `HOST_INTERCONNECT`.
persistencePolicy	Amount of time metric values are stored.
unit	Unit for the metric explicitly, and is related to the metric collected: ■ Number ■ % (percentage) ■ F (Fahrenheit) ■ C (Celsius)

Examples

Example 8–52 shows the `DESCRIBE` command with the `METRICDEFINITION` object.

Example 8–52 Describing the METRICDEFINITION Object

```
CellCLI> DESCRIBE METRICDEFINITION
```

```
name
description
metricType
objectType
persistencePolicy
unit
```

DESCRIBE METRICHISTORY

Purpose

The `DESCRIBE METRICHISTORY` command displays a list of attributes for the `METRICHISTORY` object type.

Syntax

```
DESCRIBE METRICHISTORY
```

Usage Notes

Table 8–18 lists the attributes for the `DESCRIBE METRICHISTORY` command.

Table 8–18 DESCRIBE METRICHISTORY Attributes

Attribute	Description
alertState	Indicator of the alert state. Values are `normal`, `warning`, or `critical`.
collectionTime	Time stamp when the metric value was collected.
memory	The metrics in Management Server memory. This attribute can be used instead of the `collectionTime` attribute when the collection time is less than an hour.
metricObjectName	Name of the object, such as cell disk, grid disk, and consumer group, being measured.
metricType	Specification for how the statistic was created or defined.
metricValue	Value of the metric when it was collected.
metricValueAvg	Average value of the metric.
metricValueMax	Maximum value of the metric.
metricValueMin	Minimum value of the metric.
name	Unique name of the current metric.
objectType	Type of object being measured. Values are `CELL`, `CELLDISK`, `GRIDDISK`, `IORM_CATEGORY`, `IORM_DATABASE`, `IORM_CONSUMER_GROUP`, or `HOST_INTERCONNECT`.
over	Specification for amount of time in minutes for the aggregation. This attribute works in conjunction with the `max`, `min` and `avg` keywords.

Examples

Example 8–53 shows the `DESCRIBE` command with the `METRICHISTORY` object.

Example 8–53 Describing the METRICHISTORY Object

```
CellCLI> DESCRIBE METRICHISTORY

         name
        alertState
         collectionTime
         memory
         metricObjectName
         metricType
         metricValue
```

```
metricValueAvg
metricValueMax
metricValueMin
objectType
over
```

DESCRIBE PHYSICALDISK

Purpose

The `DESCRIBE PHYSICALDISK` command displays a list of attributes for the `PHYSICALDISK` object type.

Syntax

```
DESCRIBE PHYSICALDISK
```

Usage Notes

Table 8–19 lists the attributes for the `DESCRIBE PHYSICALDISK` command.

Table 8–19 DESCRIBE PHYSICALDISK Attributes

Attribute	Description
cmdTimeoutErrors	Count of command timeout errors.
ctrlFirmware	Controller firmware name.
ctrlHwVersion	Controller version. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
deviceId	Identifier for the disk within its enclosure. This attribute is only applicable to Exadata Cell on Oracle Exadata Storage Server.
diskType	Type of the disk, whether it is a `HARDDISK` or a `FLASHDISK`.
enclosureDeviceId	Identifier for the disk enclosure. This attribute is only applicable to Exadata Cell on Oracle Exadata Storage Server.
errCmdTimeoutCount	Count of the number of command timeout SCSI errors on the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
errHardReadCount	Count of the number of hard read errors on the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
errHardWriteCount	Count of the number of hard write errors on the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
errMediaCount	Count of the number of media errors on the disk. This attribute is only applicable to Exadata Cell on Oracle Exadata Storage Server.
errOtherCount	Count of unclassified errors. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
errSeekCount	Count of the number of SCSI seek errors on the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
errorCount	Total count of all errors on the disk.
foreignState	Indicator if disk has moved from its configured location. A `TRUE` setting indicates the disk has moved. This attribute is only applicable to Exadata Cell on Oracle Exadata Storage Server.
hotPlugCount	Count of the times the disk has been inserted into the rack. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
id	Identifier assigned by the system.

Table 8–19 (Cont.) DESCRIBE PHYSICALDISK Attributes

Attribute	Description
lastFailureReason	Type of the most-recent error on the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
luns	List of LUNs converted from this physical disk.
makeModel	Model description provided by the system.
name	Unique name of the physical disk.
notPresentSince	Time that a disk was removed.
physicalFirmware	System-assigned name of the firmware for the disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
physicalInsertTime	Time that the disk was inserted.
physicalInterface	Interface type used by the disk. For example, `SAS`
physicalPort	Port on the controller for this disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
physicalRPM	Disk revolutions per minute. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
physicalSerial	System-assigned unique ID.
physicalSize	Size of the disk in bytes.
physicalUseType	Intended use of the disk. For example, `Data Drive`.
sectorRemapCount	Count of all sector remap operations for the physical disk. This attribute is only applicable to Exadata Cell on HP Oracle Exadata Storage Server.
slotNumber	Physical location of disk. This attribute is only applicable to Exadata Cell on Oracle Exadata Storage Server.
status	Status of the physical disk. Values can be as follows: ■ `critical`: disk has failed ■ `normal`: disk is functioning normally ■ `not present`: disk has been removed ■ `offline`: disk is offline ■ `poor performance`: disk is performing poorly ■ `predictive failure`: disk is expected to fail ■ `rebuilding`: controller is rebuilding LUN ■ `unknown`: MS cannot determine the current status ■ `warning`: disk is degraded

Examples

Example 8–54 shows the `DESCRIBE` command with the `PHYSICALDISK` object on Oracle Exadata Storage Server.

Example 8–54 Describing the PHYSICALDISK Object on Oracle Exadata Storage Server

```
cellcli> DESCRIBE PHYSICALDISK

        name
        ctrlFirmware
        deviceId
```

```
         diskType
         enclosureDeviceId
         errMediaCount
         errorCount
         foreignState
         id
         luns
         makeModel
         notPresentSince
         physicalInsertTime
         physicalInterface
         physicalSerial
         physicalSize
         physicalUseType
         slotNumber
         status
```

Example 8–55 shows the `DESCRIBE` command with the `PHYSICALDISK` object on HP Oracle Exadata Storage Server.

Example 8–55 Describing the PHYSICALDISK Object on HP Oracle Exadata Storage Server

```
cellcli> DESCRIBE PHYSICALDISK

         name
         ctrlFirmware
         ctrlHwVersion
         errCmdTimeoutCount
         errHardReadCount
         errHardWriteCount
         errSeekCount
         errorCount
         errOtherCount
         hotPlugCount
         id
         lastFailureReason
         luns
         makeModel
         notPresentSince
         physicalFirmware
         physicalInsertTime
         physicalInterface
         physicalPort
         physicalRPM
         physicalSerial
         physicalSize
         physicalUseType
         sectorRemapCount
         status
```

DESCRIBE QUARANTINE

Purpose

The `DESCRIBE QUARANTINE` command displays a list of quarantines.

Syntax

```
DESCRIBE QUARANTINE
```

Usage Notes

Table 8–21 lists the attributes for the `DESCRIBE QUARANTINE` command.

Table 8–20 DESCRIBE QUARANTINE Attributes

Attribute	Description
name	Identifier of the quarantine.
cellsrvChecksum	Checksum of the CELLSRV binary.
clientPID	The process identifier for the client process which crashed the cell.
comment	Comment for the quarantine.
crashReason	Reason for the crash.
creationTime	Quarantine creation time.
dbUniqueID	The database unique ID for the quarantine.
dbUniqueName	The database unique name for the quarantine.
incidentID	The incident identifier of the crash that caused the quarantine creation.
ioBytes	The bytes of quarantined disk region. This is applicable to disk region quarantine only.
ioGridDisk	The grid disk name for quarantined disk region. This is applicable to disk region quarantine only.
ioOffset	The I/O offset for quarantined disk region. This is applicable to disk region quarantine only.
planLineID	The SQL Plan Line identifier. This is applicable to SQL Plan quarantine only.
quarantineReason	The reason for creation of the quarantine.
quarantineType	The type of quarantine created.
remoteHostName	The host name of the remote host that ran the client process that crashed the cell.
rpmVersion	The RPM version of the cell being used when the cell crashed.
sqlID	The SQLID of the SQL statement that crashed a cell.
sqlPlanHashValue	The SQL Plan hash value. This is applicable to SQL Plan quarantine only.

Examples

Example 8–56 shows the `DESCRIBE` command with the `QUARANTINE` object.

Example 8–56 Describing the QUARANTINE Object

```
CellCLI> DESCRIBE QUARANTINE
         name
         cellsrvChecksum
         clientPID
         comment                 modifiable
         crashReason
         creationTime
         dbUniqueID
         dbUniqueName
         incidentID
         ioBytes
         ioGridDisk
         ioOffset
         planLineID
         quarantineReason
         quarantineType
         remoteHostName
         rpmVersion
         sqlID
         sqlPlanHashValue
```

DESCRIBE THRESHOLD

Purpose

The DESCRIBE THRESHOLD command displays a list of attributes for the THRESHOLD object type.

Syntax

```
DESCRIBE THRESHOLD
```

Usage Notes

Table 8–21 lists the attributes for the DESCRIBE THRESHOLD command.

Table 8–21 DESCRIBE THRESHOLD Attributes

Attribute	Description
comparison	Operator for comparing the metric value to the threshold value (>, >=, =, <, <=) to determine whether the value violates the threshold.
critical	Limit beyond which the metric value is considered to be in the critical state for generating alerts.
name	Unique name of the threshold.
observation	Number of measurements over which the rate metric is averaged before being compared with the threshold value.
occurrences	Number of consecutive violations of the threshold limit by the metric value before the appropriate alert is issued.
warning	Limit beyond which the metric value is considered to be in the warning state for generating alerts.

Examples

Example 8–57 shows the DESCRIBE command with the THRESHOLD object.

Example 8–57 Describing the THRESHOLD Object

```
CellCLI> DESCRIBE THRESHOLD

         name
         comparison             modifiable
         critical               modifiable
         observation            modifiable
         occurrences            modifiable
         warning                modifiable
```

DROP

Purpose

The `DROP` command removes the named objects from the cell or resets a cell.

Syntax

```
DROP object_type [object_name [, object_name]...] [options]
```

Usage Notes

- *`object_type`* can be one of the following:
  - `ALTERHISTORY`
  - `CELL`
  - `CELLDISK`
  - `FLASHCACHE`
  - `GRIDDISK`
  - `QUARANTINE`
  - `THRESHOLD`
- *`object_name`* is the name of a cell disk, grid disk, or threshold.
- When multiple objects are the target of a `DROP` command, there is the possibility of partial success. If an error occurs, then the command is interrupted, and the remaining objects are not dropped.

  See Also:

  - "DROP ALERTHISTORY" on page 8-83
  - "DROP CELL" on page 8-84
  - "DROP CELLDISK" on page 8-85
  - "DROP FLASHCACHE" on page 8-87
  - "DROP GRIDDISK" on page 8-88
  - "DROP QUARANTINE" on page 8-90
  - "DROP THRESHOLD" on page 8-91
  - "About CellCLI Object Types" on page 8-7 for additional information about object types

DROP ALERTHISTORY

Purpose

The `DROP ALERTHISTORY` command removes alerts from the alert history of a cell.

Syntax

```
DROP ALERTHISTORY {ALL | alert1 {, alert2}, ...}
```

Usage Notes

- In the command, *alertN* is the name of the alert to be dropped from the history.
- When dropping stateful alerts, you must drop all members of the alert sequence at the same time. If you do not drop all members, then an error is issued by the system.

Examples

Example 8–58 shows the `DROP ALERTHISTORY` command.

Example 8–58 Dropping a Cell Alert History

```
CellCLI> DROP ALERTHISTORY 1, 2_1, 2_2
```

DROP CELL

Purpose

The DROP CELL command resets a cell to its original state.

Syntax

```
DROP CELL [ERASE = value] [FORCE]
```

Usage Notes

- All cell disks, grid disks, and thresholds are dropped. The interdatabase plan is reset to its default state. All cell attributes are set to default values.
- This command is run from within the cell.
- The ERASE option erases the content on the disk by overwriting the content. The values are as follows:
  - 1pass: One pass, and the content is overwritten with zeros. This value is not available for flash drives.
  - 3pass: Three passes, and the content is overwritten with set data patterns. This option follows the recommendations of NNSA. This value is not available for flash drives.
  - 7pass: Seven passes, and the disk is overwritten with set data patterns. This option follows the recommendations from DOD.
- When dropping all cells using the 1pass or 3pass option, it necessary to drop the flash disks first using the 7pass option, and then drop the cells. The following is an example of the commands:

  ```
  CellCLI> DROP CELLDISK ALL FLASHDISK ERASE=7pass
  CellCLI> DROP CELL ERASE=1pass
  ```

- The FORCE option is required if the grid disks are configured on any cell disks when DROP CELL is issued. Otherwise, an error is reported.

Examples

Example 8–59 shows the DROP CELL command with the FORCE option.

Example 8–59 Dropping a Cell

```
CellCLI> DROP CELL FORCE
```

DROP CELLDISK

Purpose

The DROP CELLDISK command removes all or the named cell disks from the cell.

This command is necessary if a cell disk fails, or it is replaced by a newer model.

Before dropping the cell disk, you should drop its grid disks and the corresponding Oracle ASM disks from the disk groups. The Oracle ASM disks should be dropped before dropping the grid disks.

Syntax

```
DROP CELLDISK  { ALL [FLASHDISK | HARDDISK] | cdisk_name [, cdisk_name]... }
[ERASE = value [NOWAIT]] [FORCE]
```

Usage Notes

- If individual cell disks are specified, then the named cell disks (*cdisk_name*) are dropped.
- If the ALL option is specified, then all the cell disks on the cell are removed.
- The FLASHDISK option limits the DROP CELLDISK command to cell disks that are flash disks.
- The HARDDISK option limits the DROP CELLDISK command to cell disks that are hard disks.
- If grid disks are configured on the cell disk when DROP CELLDISK is issued, then the FORCE option must be used or an error is reported. The FORCE option causes any grid disks to be dropped first, and then the cell disk is dropped.
- The ERASE option erases the content on the disk by overwriting the content. The values are as follows:
  - 1pass: One pass, and the content is overwritten with zeros. This option is not applicable for flash drives. This value is not available for flash drives.
  - 3pass: Three passes, and the content is overwritten with set data patterns. This option follows the recommendations from NNSA. This option is not applicable for flash drives. This value is not available for flash drives.
  - 7pass: Seven passes, and the disk is overwritten with set data patterns. This option follows the recommendations from DOD.
- When dropping all cell disks using the 1pass or 3pass option, it necessary to drop the flash disks first using the 7pass option, and then drop the cell disks. The following is an example of the commands:

  ```
  CellCLI> DROP CELLDISK ALL FLASHDISK ERASE=7pass
  CellCLI> DROP CELLDISK ALL ERASE=1pass
  ```

- Use the NOWAIT option with the ERASE option to run the command asynchronously.
- If the LUN associated with the CELLDISK is flagged as automatically created, then that LUN is deleted along with the cell disk.

Examples

Example 8–60 shows the DROP CELLDISK command.

Example 8–60 Dropping a Cell Disk

```
CellCLI> DROP CELLDISK CD_03_cell01

CellCLI> DROP CELLDISK CD_02_cell06 FORCE

CellCLI> DROP CELLDISK ALL

CellCLI> DROP CELLDISK CD_02_cell09 ERASE=1pass NOWAIT
CellDisk CD_02_cell09 erase is in progress
```

DROP FLASHCACHE

Purpose

The `DROP FLASHCACHE` command removes Exadata Smart Flash Cache from a cell.

Syntax

```
DROP FLASHCACHE
```

Usage Notes

The `DROP FLASHCACHE` command removes Exadata Smart Flash Cache from the cell.

Examples

Example 8–61 shows how to remove Exadata Smart Flash Cache from a cell.

Example 8–61 Removing Exadata Smart Flash Cache

```
CellCLI> DROP FLASHCACHE
```

DROP GRIDDISK

Purpose

The `DROP GRIDDISK` command removes the named grid disks from the cell or removes all the grid disks specified by the `ALL PREFIX` option.

> **Caution:** Before dropping a grid disk that belongs to an Oracle ASM disk group, ensure that the corresponding disk was dropped from the Oracle ASM disk group.

Syntax

```
DROP GRIDDISK {ALL [FLASHDISK | HARDDISK ] PREFIX=gdisk_name_prefix | gdisk_name
              [, gdisk_name]... } [ERASE = value [NOWAIT]]  [FORCE]
```

Usage Notes

- If the *gdisk_name* is entered, then the name identifies the individual grid disk to be removed. Multiple names can be entered.
- If the `ALL PREFIX` option is entered, then the *gdisk_name_prefix* option specifies the prefix assigned to the names of the grid disks to be dropped. The `PREFIX` option is required when `ALL` is used.
- The `FLASHDISK` option limits the `DROP GRIDDISK` command to grid disks that are flash disks.
- The `HARDDISK` option limits the `DROP GRIDDISK` command to grid disks that are hard disks.
- If any of the grid disks are in use when `DROP GRIDDISK` is issued, then an error is reported. You can use `ALTER GRIDDISK` with the `INACTIVE` option to deactivate a grid disk before dropping the grid disk. This action ensures that the grid disk is not in use.
- The `FORCE` option can be used to force the drop of a grid disk that is in use.
- If the grid disk being dropped was created on a cell disk of type `FLASHDISK`, then it does not re-create that area or any part of that grid disk or cell disk as `FLASHCACHE` automatically. Use the `CREATE FLASHCACHE` command to reuse any part of the dropped area for `FLASHCACHE`.
- The `ERASE` option erases the content on the disk by overwriting the content. The values are as follows:
  - `1pass`: One pass, and the content is overwritten with zeros. This option is not applicable for flash drives. This value is not available for flash drives.
  - `3pass`: Three passes, and the content is overwritten with set data patterns. This option follows the recommendations from NNSA. This option is not applicable for flash drives. This value is not available for flash drives.
  - `7pass`: Seven passes, and the disk is overwritten with set data patterns. This option follows the recommendations from DOD.
- When dropping all grid disks using the `1pass` or `3pass` option, it necessary to drop the flash disks first using the `7pass` option, and then drop the grid disks. The following is an example of the commands:

  ```
  CellCLI> DROP GRIDDISK ALL FLASHDISK PREFIX=data, ERASE=7pass
  ```

```
CellCLI> DROP GRIDDISK ALL PREFIX=data, ERASE=1pass
```

- Use the `NOWAIT` option with the `ERASE` option to run the command asynchronously.

Examples

Example 8–62 shows the `DROP GRIDDISK` command.

Example 8–62 Dropping a Grid Disk

```
CellCLI> ALTER GRIDDISK data01_CD_03_cell01 INACTIVE

CellCLI> DROP GRIDDISK data01_CD_03_cell01

CellCLI> DROP GRIDDISK ALL PREFIX=data01

CellCLI> DROP GRIDDISK data02_CD_04_cell01 FORCE

CellCLI> DROP GRIDDISK data02_CD_04_cell01 ERASE=1pass
GridDisk data02_CD_04_cell01 successfully dropped

CellCLI> DROP GRIDDISK ALL FLASHDISK PREFIX=DATA, ERASE=7pass
CellCLI> DROP GRIDDISK ALL PREFIX=DATA, ERASE=3pass
```

See Also:

- "Dropping a Disk from an Oracle ASM Disk Group" on page 3-7 for additional information about dropping a grid disk
- "ALTER GRIDDISK" on page 8-22 for additional information about `ALTER GRIDDISK`

DROP QUARANTINE

Purpose

The `DROP QUARANTINE` command manually drops a quarantine.

Syntax

```
DROP QUARANTINE { ALL | quarantine1 [, quarantine2]... }
```

Usage Notes

In general, a quarantine can be removed if the quarantined entity is not expected to cause more problem to CELLSRV. For example, cell offload for problem SQL statements is disabled, or an Oracle Database patch is applied. Refer to the alert message for the quarantine for more details.

When a cell is patched, all quarantines are automatically dropped. It is not necessary to drop them manually.

Examples

Example 8–63 shows the `DROP QUARANTINE` command.

Example 8–63 Dropping Quarantines

```
CellCLI> DROP QUARANTINE 1
```

DROP THRESHOLD

Purpose

The DROP THRESHOLD command removes all or the specified thresholds from the cell.

Syntax

```
DROP THRESHOLD { ALL |threshold_name [, threshold_name ...] }
```

Examples

Example 8–64 shows the DROP THRESHOLD command.

Example 8–64 Dropping Thresholds

```
CellCLI> DROP THRESHOLD ct_io_wt_rq.interactive

CellCLI> DROP THRESHOLD ALL
```

See Also: "DESCRIBE QUARANTINE" on page 8-79

EXPORT CELLDISK

Purpose

The `EXPORT CELLDISK` command prepares all cell disks or a specified cell disk before moving (importing) the cell disk to a different cell.

Syntax

```
EXPORT CELLDISK { ALL | cdisk_name }
```

Usage Notes

To move a cell disk from one cell to another, use the `EXPORT CELLDISK` and `IMPORT CELLDISK` commands. Usually, all disks are moved to a new cell if the current cell is failing. First, export the cell disk on one cell. Then, import the exported cell disk using the CellCLI utility on the cell where you moved the physical drive that contains the cell disk.

When the `EXPORT CELLDISK` command is run:

- `ALL` exports all cell disks on the cells that have `normal` status.
- If the LUN associated with the cell disk is flagged as automatically-created, then that LUN is deleted as part of the export.
- A successfully exported cell disk has the status attribute set to `ImportRequired`, and the exported cell disk is displayed in the output of the `LIST CELLDISK` command.
- The following apply when a cell disk is exported (`status='ImportRequired'`) before it is imported:
  - You can change the `name` and `comment` attributes.
  - You can drop the cell disk.
  - You cannot create a new grid disk on the cell disk.
- When a disk is exported, any writes from the disk controller cache to the disk are cleared, and the disk is flagged to indicate that the disk was exported. The grid disks on the disk are no longer visible to Oracle ASM. Any I/Os to the grid disks get errors.

Examples

Example 8–65 shows the `EXPORT CELLDISK` command.

Example 8–65 Exporting a Cell Disk

```
CellCLI> EXPORT CELLDISK CD_3_cell01

CellCLI> EXPORT CELLDISK ALL
```

IMPORT CELLDISK

Purpose

The IMPORT CELLDISK command reinstates all exported cell disks or an exported cell disk on a cell where you moved the physical drives that contain the cell disks.

The cell disk is typically imported to a different cell than the one from which the cell disk was exported. For example, the physical drive that contains the exported cell disk was moved to a different cell.

When you move a disk with cell disks and grid disks on it from one machine to another, be careful to ensure that the data on it is rebalanced, as per the ASM failure groups. If all disks from one cell are moved to another cell, then there is no need to perform a ASM rebalance, since the entire failure group is moved.

Syntax

```
IMPORT CELLDISK  { ALL  |  cdisk_name  LUN=lun_id  | cdisk_name |  LUN=lun_id }
    [, comment=comment_text] [FORCE]
```

Usage Notes

To move a cell disk from one cell to another, use the EXPORT CELLDISK and IMPORT CELLDISK commands. Usually, all disks are moved to a new cell if the current cell is failing. First, export the cell disk on one cell. Then, import the exported cell disk using the CellCLI utility on the cell where you moved the physical drive that contains the cell disk.

When the IMPORT CELLDISK command is run:

- Either ALL, the cell disk name, the LUN ID, or the cell disk name and LUN ID must be specified.
  - ALL imports cell disks that have ImportRequired status.
  - If the cell disk name is provided and the LUN ID is not provided, then you can import a cell disk by the specified name in cases where Management Server recognizes this cell disk. A recognized cell disk is displayed in the output of LIST CELLDISK with status equal to ImportRequired.
  - If the LUN ID is provided and the cell disk name is not provided, then the LUN is scanned, and the cell disk is imported. This variation of the command can be used to import a newly-inserted cell disk that was not recognized by Management Server and Cell Server.
  - If the LUN ID and cell disk name are both provided, then the LUN ID is used to import the cell disk, and the name is used to rename the imported cell disk.
- A new value can be entered for the comment attribute to update the existing cell disk comment.
- The cell disk name is verified to ensure that the name is unique within the cell. Cell disks can be renamed before import to ensure uniqueness.
- The grid disk names within a cell must be unique. If a physical disk is moved from one cell (cell_A) to another cell (cell_B) using the EXPORT and IMPORT commands, then there is a chance that the target cell (cell_B) could have two grid disks with identical names. In this case, the cell software automatically resolves the naming conflict by adding a temporary suffix (_duplicate_name, _

duplicate_name2, _duplicate_name3, and so on) to the name of one of the grid disks. This additional suffix enables you to refer to a grid disk unambiguously in the CellCLI commands.

It is recommended that you rename a duplicate grid disk on a cell (cell_B) with a new permanent unique name using the following command:

```
ALTER GRIDDISK gdname_duplicate_name NAME=new_unique_name
```

If you return the physical disk to the original cell (cell_A) or move the disk to another cell rather than renaming the disk, then the grid disk displays its original name.

- If the cell disk was not successfully exported and moved between cells, then the FORCE option must be specified with IMPORT or an error occurs.
- The import command checks the disk to determine if it was exported. If it was exported, then the IMPORT command makes the grid disk visible to Oracle ASM. If the disk was not exported, then the FORCE option should be used with import command to reconstruct the grid disks on the disk, and make them visible to Oracle ASM.

Examples

Example 8–66 shows the IMPORT CELLDISK command.

Example 8–66 Importing a Cell Disk

```
CellCLI> IMPORT CELLDISK CD_7_cell04 lun=3

CellCLI> IMPORT CELLDISK ALL
```

In Example 8–66, the LUN ID is provided with the IMPORT command to identify the cell disk, and the cell disk name is used to rename the cell disk on the cell where it was imported.

LIST

Purpose

The `LIST` command displays attributes for Exadata Cell objects. Objects displayed are identified by name or by filters. The attributes displayed for each object are determined by the specified attribute list.

Syntax

```
LIST object_type  [ name |  attribute_filters] [attribute_list] [DETAIL]
```

Usage Notes

- *`object_type`* can be the following:
  - `ACTIVEREQUEST`
  - `ALERTDEFINITION`
  - `ALERTHISTORY`
  - `CELL`
  - `CELLDISK`
  - `FLASHCACHE`
  - `FLASHCACHECONTENT`
  - `GRIDDISK`
  - `IBPORT`
  - `IORMPLAN`
  - `KEY`
  - `LUN`
  - `METRICCURRRENT`
  - `METRICDEFINITION`
  - `METRICHISTORY`
  - `PHYSICALDISK`
  - `QUARANTINE`
  - `THRESHOLD`
- Using `LIST` with only an *`object_type`* (without the `DETAIL` option or an attribute list) displays the names of the existing objects of this type and a default list of attributes.
  - For an object type that has a `status` attribute, the object name and the status are displayed.
  - For the `METRICHISTORY` object type, the collection time, the object name, and value are displayed.
  - For the `PHYSICALDISK` and `LUN` object types, the ID attribute is displayed.
  - For the `ALERTHISTORY` object type, the time and alert message are displayed.
  - For the `KEY` object type, the key value is displayed.

- The attributes displayed for each object are determined by the specified *attribute_list*. Attribute values that are strings with embedded blank spaces or tabs must be enclosed in quotation marks.
- Attribute filters determine the specific objects that are displayed. Because of the amount of metrics, you should use filters when using the `LIST METRICCURRENT` or `LIST METRICHISTORY` commands to narrow the output of the command.
- In the default format without the `DETAIL` option, each object is displayed on a separate line, with successive attribute values separated by tabs in the order of the specified list of attributes.
- In the `DETAIL` format, each attribute of a specific object is displayed on a separate line, with an attribute name followed by its value. If no attribute list is provided, then all attributes that have values are displayed. Blank lines separate each object in the display. `DETAIL` is similar to the `ATTRIBUTES ALL` option, only the format is different.
- Attributes that are not set are not listed with the `DETAIL` option. However, attributes that are set to an empty value are listed with the `DETAIL` option.

See Also:

- "LIST ACTIVEREQUEST" on page 8-97
- "LIST ALERTDEFINITION" on page 8-98
- "LIST ALERTHISTORY" on page 8-99
- "LIST CELL" on page 8-100
- "LIST CELLDISK" on page 8-102
- "LIST FLASHCACHE" on page 8-104
- "LIST FLASHCACHECONTENT" on page 8-105
- "LIST GRIDDISK" on page 8-107
- "LIST IBPORT" on page 8-110
- "LIST IORMPLAN" on page 8-112
- "LIST KEY" on page 8-113
- "LIST LUN" on page 8-114
- "LIST METRICCURRENT" on page 8-116
- "LIST METRICDEFINITION" on page 8-117
- "LIST METRICHISTORY" on page 8-118
- "LIST PHYSICALDISK" on page 8-120
- "LIST QUARANTINE" on page 8-122
- "LIST THRESHOLD" on page 8-123
- "About CellCLI Object Types" on page 8-7 for additional information about the object types
- "Attribute Lists in LIST Command" on page 8-9 for additional information about attribute lists
- "Attribute Filters in LIST Commands" on page 8-10 for additional information about attribute filters

LIST ACTIVEREQUEST

Purpose

The `LIST ACTIVEREQUEST` command displays specified attributes for the outstanding active requests for the cell.

Syntax

```
LIST ACTIVEREQUEST  [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–39, "Describing the ACTIVEREQUEST Object" on page 8-49.

Examples

Example 8–67 shows the `LIST` command with the `ACTIVEREQUEST` object.

Example 8–67 Listing ACTIVEREQUEST Attributes

```
CellCLI> LIST ACTIVEREQUEST 5 DETAIL

         name:                5
         ID:                  5
         ParentID:            5
         dbName:              "test DB"
         InstNum:             5
         ConsumerGrp:         "test group"
         SessID:              5
         SerialNum:           5
         AsmFileNum:          5
         AsmDGNum:            5
         FileIncNum:          5
         ObjNum:              5
         TsNum:               5
         SqlID:               5
         FileType:            "Oracle db data file"
         IoReason:            "test io"
         IoType:              "test read"
         State:               "Queued for Test"
         GdList:              gdName=testGrid,gdOffset=0,gdSize=524288000
```

LIST ALERTDEFINITION

Purpose

The `LIST ALERTDEFINITION` command displays all available sources of the alerts on the cell.

Syntax

```
LIST ALERTDEFINITION [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–40, "Describing the ALERTDEFINITION Object" on page 8-51.

Examples

Example 8–68 shows the `LIST` command with the `ALERTDEFINITION` object.

Example 8–68 Listing ALERTDEFINITION Attributes

```
CellCLI> LIST ALERTDEFINITION StatefulAlert_CG_IO_RQ_LG DETAIL

         name:                   StatefulAlert_CG_IO_RQ_LG
         alertShortName:         CG_IO_RQ_LG
         alertSource:            Metric
         alertType:              Stateful
         description:            "Threshold Alert"
         metricName:             CG_IO_RQ_LG
```

LIST ALERTHISTORY

Purpose

The `LIST ALERTHISTORY` command displays all alerts that occurred on the cell.

Syntax

```
LIST ALERTHISTORY [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–41, "Describing the ALERTHISTORY Object" on page 8-53.

A `WHERE` clause can include the `ageInMInutes` attribute to specify the list is limited to those alerts which have the specified age. For example, the following command would show the alerts created in the previous 15 minutes:

```
CellCLI> LIST ALERTHISTORY WHERE ageInMinutes < 15
```

Examples

Example 8–69 shows the `LIST` command with the `ALERTHISTORY` object.

Example 8–69 Listing ALERTHISTORY Attributes

```
CellCLI>  LIST ALERTHISTORY 1671443714 DETAIL

          name:                  1671443714
          alertSequenceID:       1671443714
          sequenceBeginTime:     1179185707672
          beginTime:             "Sat May 18 10:14:16 PDT 2009"
          endTime:               "Sat May 25 10:14:16 PDT 2009"
          severity:              critical
          alertMessage:          "Errors in file svtrc_2840_10.trc (incident=13):"
          alertShortName:        ADR
          alertNotified:         0
          examinedBy:            johndoe
          alertType:             stateless

CellCLI> LIST ALERTHISTORY WHERE begintime > 'Jun 1, 2009 11:37:00 AM PDT'

         39      2009-10-02T12:26:53-07:00       "ORA-07445: exception
                 encountered: core dump [__kerne l_vsyscall()+5] [6] [
                 0x408C] [] [] []"
         40      2009-10-06T23:28:06-07:00       "RS-7445 [unknown_function]
                 [signum: 6] [] [] [] [] [ ] []"
         41      2009-10-07T00:50:42-07:00       "RS-7445 [Serv MS not responding]
                 [It will be restart ed] [] [] [] [] [] []"
         42      2009-10-07T02:21:19-07:00       "RS-7445 [unknown_function]
                 [signum: 6] [] [] [] [] [ ] []"
```

LIST CELL

Purpose

The `LIST CELL` command displays specified attributes of the cell.

Syntax

```
LIST CELL [attribute_list] [DETAIL]
```

Usage Notes

- The list of attributes that can be displayed is shown in Example 8–42, "Describing the CELL Object" on page 8-56.
- `LIST CELL` only lists the local cell. Name and filter options on `LIST CELL` are not required.
- To monitor the status of cell components, use the `LIST` command to verify the value of `status`, `fanStatus`, `temperatureStatus`, and `powerStatus`.

Examples

Example 8–70 shows the `LIST` command with the `CELL` object, and the corresponding output.

Example 8–70 Listing Cell Attributes

```
CellCLI> LIST CELL ATTRIBUTES name, status, location

         cell01        online        rack5:shelf1
```

Example 8–71 shows how to display cell attributes.

Example 8–71 Verifying the Status of Cell Components

```
CellCLI> LIST CELL ATTRIBUTES name, status, location, -
         fanStatus, temperatureStatus, powerStatus

         stba2s3         online   dv-2ndFloor     normal   normal   normal
```

Example 8–72 shows how to display the values of the `snmpSubscriber` attribute.

Example 8–72 Displaying the snmpSubscriber Attribute

```
CellCLI> LIST CELL ATTRIBUTES snmpSubscriber

((host=server1.example.com,port=3873,community=public, type=asr))
```

Example 8–73 shows how to display the value of the `emailFormat` attribute.

Example 8–73 Displaying E-mail Format

```
CellCLI> LIST CELL ATTRIBUTES emailFormat
         html
```

Example 8–74 shows how to display the value of the `locateLEDStatus` attribute.

Example 8–74 Displaying locateLEDStatus

```
CellCLI> LIST CELL ATTRIBUTES locateLEDStatus
         off
```

Example 8–75 shows how to display the value of the `bbuLearnCycleTime` attribute.

Example 8–75 Using the bbuLearnCycleTime Attribute

```
CellCLI> LIST CELL ATTRIBUTES bbuLearnCycleTime
         2011-04-17T02:00:00-07:00
```

LIST CELLDISK

Purpose

The `LIST CELLDISK` command displays attributes for cell disks determined by the specified attributes and filters.

Syntax

```
LIST CELLDISK [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–43, "Describing the CELLDISK Object" on page 8-58.

Examples

Example 8–76 shows the `LIST` command with the `CELLDISK` object, and the corresponding output.

Example 8–76 Listing Cell Disk Attributes

```
CellCLI> LIST CELLDISK CD_01_cell05 ATTRIBUTES size

         136.640625G

CellCLI> LIST CELLDISK WHERE status!=normal ATTRIBUTES name

         CD_01_1_stsd2s3

CellCLI> LIST CELLDISK WHERE DEVICENAME LIKE '/dev/c0d[2-5]' -
         ATTRIBUTES name, size

         CD_01_cell05             139664M

CellCLI> LIST CELLDISK CD_01_cell05 DETAIL

         name:                   CD_01_cell05
         status:                 normal
         comment:
         id:                     00000117-84d2-ed2c-0000-000000000000
         size:                   139664M
         lun:                    3_1
         freespace:              129424M
         devicepartition:        /dev/c1d0p2
         Devicename:             /dev/c1d0
         diskType:               HardDisk

CellCLI> LIST CELLDISK CD_00_sgsata1 DETAIL

         name:                   CD_00_sgsata1
         comment:
         creationTime:           2009-07-26T15:31:02-07:00
         deviceName:             /dev/sda
         devicePartition:        /dev/sda3
         diskType:               HardDisk
         errorCount:             0
         freeSpace:              286.96875G
         freeSpaceMap:           offset=102.546875G,size=286.96875G
```

```
id:                    00000122-b930-3e2c-0000-000000000000
interleaving:          normal_redundancy
lun:                   0_0
raidLevel:             0
size:                  389.515625G
status:                normal
```

LIST FLASHCACHE

Purpose

The `LIST FLASHCACHE` command displays attributes for the Exadata Smart Flash Cache determined by the specified attributes.

Syntax

```
LIST FLASHCACHE [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–44, "Describing the FLASHCACHE Object" on page 8-60.

Examples

Example 8–77 shows the `LIST` command with the `FLASHCACHE` object, and the corresponding output.

Example 8–77 Listing Exadata Smart Flash Cache Attributes

```
CellCLI> LIST FLASHCACHE

         raw_FLASHCACHE normal

CellCLI> LIST FLASHCACHE DETAIL

         name:                    raw_FLASHCACHE
         cellDisk:                c9FLASH0,FD_FLASH1_raw,FD_FLASH2_raw
         creationTime:            2009-08-04T15:42:42-07:00
         id:                      8a0adc84-9088-4c4e-8e1c-b6bcbd5cb1ba
         size:                    80M
         status:                  normal
```

LIST FLASHCACHECONTENT

Purpose

The `LIST FLASHCACHECONTENT` command displays attributes for the Exadata Smart Flash Cache entries determined by the specified attributes.

Syntax

```
LIST FLASHCACHECONTENT [filters] [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–45, "Describing the FLASHCACHECONTENT Object" on page 8-61.

Examples

Example 8–78 shows the `LIST` command with the `FLASHCACHECONTENT` object, and the corresponding output.

Example 8–78 Listing Exadata Smart Flash Cache Content Attributes

```
CellCLI> LIST FLASHCACHECONTENT DETAIL

         cachedKeepSize:           8192
         cachedSize:              16384
         dbID:               3557170052
         dbUniqueName:            ACME1
         hitCount:                    4
         hoursToExpiration:          24
         missCount:
         objectNumber:              267
         tableSpaceNumber:            1

         cachedKeepSize:              0
         cachedSize:               8192
         dbID:               4325252357
         dbUniqueName:            MYODB
         hitCount:                    1
         hoursToExpiration:          23
         missCount:                   1
         objectNumber:            65523
         tableSpaceNumber:            4
```

Example 8–78 shows a database query for an object in a partitioned table, and then lists the flash cache for the same object. In the example, a partitioned table is created in the database, and then queried for the data object numbers of the partitions. The flash cache on Exadata Cell is then queried.

Example 8–79 Listing Exadata Smart Flash Cache Content by Database Object

```
CREATE TABLE parttabl (c1 number) PARTITION BY RANGE(c1)
(
  PARTITION partt1 VALUES LESS THAN (100),
  PARTITION partt2 VALUES LESS THAN (200)
);

SQL> SELECT SUBSTR(OBJECT_NAME, 0 , 10) OBJ_NAME, SUBOBJECT_NAME, DATA_OBJECT_ID
     FROM user_objects WHERE OBJECT_NAME LIKE ('PARTT%');
```

```
OBJ_NAME   SUBOBJECT_NAME                 DATA_OBJECT_ID
---------- ------------------------------ --------------
PARTTABL
PARTTABL   PARTT1                                  63197
PARTTABL   PARTT2                                  63198

CellCLI> LIST FLASHCACHECONTENT DETAIL WHERE objectNumber=63197 DETAIL
         cachedKeepSize:        0
         cachedSize:            24576
         dbID:                  3722937438
         dbUniqueName:          VIEW6
         hitCount:              0
         missCount:             4
         objectNumber:          63197
         tableSpaceNumber:      0


  CellCLI> LIST FLASHCACHECONTENT DETAIL WHERE objectNumber=63198 DETAIL
         cachedKeepSize:        0
         cachedSize:            16384
         dbID:                  3722937438
         dbUniqueName:          VIEW6
         hitCount:              0
         missCount:             2
         objectNumber:          63198
         tableSpaceNumber:      0
```

LIST GRIDDISK

Purpose

The `LIST GRIDDISK` command displays attributes for one or more Exadata Cell grid disks determined by the specified attributes and filters.

Syntax

```
LIST GRIDDISK [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–46, "Describing the GRIDDISK Object" on page 8-62.

The ASMDeactivationOutcome attribute can be used to determine if a grid disk can be be deactivated without loss of data. This attribute is not included in the list of attributes shown by the `DESCRIBE GRIDDISK` command. When using this attribute, a `YES` in the output means the grid disk can be deactivated.

The ASMModeStatus attribute can be used to determine the current usage of a grid disk. This attribute is not included in the list of attributes shown by the `DESCRIBE GRIDDISK` command. The possible values for this attribute are as follows:

- `ONLINE`: Oracle ASM is actively using this grid disk.
- `OFFLINE`: Oracle ASM has taken this grid disk offline.
- `UNUSED`: No Oracle ASM instance has used this grid disk on the storage cell.
- `SYNCING`: Oracle ASM has started to set this grid disk to online.
- `UNKNOWN`: Oracle ASM instances that use the grid disk are not available to query, or Oracle ASM has rejected the query because it is not in a currently-mounted disk group.

Examples

Example 8–80 shows the `LIST` command with the `GRIDDISK` object, and the corresponding output.

Example 8–80 Listing Grid Disk Attributes

```
CellCLI> LIST GRIDDISK WHERE cellDisk = 'CD_01_cell05' -
         ATTRIBUTES name, status

         DATA_CD_01_cell05        active
         RECO_CD_01_cell05        active

CellCLI> LIST GRIDDISK DATA_CD_01_cell05 DETAIL

         name:                   DATA_CD_01_cell05
         status:                 active
         comment:
         id:                     00000117-84d9-0096-0000-000000000000
         creationTime:           2009-01-16T17:04:49-06:00
         cellDisk:               CD_01_cell05
         offset:                 0
         availableTo:
         size:                   10G
         errorCount:             0
```

```
         diskType:               HardDisk

CellCLI> LIST GRIDDISK DATA_CD_01_cell05 ATTRIBUTES size

         136.640625G

CellCLI> LIST GRIDDISK WHERE status!=active ATTRIBUTES name

         data_CD_01_1_stsd2s3

CellCLI> LIST GRIDDISK data4_CD_09_sgsata1 DETAIL

         name:                   data4_CD_09_sgsata1
         availableTo:
         cellDisk:               CD_09_sgsata1
         comment:
         creationTime:           2009-07-26T17:09:46-07:00
         diskType:               HardDisk
         errorCount:             0
         id:                     00000122-b98a-a47a-0000-000000000000
         offset:                 27.546875G
         size:                   75G
         status:                 active
```

Example 8–81 shows the ASMDeactivationOutcome attribute being used to determine if a grid disk can be deactivated.

Example 8–81 Determining if a Grid Disk can be Deactivated

```
CellCLI> LIST GRIDDISK ATTRIBUTES name, ASMDeactivationOutcome

         QUAL_CD_00_sgsas1       Yes
         PROD_CD_02_sgsas1       Cannot de-activate due to other offline disks in
                                 the diskgroup
         TEST_CD_03_sgsas1       Yes
         DATA_CD_04_sgsas1       Yes
         DATA_CD_05_sgsas1       Yes
         DATA_CD_06_sgsas1       Yes
         RECO_CD_01_sgsas1       Cannot de-activate due to other offline disks in
                                 the diskgroup
         DATA_CD_08_sgsas1       Yes
         DATA_CD_09_sgsas1       Yes
         DATA_CD_10_sgsas1       Yes
         DATA_CD_11_sgsas1       Yes
```

Example 8–82 shows the ASMModeStatus attribute being used to check the current usage of a grid disk.

Example 8–82 Viewing the Current Usage of a Grid Disk

```
CellCLI> LIST GRIDDISK ATTRIBUTES name, ASMModeStatus

         QUAL_CD_00_sgsas1        UNUSED
         RECO_CD_01_sgsas1        OFFLINE
         PROD_CD_02_sgsas1        SYNCING
         TEST_CD_03_sgsas1        UNKNOWN
         DATA_CD_04_sgsas1        ONLINE
         DATA_CD_05_sgsas1        ONLINE
         DATA_CD_06_sgsas1        ONLINE
         DATA_CD_07_sgsas1        ONLINE
```

```
DATA_CD_08_sgsas1        ONLINE
DATA_CD_09_sgsas1        ONLINE
DATA_CD_10_sgsas1        ONLINE
DATA_CD_11_sgsas1        ONLINE
```

Example 8–83 shows the `LIST GRIDDISK` command being used to check the status of the resize process when the `NOWAIT` option is used with the `ALTER GRIDDISK` command.

Example 8–83 Checking the Status Using the LIST GRIDDISK Command

```
CellCLI> LIST GRIDDISK DETAIL

         name:                  gd0
         availableTo:
         cellDisk:              c9standby0
         comment:
         creationTime:          2009-07-09T09:07:36-07:00
         diskType:              HardDisk
         errorCount:            0
         id:                    00000122-6045-173b-0000-000000000000
         resizeStatus:          Resize in progress
         offset:                48M
         size:                  48M
         status:                active
```

Example 8–84 shows the `LIST GRIDDISK` command being used to check the status of grid disks that are being erased.

Example 8–84 Checking the Status of Secure Erase

```
CellCLI> LIST GRIDDISK
         DATA_CD_00_sgsas1     active
         DATA_CD_05_sgsas1     active
         DATA_CD_06_sgsas1     erase in progress
         DATA_CD_07_sgsas1     erase in progress
```

LIST IBPORT

Purpose

The `LIST IBPORT` command displays attributes for InfiniBand ports determined by the specified attributes and filters.

Syntax

```
LIST IBPORT  [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–47, "Describing the IBPORT Object" on page 8-66.

Examples

Example 8–85 shows the `LIST` command with the `IBPORT` object, and the corresponding output.

Example 8–85 Listing IBPORT Attributes

```
CellCLI> LIST IBPORT
         HCA-1:1         Active
         HCA-1:2         Active

CellCLI> LIST IBPORT DETAIL
         name:                   HCA-1:1
         activeSlave:            TRUE
         dataRate:               "40 Gbps"
         hcaFWVersion:           2.7.0
         id:                     0x00212800013e8c67
         lid:                    20
         linkDowned:             0
         linkIntegrityErrs:      0
         linkRecovers:           0
         physLinkState:          LinkUp
         portNumber:             1
         rcvConstraintErrs:      0
         rcvData:                84653709
         rcvErrs:                0
         rcvRemotePhysErrs:      0
         status:                 Active
         symbolErrs:             0
         vl15Dropped:            0
         xmtConstraintErrs:      0
         xmtData:                84572496
         xmtDiscards:            0

         name:                   HCA-1:2
         activeSlave:            FALSE
         dataRate:               "40 Gbps"
         hcaFWVersion:           2.7.0
         id:                     0x00212800013e8c68
         lid:                    21
         linkDowned:             0
         linkIntegrityErrs:      0
         linkRecovers:           0
         physLinkState:          LinkUp
```

```
portNumber:             2
rcvConstraintErrs:      0
rcvData:                79355427
rcvErrs:                0
rcvRemotePhysErrs:      0
status:                 Active
symbolErrs:             0
vl15Dropped:            0
xmtConstraintErrs:      0
xmtData:                79274016
xmtDiscards:            0
```

LIST IORMPLAN

Purpose

The `LIST IORMPLAN` command lists the current plan of the local cell.

Syntax

```
LIST IORMPLAN [attribute_list]  [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–48, "Describing the IORMPLAN Object" on page 8-66.

Examples

Example 8–86 shows the `LIST` command with the `IORMPLAN` object, and the corresponding output.

Example 8–86 Listing IORMPLAN Attributes

```
CellCLI> LIST IORMPLAN ATTRIBUTES status

         active

CellCLI> LIST IORMPLAN DETAIL

         name:                 cell01_IORMPLAN
         catPlan:              name=administrative,level=1,allocation=80
                               name=interactive,level=2,allocation=90
                               name=batch,level=3,allocation=80
                               name=maintenance,level=4,allocation=50
                               name=other,level=4,allocation=50
         dbPlan:               name=sales_prod,level=1,allocation=80
                               name=finance_prod,level=1,allocation=20
                               name=sales_dev,level=2,allocation=100
                               name=sales_test,level=3,allocation=50
                               name=other,level=3,allocation=50
         objective:            balanced
         status:               active
```

LIST KEY

Purpose

The `LIST KEY` command displays key values for clients determined by the specified attributes and filters.

Syntax

```
LIST KEY [DETAIL]
```

Usage Notes

The key value assigned to a client must match the keys in the `cellkey.ora` files on cells, and the Oracle ASM and database host computers.

Examples

Example 8–87 shows the `LIST` command with the `KEY` object, and the corresponding output.

Example 8–87 Listing KEY Attributes

```
CellCLI>  LIST KEY db1 DETAIL

         name:                   db1
         key:                    b67d5587fe728118af47c57ab8da650a

CellCLI>  LIST KEY

         db1      b67d5587fe728118af47c57ab8da650a
         db456    118af47c57ab8da650ab67d5587fe728
```

See Also: Chapter 4, "Configuring Security for Oracle Exadata Storage Server Software" for additional information about Exadata Cell security

LIST LUN

Purpose

The `LIST LUN` command displays attributes for LUNs determined by the specified attributes and filters.

Syntax

```
LIST LUN [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–50, "Describing the LUN Object" on page 8-69.

On HP Oracle Exadata Storage Server there are 12 hard disks, and on Oracle Exadata Storage Server there are 12 hard disks and 16 flash disks. The flash disks also appear in the listing.

Examples

Example 8–88 shows the `LIST` command with the `LUN` object, and the corresponding output.

Example 8–88 Listing LUN Attributes

```
CellCLI> LIST LUN
         0_0      0_0      normal
         0_1      0_1      normal
         0_2      0_2      normal
         0_3      0_3      normal
         0_4      0_4      normal
         0_5      0_5      normal
         0_6      0_6      normal
         0_7      0_7      normal
         0_8      0_8      normal
         0_9      0_9      normal
         0_10     0_10     normal
         0_11     0_11     normal
         1_0      1_0      normal
         1_1      1_1      normal
         1_2      1_2      normal
         1_3      1_3      normal
         2_0      2_0      normal
         2_1      2_1      normal
         2_2      2_2      normal
         2_3      2_3      normal
         4_0      4_0      normal
         4_1      4_1      normal
         4_2      4_2      normal
         4_3      4_3      normal
         5_0      5_0      normal
         5_1      5_1      normal
         5_2      5_2      normal
         5_3      5_3      normal

CellCLI> LIST LUN 0_0 DETAIL
         name:                   0_0
         cellDisk:               CD_00_sgsas1
```

```
        deviceName:             /dev/sda
        diskType:               HardDisk
        id:                     0_0
        isSystemLun:            TRUE
        lunAutoCreate:          TRUE
        lunSize:                558.40625G
        lunUID:                 0_0
        physicalDrives:         20:0
        raidLevel:              0
        status:                 normal

CellCLI> LIST LUN 1_0 DETAIL
        name:                   1_0
        cellDisk:               FD_00_sgsas1
        deviceName:             /dev/sdr
        diskType:               FlashDisk
        id:                     1_0
        isSystemLun:            FALSE
        lunAutoCreate:          FALSE
        lunSize:                22.8880615234375G
        overProvisioning:       100.0
        physicalDrives:         [9:0:0:0]
        status:                 normal
```

LIST METRICCURRENT

Purpose

The `LIST METRICCURRENT` command displays a list of collections of all metrics.

Syntax

```
LIST METRICCURRENT [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

- The list of attributes that can be displayed is shown in Example 8–51, "Describing the METRICCURRENT Object" on page 8-70.
- To reduce the size of the output when you run the `LIST METRICCURRENT` command, use filters.

Examples

Example 8–89 shows the `LIST` command with filters to display information about the `METRICCURRRENT` object, and the corresponding output.

Example 8–89 Listing METRICCURRENT Attributes

```
CellCLI> LIST METRICCURRENT WHERE objectType = 'CELLDISK'

         CD_IO_TM_W_SM_RQ        c9controlfile0  205.5 us/request
         CD_IO_TM_W_SM_RQ        c9datafile0     93.3 us/request
         CD_IO_TM_W_SM_RQ        c9datafile1     0.0 us/request
         CD_IO_TM_W_SM_RQ        c9datafile2     110.5 us/request
         CD_IO_TM_W_SM_RQ        c9datafile3     0.0 us/request
         CD_IO_TM_W_SM_RQ        c9datafile4     541.5 us/request
         CD_IO_TM_W_SM_RQ        c9logfile0      181.2 us/request
         CD_IO_TM_W_SM_RQ        c9logfile1      0.0 us/request
         CD_IO_TM_W_SM_RQ        c9standby0      130.4 us/request

CellCLI> LIST METRICCURRENT WHERE name = CD_IO_TM_W_SM_RQ   -
         AND metricObjectName = c9datafile4 DETAIL

         name:                   CD_IO_TM_W_SM_RQ
         alertState:             normal
         collectionTime:         2009-07-01T15:19:25-07:00
         metricObjectName:       c9datafile4
         metricType:             Rate
         metricValue:            0.0 us/request
         objectType:             CELLDISK
```

See Also: "Attribute Filters in LIST Commands" on page 8-10 for additional information about attribute filters

LIST METRICDEFINITION

Purpose

The `LIST METRICDEFINITION` command displays a list of metric definitions on the cell.

Syntax

```
LIST METRICDEFINITION [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–52, "Describing the METRICDEFINITION Object" on page 8-72.

Examples

Example 8–90 shows the `LIST` command with the `METRICDEFINITION` object, and the corresponding output.

Example 8–90 Listing METRICDEFINITION Attributes

```
CellCLI> LIST METRICDEFINITION WHERE objectType ='CELL' DETAIL

         name:                    CL_CPUT
         description:             "Cell CPU Utilization is the percentage of time
                                  over the previous minute that the system
                                  CPUs were not idle (from /proc/stat)."
         metricType:              Instantaneous
         objectType:              CELL
         unit:                    %
...
```

LIST METRICHISTORY

Purpose

The `LIST METRICHISTORY` command displays a list of individual metrics.

Syntax

```
LIST METRICHISTORY [ name | attribute_filters ] [attribute_list] {over_
specification] [MEMORY] [DETAIL]
```

Usage Notes

- The list of attributes that can be displayed is shown in Example 8–53, "Describing the METRICHISTORY Object" on page 8-74.
- The retention period for metric history entry files is specified by the `metricHistoryDays` cell attribute. You can modify this setting with the CellCLI `ALTER CELL` command.
- The *over_specification* syntax is as follows:

  ```
  OVER number [aggregation_type [aggregation_type]...]
  ```

  In the preceding syntax, *number* is amount of time in minutes for the aggregation, and *aggregation_type* can be `max`, `min`, or `avg`.
- A `WHERE` clause can include the `ageInMinutes` attribute to specify the list is limited to those metrics which have the specified age. For example, the following command would show the metrics created in the previous 15 minutes:

  ```
  CellCLI> LIST METRICHISTORY WHERE ageInMinutes < 15
  ```

Examples

Example 8–91 shows the `LIST` command with the `METRICHISTORY` object, and the corresponding output. To reduce the size of the output when you run the `LIST METRICHISTORY` command, use filters.

Example 8–91 Listing METRICHISTORY Attributes

```
CellCLI> LIST METRICHISTORY WHERE name like 'CL_.*'           -
            AND collectionTime > '2009-07-01T15:28:36-07:00'

         CL_RUNQ         stbcr03_2       6.0      2009-07-01T15:28:37-07:00
         CL_CPUT         stbcr03_2       47.6 %   2009-07-01T15:29:36-07:00
         CL_FANS         stbcr03_2       1        2009-07-01T15:29:36-07:00
         CL_TEMP         stbcr03_2       0.0 C    2009-07-01T15:29:36-07:00
         CL_RUNQ         stbcr03_2       5.2      2009-07-01T15:29:37-07:00
```

Example 8–92 shows the `LIST METRICHISTORY` command with the `over` and `memory` attributes.

Example 8–92 Listing METRICHISTORY Using the OVER and MEMORY Attributes

```
CellCLI> LIST METRICHISTORY cl_cput OVER 10 MIN MAX MEMORY
        CL_CPUT         firstcell       55.0 %   2009-11-15T06:00:17-08:00      55.0 %   57.1 %
        CL_CPUT         firstcell       54.7 %   2009-11-15T06:10:17-08:00      54.7 %   56.3 %
        CL_CPUT         firstcell       54.8 %   2009-11-15T06:20:18-08:00      54.7 %   57.2 %
        CL_CPUT         firstcell       55.0 %   2009-11-15T06:30:18-08:00      54.3 %   55.9 %
```

```
CL_CPUT         firstcell       55.0 %  2009-11-15T06:40:18-08:00      54.9 %   57.0 %
CL_CPUT         firstcell       55.1 %  2009-11-15T06:50:18-08:00      54.8 %   56.4 %
CL_CPUT         firstcell       58.0 %  2009-11-15T07:00:18-08:00      55.2 %   58.0 %
CL_CPUT         firstcell       55.5 %  2009-11-15T07:10:18-08:00      55.5 %   67.5 %
```

See Also:

- "ALTER CELL" on page 8-15 for additional information about altering the metric history
- "DESCRIBE CELL" on page 8-54 for a description of the `metricHistoryDays` cell attribute
- "Attribute Filters in LIST Commands" on page 8-10 for additional information about attribute filters

LIST PHYSICALDISK

Purpose

The `LIST PHYSICALDISK` command displays attributes for one or more physical disks determined by the specified attributes and filters.

Syntax

```
LIST PHYSICALDISK [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–54, "Describing the PHYSICALDISK Object on Oracle Exadata Storage Server" and Example 8–55, "Describing the PHYSICALDISK Object on HP Oracle Exadata Storage Server" on page 8-78.

Examples

Example 8–93 shows the `LIST` command with the `PHYSICALDISK` object, and the corresponding output.

Example 8–93 Listing Physical Disk Attributes

```
CellCLI> LIST PHYSICALDISK

         20:0            E01E4X                  normal
         20:1            E01E92                  normal
         20:2            E01F94                  normal
         20:3            E01CS4                  normal
         20:4            E01CYN                  normal
         20:5            E01CSN                  normal
         20:6            E01DST                  normal
         20:7            E01EFF                  normal
         20:8            E01EAP                  normal
         20:9            E01CPY                  normal
         20:10           E01E1T                  normal
         20:11           E01CS2                  normal
         [10:0:0:0]      50800200009189aFMOD0    normal
         [10:0:1:0]      50800200009189aFMOD1    normal
         [10:0:2:0]      50800200009189aFMOD2    normal
         [10:0:3:0]      50800200009189aFMOD3    normal
         [11:0:0:0]      508002000093210FMOD0    normal
         [11:0:1:0]      508002000093210FMOD1    normal
         [11:0:2:0]      508002000093210FMOD2    normal
         [11:0:3:0]      508002000093210FMOD3    normal
         [8:0:0:0]       508002000092e9aFMOD0    normal
         [8:0:1:0]       508002000092e9aFMOD1    normal
         [8:0:2:0]       508002000092e9aFMOD2    normal
         [8:0:3:0]       508002000092e9aFMOD3    normal
         [9:0:0:0]       508002000092f28FMOD0    normal
         [9:0:1:0]       508002000092f28FMOD1    normal
         [9:0:2:0]       508002000092f28FMOD2    normal
         [9:0:3:0]       508002000092f28FMOD3    normal

CellCLI> LIST PHYSICALDISK 20:0 DETAIL

         name:                   20:0
         deviceId:               32
```

```
        diskType:               HardDisk
        enclosureDeviceId:      20
        errMediaCount:          0
        errOtherCount:          0
        foreignState:           false
        id:                     E01E4X
        luns:                   0_0
        makeModel:              "SEAGATE ST360057SSUN600G"
        physicalFirmware:       0605
        physicalInsertTime:     2009-10-22T18:40:26-07:00
        physicalInterface:      sas
        physicalSerial:         E01E4X
        physicalSize:           558.9109999993816G
        slotNumber:             0
        status:                 normal

CellCLI> LIST PHYSICALDISK '[10:0:0:0]' DETAIL

        name:                   [10:0:0:0]
        diskType:               FlashDisk
        id:                     50800200009189aFMOD0
        luns:                   5_0
        makeModel:              "MARVELL SD88SA02"
        physicalFirmware:       D20R
        physicalInsertTime:     2009-10-21T18:35:26-07:00
        physicalInterface:      sas
        physicalSerial:         50800200009189aFMOD0
        physicalSize:           22.8880615234375G
        slotNumber:             "PCI Slot: 5; FDOM: 0"
        status:                 normal
```

LIST QUARANTINE

Purpose

The `LIST QUARANTINE` command displays specified attributes for quarantines.

Syntax

```
LIST QUARANTINE [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Examples

Example 8–94 shows the `LIST` command with the `QUARANTINE` object.

Example 8–94 Listing QUARANTINE Attributes

```
CellCLI> LIST QUARANTINE  DETAIL

CellCLI> LIST QUARANTINE where comment like 'added.*'
```

See Also: "DESCRIBE QUARANTINE" on page 8-79

LIST THRESHOLD

Purpose

The `LIST THRESHOLD` command displays attributes for one or more thresholds determined by the specified attributes and filters.

Syntax

```
LIST THRESHOLD [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

The list of attributes that can be displayed is shown in Example 8–57, "Describing the THRESHOLD Object" on page 8-81.

Examples

Example 8–95 shows the `LIST` command with the `THRESHOLD` object, and the corresponding output.

Example 8–95 Listing Threshold Attributes

```
CellCLI> LIST THRESHOLD

         ct_io_wt_rq.interactive
         db_io_rq_sm_sec.db123
         ....

CellCLI> LIST THRESHOLD ct_io_wt_rq.interactive DETAIL

         comparison:             =
         critical:               20.0
         name:                   ct_io_wt_rq.interactive
         observation:            5
         occurences:             2
         warning:                10.0

CellCLI> LIST THRESHOLD db_io_rq_sm_sec.db123 DETAIL

         comparison:             >
         name:                   db_io_rq_sm_sec.db123
         critical:               120.0
```

9

Using the dcli Utility

The dcli utility facilitates centralized management across an Oracle Exadata Storage Server Software (Exadata Cell) realm by automating the execution of CellCLI commands on a set of cells and returning the output to the centralized management location where the dcli utility was run. This chapter contains the following topics:

Overview of the dcli Utility

The dcli utility runs commands on multiple cells in parallel threads. However, it does not support an interactive session with a remote application on a cell.

To use the dcli utility, copy the utility from the `bin` directory on a cell to a host computer from which central management can be performed. You can issue a command to be run on multiple cells, or use files that can be copied to cells and then run. The cells are referenced by their domain name or IP address.

The dcli utility requires Python version 2.3 or later. You can determine the version of Python by running the `python -V` command. In addition, use of this tool assumes prior setup of SSH user-equivalence to a cell. You can use the dcli utility initially with the `-k` option to set up SSH user-equivalence to a cell. Also you can manually set up SSH user-equivalence to cells following the instructions in "Setting Up SSH User-Equivalence on Exadata Cell" on page 9-6.

Command output (`stdout` and `stderr`) is collected and displayed after the copy and command execution is finished on the specified cells. The dcli options allow command output to be abbreviated to minimize nonerror output, such as messages showing normal status.

This sections contains the following topics:

dcli Syntax

The dcli utility syntax is:

```
dcli [options] [command]
```

In the preceding command, the following arguments are used:

- *options*: Table 9–1 lists the available options and their descriptions.

Table 9–1 dcli Options

Option	Description
`--version`	Shows the version number of the program and then exits.
`-c` *cells*	Specifies a comma-delimited list of target cells to which commands are sent.
`-d` *destfile*	Specifies the target destination directory or file on remote cells to be used when copying files or directories using the `-f` option.
`-f` *file*	Specifies the files to be copied to the cells. These files are not run. These files can be script files to be run later. The files are copied to the default home directory of the user on the target cell.
`-g` *groupFile*	Specifies a file containing a list of target cells to which commands are sent. The cells can be identified by cell names or IP addresses.
`-h`, `--help`	Displays help text and then exits.
`-k`	Sets up SSH user-equivalence for the current user to the cells specified with the `-c` or `-g` option by appending public key files to the `authorized_keys` file on cells.
`-l` *userId*	Identifies the user to log in as on remote cells. The default is the `celladmin` user.
`-n`	Abbreviates nonerror output. Cells that return normal output (return code of `0`) only have the cell name listed. The `-n` and `-r` options cannot be used together.
`-r` *regexp*	Abbreviates the output lines that match a regular expression. All output lines with that pattern are deleted from output, and the cells names from those output lines are listed on one line. The `-r` and `-n` options cannot be used together.
`-s` *sshOptions*	Passes a string of options to SSH.
`--scp=` *scpOptions*	Passes a string of options to `scp` if different from *sshoptions*.
`--serial`	Serializes the process over Exadata Cells.
`-t`	Displays the target cells that are named with the `-c` option or in the *groupfile* identified by the `-g` option.
`--unkey`	Drops keys from the target `authorized_keys` file on Exadata Cells.
`-v`	Prints the verbose version of messages to `stdout`.
`--vmstat=VMSTATOPS`	Displays view process, virtual memory, disk, trap, and CPU activity information, depending on the switches.
`-x` *execFile*	Specifies the command file to be copied and run on the cells. The specified file contains a list of commands. A file with the `.scl` extension is run by the CellCLI utility. A file with a different extension is run by the operating system shell on the cell. The file is copied to the default home directory of the user on the target cell.

- *command:* Any command that can be run from an operating system prompt. For commands that contain punctuation that would be interpreted by the local shell, enclose the command in double quotation marks. If the command includes the following characters, then outer quotation marks and escape characters are required:

- $ (dollar sign)
- ' (quotation mark)
- < (less than)
- > (greater than)
- ( ) (parentheses)

The backslash (\) is the escape character that allows the characters to be passed to the CellCLI utility without being interpreted by the remote shell.

If the command is complex in terms of punctuation that need escape characters, then it may require that the command be put in a script, and run using the `-x` option. Within a script, the escape character is not required.

If the local dcli process is terminated, then remote commands might continue, but their output and status is unknown.

Return values from the dcli utility are:

- 0: The file or command was copied, and run successfully on all cells.
- 1: One or more cells could not be reached or remote execution returned a nonzero status.
- 2: A local error prevented any command execution.

If any cells are down or do not respond, then a message is written to `stderr` listing the unresponsive cells. The operations continue on the other cells, and the return code after completion is 1.

dcli Examples

This section contains examples of the dcli utility.

Example 9–1 shows how to set SSH user-equivalence for a current user using the `-k` option.

Example 9–1 Setting up SSH User-equivalence for a Current User

```
$ ./dcli -k -g mycells
```

The `-k` option assumes the user has accepted the default key file names for the SSH protocol, version 2. These file names are `id_dsa.pub` or `id_rsa.pub`, and are located in the `~/.ssh` directory.

You may be prompted to acknowledge cell authenticity, and may be prompted for the remote user password. The `-k` key exchange is done serially over the cells to prevent the user from getting password prompts from all cells simultaneously. After the `-k` option is used once, subsequent commands to the same cells do not require the `-k` option and do not require passwords for that user from the host.

Example 9–2 shows how to run the CellCLI command `ALTER IORMPLAN INACTIVE`, and abbreviates nonerror output.

Example 9–2 Using the -n Option

```
$ ./dcli -g mycells -l celladmin -n "cellcli -e alter iormplan inactive"
```

The abbreviated output would be similar to the following:

```
OK: ['stsd2s3']
stsd2s2:
```

```
stsd2s2: CELL-02619: Current IORMPLAN state is not 'active'.
```

Example 9–3 shows how to run the CellCLI command `LIST GRIDDISK`, and deletes the lines in the output that contain `normal`. The command is run on the target cells listed in the `mycells` group file.

Example 9–3 Using the -r Option

```
$ ./dcli -l celladmin -r '.*normal' -g mycells "cellcli -e list celldisk"
```

The output would be similar to the following:

```
.*normal: ['stsd2s2', 'stsd2s3']
stsd2s3: CD_06_stsd2s3    importRequired
```

Example 9–4 shows how to use the verbose (`-v`) option with SSH.

Example 9–4 Using the -v Option

```
$ ./dcli -s "-v" -c mycell date
```

Example 9–5 shows how to use the `-t` option to list target cells. This option should be used with `-c` or `-g` option.

Example 9–5 Using the -t Option

```
$ ./dcli -t -c stsd2s1 date
Target cells: ['stsd2s1']
stsd2s1: Fri Jul 17 15:37:31 PDT 2009
```

Example 9–6 shows how to use the `--vmstat` option.

Example 9–6 Using the --vmstat Option

```
$ ./dcli -g 123 -l sage --vmstat="-a 3 5"

procs -----------memory---------- ---swap-- -----io----  --system-- ...
13:43:03: r   b    swpd    free   inact active   si    so     bi     bo     in     cs  us
stsd2s1: 2  0       0  22656 178512 792272    0     0      1     21      7      2   2
stsd2s2: 0  0 452304   21432 108760 867712    0     0    178    269      2      0   2
stsd2s3: 0  0   49252 912164   70156  49996    1     1     74    249      2      2   1
Minimum: 0  0       0  21432   70156  49996    0     0      1     21      2      0   1
Maximum: 2  0 452304 912164 178512 867712    1     1    178    269      7      2   2
Average: 0  0 167185 318750 119142 569993    0     0     84    179      3      1   1
```

Example 9–7 shows a CellCLI command that changes the IORMPLAN to active on the target cells in the mycells group file as the `celladmin` user. The `-t` option displays the cells that are in the mycells groupfile.

Example 9–7 Using dcli to Change an IORM Plan

```
$ ./dcli -g mycells -l root -t "cellcli -e alter iormplan active"
```

Example 9–8 shows a CellCLI commands in the `reConfig.scl` file on the target cells in the mycells group file as the default `celladmin` user.

Example 9–8 Using dcli with a Script

```
$ ./dcli -g mycells -x reConfig.scl
```

Example 9–9 shows a CellCLI command that lists the name and status of grid disks on the target cells in the mycells group file as the default `celladmin` user. Output lines that contain `active` as the status are deleted.

Example 9–9 Using dcli to List Grid Disk Status

```
$ ./dcli -r '.*active' -g mycells "cellcli -e list griddisk"
```

Example 9–10 shows a CellCLI command that lists alert history name, examined by, severity on the target cells in the mycells group file as the default `celladmin` user. Output lines that contain `clear` for severity are deleted.

Example 9–10 Using dcli to List Alert History Information

```
$ ./dcli -r '.*clear' -g mycells \
   "cellcli -e list alerthistory attributes name, examinedby, severity"
```

Example 9–11 shows a CellCLI command that lists alert history where examined by has not been set on the target cells in the mycells group file as the `celladmin` user.

Example 9–11 Using dcli to List Alert History where examinedby is not Set

```
$ ./dcli -g allcells -l celladmin \
   "cellcli -e list alerthistory where examinedby=\'\' "
```

Example 9–12 shows a CellCLI command that lists the current metric alert state and metric value for metric GD_IO_BY_R_LG on the target cells in the mycells group file as the `celladmin` user. This query retrieves metric current objects for the number of MB read in large blocks on a grid disk.

Example 9–12 Using dcli to List Current Metric Alert State

```
$ ./dcli -g mycells "cellcli -e list metriccurrent GD_IO_BY_R_LG \
  attributes alertstate, metricvalue"
```

Example 9–13 shows a CellCLI command that lists metric current objects for names that begin with GD_IO_RQ on the target cells in the mycells group file as the `celladmin` user. This query retrieves metric current objects for the number of requests to read or write blocks on a grid disk.

Example 9–13 Using dcli to List Specific Metric Current Objects in a Group

```
$ ./dcli -g mycells "cellcli -e list metriccurrent where name like \'GD_IO_RQ.*\'"
```

Example 9–14 shows a CellCLI command that lists metric current objects with name equal to `cl_put` (cell CPU utilization) on the target cells in the mycells group file as the default `celladmin` user.

Example 9–14 Using dcli to List Specific Metric Current Objects

```
$ ./dcli -g mycells "cellcli -e list metriccurrent cl_cput"
```

Example 9–15 shows a CellCLI command that lists physical disks where status is not equal to normal on the target cells in the mycells group file as the default `celladmin` user.

Example 9–15 Using dcli to List Physical Disks

```
$ ./dcli -g allcells "cellcli -e list physicaldisk where status not = normal"
```

Example 9–16 shows a CellCLI command that lists cell disks where free space is less than 100 MB on the target cells in the mycells group file as the default `celladmin` user.

Example 9–16 Using dcli to List Cells with Free Space

```
$ ./dcli -g allcells "cellcli -e list celldisk where freespace \< 100M"
```

Note: In the preceding example, the backslash (\) is an escape character that allows the less than character (<) to be passed to the CellCLI utility without being interpreted by the remote shell.

Example 9–17 shows a CellCLI command to view the alert history from a particular period.

Example 9–17 Using dcli to View Alert History

```
dcli -g lab.cells "cellcli -e  list alerthistory where begintime \> \'Aug 4, 2009
12:06:38 PM\'"
```

Note: In the preceding example, the backslash (\) is an escape character that allows the greater than character (>) and the quotation marks to be passed to the CellCLI utility without being interpreted by the remote shell.

Setting Up SSH User-Equivalence on Exadata Cell

To set up SSH user-equivalence for use with the dcli utility, use the `-k` option. Setting user-equivalence enables you to issue commands to remote cells without having to enter the password for the cell.

> **See Also:** *Oracle Clusterware Installation Guide for Linux* for additional information about setting up SSH and SSH user-equivalence between the managing host and cells, see "Configuring SSH on All Cluster Nodes"

A

Upgrading Oracle Exadata Storage Server Software

To upgrade from Oracle Exadata Storage Server Software 11*g* Release 1 (11.1.3) to Oracle Exadata Storage Server Software 11*g* Release 2 (11.2), refer to My Oracle Support note 888834.1, and the documentation for the patch 8919508. My Oracle Support is available at

`http://support.oracle.com/`

B

Alert and Error Messages

This appendix lists the alert and error messages that can be encountered when using Oracle Exadata Storage Server Software (Exadata Cell). This appendix contains the following topics:

- Understanding Alert, Incident, and Trace Files
- Contacting Oracle Support Services
- Exadata Cell Error Messages
- Alert Messages

> **See Also:**
>
> - *Oracle Database Error Messages* for additional information about Oracle Database error messages
> - *Oracle Database Administrator's Guide* for additional information about collecting and managing diagnostic data
> - *Oracle Exadata Database Machine Owner's Guide* for additional information about Oracle Exadata Database Machine hardware alerts

Understanding Alert, Incident, and Trace Files

Alert, incident, and trace files provide a diagnostic record of useful information. Alert files contain information about internal errors and administrative tasks. Incident files contain information about single occurrences. Trace files can contain information about server and background processes. This section contains the following topics:

- About Alert Files
- About Incident Packages
- About Trace Files
- Diagnostic File Locations
- Managing Diagnostic Files

> **See Also:** "Understanding Automated Cell Maintenance" on page 2-16 for information about the file deletion policy

About Alert Files

An alert file is a log file that records information about internal errors and administrative activities, such as backups. When an internal error occurs, a message is sent to the terminal screen and is written to the alert file. Additional information about internal errors is also written to the alert file, such as the location and name of any trace files generated because of the error. Alert files are located in the following directory:

```
/opt/oracle/cell/log/diag/asm/cell/hostname/trace/alert.log
```

If your system uses an operator console, then some messages from Oracle may appear on the console. All important messages are written to the alert file and the operator console. Because all messages, not just Oracle messages, appear on this console, the alert file is a better record for tracing all Oracle administrative activity and errors than the console log.

About Incident Packages

An incident is a single occurrence of a problem. When a problem occurs multiple times, an incident is created for each occurrence. Incidents are tracked in the Automatic Diagnostic Repository (ADR). Each incident is identified by a numeric incident identifier, which is unique within ADR. When an incident occurs, the database makes an entry in the alert log, sends an incident alert to Oracle Enterprise Manager, gathers diagnostic data about the incident in the dump files (incident dumps), tags the incident dumps with the incident ID, and stores the incident dumps in the ADR subdirectory created for that incident.

Diagnosis and resolution of a critical error usually starts with an incident alert. You can obtain a list of all incidents in ADR using an ADR Command Interpreter (ADRCI) command.

Each incident is mapped to a single problem only. Incidents are compared so that a single problem does not generate too many incidents and incident dumps.

See Also:

Oracle Database Administrator's Guide for additional information about the following:

- Incident flood control
- Creating, editing, and uploading custom incident packages

About Trace Files

A trace file is created each time an Oracle instance starts or an unexpected event occurs in a user process or background process. The file extension or file type is usually `.trc`. If it is different, then it is noted in your operating system-specific Oracle documentation. The contents of the trace file may include dumps of the system global area, process global area, operating call stack, and registers.

Note: If you change a traceLevel attribute setting, then you need to restart Management Server for the change to take effect. Restarting Management Server does not affect the database or the flow of data.

About Automatic Diagnostic Repository

ADR is a core component of the Oracle diagnosability framework for Oracle Database. It is a repository of diagnostic data associated with major Oracle software components, such as the database, application server, or Exadata Cell. The repository stores problem data, incident data, package metadata, trace files, and dump files outside of the database.

Automatic Diagnostic Repository Incident

ADR incidents are generated and stored in ADR by the Oracle diagnosability framework when Oracle encounters critical software errors. Incidents may have associated trace files and dump files that can be packaged together to send to Oracle Support Services.

Automatic Diagnostic Repository Package

An ADR package is a collection of ADR incidents, and their related trace and dump files organized into a single (compressed) file for transfer to Oracle Support Services.

Diagnostic File Locations

Alert, incident, and trace files are written to the `alert`, `incident`, and `trace` subdirectories in the ADR home directory (`$ADR_BASE/diag/asm/cell/`*`cell_name`*) on the cell. The ADR home is located within the ADR base directory (`$ADR_BASE`). The retention period for ADR files is specified by the `metricHistoryDays` cell attribute. You can modify this setting with the CellCLI `ALTER CELL` command.

If you use Secure Shell (SSH) to access the cell, then you can display the value of `$ADR_BASE` that was set during installation.

See Also:

- "ALTER CELL" on page 8-15
- "DESCRIBE CELL" on page 8-54 for a description of the `metricHistoryDays` cell attribute

Managing Diagnostic Files

ADR Command Interpreter (ADRCI) is a command-line tool that you use to manage diagnostic data. Using ADRCI, you can perform the following duties:

- View diagnostic data within ADR
- Package incident and problem information into a zip file for transmission to Oracle Support Services

In order to use ADRCI with Oracle Exadata Storage Server Software, set the ADR base using the following command:

```
adrci> SET BASE /opt/oracle/cell/log
```

For example, with ADRCI you can view the alert, incident, and trace files for a cell, as shown in Example B–1.

Example B–1 Viewing Alert, Incident, and Trace Files

```
$ ADRCI
ADRCI: Release 11.2.0.1.0 - Production on Wed May 20 02:17:38 2009
Copyright (c) 1982, 2009, Oracle.  All rights reserved.

adrci> SET BASE /opt/oracle/cell/log

adrci> SHOW HOMES
ADR Homes:
diag/asm/cell/st-cell03-2
```

```
...

adrci> SET HOMEPATH diag/asm/cell/st-cell03-2

adrci> SHOW ALERT
...
adrci> SHOW INCIDENT
...
adrci> SHOW TRACEFILE
...
```

See Also: *Oracle Database Utilities* for additional information about ADRCI

Contacting Oracle Support Services

Some messages recommend contacting Oracle Support Services to report a problem. You may also want to contact Oracle Support Services when you have a service request submitted by Auto Service Request. When you contact Oracle Support Services, have the following information available:

- The hardware, operating system, and release number of the operating system running Oracle Database.
- The complete release number of Oracle Database, such as release 11.2.0.2.
- All Oracle programs (with release numbers) in use when the error occurred, such as SQL*Plus release 11.2.0.1.0.
- If you encountered one or more error codes or messages, then the exact code numbers and message text, in the order in which they appeared.
- The problem severity, according to the following codes:
  - 1: Program not usable. Critical effect on operations.
  - 2: Program usable. Operations severely restricted.
  - 3: Program usable with limited functions. Not critical to overall operations.
  - 4: Problem circumvented by customer. Minimal effect, if any, on operations.

You are also expected to provide the following:

- Your name
- The name of your organization
- Your Oracle Support ID number
- Your telephone number

Exadata Cell Error Messages

The following categories of Oracle Exadata Storage Server Software (Exadata Cell) error messages are described in this section:

- Error Messages: CELL-00001 to CELL-00500
- Error Messages: CELL-00501 to CELL-01000
- Error Messages: CELL-01001 to CELL-01500
- Error Messages: CELL-01501 to CELL-02000

Error Messages: CELL-00001 to CELL-00500

These error messages are in the range CELL-00001 through CELL-00500.

CELL-00001: Invalid date/time format. Please use a valid format, for example: <valid_date>

Cause: An invalid date/time format was entered.

Action: Enter the date/time in a valid format, such as `'2009-09-01T15:28:36-07:00'`.

See Also:

- Example 8–69 on page 8-99 for an example of the use of the date/time format in a `LIST` command
- "SET" on page 8-6 for additional information about setting the default date/time format

CELL-00002: IORMPLAN contains an invalid level.

Cause: An invalid level was entered in an IORMPLAN plan.

Action: Enter a level number from 1 to 8.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00003: IORMPLAN contains an invalid allocation.

Cause: An invalid allocation value was entered in an IORMPLAN plan.

Action: Enter an allocation value from 0 to 100.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00004: IORMPLAN contains an invalid name.

Cause: A name value was entered in an IORMPLAN plan that is a reserved word or contains too many characters.

Action: Enter a valid database or category name, or use `other` or `emergency` for the name.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00005: IORMPLAN contains too many directives.

Cause: More than 32 directives were entered in an IORMPLAN plan.

Action: Enter a maximum of 32 directives for each `dbPlan` or `catPlan` category.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00006: IORMPLAN contains an invalid allocation total.

Cause: The total of the allocation values at a given level exceeded 100.

Action: Enter allocation values such that the total of the allocation values at a given level does not exceed 100.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00007: IORMPLAN does not contain an 'other' allocation.

Cause: At least one directive with `name=other` was required for both the `dbPlan` and `catPlan` categories.

Action: Enter a directive with `name=other` for both the `dbPlan` and `catPlan` categories.

See Also:

- "Administering Database Resource Management" on page 6-8
- "ALTER IORMPLAN" on page 8-25

CELL-00010: This attribute requires 'true' or 'false' values: <attribute>

Cause: An incorrect value was entered for a Boolean attribute, such as the `smtpstarttls` cell attribute.

Action: Enter `true` or `false` for the value of the Boolean attribute.

See Also:

- "ALTER" on page 8-12 for additional information about altering attributes
- "DESCRIBE" on page 8-47 for lists of attributes that can be modified for each object type

CELL-00011: Maximum length exceeded for attribute: <attribute>

Cause: The value was longer than the attribute field allowed. For example, the text provided for the attribute value for `name` or the `comment` must be fewer than 256 characters.

Action: Enter a value equal to or less than the maximum length allowed for the attribute.

> **See Also:** "ALTER" on page 8-12 for additional information about altering attributes

CELL-00012: Name value is not valid. ASCII characters a-zA-Z_0-9 required.

Cause: Invalid characters, such as asterisk (`*`) or percent (`%`), were entered for the name attribute value.

Action: Enter a `name` value that contains only ASCII alphanumeric and underscore characters.

> **See Also:** "Restrictions on Values of Common Attributes" on page 8-9

CELL-00013: Attribute is not modifiable: <attribute>

Cause: The attribute could not be modified.

Action: Use `DESCRIBE` *`object_type`* to display a list of modifiable attributes for the specified object.

> **See Also:** "DESCRIBE" on page 8-47

CELL-00014: Object does not support this attribute: <attribute>

Cause: The specified attribute was not supported by the object type.

Action: Use `DESCRIBE` *`object_type`* to display a list of valid attributes for the specified object.

> **See Also:** "DESCRIBE" on page 8-47

Error Messages: CELL-00501 to CELL-01000

These error messages are in the range CELL-00501 through CELL-01000.

CELL-00501: Invalid syntax for powerCount attribute. Error: <error>

Cause: IPMI/ILO produced unexpected output.

Action: Contact Oracle Support Services.

CELL-00502: Cannot parse temp limits. Error: <error>

Cause: IPMI/ILO produced unexpected output.

Action: Contact Oracle Support Services.

CELL-00503: Invalid syntax for fanCount attribute. Error: <error>

Cause: The syntax for the `fanCount` attribute was invalid.

Action: Contact Oracle Support Services.

CELL-00506: Call updateHealth first.

Cause: A program error occurred.

Action: Contact Oracle Support Services.

CELL-00507: The first node of the doc must be sensorlist. Instead, it is: <name>

Cause: IPMI/ILO produced unexpected output.

Action: Contact Oracle Support Services.

CELL-00508: Reseting LO100 BMC requires that Linux is rebooted.

Cause: Cannot reset LO100 on DL180 G5 hardware because it leads to incorrect sensor readings and thresholds when the server is not rebooted after the BMC is reset.

Action: If you would like to reset BMC, then run the following commands as the `root` user. You may have to set the Oracle ASM disks to `online` on the cell when the cell comes back after the reboot on Exadata Cell software earlier than 11*g* Release 2 (11.2).

```
ipmitool mc reset cold
reboot
```

CELL-03510: MSHP3HardwareImpl not initialized properly. Received exception: <exception>.

Cause: An exception occurred during the HP hardware adapter initialization process.

Action: Examine the exception text for a possible solution. If a solution is not apparent, then contact Oracle Support Services.

CELL-03511: MSSN1HardwareImpl not initialized properly. Received exception: <exception>.

Cause: An exception occurred during the Sun hardware adapter initialization process.

Action: Examine the exception text for a possible solution. If a solution is not apparent, then contact Oracle Support Services.

Error Messages: CELL-01001 to CELL-01500

These error messages are in the range CELL-01001 through CELL-01500.

CELL-01005: Error in BMC adapter: <error>

Cause: A problem occurred with the BMC adapter.

Action: Contact Oracle Support Services.

Error Messages: CELL-01501 to CELL-02000

These error messages are in the range CELL-01501 through CELL-02000.

CELL-01501: Error spooling to file: <file>

Cause: The `SPOOL` command could not write to the specified file.

Action: Ensure that you have permissions to write to the file and to the directory.

See Also: "SPOOL" on page 8-6

CELL-01502: Error accessing file: <file>

Cause: The file could not be found.

Action: Check that the file exists in the specified location.

See Also: "SPOOL" on page 8-6

CELL-01503: File <file> already exists.

Cause: The spool file already exists so it could not be created.

Action: Specify the `REPLACE` option with the `SPOOL` command to overwrite an existing file, or specify the `APPEND` option with the `SPOOL` command to add to an existing file.

See Also: "SPOOL" on page 8-6

CELL-01504: Invalid command syntax.

Cause: The command had a syntax error.

Action: Check the help text for the valid command syntax. The caret (`^`) character points to the location in the command where the error was detected.

See Also: "HELP" on page 8-5

CELL-01505: The Restart Server (RS) is not running, so the state of Management Server (MS) or Cell Server (CELLSRV) is unknown.

Cause: Restart Server (RS) was not running.

Action: Start or restart the Restart Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-01506: No response from Restart Server (RS).

Cause: Restart Server (RS) was not available.

Action: Start or restart the Restart Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-01507: Cannot talk to the Restart Server (RS). Error: <error>

Cause: Restart Server (RS) was not available.

Action: Start or restart the Restart Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-01508: Restart Server (RS) is not running. Ensure that RS is running before requesting other 'ALTER CELL SERVICE' commands. <error>

Cause: Restart Server (RS) was not running.

Action: Start or restart the Restart Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-01509: Restart Server (RS) is not responding.

Cause: Restart Server (RS) was not available.

Action: Start or restart the Restart Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-01510: Cannot start a new Restart Server (RS) at port number: <port_number>. An instance of RS might be already running.

Cause: Restart Server (RS) could not be started with the specified port number.

Action: Contact Oracle Support Services.

CELL-01512: Cannot start a new Restart Server (RS). Exception received: <exception>

Cause: A new Restart Server (RS) could not be started.

Action: Contact Oracle Support Services.

CELL-01513: Remote host timed out during read operation.

Cause: The read operation was not successful because the remote host timed out.

Action: Contact Oracle Support Services.

CELL-01514: Connect Error. Verify that Management Server is listening at the specified HTTP port: <port_number>.

Cause: A problem occurred with the connection to the Management Server (MS).

Action: Contact Oracle Support Services.

CELL-01515: Unable to retrieve CellCLI process ID.

Cause: A problem occurred when running the CellCLI utility.

Action: Contact Oracle Support Services.

CELL-01516: Exception occurred in setup of CellCLI: <exception>.

Cause: A problem occurred when configuring the CellCLI utility.

Action: Contact Oracle Support Services.

CELL-01517: Cannot start Restart Server (RS). The operation timed out.

Cause: Restart Server (RS) could not be started.

Action: Contact Oracle Support Services.

CELL-01518: Stop CELLSRV. Create Cell cannot continue with CELLSRV running.

Cause: The cell could not be created when Cell Server (CELLSRV) was running.

Action: Contact Oracle Support Services.

CELL-01519: Cannot talk to the Management Server (MS). Error: <error_text>

Cause: A problem occurred when configuring the CellCLI utility.

Action: Contact Oracle Support Services.

CELL-01520: This command is not permitted in monitor mode.

Cause: The CellCLI utility was run in `monitor` mode. This mode is intended for read-only users. Commands other than `LIST` are not allowed. This error indicates that an `ALTER`, `CREATE`, or `DROP` command was issued.

Action: Run the CellCLI utility in regular mode to issue commands that update CellCLI objects.

CELL-01521: Only ASCII text can be used in CellCLI commands.

Cause: Non-ASCII text was entered in a CellCLI command.

Action: Enter the command with ASCII text only.

CELL-01522: CALIBRATE must be run as the root user id.

Cause: You were not logged into the cell as the `root` user.

Action: Exit the CellCLI utility. Log out of the cell, and then log in as the `root` user, and run `CALIBRATE`.

CELL-01523: CALIBRATE should not run while CELLSRV is running. Stop CELLSRV or enter CALIBRATE FORCE.

Cause: Oracle Storage Server was running on the cell.

Action: Shut down Oracle Storage Server with `ALTER CELL SHUTDOWN SERVICES CELLSRV`, and then run `CALIBRATE` again.

> **See Also:** "ALTER CELL" on page 8-15

CELL-01524: An error occurred while running CALIBRATE tests.

Cause: An internal error occurred.

Action: Check the error that is detailed in the `$LOG_HOME/cellcli.lst` log file, then contact Oracle Support Services.

CELL-01525: An error occurred while running CALIBRATE tests on LUN <lun_id>.

Cause: An internal error occurred.

Action: Check the error that is detailed in the `$LOG_HOME/cellcli.lst` log file, then contact Oracle Support Services.

CELL-01526: Local host name mapping is inconsistent. Verify cell /etc/hosts file content.

Cause: The cell was unable to ping the host name because of a network configuration error.

Action: The `/etc/hosts` file should be examined to verify that the host name of the cell is mapped to an IP address which is valid for the cell. Correct any errors, then verify that the host name can be pinged successfully before starting the cell services.

CELL-01527: CALIBRATE did not find any LUNs that qualified for measurement.

Cause: No LUNs were found that were in the normal operating state and had a valid size.

Action: Use the `LIST LUN DETAIL` command to retrieve data about the LUN, then determine the reason for the invalid LUN status.

> **See Also:** "LIST LUN" on page 8-114

CELL-01528: Unable to create the log file in directory {0}. Error received: {1}.

Cause: Permissions for the directory or the file do not allow the CellCLI log file to be written. The error should be checked to diagnose the problem. The CellCLI utility continues to work using the default logging configuration.

Action: Correct the error indicated in the directory.

Error Messages: CELL-02001 to CELL-02500

These error messages are in the range CELL-02001 through CELL-02500.

CELL-02001: Object name expected.

Cause: An object name was not entered with the command.

Action: Enter the object name with the CellCLI command.

> **See Also:** "LIST CELLDISK" on page 8-102

CELL-02002: Unexpected object in list for LIST.

Cause: An invalid object was entered with the `LIST` command.

Action: Use `DESCRIBE` *`object_type`* to display a list of valid attributes for the specified object. Check the Help text for the valid command syntax.

See Also:

- "DESCRIBE" on page 8-47
- "HELP" on page 8-5

CELL-02004: Cell object filtering is not supported with CellCLI commands.

Cause: Filtering could not be performed on the cell.

Action: Enter the `LIST CELL` command without a filtering clause.

See Also: "LIST CELLDISK" on page 8-102

CELL-02005: Invalid command.

Cause: There was an invalid command-object combination, such as `DROP CELL` or `ALTER PHYSICALDISK`.

Action: Check the help text for the valid command syntax.

See Also: "HELP" on page 8-5

CELL-02006: Physical disk does not exist: <physical_disk>

Cause: The name provided did not match any existing physical disks.

Action: Use the `LIST PHYSICALDISK` command to display the names of the existing physical disks.

See Also: "LIST PHYSICALDISK" on page 8-120

CELL-02007: Grid disk does not exist: <grid_disk>

Cause: The name provided did not match any existing grid disks.

Action: Use the `LIST GRIDDISK` command to display the names of the existing grid disks.

See Also: "LIST GRIDDISK" on page 8-107

CELL-02008: LUN does not exist: <lun>

Cause: The name provided did not match any existing LUNs.

Action: Use the `LIST LUN` command to display the names of the existing LUNs.

See Also: "LIST LUN" on page 8-114

CELL-02010: Metric definition does not exist: <metric_definition>

Cause: The name provided does not match any existing metric definitions.

Action: Use the `LIST METRICDEFINITION` command to display the names of the existing metric definitions.

See Also: "LIST METRICDEFINITION" on page 8-117

CELL-02012: ACTIVEREQUEST does not exist: <active_request>

Cause: The name provided did not match any existing active requests.

Action: Use the `LIST ACTIVEREQUEST` command to display the names of the existing active requests.

See Also: "LIST ACTIVEREQUEST" on page 8-97

CELL-02013: Alert definition does not exist: <alert_definition>

Cause: The name provided did not match any existing alert definitions.

Action: Use the `LIST ALERTDEFINITION` command to display the names of the existing alert definitions.

See Also: "LIST ALERTDEFINITION" on page 8-98

CELL-02014: Invalid object type.

Cause: An invalid object type was specified.

Action: Enter a valid object type with the CellCLI command.

CELL-02015: Alert does not exist: <alert>

Cause: An alert named in `LIST ALERTHISTORY` did not exist.

Action: Use the `LIST ALERTHISTORY` command to display the names of the existing alerts.

See Also: "LIST ALERTHISTORY" on page 8-99

CELL-02016: Metric does not exist: <metric>

Cause: A metric named in `LIST METRICHISTORY` did not exist.

Action: Use the `LIST METRICDEFINITION` command to display the names of the existing metrics.

See Also: "LIST METRICDEFINITION" on page 8-117

CELL-02017: Metrics do not exist: <metric>

Cause: Metrics named in `LIST METRICHISTORY` do not exist.

Action: Use the `LIST METRICDEFINITION` command to display the names of the existing metrics.

See Also: "LIST METRICDEFINITION" on page 8-117

CELL-02018: Internal error detected: <error>

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02019: Internal CLI/MS version error. MS expected <version>, but CLI sent <version>.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

Error Messages: CELL-02501 to CELL-03000

These error messages are in the range CELL-02501 through CELL-03000.

CELL-02501: Preexisting ALERTDEFINITION: <alert_definition>

Cause: An internal error occurred. Alert definitions could not be added or renamed.

Action: Contact Oracle Support Services.

CELL-02503: Invalid operation.

Cause: This generic error message displayed because invalid command syntax, such as `DROP CELL` or `ALTER CELL INACTIVE`, was entered.

Action: Check the Help text for the valid command syntax.

See Also: "HELP" on page 8-5

CELL-02504: Invalid IP address format.

Cause: An invalid string was entered for the IP address value.

Action: Enter the IP address in the valid Classless Inter-Domain Routing (CIDR) format: `ip1.ip2.ip2.ip4/bitmask`.

For example:

```
172.16.201.232/21
```

CELL-02505: Invalid number value for attribute: <attribute>

Cause: An invalid number was entered, such as a negative number.

Action: Enter a valid number. For example, enter a positive integer for the SMTP port number.

See Also: "ALTER CELL" on page 8-15

CELL-02507: Invalid cellnumber value.

Cause: An invalid number was entered, such as a negative number.

Action: Enter a valid positive number for the `cellNumber` attribute.

See Also: "ALTER CELL" on page 8-15

CELL-02508: Error writing the configuration file. Error: <error>

Cause: The configuration file on the cell could not be written to.

Action: Check the write permissions on the configuration file and directory. For example, ensure that it is possible to write to the `cellinit.ora` file on the cell.

CELL-02509: The 'ALTER CELL ONLINE/OFFLINE' command option is not supported.

Cause: The cell could not be taken online or offline.

Action: No action is required. The cell is online by default.

CELL-02510: Invalid IPblock format: <format >

Cause: An invalid string was entered for the IPBlock value.

Action: Enter the IPblock in valid Classless Inter-Domain Routing (CIDR) format `ipaddress/bitmask`.

For example:

```
172.16.201/24
```

CELL-02512: There was an error setting event: <event>

Cause: This error may occur when Management Server (MS) could not communicate properly with Cell Server (CELLSRV).

Action: Restart CELLSRV with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-02513: There is a communication error between MS and CELLSRV while getting event: <event>

Cause: This error could occur when Management Server (MS) could not communicate properly with Cell Server (CELLSRV).

Action: Restart CELLSRV with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-02518: Either LUN or PHYSICALDISK, but not both, is required to create a cell disk.

Cause: Both the `lun` and `physicalDisk` attribute values were specified when creating a cell disk.

Action: Specify either the `lun` or `physicalDisk` attribute value when creating a named cell disk.

See Also: "CREATE CELLDISK" on page 8-39

CELL-02522: Celldisk attribute required.

Cause: No value was entered for the `cellDisk` attribute in a `CREATE GRIDDISK` command.

Action: Enter a valid name for the `cellDisk` attribute.

See Also: "CREATE GRIDDISK" on page 8-42

CELL-02523: Celldisk name is invalid.

Cause: An invalid name was entered for the `cellDisk` attribute.

Action: Enter a valid name for the `cellDisk` attribute. Use `LIST CELLDISK` to display valid cell disk names.

See Also:

- "LIST CELLDISK" on page 8-102
- "CREATE GRIDDISK" on page 8-42

CELL-02524: Cell Server (CELLSRV) cannot successfully add grid disk: <grid_disk>

Cause: Either Cell Server (CELLSRV) failed or some other indication of the problem was included with the error message.

Action: Restart CELLSRV with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-02525: Unknown cell disk: <cell_disk>

Cause: The specified cell disk in the CellCLI command did not exist.

Action: Use `LIST CELLDISK` to display existing cell disks, then reenter the command with an existing cell disk name.

See Also: "LIST CELLDISK" on page 8-102

CELL-02526: Preexisting cell disk: <cell_disk>

Cause: A new cell disk could not be created with the name of an existing cell disk.

Action: Use `LIST CELLDISK` to display existing cell disks, then reenter the command with a unique cell disk name that does not already exit.

See Also: "LIST CELLDISK" on page 8-102

CELL-02528: Preexisting grid disk: <grid_disk>

Cause: A new grid disk could not be created with the name of an existing grid disk.

Action: Use `LIST GRIDDISK` to display existing grid disks, then reenter the command with a unique grid disk name that does not already exist.

See Also: "LIST GRIDDISK" on page 8-107

CELL-02529: CREATE command is not applicable to metric definitions.

Cause: Metric definitions could not be created with the CellCLI `CREATE` command.

Action: No action is required.

CELL-02531: CREATE command is not applicable to alert definitions.

Cause: Alert definitions could not be created with the CellCLI `CREATE` command.

Action: No action is required.

CELL-02532: ALTER command is not applicable to alert definitions.

Cause: Alert definitions could not be altered with the CellCLI `ALTER` command.

Action: No action is required.

CELL-02534: The command has not been implemented.

Cause: The specified CellCLI command was not supported in this release.

Action: No action is required.

CELL-02539: Cannot get cell disk GUIDs from Cell Server (CELLSRV) during initial Management Server (MS) - CELLSRV synchronization.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02540: Cannot get cell disk information from Cell Server (CELLSRV) for cell disk GUID: <guid>

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02541: Cannot get grid disk GUIDs from Cell Server (CELLSRV).

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02542: Cannot get grid disk information from Cell Server (CELLSRV).

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02543: Cannot locate LUN for cell disk: <cell_disk>, partition: <partition>

Cause: During Management Server (MS) startup, MS could not synchronize the cell disk information with information that was on the disks because the LUN had disappeared since the cell disks were created.

Action: Identify the missing LUN and diagnose the cell disk.

CELL-02544: Invalid IORMPLAN attribute list.

Cause: An invalid attribute was specified in a CellCLI command.

Action: Use `DESCRIBE IORMPLAN` to display a list of valid attributes.

See Also: "DESCRIBE IORMPLAN" on page 8-66

CELL-02546: I/O Resource Management (IORM) plan push: <error>

Cause: The IORMPLAN plan could not be pushed from CellCLI or Management Server to Cell Server.

Action: Contact Oracle Support Services.

CELL-02547: Cannot communicate with Cell Server (CELLSRV).

Cause: Management Server could not communicate with Cell Server. Cell Server may have failed.

Action: Restart Cell Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-02549: Grid disk is in use and FORCE is not specified for the operation.

Cause: The grid disk was in use and required the `FORCE` option to complete the operation.

Action: Enter the CellCLI command using the `FORCE` option.

See Also: "DROP GRIDDISK" on page 8-88

CELL-02550: Cell Server (CELLSRV) cannot drop the grid disk.

Cause: Cell Server (CELLSRV) could not drop the grid disk due to an error.

Action: Restart CELLSRV with the `ALTER CELL` command. Then drop the grid disk.

See Also:

- "ALTER CELL" on page 8-15
- "DROP GRIDDISK" on page 8-88

CELL-02551: Invalid command arguments.

Cause: Invalid or missing arguments raised an error with the CellCLI command.

Action: Check the help text for the valid command syntax.

See Also: "HELP" on page 8-5

CELL-02552: Current disk state is not 'inactive'.

Cause: The operation could not be performed because the grid disk was active.

Action: Use `LIST GRIDDISK` *`gdisk_name`* `DETAIL` or `LIST GRIDDISK` *`gdisk_name`* `ATTRIBUTES status` to display the status of the grid disk. Use

ALTER GRIDDISK with the INACTIVE option to set the grid disk to inactive mode.

See Also:

- "ALTER GRIDDISK" on page 8-22
- "LIST GRIDDISK" on page 8-107

CELL-02553: Current disk state is not 'active'.

Cause: The operation could not be performed because the grid disk was inactive.

Action: Use LIST GRIDDISK *gdisk_name* DETAIL or LIST GRIDDISK *gdisk_name* ATTRIBUTES status to display the status of the grid disk. Use ALTER GRIDDISK with the ACTIVE option to set the grid disk to inactive mode.

See Also:

- "ALTER GRIDDISK" on page 8-22
- "LIST GRIDDISK" on page 8-107

CELL-02556: Attribute value must be greater than zero: <attribute>

Cause: A value equal to or less than zero was entered for the attribute.

Action: Enter a numeric value greater than zero for the attribute.

CELL-02557: Attribute value must be positive: <attribute>

Cause: A value equal to or less than zero was entered for the attribute.

Action: Enter a numeric value greater than zero.

CELL-02558: GUID is NULL. Cannot drop from Cell Server (CELLSRV).

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02559: There is a communication error between MS and Cell Server (CELLSRV).

Cause: Management Server could not communicate with Cell Server.

Action: Restart Cell Server with the ALTER CELL command.

See Also: "ALTER CELL" on page 8-15

CELL-02560: Duplicate name.

Cause: There was an existing object with the same name.

Action: Choose a unique name for the object for the cell.

CELL-02561: Cell disk already exists on the specified device.

Cause: There was a cell disk already on the device.

Action: Drop the existing cell disk before retrying the operation, or specify a different device for the new cell disk.

CELL-02563: Specified partition cannot be formatted.

Cause: An error occurred when trying to format the specified partition.

Action: Contact Oracle Support Services.

CELL-02564: Specified object does not exist.

Cause: An invalid object was specified with a command.

Action: Use `LIST` *`object_type`* to display a list of valid objects for the specified object type. Check the help text for the valid command syntax.

See Also:

- "LIST" on page 8-95
- "HELP" on page 8-5

CELL-02565: Wrong offset specified.

Cause: An incorrect offset value was entered.

Action: Reenter the command with a valid offset value.

See Also: "CREATE GRIDDISK" on page 8-42

CELL-02566: No space for the specified allocation.

Cause: No free space was available for the allocation specified.

Action: Contact Oracle Support Services.

CELL-02568: One or more grid disks already exist on the specified cell disk.

Cause: Existing grid disks were found on the cell disk specified in the operation.

Action: Drop the existing grid disks before retrying the operation on the specified cell disk or specify a different cell disk for the operation.

CELL-02569: Invalid grid disk state.

Cause: A problem occurred with the state of the grid disk.

Action: Contact Oracle Support Services.

CELL-02574: Invalid message sent by management command.

Cause: An invalid message was sent by Management Server.

Action: Contact Oracle Support Services.

CELL-02575: The value for notificationPolicy should be NONE or any combination of CRITICAL, WARNING and CLEAR.

Cause: The valid value for `notificationPolicy` was not `none` or a combination of `critical`, `warning`, or `clear`.

Action: Enter one of the valid combinations displayed in the error message.

See Also: "ALTER CELL" on page 8-15

CELL-02576: The value for notificationMethod should be MAIL, SNMP, or both.

Cause: The valid value for `notificationMethod` was not `mail`, `snmp`, or a combination of those two values.

Action: Enter one of the valid combinations displayed in the error message.

See Also: "ALTER CELL" on page 8-15

CELL-02577: CREATE GRIDDISK ALL and DROP GRIDDISK ALL commands require a prefix attribute.

Cause: When the `CREATE GRIDDISK` and `DROP GRIDDISK` commands are run with the `ALL` option, the `prefix` attribute must be specified to indicate the prefix of the disk names.

Action: Check the Help text for the valid command syntax, and the syntax of a specific command.

See Also:

- "HELP" on page 8-5
- "CREATE GRIDDISK" on page 8-42
- "DROP GRIDDISK" on page 8-88

CELL-02578: An error was detected in the SMTP configuration: <error text> Please verify your SMTP configuration.

Cause: A problem occurred with the SMTP configuration for the cell.

Action: Use ALTER CELL to correct the SMTP configuration.

See Also: "ALTER CELL" on page 8-15

CELL-02579: An error was detected in the SNMP configuration: <error text> Please verify your SNMP configuration.

Cause: A problem occurred with the SNMP configuration for the cell.

Action: Use ALTER CELL to correct the SNMP configuration.

See Also: "ALTER CELL" on page 8-15

CELL-02580: An error occurred while retrieving metrics: <error text>.

Cause: A problem occurred when retrieving metrics for the cell.

Action: Contact Oracle Support Services.

CELL-02581: Cell disk <cell_disk_name> on LUN <lun_name> has not been exported and FORCE is not specified for import.

Cause: The cell disk had not been properly exported with the `EXPORT` command on a cell before the cell disk was moved to a different cell. This problem can occur if `EXPORT` had not been run on the cell disk.

Action: Use the `FORCE` option when running the `IMPORT` command on the cell disk.

CELL-02582: The structures for the cell disk on LUN <lun> cannot be erased from CELLSRV memory.

Cause: A problem occurred while attempting an `EXPORT` or `IMPORT` operation.

Action: Contact Oracle Support Services.

CELL-02583: The operation is not permitted on this cell disk.

Cause: A problem occurred while attempting an `EXPORT` or `IMPORT` operation. For example, a grid disk cannot be created on a cell disk that was in a `Requires Import` state.

Action: Contact Oracle Support Services.

CELL-02590: Preexisting threshold: <name>.

Cause: The threshold names were already used.

Action: Use a unique threshold name.

See Also: "CREATE THRESHOLD" on page 8-46

CELL-02591: Threshold does not exist: <name>.

Cause: The threshold name did not exist.

Action: Use an existing threshold name.

See Also: "ALTER THRESHOLD" on page 8-32

CELL-02592: Threshold comparison operator must be '<', '<=' '=', '>=', or '>'.

Cause: An incorrect comparison operator was entered.

Action: Use `'<'`, `'<='`, `'='`, `'>='`, or `'>'` for the comparison operator.

See Also: "CREATE THRESHOLD" on page 8-46

CELL-02593: Threshold name must identify a valid metric: <name>.

Cause: An invalid metric name was entered.

Action: Use an existing metric name.

See Also: "CREATE THRESHOLD" on page 8-46

CELL-02594: Threshold comparison attribute required.

Cause: A comparison operator was not entered with this specified metric.

Action: Enter a comparison operator when creating a threshold for this metric.

See Also: "CREATE THRESHOLD" on page 8-46

CELL-02595: Thresholds are not supported for this metric: <name>.

Cause: A threshold was created on a metric that was not supported.

Action: Create a threshold for a supported metric.

See Also: "CREATE THRESHOLD" on page 8-46

CELL-02596: The IPaddress attribute <name> is not in correct CIDR format.

Cause: The `ipaddress` attribute was incorrect.

Action: Enter a valid IP address format.

See Also: "CREATE CELL" on page 8-37

CELL-02597: Could not update the network configuration file: <config_file>.

Cause: The network configuration was not updated.

Action: Contact Oracle Support Services.

CELL-02598: ipaddress/Netmask attribute is not properly configured for interconnect <name>.

Cause: The `ipaddress` attribute was not valid for the corresponding `interconnect` attribute.

Action: Enter a valid IP address.

See Also: "CREATE CELL" on page 8-37

CELL-02599: Cannot determine the host name. Please specify the cellname.

Cause: The host name of the cell was not identified.

Action: Enter a name for the cell.

See Also: "CREATE CELL" on page 8-37

CELL-02600: Interconnect1 must be specified if interconnect2 is specified.

Cause: The `interconnect2` attribute was specified, but `interconnect1` was not.

Action: Specify `interconnect1` before specifying `interconnect2`.

See Also: "CREATE CELL" on page 8-37

CELL-02601: Interconnect2 must be specified if interconnect3 is specified.

Cause: The `interconnect3` attribute was specified, but `interconnect2` was not.

Action: Specify `interconnect2` before specifying `interconnect3`.

See Also: "CREATE CELL" on page 8-37

CELL-02602: Interconnect3 must be specified if interconnect4 is specified.

Cause: The `interconnect4` attribute was specified, but `interconnect3` was not.

Action: Specify `interconnect3` before specifying `interconnect4`.

See Also: "CREATE CELL" on page 8-37

CELL-02603: If the IPblock attribute is specified, four network interconnects must be configured.

Cause: The `ipBlock` attribute was specified, but four network interconnections were not configured.

Action: Configure four network interconnections before specifying the `ipBlock` attribute.

See Also: "CREATE CELL" on page 8-37

CELL-02604: Specify no more than one IPaddress attribute for each network interconnect (at most 4 IP addresses).

Cause: More `ipaddress` attributes were specified than there were configured interconnections.

Action: Specify one IP address for each configured network interconnection.

See Also: "CREATE CELL" on page 8-37

CELL-02605: Cannot specify an IPaddress attribute if an interconnect is not specified.

Cause: An `ipaddress` attribute was specified without a corresponding interconnection.

Action: Specify one IP address for each specified network interconnection.

See Also: "CREATE CELL" on page 8-37

CELL-02606: Restarting the network with the new configuration failed.

Cause: The `interconnect` and `ipaddress` attribute values were incorrect.

Action: Specify the correct values for the `interconnect` and `ipaddress` attributes.

See Also:

- "ALTER CELL" on page 8-15
- "CREATE CELL" on page 8-37

CELL-02607: Specify at least one network interconnect.

Cause: A network interconnection was not specified.

Action: Specify at least the `interconnect1` attribute.

See Also: "CREATE CELL" on page 8-37

CELL-02608: A valid network interconnect is not specified.

Cause: A valid network interconnection was not specified.

Action: Specify a valid network interconnection.

See Also:

- "ALTER CELL" on page 8-15
- "CREATE CELL" on page 8-37

CELL-02609: Specify both the IPblock and cellnumber together and without any IPaddress attributes.

Cause: An `ipaddress` attribute was specified with the `ipBlock` and `cellNumber` attributes.

Action: Specify both the `ipBlock` and `cellNumber` attributes without an `ipaddress` attribute.

See Also: "CREATE CELL" on page 8-37

CELL-02610: An invalid setting was specified for the traceLevel attribute. Specify a valid JAVA logging level (SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST), or a valid ODL logging level (INCIDENT_ERROR:1, ERROR:1, WARNING:1, NOTIFICATION:1, NOTIFICATION:16, TRACE:1, TRACE:16, TRACE:32).

Cause: An invalid setting was specified for the `traceLevel` attribute with the `ALTER CELL` command.

Action: Specify a valid JAVA logging level or a valid ODL logging level for the `traceLevel` attribute.

CELL-02611: Cannot access the specified device.

Cause: A device was not accessible.

Action: Contact Oracle Support Services.

CELL-02612: Operation cannot be completed due to IO errors.

Cause: An operation failed because of I/O errors.

Action: Contact Oracle Support Services.

CELL-02613: Operation failed because the cell disk is corrupted.

Cause: A cell disk was corrupted.

Action: Contact Oracle Support Services.

CELL-02614: The role attribute value must be PRIMARY or STANDBY.

Cause: The role attribute was not specified correctly.

Action: Specify the `role` attribute as `primary` or `standby`.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02615: The role attribute is not permitted on catPlan directives.

Cause: The `role` attribute was specified for a `catPlan` directive.

Action: Remove the `role` attribute from the `catPlan` directive.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02616: The role attribute is not permitted on dbPlan 'other' directives.

Cause: The `role` attribute was specified for the `dbPlan other` directive.

Action: Remove the `role` attribute from the `dbPlan other` directive.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02617: A name list is permitted only for dbPlan directives with 'role=primary'.

Cause: A name list was specified for `dbPlan` directives that did not have `role` equal to `primary`.

Action: Remove the `role` attribute from the `dbPlan` directive.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02618: Current IORMPLAN state is not 'inactive'.

Cause: The IORMPLAN state was not inactive.

Action: Change the IORMPLAN state to inactive.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02619: Current IORMPLAN state is not 'active'.

Cause: The IORMPLAN state was not active.

Action: Change the IORMPLAN state to active.

See Also: "ALTER IORMPLAN" on page 8-25

CELL-02620: An unmapped CELLSRV error has occurred. The internal message is: <error_msg>.

Cause: An unexpected error was returned to Management Server from Cell Server.

Action: Contact Oracle Support Services.

CELL-02621: Interface <interface> is neither Ethernet nor InfiniBand.

Cause: An invalid interface was entered.

Action: Enter a valid Ethernet or InfiniBand interface.

CELL-02622: Interface <interface> supports bootproto <value>. It must be either static or dhcp.

Cause: An invalid interface was entered.

Action: Enter `static` or `dhcp`.

CELL-02623: The command <command> returned an error code <error_code>.

Cause: An error occurred while running the command.

Action: Contact Oracle Support Services.

CELL-02624: Error while executing command <command>.

Cause: An error occurred while running the command.

Action: Contact Oracle Support Services.

CELL-02627: There is a communication error between MS and CELLSRV. Configuration file cellinit.ora is malformed or does not include required information.

Cause: The `cellinit.ora` file did not have the correct information or the correct format.

Action: Edit the `cellinit.ora` file to correct the problem.

See Also: "Setting Up Configuration Files for a Database Server Host" on page 2-15

CELL-02628: There is an internal error in the MS-CELLSRV communication module. Create an incident package and submit the package to Oracle Support Services.

Cause: An error occurred in the MS-CELLSRV communication module.

Action: Create an incident package and submit the package to Oracle Support Services.

CELL-02630: There is a communication error between Management Server and Cell Server caused by a mismatch of security keys. Check that both servers have access to and use of the same $OSSCONF/cellmskey.ora file.

Cause: The security keys for Management Server and Cell Server did not match.

Action: Contact Oracle Support Services.

CELL-02631: Restarting the openibd stack with the new configuration failed.

Cause: The `service openibd restart` command has failed.

Action: Verify that the InfiniBand configuration is valid. Correct any errors, then reenter the command.

CELL-02632: Operation failed because the disk GUID is not unique.

Cause: Two cell disks have identical identifiers (GUID). The most likely cause is that a cell disk was duplicated by copying its content of underlying LUN or partition to another disk using OS utilities.

Action: Make underlying LUN or partition with duplicated content unavailable to the operating system, such as by removing the physical disk from the system.

CELL-02640: Interconnect{0} cannot be removed because interconnect{1} has previously been specified, or is specified in the current command.

Cause: The user was attempting to remove an IP address that is between two other IP addresses, such as removing `ipaddress2` when `ipaddress1` and `ipaddress3` are in use.

Action: Remove `interconnectN+1` before removing `interconnectN`.

CELL-02641: Interconnect{1} must be specified if interconnect{0} is specified.

Cause: The user was attempting to specify `interconnectN+1` before specifying `interconnectN`.

Action: Specify the interconnections in order. Multiple interconnections can be specified in the same command.

CELL-02642: Empty string IP address was specified with non-empty interconnect{0}.

Cause: An IP address was specified for a non-empty interconnection.

Action: The user must specify either one, or both, as empty. For example: `ALTER CELL interconnect1=""`,`ALTER CELL ipaddress1=""`, or `ALTER CELL interconnect1="", ipaddress1=""`.

CELL-02643: DROP ALERTHISTORY command did not include all members of the alert sequence for <alertNum>. All members of the sequence must be dropped together.

Cause: The command did not include all members of the alert sequence.

Action: The user must include all members in the alert sequence when dropping stateful alerts.

CELL-02644: Flash cache does not exist.

Cause: The `DROP FLASHCACHE` or `ALTER FLASHCACHE` command failed because flash cache does not exist.

Action: Do nothing, or use the `CREATE FLASHCACHE` command to create the flash cache.

CELL-02645: Flash cache already exists.

Cause: The `CREATE FLASHCACHE` command failed because flash cache already exists.

Action: Do nothing, or use the `ALTER FLASHCACHE` command to change the existing flash cache.

CELL-02646: CREATE GRIDDISK failed due to insufficient free space on cell disks: cd1, cd2, ...

Cause: The `CREATE GRIDDISK` command failed because there was insufficient free space on the cells based on the size attributed specified in the command.

Action: Do nothing. The grid disks are created on other cell disks.

CELL-02647: Flashcache requires Flash cell disks. The cell disk provided is not a working Flash cell disk: <cd-name>.

Cause: The `CREATE FLASHCACHE CELLDISK` command failed because a flash cell disk was not specified.

Action: Specify a flash cell disk when creating Exadata Smart Flash Cache.

CELL-02648: Cannot export cell disk that has Smart Flash Cache on it: {0}..

Cause: The cell disk cannot be exported because the cell disk was defined on Exadata Smart Flash Cache.

Action: Correct the command to export a different cell disk.

CELL-02649 Cannot get flash cache information from Cell Server (CELLSRV).

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02650: An invalid setting <level> was specified for the collectionLevel attribute. Specify a valid collection level.

Cause: An invalid collection level was specified in the command.

Action: Specify a valid collection level.

CELL-02651: There is a communication error between MS and CELLSRV caused by an unrecognized configuration file version.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-02652: CREATE FLASHCACHE failed due to insufficient free space on cell disks: <list>

Cause: There was not enough free space on the cell.

Action: Do nothing. The flash cache are created on other cell disks.

CELL-02654: Terminating write operation to metrichistory pipe: Write timed out.

Cause: The write operation timed out.

Action: Call Oracle Support Services.

CELL-02655: Could not write to metrichistory pipe: {0}.

Cause: The `LIST METRICHISTORY` command encountered a timeout during a write to an internal pipe file.

Action: Call Oracle Support Services.

CELL-02656: Could not create metrichistory pipe directory: {0}.

Cause: The `LIST METRICHISTORY` command encountered an error while creating a temporary directory for an internal pipe file.

Action: Call Oracle Support Services.

CELL-02657: Could not create metrichistory pipe file {0}: {1}.

Cause: The `LIST METRICHISTORY` command encountered an error while creating an internal pipe file.

Action: Call Oracle Support Services.

CELL-02658: Could not find metrichistory pipe file {0}.

Cause: The `LIST METRICHISTORY` command encountered an error finding its internal pipe file.

Action: Call Oracle Support Services.

CELL-02659: Could not read from metrichistory pipe file {0}: {1}.

Cause: The `LIST METRICHISTORY` command encountered a error reading from an internal pipe file.

Action: Call Oracle Support Services.

CELL-02660: Invalid value for attribute: {0}.

Cause: The `ALTER` command that was entered had an incorrect value in its assignment to an attribute.

Action: Fix the `ALTER` command to use the correct value format.

CELL-02661: The grid disk resize operaton has failed for {0}.

Cause: An invalid size was requested for the grid disk, and there is not sufficient space available on the cell disk.

Action: Use `LIST GRIDDISK DETAIL` to determine the current size of the grid disk, and `LIST CELLDISK DETAIL` to determine the available free space on the cell disk.

CELL-02662: This operation is denied because a resize operation is currently running for grid disk {0}.

Cause: A resize operation was already in progress on the grid disk. Only one resize is allowed.

Action: Use `LIST GRIDDISK` *`grid_disk`* `ATTRIBUTES lastResizeStatus` to monitor the resize status of *grid_disk* before issuing a new resize request.

CELL-02663: This operation is denied becauase a resize operation is currently running on one of the grid disk onf the cell disk.

Cause: A resize operation was already in progress on another grid disk of the same cell disk.

Action: Use `LIST GRIDDISK` to find all the grid disks of the cell disk. Next, use `LIST GRIDDISK` *`grid_disk`* `ATTRIBUTES lastResizeStatus` to determine if any of the grid disks are still being resized before issuing a new resize request.

CELL-02675: Cannot get quarantine information from Cell Server (CELLSRV).

Cause: A communication error prevents retrieval of quarantine information from CELLSRV.

Action: Check that CELLSRV is running, and start it if it is not running. Otherwise, contact Oracle Support Services.

CELL-02676: Quarantine does not exist: <quarantine id>.

Cause: The quarantine identifier specified on the CellCLI command does not match an existing quarantine.

Action: Use the `LIST QUARANTINE` command to list the available quarantine entries. Use the name of an available quarantine.

CELL-02677: Cannot drop the quarantine. Error: <error-message>.

Cause: The quarantine could not be dropped due to an error

Action: Correct the specified error before attempting to drop the quarantine again.

CELL-02678: Cannot alter the quarantine. Error: <error-message>.

Cause: The quarantine could not be altered due to an error.

Action: Correct the specified error before attempting to alter the quarantine again.

CELL-02679: Cannot add the quarantine. Error: <error-message>.

Cause: The quarantine could not be created due to an error.

Action: Correct the specified error before attempting to create the quarantine again.

CELL-02680: Quarantine creation requires that attribute quarantineType be assigned one of the following values: <type-list>.

Cause: The quarantine type attribute was not provided or it was not assigned a correct type value.

Action: Correct the `CREATE QUARANTINE` command to contain a valid `quarantineType`.

CELL-02682 : Invalid Quarantine ID.

Cause: The quarantine operation used an invalid quarantine identifier.

Action: Contact Oracle Support Services.

CELL-02683: Quarantine Manager Disabled.

Cause: The fault isolation feature has been disabled.

Action: Contact Oracle Support Services.

CELL-02684 : Quarantine Manager Unknown Error.

Cause: Action failed due to an unknown error.

Action: Contact Oracle Support Services.

CELL-02685 : Invalid Attribute for the Quarantine Type Specified.

Cause: A `CREATE QUARANTINE` or `ALTER QUARANTINE` command specified invalid attributes.

Action: Correct the attributes, and reissue the command.

CELL-02686 : Duplicate Quarantine Entry.

Cause: A `CREATE QUARANTINE` command attempted to create a quarantine that already exists.

Action: If a different quarantine was intended, then modify the attributes to specify the new quarantine.

Error Messages: CELL-03001 to CELL-03500

These error messages are in the range CELL-03001 through CELL-03500.

CELL-03001: An exception occurred during ADR mining: <exception>

Cause: An internal error occurred while searching for incidents in the Automatic Diagnostic Repository (ADR).

Action: Contact Oracle Support Services.

CELL-03002: Failed to parse value for key: <key>

Cause: An internal error occurred while searching for incidents in the Automatic Diagnostic Repository (ADR).

Action: Contact Oracle Support Services.

Error Messages: CELL-03501 to CELL-04000

These error messages are in the range CELL-03501 through CELL-04000.

CELL-03501: MSFK1HardwareImpl not initialized properly. Received exception: <exception>

Cause: `FK1` represents the fake hardware adapter class. This virtual hardware was only used for testing purposes and could not be initialized.

Action: Contact Oracle Support Services.

CELL-03502: MSHP1HardwareImpl not initialized properly. Received exception: <exception>

Cause: A problem occurred with the `HP1` hardware (`HP1` is the cell hardware appliance code).

Action: Contact Oracle Support Services.

CELL-03503: MSHardwareImpl not initialized properly. Received exception: <exception>

Cause: A problem occurred with the hardware (Exadata Cell).

Action: Contact Oracle Support Services.

CELL-03504: Cannot recognize appliance type: <type>

Cause: The cell hardware was not a recognized type.

Action: Contact Oracle Support Services.

CELL-03505: Cannot get appliance type: <type>

Cause: The cell hardware type was not obtained.

Action: Contact Oracle Support Services.

CELL-03506: Cannot configure BMC device on fake hardware.

Cause: This virtual hardware was only used for testing purposes and could not be configured.

Action: No action is required.

CELL-03507: MSIN1HardwareImpl not initialized properly. Received exception: <exception>.

Cause: A problem occurred with the `IN1` hardware (`IN1` is the cell hardware appliance code).

Action: Contact Oracle Support Services.

CELL-03508: Exception raised during hardware poll: <exception>.

Cause: A problem occurred with this hardware polling operation.

Action: Contact Oracle Support Services.

CELL-03509: MSHP2HardwareImpl not initialized properly. Received exception: <exception>.

Cause: An exception occurred during the HP hardware adaptor initialization process.

Action: Examine the exception text for a possible fix. If a fix is not apparent, then contact Oracle Support Services.

Error Messages: CELL-04501 to CELL-05000

These error messages are in the range CELL-04501 through CELL-05000.

CELL-04501: Multiple responses received while expecting a single response for <error>.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-04502: Appliance type <type> is not recognized.

Cause: Invalid hardware was used for the cell or the hardware was not properly configured.

Action: Check the hardware configuration for the cell.

CELL-04503: PopulateAll not implemented for this hardware.

Cause: The hardware adapter was not completely implemented for this hardware.

Action: Contact Oracle Support Services.

CELL-04504: Partition append string not set for this appliance.

Cause: The hardware adapter was not completely implemented for this hardware.

Action: Contact Oracle Support Services.

CELL-04505: Function createSDL is not implemented

Cause: The hardware adapter was not completely implemented for this hardware.

Action: Contact Oracle Support Services.

CELL-04506: Failed to generate new unique cell disk name: <cell_disk>

Cause: An internal error occurred in the program.

Action: Contact Oracle Support Services.

CELL-04507: Function getOSNameForLun is not overridden.

Cause: The hardware adapter was not completely implemented for this hardware.

Action: Contact Oracle Support Services.

CELL-04508: Cannot get validity check for a cell disk partition from CELLSRV.

Cause: Cell Server and Management Server could not communicate with each other.

Action: Retry the command that was entered when the error occurred. If the problem persists, then try restarting the CELLSRV and MS services with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-04509: The LUN name cannot be found.

Cause: The specified LUN name was not found.

Action: Enter the `LIST LUN` command to display the names of the existing LUNs.

See Also: "LIST LUN" on page 8-114

CELL-04510: Disk object <name> does not have ATTR_LUN set.

Cause: An internal error occurred. An error occurred in the vendor-specific controller utility.

Action: Contact Oracle Support Services.

CELL-04511: For LUN <lun> cell disk is added <cell_disk >, LUN is part of Scratch/Swap: <disk>. Drop the cell disk with force and add it again.

Cause: A configuration problem was caused by the explicit use of the software RAID utility `mdadm`. Exadata Cell software uses the `mdadm` utility for configurations made through Cell Server. You should not change the software RAID configurations with this utility. If you change the default `mdadm` configurations, then there is a mismatch.

Action: Drop the cell disk with the `FORCE` option to delete the mismatched configuration. Re-create the cell disk to install the correct configuration.

See Also:

- "DROP CELLDISK" on page 8-85
- "CREATE CELLDISK" on page 8-39

CELL-04512: Invalid machine. Cannot create cell disk.

Cause: The cell was invalid.

Action: Contact Oracle Support Services.

CELL-04513: Cell Server (CELLSRV) cannot successfully add cell disk: <cell_disk>

Cause: The Cell Server could not add the specified cell disk.

Action: Restart Cell Server with the `ALTER CELL` command.

See Also: "ALTER CELL" on page 8-15

CELL-04514: LUN name cannot be obtained while adding cell disk to configuration file.

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04515: Disk object cannot be obtained for LUN: <lun>

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04516: LUN object cannot be obtained for cell disk: <celldisk>

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04517: Cell Server (CELLSRV) cannot drop the cell disk.

Cause: The Cell Server could not drop the specified cell disk.

Action: Drop the cell disk with the `FORCE` option. If that operation is not successful, then contact Oracle Support Services.

See Also: "DROP CELLDISK" on page 8-85

CELL-04519: Cannot complete the drop of cell disk <cell_disk>. Received error: Cell disks not dropped: <error>.

Cause: The specified cell disks could not be dropped.

Action: Individually drop the cell disks that could not be dropped.

See Also: "DROP CELLDISK" on page 8-85

CELL-04520: An existing celldisk entry is present for the LUN <lun>.

Cause: The LUN already had a cell disk present.

Action: Determine the name of the cell disk that is present on the LUN; see "LIST LUN" on page 8-114. Drop this cell disk using the `FORCE` option if appropriate; see "DROP CELLDISK" on page 8-85.

CELL-04521: The LUN <lun> has a valid celldisk (which is not imported).

Cause: The LUN already had a cell disk present.

Action: Use `CREATE CELLDISK` with the `FORCE` option to drop and create a new cell disk on this LUN.

See Also: "CREATE CELLDISK" on page 8-39

CELL-04522: The LUN <lun> has a valid celldisk.

Cause: The LUN already had a cell disk present.

Action: Determine the name of the cell disk that is present on the LUN. Drop this cell disk using the `FORCE` option if appropriate. Then create a new cell disk on the LUN. Or use `CREATE CELLDISK` with the `FORCE` option to drop and create a new cell disk on this LUN.

See Also:

- "LIST LUN" on page 8-114
- "DROP CELLDISK" on page 8-85
- "CREATE CELLDISK" on page 8-39

CELL-04523: Object name not found in attribute.

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04524: Physicaldisk object cannot be obtained for physical disk: <physical_disk>

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04525: LUN object cannot be obtained for LUN: <lun>

Cause: An internal program error occurred.

Action: Contact Oracle Support Services.

CELL-04526: Invalid type of object. Only LUN or physical disk can be provided as arguments for create celldisk.

Cause: An invalid object was provided as an argument for the `CREATE CELLDISK` command.

Action: Use `CREATE CELLDISK` with a valid LUN or physical disk attribute value.

See Also: "CREATE CELLDISK" on page 8-39

CELL-04527: Cannot complete creation of cell disk on object: <object>. Received error: <error> Cell disk not created on: <object>

Cause: A problem occurred when creating a cell disk on the named object.

Action: Use `CREATE CELLDISK` with the `FORCE` option.

See Also: "CREATE CELLDISK" on page 8-39

CELL-04528: Non-Single-Disk LUN found and needs to be reenabled. CREATE CELLDISK ALL cannot enable non-single-disk LUNs.

Cause: The cells support only single-disk LUNs.

Action: Remove the non-single-disk LUN, then use `CREATE CELLDISK` with the `ALL` option.

See Also: "CREATE CELLDISK" on page 8-39

CELL-04529: A failed LUN cannot be enabled.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-04530: Physical disk to be removed was not found in the list.

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-04531: Error encountered while creating Single-Disk LUN (SDL) on physical disk: <disk>. Received error: <error> Cell disk not created on: <cell_disk>

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-04532: Error encountered while creating cell disk on LUN: <lun>. Received error: <error> Cell disk not created on: <cell_disk>

Cause: An internal error could have occurred.

Action: The error text explains the error condition. Follow the recommended steps (if any) in the error text. If no steps are provided, then contact Oracle Support Services.

CELL-04534: Non-Single-Disk LUN found on physical disk: <physical_disk>

Cause: A non-single-disk LUN was found on the physical disk.

Action: Remove the non-single-disk LUN from the specified physical disk.

CELL-04535: No LUN object found for the created SDL on: <object>

Cause: An internal error occurred.

Action: Contact Oracle Support Services.

CELL-04536: The command <command> returned an exit status: <error_code>.

Cause: An internal error could have occurred.

Action: The error text explains the error condition. Follow the recommended steps (if any) in the error text. If no steps are provided, then contact Oracle Support Services.

CELL-04537: Cannot get disk controller ID.

Cause: An unsupported version was used on the vendor-specific controller utility.

Action: Contact Oracle Support Services.

CELL-04538: Physical disk <physical_disk> is part of a non-Single-Disk LUN. Cannot delete it to create a Single-Disk LUN.

Cause: A non-single-disk LUNs was present on a physical disk.

Action: Remove the non-single-disk LUN manually using the vendor specific controller utility.

CELL-04539: Cannot create single-disk LUN on physical disk <physical_disk> because there is already a LUN on this physical disk.

Cause: A LUN was already on this physical disk.

Action: Use the `FORCE` option if you want to overwrite an existing LUN.

CELL-04541: Operating system name for LUN <lun> not found.

Cause: An error occurred in the vendor-specific controller utility.

Action: Contact Oracle Support Services.

CELL-04543: Invalid physical drive ID for: <physical_disk>

Cause: An invalid ID was entered for the specified physical disk.

Action: Use the `LIST PHYSICALDISK` *`pdisk_name`* command to display the ID for the specified physical disk.

See Also: "LIST PHYSICALDISK" on page 8-120

CELL-04544: Invalid ID for LUN: <lun>

Cause: An invalid ID was entered for the specified LUN.

Action: Use the `LIST LUN` *`lun_name`* command to display the ID for the specified LUN.

See Also: "LIST LUN" on page 8-114

CELL-04545: Exception in Disk Adapter: <disk_adapter>

Cause: A problem occurred with the disk adapter.

Action: Contact Oracle Support Services.

CELL-04546: The command <command> returned an exit code of <code>. Kill any other instances of disk configuration tool processes.

Cause: A problem occurred with the disk configuration tool process.

Action: Contact Oracle Support Services.

CELL-04547: Cell Server (CELLSRV) cannot successfully export cell disk: <cell_disk>.

Cause: A problem occurred when exporting the cell disk.

Action: Contact Oracle Support Services.

CELL-04548: Cannot export cell disk <cell_disk> because its status is not normal.

Cause: Because the status of the cell disk was not normal, the cell disk could not be exported.

Action: Contact Oracle Support Services.

CELL-04549: Cannot obtain LUN for cell disk: <cell_disk>.

Cause: The LUN could not be accessed for the specified cell disk.

Action: Contact Oracle Support Services.

CELL-04550: Export is not implemented for this hardware.

Cause: The export function was not implemented on the hardware.

Action: Contact Oracle Support Services.

CELL-04551: Error encountered during export of cell disk <cell_disk>. Received error: <code>. Cell disks not exported: <cell_disks>

Cause: Because of an error, the cell disk had not been exported.

Action: Contact Oracle Support Services.

CELL-04552: Cell Server (CELLSRV) could not rescan: <lun>.

Cause: Cell Server could not rescan the specified LUN.

Action: Contact Oracle Support Services.

CELL-04553: Could not import <lun> because of a preexisting cell disk named <cell_disk>.

Cause: The LUN could not be imported because there was an existing cell disk with the same name as the cell disk to be imported.

Action: Contact Oracle Support Services.

CELL-04554: Cell Server (CELLSRV) cannot successfully import the LUN: <lun>.

Cause: Cell Server could not import the specified LUN.

Action: Contact Oracle Support Services.

CELL-04555: Cell Server (CELLSRV) cannot rename imported celldisk <cell_disk>.

Cause: Cell Server could not rename the imported cell disk with the supplied cell disk name.

Action: Contact Oracle Support Services.

CELL-04556: LUN <lun> does not contain a valid cell disk.

Cause: The specified LUN did not contain a valid cell disk.

Action: Contact Oracle Support Services.

CELL-04557: LUN could not be obtained for <attribute>.

Cause: The LUN with the specified attribute could not be accessed.

Action: Contact Oracle Support Services.

CELL-04558: LUN must be used to identify the import target.

Cause: The LUN ID must be specified to identify the cell disk to be imported.

Action: Use the `LUN=`*`lun_id`* option with `IMPORT CELLDISK`.

CELL-04559: Cannot complete import of cell disk on LUN <lun>. Received Error: <error>.

Cause: An error occurred when trying to import the cell disk on the specified LUN.

Action: Contact Oracle Support Services.

CELL-04560: Cannot complete import of cell disk <cell disk>. Received Error: <error>

Cause: An error occurred when trying to import the cell disk.

Action: Contact Oracle Support Services.

CELL-04570: Internal MS error due to a mismatch of software and hardware. Contact Oracle Support Services.

Cause: Software and hardware did not match.

Action: Contact Oracle Support Services.

CELL-04571: The command <command_name> returned error string <error_string>.

Cause: An error occurred when running the command.

Action: Review the error string to determine the cause of the error. Reenter the command or Contact Oracle Support Services.

CELL-04572: Could not determine slot number for physical drive <physical_drive>.

Cause: The disk configuration tool process cannot determine the slot number for this physical drive.

Action: Contact Oracle Support Services.

CELL-04573: Could not determine slot number for <logical_drive>.

Cause: The disk configuration tool process cannot determine the slot number for this logical drive.

Action: Contact Oracle Support Services.

CELL-04574: Must use the FORCE option to reenable a LUN.

Cause: The `FORCE` option was not used when reenabling the LUN.

Action: Use the `FORCE` option with `ALTER LUN REENABLE`.

See Also: "ALTER LUN" on page 8-29

CELL-04575: Could not determine slot OR serial number for ctrl <ctrl>.

Cause: The disk configuration tool process cannot determine the slot number for this controller.

Action: Contact Oracle Support Services.

CELL-04576: The command \"{0}\" returned error string {1}.

Cause: A program error occurred.

Action: Contact Oracle Support Services.

Error Messages: CELL-05001 to CELL-05500

These error messages are in the range CELL-05001 through CELL-05500.

CELL-05001: Error code returned while executing: <error_code>

Cause: A program error occurred.

Action: Contact Oracle Support Services.

CELL-05002: Error occurred while executing: <program>

Cause: A program error occurred.

Action: Contact Oracle Support Services.

CELL-05003: Error encountered while opening or reading <file>. The content of <file>:<content>. Error: <error_code>.

Cause: A program error occurred.

Action: Contact Oracle Support Services.

CELL-05005: Illegal value for interval specified to find the average CPU queue length.

Cause: A program error occurred.

Action: Contact Oracle Support Services.

Error Messages: CELL-05501 to CELL-06000

These error messages are in the range CELL-05501 through CELL-06000.

CELL-05501: Invalid mail configuration. Please make sure smtpServer, fromAddr and toAddr are set before verifying the mail configuration. In addition, in order to get mail notifications for alerts, notificationPolicy should be set properly. A valid command could be: alter cell smtpServer='mailserver.example.com', fromAddr='yourname@example.com', toAddr='yourname@example.com', notificationPolicy=critical, notificationMethod=mail

Cause: An invalid value was entered for an e-mail configuration attribute.

Action: Set the values for the `smtpServer`, `fromAddr`, `toAddr`, `notificationMethod`, and `notificationPolicy` attributes with the `ALTER CELL` command before verifying the mail configuration.

See Also: "ALTER CELL" on page 8-15

CELL-05503: An error was detected during notification. The text of the associated internal error is: <error_text>. The notification recipient is <recipient>.

Cause: A problem occurred when processing the notification.

Action: Ensure that the recipient is valid. Check the `toAddr` cell attribute.

See Also:

- "LIST CELL" on page 8-100
- "ALTER CELL" on page 8-15

Alert Messages

This section describes the alert messages. This section contains the following topics:

- Format of Alert Messages for E-mail Notification
- Format of Alert Messages for SNMP Notification
- Threshold Alert Messages
- ADR Alert Messages
- Software Alert Messages

Format of Alert Messages for E-mail Notification

The format of an e-mail notification for an alert message is as follows:

```
Subject:
cell_name: alert level: { critical | warning | clear } alert

E-mail Content:
Alert Type: { ADR | Hardware | Threshold } Alert alert_name is triggered at
alert_time with message:
alert_message

The suggested action is:
alert_action
```

Format of Alert Messages for SNMP Notification

Simple Network Management Protocol (SNMP) alerts sent by Exadata Cells conform to a Management Information Base (MIB) which is included in each Oracle Exadata Storage Server Software installation. The MIB file on Exadata Cell is available at `/opt/oracle/cell/cellsrv/deploy/config/cell_alert.mib`. The SNMP alerts and MIB conform to SNMP version 1 (SNMPv1). The alerts contain variables, as described in Table B–1.

Table B–1 Definitions for Cell Alert SNMP Variables

Variable	Description
oraCellAlertAction	Recommended action to perform for this alert.
oraCellAlertBeginTime	Time stamp when an alert changes state.
oraCellAlertEndTime	Time stamp for the end of the period when an alert changes state.
oraCellAlertExaminedBy	Administrator who reviewed the alert.
oraCellAlertMsg	Brief explanation of the alert.
oraCellAlertNotif	Number indicating progress in notifying subscribers to alert messages: ■ 0: Never tried ■ 1: Sent successfully ■ 2: Retrying, up to five times ■ 3: Five failed retries
oraCellAlertObjectName	Object, such as cell disk or grid disk, for which a metric threshold has caused an alert.
oraCellAlertSeqBeginTime	Time stamp when an alert sequence ID is first created.
oraCellAlertSeqID	Unique sequence ID for the alert. When an alert changes state, such as from `warning` to `critical`, or `critical` to `clear`, another occurrence of the alert is created with the same sequence number and a time stamp of the transition.
oraCellAlertSeverity	Severity level. Values are `clear`, `info`, `warning`, or `critical`.
oraCellAlertShortName	Abbreviated name for the alert. If the alert is based on a metric, then the short name is the same as the corresponding metric name attribute.
oraCellAlertType	Type of the alert. Values are `stateful` or `stateless`. ■ Stateful alerts are automatically cleared on transition to normal. ■ Stateless alerts are never cleared.

See Also:

- "Receiving Alert Notifications" on page 7-19
- "DESCRIBE ALERTHISTORY" on page 8-52

Threshold Alert Messages

Threshold alerts help you monitor your database. Most alerts notify you when particular metric thresholds are exceeded. For each alert, you can set critical and warning threshold values. These threshold values are boundary values that when exceeded, indicate that the system is in an undesirable state. For example, when a tablespace becomes 97 percent full, this can be considered undesirable, and Oracle Database generates a critical alert. The following are examples of threshold alerts:

Threshold *name* triggered alert state *severity*

The threshold alert was triggered. Examine the metric value that is violating the specified threshold. Correct the problem indicated by threshold *name*.

The threshold value is no longer violated. No further action is required for threshold *name*.

The threshold alert was cleared.

ADR Alert Messages

Problems are tracked in the Automatic Diagnostic Repository (ADR). ADR is a file-based repository for storing diagnostic data. Because this repository is stored outside the database, the diagnostic data is available even when the database is down. As of Oracle Database release 11*g*, the alert log, all trace and dump files, and other diagnostic data are also stored in ADR.

Each problem has a problem key, which is a text string that describes the problem. The problem key includes the error code (such as `ORA 00600`), and in some cases, one or more error parameter values or other information. The following is an example of an ADR message:

```
Errors in file /opt/oracle/log/diag/asm/cell/stado54/trace/svtrc_2763_0.trc
 (incident=1): ORA-00600: internal error code, arguments: [main_5], [3],
[Invalid IP Param], [], [], [], [], []
```

The action to be taken for ADR messages is:

```
Create an incident package for incident <incident number> using ADRCI
and upload the incident packages to Oracle Support Services.
```

Software Alert Messages

This section describes the Exadata Cell software alerts.

SALRT-00001, MS has written its configuration file successfully.

Cause: Management Server (MS) successfully wrote the configuration file.

Action: No action is required.

SALRT-00003, The cell configuration check was successful.

Cause: The cell configuration check ran successfully.

Action: No action is required.

SALRT-00007, Hugepage allocation was successful in service CELLSRV.

Cause: Hugepage allocation was successful.

Action: No action is required.

SALRT-00008, Cell USB was restored from a failure.

Cause: The Oracle Exadata Storage Server Software CELLBOOT USB flash drive was restored.

Action: No action is required.

SALRT-00038, A SQL PLAN quarantine has been added. As a result, Smart Scan is disabled for SQL statements with the quarantined SQL plan.

Cause: A SQL statement caused the Cell Server (CELLSRV) service on the cell to crash. A SQL PLAN quarantine has been created to prevent the same SQL statement causing the same cell to crash.

This alert is also generated with warning severity if the quarantine is manually created using `CREATE QUARANTINE` command.

The disk regions associated with the fault are also quarantined, and these regions are listed in the alert message.

Action: When possible, disable offload for the SQL statement or apply the Oracle Database patch that fixes the crash. Then you can remove the quarantine with the following CellCLI command:

```
CellCLI> DROP QUARANTINE id
```

If the quarantine was manually created using the `CREATE QUARANTINE` command, then make sure the quarantine is eventually dropped to prevent potential performance degradation.

All quarantines are automatically removed when a cell is patched or upgraded.

SALRT-00045, A Database quarantine has been added. As a result, Smart Scan is disabled for the database.

Cause: This database caused instability on the cell. As a result, a quarantine has been created to prevent the same database causing further instability on the cell. All cell offload features for the database are disabled.

Instability detection is based on the number of SQL plan step quarantines for a database, and currently, this database has three SQL plan step quarantines.

Action: When possible, disable offload for the SQL statement or apply the Oracle Database patch that fixes the crash, then you can remove the Database quarantine by doing the following using CellCLI commands:

```
CellCLI> DROP QUARANTINE SQL_plan_quarantine
CellCLI> DROP QUARANTINE  database_quarantine
```

All quarantines are automatically removed when a cell is patched or upgraded.

SALRT-00046, A Cell Offload quarantine has been added. As a result, Smart Scan is disabled on the cell.

Cause: Some cell offload features appeared to cause instability on the cell. In order to avoid further cell instability, cell offload has been completely disabled for all databases. Some Oracle Exadata Storage Server Software features such as Smart Scan have been disabled. The instability detection is based on the number of database quarantines for a cell. Currently, the cell has three database quarantines.

This alert is also generated with warning severity if the quarantine is manually created using `CREATE QUARANTINE` command.

Action: When it is possible to determine that one of the databases does not cause instability, then you can remove the Cell Offload Quarantine by doing the following:

- Refer to the Database Quarantine alert or the *Oracle Exadata Storage Server Software User's Guide* for information about how to remove a database quarantine in order to drop a database quarantine.
- Run the following command:

```
CellCLI> drop quarantine cell_quarantine
```

If the quarantine was manually created using the `CREATE QUARANTINE` command, then make sure the quarantine is eventually dropped to prevent potential performance degradation.

All quarantines are automatically removed when a cell is patched or upgraded.

SALRT-00047, A SQLID quarantine has been added. As a result, Smart Scan is disabled for the SQLID.

Cause: A SQL statement caused CELLSRV on the cell to crash. A SQLID quarantine has been manually created to prevent a SQL statement with the same SQLID from causing a cell to crash. This alert is also generated with warning severity if the quarantine is manually created using the `CREATE QUARANTINE` command.

Action: When possible, disable offload for the SQL statement or apply the Oracle Database patch that fixes the crash. Then you can remove the quarantine with the following CellCLI command

```
CellCLI> DROP QUARANTINE id
```

If the quarantine was manually created using the `CREATE QUARANTINE` command, then make sure the quarantine is eventually dropped to prevent potential performance degradation.

All quarantines are automatically removed when a cell is patched or upgraded.

SALRT-00048, A DISK REGION quarantine has been added. As a result, Smart Scan is disabled for the quarantined physical disk region.

Cause: This alert is generated with warning severity when a DISK REGION quarantine is manually created using `CREATE QUARANTINE` command.

Action: If the quarantine was manually created using the `CREATE QUARANTINE` command, then make sure the quarantine is eventually dropped to prevent potential performance degradation.

All quarantines are automatically removed when a cell is patched or upgraded.

SALRT-00050, A quarantine has been removed.

Cause: The quarantine was dropped by `DROP QUARANTINE` command, or was automatically dropped due to software upgrade.

Action: This message is informational. No action is needed.

SALRT-00052: A quarantine has been automatically removed due to software upgrade.

Cause: The quarantine was automatically dropped due to a software upgrade.

Action: This message is informational. No action is needed.

C

Installation Information

This appendix contains information about Oracle Exadata Storage Server Software (Exadata Cell) installation. Refer to this installation information if the Exadata Cell hardware and software are not already installed.

This appendix contains the following topics:

- Installation of Exadata Cell
- Background Processes in the Exadata Cell Environment

> **See Also:** Chapter 2, "Configuring Oracle Exadata Storage Server Software" for additional information about configuring cells, database instances, Oracle ASM instances, and Oracle ASM disk groups

Installation of Exadata Cell

This section discusses the installation of a cell, and contains the following topics:

- Installing Exadata Cell Hardware
- Installing the Exadata Cell Software on a Cell
- Updates Made During Installation

Installing Exadata Cell Hardware

This section provides a summary of the hardware installation necessary for standalone Exadata Cells connecting to Oracle Exadata Database Machine.

When you receive your cell devices, perform the following procedure:

1. Unpack, situate, and set up the cell devices.
2. Read the printed vendor documentation supplied with the cell devices.
3. Connect the management and ILOM Ethernet ports to the management network.
4. Connect the InfiniBand ports of the Exadata Cell to the InfiniBand network of Oracle Exadata Database Machine.
5. Attach a computer or terminal to one of the new cell devices so you can configure the cell as described in "Configuring Oracle Exadata Storage Server Software for Your Location" on page 2-9. You must configure each new cell.
6. Connect the power supply on the new cell devices.
7. Power on the cell devices.
8. Assign IP addresses for the network interconnections of the new cells.

See Also: "Assigning IP Addresses for Oracle Exadata Storage Server Software" on page 2-8

Installing the Exadata Cell Software on a Cell

Perform the following procedure to install the Exadata Cell software:

1. Power on the cell to boot its operating system.
2. Log in to the cell with super user (`root`) privileges. The password is `welcome1`.
3. Change the `root` password on the cell.
4. Verify that the cell meets the software requirements.
5. Copy the self-extracting binary to the cell, if necessary.

   You can use the `ftp` command to copy the binary to the cell from a network location, or you can copy the binary from a portable storage device.
6. Install the self-extracting binary on the cell if it is not already installed using the following command:

   ```
   ./cell_package_version.bin
   ```
7. Review the `/opt/oracle/cell/.install_log.txt` file to verify that the installation completed correctly. The installation process adds two new users, `celladmin` and `cellmonitor`. The `celladmin` user should be used to run all services on the cell.The `cellmonitor` user is for monitoring purposes.
8. After successful installation of the Exadata Cell software, follow the instructions at "Preparing the Servers" on page 2-10.

Updates Made During Installation

Changes that are made to `snmpd` files during installation include the following:

- Updates to the public community string
- New entries for `trapdestination`
- Updates to the kernel configuration files (limits for `aio`, `udprecv`, and number of open files)

Background Processes in the Exadata Cell Environment

The background processes for the database and Oracle ASM instance for an Exadata Cell environment are the same as other environments, except for the following background process:

- diskmon Process
- XDMG Process
- XDWK Process

diskmon Process

The `diskmon` process is a fundamental component of Exadata Cell, and is responsible for implementing I/O fencing. The process is located on the database server host computer, and is part of Oracle Clusterware Cluster Ready Services (Oracle Clusterware CRS). This process is important for Exadata Cell and should not be modified.

The log files for `diskmon` are located in the `$CRS_HOME/log/hostname/diskmon` directory.

See Also:

- *Oracle Clusterware Administration and Deployment Guide* for additional information about the following:

  - Oracle Clusterware CRS, including troubleshooting Oracle Clusterware and the location of clusterware log files

  - Oracle Clusterware cluster subcomponent processes and background processes
- *Oracle Automatic Storage Management Administrator's Guide* for additional information about Oracle ASM background processes
- *Oracle Database Concepts* for additional information about Oracle Database background processes
- *Oracle Database Reference* for a description of the `V$BGPROCESS` view that displays information about background processes

XDMG Process

The `XDMG` (Exadata Automation Manager) process initiates automation tasks used for monitoring storage. This background process monitors all configured Exadata Cells for state changes, such as replaced disks, and performs the required tasks for such changes. Its primary task is to watch for inaccessible disks and cells, and to detect when the disks and cells become accessible. When the disks and cells are accessible, the `XDMG` process initiates the `ASM ONLINE` process, which is handled by the `XDWK` background process. The `XDMG` process runs in the Oracle ASM instances.

XDWK Process

The `XDWK` (Exadata Automation Worker) process performs automation tasks by requested by the `XDMG` background process. The `XDWK` process begins when asynchronous actions, such as `ONLINE`, `DROP` or `ADD` for an Oracle ASM disk are requested by the `XDMG` process. The `XDWK` process stops after 5 minutes of inactivity. The `XDWK` process runs in the Oracle ASM instances.

Glossary

cell number

An attribute of a cell. The cell number is used with the **IP block** attribute to compute the range of IP addresses and the netmask assigned to the cell.

Cell Server (CELLSRV)

See **CELLSRV**.

CELLSRV

Cell Server, a block server that runs on the cell and provides a read/write application programming interface (API) for Exadata Cell and scheduling of I/O on the disks.

client application

Any application that directly uses the storage server to store persistent data, such as Oracle Database.

database administrator

A user responsible for managing individual databases. In larger deployments, this responsibility is separate from storage administration, but database administrators and storage administrators cooperate in managing the storage-related aspects of application support.

database resource plan

A resource plan that specifies how CPU and I/O resources are allocated among consumer groups in its database. The plan is created using PL/SQL commands.

host

A computer that runs client applications and databases. It is connected to cells by some form of network, such as Ethernet.

interdatabase plan

Interdatabase resource plan manages multiple databases. The plan is created using the CellCLI utility.

intradatabase plan

See **database resource plan**.

I/O Resource Management (IORM)

See IORM.

IORM

I/O Resource Management allows workloads and databases to share Exadata Cells according to user-defined policies.

IP block

An attribute of a cell. The IP block is used with the **cell number** attribute to compute the range of IP addresses and the netmask assigned to the cell.

KVM switch

Keyboard, video, and mouse switch is a hardware device that enables a user to control multiple computers from a single keyboard, video monitor, and mouse.

Management Server (MS)

See **MS**.

MS

Management Server, an application server running on a cell device and exposing basic hardware configuration and management.

Network Interface Card (NIC)

See **NIC**

NIC

Network interface card is a circuit board or card in a computer that enables computers to communicate over a network.

Oracle ASM

Oracle Automatic Storage Management is the Oracle volume manager that supports software striping and mirroring of Oracle Database data using multiple disks.

Oracle Automatic Storage Management (Oracle ASM)

See **Oracle ASM**.

resource consumer groups

A group that categorizes user sessions according to resource requirements. Resource consumer groups are different from user roles. One database user can have different sessions assigned to different resource consumer groups. Sessions are mapped to resource consumer groups through a series of hierarchical mapping rules. For additional information about resource consumer groups, see *Oracle Database Administrator's Guide*.

resource plan

A plan that specifies how CPU and I/O resources are to be distributed among various users (**resource consumer groups**). For additional information about resource plans, see *Oracle Database Administrator's Guide*.

Restart Server (RS)

See **RS**.

RS

Restart Server is a server that restarts the CELLSRV and MS services, and monitors these services to verify when they need to be restarted.

Secure Shell (SSH)

See **SSH**.

SSH

Secure Shell is a network protocol that allows data to be exchanged over a secure channel between two computers.

storage pool

A homogeneous collection of storage resources managed as a unit that supports multiple consumers, for example, databases.

storage provisioning

Dynamic or static assignment of disk storage to the clients that need it. For example, a given database may need more storage for its data files, and storage servers can automatically provision the space. Alternatively, additional storage can be explicitly assigned by the storage administrator.

storage realm

A collection of cells and attached grid disks that are associated with a group of Oracle ASM instances in a cluster. The clustered Oracle ASM instances manage the related set of disk groups.

Index

Symbols

A

B

C

D

E

F

K

L

M

N

O

P

Q

R

X