Oracle® Exadata System Software

User's Guide

24.1
F29254-32
May 2024

ORACLE

Oracle Exadata System Software User's Guide, 24.1

F29254-32

Primary Author: Peter Fusek

Contributors: Ravikiran Akkayajhula, Andrew Babb, Nilesh Choudhury, Siddhartha Datta, Boris Erlikhman, Jaime Figueroa, Cecilia Gervasio Grant, Roger Hansen, Mark Hollinger, Iori Honda, Yiliang Jin, Rakesh Kashyap, Frank Kobylanski, Kishy Kumar, Yang Liu, Juan Loaiza, Scott Martin, Krishnan Meiyyappan, Michael Nowak, Dmitry Potapov, Darryl Presley, Ashish Ray, Akshay Shah, Vivek S. Sharma, Jia Shi, Kesavan Srinivasan, Mahesh Subramaniam, Alex Tsukerman, Shreyas Udgaonkar, Kothanda Umamageswaran, Doug Utzig, Zheren Zhang, Alex Blyth

Contents

Preface

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

Audience

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

- Configure Oracle Exadata System Software
- Manage Oracle Exadata System Software
- Troubleshoot Oracle Exadata System Software

Documentation Accessibility

Diversity and Inclusion

we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

For additional information, see the following Oracle resources:

- *Oracle Exadata Database Machine System Overview*
- *Oracle Exadata Database Machine Installation and Configuration Guide*
- *Oracle Exadata Database Machine Maintenance Guide*
- *Oracle Exadata Database Machine Extending and Multi-Rack Cabling Guide*
- *Oracle Exadata Database Machine Security 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 Reference*
- *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.

1

Introducing Oracle Exadata System Software

This chapter introduces Oracle Exadata System Software.

- Overview of Oracle Exadata System Software
  Oracle Exadata Storage Server is a highly optimized storage server that runs Oracle Exadata System Software to store and access Oracle Database data.
- Key Features of Oracle Exadata System Software
  This section describes the key features of Oracle Exadata System Software.
- Oracle Exadata System Software Components
  This section provides a summary of the following Oracle Exadata System Software components.

1.1 Overview of Oracle Exadata System Software

Oracle Exadata Storage Server is a highly optimized storage server that runs Oracle Exadata System Software to store and access Oracle Database data.

With traditional storage, data is transferred to the database server for processing. In contrast, Oracle Exadata System Software 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. Oracle Exadata storage servers process data at the storage level, and pass only what is needed to the database servers.

Oracle Exadata System Software is installed on both the storage servers and the database servers. Oracle Exadata System Software offloads some SQL processing from the database server to the storage servers. Oracle Exadata System Software 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 Oracle Exadata System Software is optimized to take maximum advantage of the hardware components. Each storage server delivers outstanding processing bandwidth for data stored on disk, often several times better than traditional solutions.

Oracle Exadata storage servers use state-of-the-art RDMA Network Fabric interconnections between servers and storage. Each RDMA Network Fabric link provides bandwidth of 40 Gb/s for InfiniBand Network Fabric or 100 Gb/s for RoCE Network Fabric. Additionally, the 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 RDMA Network Fabric has the flexibility of a LAN network with the efficiency of a storage area network (SAN). With an RDMA Network Fabric network, Oracle Exadata eliminates network bottlenecks that could reduce performance. This RDMA Network

Fabric network also provides a high performance cluster interconnection for Oracle Real Application Clusters (Oracle RAC) servers.

There are several configurations of storage server each configured to maximize some aspect of storage based on your requirements. Each storage server contains persistent storage media, which may be a mixture of hard disk drives (HDDs) and flash devices. High Capacity (HC) storage servers contain HDDs for primary data storage and high-performance flash memory, which is mainly used for caching purposes. Extreme Flash (EF) storage servers are geared towards maximum performance and have an all-flash storage configuration. Extended (XT) storage servers are geared towards extended-capacity applications, such as online data archiving, and contain HDDs only. HC and EF storage servers are also equipped with additional memory for Exadata RDMA Memory Cache (XRMEM cache), which supports high-performance data access using Remote Direct Memory Access (RDMA).

The Oracle Exadata architecture scales to any level of performance. To achieve higher performance or greater storage capacity, you add more storage servers (cells) to the configuration. As more storage servers are added, capacity and performance increase linearly. Data is mirrored across storage servers to ensure that the failure of a storage server 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 System Software must be used with Oracle Exadata storage server hardware, and only supports Oracle databases on the Oracle Exadata database servers. 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/technetwork/index.html`

1.2 Key Features of Oracle Exadata System Software

This section describes the key features of Oracle Exadata System Software.

- Reliability, Modularity, and Cost-Effectiveness
  Oracle Exadata Storage Server employs cost-effective modular storage hardware in a scale-out architecture enabling high availability and reliability.
- Compatibility with Oracle Database
  When the minimum required versions are met, all Oracle Database features are fully supported with Oracle Exadata System Software.
- Smart Flash Technology
  The Exadata Smart Flash Cache feature of the Oracle Exadata System Software intelligently caches database objects in flash memory, replacing slow, mechanical I/O operations to disk with very rapid flash memory operations.
- Persistent Memory Accelerator and RDMA
  Persistent Memory (PMEM) Accelerator provides direct access to persistent memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies.

- Exadata RDMA Memory
  Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM incorporates all of the Exadata capabilities that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies.
- Centralized Storage
  You can use Oracle Exadata to consolidate your storage requirements into a central pool that can be used by multiple databases.
- I/O Resource Management (IORM)
  I/O Resource Management (IORM) and the Oracle Database Resource Manager enable multiple databases and pluggable databases to share the same storage while ensuring that I/O resources are allocated across the various databases.
- Exadata Hybrid Columnar Compression
  Exadata Hybrid Columnar Compression stores data using column organization, which brings similar values close together and enhances compression ratios.
- In-Memory Columnar Format Support
  In an Oracle Exadata environment, the data is automatically stored in In-Memory columnar format in the flash cache when it will improve performance.
- Offloading Data Search and Retrieval Processing
  Exadata Smart Scan offloads search and retrieval processing to the storage servers.
- Offloading of Incremental Backup Processing
  To optimize the performance of incremental backups, the database can offload block filtering to Oracle Exadata Storage Server.
- Fault Isolation with Quarantine
  Oracle Exadata System Software has the ability to learn from the past events to avoid errors.
- Protection Against Data Corruption
  Data corruptions, while rare, can have a catastrophic effect on a database, and therefore on a business.
- Fast File Creation
  File creation operations are offloaded to Oracle Exadata Storage Servers.
- Storage Index
  Oracle Exadata Storage Servers maintain a storage index which contains a summary of the data distribution on the disk.

1.2.1 Reliability, Modularity, and Cost-Effectiveness

Oracle Exadata Storage Server employs cost-effective modular storage hardware in a scale-out architecture enabling high availability and reliability.

Oracle Exadata Storage Server is architected to eliminate single points of failure by mirroring data, employing fault isolation technologies, and protecting against disk and other storage hardware failures. Even brownouts are limited or eliminated when failures occur.

In the Oracle Exadata Storage Server architecture, multiple storage servers provide a pool of storage to support one or more databases. Intelligent software automatically and transparently distributes data evenly across the storage servers. Oracle Exadata Storage Server supports dynamic disk replacement, and Oracle Exadata System Software provides online dynamic data redistribution, which ensures that data is balanced across the storage without interrupting database processing. Oracle Exadata Storage Server also provides data protection from disk and storage server failures.

1.2.2 Compatibility with Oracle Database

When the minimum required versions are met, all Oracle Database features are fully supported with Oracle Exadata System Software.

Oracle Exadata System Software works equally well with single-instance or Oracle Real Application Clusters (Oracle RAC) deployments of Oracle Database. Oracle Data Guard, Oracle Recovery Manager (RMAN), Oracle GoldenGate, and other database features are managed the same with Exadata storage cells as with traditional storage. This enables database administrators to use the same tools with which they are familiar.

Refer to My Oracle Support Doc ID 888828.1 for a complete list of the minimum required software versions.

Related Topics

- Exadata Database Machine and Exadata Storage Server Supported Versions (My Oracle Support Doc ID 888828.1)

1.2.3 Smart Flash Technology

The Exadata Smart Flash Cache feature of the Oracle Exadata System Software intelligently caches database objects in flash memory, replacing slow, mechanical I/O operations to disk with very rapid flash memory operations.

- Exadata Smart Flash Cache
  Exadata Smart Flash Cache stores frequently accessed data in high-performance flash storage.
- Write-Back Flash Cache
  Write-Back Flash Cache enables write I/Os directly to Exadata Smart Flash Cache.
- Exadata Smart Flash Log
  Exadata Smart Flash Log improves transaction response times and increases overall database throughput for I/O intensive workloads by accelerating performance-critical log write operations.

1.2.3.1 Exadata Smart Flash Cache

Exadata Smart Flash Cache stores frequently accessed data in high-performance flash storage.

Exadata Smart Flash Cache automatically works in conjunction with Oracle Database to intelligently optimize cache efficiency by favoring frequently accessed and high-value data. Each database I/O contains a tag that indicates the likelihood of repeated data access. This information is combined with internal statistics and other measures, such as the object size and access frequency, to determine whether to cache the data.

Just as importantly, Exadata Smart Flash Cache avoids caching data that will never be reused or will not fit into the cache. For example, a backup operation does not read data repeatedly, so backup-related I/O is not cached.

By default, caching occurs automatically and requires no user or administrator effort.

Although it is generally not required or recommended, Oracle Exadata System Software also enables administrators to override the default caching policy and keep specific table and index segments in or out of the cache.

Originally, Exadata Smart Flash Cache operated exclusively in Write-Through mode. In Write-Through mode, database writes go to disk first and subsequently populate Flash Cache. If a flash device fails with Exadata Smart Flash Cache operating in Write-Through mode, there is no data loss because the data is already on disk.

1.2.3.2 Write-Back Flash Cache

Write-Back Flash Cache enables write I/Os directly to Exadata Smart Flash Cache.

With Exadata Smart Flash Cache in Write-Back mode, database writes go to Flash Cache first and later to disk. Write-Back mode was introduced with Oracle Exadata System Software release 11.2.3.2.0.

Write-intensive applications can benefit significantly from Write-Back mode by taking advantage of the fast latencies provided by flash. If your application writes intensively and if you experience high I/O latency or significant waits for `free buffer waits`, then you should consider using Write-Back Flash Cache.

With Exadata Smart Flash Cache in Write-Back mode, the total amount of disk I/O also reduces when the cache absorbs multiple writes to the same block before writing it to disk. The saved I/O bandwidth can be used to increase the application throughput or service other workloads.

However, if a flash device fails while using Write-Back mode, data that is not yet persistent to disk is lost and must be recovered from a mirror copy. For this reason, Write-Back mode is recommended in conjunction with using high redundancy (triple mirroring) to protect the database files.

The contents of the Write-Back Flash Cache is persisted across reboots, eliminating any warm-up time needed to populate the cache.

1.2.3.3 Exadata Smart Flash Log

Exadata Smart Flash Log improves transaction response times and increases overall database throughput for I/O intensive workloads by accelerating performance-critical log write operations.

The time to commit user transactions is very sensitive to the latency of log write operations. In addition, many performance-critical database algorithms, such as space management and index splits, are very sensitive to log write latency.

Although the disk controller has a large battery-backed DRAM cache that can accept writes very quickly, some write operations to disk can still be slow during busy periods when the disk controller cache is occasionally filled with blocks that have not been written to disk. Noticeable performance issues can arise even with relatively few slow redo log write operations.

Exadata Smart Flash Log reduces the average latency for performance-sensitive redo log write I/O operations, thereby eliminating performance bottlenecks that may occur due to slow redo log writes. Exadata Smart Flash Log removes latency spikes by simultaneously performing redo log writes to two media devices. The redo write is acknowledged as soon as the first write to either media device completes.

Prior to Oracle Exadata System Software release 20.1, Exadata Smart Flash Log performs simultaneous writes to disk and flash storage. With this configuration, Exadata Smart Flash Log improves average log write latency and increases overall database throughput. But, because all log writes must eventually persist to disk, this configuration is limited by the overall disk throughput, and provides little relief for applications that are constrained by disk throughput.

Oracle Exadata System Software release 20.1 adds a further optimization, known as Smart Flash Log Write-Back, that uses Exadata Smart Flash Cache in Write-Back mode instead of disk storage. With this configuration, Exadata Smart Flash Log improves average log write latency and overall log write throughput to eliminate logging bottlenecks for demanding throughput-intensive applications.

1.2.4 Persistent Memory Accelerator and RDMA

Persistent Memory (PMEM) Accelerator provides direct access to persistent memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies.

Note:

PMEM is only available in Exadata X8M-2 and X9M-2 storage server models with High Capacity (HC) or Extreme Flash (EF) storage.

Starting with Oracle Exadata System Software release 19.3.0, workloads that require ultra low response time, such as stock trades and IOT devices, can take advantage of PMEM and RDMA in the form of a PMEM Cache and PMEM Log.

When database clients read from the PMEM cache, the client software performs an RDMA read of the cached data, which bypasses the storage server software and delivers results much faster than Exadata Smart Flash Cache.

PMEM cache works in conjunction with Exadata Smart Flash Cache. The following table describes the supported caching mode combinations when PMEM Cache is configured:

PMEM Cache Mode	Flash Cache Mode	Supported Configuration?
Write-Through	Write-Through	Yes. This is the default configuration for High Capacity (HC) servers with Normal Redundancy.
Write-Through	Write-Back	Yes. This is the default configuration for HC servers with High Redundancy. This is also the default configuration for Extreme Flash (EF) servers.
Write-Back	Write-Back	Yes. However, the best practice recommendation is to configure PMEM cache in write-through mode. This configuration provides the best performance and availability. **Note:** Commencing with Oracle Exadata System Software release 23.1.0, PMEM cache only operates in write-through mode.

PMEM Cache Mode	Flash Cache Mode	Supported Configuration?
Write-Back	Write-Through	No. Without the backing of Write-Back Flash Cache, write-intensive workloads can overload the PMEM Cache in Write-Back mode.

If PMEM Cache is not configured, Exadata Smart Flash Cache is supported in both Write-Back and Write-Through modes.

Redo log writes are critical database operations and need to complete in a timely manner to prevent load spikes or stalls. Exadata Smart Flash Log is designed to prevent redo write latency outliers. PMEM Log helps to further reduce redo log write latency by using PMEM and RDMA.

With PMEM Log, database clients send redo log I/O buffers directly to PMEM on the storage servers using RDMA, thereby reducing transport latency. The cell server (`cellsrv`) then writes the redo to Exadata Smart Flash Log (if enabled) and disk at a later time.

Reduced redo log write latency improves OLTP performance, resulting in higher transaction throughput. In cases where PMEM Log is bypassed, Exadata Smart Flash Log can still be used.

1.2.5 Exadata RDMA Memory

Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM incorporates all of the Exadata capabilities that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies.

XRMEM encompasses previous Exadata data and commit accelerators based on persistent memory (PMEM), which is only available in Exadata X8M and X9M storage server models. Starting with Exadata X10M, XRMEM enables the benefits of RDMA without needing specialized persistent memory and is primed to leverage developments in memory and storage hardware.

On Exadata X10M systems, workloads requiring ultra-low response times, such as stock trades and IOT devices, can take advantage of XRMEM cache. When database clients read from XRMEM cache, the client software performs an RDMA read of the cached data, which bypasses the storage server software and networking layers, eliminates expensive CPU interrupts and context switches, and delivers results much faster than Exadata Smart Flash Cache. In this case, XRMEM cache operates only in write-through mode, with database writes saved to persistent storage.

On Exadata X10M, XRMEM cache is available on High Capacity (HC) and Extreme Flash (EF) Exadata X10M storage servers, leveraging the high-performance dynamic random-access memory (DRAM) available on the servers. In this environment, XRMEM cache functions automatically, requiring no separate configuration or ongoing administration.

On existing Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory data accelerator, also known as PMEM cache (or `PMEMCACHE`), is now called the XRMEM cache (or `XRMEMCACHE`). Likewise, the persistent memory commit accelerator, also known as PMEM log (or `PMEMLOG`), is now XRMEM log (or `XRMEMLOG`). However, the CellCLI commands to manage `PMEMCACHE` and `PMEMLOG` resources are still available for backward compatibility.

Oracle Exadata System Software release 23.1.0 does not implement `XRMEMLOG` on Exadata X10M systems.

1.2.6 Centralized Storage

You can use Oracle Exadata to consolidate your storage requirements into a central pool that can be used by multiple databases.

Oracle Exadata evenly distributes the data and I/O load for every database across available disks in the Exadata storage servers. Every database can use all of the available disks to achieve superior I/O rates. Oracle Exadata provides higher efficiency and performance at a lower cost while also lowering your storage administration overhead.

1.2.7 I/O Resource Management (IORM)

I/O Resource Management (IORM) and the Oracle Database Resource Manager enable multiple databases and pluggable databases to share the same storage while ensuring that I/O resources are allocated across the various databases.

Oracle Exadata System Software works with IORM and Oracle Database Resource Manager to ensure that customer-defined policies are met, even when multiple databases share the same set of storage servers. 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 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 and flash I/O bandwidth than throughput-sensitive batch applications. Oracle Database Resource Manager enables the administrator to control processor utilization on the database host on a per-application basis. Combining IORM and Oracle Database Resource Manager enables the administrator to establish more accurate policies.

IORM also manages the space utilization for Exadata Smart Flash Cache and Exadata RDMA Memory Cache (XRMEM cache). Critical OLTP workloads can be guaranteed space in Exadata Smart Flash Cache or XRMEM cache to provide consistent performance.

IORM for a database or pluggable database (PDB) is implemented and managed from the Oracle Database Resource Manager. Oracle 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.

Related Topics

- Managing I/O Resources

1.2.8 Exadata Hybrid Columnar Compression

Exadata Hybrid Columnar Compression stores data using column organization, which brings similar values close together and enhances compression ratios.

Using Exadata Hybrid Columnar Compression, data is organized into sets of rows called compression units. Within a compression unit, data is organized by column and then compressed. Each row is self-contained within a compression unit.

Database operations work transparently against compressed objects, so no application changes are required. The database compresses data manipulated by any SQL operation, although compression levels are higher for direct path loads.

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.
- Archive compression: This type of compression is optimized for maximum compression levels, and is intended for historic 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 the following table.

Table 1-1 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
1005	01-JUN-07	1	WAREHOUSE2	N

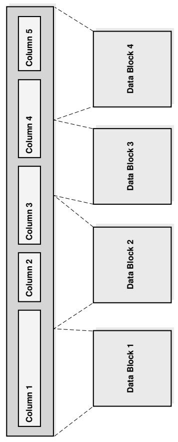
The database stores a set of rows in an internal structure called a compression unit. For example, assume that the rows in the previous table are stored in one unit. Exadata Hybrid Columnar Compression stores each unique value from column 4 with metadata that maps the values to the rows. Conceptually, the compressed value can be represented as:

```
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 the following illustration, each compression unit can span multiple data blocks. The values for a particular column may or may not span multiple blocks.

Figure 1-1   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.

Table 1-2   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 (compression level depends on compression level specified (LOW or HIGH))	Higher (CPU overhead depends on compression level specified (LOW or HIGH))	DSS
Archive compression	Highest (compression level depends on compression level specified (LOW or HIGH))	Highest (CPU overhead depends on compression level specified (LOW or HIGH))	Archiving

Warehouse compression and archive 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.

The following table lists characteristics of each table compression method.

Table 1-3 Table Compression Characteristics

Table Compression Method	CREATE/ALTER TABLE Syntax	Direct-Path Insert	DML
Basic compression	`COMPRESS [BASIC]` `COMPRESS` and `COMPRESS BASIC` are equivalent	Yes	Yes **Note:** Inserted and updated rows are uncompressed.
OLTP compression	`COMPRESS FOR OLTP`	Yes	Yes
Warehouse compression	`COMPRESS FOR QUERY [LOW|HIGH]`	Yes	Yes High CPU overhead. **Note:** Inserted and updated rows go to a block with a less compressed format and have lower compression level.
Archive compression	`COMPRESS FOR ARCHIVE [LOW|HIGH]`	Yes	Yes **Note:** Inserted and updated rows are uncompressed. 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 archive 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.

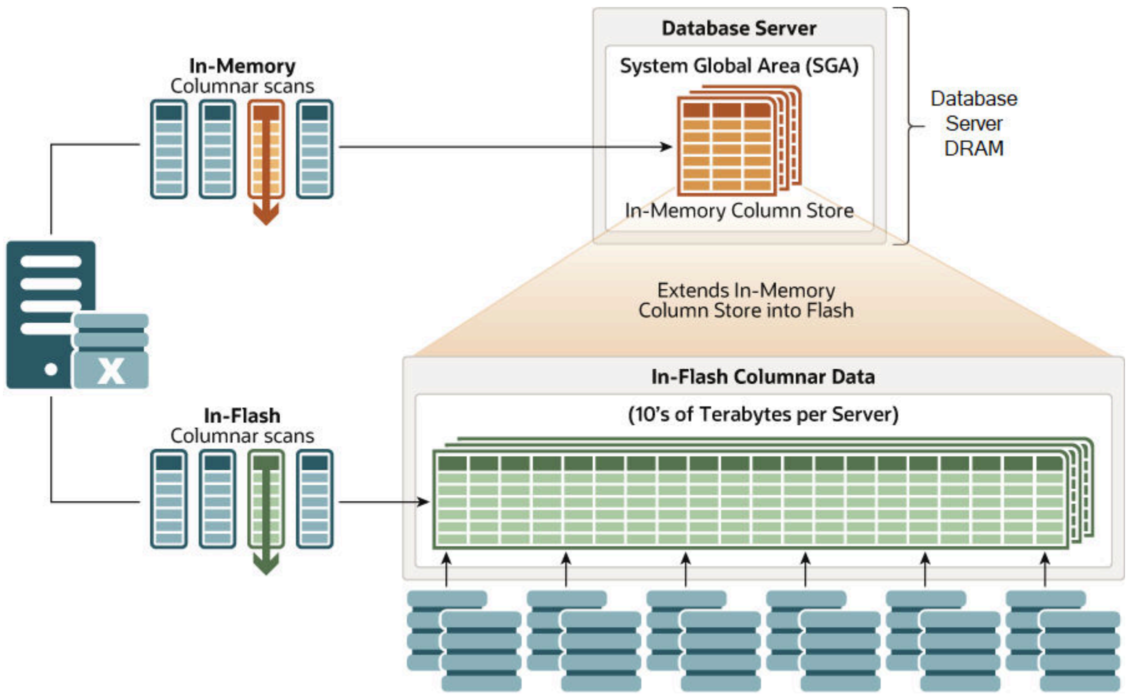
You specify table compression with the `COMPRESS` clause of the `CREATE TABLE` command. You can enable compression for an existing table by using these clauses in an `ALTER TABLE` statement. In this case, only data that is inserted or updated is compressed after compression is enabled. Similarly, you 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.

1.2.9 In-Memory Columnar Format Support

In an Oracle Exadata environment, the data is automatically stored in In-Memory columnar format in the flash cache when it will improve performance.

Oracle Exadata supports all of the In-Memory optimizations, such as accessing only the compressed columns required, SIMD vector processing, storage indexes, and so on.

If you set the `INMEMORY_SIZE` database initialization parameter to a non-zero value (requires the Oracle Database In-Memory option), then objects accessed using a Smart Scan are brought into the flash cache and are automatically converted into the In-Memory columnar format. The data is converted initially into a columnar cache format, which is different from Oracle Database In-Memory's columnar format. The data is rewritten in the background into Oracle Database In-Memory columnar format. As a result, all subsequent accesses to the data benefit from all of the In-Memory optimizations when that data is retrieved from the flash cache.



Any write to an in-memory table does not invalidate the entire columnar cache of that table. It only invalidates the columnar cache unit of the disk region in which the block resides. For subsequent scans after a table update, a large part of the table is still in the columnar cache. The scans can still make use of the columnar cache, except for the units in which the writes were made. For those units, the query uses the original block version to get the data. After a sufficient number of scans, the invalidated columnar cache units are automatically repopulated in the columnar format.

A new segment-level attribute, `CELLMEMORY`, has also been introduced to help control which objects should not be populated into flash using the In-Memory columnar format and which type of compression should be used. Just like the `INMEMORY` attribute, you can specify different compression levels as sub-clauses to the `CELLMEMORY` attribute. However, not all of the `INMEMORY` compression levels are available; only `MEMCOMPRESS FOR QUERY LOW` and

`MEMCOMPRESS FOR CAPACITY LOW` (default). You specify the `CELLMEMORY` attribute using a SQL command, such as the following:

```
ALTER TABLE trades CELLMEMORY MEMCOMPRESS FOR QUERY LOW
```

The `PRIORTY` sub-clause available with Oracle Database In-Memory is not available on Oracle Exadata because the process of populating the flash cache on Exadata storage servers if different from populating DRAM in the In-Memory column store on Oracle Database servers.

1.2.10 Offloading Data Search and Retrieval Processing

Exadata Smart Scan offloads search and retrieval processing to the storage servers.

Smart Scan performs selected Oracle Database functions inside Oracle Exadata Storage Server. This capability improves query performance by minimizing the amount of database server I/O, which reduces the amount of I/O-related communication between the database servers and storage servers. Furthermore, the database server CPU saved by Smart Scan is available to boost overall system throughput.

Smart Scan automatically optimizes full table scans, fast full index scans, and fast full bitmap index scans that use the Direct Path Read mechanism, namely parallel operations and large sequential scans. The primary functions performed by Smart Scan inside Oracle Exadata Storage Server are:

- **Predicate Filtering**

  Rather than transporting all rows to the database server for predicate evaluation, Smart Scan predicate filtering ensures that the database server receives only the rows matching the query criteria. The supported conditional operators include `=`, `!=`, `<`, `>`, `<=`, `>=`, `IS [NOT] NULL`, `LIKE`, `[NOT] BETWEEN`, `[NOT] IN`, `EXISTS`, `IS OF type`, `NOT`, `AND`, and `OR`. In addition, Exadata Storage Server evaluates most common SQL functions during predicate filtering.

- **Column Filtering**

  Rather than transporting entire rows to the database server, Smart Scan column filtering ensures that the database server receives only the requested columns. For tables with many columns, or columns containing LOBs, the I/O bandwidth saved by column filtering can be very substantial.

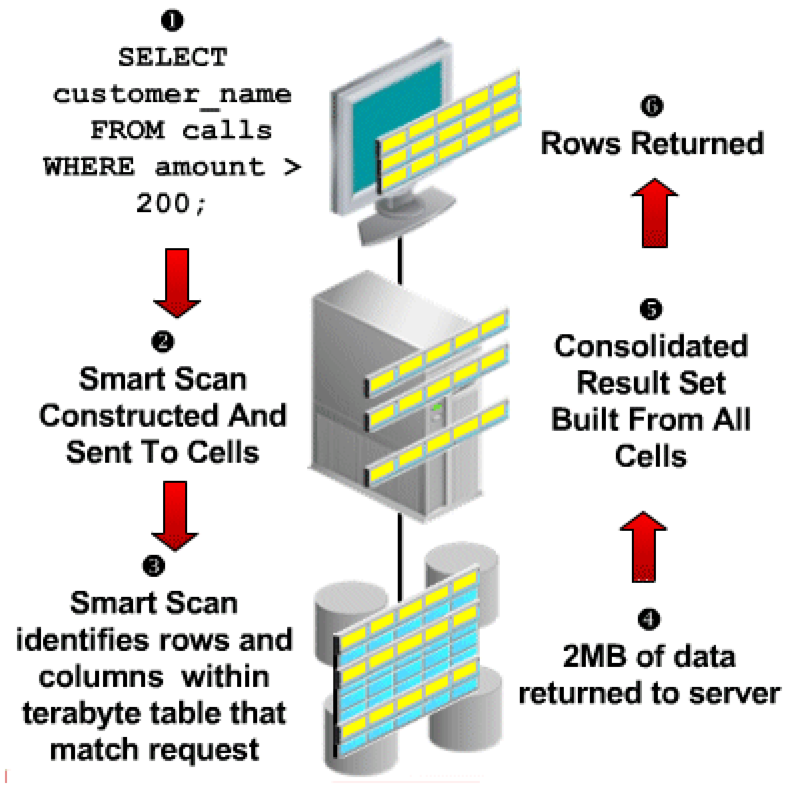
For example, consider a simple query:

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

In this case, Smart Scan offloads predicate filtering (`WHERE amount > 200`) and column filtering (`SELECT customer_name`) to the Exadata storage servers. The effect can be dramatic depending on the table size, structure, and contents. For example, if the table contains 1 TB of data, but the query result is only 2 MB, then only 2MB of data is transported from the storage servers and the database server.

The following diagram illustrates how Smart Scan avoids unnecessary data transfer between the storage servers and the database server.

Figure 1-2   Offloading Data Search and Retrieval


❶
SELECT customer_name FROM calls WHERE amount > 200;
❷
Smart Scan Constructed And Sent To Cells
❸
Smart Scan identifies rows and columns within terabyte table that match request
❹
2MB of data returned to server
❺
Consolidated Result Set Built From All Cells
❻
Rows Returned

In addition to offloading predicate filtering and column filtering, Smart Scan enables:

- Optimized join processing for star schemas (between large tables and small lookup tables). This is implemented using Bloom Filters, which provide a very efficient probabilistic method to determine whether an element is a member of a set.
- Optimized scans on encrypted tablespaces and encrypted columns. For encrypted tablespaces, Exadata Storage Server can decrypt blocks and return the decrypted blocks to Oracle Database, or it can perform row and column filtering on encrypted data. Significant CPU savings occur within the database server by offloading the CPU-intensive decryption task to Exadata cells.
- Optimized scans on compressed data. Smart Scan works in conjunction with Hybrid Columnar Compression so that column projection, row filtering, and decompression run in the Exadata Storage Servers to save CPU cycles on the database servers.
- Offloading of scoring functions, such as `PREDICTION_PROBABILITY`, for data mining models. This accelerates analysis while reducing database server CPU consumption and the I/O load between the database server and storage servers.

1.2.11 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, Oracle Exadata System Software 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.

Related Topics

- Making Incremental Backups: Quick Start

1.2.12 Fault Isolation with Quarantine

Oracle Exadata System Software has the ability to learn from the past events to avoid errors.

When a faulty SQL statement caused a crash of the server in the past, Oracle Exadata System Software quarantines the SQL statement so that when the faulty SQL statement occurs again, Oracle Exadata System 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 System 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 System 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 System 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 System 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.
- Intra-Database Plan: Created when Oracle Exadata System Software crashes while processing an intra-database resource plan. Consequently, the intra-database resource plan is quarantined and not enforced. Other intra-database resource plans in the same database are still enforced. Intra-database resource plans in other databases are not affected.
- Inter-Database Plan: Created when Oracle Exadata System Software crashes while processing an inter-database resource plan. Consequently, the inter-database resource plan is quarantined and not enforced. Other inter-database resource plans are still enforced.
- I/O Resource Management (IORM): Created when Oracle Exadata System Software crashes in the I/O processing code path. IORM is effectively disabled by setting the IORM objective to `basic` and all resource plans are ignored.
- Cell-to-Cell Offload: See "Quarantine Manager Support for Cell-to-Cell Offload Operations".

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.

- Quarantine Manager Support for Cell-to-Cell Offload Operations

Related Topics

- Using the CellCLI Utility
  You use the Cell Control Command-Line Interface (CellCLI) utility to manage Oracle Exadata System Software.
- Quarantine Manager Support for Cell-to-Cell Offload Operations
- *Oracle Database Administrator's Guide*

1.2.12.1 Quarantine Manager Support for Cell-to-Cell Offload Operations

Commencing with Oracle Exadata System Software 12.2.1.1.0, quarantine manager support is provided for cell-to-cell offload operations that support ASM rebalance and high throughput writes. If Exadata detects a crash during these operations, the offending operation is quarantined, and Exadata falls back to using non-offloaded operations.

These types of quarantines are most likely caused by incompatible versions of CELLSRV. If such quarantines occur on your system, contact Oracle Support Services.

To identify these types of quarantine, run the `LIST QUARANTINE DETAIL` command and check the value of the `quarantineType` attribute. Values associated with these quarantines are `ASM_OFFLOAD_REBALANCE` and `HIGH_THROUGHPUT_WRITE`.

Following are some examples of the output produced by the `LIST QUARANTINE` command.

For ASM rebalance:

```
CellCLI> list quarantine detail
 name:                    2
 asmClusterId:            b6063030c0ffef8dffcc99bd18b91a62
 cellsrvChecksum:         9f98483ef351a1352d567ebb1ca8aeab
 clientPID:               10308
 comment:                 None
 crashReason:             ORA-600[CacheGet::process:C2C_OFFLOAD_CACHEGET_CRASH]
 creationTime:            2016-06-23T22:33:30-07:00
 dbUniqueID:              0
 dbUniqueName:            UnknownDBName
 incidentID:              1
 quarantineMode:          "FULL Quarantine"
 quarantinePlan:          SYSTEM
 quarantineReason:        Crash
 quarantineType:          ASM_OFFLOAD_REBALANCE
 remoteHostName:          slc10vwt
 rpmVersion:              OSS_MAIN_LINUX.X64_160623
```

For high throughput writes originating from a non-container database:

```
CellCLI> list quarantine detail
 name:                    10
 asmClusterId:            b6063030c0ffef8dffcc99bd18b91a62
 cellsrvChecksum:         9f98483ef351a1352d567ebb1ca8aeab
 clientPID:               8377
 comment:                 None
 crashReason:             ORA-600[CacheGet::process:C2C_OFFLOAD_CACHEGET_CRASH]
 creationTime:            2016-06-23T23:47:01-07:00
 conDbUniqueID:           0
 conDbUniqueName:         UnknownDBName
 dbUniqueID:              4263312973
 dbUniqueName:            WRITES
 incidentID:              25
 quarantineMode:          "FULL Quarantine"
 quarantinePlan:          SYSTEM
 quarantineReason:        Crash
 quarantineType:          HIGH_THROUGHPUT_WRITE
 remoteHostName:          slc10vwt
 rpmVersion:              OSS_MAIN_LINUX.X64_160623
```

For high throughput writes originating from a container database (CDB):

```
CellCLI> list quarantine detail
 name:                    10
 asmClusterId:            eff096e82317ff87bfb2ee163731f7f7
 cellsrvChecksum:         9f98483ef351a1352d567ebb1ca8aeab
 clientPID:               17206
 comment:                 None
 crashReason:             ORA-600[CacheGet::process:C2C_OFFLOAD_CACHEGET_CRASH]
 creationTime:            2016-06-24T12:59:06-07:00
 conDbUniqueID:           4263312973
 conDbUniqueName:         WRITES
 dbUniqueID:              0
 dbUniqueName:            UnknownDBName
 incidentID:              25
 quarantineMode:          "FULL Quarantine"
 quarantinePlan:          SYSTEM
 quarantineReason:        Crash
 quarantineType:          HIGH_THROUGHPUT_WRITE
 remoteHostName:          slc10vwt
 rpmVersion:              OSS_MAIN_LINUX.X64_160623
```

1.2.13 Protection Against Data Corruption

Data corruptions, while rare, can have a catastrophic effect on a database, and therefore on a business.

Oracle Exadata System Software 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 Oracle Exadata System Software detects corruptions introduced into the I/O path between the database and storage. The Storage Server stops corrupted data from being written to disk. This eliminates a large class of failures that the database industry had previously been unable to prevent.

Unlike other implementations of corruption checking, checks with Oracle Exadata System Software operate completely transparently. No parameters need to be set at the database or storage tier. These checks transparently handle all cases, including Oracle Automatic Storage Management (Oracle ASM) disk rebalance operations and disk failures.

1.2.14 Fast File Creation

File creation operations are offloaded to Oracle Exadata Storage Servers.

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

File resize operations are also offloaded to the storage servers, which are important for auto-extensible files.

1.2.15 Storage Index

Oracle Exadata Storage Servers 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, prior to Exadata 12.2.1.1.0 each region index stores summaries for up to eight columns, and from Exadata 12.2.1.1.0, each region index may store summaries for up to 24 columns. If set summaries are used, the maximum number of 24 may not be achieved. There is one region index for each 1 MB of disk space. Storage indexes work with any non-linguistic data type, and work with linguistic data types similar to non-linguistic indexes.

Each region 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$SYS_STAT` and `V$SESSTAT` views, shows the number of bytes of I/O saved using storage index. The content stored in one region index is independent of the other region indexes. This makes them highly scalable, and avoids latch contention.

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

Oracle Exadata System Software automatically builds Storage indexes after a query with a comparison predicate that is greater than the maximum or less than the minimum value for the column in a region, and would have benefited if a storage index had been present. Oracle Exadata System Software automatically learns which storage indexes would have benefited a query, and then creates the storage index automatically so that subsequent similar queries benefit.

In Oracle Exadata System Software release 12.2.1.1.0 and later, when data has been stored using the in-memory format columnar cache, Oracle Exadata Database Machine stores these columns compressed using dictionary encoding. For columns with fewer than 200 distinct values, the storage index creates a very compact in-memory representation of the dictionary and uses this compact representation to filter disk reads based on equality predicates. This feature is called **set membership**. A more limited filtering ability extends up to 400 distinct values.

For example, suppose a region of disk holds a list of customers in the United States and Canada. When you run a query looking for customers in Mexico, Oracle Exadata Storage Server can use the new set membership capability to improve the performance of the query by filtering out disk regions that do not contain customers from Mexico. In Oracle Exadata System Software releases earlier than 12.2.1.1.0, which do not have the set membership capability, a regular storage index would be unable to filter those disk regions.

Note:

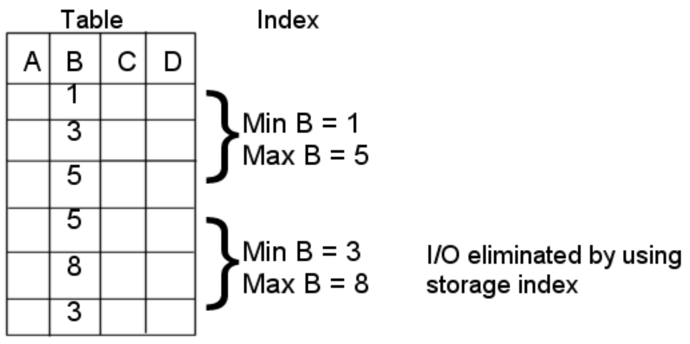
The effectiveness of storage indexes can be improved by ordering the rows based on columns that frequently appear in `WHERE` query clauses.

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 invalidate a region index, and only the storage index on a specific region. The storage index for Exadata Hybrid Columnar Compression is rebuilt on subsequent scans.

Example 1-1 Elimination of Disk I/O with Storage Index

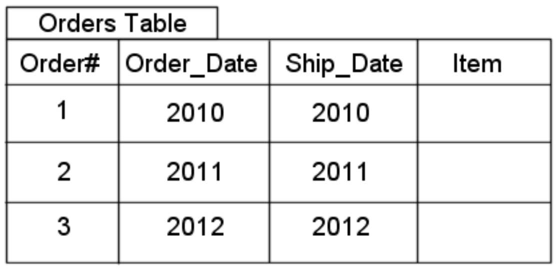
The following figure shows a table and region indexes. The values in the table range from one to eight. One region index stores the minimum 1, and the maximum of 5. The other region index stores the minimum of 3, and the maximum of 8.



For a query such as `SELECT * FROM TABLE WHERE B<2`, only the first set of rows match. Disk I/O is eliminated because the minimum and maximum of the second set of rows do not match the `WHERE` clause of the query.

Example 1-2 Partition Pruning-like Benefits with Storage Index

In the following figure, there is a table named `Orders` with the columns `Order_Number`, `Order_Date`, `Ship_Date`, and `Order_Item`. The table is range partitioned by `Order_Date` column.


Orders Table

Order#	Order_Date	Ship_Date	Item
1	2010	2010	
2	2011	2011	
3	2012	2012	

The following query looks for orders placed since January 1, 2015:

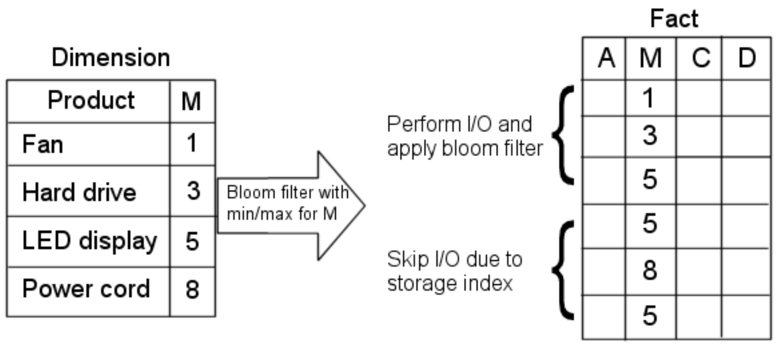
```
SELECT count (*) FROM Orders WHERE Order_Date >= to_date ('2015-01-01', \
'YYY-MM-DD')
```

Because the table is partitioned on the `Order_Date` column, the preceding query avoids scanning unnecessary partitions of the table. Queries on `Ship_Date` do not benefit from `Order_Date` partitioning, but `Ship_Date` and `Order_Number` are highly correlated with `Order_Date`. Storage indexes take advantage of ordering created by partitioning or sorted loading, and can use it with the other columns in the table. This provides partition pruning-like performance for queries on the `Ship_Date` and `Order_Number` columns.

Example 1-3 Improved Join Performance Using Storage Index

Using storage index allows table joins to skip unnecessary I/O operations. For example, the following query would perform an I/O operation and apply a Bloom filter to only the first block of the fact table.

```
SELECT count(*) FROM fact, dim WHERE fact.m=dim.m AND dim.product="Hard drive"
```


Dimension
Product
M
Fan
1
Hard drive
3
LED display
5
Power cord
8
Bloom filter with min/max for M
Fact
A
M
C
D
Perform I/O and apply bloom filter
1
3
5
Skip I/O due to storage index
5
8
5

The I/O for the second block of the fact table is completely eliminated by storage index as its minimum/maximum range (5,8) is not present in the Bloom filter.

1.3 Oracle Exadata System Software Components

This section provides a summary of the following Oracle Exadata System Software components.

- About Oracle Exadata System Software
  Unique software algorithms in Oracle Exadata System Software implement database intelligence in storage, PCI-based flash, and RDMA Network Fabric networking to deliver higher performance and capacity at lower costs than other platforms.
- About Cell Management
  Each cell in the Oracle Exadata Storage Server grid is individually managed with Cell Control Command-Line Interface (CellCLI).
- About Storage Server Security
  Security for Exadata Storage Servers is enforced by identifying which clients can access storage servers and grid disks.

- About Oracle Automatic Storage Management
  Oracle Automatic Storage Management (Oracle ASM) is the cluster volume manager and file system used to manage Oracle Exadata Storage Server resources.
- Maintaining High Performance During Storage Interruptions
- About Database Server Software
  Oracle software is installed on the Exadata database servers.
- About Oracle Enterprise Manager for Oracle Exadata
  Oracle Enterprise Manager provides a complete target that enables you to monitor Oracle Exadata, including configuration and performance, in a graphical user interface (GUI).

1.3.1 About Oracle Exadata System Software

Unique software algorithms in Oracle Exadata System Software implement database intelligence in storage, PCI-based flash, and RDMA Network Fabric networking to deliver higher performance and capacity at lower costs than other platforms.

Oracle Exadata Storage Server is a network-accessible storage device that contains Oracle Exadata System Software. Exadata software communicates with Oracle Database using a specialized *i*DB protocol, and provides both standard I/O functionality, such as block-oriented reads and writes, and advanced I/O functionality, including predicate offload and I/O Resource Management (IORM).

Each storage server has physical disks. The physical disk is an actual device within the storage server that constitutes a hard disk drive (HDD) or flash device. Each physical disk has a logical representation in the operating system (OS) known as a Logical Unit Number (LUN). Typically, there is a one-to-one relationship between physical disks and LUNs on all Exadata Storage Server models. However, on Exadata X10M Extreme Flash (EF) storage servers only, each of the four 30.72 TB capacity-optimized flash devices is configured with 2 LUNs, resulting in 8 LUNs on Exadata X10M EF storage servers.

A cell disk is an Oracle Exadata System Software abstraction built on the top of a LUN. After a cell disk is created from the LUN, it is managed by Oracle Exadata System Software and can be further subdivided into multiple grid disks.

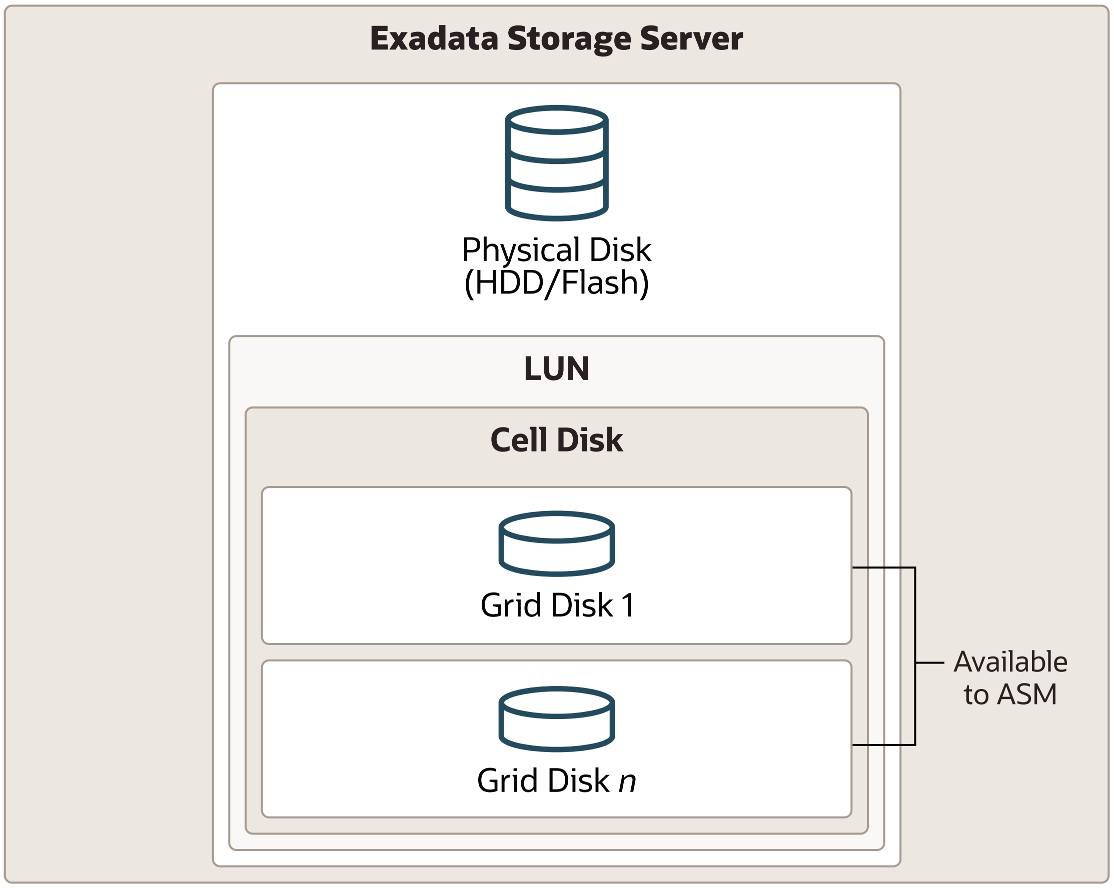
Each grid disk is directly exposed to Oracle ASM for use in an Oracle ASM disk group. This level of virtualization enables multiple Oracle ASM clusters and multiple databases to share the same physical disk. This sharing provides optimal use of disk capacity and bandwidth.

Various metrics and statistics collected on the cell disk level enable you to evaluate the performance and capacity of storage servers. IORM schedules the cell disk access in accordance with user-defined policies.

The following image illustrates the main data storage components in a storage server (also called a **cell**).

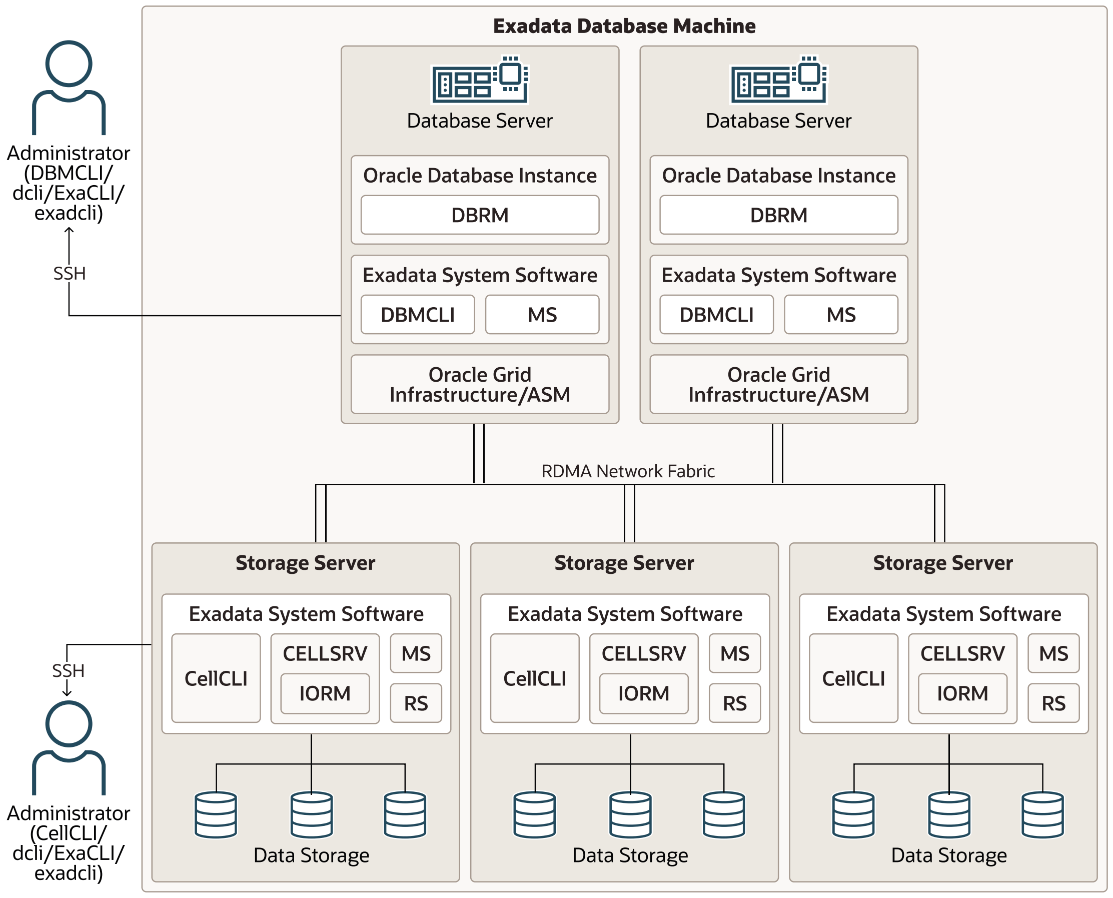
- Oracle Exadata Storage Server includes physical disks, which can be hard disk drives (HDDs) or flash devices.
- Each physical disk is typically represented as a LUN in the cell OS (except for Exadata X10M EF storage servers only, where each capacity-optimized flash device contains 2 LUNs).
- The cell disk is a higher level of abstraction that represents the data storage area on each LUN that is managed by Oracle Exadata System Software. A LUN may contain only one cell disk.
- Each cell disk can contain multiple grid disks, which are directly available to Oracle ASM.

Figure 1-3 Oracle Exadata Data Storage Components



The following image illustrates the major software components in Oracle Exadata.

Figure 1-4 Software Components in Oracle Exadata



The figure illustrates the following environment:

- Single-instance or Oracle RAC databases access Oracle Exadata Storage Servers using the RDMA Network Fabric. Each database server runs the Oracle Database and Oracle Grid Infrastructure software. Resources are managed within each database instance by Oracle Database Resource Manager (DBRM).
- The database servers also include Oracle Exadata System Software components, such as Management Server (MS) and a command-line management interface (DBMCLI).
- Storage servers contain physical data storage devices, along with cell-based utilities and processes from Oracle Exadata System Software, including:
  - Cell Server (CELLSRV) is the primary Exadata storage server software component and provides the majority of Exadata storage services. It serves simple block requests, such as database buffer cache reads, and facilitates Smart Scan requests, such as table scans with projections and filters. CELLSRV implements Exadata I/O Resource Management (IORM), which works in conjunction with Oracle Database resource management to meter out I/O bandwidth to the various databases and consumer groups that are issuing I/Os. CELLSRV also collects numerous statistics relating to its operations. CELLSRV is a multithreaded server and typically uses the largest portion of CPU resources on a storage server.
  - Offload server (CELLOFLSRV <*version*>) is a helper process to CELLSRV that processes offload requests from a specific Oracle Database version. The offload servers enable a storage server to support all offload operations from multiple Oracle

Database versions. The offload servers automatically run in conjunction with CELLSRV, and they require no additional configuration or maintenance.

  – Management Server (MS) provides standalone Oracle Exadata System Software management and configuration functions. MS works in cooperation with and processes most of the commands from the Cell Control Command-Line Interface (CellCLI), which is the primary user interface to administer, manage, and query server status. MS is also responsible for sending alerts, and it collects some statistics in addition to those collected by CELLSRV.
  – Restart Server (RS) monitors the CELLSRV, offload server, and MS processes and restarts them, if necessary.

- The CellCLI utility is the primary management interface for Oracle Exadata System Software on each storage server, while each database server contains a similar utility called DBMCLI. The dcli utility enables administrators to perform management operations across multiple servers, while ExaCLI and exadcli facilitate remote management of Exadata servers, either individually or as a group.

1.3.2 About Cell Management

Each cell in the Oracle Exadata Storage Server 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 (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.

To manage a cell remotely from a compute node, you can use the ExaCLI utility. ExaCLI enables you to run most CellCLI commands on a cell. This is necessary if you do not have direct access to a cell to run CellCLI, or if SSH service on the cell has been disabled. To run commands on multiple cells remotely, you can use the `exadcli` utility.

Related Topics

- Using the CellCLI Utility
  You use the Cell Control Command-Line Interface (CellCLI) utility to manage Oracle Exadata System Software.
- Using the dcli Utility
  The dcli utility facilitates centralized management across Oracle Exadata Database Machine.

See Also:

- Using the ExaCLI Utility in *Oracle Exadata Database Machine Maintenance Guide*, for additional information about managing cells remotely
- Using the exadcli Utility in *Oracle Exadata Database Machine Maintenance Guide*, for additional information about managing multiple cells remotely

1.3.3 About Storage Server Security

Security for Exadata Storage Servers is enforced by identifying which clients can access storage servers 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.

Related Topics

- About Exadata Storage Server Security Modes

1.3.4 About Oracle Automatic Storage Management

Oracle Automatic Storage Management (Oracle ASM) is the cluster volume manager and file system used to manage Oracle Exadata Storage Server 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 non-intrusive cell and disk allocation, deallocation, and reallocation.
- Enabling multiple databases to share storage cells and disks.

The following topics provide a brief overview of Oracle ASM:

- Oracle ASM Disk Groups
  An Oracle Automatic Storage Management (Oracle ASM) disk group is the primary storage abstraction within Oracle ASM, and is composed of one or more grid disks.
- Oracle ASM 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.
- Maximum Availability with Oracle ASM
  Oracle recommends high redundancy Oracle ASM disk groups, and file placement configuration which can be automatically deployed using Oracle Exadata Deployment Assistant.

Related Topics

- Overview of Oracle Automatic Storage Management

1.3.4.1 Oracle ASM Disk Groups

An Oracle Automatic Storage Management (Oracle ASM) disk group is the primary storage abstraction within Oracle ASM, and is composed of one or more grid 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 Oracle Exadata System Software.

Typically, Oracle Exadata is configured with the following Oracle ASM disk groups:

- DATA is the primary data disk group.
- RECO is the primary recovery disk group, which contains the Oracle Database Fast Recovery Area (FRA).
- SPARSE is an optionally configured sparse disk group that is required to support Exadata snapshots.
- XTND is the default name for the disk group that is used to collect storage from Exadata XT (Extended) storage servers.
- DBFS is the system disk group that is typically configured on systems prior to Exadata X7. The DBFS disk group is primarily used to store the shared Oracle Clusterware files (Oracle Cluster Registry and voting disks) and provide some space to host Oracle Database File System (DBFS). This disk group is not configured on Exadata X7, and later, systems.

To take advantage of Oracle Exadata System Software features, such as predicate processing offload, the disk groups must contain only Oracle Exadata Storage Server grid disks, the tables must be fully inside these disk groups, and the group should have `cell.smart_scan_capable` attribute set to TRUE.

Note:

The Oracle Database and Oracle Grid Infrastructure software must be release 12.1.0.2.0 BP3 or later when using sparse grid disks.

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

1.3.4.2 Oracle ASM 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 Oracle Exadata Storage Servers, 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 Oracle Exadata System Software and Oracle ASM to tolerate the failure of either storage cell.

Failure groups for Oracle Exadata Storage Server 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 Oracle Exadata Storage Servers.

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 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.

Related Topics

- Administering Oracle ASM on Exadata

1.3.4.3 Maximum Availability with Oracle ASM

Oracle recommends high redundancy Oracle ASM disk groups, and file placement configuration which can be automatically deployed using Oracle Exadata Deployment Assistant.

High redundancy can be configured for DATA, RECO, or any other Oracle ASM disk group with a minimum of 3 storage cells. Starting with Exadata Software release 12.1.2.3.0, the voting disks can reside in a high redundancy disk group, and additional quorum disks (essentially equivalent to voting disks) can reside on database servers if there are fewer than 5 Exadata storage cells.

Maximum availability architecture (MAA) best practice uses two main Oracle ASM disk groups: DATA and RECO. The disk groups are organized as follows:

- The disk groups are striped across all disks and Oracle Exadata Storage Servers to maximize I/O bandwidth and performance, and to simplify management.
- The DATA and RECO disk groups are configured for high (3-way) redundancy.

The benefits of high redundancy disk groups are illustrated by the following outage scenarios:

- Double partner disk failure: Protection against loss of the database and Oracle ASM disk group due to a disk failure followed by a second partner disk failure.
- Disk failure when Oracle Exadata Storage Server is offline: Protection against loss of the database and Oracle ASM disk group when a storage server is offline and one of the storage server's partner disks fails. The storage server may be offline because of Exadata storage planned maintenance, such as Exadata rolling storage server patching.
- Disk failure followed by disk sector corruption: Protection against data loss and I/O errors when latent disk sector corruptions exist and a partner storage disk is unavailable either due to planned maintenance or disk failure.

If the voting disk resides in a high redundancy disk group that is part of the default Exadata high redundancy deployment, the cluster and database will remain available for the above failure scenarios. If the voting disk resides on a normal redundancy disk group, then the database cluster will fail and the database has to be restarted. You can eliminate that risk by moving the voting disks to a high redundancy disk group and creating additional quorum disks on database servers.

Oracle recommends high redundancy for ALL (DATA and RECO) disk groups because it provides maximum application availability against storage failures and operational simplicity during a storage outage. In contrast, if all disk groups were configured with normal redundancy and two partner disks fail, all clusters and databases on Exadata will fail and you will lose all your data (normal redundancy does not tolerate double partner disk failures). 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. The additional write I/O normally has negligible impact with Exadata smart write-back flash cache.

The following table describes that redundancy option, as well as others, and the relative availability trade-offs. The table assumes that voting disks reside in high redundancy disk group. Refer to *Oracle Exadata Database Machine Maintenance Guide* to migrate voting disks to high redundancy disk group for existing high redundancy disk group configurations.

Redundancy Option	Availability Implications	Recommendation
High redundancy for ALL (DATA and RECO)	Zero application downtime and zero data loss for the preceding storage outage scenarios if voting disks reside in high redundancy disk group. If voting disks currently reside in normal redundancy disk group, refer to *Oracle Exadata Database Machine Maintenance Guide* to migrate them to high redundancy disk group.	Use this option for best storage protection and operational simplicity for mission-critical applications. Requires more space for higher redundancy.

Redundancy Option	Availability Implications	Recommendation
High redundancy for DATA only	Zero application downtime and zero data loss for preceding storage outage scenarios. This option requires an alternative archive destination.	Use this option for best storage protection for DATA with slightly higher operational complexity. More available space than high redundancy for ALL. Refer to My Oracle Support note 2059780.1 for details.
High redundancy for RECO only	Zero data loss for the preceding storage outage scenarios.	Use this option when longer recovery times are acceptable for the preceding storage outage scenarios. Recovery options include the following: • Restore and recover: - Recreate DATA disk group - Restore from RECO and tape-based backups, if required - Recover database • Switch and recover: - Use RMAN switch to copy Recover database
Normal Redundancy for ALL (DATA and RECO) **Note:** Cross-disk mirror isolation by using ASM disk group content type limits an outage to a single disk group when two disk partners are lost in a normal redundancy group that share physical disks and storage servers.	The preceding storage outage scenarios resulted in failure of all Oracle ASM disk groups. However, using cross-disk group mirror isolation the outage is limited to one disk group. **Note:** This option is not available for eighth or quarter racks.	Oracle recommends a minimum of high redundancy for DATA only. Use the Normal Redundancy for ALL option when the primary database is protected by an Oracle Data Guard standby database deployed on a separate Oracle Exadata Database Machine or when the Exadata Database Machine is servicing only development or test databases. Oracle Data Guard provides real-time data protection and fast failover for storage failures. If Oracle Data Guard is not available and the DATA or RECO disk groups are lost, then leverage recovery options described in My Oracle Support note 1339373.1.

The optimal file placement for setup for MAA is:

- **Oracle Database files** — DATA disk group
- **Flashback log files, archived redo files, and backup files** — RECO disk group
- **Redo log files** — First high redundancy disk group. If no high redundancy disk group exists, then redo log files are multiplexed across the DATA and RECO disk groups.
- **Control files** — First high redundancy disk group. If no high redundancy disk groups exist, the use one control file in the DATA disk group. The backup control files should reside in the RECO disk group, and `RMAN CONFIGURE CONTROLFILE AUTOBACKUP ON` should be set.

- **Server parameter files (SPFILE)** — First high redundancy disk group. If no high redundancy disk group exists, then SPFILE should reside in the DATA disk group. SPFILE backups should reside in the RECO 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. If no high redundancy disk group exists, then the files should reside in the DATA disk group.
- **Voting disks for Oracle Exadata Database Machine Quarter Rack or Eighth Rack** — First high redundancy disk group, otherwise in normal redundancy disk group. If there are fewer than 5 Exadata storage cells with high redundancy disk group, additional quorum disks will be stored on Exadata database servers during OEDA deployment. Refer to *Oracle Exadata Database Machine Maintenance Guide* to migrate voting disks to high redundancy disk group for existing high redundancy disk group configurations.
- **Temporary files** — First normal redundancy disk group. If the high redundancy for ALL option is used, then the use the first high redundancy disk group.
- **Staging and non-database files** — DBFS disk group or ACFS volume
- **Oracle software (including audit and diagnostic destinations)** — Exadata database server local file system locations configured during OEDA deployment

Related Topics

- Database High Availability Checklist
- Configuration Prerequisites and Operational Steps for Higher Availability for a RECO disk group or Fast Recovery Area Failure (My Oracle Support Doc ID 2059780.1)
- Operational Steps for Recovery after Losing a Disk Group in an Exadata Environment (My Oracle Support Doc ID 1339373.1)

1.3.5 Maintaining High Performance During Storage Interruptions

Exadata is engineered to deliver high performance by intelligently managing data across multiple storage tiers and caches. Early Exadata models feature high-performance low-latency flash storage, while later models deliver higher performance and extremely low latency by adding persistent memory (PMEM) or Exadata RDMA Memory (XRMEM). During normal operations, Exadata intelligently manages the storage tiers to ensure that the most relevant data uses the storage location with the highest performance and lowest latency. However, special features also ensure that high performance and low I/O latency are maintained when storage interruptions occur, regardless of whether the event is planned or unplanned.

One of the most common types of unplanned storage events is the failure of a flash device or hard disk, which can come in a variety of forms, including outright hardware failure, predictive failure, confinement, and so on. Other unplanned storage events can arise from different hardware or software component failures, such as an Operating System kernel crash that results in a cell reboot. The most common type of planned storage event is a storage server software update. Storage server software updates are performed in a rolling manner, where one cell is updated and re-synchronized before moving on to the next cell.

Whatever the storage interruption, I/O latency is primarily impacted by two factors:

- Additional I/O load on the system required to deal with the interruption.
- Cache misses caused by the interruption.

Exadata deals with these impacts using a suite of measures as follows:

Managing the I/O load associated with a storage interruption

- When storage events occur, Exadata automatically orchestrates the appropriate response to efficiently maintain or restore data redundancy. By using the right approach for each situation, Exadata minimizes the impact of the I/O load required to maintain redundancy:
  - When a hard drive fails, Exadata automatically drops the disk from Oracle ASM. This action automatically triggers an ASM rebalance operation, which restores redundancy to the ASM disk group. The same occurs when failure impacts ASM disks that reside on a flash drive.
  - If a hard drive displays poor performance or enters a predictive failure state, Exadata provides the option to proactively drop the disk and perform a rebalance to maintain redundancy before the drive is replaced.
  - When using Exadata Smart Flash Cache in write-back mode, Exadata automatically maintains metadata that describes the cache contents. When a flash drive failure impacts the cache, Exadata automatically repopulates the cache after the device is replaced. Known as resilvering, this process repopulates the cache by reading from surviving mirrors using highly efficient cell-to-cell direct data transfers over the RDMA network fabric.
  - When storage comes back online after a short-term interruption, Exadata automatically instructs ASM to perform a resync operation to restore redundancy. A resync operation uses a bitmap that tracks storage extent changes. For short-term interruptions, such as those associated with software updates or cell reboots, a resync is a very efficient way to restore redundancy by copying just the changed data.
- ASM provides a throttle, known as the ASM power limit, for I/Os associated with asynchronous operations, such as rebalance and resync. If the ASM power limit is too high, the ASM I/Os can overload the hard disks and increase I/O latency. A high limit can also increase ASM extent locking, potentially impacting database I/Os. However, on Exadata, the default (and recommended) ASM power limit setting is very low, which ensures minimal impact on application I/O latency. This setting is also monitored by EXAchk, and any variation is included in the EXAchk report.
- Exadata I/O Resource Management (IORM) distinguishes between system and application I/Os and intelligently prioritizes the application I/Os. For example, application I/Os get priority access to the Exadata caches while an ASM rebalance can only access the hard disks and unused cache space.

Minimizing Cache Misses

- In the course of normal operations, data read into the database buffer cache is also loaded into an Exadata cache (either flash cache, PMEM cache, or XRMEM cache depending on availability) on the cell containing the primary data copy. And, data is always read from the primary copy, when available.

  However, if the primary copy is unavailable, then the secondary copy must be used. To prepare for this possibility, as part of maintaining the primary cache, Exadata also loads data into flash cache on the secondary cell. By proactively loading the secondary cache, Exadata ensures that the most important data is still cached when the primary copy is unavailable and the secondary copy must be used.

  If the secondary data copy is also unavailable and the data is protected by high redundancy (triple-mirrored), then the tertiary data copy is used as a last resort. This is a rare scenario that requires a simultaneous double failure. Consequently, Exadata does not provide proactive caching of the tertiary data copy.

- Exadata preserves cache contents when data is moved between cells. For example, if some data on cell 1 is loaded into the flash cache, and that data is moved to cell 2, then the data is also loaded into the flash cache on cell 2. This ensures that data moved by a rebalance operation is cached in the same way as before the move.
- Exadata expedites flash cache recovery after a failure. For process failures, Exadata can reattach the flash cache and continue with previously populated data. To quickly rebuild the flash cache after system failures, Exadata maintains flash cache metadata on the M.2 solid-state drives (SSD) found on Oracle Exadata X7 and later systems.
- When a new storage device is detected, for example after the replacement of a flash drive or hard drive, Exadata ensures that the flash cache is properly 'warmed up' before fully enabling the new storage. This includes extensive health checks to ensure that the flash cache hit ratio associated with the new storage is similar to the rest of the system.

1.3.6 About Database Server Software

Oracle software is installed on the Exadata database servers.

Oracle Exadata System Software works seamlessly with Oracle Database. The software on the database servers 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. The database server software also includes utilities for administration, performance management, and support of the database.
- Oracle Automatic Storage Management (Oracle ASM), is a clustered file system and volume manager which provides storage management optimized for the database and Oracle Exadata Storage Servers. Oracle ASM is part of Oracle Grid Infrastructure. The Oracle Grid Infrastructure software provides the essential functions to maintain cluster coherence for all the Exadata servers. The Oracle Grid Infrastructure software also monitors the health and liveness of both database and storage servers, providing database high availability in case of planned and unplanned storage outages.

  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 Oracle Database Resource Manager, 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.

Related Topics

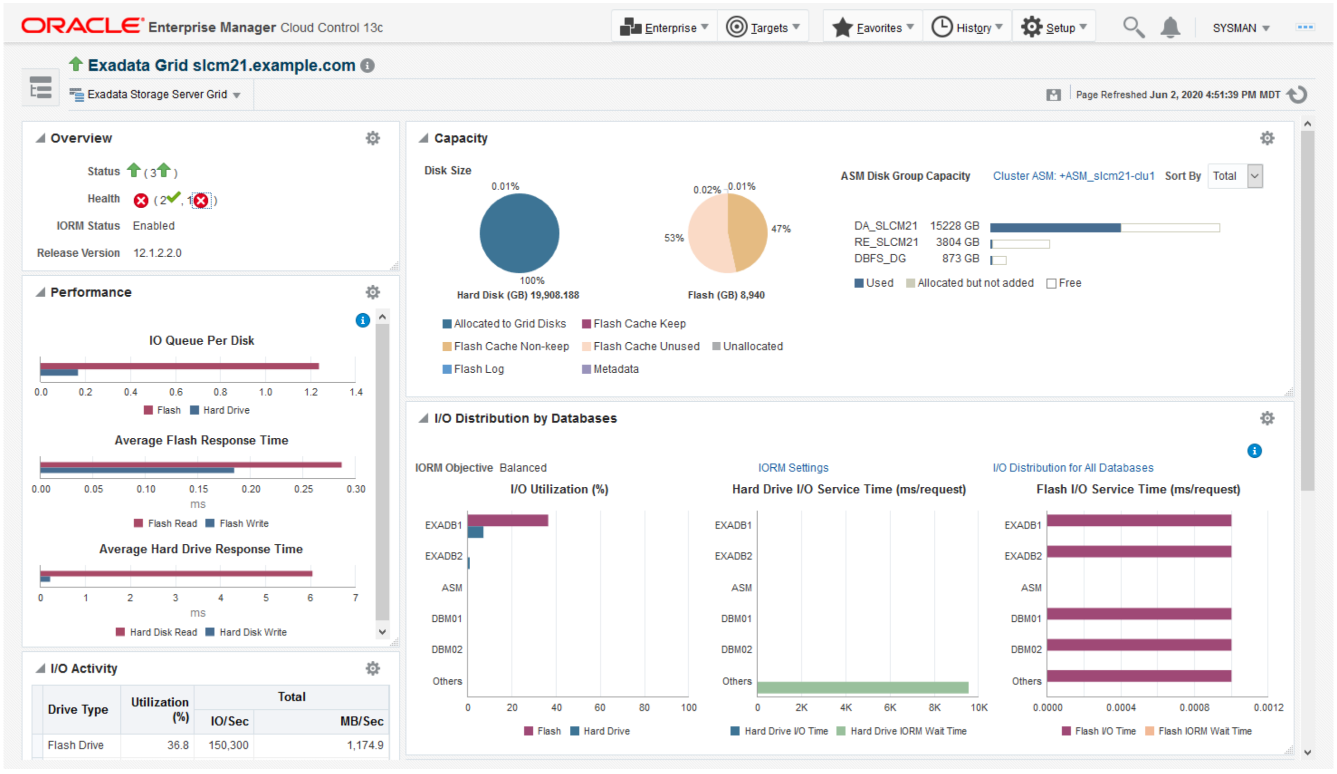
- Managing I/O Resources
  Exadata I/O Resource Management (IORM) is a tool for managing how multiple workloads and databases share the I/O resources of Oracle Exadata Database Machine.
- About Oracle Exadata System Software
  Unique software algorithms in Oracle Exadata System Software implement database intelligence in storage, PCI-based flash, and RDMA Network Fabric networking to deliver higher performance and capacity at lower costs than other platforms.

1.3.7 About Oracle Enterprise Manager for Oracle Exadata

Oracle Enterprise Manager provides a complete target that enables you to monitor Oracle Exadata, including configuration and performance, in a graphical user interface (GUI).

The following figure shows the Exadata Storage Server Grid home page. Viewing this page, you can quickly see the health of the storage servers, key storage performance characteristics, and resource utilization of storage by individual databases.

Figure 1-5 Exadata Storage Server Grid home page in Oracle Enterprise Manager



Page Refreshed Jun 2, 2020 4:51:39 PM MDT

Status (3)
Health (2, 1)
IORM Status Enabled
Release Version 12.1.2.2.0

DA_SLCM21	15228 GB
RE_SLCM21	3804 GB
DBFS_DG	873 GB

IORM Objective Balanced

Drive Type	Utilization (%)	Total	
		IO/Sec	MB/Sec
Flash Drive	36.8	150,300	1,174.9

In addition to reports, Oracle Enterprise Manager enables you to set metric thresholds for alerts and monitor metric values to determine the health of your Exadata systems.

Related Topics

- *Oracle Enterprise Manager Exadata Management Getting Started Guide*

2

Administering Oracle ASM on Exadata

Related Topics

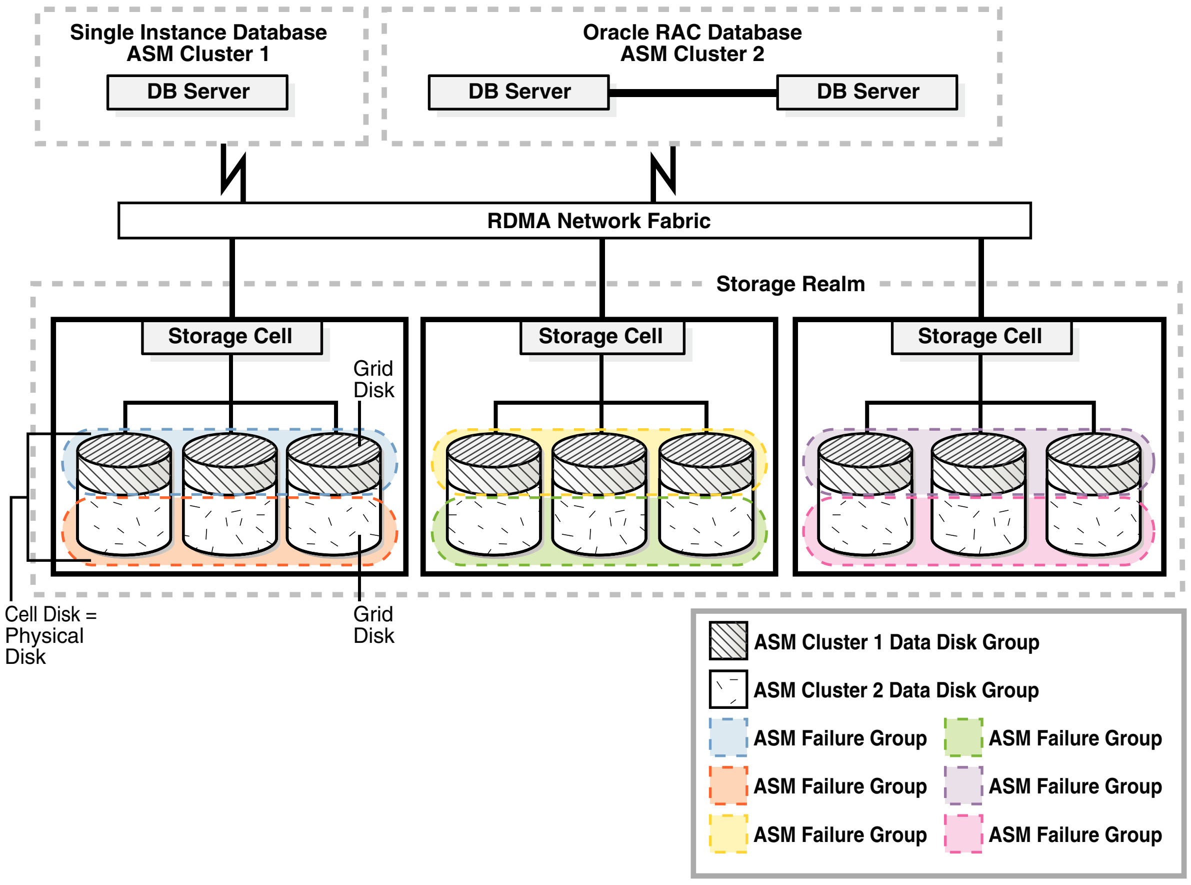
- Maintaining the Hard Disks of Exadata Storage Servers

2.1 Overview of Oracle Exadata Storage

Storage in Oracle Exadata consists of servers, cell disks, grid disks, Oracle ASM disk groups, and Oracle ASM failure groups.

The following image shows Oracle ASM disk groups created from Oracle Exadata Storage Server grid disks. It represents a typical, but simplified configuration, that can be used as a model for building larger storage grids with additional storage servers and disks.

Figure 2-1   Sample Oracle Exadata Storage Server Grid



This example storage grid illustrates the following:

- The storage servers in the grid use an RDMA Network Fabric network to connect to the database servers that have a single-instance database or Oracle Real Application Clusters (Oracle RAC) database installation.
- Each storage server contains multiple 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 created using the grid disks.

Oracle ASM failure groups are created to ensure that files are not mirrored on the same storage server, enabling the system to tolerate the failure of a storage server. The number of failure groups equals the number of storage servers. Each failure group is composed of a subset of grid disks in the Oracle ASM disk group that belong to a single storage server.

2.2 Administering Oracle ASM on Exadata

There are some administration tasks that may be required to use Oracle ASM on Exadata.

- Configuring Exadata Storage Discovery for Oracle ASM

- Understanding Oracle ASM Disk Groups for Oracle Exadata Storage Servers
  This topic explains Oracle Automatic Storage Management (Oracle ASM) disk groups, and how to create an Oracle ASM disk group for Oracle Exadata System Software using the `CREATE DISKGROUP` SQL command.
- Creating Oracle ASM Disk Groups
  You can create Oracle ASM disk groups on Oracle Exadata Storage Server grid disks.
- Adding a Disk to an Oracle ASM Disk Group
  You can add a disk to an Oracle ASM disk group.
- Mounting or Dismounting an Oracle ASM Disk Group
  A disk group must be mounted by Oracle ASM before a database can access the files in the disk group.
- Changing a Disk to Offline or Online
  You can change an Oracle ASM disk to `INACTIVE` or `ACTIVE`.
- Dropping a Disk from an Oracle ASM Disk Group
  You can drop a grid disk from a disk group.
- Dropping an Oracle ASM Disk Group
  You can drop an Oracle ASM disk group.
- Enabling the Oracle ASM appliance.mode Attribute
  The Oracle ASM `appliance.mode` attribute improves disk rebalance completion time when dropping one or more Oracle ASM disks.
- Checking Disk Group Balance
- Setting the Oracle ASM Disk Repair Timer

2.2.1 Configuring Exadata Storage Discovery for Oracle ASM

To enable Oracle ASM to discover and access Exadata grid disks, you must configure the `ASM_DISKSTRING` initialization parameter.

Exadata grid disks are specified by using a discovery string with the following format:

```
o/<cell_IP_pattern>/<griddisk_name_pattern>
```

In the discovery string:

- *`<cell_IP_pattern>`* identifies Exadata storage server IP address (as listed in the `cellip.ora` file).
- *`<griddisk_name_pattern>`* identifies grid disks by name.

The wildcard character (`*`) can be used to expand the *`<cell_IP_pattern>`* and *`<griddisk_name_pattern>`* values.

For example, the following `ASM_DISKSTRING` setting discovers all Exadata grid disks on all cells specified in the `cellip.ora` file:

```
ASM_DISKSTRING = 'o/*/*'
```

You can use a more specific setting to discover a subset of grid disks. For example, the following `ASM_DISKSTRING` setting discovers only the grid disks with names that begin with `DATA`:

```
ASM_DISKSTRING = 'o/*/DATA*'
```

You can change the `ASM_DISKSTRING` initialization parameter when the Oracle ASM instance is running with the SQL `ALTER SYSTEM` command. If you edit the `ASM_DISKSTRING` initialization parameter in the initialization parameter file when the Oracle ASM instance is running, then the Oracle ASM instance must be shut down and restarted for the change to take effect.

See Also:

- *Oracle Automatic Storage Management Administrator's Guide* for additional information about the following:
  - Oracle ASM discovery strings
  - starting up and connecting to an Oracle ASM instance
- *Oracle Database Reference* for additional information about the ASM_DISKSTRING initialization parameter
- *Oracle Database SQL Language Reference* for additional information about the `ALTER SYSTEM` command

2.2.2 Understanding Oracle ASM Disk Groups for Oracle Exadata Storage Servers

This topic explains Oracle Automatic Storage Management (Oracle ASM) disk groups, and how to create an Oracle ASM disk group for Oracle Exadata System Software 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 Oracle Exadata Storage Server 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, and *griddisk_name* is the name of the grid disk.

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

- Wildcards in the form of '*' may be used in the *cell_IPaddress* or *griddisk_name* to add multiple disks with one command. For example:

  ```
  CREATE DISKGROUP reco HIGH REDUNDANCYDISK 'o/*/DATA*'
  ```

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 Servers 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 Oracle Exadata Storage Server grid disks. You cannot include conventional disks with Oracle Exadata Storage Server grid disks.

- About Fast Disk Scan Rates
- Setting the Oracle ASM Content Type
  Setting the `content.type` disk group attribute enhances fault tolerance, especially for normal redundancy disk groups.

Related Topics

- Naming Conventions for Oracle Exadata Storage Server Grid Disks
  Using a consistent naming convention helps to identify Exadata components.
- Oracle ASM Disk Groups
  An Oracle Automatic Storage Management (Oracle ASM) disk group is the primary storage abstraction within Oracle ASM, and is composed of one or more grid disks.

2.2.2.1 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. The `compatible.rdbms` attribute is set to 11.2.0.2 in order to support both release 11.2.0.2 and release 11.2.0.3 databases in a consolidated environment.

```
SQL> CREATE DISKGROUP data NORMAL REDUNDANCY
     DISK 'o/*/data_CD*'
     ATTRIBUTE 'compatible.rdbms' = '11.2.0.2',
               'compatible.asm' = '11.2.0.3',
               'content.type' = 'data',
               'cell.smart_scan_capable' = 'TRUE',
               'au_size' = '4M';
```

Related Topics

- *Oracle Database SQL Language Reference*

2.2.2.2 Setting the Oracle ASM Content Type

Setting the `content.type` disk group attribute enhances fault tolerance, especially for normal redundancy disk groups.

Commencing with Oracle Grid Infrastructure release 11.2.0.3, Oracle ASM provides administrators with the option to specify the content type associated with each disk group. This capability is provided by the `content.type` disk group attribute. Three possible settings are allowed: `data`, `recovery`, or `system`. Each content type setting modifies the adjacency measure used by the secondary extent placement algorithm.

The result is that the contents of disk groups with different content type settings are distributed differently across the available disks. This decreases the likelihood that a double failure will result in data loss across multiple normal redundancy disk groups with different content type settings. Likewise, a triple failure is less likely to result in data loss on multiple high redundancy disk groups with different content type settings.

The value of `content.type` attribute should be set as follows:

- DATA and SPARSE disk groups — `data`
- RECO disk group — `recovery`
- DBFS_DG disk group (if present) — `system`

Following this recommendation enhances fault-tolerance. For example, even if the DATA and RECO disk groups use normal redundancy, at least one of the disk groups will remain if two disks fail simultaneously. Therefore, even if DATA is dismounted, databases can typically be recovered from backup objects in RECO.

Note:

- Do not use the `content.type` attribute to distinguish the availability characteristics of disk groups that are used for a different purpose, such as those created to support a particular service.
- The Oracle Database and Oracle Grid Infrastructure software must be release 12.1.0.2.0 BP5 or later when using sparse grid disks.

1. Use the `ALTER DISKGROUP` command to set the `content.type` attribute for an existing disk group, and then rebalance the disk group.

   For example:

   ```
   ALTER DISKGROUP reco SET ATTRIBUTE 'content.type'='recovery';
   ALTER DISKGROUP reco REBALANCE POWER preferred_power_setting ;
   ```

   The rebalance operation can take a long time, but the data in the disk group is fully redundant throughout the operation. Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation completes.

2. Check the `content.type` attributes using the following query:

   ```
   SQL> SELECT dg.name,a.value FROM v$asm_diskgroup dg,         \
        v$asm_attribute a WHERE dg.group_number=a.group_number \
        AND a.name='content.type' AND (dg.name LIKE 'DATA%'    \
        OR dg.name LIKE 'RECO%' OR dg.name LIKE 'DBFS_DG%');

   NAME                 VALUE
   -------------------- --------------------
   DATA                 data
   RECO                 recovery
   DBFS_DG              system
   ```

Example 2-1 Specifying `content.type` While Creating a Disk Group

In this example, the `compatible.rdbms` attribute is set to 11.2.0.2 in order to support both Oracle Database release 11.2.0.2 and release 11.2.0.3 databases in a consolidated environment.

```
CREATE DISKGROUP data NORMAL REDUNDANCY
DISK 'o/*/DATA*'
ATTRIBUTE 'content.type' = 'DATA',
'AU_SIZE' = '4M',
'cell.smart_scan_capable'='TRUE',
'compatible.rdbms'='11.2.0.2',
'compatible.asm'='11.2.0.3';
```

2.2.3 Creating Oracle ASM Disk Groups

You can create Oracle ASM disk groups on Oracle Exadata Storage Server grid disks.

To create an Oracle ASM disk group to use Oracle Exadata Storage Server 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 Oracle Exadata Storage Server 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.

   In this example, the `ALTER` command is needed to change `compatible.rdbms` for the disk group created during installation to hold the OCR and voting disks. The `compatible.rdbms` attribute is set to 11.2.0.2 in order to support Oracle Database release 11.2.0.2 and later release databases in a consolidated environment.

```
CREATE DISKGROUP data HIGH REDUNDANCY
DISK 'o/*/DATA*'
ATTRIBUTE 'AU_SIZE' = '4M',
          'content.type' = 'data',
          'compatible.rdbms'='11.2.0.4',
          'compatible.asm'='19.0.0.0';

SQL> CREATE DISKGROUP reco HIGH REDUNDANCY
DISK 'o/*/RECO*'
ATTRIBUTE 'AU_SIZE' = '4M',
          'content.type' = 'recovery',
          'compatible.rdbms'='11.2.0.4',
          'compatible.asm'='19.0.0.0';

REM for Exadata systems prior to X7
SQL> ALTER DISKGROUP dbfs_dg SET ATTRIBUTE
     'content.type' = 'system',
     'compatible.rdbms' = '11.2.0.4';
```

When creating sparse disk groups, use a command similar to the following:

```
SQL> CREATE DISKGROUP sparsedg NORMAL REDUNDANCY
DISK 'o.*/sparse_*'
ATTRIBUTE 'AU_SIZE' = '4M',
          'content.type' = 'data',
          'cell.smart_scan_capable'='TRUE',
          'compatible.rdbms' = '12.1.0.2',
          'compatible.asm' = '19.0.0.0',
          'cell.sparse_dg' = 'allsparse';
```

In the preceding command, the `cell.sparse_dg` attribute defines the disk group as a sparse disk group. The attribute is not required if the disk group is not a sparse disk group.

Note:

- When defining sparse grid disks, the `compatible.asm` and `compatible.rdbms` attributes must be at least `12.1.0.2.0`.
- The Oracle ASM disk group `compatible` attributes take precedence over the `COMPATIBLE` initialization parameter for the Oracle ASM instance.
- The Oracle Database and Oracle Grid Infrastructure software must be release 12.1.0.2.0 BP5 or later when using sparse grid disks.
- The recommended allocation unit size (`AU_SIZE`) is 4 MB for Oracle ASM disk groups on Exadata.

6. View the Oracle ASM disk groups and associated attributes with a SQL query on `V$ASM` dynamic views.

```
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           11.2.0.4
DATA                         compatible.asm             19.0.0.0
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 Oracle Exadata System Software features, such as offload processing. The tablespace should contain the tables that you want to query with offload processing.

```
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 Oracle Exadata Storage Server disk group. The `PREDICATE_EVALUATION` column of the `DBA_TABLESPACES` view indicates whether predicates are evaluated by host (`HOST`) or by storage (`STORAGE`).

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

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

Related Topics

- Introducing Oracle Automatic Storage Management
- Administration of Oracle ASM Instances
- ALTER DISKGROUP
- CREATE DISKGROUP
- CREATE TABLESPACE

2.2.4 Adding a Disk to an Oracle ASM Disk Group

You can add a disk to an Oracle ASM disk group.

You might need to do this if you are adding a new Oracle Exadata Storage Server or managing a custom disk group.

Do not add Oracle Exadata Storage Server grid disks to an Oracle ASM disk group that is not on an Oracle Exadata Storage Server unless you are planning to migrate the disk group to an Oracle Exadata Storage Server disk group.

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.

2. Use the SQL command `ALTER DISKGROUP` with the `ADD DISK` clause to add the disk to the Oracle ASM disk group.

   For example:

   ```
   SQL> ALTER DISKGROUP disk_group_name
   ADD DISK 'o/cell_IP_address/grid_disk_prefix*';
   ```

After the disk is added, Oracle ASM rebalances the disk group. Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation is complete.

You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

Related Topics

- Naming Conventions for Oracle Exadata Storage Server Grid Disks
  Using a consistent naming convention helps to identify Exadata components.

2.2.5 Mounting or Dismounting an Oracle ASM Disk Group

A disk group must be mounted by Oracle ASM before a database can access the files in the disk group.

Mounting a disk group requires discovering all of the disks and locating the files in the disk group. When an Oracle ASM starts, the disk groups mentioned in the `ASM_DISKGROUPS` instance parameter are automatically mounted.

Additionally:

- To mount a disk group, you can use the SQL `ALTER DISKGROUP` command with the `MOUNT` option.

  You can use the `FORCE` option in conjunction with the `ALTER DISKGROUP ... MOUNT` command to mount a disk group even if a disk is unavailable. However, this compromises redundancy in the disk group.

- To dismount a disk group, you can use the SQL `ALTER DISKGROUP` command with the `DISMOUNT` option.

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

2.2.6 Changing a Disk to Offline or Online

You can change an Oracle ASM disk to `INACTIVE` or `ACTIVE`.

1. Determine which disk you want offline or online in the Oracle ASM disk group.

   Query 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, the disk group is rebalanced.

Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation is complete.

You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

Related Topics

- 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.
- ALTER GRIDDISK

2.2.7 Dropping a Disk from an Oracle ASM Disk Group

You can drop a grid disk from a disk group.

1. Determine which disks you want to drop from the Oracle ASM disk group.

   Query the `V$ASM_DISK` and `V$ASM_DISKGROUP` views on the Oracle ASM instance.

   If you are removing an Oracle Exadata Storage Server 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.

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

   Do not use the `FORCE` option when dropping the disk from the Oracle ASM disk group. If you use the `FORCE` option, Oracle Exadata System Software will attempt to add the disk back to the disk group if the disk online automation operation is triggered, by rebooting the storage server, for example. See Enhanced Manageability Features in *Oracle Exadata Database Machine System Overview*.

When the disk is dropped from the Oracle ASM disk group, Oracle ASM rebalances the disk group. Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation is complete.

You can query the `V$ASM_OPERATION` view for the status of the rebalance operation.

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

Related Topics

- 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.
- 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.
- Dropping an Oracle Exadata Storage Server Grid Disk
  To drop an Oracle Exadata Storage Server grid disk, use the CellCLI `DROP GRIDDISK` command.

2.2.8 Dropping an Oracle ASM Disk Group

You can drop an 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.

1. Determine the disk group that you want to drop.

   Query the `V$ASM_DISKGROUP` view on the Oracle ASM instance.
2. Use the SQL `DROP DISKGROUP` command to drop the Oracle ASM disk group.

2.2.9 Enabling the Oracle ASM appliance.mode Attribute

The Oracle ASM `appliance.mode` attribute improves disk rebalance completion time when dropping one or more Oracle ASM disks.

Setting the `appliance.mode` attribute helps restore redundancy faster after a failure. The attribute can only be enabled on disk groups that meet the following requirements:

- The Oracle ASM disk group attribute `compatible.asm` is set to release 11.2.0.4, or 12.1.0.2 or later.
- The `cell.smart_scan_capable` attribute is set to `TRUE`.
- All disks in the disk group are the same type; for example, all disks are hard disks or all disks are flash disks.
- All disks in the disk group are the same size.
- All failure groups in the disk group have an equal number of disks:
  - For eighth rack configurations, all failure groups have 4 disks, or all failure groups have 6 disks.
  - For all other rack configurations, all failure groups have 10 disks, or all failure groups have 12 disks.
- There are at least 3 failure groups in the disk group.
- No disk in the disk group is offline.

> **Note:**
>
> Enabling the `appliance.mode` attribute for existing disk groups may cause an increase of data movement during the next rebalance operation.

The `appliance.mode` attribute is automatically enabled when creating a new disk group. Existing disk groups must explicitly set the attribute using the `ALTER DISKGROUP` command.

```
SQL> ALTER DISKGROUP disk_group SET ATTRIBUTE 'appliance.mode'='TRUE';
```

> **Note:**
>
> The `appliance.mode` attribute should normally be set to `TRUE`. In rare cases it may be necessary to disable `appliance.mode` as a workaround when adding disks to a disk group. After the disk group is ALTERed enable `appliance.mode`, and perform a REBALANCE operation.

To disable the `appliance.mode` attribute during disk group creation, set the attribute to `FALSE`.

```
SQL> CREATE DISKGROUP data NORMAL REDUNDANCY
DISK
'o/*/DATA*'
ATTRIBUTE 'content.type' = 'data',
          'au_size' = '4M',
          'cell.smart_scan_capable'='TRUE',
          'compatible.rdbms'='11.2.0.3',
          'compatible.asm'='11.2.0.4',
          'appliance.mode'='FALSE';
```

2.2.10 Checking 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 values in 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 document 367445.1.

2.2.11 Setting the 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:

```
SQL> 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:

```
SQL> ALTER DISKGROUP data OFFLINE DISK data_CD_06_cell11 DROP AFTER 20h;
```

Note:

Vulnerability to a double failure increases in line with increases to the disk repair time value.

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

2.3 Administering Oracle Exadata Storage Server Grid Disks with Oracle ASM

Use the following procedures for managing grid disks used with Oracle ASM.

- Naming Conventions for Oracle Exadata Storage Server Grid Disks
  Using a consistent naming convention helps to identify Exadata components.
- Changing an Oracle Exadata Storage Server Grid Disk That Belongs to an Oracle ASM Disk Group
  Before 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.
- Resizing Grid Disks
  You can resize grid disks and Oracle ASM disk groups to shrink one with excess free space and increase the size of another that is near capacity.
- 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.
- 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.

- Handling Disk Replacement
  If a disk has a problem, the physical disk status changes.

2.3.1 Naming Conventions for Oracle Exadata Storage Server Grid Disks

Using a consistent naming convention helps to identify Exadata components.

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. You cannot have multiple disks with the same name in an Oracle ASM disk group.

Related Topics

- 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.
- 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.
- CREATE CELLDISK
- CREATE GRIDDISK

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

Before 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.

- Changing an Oracle Exadata Storage Server Grid Disk Name
  Use the CellCLI interface to change the name of a grid disk.
- Dropping an Oracle Exadata Storage Server Grid Disk
  To drop an Oracle Exadata Storage Server grid disk, use the CellCLI `DROP GRIDDISK` command.

2.3.2.1 Changing an Oracle Exadata Storage Server Grid Disk Name

Use the CellCLI interface to change the name of a grid disk.

- To change attributes of a grid disk, use the CellCLI `ALTER GRIDDISK` command.

Use the `DESCRIBE GRIDDISK` command to determine which Oracle Exadata Storage Server 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.

Example 2-2 Changing an Oracle Exadata Storage Server Grid Disk Name

Use the `ALTER GRIDDISK` command to rename a grid disk.

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

Related Topics

- Changing a Disk to Offline or Online
  You can change an Oracle ASM disk to `INACTIVE` or `ACTIVE`.
- ALTER GRIDDISK
- Naming Conventions for Oracle Exadata Storage Server Grid Disks
  Using a consistent naming convention helps to identify Exadata components.

2.3.2.2 Dropping an Oracle Exadata Storage Server Grid Disk

To drop an Oracle Exadata Storage Server 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. If you drop a grid disk that is still part of an ASM disk group, you may compromise data redundancy in the disk group or cause the disk group to dismount.

1. Drop the Oracle ASM disk from the disk group.

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

2. Make the corresponding grid disk inactive.

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

3. Drop the grid disk.

   ```
   CellCLI> DROP GRIDDISK disk_name
   ```

Example 2-3   Dropping a specific grid disk

After you have dropped the Oracle ASM disk from the disk group, you can drop the related grid disk.

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

Example 2-4   Dropping all grid disks

After you have dropped the Oracle ASM disks from the disk group, you can drop multiple grid disks using a single command.

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

Example 2-5   Using the FORCE option when dropping a grid disk

The `FORCE` option forces an active grid disk to be dropped. For example, if you cannot make a grid disk `INACTIVE`, but must drop the grid disk, you can use the `FORCE` option.

Use the `FORCE` option cautiously. If you drop a grid disk that is still part of an ASM disk group, you may compromise data redundancy in the disk group or cause the disk group to dismount.

```
CellCLI> DROP GRIDDISK data02_CD_04_cell01 FORCE
```

Related Topics

- DROP GRIDDISK

2.3.3 Resizing Grid Disks

You can resize grid disks and Oracle ASM disk groups to shrink one with excess free space and increase the size of another that is near capacity.

Initial configuration of Oracle Exadata disk group sizes is based on Oracle best practices and the location of the backup files.

- For internal backups: allocation of available space is 40% for the DATA disk groups, and 60% for the RECO disk groups.
- For external backups: allocation of available space is 80% for the DATA disk group, and 20% for the RECO disk group.

The disk group allocations can be changed after deployment. For example, the DATA disk group allocation may be too small at 60%, and need to be resized to 80%.

If your system has no free space available on the cell disks and one disk group, for example RECO, has plenty of free space, then you can resize the RECO disk group to a smaller size and reallocate the free space to the DATA disk group. The free space available after shrinking the RECO disk group is at a non-contiguous offset from the existing space allocations for the DATA disk group. Grid disks can use space anywhere on the cell disks and do not have to be contiguous.

If you are expanding the grid disks and the cell disks already have sufficient space to expand the existing grid disks, then you do not need to first resize an existing disk group. You would skip steps 2 and 3 below where the example shows the RECO disk group and grid disks are shrunk (you should still verify the cell disks have enough free space before growing the DATA grid disks). The amount of free space the administrator should reserve depends on the level of failure coverage.

If you are shrinking the size of the grid disks, you should understand how space is reserved for mirroring. Data is protected by Oracle ASM using normal or high redundancy to create one or two copies of data, which are stored as file extents. These copies are stored in separate failure groups. A failure in one failure group does not affect the mirror copies, so data is still accessible.

When a failure occurs, Oracle ASM re-mirrors, or **rebalances**, any extents that are not accessible so that redundancy is reestablished. For the re-mirroring process to succeed, sufficient free space must exist in the disk group to allow creation of the new file extent mirror copies. If there is not enough free space, then some extents will not be re-mirrored and the subsequent failure of the other data copies will require the disk group to be restored from backup. Oracle ASM sends an error when a re-mirror process fails due to lack of space.

You must be using Oracle Exadata System Software release 12.1.2.1.0 or higher, or have the patch for bug 19695225 applied to your software.

This procedure for resizing grid disks applies to bare metal and virtual machine (VM) deployments.

- Determine the Amount of Available Space
  To increase the size of the disks in a disk group you must either have unallocated disk space available, or you have to reallocate space currently used by a different disk group.
- Shrink the Oracle ASM Disks in the Donor Disk Group
  If there is no free space available on the cell disks, you can reduce the space used by one disk group to provide additional disk space for a different disk group.
- Shrink the Grid Disks in the Donor Disk Group
  After shrinking the disks in the Oracle ASM disk group, you then shrink the size of the grid disks on each cell.
- Increase the Size of the Grid Disks Using Available Space
  You can increase the size used by the grid disks if there is unallocated disk space either already available, or made available by shrinking the space used by a different Oracle ASM disk group.
- Increase the Size of the Oracle ASM Disks
  You can increase the size used by the Oracle ASM disks after increasing the space allocated to the associated grid disks.

Related Topics

- Understanding ASM Capacity and Reservation of Free Space in Exadata (My Oracle Support Doc ID 1551288.1)
- Bug 19695225 - Running Many Create or Alter Griddisk Commands Over Time Causes Cell Disk Metadata Corruption (ORA-600 [addNewSegmentsToGDisk_2]) and Loss of Cell Disk Content (My Oracle Support Doc ID 1991445.1)

2.3.3.1 Determine the Amount of Available Space

To increase the size of the disks in a disk group you must either have unallocated disk space available, or you have to reallocate space currently used by a different disk group.

You can also use a script available in "Script to Calculate New Grid Disk and Disk Group Sizes in Exadata (My Oracle Support Doc ID 1464809.1)" to assist in determining how much free space is available to shrink a disk group.

1. View the space currently used by the disk groups.

   ```
   SELECT name, total_mb, free_mb, total_mb - free_mb used_mb,
   round(100*free_mb/total_mb,2) pct_free
   FROM v$asm_diskgroup
   ORDER BY 1;

   NAME                           TOTAL_MB    FREE_MB    USED_MB   PCT_FREE
   ------------------------------ ---------- ---------- ---------- ----------
   DATAC1                           68812800    9985076   58827724      14.51
   RECOC1                           94980480   82594920   12385560      86.96
   ```

   The example above shows that the DATAC1 disk group has only about 15% of free space available while the RECOC1 disk group has about 87% free disk space. The `PCT_FREE` displayed here is raw free space, not usable free space. Additional space is needed for rebalancing operations.

2. For the disk groups you plan to resize, view the count and status of the failure groups used by the disk groups.

   ```
   SELECT dg.name, d.failgroup, d.state, d.header_status, d.mount_mode,
    d.mode_status, count(1) num_disks
   FROM V$ASM_DISK d, V$ASM_DISKGROUP dg
   WHERE d.group_number = dg.group_number
   AND dg.name IN ('RECOC1', 'DATAC1')
   GROUP BY dg.name, d.failgroup, d.state, d.header_status, d.mount_status,
     d.mode_status
   ORDER BY 1, 2, 3;

   NAME       FAILGROUP      STATE      HEADER_STATU MOUNT_S  MODE_ST
   NUM_DISKS
   ---------- -------------  ---------- ------------ -------- -------
   ---------
   DATAC1     EXA01CELADM01  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM02  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM03  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM04  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM05  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM06  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM07  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM08  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM09  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM10  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM11  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM12  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM13  NORMAL     MEMBER        CACHED  ONLINE   12
   DATAC1     EXA01CELADM14  NORMAL     MEMBER        CACHED  ONLINE   12
   RECOC1     EXA01CELADM01  NORMAL     MEMBER        CACHED  ONLINE   12
   RECOC1     EXA01CELADM02  NORMAL     MEMBER        CACHED  ONLINE   12
   RECOC1     EXA01CELADM03  NORMAL     MEMBER        CACHED  ONLINE   12
   RECOC1     EXA01CELADM04  NORMAL     MEMBER        CACHED  ONLINE   12
   RECOC1     EXA01CELADM05  NORMAL     MEMBER        CACHED  ONLINE   12
   ```

```
RECOC1      EXA01CELADM06  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM07  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM08  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM09  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM10  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM11  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM12  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM13  NORMAL     MEMBER        CACHED  ONLINE   12
RECOC1      EXA01CELADM14  NORMAL     MEMBER        CACHED  ONLINE   12
```

The above example is for a full rack, which has 14 cells and 14 failure groups for DATAC1 and RECOC1. Verify that each failure group has at least 12 disks in the `NORMAL` state (`num_disks`). If you see disks listed as `MISSING`, or you see an unexpected number of disks for your configuration, then do not proceed until you resolve the problem.

Extreme Flash systems should see a disk count of 8 instead of 12 for `num_disks`.

3. List the corresponding grid disks associated with each cell and each failure group, so you know which grid disks to resize.

```
SELECT dg.name, d.failgroup, d.path
FROM V$ASM_DISK d, V$ASM_DISKGROUP dg
WHERE d.group_number = dg.group_number
AND dg.name IN ('RECOC1', 'DATAC1')
ORDER BY 1, 2, 3;

NAME        FAILGROUP      PATH
----------- -------------  ----------------------------------------------
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_00_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_01_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_02_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_03_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_04_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_05_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_06_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_07_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_08_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_09_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_10_exa01celadm01
DATAC1      EXA01CELADM01  o/192.168.74.43/DATAC1_CD_11_exa01celadm01
DATAC1      EXA01CELADM02  o/192.168.74.44/DATAC1_CD_00_exa01celadm01
DATAC1      EXA01CELADM02  o/192.168.74.44/DATAC1_CD_01_exa01celadm01
DATAC1      EXA01CELADM02  o/192.168.74.44/DATAC1_CD_02_exa01celadm01
...
RECOC1      EXA01CELADM13  o/192.168.74.55/RECOC1_CD_00_exa01celadm13
RECOC1      EXA01CELADM13  o/192.168.74.55/RECOC1_CD_01_exa01celadm13
RECOC1      EXA01CELADM13  o/192.168.74.55/RECOC1_CD_02_exa01celadm13
...
RECOC1      EXA01CELADM14  o/192.168.74.56/RECOC1_CD_09_exa01celadm14
RECOC1      EXA01CELADM14  o/192.168.74.56/RECOC1_CD_10_exa01celadm14
RECOC1      EXA01CELADM14  o/192.168.74.56/RECOC1_CD_11_exa01celadm14

168 rows returned.
```

4. Check the cell disks for available free space.

Free space on the cell disks can be used to increase the size of the DATAC1 grid disks. If there is not enough available free space to expand the DATAC1 grid disks, then you must shrink the RECOC1 grid disks to provide the additional space for the desired new size of DATAC1 grid disks.

```
[root@exa01adm01 tmp]# dcli -g ~/cell_group -l root "cellcli -e list
celldisk \
  attributes name,freespace"
exa01celadm01: CD_00_exa01celadm01 0
exa01celadm01: CD_01_exa01celadm01 0
exa01celadm01: CD_02_exa01celadm01 0
exa01celadm01: CD_03_exa01celadm01 0
exa01celadm01: CD_04_exa01celadm01 0
exa01celadm01: CD_05_exa01celadm01 0
exa01celadm01: CD_06_exa01celadm01 0
exa01celadm01: CD_07_exa01celadm01 0
exa01celadm01: CD_08_exa01celadm01 0
exa01celadm01: CD_09_exa01celadm01 0
exa01celadm01: CD_10_exa01celadm01 0
exa01celadm01: CD_11_exa01celadm01 0
...
```

In this example, there is no free space available, so you must shrink the RECOC1 grid disks first to provide space for the DATAC1 grid disks. In your configuration there might be plenty of free space available and you can use that free space instead of shrinking the RECOC1 grid disks.

5. Calculate the amount of space to shrink from the RECOC1 disk group and from each grid disk.

   The minimum size to safely shrink a disk group and its grid disks must take into account the following:

   - Space currently in use (`USED_MB`)
   - Space expected for growth (`GROWTH_MB`)
   - Space needed to rebalance in case of disk failure (`DFC_MB`), typically 15% of total disk group size

   The minimum size calculation taking the above factors into account is:

   ```
   Minimum DG size (MB) = ( USED_MB + GROWTH_MB ) * 1.15
   ```

   - `USED_MB` can be derived from V$ASM_DISKGROUP by calculating `TOTAL_MB - FREE_MB`
   - `GROWTH_MB` is an estimate specific to how the disk group will be used in the future and should be based on historical patterns of growth

   For the RECOC1 disk group space usage shown in step 1, we see the minimum size it can shrink to assuming no growth estimates is:

   Minimum RECOC1 size = (TOTAL_MB - FREE_MB + GROWTH_MB) * 1.15

   = (94980480 - 82594920 + 0) * 1.15 = 14243394 MB = 13,910 GB

   In the example output shown in Step 1, RECOC1 has plenty of free space and DATAC1 has less than 15% free. So, you could shrink RECOC1 and give the freed disk space to DATAC1. If you decide to reduce RECOC1 to half of its current size, the new size is

94980480 / 2 = 47490240 MB. This size is significantly above the minimum size we calculated for the RECOC1 disk group above, so it is safe to shrink it down to this value.

The query in Step 2 shows that there are 168 grid disks for RECOC1, because there are 14 cells and 12 disks per cell (14 * 12 = 168). The estimated new size of each grid disk for the RECOC1 disk group is 47490240 / 168, or 282,680 MB.

Find the closest 16 MB boundary for the new grid disk size. If you do not perform this check, then the cell will round down the grid disk size to the nearest 16 MB boundary automatically, and you could end up with a mismatch in size between the Oracle ASM disks and the grid disks.

```
SQL> SELECT 16*TRUNC(&new_disk_size/16) new_disk_size FROM dual;
Enter value for new_disk_size: 282680

NEW_DISK_SIZE
-------------
       282672
```

Based on the above result, you should choose 282672 MB as the new size for the grid disks in the RECOC1 disk group. After resizing the grid disks, the size of the RECOC1 disk group will be 47488896 MB.

6. Calculate how much to increase the size of each grid disk in the DATAC1 disk group.

   Ensure the Oracle ASM disk size and the grid disk sizes match across the entire disk group. The following query shows the combinations of disk sizes in each disk group. Ideally, there is only one size found for all disks and the sizes of both the Oracle ASM (`total_mb`) disks and the grid disks (`os_mb`) match.

```
SELECT dg.name, d.total_mb, d.os_mb, count(1) num_disks
FROM v$asm_diskgroup dg, v$asm_disk d
WHERE dg.group_number = d.group_number
GROUP BY dg.name, d.total_mb, d.os_mb;

NAME                             TOTAL_MB      OS_MB  NUM_DISKS
------------------------------ ---------- ---------- ----------
DATAC1                             409600     409600        168
RECOC1                             565360     565360        168
```

   After shrinking RECOC1's grid disks, the following space is left per disk for DATAC1:

   Additional space for DATAC1 disks = *RECOC1_current_size* - *RECOC1_new_size*
   = 565360 - 282672 = 282688 MB

   To calculate the new size of the grid disks for the DATAC1 disk group, use the following:

   DATAC1's disks new size = *DATAC1_ disks_current_size* + *new_free_space_from_RECOC1*
   = 409600 + 282688 = 692288 MB

   Find the closest 16 MB boundary for the new grid disk size. If you do not perform this check, then the cell will round down the grid disk size to the nearest 16 MB boundary automatically, and you could end up with a mismatch in size between the Oracle ASM disks and the grid disks.

```
SQL> SELECT 16*TRUNC(&new_disk_size/16) new_disk_size FROM dual;
Enter value for new_disk_size: 692288
```

```
NEW_DISK_SIZE
-------------
       692288
```

Based on the query result, you can use the calculated size of 692288 MB for the disks in the DATAC1 disk groups because the size is on a 16 MB boundary. If the result of the query is different from the value you supplied, then you must use the value returned by the query because that is the value to which the cell will round the grid disk size.

The calculated value of the new grid disk size will result in the DATAC1 disk group having a total size of 116304384 MB (168 disks * 692288 MB).

2.3.3.2 Shrink the Oracle ASM Disks in the Donor Disk Group

If there is no free space available on the cell disks, you can reduce the space used by one disk group to provide additional disk space for a different disk group.

This task is a continuation of an example where space in the RECOC1 disk group is being reallocated to the DATAC1 disk group.

Before resizing the disk group, make sure the disk group you are taking space from has sufficient free space.

1. Shrink the Oracle ASM disks for the RECO disk group down to the new desired size for all disks.

   Use the new size for the disks in the RECO disk group that was calculated in Step 5 of Determine the Amount of Available Space.

   ```
   SQL> ALTER DISKGROUP recoc1 RESIZE ALL SIZE 282672M REBALANCE POWER 64;
   ```

   Note:

   The `ALTER DISKGROUP` command may take several minutes to complete. The SQL prompt will not return until this operation has completed.

   If the specified disk group has quorum disks configured within the disk group, then the `ALTER DISKGROUP ... RESIZE ALL` command could fail with error `ORA-15277`. Quorum disks are configured if the requirements specified in Managing Quorum Disks for High Redundancy Disk Groups are met.

   As a workaround, for regular storage server failure groups (`FAILGROUP_TYPE=REGULAR`, not `QUORUM`), you can specify the failure group names explicitly in the SQL command, for example:

   ```
   SQL> ALTER DISKGROUP recoc1 RESIZE DISKS IN FAILGROUP exacell01
   SIZE 282672M,
   exacell02 SIZE 282672M, exacell03 SIZE 282672M REBALANCE POWER 64;
   ```

   Wait for rebalance to finish by checking the view GV$ASM_OPERATION.

   ```
   SQL> set lines 250 pages 1000
   SQL> col error_code form a10
   ```

```
SQL> SELECT dg.name, o.*
  2  FROM gv$asm_operation o, v$asm_diskgroup dg
  3  WHERE o.group_number = dg.group_number;
```

Proceed to the next step ONLY when the query against GV$ASM_OPERATION shows no rows for the disk group being altered.

2. Verify the new size of the ASM disks using the following queries:

```
SQL> SELECT name, total_mb, free_mb, total_mb - free_mb used_mb,
  2   ROUND(100*free_mb/total_mb,2) pct_free
  3  FROM v$asm_diskgroup
  4  ORDER BY 1;

NAME                             TOTAL_MB    FREE_MB    USED_MB   PCT_FREE
------------------------------ ---------- ---------- ---------- ----------
DATAC1                           68812800    9985076   58827724      14.51
RECOC1                           47488896   35103336   12385560      73.92

SQL> SELECT dg.name, d.total_mb, d.os_mb, COUNT(1) num_disks
  2  FROM v$asm_diskgroup dg, v$asm_disk d
  3  WHERE dg.group_number = d.group_number
  4  GROUP BY dg.name, d.total_mb, d.os_mb;

NAME                             TOTAL_MB      OS_MB  NUM_DISKS
------------------------------ ---------- ---------- ----------
DATAC1                             409600     409600        168
RECOC1                             282672     565360        168
```

The above query example shows that the disks in the RECOC1 disk group have been resized to a size of 282672 MG each, and the total disk group size is 47488896 MB.

2.3.3.3 Shrink the Grid Disks in the Donor Disk Group

After shrinking the disks in the Oracle ASM disk group, you then shrink the size of the grid disks on each cell.

This task is a continuation of an example where space in the RECOC1 disk group is being reallocated to the DATAC1 disk group.

You must have first completed the task Shrink the Oracle ASM Disks in the Donor Disk Group.

1. Shrink the grid disks associated with the RECO disk group on all cells down to the new, smaller size.

   For each storage cell identified in Determine the Amount of Available Space in Step 3, shrink the grid disks to match the size of the Oracle ASM disks that were shrunk in the previous task. Use commands similar to the following:

```
dcli -c exa01celadm01 -l root "cellcli -e alter griddisk
RECOC1_CD_00_exa01celadm01 \
,RECOC1_CD_01_exa01celadm01 \
,RECOC1_CD_02_exa01celadm01 \
,RECOC1_CD_03_exa01celadm01 \
,RECOC1_CD_04_exa01celadm01 \
,RECOC1_CD_05_exa01celadm01 \
,RECOC1_CD_06_exa01celadm01 \
,RECOC1_CD_07_exa01celadm01 \
,RECOC1_CD_08_exa01celadm01 \
,RECOC1_CD_09_exa01celadm01 \
,RECOC1_CD_10_exa01celadm01 \
,RECOC1_CD_11_exa01celadm01 \
size=282672M "

dcli -c exa01celadm02 -l root "cellcli -e alter griddisk
RECOC1_CD_00_exa01celadm02 \
,RECOC1_CD_01_exa01celadm02 \
,RECOC1_CD_02_exa01celadm02 \
,RECOC1_CD_03_exa01celadm02 \
,RECOC1_CD_04_exa01celadm02 \
,RECOC1_CD_05_exa01celadm02 \
,RECOC1_CD_06_exa01celadm02 \
,RECOC1_CD_07_exa01celadm02 \
,RECOC1_CD_08_exa01celadm02 \
,RECOC1_CD_09_exa01celadm02 \
,RECOC1_CD_10_exa01celadm02 \
,RECOC1_CD_11_exa01celadm02 \
size=282672M "

...

dcli -c exa01celadm14 -l root "cellcli -e alter griddisk
RECOC1_CD_00_exa01celadm14 \
,RECOC1_CD_01_exa01celadm14 \
,RECOC1_CD_02_exa01celadm14 \
,RECOC1_CD_03_exa01celadm14 \
,RECOC1_CD_04_exa01celadm14 \
,RECOC1_CD_05_exa01celadm14 \
,RECOC1_CD_06_exa01celadm14 \
,RECOC1_CD_07_exa01celadm14 \
,RECOC1_CD_08_exa01celadm14 \
,RECOC1_CD_09_exa01celadm14 \
,RECOC1_CD_10_exa01celadm14 \
,RECOC1_CD_11_exa01celadm14 \
size=282672M "
```

2. Verify the new size of the grid disks using the following command:

```
[root@exa01adm01 tmp]# dcli -g cell_group -l root "cellcli -e list griddisk
attributes name,size where name like \'RECOC1.*\' "

exa01celadm01: RECOC1_CD_00_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_01_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_02_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_03_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_04_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_05_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_06_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_07_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_08_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_09_exa01celadm01 276.046875G
exa01celadm01: RECOC1_CD_10_exa01celadm01 276.046875G
```

```
exa01celadm01: RECOC1_CD_11_exa01celadm01 276.046875G
...
```

The above example shows that the disks in the RECOC1 disk group have been resized to a size of 282672 MB each (276.046875 * 1024).

2.3.3.4 Increase the Size of the Grid Disks Using Available Space

You can increase the size used by the grid disks if there is unallocated disk space either already available, or made available by shrinking the space used by a different Oracle ASM disk group.

This task is a continuation of an example where space in the RECOC1 disk group is being reallocated to the DATAC1 disk group. If you already have sufficient space to expand an existing disk group, then you do not need to reallocate space from a different disk group.

1. Check that the cell disks have the expected amount of free space.

   After completing the tasks to shrink the Oracle ASM disks and the grid disks, you would expect to see the following free space on the cell disks:

   ```
   [root@exa01adm01 tmp]# dcli -g ~/cell_group -l root "cellcli -e list
   celldisk \
   attributes name,freespace"

   exa01celadm01: CD_00_exa01celadm01 276.0625G
   exa01celadm01: CD_01_exa01celadm01 276.0625G
   exa01celadm01: CD_02_exa01celadm01 276.0625G
   exa01celadm01: CD_03_exa01celadm01 276.0625G
   exa01celadm01: CD_04_exa01celadm01 276.0625G
   exa01celadm01: CD_05_exa01celadm01 276.0625G
   exa01celadm01: CD_06_exa01celadm01 276.0625G
   exa01celadm01: CD_07_exa01celadm01 276.0625G
   exa01celadm01: CD_08_exa01celadm01 276.0625G
   exa01celadm01: CD_09_exa01celadm01 276.0625G
   exa01celadm01: CD_10_exa01celadm01 276.0625G
   exa01celadm01: CD_11_exa01celadm01 276.0625G
   ...
   ```

2. For each storage cell, increase the size of the DATA grid disks to the desired new size.

   Use the size calculated in Determine the Amount of Available Space.

   ```
   dcli -c exa01celadm01 -l root "cellcli -e alter griddisk
   DATAC1_CD_00_exa01celadm01 \
   ,DATAC1_CD_01_exa01celadm01 \
   ,DATAC1_CD_02_exa01celadm01 \
   ,DATAC1_CD_03_exa01celadm01 \
   ,DATAC1_CD_04_exa01celadm01 \
   ,DATAC1_CD_05_exa01celadm01 \
   ,DATAC1_CD_06_exa01celadm01 \
   ,DATAC1_CD_07_exa01celadm01 \
   ,DATAC1_CD_08_exa01celadm01 \
   ,DATAC1_CD_09_exa01celadm01 \
   ,DATAC1_CD_10_exa01celadm01 \
   ,DATAC1_CD_11_exa01celadm01 \
   size=692288M "
   ```

```
...
dcli -c exa01celadm14 -l root "cellcli -e alter griddisk
DATAC1_CD_00_exa01celadm14 \
,DATAC1_CD_01_exa01celadm14 \
,DATAC1_CD_02_exa01celadm14 \
,DATAC1_CD_03_exa01celadm14 \
,DATAC1_CD_04_exa01celadm14 \
,DATAC1_CD_05_exa01celadm14 \
,DATAC1_CD_06_exa01celadm14 \
,DATAC1_CD_07_exa01celadm14 \
,DATAC1_CD_08_exa01celadm14 \
,DATAC1_CD_09_exa01celadm14 \
,DATAC1_CD_10_exa01celadm14 \
,DATAC1_CD_11_exa01celadm14 \
size=692288M "
```

3. Verify the new size of the grid disks associated with the DATAC1 disk group using the following command:

```
dcli -g cell_group -l root "cellcli -e list griddisk attributes name,size
\
where name like \'DATAC1.*\' "

exa01celadm01: DATAC1_CD_00_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_01_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_02_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_03_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_04_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_05_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_06_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_07_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_08_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_09_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_10_exa01celadm01 676.0625G
exa01celadm01: DATAC1_CD_11_exa01celadm01 676.0625G
```

Instead of increasing the size of the DATA disk group, you could instead create new disk groups with the new free space or keep it free for future use. In general, Oracle recommends using the smallest number of disk groups needed (typically DATA, RECO, and DBFS_DG) to give the greatest flexibility and ease of administration. However, there may be cases, perhaps when using virtual machines or consolidating many databases, where additional disk groups or available free space for future use may be desired.

If you decide to leave free space on the grid disks in reserve for future use, please see the My Oracle Support Note 1684112.1 for the steps on how to allocate free space to an existing disk group at a later time.

Related Topics

- How to resize ASM disks in Exadata (My Oracle Support Doc ID 1684112.1)
- Determine the Amount of Available Space
  To increase the size of the disks in a disk group you must either have unallocated disk space available, or you have to reallocate space currently used by a different disk group.

2.3.3.5 Increase the Size of the Oracle ASM Disks

You can increase the size used by the Oracle ASM disks after increasing the space allocated to the associated grid disks.

This task is a continuation of an example where space in the RECOC1 disk group is being reallocated to the DATAC1 disk group.

You must have completed the task of resizing the grid disks before you can resize the corresponding Oracle ASM disk group.

1. Increase the Oracle ASM disks for DATAC1 disk group to the new size of the grid disks on the storage cells.

   ```
   SQL> ALTER DISKGROUP datac1 RESIZE ALL;
   ```

   This command resizes the Oracle ASM disks to match the size of the grid disks.

   > **Note:**
   >
   > If the specified disk group has quorum disks configured within the disk group, then the `ALTER DISKGROUP ... RESIZE ALL` command could fail with error `ORA-15277`. Quorum disks are configured if the requirements specified in *Oracle Exadata Database Machine Maintenance Guide* are met.
   >
   > As a workaround, for regular storage server failure groups (`FAILGROUP_TYPE=REGULAR`, not `QUORUM`), you can specify the failure group names explicitly in the SQL command, for example:
   >
   > ```
   > SQL> ALTER DISKGROUP datac1 RESIZE DISKS IN FAILGROUP exacell01,
   > exacell02, exacell03;
   > ```

2. Wait for the rebalance operation to finish.

   ```
   SQL> set lines 250 pages 1000
   SQL> col error_code form a10
   SQL> SELECT dg.name, o.* FROM gv$asm_operation o, v$asm_diskgroup dg
        WHERE o.group_number = dg.group_number;
   ```

   Do not continue to the next step until the query returns zero rows for the disk group that was altered.

3. Verify that the new sizes for the Oracle ASM disks and disk group is at the desired sizes.

   ```
   SQL> SELECT name, total_mb, free_mb, total_mb - free_mb used_mb,
        ROUND(100*free_mb/total_mb,2) pct_free
        FROM v$asm_diskgroup
        ORDER BY 1;

   NAME                             TOTAL_MB    FREE_MB    USED_MB   PCT_FREE
   ------------------------------ ---------- ---------- ---------- ----------
   DATAC1                          116304384   57439796   58864588      49.39
   ```

```
RECOC1                           47488896   34542516   12946380      72.74

SQL> SELECT dg.name, d.total_mb, d.os_mb, COUNT(1) num_disks
     FROM  v$asm_diskgroup dg, v$asm_disk d
     WHERE dg.group_number = d.group_number
     GROUP BY dg.name, d.total_mb, d.os_mb;

NAME                              TOTAL_MB      OS_MB  NUM_DISKS
------------------------------ ---------- ---------- ----------
DATAC1                              692288     692288        168
RECOC1                              282672     282672        168
```

The results of the queries show that the RECOC1 and DATAC1 disk groups and disk have been resized.

Related Topics

- Determine the Amount of Available Space
  To increase the size of the disks in a disk group you must either have unallocated disk space available, or you have to reallocate space currently used by a different disk group.

2.3.4 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 2-6 Determining Grid Disks in an Oracle ASM Disk Group

This example shows how to find the Oracle ASM disk group that contains grid disks that begin with `DATA0`, for example `DATA0_CD_03_CELL04`.

```
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
```

2.3.5 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.

You can use the CellCLI utility to search for specific grid disk names.

Example 2-7 Displaying Oracle ASM Disk Group Names

This example shows how to use a SQL command to display the Oracle ASM disk group names on the Oracle ASM instance.

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

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

Example 2-8 Searching for Grid Disks by Name

This example shows how to display similar grid disk group names on the cell using the `dcli` utility.

```
$ ./dcli "cellcli -e list griddisk where -c cell04"

data0_CD_01_cell04
data0_CD_02_cell04
data0_CD_03_cell04
...
```

Related Topics

- Naming Conventions for Oracle Exadata Storage Server Grid Disks
  Using a consistent naming convention helps to identify Exadata components.

2.3.6 Handling Disk Replacement

If a disk has a problem, the physical disk status changes.

When a physical disk is removed, its status becomes `not present`. Oracle ASM may take a grid disk offline when getting I/O errors while trying to access a grid disk on the physical disk. When the physical disk is replaced, Oracle Exadata System Software automatically puts the grid disks on the physical disk online in their respective Oracle ASM disk groups. If a grid disk remains offline longer than the time specified by the `disk_repair_time` attribute, then Oracle ASM force drops that grid disk and starts a rebalance to restore data redundancy. Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation is complete.

The following table summarizes the physical disk statuses, and how Oracle ASM handles grid disks when the physical disk has a problem.

Table 2-1 Physical Disk Status

Physical Disk Status	Oracle Exadata System Software Action
`normal` Disk is functioning normally	No action.
`not present` Disk has been removed	Oracle Exadata System Software offlines disk, then uses the `DROP ... FORCE` command after `disk_repair_time` limit exceeded. The rebalance operation begins.
`predictive failure` Disk is having problems, and may fail	Oracle Exadata System Software drops the grid disks on the affected physical disk without the `FORCE` option from Oracle ASM, and the rebalance operation copies the data on the affected physical disk to other disks. After all grid disks have been successfully removed from their respective Oracle ASM disk groups, administrators can proceed with disk replacement.
`critical` Disk has failed	Oracle Exadata System Software drops the grid disks using the `DROP ... FORCE` command on the affected physical disk from Oracle ASM, and the rebalance operation restores data redundancy. Administrators can proceed with disk replacement immediately. This status is only available for releases 11.2.3.1.1 and earlier.
`poor performance` Disk is performing poorly	Oracle Exadata System Software attempts to drop the grid disks using the `FORCE` option on the affected physical disk from Oracle ASM. If the `DROP ... FORCE` command is successful, then the rebalance operation begins to restore data redundancy and administrators can proceed with disk replacement immediately. If the `DROP ... FORCE` command fails due to offline partners, Oracle Exadata System Software drops the grid disks on the affected physical disk without the `FORCE` option from Oracle ASM, and the rebalance operation copies the data on the affected physical disk to other disks. After all grid disks have been successfully removed from their respective Oracle ASM disk groups, administrators can proceed with disk replacement.

After a physical disk is replaced, Oracle Exadata System Software automatically creates the grid disks on the replacement disk, and adds them to the respective Oracle ASM disk groups. An Oracle ASM rebalance operation relocates data to the newly-added grid disks. Oracle ASM monitors the rebalance operation, and Oracle Exadata System Software sends an e-mail message when the operation is complete.

Related Topics

- Maintaining the Hard Disks of Exadata Storage Servers

3

Administering Oracle Database on Exadata

3.1 Administering 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 Exadata storage servers. This feature is controlled by database initialization parameters:

- CELL_OFFLOAD_PROCESSING
  The CELL_OFFLOAD_PROCESSING initialization parameter enables SQL processing offload to Oracle Exadata Storage Server.
- 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 Oracle Exadata System Software as `STORAGE` predicates for a given SQL command.
- CELL_OFFLOAD_DECRYPTION

3.1.1 CELL_OFFLOAD_PROCESSING

The CELL_OFFLOAD_PROCESSING initialization parameter enables SQL processing offload to Oracle Exadata Storage Server.

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.

- 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 Oracle Exadata Storage Server with conventional storage. Even when CELL_OFFLOAD_PROCESSING is set to `FALSE`, Oracle Exadata Storage Server has many advantages over conventional storage. Oracle Exadata Storage Server is highly optimized for fast processing of large queries. It has no bottlenecks at the controller or other levels inside the cell. Oracle Exadata uses a modern scale-out architecture and state-of-the-art RDMA Network Fabric that has much higher throughput than conventional storage networks. Oracle Exadata is tightly integrated with the Oracle Database, and has unique capabilities for setup, execution, monitoring, diagnostics, resource management, and corruption prevention.

Related Topics

- *Oracle Database SQL Language Reference*

3.1.2 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 Oracle Exadata System Software 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 Oracle Exadata System Software, whether or not Oracle Exadata System Software is present or the table is on the cell. You can use this setting to see what can be offloaded to Oracle Exadata Storage Server before migrating to Oracle Exadata Storage Server.
- `NEVER` produces no changes to the SQL `EXPLAIN PLAN` command for Oracle Exadata System Software.

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;
```

Related Topics

- Monitoring Smart I/O Using SQL Explain Plan

3.1.3 CELL_OFFLOAD_DECRYPTION

The CELL_OFFLOAD_DECRYPTION initialization parameter enables decryption offload to Oracle Exadata Storage Servers. This decryption applies to both encrypted tablespaces and encrypted columns. When the value of the parameter is set to `TRUE`, decryption can be offloaded to cells. The default value of the parameter is `TRUE`. If the parameter is set to `FALSE` at the system level, then the database performs all decryption with cells serving blocks. You can set CELL_OFFLOAD_DECRYPTION dynamically with the SQL `ALTER SYSTEM` command. For example:

```
SQL> ALTER SYSTEM SET CELL_OFFLOAD_DECRYPTION = FALSE;
```

3.2 Administering Exadata Smart Flash Cache

Exadata Smart Flash Cache automatically caches frequently-used data in high-performance flash memory.

- Overriding the Default Caching Policy
- Administering In-Memory Columnar Caching
- Optimizing Exadata Smart Flash Cache for Analytical Workloads
- Manually Populating Exadata Smart Flash Cache

3.2.1 Overriding the Default Caching Policy

Though typically not required, you can use the `CELL_FLASH_CACHE` segment storage option to override the automatic caching policy for Exadata Smart Flash Cache. The `CELL_FLASH_CACHE` option may be included in the `STORAGE` clause as part of the SQL `CREATE` or `ALTER` command for various database objects, such as regular tables and indexes, partitioned tables and indexes, and index-organized tables.

For example:

```
SQL> CREATE TABLE t1 (c1 number, c2 varchar2(200)) STORAGE (CELL_FLASH_CACHE
NONE);


SQL> ALTER TABLE t2 STORAGE (CELL_FLASH_CACHE KEEP);
```

The `CELL_FLASH_CACHE` option supports the following settings:

- `NONE`: This value ensures that Exadata Smart Flash Cache never caches the corresponding segment. By using this setting on peripheral database segments, more cache space is available for more-important and frequently-accessed database segments.
- `DEFAULT`: This value specifies that database segments are cached using the default LRU (least recently used) algorithm of Exadata Smart Flash Cache. This value is the default setting for `CELL_FLASH_CACHE`.
- `KEEP`: This value elevates the segment priority in Exadata Smart Flash Cache. By using this setting, you can increase the likelihood of keeping data from the corresponding segment in the cache.

Starting with Oracle Exadata System Software release 24.1.0 in conjunction with Oracle Database 23ai, segments with this setting are automatically populated into Exadata Smart Flash Cache. Previously, the segment data was populated when the segment was read.

You can set the `CELL_FLASH_CACHE` segment storage option separately on each partition in a partitioned segment, which is particularly useful if you want to influence the caching priority for different partitions based on predictable usage patterns. While setting the `CELL_FLASH_CACHE` segment storage option on a partition, you can add the `DEFERRED INVALIDATION` clause. For example:

```
SQL> ALTER TABLE ptable MODIFY PARTITION p1 STORAGE (CELL_FLASH_CACHE KEEP)
DEFERRED INVALIDATION;
```

By using this option, you can dynamically modify the segment storage option without immediately invalidating dependent cursors. This option requires Oracle Database software that contains the patch for bug 33456703, which is included in Oracle Database version 19.15, Oracle Database version 21.6, and later releases.

Example 3-1   Setting CELL_FLASH_CACHE on partitions

This example shows setting `CELL_FLASH_CACHE` individually on multiple partitions in a `CREATE TABLE` command.

```
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));
```

Example 3-2   Setting CELL_FLASH_CACHE on a LOB segment

This example shows setting `CELL_FLASH_CACHE` for a LOB segment in a `CREATE TABLE` command.

```
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);
```

Example 3-3   Using ALTER TABLE with CELL_FLASH_CACHE

For objects where altering the storage clause is allowed, the `ALTER` command can be used with `CELL_FLASH_CACHE`, as shown in these examples.

```
ALTER TABLE tkbcsrbc STORAGE(CELL_FLASH_CACHE KEEP);

ALTER TABLE tkbcsrbc MODIFY LOB (l1) (STORAGE (CELL_FLASH_CACHE DEFAULT));
```

Example 3-4 Using Views to Query CELL_FLASH_CACHE Storage Clause

The `CELL_FLASH_CACHE` storage clause attribute can be queried using database views based on the object involved.

```
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';
```

3.2.2 Administering In-Memory Columnar Caching

The columnar cache is a section of Exadata Smart Flash Cache that stores data in columnar format. When directed by Oracle Database, Exadata automatically uses a portion of the columnar cache to maintain data using the Oracle Database In-Memory format. You do not need to configure anything on Exadata to use this enhancement.

This feature is available if you have licensed the Oracle Database In-Memory option. To enable this feature, use either of the following database instance parameters:

- Set the `INMEMORY_SIZE` database instance parameter to a value greater than zero.
- Starting with Oracle Database version 19.8.0.0.200714, you can set `INMEMORY_FORCE=cellmemory_level`.

  This option enables you to use In-Memory Columnar Caching in Exadata Smart Flash Cache without a dedicated In-Memory cache in the database instance.

You can use the `CELLMEMORY` segment option to override of the default behavior for In-Memory Columnar Caching in Exadata Smart Flash Cache:

```
SQL> ALTER TABLE table_name  [ [ NO ] CELLMEMORY [ MEMCOMPRESS FOR [ QUERY |
CAPACITY ] [ LOW | HIGH ] ]
```

Options and Clauses	Usage Description
`NO CELLMEMORY`	Indicates a table is ineligible for the rewrite from 12.1.0.2 columnar flash cache format into the 12.2 Database In-Memory format.
`CELLMEMORY` and `CELLMEMORY MEMCOMPRESS FOR CAPACITY`	Allows a table to be cached in the default Oracle Database 12.2 In-Memory format. You only need to use this clause to undo a previously specified `NO CELLMEMORY` statement or to change the specified compression level.
`CELLMEMORY MEMCOMPRESS FOR QUERY`	This option indicates that In-Memory column store data should be compressed less than if `MEMCOMPRESS FOR CAPACITY` was specified. This option can provide an increase in query performance but requires almost twice as much flash space.
`LOW` and `HIGH`	Not implemented at this time.

Example 3-5 Using CELLMEMORY and INMEMORY Options on the Same Table

You can use both `INMEMORY` and `CELLMEMORY` on the same table. For example:

```
CREATE TABLE t (c1 NUMBER) INMEMORY CELLMEMORY MEMCOMPRESS FOR QUERY;
```

Specifying both options is useful for when you have a low priority table that is unlikely to get loaded in memory. By also specifying `CELLMEMORY` you still get columnar performance.

3.2.3 Optimizing Exadata Smart Flash Cache for Analytical Workloads

By default, Exadata Smart Flash Cache favors caching of frequently accessed data and limits caching of data with low potential for reuse. Specifically, Exadata Smart Flash Cache limits the amount of cache space occupied by large I/Os with low potential for reuse, which includes I/Os associated with temporary segments. Furthermore, if thrashing (caching of data recently removed from the cache) is detected, additional cache space is created by removing data associated with large I/Os (including temporary segment data).

This approach naturally suits online transaction processing (OLTP) workloads. However, analytical workloads typically under-utilize the resources of Exadata Smart Flash Cache when processing large join and sort operations with large temporary segments.

To better utilize Exadata Smart Flash Cache for analytical workloads, Exadata allows administrators to specify the main workload type for each database and pluggable database (PDB) by setting the `main_workload_type` database instance parameter.

For example:

- In a container database (CDB) or non-CDB:

  ```
  SQL> alter system set main_workload_type = [ OLTP | ANALYTICS ];
  ```

  In a PDB:

  ```
  SQL> alter session container = <pdb_name>;
  SQL> alter system set main_workload_type = [ OLTP | ANALYTICS ];
  ```

The available options are:

- `OLTP`: Exadata Smart Flash Cache favors caching of frequently accessed data and limits caching of data with low potential for reuse.
- `ANALYTICS`: Exadata Smart Flash Cache relaxes the limits on caching data with low potential for reuse, enabling temporary segments to occupy most of the cache.

Note the following details regarding this capability:

- To enable this capability, the Exadata I/O resource management plan (IORMPLAN) must contain the associated database as part of the interdatabase plan (DBPLAN), and the plan directive must include a flash cache quota specified using the `flashCacheSize` attribute. For example, on a system using the following IORMPLAN, the `main_workload_type` setting only takes effect in the `sales` database and any PDB it contains:

  ```
  CellCLI> ALTER IORMPLAN dbplan=((name=sales, share=8, flashCacheSize=20G),
  -
                                  (name=finance, share=8,
  flashCacheLimit=10G, flashCacheMin=2G), -
                                  (name=dev, share=2, flashCacheLimit=4G,
  flashCacheMin=1G), -
                                  (name=test, share=1))
  ```

- The `main_workload_type` setting can differ across PDBs or between the CDB and any PDB. Any PDB-level setting overrides the CDB setting.

- If `main_workload_type` is not set in a PDB, the corresponding CDB setting is used. If `main_workload_type` is not set in a CDB, the default setting is `OLTP`.
- Minimum system requirements:
  - Exadata System Software release 22.1.12.0.0 or 23.1.3.0.0.
  - Oracle Database 19c with patch 35017301.

3.2.4 Manually Populating Exadata Smart Flash Cache

By default, Exadata Smart Flash Cache is automatically populated when reading data from disk, which is optimal for most application scenarios. However, if your application is sensitive to the increased I/O latency of the initial disk reads, you can manually populate Exadata Smart Flash Cache. This topic describes approaches and considerations for manually populating Exadata Smart Flash Cache.

Scanning Data

If Exadata Smart Flash Cache is not full, you can read data into the cache by simply accessing the desired data, usually through a full table scan.

To determine whether Exadata Smart Flash Cache has available space, compare the `FC_BY_ALLOCATED` metric with the `effectiveFlashCacheSize` attribute. For example, the following output shows that each cell contains approximately 4 TB of available space.

```
# FC_BY_ALLOCATED shows that each cell contains approximately 19 TB of data
allocated
$ dcli -g cell_group cellcli -e list metriccurrent where
name\=\"FC_BY_ALLOCATED\"
db01celadm01: FC_BY_ALLOCATED        FLASHCACHE      19,313,940 MB
db01celadm02: FC_BY_ALLOCATED        FLASHCACHE      19,311,784 MB
db01celadm03: FC_BY_ALLOCATED        FLASHCACHE      19,311,688 MB

# effectiveCacheSize shows that each cell contains approximately 23 TB of
flash cache space
$ dcli -g cell_group cellcli -e list flashcache attributes effectiveCacheSize
detail
db01celadm01: effectiveCacheSize:     23.28692626953125T
db01celadm02: effectiveCacheSize:     23.28692626953125T
db01celadm03: effectiveCacheSize:     23.28692626953125T
```

You can use the following query to confirm whether the desired data is populated in Exadata Smart Flash Cache.

```
select name,value
  from v$statname n,
       v$mystat s
  where s.statistic# = n.statistic#
    and name in ('physical read IO requests','physical read requests
optimized')
  order by name;
```

For example:

```
-- get session statistics
SQL> select name,value
  2    from v$statname n,
  3         v$mystat s
  4    where s.statistic# = n.statistic#
  5      and name in ('physical read IO requests','physical read requests
optimized')
  6    order by name;

NAME                                                             VALUE
---------------------------------------------------------------- ----------
physical read IO requests                                                 5
physical read requests optimized                                         11

// run the desired workload...

-- get session statistics again
SQL> select name,value
  2    from v$statname n,
  3         v$mystat s
  4    where s.statistic# = n.statistic#
  5      and name in ('physical read IO requests','physical read requests
optimized')
  6    order by name;

NAME                                                             VALUE
---------------------------------------------------------------- ----------
physical read IO requests                                            45,193
physical read requests optimized                                     23,140
```

In the previous example, the workload performed 45188 physical read requests (45193 - 5), of which 23129 (23140 - 11) are optimized. In this case, 22059 (45188 - 23129) unoptimized (disk) reads are performed.

When you repeat the workload, all of the reads should be optimized (`physical read IO requests` = `physical read requests optimized`), which indicates that all of the desired data is populated in Exadata Smart Flash Cache.

Using `flashCacheMin`

If Exadata Smart Flash Cache is fully populated in a consolidated environment, you can use the I/O Resource Management (IORM) `flashCacheMin` setting to free up space so that the desired data can be read into the cache. In this case, the `flashCacheMin` setting must be larger than the current space occupied by the database with sufficient space to add the desired data.

Note:

When using a container database (CDB), a CDB resource plan is required to govern the resource allocation to each pluggable database (PDB), even if there is only one PDB in the CDB.

To determine the current Exadata Smart Flash Cache space allocation for each database, examine the `DB_FC_BY_ALLOCATED` metric. For example:

```
$ dcli -g cell_group cellcli -e list metriccurrent where
name\=\"DB_FC_BY_ALLOCATED\"
db01celadm01: DB_FC_BY_ALLOCATED     ASM                    0.000 MB
db01celadm01: DB_FC_BY_ALLOCATED     DBCDB1             1,694,241 MB
db01celadm01: DB_FC_BY_ALLOCATED     DBCDB2             4,851,611 MB
db01celadm01: DB_FC_BY_ALLOCATED     DBCDB3             4,638,129 MB
db01celadm01: DB_FC_BY_ALLOCATED     DBCDB4             2,157,755 MB
db01celadm01: DB_FC_BY_ALLOCATED     DBCDB5             9,509,356 MB
db01celadm01: DB_FC_BY_ALLOCATED     _OTHER_DATABASE_     365,790 MB
db01celadm02: DB_FC_BY_ALLOCATED     ASM                    0.000 MB
db01celadm02: DB_FC_BY_ALLOCATED     DBCDB1             1,629,001 MB
db01celadm02: DB_FC_BY_ALLOCATED     DBCDB2             4,761,316 MB
db01celadm02: DB_FC_BY_ALLOCATED     DBCDB3             4,495,902 MB
db01celadm02: DB_FC_BY_ALLOCATED     DBCDB4             2,106,805 MB
db01celadm02: DB_FC_BY_ALLOCATED     DBCDB5             9,848,567 MB
db01celadm02: DB_FC_BY_ALLOCATED     _OTHER_DATABASE_     377,023 MB
db01celadm03: DB_FC_BY_ALLOCATED     ASM                    0.000 MB
db01celadm03: DB_FC_BY_ALLOCATED     DBCDB1             1,664,919 MB
db01celadm03: DB_FC_BY_ALLOCATED     DBCDB2             4,872,123 MB
db01celadm03: DB_FC_BY_ALLOCATED     DBCDB3             4,459,631 MB
db01celadm03: DB_FC_BY_ALLOCATED     DBCDB4             2,096,412 MB
db01celadm03: DB_FC_BY_ALLOCATED     DBCDB5             9,750,586 MB
db01celadm03: DB_FC_BY_ALLOCATED     _OTHER_DATABASE_     315,181 MB
```

In the previous example, `DBCDB1` consumes approximately 1.6 TB of cache space on each cell. To create a minimal IROM plan that increases the `DBCDB1` allocation to 2 TB on each cell you could use the following command:

```
$ dcli -g cell_group cellcli -e alter iormplan dbplan=\(\(name=DBCDB1,
flashCacheMin=2T\)\)
```

If Exadata Smart Flash Cache is fully populated in a consolidated environment, then using the IORM `flashCacheMin` to increase the allocation for one database effectively steals space from all of the others. In such cases, the cache space is not transferred immediately, and it may take more than one scan of the desired data to bring it into the cache.

Using `CELL_FLASH_CACHE KEEP`

In addition to other approaches, you can set the `CELL_FLASH_CACHE` segment storage option to `KEEP` to elevate the segment priority and keep the segment data in Exadata Smart Flash Cache, even when the data is not being used.

For example:

```
SQL> CREATE TABLE t1 (c1 number, c2 varchar2(200)) STORAGE (CELL_FLASH_CACHE
KEEP);


SQL> ALTER TABLE t2 STORAGE (CELL_FLASH_CACHE KEEP);
```

Starting with Oracle Exadata System Software release 24.1.0 in conjunction with Oracle Database 23ai, segments with this setting are automatically populated into Exadata Smart Flash Cache. Previously, the segment data was populated when the segment was read.

Identifying Objects to Populate Manually

You can use the Segments by UnOptimized Reads section of the AWR report to identify objects that are not being cached.

This information is also available by comparing `physical reads` with `optimized physical reads` in `V$SEGMENT_STATISTICS` and `V$SEGSTAT`.

Other Considerations

- Small segments may not qualify for direct path reads and end up populating the database buffer cache when manual population is performed. This can be avoided by using a parallel query to perform the scan.
- You need to perform multiple scans to populate the cache with data from tables and their associated indexes. To populate the cache with index blocks, use an `INDEX FAST FULL SCAN`.

3.3 Administering Exadata Hybrid Columnar Compression

Use these procedures to administer Oracle Database objects that use Exadata Hybrid Columnar Compression.

- Determining If a Table Is Compressed
  Query the `*_TABLES` or `*_TAB_PARTITIONS` data dictionary views to determine whether a table or partitioned is compressed.
- Determining Which Rows are Compressed
  When Exadata Hybrid Columnar Compression tables are updated, the rows change to a lower level of compression.
- Changing Compression Level
  You can change the compression level for a partition, table, or tablespace.
- Importing and Exporting Exadata Hybrid Columnar Compression Tables
  You can use the `impdp` and `expdp` commands to import and export Exadata Hybrid Columnar Compression tables.
- Restoring an Exadata Hybrid Columnar Compression Table
  The compressed table backup can be restored to a system that supports Exadata Hybrid Columnar Compression, or to a system that does not support Exadata Hybrid Columnar Compression.

3.3.1 Determining If a Table Is Compressed

Query the `*_TABLES` or `*_TAB_PARTITIONS` data dictionary views to determine whether a table or partitioned is compressed.

- Query the `*_TABLES` data dictionary views to determine table compression.

  In the `*_TABLES` data dictionary views, compressed tables have `ENABLED` in the `COMPRESSION` column.

  ```
  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
```

For partitioned tables, the `COMPRESSION` column is `NULL` in the `*_TABLES` data dictionary views.

- For partitioned tables, query the `*_TAB_PARTITIONS` data dictionary views.

  The `COMPRESSION` column of the `*_TAB_PARTITIONS` views indicates the table partitions that are compressed. The `COMPRESS_FOR` column indicates the compression method in use for the table or partition.

```
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
```

3.3.2 Determining Which Rows are Compressed

When Exadata Hybrid Columnar Compression tables are updated, the rows change to a lower level of compression.

For example, the compression level might change from `COMP_FOR_QUERY_HIGH` to `COMP_FOR_OLTP` or `COMP_NOCOMPRESS`.

By sampling the table rows, you can determine the percentage of rows that are no longer at the higher compression level.

- Use the following query to determine the compression level of a row:

```
DBMS_COMPRESSION.GET_COMPRESSION_TYPE (
   ownname    IN    VARCHAR2,
   tabname    IN    VARCHAR2,
   row_id     IN    ROWID)
  RETURN NUMBER;
```

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.

Related Topics

- GET_COMPRESSION_TYPE Function

3.3.3 Changing Compression Level

You can change the compression level for a partition, table, or tablespace.

The following example describes a scenario when you might want to change the compression level.

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 archive compression to free up disk space.

- To change the compression level of a partitioned table you can use the `DBMS_REDEFINITION` package.

  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.
- To change the compression level for a single partition of a partitioned table, you can use the `ALTER TABLE ... MODIFY PARTITION` command.
- To change the compression level of a non-partitioned table use the `ALTER TABLE ... MOVE` command with the `COMPRESS FOR` clause.

  To perform DML statements against the table while the `ALTER TABLE ... MOVE` command is running, you must also add the `ONLINE` clause.
- To change the compression level for a tablespace, use the `ALTER TABLESPACE` command.

  This defines the default for new objects created in the tablespace. Existing objects are not changed or moved.
- You can use Automatic Data Optimization (ADO) to create policies that automatically adjust the compression level.

Related Topics

- *Oracle Database Administrator’s Guide*
- *Oracle Database PL/SQL Packages and Types Reference*
- Using Automatic Data Optimization

3.3.4 Importing and Exporting Exadata Hybrid Columnar Compression Tables

You can use the `impdp` and `expdp` commands to import and export 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-64307: Exadata Hybrid Columnar Compression is not supported for
tablespaces on this storage type
```

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.

Related Topics

- Oracle Data Pump Import
- ALTER TABLESPACE

3.3.5 Restoring an Exadata Hybrid Columnar Compression Table

The compressed table backup 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).
- When an Exadata Hybrid Columnar Compression table is restored to a system that does not support Exadata Hybrid Columnar Compression, you must convert the table from Exadata Hybrid Columnar Compression to an uncompressed format.

Use the following steps to convert an Exadata Hybrid Columnar Compression table to an uncompressed format.

1. Ensure that there is sufficient space to store the data in uncompressed format.
2. Use RMAN to restore the Exadata Hybrid Columnar Compression tablespace.
3. Alter the table compression from Exadata Hybrid Columnar Compression to `NOCOMPRESS`:

   For example:

   ```
   SQL> ALTER TABLE table_name MOVE ONLINE NOCOMPRESS;
   ```

   Alternatively, you can use the following command to move the data in parallel:

   ```
   SQL> ALTER TABLE table_name MOVE ONLINE NOCOMPRESS PARALLEL;
   ```

If the table partitioned, then alter the compression method for each partition separately. For example:

```
SQL> ALTER TABLE table_name MOVE PARTITION partition_name NOCOMPRESS
ONLINE;
```

After you convert the table to an uncompressed format, you can optionally use another form of compression, such as OLTP compression or Oracle Database In-Memory compression.

Related Topics

- Recovering Tables and Table Partitions from RMAN Backups
- ALTER TABLE

3.4 Administering Oracle Database Features on Exadata

Use these guidelines to use and administer the following Oracle Database features in conjunction with Oracle Exadata.

- Using Indexes on Exadata
  In the past, databases required indexes for good performance. However, Oracle Exadata can deliver superior scan rates without using indexes.
- Using SQL Tuning Advisor on Exadata
  SQL Tuning Advisor takes one or more SQL statements as input and uses the Automatic Tuning Optimizer to perform SQL tuning on the statements.
- Using SQL Plan Management on Exadata
  SQL plan management prevents performance regressions resulting from sudden changes to the execution plan of a SQL statement by recording and evaluating the execution plans of SQL statements over time.
- Using Exadata System Statistics
  System statistics measure the performance of CPU and storage so that the Oracle Database optimizer can use these inputs when evaluating SQL execution plans.
- Using the Same DB_UNIQUE_NAME for Multiple Database Instances
  You can create database instances that use the same `DB_UNIQUE_NAME` value if the databases are associated with separate Oracle ASM clusters.

3.4.1 Using Indexes on Exadata

In the past, databases required indexes for good performance. However, Oracle Exadata can deliver superior scan rates without using indexes.

Review the application execution plans that use indexes to determine if they would run faster with Exadata Smart Scan Offload. 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 an index invisible, use the following command and substitute the name of the index in place of *index_name*:

```
SQL> ALTER INDEX index_name INVISIBLE;
```

3.4.2 Using SQL Tuning Advisor on Exadata

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

Related Topics

- *Oracle Database 2 Day DBA*
- *Oracle Database SQL Tuning Guide*

3.4.3 Using SQL Plan Management on Exadata

SQL plan management prevents performance regressions resulting from sudden changes to the execution plan of a SQL statement by recording and evaluating the execution plans of SQL statements over time.

With SQL Plan management, you build SQL plan baselines composed of a set of existing plans known to be efficient. The SQL plan baselines are then used to preserve performance of corresponding SQL statements, regardless of changes occurring in the system, such as software upgrades or the deployment of new application modules.

You can also use SQL plan management to gracefully adapt to changes such as new optimizer statistics or indexes. You can verify and accept only plan changes that improve performance. SQL plan evolution is the process by which the optimizer verifies new plans and adds them to an existing SQL plan baseline.

Both SQL profiles and SQL plan baselines help improve the performance of SQL statements by ensuring that the optimizer uses only optimal plans. Typically, you create SQL plan baselines before significant performance problems occur. SQL plan baselines prevent the optimizer from using suboptimal plans in the future. The database creates SQL profiles when you invoke SQL Tuning Advisor, which you do typically only after a SQL statement has shown high-load symptoms. SQL profiles are primarily useful by providing the ongoing resolution of optimizer mistakes that have led to suboptimal plans.

The `DBMS_SPM` package supports the SQL plan management feature by providing an interface for you to perform controlled manipulation of plan history and SQL plan baselines maintained for various SQL statements. The `DBMS_SPM` package provides procedures and functions for plan evolution.

Starting with Oracle Exadata System Software release 19.1, there is a new parameter `AUTO_SPM_EVOLVE_TASK` for `DBMS_SPM.CONFIGURE`, which can be used with only Exadata Database Machine on-premises and Oracle Cloud deployments. The `AUTO_SPM_EVOLVE_TASK` parameter can have one of three values:

- `ON`: The feature is enabled. The SPM evolve advisor creates a task to periodically manage the SQL plan history. The task determines whether there are alternatives and if SQL execution plans should be evolved and accepted. The task runs outside the normal maintenance window in a similar manner to high-frequency statistics gathering.

- `OFF`: The feature is disabled. This is the default value.
- `AUTO`: Oracle Database decides when to use the feature.

Related Topics

- Overview of SQL Plan Management
- DBMS_SPM

3.4.4 Using Exadata System Statistics

System statistics measure the performance of CPU and storage so that the Oracle Database optimizer can use these inputs when evaluating SQL execution plans.

During first instance startup, Oracle Database automatically gathers default system statistics, which are also known as noworkload statistics. However, you can also gather Exadata-specific system statistics. The Exadata system statistics ensure that the SQL optimizer factors in the performance characteristics of Oracle Exadata Database Machine.

Use the following SQL command to see if Exadata-specific statistics are in use.

```
SQL> SELECT pname, PVAL1 FROM aux_stats$ WHERE pname='MBRC';
```

If `PVAL1` is NULL, then the default system statistics are in use.

The following command gathers Exadata system statistics for use by the database optimizer:

```
SQL> exec DBMS_STATS.GATHER_SYSTEM_STATS('EXADATA');
```

If required, you can revert back to the default system statistics by using the `DBMS_STATS.DELETE_SYSTEM_STATS` procedure and then restarting the database.

For new applications or deployments, you should perform pre-production system testing to compare the default system performance with the performance using Exadata system statistics.

For existing applications or deployments, Exadata system statistics should be generated and tested on an equivalent test system before they are first used on a production system. This approach mitigates the risk of an unexpected performance regression on a production system.

If an equivalent test system is unavailable, you can perform more realistic testing by copying the production Exadata system statistics to your test system. The following outlines the procedure:

1. On the production system, create a statistics table.

   For example:

   ```
   SQL> exec DBMS_STATS.CREATE_STAT_TABLE(stattab => 'exadata_stats');
   ```

2. On the production system, gather the Exadata system statistics into the newly created table.

For example:

```
SQL> exec DBMS_STATS.GATHER_SYSTEM_STATS(gathering_mode => 'EXADATA',
stattab => 'exadata_stats');
```

Statistics stored in a user-defined statistics table are not used by the system. So, gathering these statistics does not impact the production system performance.

3. Copy the statistics table from the production system to the test system.

   You can copy the table by various means, including data pump export and import.

4. On the test system, import the statistics table into the Oracle data dictionary.

   For example:

```
SQL> exec DBMS_STATS.IMPORT_SYSTEM_STATS(stattab => 'exadata_stats');
```

Related Topics

- DBMS_STATS Package

3.4.5 Using the Same DB_UNIQUE_NAME for Multiple Database Instances

You can create database instances that use the same `DB_UNIQUE_NAME` value if the databases are associated with separate Oracle ASM clusters.

Starting with Oracle Exadata System Software release 19.1.0, multiple Oracle databases sharing the same Oracle ASM storage can use the same `DB_UNIQUE_NAME`.

WARNING:

If you configure databases to have the same `DB_UNIQUE_NAME`, then those databases cannot be backed up to Oracle Zero Data Loss Recovery Appliance.

To use the same `DB_UNIQUE_NAME` across multiple Oracle databases you must:

- Use separate Oracle ASM clusters for the databases using the same `DB_UNIQUE_NAME` value. The Oracle ASM cluster name is used to qualify the `DB_UNIQUE_NAME` in I/O Resource Management (IORM), Exadata Smart Flash Cache, and Exadata Smart Scan Offload operations.
- Implement ASM-scoped security for each Oracle ASM cluster that contains the databases using the same `DB_UNIQUE_NAME` value.

Related Topics

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

4 Maintaining Oracle Exadata System Software

This section explains how to maintain Oracle Exadata System Software.

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.

- Understanding Oracle Exadata System Software Release Numbering
  The Oracle Exadata System Software release number is related to the Oracle Database release number.
- Understanding Automated Cell Maintenance
  The Management Server (MS) includes a file deletion policy based on the date.
- Recommendations for Changing the Exadata Storage Server Network Address
  Review the following advice before changing the fundamental configuration of a storage server, such as changing the IP address, host name, or RDMA Network Fabric address.
- Using the ipconf Utility
  The `ipconf` utility is used to set and change the following parameters on Oracle Exadata servers.
- Oracle Exadata System Software Validation Tests and Utilities
  You can use a variety of commands and utilities to validate the Oracle Exadata System Software and hardware configurations.
- Locating Serial Numbers for System Components
  You may need to provide the serial numbers for the system components when contacting Oracle Support Services.
- Diagnostic and Repair Utilities
  Oracle Exadata System Software includes utilities for diagnostics and repair of Oracle Exadata Storage Server.
- System Diagnostics Data Gathering with sosreports and Oracle ExaWatcher
  You can use the sosreport utility and Oracle ExaWatcher to diagnose problems with your system.
- Host Console Support
  The storage servers and database servers of Oracle Exadata are configured to provide host console access.
- Oracle Linux Kernel Crash Core Files
  The storage servers and database servers of Oracle Exadata are configured to generate Oracle Linux kernel crash core files in the `/var/crash` directory, when the Oracle Linux operating system malfunctions or crashes.
- Monitoring syslog Messages Remotely
  By default, storage server syslog messages are written to local log files.

4.1 Understanding Oracle Exadata System Software Release Numbering

The Oracle Exadata System Software release number is related to the Oracle Database release number.

The Oracle Exadata System Software release number matches the highest Oracle Grid Infrastructure and Oracle Database version it supports. For example, the highest version Oracle Exadata System Software release 18 supports is Oracle Grid Infrastructure and Oracle Database release 18. The highest version Oracle Exadata System Software release 12.2 supports is Oracle Grid Infrastructure and Oracle Database release 12.2.0.1.

Release 18c and Later Numbering

The Oracle Exadata System Software release that followed release 12.2.1.1.8 was renamed to 18.1.0 and a new numbering scheme for the Oracle Exadata System Software was implemented. Instead of a legacy nomenclature such as 12.2.1.1.5, a three field format consisting of: *Year*.*Update*.*Revision* is used, for example 18.1.0. This new numbering scheme allows you to clearly determine:

- The annual release designation of the software
- The latest software update, which can contain new features
- The latest software revision, which includes security and software fixes

If there are new features or new hardware supported, a new software update will be release during the year, for example, 19.2. To allow you to keep current on just security-related and other software fixes after your feature environment becomes stable, software revisions are made available approximately once a month, for example 19.1.3.

Numbering for Releases Prior to 18c

- The first two digits of the Oracle Exadata System Software release number represent the major Oracle Database release number, such as Oracle Database 12c Release 1 (12.1). Oracle Exadata System Software release 12.1 is compatible with all Oracle Database 12c Release 1 (12.1) releases.
- The third digit usually represents the component-specific Oracle Database release number. This digit usually matches the fourth digit of the complete release number, such as 12.1.0.1.0 for the current release of Oracle Database.
- The last two digits represent the Oracle Exadata System Software release.

Related Topics

- Exadata Database Machine and Exadata Storage Server Supported Versions (Doc ID 888828.1)
- Exadata Software and Hardware Support Lifecycle (Doc ID 1570460.1)

4.2 Understanding Automated Cell Maintenance

The Management Server (MS) includes a file deletion policy based on the date.

When there is a shortage of space in the Automatic Diagnostic Repository (ADR) directory, then MS deletes the following files:

- All files in the 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.

The retention period of seven days is the default. The retention period can be modified using the `metricHistoryDays` and `diagHistoryDays` attributes with the `ALTER CELL` command. The `diagHistoryDays` attribute controls the ADR files, and the `metricHistoryDays` attribute controls the other files.

If there is sufficient disk space, then trace files are not purged. This can result in files persisting in the ADR base directory past the time limit specified by `diagHistoryDays`.

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.

MS includes a file deletion policy that is triggered when file system utilization is high. Deletion of files in the `/` (root) directory and the `/var/log/oracle` directory is triggered when file utilization is 80 percent. Deletion of files in the `/opt/oracle` file system is triggered when file utilization reaches 90 percent, and the alert is cleared when utilization is below 85 percent. An alert is sent before the deletion begins. The alert includes the name of the directory, and space usage for the subdirectories. In particular, the deletion policy is as follows:

- The `/var/log/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
  - 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 that are more than one day old can be deleted. If the space pressure is not heavy, then the retention time for crash files is the same as for other files. If there are empty directories under `/var/log/oracle/crashfiles`, these directories are also deleted.
- For the `/opt/oracle` file system, the deletion policy is similar to the preceding settings. However, the file threshold is 90 percent, and files are deleted until the file system utilization is less than 85 percent.
- When file system utilization is full, the files controlled by the `diagHistoryDays` and `metricHistoryDays` attributes are purged using the same purging policy.
- 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.

Every hour, MS deletes eligible alerts from the alert history using the following criteria. Alerts are considered eligible if they are stateless or they are stateful alerts which have been resolved.

- If there are less than 500 alerts, then alerts older than 100 days are deleted.
- If there are between 500 and 999 alerts, then the alerts older than 7 days are deleted.
- If there are 1,000 or more alerts, then all eligible alerts are deleted every minute.

Note:

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

Related Topics

- ALTER CELL
- DESCRIBE CELL

4.3 Recommendations for Changing the Exadata Storage Server Network Address

Review the following advice before changing the fundamental configuration of a storage server, such as changing the IP address, host name, or RDMA Network Fabric address.

- Before changing the storage server configuration, ensure that all Oracle Automatic Storage Management (Oracle ASM), Oracle Real Application Clusters (Oracle RAC) and database instances that use the storage servers do not access the storage server while you are changing the IP address.
- After changing the storage server configuration, ensure that consumers of storage server services are correctly reconfigured to use the new connect information of the storage server. If Oracle Auto Service Request (ASR) is being used, then deactivate the asset from Oracle ASR Manager, and activate the asset with the new IP address.
- When changing a storage server configuration, change only one storage server at a time to ensure that Oracle ASM and Oracle RAC work properly during the changes.

Related Topics

- About Oracle Auto Service Request

4.4 Using the ipconf Utility

The `ipconf` utility is used to set and change the following parameters on Oracle Exadata 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
- RDMA Network Fabric addresses

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

Table 4-1 ipconf Options

Option	Description
no option	Utility starts in main editing mode.
`-check-consistency [-pkey-file pkey.conf]`	Determine if pkey is configured on current host by looking into `cell.conf`. If pkey is configured, this command reversely checks the sub interfaces and the `ifcfg` config files, compares them with the `pkey.conf`, and reports any inconsistencies it finds.
`-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.
`-pkey-add $PKEY.CONF -pkey-apply [-force]`	Apply the pkey configuration to an unconfigured system or add a new pkey configuration to the current pkey configured host (without changing the current pkey settings). If the interface specified in the pkey file is already, the command exits with an error. If you include the `-force` option, then the current pkey configuration for the already configured interface is deleted, and the configuration based on the new `pkey.conf` file is applied.
`-pkey-delete $PKEY.CONF -pkey-apply`	To delete some pkeys on the current pkey configured host, where `$PKEY.CONF` is a simple pkey file that includes only the pkey and physical devices in the entries. If the system is not already configured with pkeys, the command exits with an error.
`-pkey-getruntime`	Get the InfiniBand partitioning pkey configuration for the current system.
`-pkey-matchruntime [-pkey-file pkey.conf]`	Get the run-time pkey configuration, and match it with the file specified with the `-pkey-file` option. If a file is not specified, then `/opt/oracle.cellos/pkey.conf` is used as the default.
`-preconf preconf.scv [-pkey-file pkey.conf] {-generate | -generateall | -verify}`	Verifies the `pkey.conf` file and `preconf.scv`, then generates `cell.conf`, and finally cross-validates `pkey.conf` and `cell.conf`.
`-semantic`	Checks only the DNS and NTP configuration.
`-semantic-min`	Checks for access to at least one NTP and one DNS server.

Table 4-1 (Cont.) ipconf Options

Option	Description
`-update [-dns` *`dns_ip_list`*`] [-ntp` *`ntp_ip_list`*`] [-ilom-dns` *`ilom_dns_list`*`] [-ilom-ntp` *`ilom_ntp_list`*`] [-force | -dry]`	Update DNS and NTP servers for Linux and ILOM. For the DNS and NTP servers, specify a list of IP addresses separated by commas. A maximum of three DNS servers are allowed with `-ilom-dns`. A maximum of two NTP servers are allowed with `-ilom-ntp`. If the timestamp obtained from the new NTP server differs from the current time known to the system by more than 1 second (*time step*), then the command errors out and does not update the NTP settings. The `-force` flag on command line overrides this check. The `-dry` option indicates the command should check all settings, but not apply them.
`-verbose`	The `-verbose` option shows all details. If `-verbose` is not used, then only errors are displayed.
`-verify`	Verifies the consistency between the stored cell configuration and the running configuration. Success returns zero errors. You can use the following options with `-verify`: `-pkey-file`, `-semantic`, `-semantic-min`

The following example shows the display for the `ipconf` utility when setting the Sun ILOM interface.

Example 4-1 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): 192.0.2.10 192.0.2.12 192.0.2.13
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): 192.0.2.06 192.0.2.12 1192.0.2.13
Do you want to change it (y/n) [n]:

Network interfaces
Name  State   IP address      Netmask       Gateway        Hostname
eth0  Linked  192.0.2.151  255.255.252.0 192.0.2.15  Managment myg.example.com
eth1  Unlinked
eth2  Unlinked
eth3  Unlinked
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 hostname from the list below
1: myg.example.com
2: myg-private.example.com
Canonical fully qualified domain name [1]:

Select default gateway interface from the list below
1: eth01
Default gateway interface [1]:

Canonical hostname: myg.example.com
Nameservers: 192.0.2.10 192.0.2.12 192.0.2.13
Timezone: America/Los_Angeles
NTP servers: 192.0.2.06 192.0.2.12 192.0.2.13
Network interfaces
Name  State  IP address      Netmask        Gateway        Hostname
eth0  Linked 192.0.2.151 255.255.252.0  192.0.2.15  myg.example.com
eth1  Unlinked
eth2  Unlinked
eth3  Unlinked
bond0 ib0,ib1  192.168.13.101 255.255.252.0 Private  myg-priv.example.com
Is this correct (y/n) [y]:

Do you want to configure basic ILOM settings (y/n) [y]: y
Loading configuration settings from ILOM ...
ILOM Fully qualified hostname [myg_ilom.example.com]:
ILOM IP discovery (static/dhcp) [static]:
ILOM IP address [192.0.2.201]:
ILOM Netmask [255.255.252.0]:
ILOM Gateway or none [192.0.2.15]:
ILOM Nameserver or none: [192.0.2.10]:
ILOM Use NTP Servers (enabled/disabled) [enabled]:
ILOM First NTP server. Fully qualified hostname or ip address or none
[192.0.2.06]:
ILOM Second NTP server. Fully qualified hostname or ip address or none [none]:
```

```
Basic ILOM configuration settings:
Hostname             : myg.example.com
IP Discovery         : static
IP Address           : 192.0.2.10
Netmask              : 255.255.252.0
Gateway              : 192.0.2.15
DNS servers          : 192.0.2.10
Use NTP servers      : enabled
First NTP server     : 192.0.2.06
Second NTP server    : none
Timezone (read-only) : America/Los Angeles

Is this correct (y/n) [y]:
```

Related Topics

- Set the NTP Server on Exadata Storage Servers
- Set the NTP Server Address on the Database Servers
- Change the DNS Server Address on the Database Server
- Change the DNS Server on Exadata Storage Server

4.5 Oracle Exadata System Software Validation Tests and Utilities

You can use a variety of commands and utilities to validate the Oracle Exadata System Software and hardware configurations.

- Summary of Software and Firmware Components on Oracle Exadata Storage Servers
  The `imageinfo` command located in the `/usr/local/bin/` directory provides a summary of release and status of the software and firmware components on Oracle Exadata Storage Servers.
- Oracle Exadata Storage Server Image History
  The `imagehistory` command lists the version history for Oracle Exadata Storage Server.
- Validation of the State and Health of the System
  The validation framework runs different tests under certain conditions, such as on first boot after recovery of an Oracle Exadata Storage Server using the rescue and recovery functionality of the CELLBOOT USB flash drive, or when patching an Oracle Exadata Storage Server.

4.5.1 Summary of Software and Firmware Components on Oracle Exadata Storage Servers

The `imageinfo` command located in the `/usr/local/bin/` directory provides a summary of release and status of the software and firmware components on Oracle Exadata Storage Servers.

The software and firmware components comprise the storage server image. The release and status information is required when working with My Oracle Support.

The following table lists the output fields from the `imageinfo` command.

Table 4-2 Description of `imageinfo` Command Output

Field	Description
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*. 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.
Active image version	Main release version of the overall cell image indicating a specific combination of releases of operating system, core Oracle Exadata System 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 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*.
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.
Cell boot usb partition	Oracle Exadata Storage Server 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.
Cell rpm version	Cell software version or cell rpm version as reported by the CellCLI utility.
Cell version	Release version as reported by the CellCLI utility.
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.
Inactive image activated	Time stamp for activation of the inactive image. This field is similar to active image activated field.

Table 4-2 (Cont.) Description of `imageinfo` Command Output

Field	Description
Inactive image status	Status of the inactive image. This field is similar to the status of the active image.
Inactive image version	Version of the cell before the most-recent patch was applied.
Inactive software partition on device	Oracle Exadata System Software file system partition, `/opt/oracle`, for the inactive image.
Inactive system partition on device	The root (`/`) file system partition for the inactive image.
Kernel version	Operating system kernel version of the cell.
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
```

Related Topics

- Validation of the State and Health of the System
  The validation framework runs different tests under certain conditions, such as on first boot after recovery of an Oracle Exadata Storage Server using the rescue and recovery functionality of the CELLBOOT USB flash drive, or when patching an Oracle Exadata Storage Server.

4.5.2 Oracle Exadata Storage Server Image History

The `imagehistory` command lists the version history for Oracle Exadata Storage Server.

For example, if a storage server was updated from release 11.2.1.2.6 to release 11.2.1.3.1, and then updated to release 11.2.1.2.3, the `imagehistory` command displays this history. The following is an example of the output:

```
# imagehistory
Version                    : 11.2.1.2.3
Image activation date      : 2012-12-03 06:06:46 -0700
Imaging mode               : fresh
Imaging status             : success

Version                    : 11.2.3.2.0.120713
Image activation date      : 2012-12-12 17:56:31 -0700
Imaging mode               : out of partition upgrade
Imaging status             : success
```

4.5.3 Validation of the State and Health of the System

The validation framework runs different tests under certain conditions, such as on first boot after recovery of an Oracle Exadata Storage Server using the rescue and recovery functionality of the CELLBOOT USB flash drive, or when patching an Oracle Exadata Storage Server.

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.

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 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.

4.6 Locating 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. Use the `CheckHWnFWProfile` command to view the serial numbers.

```
# /opt/oracle.SupportTools/CheckHWnFWProfile -action list -mode
serial_numbers
```

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 historic record of the serial numbers. The file also contains configuration information for some components.

4.7 Diagnostic and Repair Utilities

Oracle Exadata System Software includes utilities for diagnostics and repair of Oracle Exadata Storage Server.

The utilities help diagnose and repair problems that may occur during the normal life cycle of Oracle Exadata Storage Servers. 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
  The CheckHWnFWProfile utility checks that the system meets the required hardware and firmware specifications, and reports any mismatches.
- The Diagnostic ISO File
  Use the diagnostic ISO file (`diagnostics.iso`) to diagnose and recover from serious problems.
- The ibdiagtools Utilities
  The ibdiagtools utilities are a series of utility programs that perform monitoring and diagnostic functions on the RDMA Network Fabric.
- The make_cellboot_usb Utility
  The `make_cellboot_usb` utility allows you to rebuild a damaged CELLBOOT USB flash drive.

4.7.1 The CheckHWnFWProfile Utility

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

Table 4-3 CheckHWnFWProfile Utility Commands

Command	Description
`./CheckHWnFWProfile`	When run without options, the utility checks the existing hardware and firmware components against the expected values.
`./CheckHWnFWProfile -action` *`list`*	View the existing hardware and firmware versions on the system.

Table 4-3 (Cont.) CheckHWnFWProfile Utility Commands

Command	Description
`./CheckHWnFWProfile -action alter_config -property HWFW_Checker_Updater_Status -value Disabled`	Disable the `CheckHWnFWProfile` utility.
`./CheckHWnFWProfile -action alter_config -property HWFW_Checker_Updater_Status -value Enabled`	Enable the `CheckHWnFWProfile` utility.
`./CheckHWnFWProfile -action check -component` *`list_of_components`*	Check specified components against the expected values.
`./CheckHWnFWProfile -action list -component` *`list_of_components`*	View the hardware and firmware versions of specified components on the system.
`./CheckHWnFWProfile -action list -mode serial_numbers`	List serial numbers. The list includes the following serial numbers: • System • Disk controller • Each disk • RDMA Network Fabric host channel adapter (HCA) Depending on the system, serial numbers for all the memory (RAM) modules may be included.
`./CheckHWnFWProfile -action list -mode supported_info`	View the expected hardware and firmware.
`./CheckHWnFWProfile -h` `./CheckHWnFWProfile --help`	View help and utility usage.

4.7.2 The Diagnostic ISO File

Use the diagnostic ISO file (`diagnostics.iso`) to diagnose and recover from serious problems.

You may need to boot a server using the diagnostic ISO file to diagnose and recover from serious problems. For example, when the system is inaccessible due to system damage or damage to the CELLBOOT USB flash drive.

Use this facility only as directed to perform specific documented maintenance tasks, or under the guidance of Oracle Support Services.

- Booting a Server using the Diagnostic ISO File
  Use this procedure to boot an Oracle Exadata Storage Server or Oracle Exadata Database Server using the diagnostic ISO file (`diagnostics.iso`).

4.7.2.1 Booting a Server using the Diagnostic ISO File

Use this procedure to boot an Oracle Exadata Storage Server or Oracle Exadata Database Server using the diagnostic ISO file (`diagnostics.iso`).

Note:

For information on booting an Oracle Linux KVM guest using the diagnostic ISO file, see Starting a Guest using the Diagnostic ISO File.

1. Download the diagnostic ISO file (`diagnostics.iso`) corresponding to your current Oracle Exadata System Software release.

   If required, use the `imageinfo` command on a running server to determine your current Oracle Exadata System Software release.

   Download the diagnostic ISO file to a client workstation or server that you will use to access the ILOM interface on the server that you want to boot in diagnostics mode. The client machine must have a Web browser and network access to the ILOM interface on the target server.

   If you are using Oracle Exadata System Software Release 18.x, or earlier, you can find the diagnostic ISO file at `/opt/oracle.SupportTools/diagnostics.iso` on your Oracle Exadata Storage Servers or Oracle Exadata Database Servers. Otherwise, search the My Oracle Support (MOS) patch repository using "`exadata diagnostic iso`" as the search term.

   You can also locate the diagnostic ISO file in the Supplemental README that is associated with your Oracle Exadata System Software release. The Supplemental README for each Oracle Exadata System Software release is documented in My Oracle Support document 888828.1.

2. Start the ILOM Web client.

   Start a Web browser on your client machine and navigate to `http://`*`ILOM_SP_IPaddress`*. In the URL, *ILOM_SP_IPaddress* is the IP address of the ILOM service processor (SP) on the target server.

   If you do not know the IP address of the ILOM SP, you can make a serial connection to the ILOM SP and list the network properties by using the `show /network` and `show /networkipv6` commands. For more details, see Connect to Oracle ILOM.

   The Oracle ILOM Web client login dialog appears.

3. Log in to the Oracle ILOM Web client using the `root` account and the password.

4. Use the ILOM Web interface to attach the `diagnostics.iso` file as a virtual CDROM device.

   a. In the ILOM Web interface navigation hierarchy, click **Remote Control**, and then click **Redirection**

      Depending on the version of the ILOM Web interface that you are using, **Redirection** may appear in the left hand navigation pane or in a strip of options running along the top of the page.

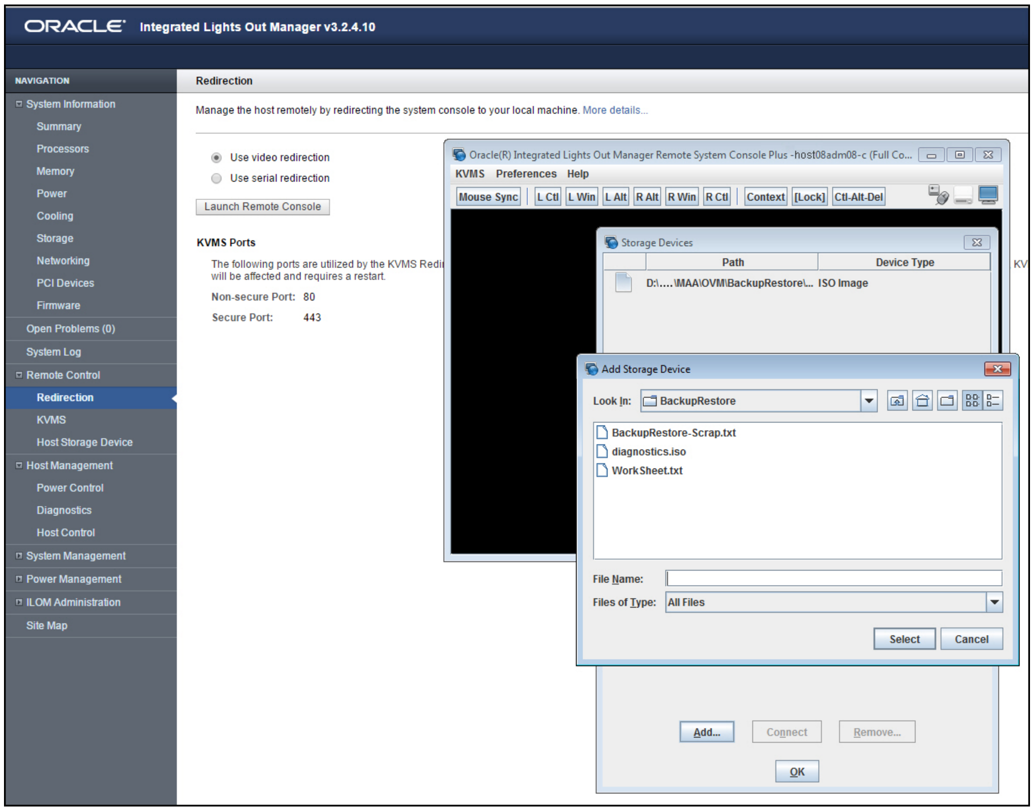
   b. In the Redirection pane, click **Use Video Redirection**, then **Launch Remote Console**.

      In some versions of the ILOM Web interface, the **Use Video Redirection** button does not exist. In that case, go directly to **Launch Remote Console**.

   c. In the Remote Console window, choose the **Storage** menu option in the **KVMS** menu.

   d. In the Storage Devices dialog, click **Add**. And in the resulting dialog, select the `diagnostics.iso` file.

      The following screen image shows an example of what you may see.



Manage the host remotely by redirecting the system console to your local machine. More details...

The following ports are utilized by the KVMS Redi... will be affected and requires a restart.

Non-secure Port: 80

Secure Port: 443

D:\....\MAA\OVM\BackupRestore\... ISO Image

BackupRestore-Scrap.txt
diagnostics.iso
WorkSheet.txt

e. Back in the Storage Devices dialog, select the entry associated with the `diagnostics.iso` file and click **Connect**.

   After the connection is established, the label on the **Connect** button in the Storage Devices dialog changes to **Disconnect**.

   Keep in mind the navigation path to the Storage Devices dialog so that you can easily disconnect the `diagnostics.iso` file after you are done with it.

5. Configure the server to boot from the `diagnostics.iso` file by using the newly configured virtual CDROM device.

   a. In the ILOM Web interface navigation hierarchy, click **Host Management**, and then click **Host Control**

      Depending on the version of the ILOM Web interface that you are using, **Host Control** may appear in the left hand navigation pane or in a strip of options running along the top of the page.

   b. From the resulting list of values, select **CDROM**.

   c. Click **Save**.

   Now, when the system is booted, the `diagnostics.iso` file is used as the default boot image.

6. Back in the Remote Console window, restart the system by using the following command:

```
->  Start /SP/console -script
```

The system now reboots using the `diagnostics.iso` file as the default boot image.

Related Topics

- Exadata Database Machine and Exadata Storage Server Supported Versions (My Oracle Support Doc ID 888828.1)

4.7.3 The ibdiagtools Utilities

The ibdiagtools utilities are a series of utility programs that perform monitoring and diagnostic functions on the RDMA Network Fabric.

The following ibdiagtools utilities reside under `/opt/oracle.SupportTools/ibdiagtools/`:

- `verify-topology` checks the correctness and health of InfiniBand Network Fabric connections. For example, it can determine if both cables from the server go to the same switch in the Oracle Exadata Rack. When both cables go to the same switch, the system loses the ability to fail over to another switch if the first switch fails.
- `checkbadlinks.pl` reports InfiniBand Network Fabric links that are operating at suboptimal speed. This is often an indication that a cable is loose, and needs to be reseated.
- `verify_roce_cables.py` checks that hosts and switches are correctly connected in the RoCE Network Fabric. The utility performs essentially the same function on the RoCE Network Fabric as `verify-topology` does for the InfiniBand Network Fabric.
- `infinicheck` checks and reports the base RDMA Network Fabric performance between servers in Oracle Exadata, such as expected minimum throughput between the database server and storage server, between storage servers, and between a database server and another database server. This utility can help to identify issues in the RDMA Network Fabric. Because the utility runs stress tests on the RDMA Network Fabric, Oracle recommends using the utility when the system is idle and with all cell services shut down. This utility works on both the InfiniBand Network Fabric and RoCE Network Fabric.
- `configure_roce_hostinfo.sh` scans the RoCE Network Fabric and labels each RoCE switch port, identifying the hostname, IP address, and port number of the link target. This information makes it easy to understand what each RoCE switch port is connected to and can be used to construct a map of the RoCE Network Fabric.
- `rocelinkinfo` scans the RoCE Network Fabric and presents real-time status information for each link. The output is presented using the same format as the output of the Linux `iblinkinfo` command. This utility does not depend on `configure_roce_hostinfo.sh`. However, the output from `rocelinkinfo` is more informative after all of the RoCE switch ports have been labeled by running `configure_roce_hostinfo.sh`.
- `setup-ssh` configures SSH user equivalence across the RDMA Network Fabric. Many of the other utilities require SSH user equivalence to operate across the servers and switches connected to the RDMA Network Fabric.

See Also:

- For command syntax and usage information run the command with the `--help` option and no other arguments or options. Additional usage information for some of the ibdiagtools utilities is included in `/opt/oracle.SupportTools/ibdiagtools/README`.
- Sample output from some of the ibdiagtools utilities is provided in `/opt/oracle.SupportTools/ibdiagtools/SampleOutputs.txt`.

4.7.4 The make_cellboot_usb Utility

The `make_cellboot_usb` utility allows you to rebuild a damaged CELLBOOT USB flash drive.

Do not connect more than one USB flash drive to the system when running this utility. The utility builds on the first discovered USB flash drive on the system.

Note:

This utility can only be used on Oracle Exadata Storage Server.

- To see what is done before rebuilding the USB flash drive:

  ```
  cd /opt/oracle.SupportTools
  ./make_cellboot_usb -verbose
  ```

- To rebuild the USB flash drive, run the command with one of the following options: `-execute`, `-force`, or `-rebuild`.

  ```
  ./make_cellboot_usb -execute
  ```

  Or:

  ```
  ./make_cellboot_usb -force
  ```

  Or:

  ```
  ./make_cellboot_usb -rebuild
  ```

4.8 System Diagnostics Data Gathering with sosreports and Oracle ExaWatcher

You can use the sosreport utility and Oracle ExaWatcher to diagnose problems with your system.

Every time a server is started, system-wide configuration information is collected by the sosreport utility, and stored in the `/var/log/cellos/sosreports` directory. You can generate a new sosreport by running the following command as the `root` user. The script starts collecting the information 30 minutes after entering the command.

```
/opt/oracle.cellos/vldrun -script sosreport
```

In addition, the `/opt/oracle.ExaWatcher` directory contains the Oracle ExaWatcher system data gathering and reporting utilities. Gathered data is stored in archive subdirectories. The following table describes the data gathered at different intervals by the utility:

Table 4-4 Oracle ExaWatcher Collector Names and Descriptions

Collector Name	Description
CellSrvStat	Cell server status.
Diskinfo	I/O statistics of the disk, such as successfully completed reads, merged reads, time spent reading, and so on.
FlashSpace	RAW value of the flash card space. Minimum interval limit is 300 seconds.
IBCardInfo (Currently not available for X8M systems)	RDMA Network Fabric card information, and status of InfiniBand Network Fabric ports. Minimum interval is 300 seconds.
IBprocs	Commands that check the RDMA Network Fabric card status. Minimum interval is 600 seconds.
Iostat	CPU statistics, and I/O statistics for devices and partitions.
Lsof	Files opened by current processes. Minimum interval limit is 120 seconds.
MegaRaidFW	MegaRaid firmware information, such as battery information. Minimum interval is 86400 seconds.
Meminfo	Memory management by the kernel.
Mpstat	Microprocessor statistics.
Netstat	Current network connection statistics.
Ps	Active processes statistics.
RDSinfo	Availability of cell servers. Interval limit is 30 seconds.
Slabinfo	Caches for frequently-used objects in the kernel.
Top	Dynamic, real-time view of the system.
Vmstat	Virtual memory status.

To use Oracle ExaWatcher, do the following:

1. As the `root` user, start the Oracle ExaWatcher processes and service.

   ```
   # systemctl start ExaWatcher
   ```

2. Run the Oracle ExaWatcher utility at the `root` user.

   ```
   /opt/oracle.ExaWatcher/ExaWatcher.sh [options]
   ```

The following options are available for use with the Oracle ExaWatcher utility:

Option	Description
No options specified	The utility runs using the default options.

Option	Description

`-c`|`--command '`*`collector_name ;;`*` "`*`default_command; ...`*` " '`	To change the core command to be run on the current group. Only the following core commands can be changed:
	`CellSrvStat`
	`Iostat`
	`Mpstat`
	`Netstat`
	`Ps`
	`Top`
	`Vmstat`
	Example: `--command 'Vmstat;; "vmstat -a"'`

`--createconf "`*`config_file_to_create`*`"`| null	The utility parses all command line inputs, validates them, and creates a configuration file. If the file path and name is not specified, then the utility overwrites the default configuration file.
`-d`|`--disable "`*`collector_name`*`"`	The name of the collector to be disabled on the utility. Example: `--disable "Vmstat"`
`-e` |`--end "`*`end_time`*`"`	The ending tine for the current group. The default value is 10 years from current time. Example: `--end "11/06/2013 12:01:00"`
`--fromconf "`*`configuration_file`*`"`|`null`	The configuration file to use with the Oracle ExaWatcher utility. The default configuration files are as follows: `/opt/oracle.ExaWatcher/ExaWatcher.conf` for Oracle Linux
`-g`|`--group`	Starts a new group for gathering data. Other options can be specified with the `group` option.
`-h`|`--help`	Displays help information.
`-i`|`--interval "`*`interval_length`*`"`	The sampling interval for the current group, in seconds. The default value is 5 seconds. Certain collection modules cannot be run every second because the modules consume resources. Example: `--interval 10`
`-l`|`--spacelimit`	Sets the limit for the amount of storage space used by the utility. The limit is specified in MB. On database servers, the default is 6 GB. On storage servers, the default is 600 MB. Example: `--spacelimit 900`
`--lastconf`	The most-recent configuration file used with the utility. Data is not collected when using this option.

Option	Description
`--listcmd "Full"|"Nameonly"|"Core"|"CMD"|"Enabled"|null`	The information about the command inputs. The following are the options: `Full` displays all the information about the commands and samplers. `Nameonly` displays all names and if it is enabled. `Core` displays only the core sampler information. `CMD` displays the name, if it is enabled, and the default commands.
`-m | --commandmode {"ALL" | "CORE" | "SELECTED"}`	The type of collection modules to run for the current group. The following are the options: `ALL` runs all collection modules. `CORE` runs only the core collection modules. `SELECTED` runs only the specified collection modules. The default value is `ALL`. Example: `--commandmode "CORE"`
`-o | --count` *"archiving_count"*	The archive count of the current group. The default value is 720. Example: `--count 500`
`-r |--resultdir "`*result_directory*`"`	The directory path to store the results of the data collection. Example: `--r "/opt/oracle.ExaWatcher/archive"`
`--stop`	To stop the utility and all its processes, and then to zip the data files.
`-t |--start` *"start_time"*	The starting time for the current group. The default is 20 seconds from the current time. Example: `--start "11/05/2013 12:00:00"`
`-u |--customcmd '`*sample_name ;; "custom_command;...* `" '`	To include a custom collection module in the current group. Example: `--customcmd 'Lsl; "/bin/ls -l"'`
`-z |--zip` *"bzip2" "gzip"*	The compression program to use on the collected data. The default program is bzip2. Example: `--zip "gzip"`

4.9 Host Console Support

The storage servers and database servers of Oracle Exadata are configured to provide host console access.

The host console is useful when collecting Oracle Linux kernel traces or creating crash dump files to help diagnose severe malfunctions.

To access the host console, perform the following procedure:

1. Connect to the Integrated Lights Out Manager (ILOM) using SSH and log in as an ILOM administrator.
2. Run the `start /SP/console` command

   To stop using the console, use the `stop /SP/console` command.

4.10 Oracle Linux Kernel Crash Core Files

The storage servers and database servers of Oracle Exadata are configured to generate Oracle Linux kernel crash core files in the `/var/crash` directory, when the Oracle Linux operating system malfunctions or crashes.

The `crash` utility can be used to analyze the crash files. The crash files are automatically removed by the ExaWatcher 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.

4.11 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 Oracle Exadata database servers and storage servers.

- 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.
- Use the `ALTER CELL` or `ALTER DBSERVER` command configure each server to forward specified syslog messages to the loghost server by setting the `syslogconf` attribute.

  The server configuration is maintained across restarts and updates.
- Use the `ALTER CELL VALIDATE SYSLOGCONF` or `ALTER DBSERVER VALIDATE SYSLOGCONF` command to test message transmission from the Exadata servers to the loghost server.

Starting with Oracle Exadata System Software release 21.2.0, you can also configure each server to forward syslog messages from the Integrated Lights Out Manager (ILOM) service processor (SP). To configure this facility, use the `ALTER CELL` or `ALTER DBSERVER` command to set the `ilomSyslogClients` attribute.

Related Topics

- ALTER CELL
- ALTER DBSERVER

5 Managing I/O Resources

Exadata I/O Resource Management (IORM) is a tool for managing how multiple workloads and databases share the I/O resources of Oracle Exadata Database Machine.

To manage workloads within a database, Oracle Database Resource Manager is enhanced to work with Exadata IORM.

- Understanding I/O Resource Management (IORM)
  IORM manages the storage server 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.
- Administering IORM
  You can perform various administrative tasks related to I/O Resource Management (IORM).

Related Topics

- Managing Resources with Oracle Database Resource Manager
- Master Note for Oracle Database Resource Manager (My Oracle Support Doc ID 1339769.1)
- Tool for Gathering I/O Resource Manager Metrics: metric_iorm.pl (My Oracle Support Doc ID 1337265.1)

5.1 Understanding I/O Resource Management (IORM)

IORM manages the storage server 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.

- About I/O Resource Management (IORM) in Exadata Database Machine
  IORM provides many features for managing resource allocations. Each feature can be used independently or in conjunction with other features.
- About Database Resource Management Within a Database
  Oracle Database Resource Manager enables you to manage workloads within a database.
- About Interdatabase Resource Management
  Interdatabase resource management enables you to manage resources for multiple databases using the same shared storage.
- About Cluster Resource Management
  Cluster resource management enables you to manage resources for multiple clusters using the same shared storage.
- About Category Resource Management
  Categories represent collections of consumer groups across all databases.
- About Consumer Groups and Resource Plans
  Oracle Exadata provides out-of-the-box consumer groups and resource plans specifically designed for data warehouses that use Oracle Exadata System Software.

- About CDB Plans and Pluggable Databases
  The container database (CDB) can have multiple workloads within multiple PDBs competing for resources.

5.1.1 About I/O Resource Management (IORM) in Exadata Database Machine

IORM provides many features for managing resource allocations. Each feature can be used independently or in conjunction with other features.

Storage is often shared by multiple types of workloads and databases, which can often lead to performance problems. For example, large parallel queries on a production data warehouse can impact the performance of critical OLTP in another database. You can mitigate these problems by over-provisioning the storage system, but this diminishes the cost-saving benefits of shared storage. You can also schedule non-critical tasks at off-peak hours, but this manual process is laborious.

IORM allows workloads and databases to share Oracle Exadata Storage Servers according to user-defined policies. To manage workloads within a database, you can define database resource plans, using Oracle Database Resource Manager which is enhanced to manage Oracle Exadata Storage Server I/O resources. You can also define a container database (CDB) resource plan that allows management for the pluggable databases (PDBs) that it contains. To manage multiple databases, you can define an interdatabase plan. Or, you can define a cluster plan to perform cluster-based resource management.

When there is contention for I/O resources, IORM schedules I/O by immediately issuing some I/O requests and queuing others. I/O requests are serviced immediately for workloads that are not exceeding their resource allocation, according to the resource plans. I/O requests are queued for workloads exceeding their resource allocation. Queued I/Os are serviced according to the priorities in the resource plans when the workload no longer exceeds its resource allocation. When the cell is operating below capacity and there is no contention for I/O resources, IORM does not queue I/O requests, and lets a workload exceed its resource allocation to consume available I/O resources.

For example, if a production database and test database share Oracle Exadata Storage Server resources, 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.

Flash IORM protects the latency of critical OLTP I/O requests in flash cache. When table scans are running on flash concurrently with OLTP I/O requests, the OLTP latency is impacted significantly. Flash IORM queues and throttles the table scan, and other low priority I/O requests. The critical OLTP I/O requests are never queued. When the flash disks are not busy serving critical OLTP I/O requests, the queued I/O requests are issued based on the resource allocations in the IORM plan.

- About the IORM Objective
  The I/O Resource Management (IORM) `objective` option specifies the optimization mode for IORM.
- IORM Plans
  Exadata I/O Resource Management (IORM) manages resources using various IORM plans.
- Resource Assignment Methods
  You can use shares or allocations to assign resources in an IORM plan.

5.1.1.1 About the IORM Objective

The I/O Resource Management (IORM) `objective` option specifies the optimization mode for IORM.

The IORM objective is configured individually on every storage server using the CellCLI `ALTER IORMPLAN` command. To deliver consistent overall system performance, ensure every storage server in the storage cluster uses the same IORM configuration settings.

The valid `objective` values are:

- `auto` - Causes IORM to determine the best mode based on active workloads and resource plans. IORM continuously and dynamically determines the optimization objective based on the observed workloads and enabled resource plans. For most use cases, `auto` is the recommended value.

  Starting with Oracle Exadata System Software release 21.2.0, `auto` is the default setting for new deployments.
- `high_throughput` - Optimizes critical DSS workloads that require high throughput. This setting improves disk throughput at the cost of I/O latency.
- `low_latency` - Optimizes critical OLTP workloads that require extremely good disk latency. This setting provides the lowest possible latency at the cost of throughput by limiting disk utilization.
- `balanced` - Balances low disk latency and high throughput, which is useful for a mixture of critical OLTP and DSS workloads. This setting limits disk utilization of large I/Os to a lesser extent than `low_latency` to achieve a balance between latency and throughput.
- `basic` - Use this setting to limit the maximum small I/O latency and otherwise disable I/O prioritization. This is the default setting for new deployments in Oracle Exadata System Software release 20.1.0 and earlier. Specifically, using this setting:
  - IORM guards against extremely high latencies for log writes, buffer cache reads, and other critical I/Os by prioritizing them ahead of smart scans and other disk intensive I/O operations.
  - IORM manages access to flash cache and flash log.
  - IORM manages scans to ensure that resources are shared between different workloads.
  - IORM does not enforce the maximum utilization limits in the IORM plan. Plan allocations are only used to guard against extremely high latencies, and there is no consideration for plan conformance. For stricter plan conformance, and enforcement of maximum utilization limits, the `objective` option must be set to something other than `basic`.

On High Capacity (HC) storage servers, flash IORM protects OLTP latency by prioritizing critical I/Os to flash devices ahead of smart scan and low priority I/Os. This occurs by default and regardless of the `objective` option setting.

On Extreme Flash (EF) storage servers, the IORM objective provides some additional control over flash IORM. On EF storage servers, flash IORM behaves the same as on HC servers when the `objective` option is set to `basic`, `auto`, or `balanced`. However, on EF storage servers, `high_throughput` increases the scan throughput at the cost of critical I/O latency, and `low_latency` provides maximum latency protection at the cost of significant scan throughput degradation when both workloads run concurrently.

5.1.1.2 IORM Plans

Exadata I/O Resource Management (IORM) manages resources using various IORM plans.

Oracle Database Resource Manager enables you to manage resources within a database, also known as intradatabase resource management. A database resource plan specifies how resources are allocated across different workloads, or consumer groups. When a database uses Oracle Exadata Storage Server the database resource plan is communicated to the storage server and is used by IORM. If a database does not have a database resource plan enabled, then intradatabase resource management is disabled and all I/O requests from the database are treated as a single workload.

Interdatabase resource management enables you to manage resources across multiple databases. Interdatabase resource management is configured by using the CellCLI `ALTER IORMPLAN` command to specify an interdatabase plan (`dbplan`). The `dbplan` contains directives that specify resource allocations for specific databases. You can use this feature if you have multiple databases sharing Oracle Exadata Storage Server resources and you want to control the resource allocation to a specific database.

Cluster resource management enables you to manage resources across multiple Oracle Grid Infrastructure clusters sharing the same storage server resources. Cluster resource management is configured by using the CellCLI `ALTER IORMPLAN` command to specify a cluster plan (`clusterplan`). The `clusterplan` contains directives that specify resource allocations that apply to all databases in a specific cluster.

> **Note:**
>
> The cluster plan is first introduced in Oracle Exadata System Software release 21.2.0.

Category resource management allocates resources according to 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 could use category resource management. Category resource management uses Oracle Database Resource Manager to define workload categories in each database, and a category plan (`catplan`) that is defined in each storage server.

> **Note:**
>
> Starting with Oracle Exadata System Software release 21.2.0, the category plan is deprecated and a warning message is issued when a category plan is set.

Apart from the database resource plan definitions, which are configured within each database, Exadata IORM plans (such as the `dbplan` and `clusterplan`) are configured individually on every storage server using the CellCLI `ALTER IORMPLAN` command. To deliver consistent overall system performance, ensure every storage server in the storage cluster uses the same IORM configuration settings.

5.1.1.3 Resource Assignment Methods

You can use shares or allocations to assign resources in an IORM plan.

A **share** value represents the relative importance of each entity. With share-based resource allocation, a higher share value implies higher priority and more access to the I/O resources. For example, a database with a share value of 2 gets twice the resource allocation of a database with a share value of 1.

Valid share values are 1 to 32, with 1 being the lowest share, and 32 being the highest share. The sum of all share values in a plan cannot be greater than 32768.

Share-based resource allocation is the recommended method for the interdatabase plan (`dbplan`). For the cluster plan (`clusterplan`), share-based resource allocation is the only option.

With allocation-based resource management, an **allocation** specifies the resource allocation as a percentage (0-100). Each allocation is associated with a **level**. Valid level values are from 1 to 8, and the sum of allocation values cannot exceed 100 for each level. Resources are allocated to level 1 first, and then remaining resources are allocated to level 2, and so on.

Though not recommended, allocation-based resource management can be used in the interdatabase plan (`dbplan`). For the category plan (`catplan`), allocation-based resource management is the only option.

5.1.2 About Database Resource Management Within a Database

Oracle Database Resource Manager enables you to manage workloads within a database.

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. Database resource management is configured at the database level, using Oracle Database Resource Manager to create database resource plans. You should use this feature if you have multiple types of workloads within a database. You can define a policy for specifying how these workloads share the database resource allocations. If only one database is using the Oracle Exadata Storage Server resources, then this is the only resource management feature that you need.

For each database, you can use Oracle Database Resource Manager for the following tasks:

- **Create resource consumer groups**

  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. Oracle 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 CDB resource plans**

A container database (CDB) resource plan specifies how CPU and I/O resources are allocated among the different pluggable databases (PDBs) that are associated with the same container. The CDB plan is created using Oracle Database Resource Manager. The CDB plan contains a directive for each PDB. The directive defines the number of shares that are allocated to that PDB. The shares define the relative priority of that PDB as compared to other PDBs in the plan.

A maximum utilization limit can be specified for a PDB.

A CDB resource plan also lets you specify `memory_min` and `memory_limit` for each PDB. These parameters define the various cache quotas for each PDB and have no bearing on memory sizing in the database instance.

- **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 Oracle 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.
  - 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.

  In addition to a CDB plan, each PDB can also create a resource plan to manage the workloads running within the PDB. PDBs only support single level plans with a maximum of 8 consumer groups.

- **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. For Oracle Real Application Clusters (Oracle RAC) database running on Oracle Exadata, 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 storage servers (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`.

Related Topics

- Managing Resources with Oracle Database Resource Manager

5.1.3 About Interdatabase Resource Management

Interdatabase resource management enables you to manage resources for multiple databases using the same shared storage.

An interdatabase plan specifies how resources are allocated by percentage or share among multiple databases for each storage server. 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 storage server.

- About Interdatabase IORM Plan Directives
  Interdatabase plan directives are specified using the `DB_UNIQUE_NAME` of the database as the identifier.
- Using Interdatabase Plans for Consolidation and Isolation
  Interdatabase resource management plans help manage resources when consolidating or isolating databases.
- Using IORM to Control Database Access to Exadata Cache Resources
  You can use IORM to manage access to valuable Oracle Exadata Storage Server cache resources.
- About Flash Cache Management in IORM Plans
  I/O Resource Management can provide predictable performance by guaranteeing space in Exadata Smart Flash Cache.
- About XRMEM Cache Management in IORM Plans
- About PMEM Cache Management in IORM Plans
- Tips for Managing Interdatabase Resource Plans
  Note the following information when creating and managing interdatabase resource plans.

Related Topics

- Administering the IORM Plan
  You can administer the IORM plan by using the CellCLI `ALTER IORMPLAN` command.
- ALTER IORMPLAN

5.1.3.1 About Interdatabase IORM Plan Directives

Interdatabase plan directives are specified using the `DB_UNIQUE_NAME` of the database as the identifier.

When using allocation-based resource management, each directive consists of an allocation amount and a level from 1 to 8, which is similar to a database resource plan. For a given plan, the total allocations for 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.

Share-based plans use a relative share instead of percentage allocations and levels. These plans are simpler to implement, but as effective as percentage allocations. Each database is given a share value which is an integer between 1 and 32. The sum of the shares can be up to 32768. Share-based plans support up to 1024 directives within the interdatabase plan. For example, if a critical database, `FINANCE`, has 4 shares, and a low-priority database, `REPORTING`, has 1 share, then during periods of resource contention the `FINANCE` database is four times more likely to issue I/Os compared to the `REPORTING` database.

Oracle Exadata System Software uses the interdatabase plan and database resource plans together to allocate I/O resources.

- First, the interdatabase plan allocates the I/O resources to individual databases.
- 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 shares the storage. This is done automatically for shared-based plans, but not for allocation-based plans. To ensure that any database without an explicit directive can be managed with percentage allocation plans, create an allocation named `OTHER`. Databases without explicit directives are managed using the allocation of the `OTHER` group directive.

Each database that is not explicitly mapped in a share-based plan gets the default share of 1. However, share-based plans can use the `DEFAULT` directive to specify the default values for the different IORM attributes.

If you have databases with the same `DB_UNIQUE_NAME` in different Oracle ASM clusters, then starting with Oracle Exadata System Software release 19.1.0, you can use the `asmcluster` attribute to uniquely identify each database in the interdatabase plan.

5.1.3.2 Using Interdatabase Plans for Consolidation and Isolation

Interdatabase resource management plans help manage resources when consolidating or isolating databases.

Consider the consolidation of four different databases sharing the same Oracle Exadata Storage Servers. If all the databases have equal priority, you can use IORM to equally allocate 25% of the I/O resources to each database. If one database is more important than the others, then you can use IORM to give it a larger share of the I/O resources.

Defining maximum utilization limits may also be useful in consolidation scenarios. By using limits, you can isolate each database in case one sees a sudden workload increase. For example, you could define a maximum utilization limit of 40% for each database, which would guarantee that no database can monopolize the system. However, when a database reaches its assigned limit it cannot take advantage of any spare capacity. Consequently, it is possible for disks to run below full capacity when limits are specified.

Resource management using limits is also ideal for pay-for-performance use cases, where service providers want to guarantee performance to their customers corresponding to the level of service being purchased. For example, customers paying for a lower performance tier should be limited more than customers paying for higher performance.

5.1.3.3 Using IORM to Control Database Access to Exadata Cache Resources

You can use IORM to manage access to valuable Oracle Exadata Storage Server cache resources.

You can use the `flashcache` and `flashlog` attributes in the interdatabase IORM plan to control access to Exadata Smart Flash Cache and flash log. You can set `flashcache=off` to prevent the specified database from using the flash cache. Likewise, setting `flashlog=off` prevents the specified database from using the flash log. By default, all databases can use flash memory for caching and logging.

Likewise, on appropriately configured servers, you can use `xrmemcache` and `xrmemlog` attributes to control access to Exadata RDMA Memory (XRMEM) resources.

> **Note:**
>
> On Exadata X8M and X9M storage server models only, you can use `pmemcache` and `pmemlog` attributes interchangeably with `xrmemcache` and `xrmemlog`. However, you cannot have a mixture of PMEM and XRMEM attributes in the IORM plan.

These attributes allow valuable cache resources to be reserved for mission-critical databases, especially in consolidated environments.

5.1.3.4 About Flash Cache Management in IORM Plans

I/O Resource Management can provide predictable performance by guaranteeing space in Exadata Smart Flash Cache.

With multiple databases and pluggable databases (PDBs) sharing the storage, flash cache space becomes a critical resource that requires management. IORM can reserve space for critical databases or PDBs while preventing less important or rogue entities from consuming the entire flash cache.

You can use the following attributes in the interdatabase plan to set limits for flash cache resources. A hard limit means that the specified limit cannot be exceeded even when the cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `flashCacheMin` — Specifies the minimum amount of flash cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

  Because `flashCacheMin` is a guaranteed reservation, the sum of `flashCacheMin` across all directives should be less than the size of the flash cache to ensure that each database gets its respective quota.

- `flashCacheLimit` — Specifies the soft maximum amount of flash cache space that is available to the database. If the flash cache is not full, a database can exceed the `flashCacheLimit` value.

- `flashCacheSize` — Specifies the hard maximum amount of flash cache space that is available to the database. The `flashCacheSize` value cannot be exceeded at any time.

  However, if you set `flashCacheSize` to a value that is lower than the current space occupied by the database, then starting with Oracle Exadata System Software release 20.1.0 excess data is proactively cleared from the cache. Previously, excess data was only removed when overwritten by other data.

  Starting with Oracle Exadata System Software release 19.2.0, `flashCacheSize` is not a guaranteed reservation if the sum of the `flashCacheSize` across all directives is more than the size of the flash cache. In this case, you can also specify `flashCacheMin` to define a guaranteed minimum quota.

Within each database, you can manage the minimum and maximum quotas for PDBs with Oracle Database Resource Manager, in a container database (CDB) resource plan using the `memory_min` and `memory_limit` directives.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies `flashcachemin` and `flashcachesize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `flashcachemin` setting, and the PDB-specific `memory_limit` values represent fractions of the `flashcachesize`.

However, if the interdatabase IORM plan directive specifies `flashcachesize` without `flashcachemin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `flashcachesize`.

5.1.3.5 About XRMEM Cache Management in IORM Plans

I/O Resource Management (IORM) can provide predictable performance by guaranteeing space in Exadata RDMA Memory Cache (XRMEM cache).

With multiple databases and pluggable databases (PDBs) sharing the storage, XRMEM cache space becomes a critical resource that requires management. IORM can reserve space for critical databases or PDBs while preventing less important or rogue entities from consuming the entire XRMEM cache.

You can use the following attributes in the interdatabase plan to set limits for XRMEM cache resources. A hard limit means that the specified limit cannot be exceeded even when the cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `xrmemCacheMin` — Specifies the minimum amount of XRMEM cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

  Because `xrmemCacheMin` is a guaranteed reservation, the sum of `xrmemCacheMin` across all directives should be less than the size of the XRMEM cache to ensure that each database gets its respective quota.
- `xrmemCacheLimit` — Specifies the soft maximum amount of XRMEM cache space that is available to the database. If the XRMEM cache is not full, a database can exceed the `xrmemCacheLimit` value.
- `xrmemCacheSize` — Specifies the hard maximum amount of XRMEM cache space that is available to the database. The `xrmemCacheSize` value cannot be exceeded at any time.

  However, if you set `xrmemCacheSize` to a value that is lower than the current space occupied by the database, then excess data is proactively cleared from the cache.

  `xrmemCacheSize` is not a guaranteed reservation if the sum of the `xrmemCacheSize` across all directives is more than the size of the XRMEM cache. In this case, you can also specify `xrmemCacheMin` to define a guaranteed minimum quota.

Note:

On Exadata X8M and X9M storage server models only, you can interchangeably use attribute names starting with `pmem` instead of `xrmem`. For example, `pmemCacheMin` instead of `xrmemCacheMin`. However, you cannot have a mixture of PMEM and XRMEM attributes in the IORM plan.

Within each database, you can manage the minimum and maximum quotas for PDBs with Oracle Database Resource Manager, in a container database (CDB) resource plan using the `memory_min` and `memory_limit` directives.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies `xrmemCacheMin` and `xrmemCacheSize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `xrmemCacheMin` setting, and the PDB-specific `memory_limit` values represent fractions of the `xrmemCacheSize`.

However, if the interdatabase IORM plan directive specifies `xrmemCacheSize` without `xrmemCacheMin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `xrmemCacheSize`.

5.1.3.6 About PMEM Cache Management in IORM Plans

Note:

This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see About XRMEM Cache Management in IORM Plans.

I/O Resource Management (IORM) can provide predictable performance by guaranteeing space in the persistent memory (PMEM) cache, which is available in Oracle Exadata Storage Server X8M and X9M.

With multiple databases and pluggable databases (PDBs) sharing the storage, PMEM cache space becomes a critical resource that requires management. IORM can reserve space for critical databases or PDBs while preventing less important or rogue entities from consuming the entire PMEM cache.

You can use the following attributes in the interdatabase plan to set limits for PMEM cache resources. A hard limit means that the specified limit cannot be exceeded even when the cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `pmemCacheMin` — Specifies the minimum amount of PMEM cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

  Because `pmemCacheMin` is a guaranteed reservation, the sum of `pmemCacheMin` across all directives should be less than the size of the PMEM cache to ensure that each database gets its respective quota.

- `pmemCacheLimit` — Specifies the soft maximum amount of PMEM cache space that is available to the database. If the PMEM cache is not full, a database can exceed the `pmemCacheLimit` value.

- `pmemCacheSize` — Specifies the hard maximum amount of PMEM cache space that is available to the database. The `pmemCacheSize` value cannot be exceeded at any time.

  However, if you set `pmemCacheSize` to a value that is lower than the current space occupied by the database, then starting with Oracle Exadata System Software release 20.1.0 excess data is proactively cleared from the cache. Previously, excess data was only removed when overwritten by other data.

  `pmemCacheSize` is not a guaranteed reservation if the sum of the `pmemCacheSize` across all directives is more than the size of the PMEM cache. In this case, you can also specify `pmemCacheMin` to define a guaranteed minimum quota.

Within each database, you can manage the minimum and maximum quotas for PDBs with Oracle Database Resource Manager, in a container database (CDB) resource plan using the `memory_min` and `memory_limit` directives.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies `pmemCacheMin` and `pmemCacheSize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `pmemCacheMin` setting, and the PDB-specific `memory_limit` values represent fractions of the `pmemCacheSize`.

However, if the interdatabase IORM plan directive specifies `pmemCacheSize` without `pmemCacheMin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `pmemCacheSize`.

5.1.3.7 Tips for Managing Interdatabase Resource Plans

Note the following information when creating and managing interdatabase resource plans.

- If Oracle Exadata 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 Oracle Exadata System Software 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 Oracle Exadata System Software does not use Oracle Database Resource Manager for these databases. All of the I/O requests are treated the same.
- For share-based plans, each database gets its own directive even when it is not explicitly named in the plan. Oracle Exadata System Software uses the database resource plan of the database to determine how the allocation is to be distributed between the consumer groups in the database.
- Interdatabase plan directives are specified using the `DB_UNIQUE_NAME` of the database as the identifier.

  By itself, the `DB_UNIQUE_NAME` value is not case-sensitive. However, IORM may not work correctly when multiple instances of the same Oracle RAC database use inconsistent cases for `DB_UNIQUE_NAME` (for example, `PROD` and `prod`). Therefore, ensure that the `DB_UNIQUE_NAME` value is case-consistent across each Oracle RAC database.

  If no value is specified, the `DB_UNIQUE_NAME` value is derived from `DB_NAME`. In that case, ensure that the `DB_NAME` value is case-consistent across the Oracle RAC database.
- If you have databases with the same `DB_UNIQUE_NAME` but associated with different Oracle ASM clusters, then, starting with Oracle Exadata System Software release 19.1.0, you can use the `asmcluster` attribute to uniquely identify each database when specifying directives.
- Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding container database (CDB) plan.

  Consequently, if an interdatabase IORM plan directive specifies `flashcachemin` and `flashcachesize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `flashcachemin` setting, and the PDB-specific `memory_limit` values represent fractions of the `flashcachesize`.

  However, if the interdatabase IORM plan directive specifies `flashcachesize` without `flashcachemin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `flashcachesize`.

  The same applies to the Exadata RDMA Memory Cache (XRMEM cache) regarding the relationship between the PDB-specific quotas in the CDB plan (`memory_min` and `memory_limit`) and the cache-specific settings in the interdatabase IORM plan (`xrmemcachemin` and `xrmemcachesize`).

5.1.4 About Cluster Resource Management

Cluster resource management enables you to manage resources for multiple clusters using the same shared storage.

Note:

The cluster plan is first introduced in Oracle Exadata System Software release 21.2.0.

An I/O Resource Management (IORM) cluster plan specifies how storage server resources are allocated across multiple clusters. Each directive in a cluster plan specifies an allocation to a cluster, rather than an individual database or consumer group.

For example, consider a system hosting two clusters; `clusterA` and `clusterB`. Now, imagine a cluster plan with share-based resource allocation where `clusterA` has three shares, and `clusterB` has one share. In that case, and in the absence of any other IORM plans, all of the databases running in `clusterA` share 75% of the I/O resources, while the databases in `clusterB` share the remaining 25%.

The cluster plan can work in conjunction with an interdatabase resource plan, but only if the interdatabase resource plan does not use allocation-based resource management (using `allocation` and `level` directives). In this case, directives from both plans are applied to determine the share of resources.

So, continuing from the previous example, imagine that the databases in `clusterA` are in an interdatabase resource plan with share-based resource allocation. In that case, the resources allocated in the cluster plan to `clusterA` are further divided and shared amongst the databases by using the directives in the interdatabase resource plan.

The cluster plan cannot work in conjunction with a category plan. That is, you cannot set IORM cluster plan directives when category plan directives exist. Likewise, you cannot set category plan directives when cluster plan directives exist.

The cluster plan is configured and enabled on each storage server by using the CellCLI `ALTER IORMPLAN` command. To operate the cluster plan, ASM-scoped security must also be configured.

Related Topics

- ALTER IORMPLAN
- Configuring Data Security for Exadata Storage Servers

5.1.5 About Category Resource Management

Categories represent collections of consumer groups across all databases.

Note:

Starting with Oracle Exadata System Software release 21.2.0, the category plan is deprecated and a warning message is issued when a category plan is set.

Oracle Database Resource Manager enables you to specify a category for every consumer group. 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 Oracle Exadata Storage Server.

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.

Category plans are configured and enabled using the CellCLI utility on the cell. Only one category plan can be enabled at a time. The predefined categories provided in Oracle Database are described in the following table, along with sample percentages.

Table 5-1 Sample Category Resource Management 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	not set	not set
INTERACTIVE	For high-priority, performance-sensitive work, such as OLTP transactions.	not set	70	not set
BATCH	For low-priority work, such as noncritical reports and backup.	not set	not set	70
MAINTENANCE	For low-priority work, such as automated tasks.	not set	not set	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.	not set	not set	20

The sample plan shown in the above table 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.

Related Topics

- Administering Database Resource Management
  To set up database resource management, you must use Oracle Database Resource Manager to configure the consumer groups, assign sessions to consumer groups, create a database resource plan, and enable the plan.

5.1.6 About Consumer Groups and Resource Plans

Oracle Exadata provides out-of-the-box consumer groups and resource plans specifically designed for data warehouses that use Oracle Exadata System Software.

These resource plans can be modified to suit the needs of your environment.

The following consumer groups are 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 resource plans are for data warehouses:

- 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.
- ETL_CRITCAL_PLAN Resource Plan
  The ETL_CRITICAL_PLAN prioritizes ETL over DSS queries.

5.1.6.1 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 the bandwidth.

Table 5-2 DSS_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
SYS_GROUP	75	not set	not set	not set
DSS_CRITICAL_GROUP	not set	75	not set	not set
DSS_GROUP	not set	not set	75	not set
ETL_GROUP	not set	not set	not set	45
BATCH_GROUP	not set	not set	not set	45
ORA$DIAGNOSTICS	not set	5	not set	not set
ORA$AUTOTASK_SUB_PLAN	not set	5	not set	not set

Table 5-2 (Cont.) DSS_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
OTHER_GROUPS	not set	not set	not set	10

As shown in the previous table, 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, but instead the allocation is given to the DSS_GROUP group at level 3.

5.1.6.2 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 5-3 ETL_CRITICAL_PLAN Resource Plan for Data Warehouses

Consumer Group	Level 1 (%)	Level 2 (%)	Level 3 (%)	Level 4 (%)
SYS_GROUP	75	not set	not set	not set
DSS_CRITICAL_GROUP	not set	35	not set	not set
DSS_GROUP	not set	10	not set	not set
ETL_GROUP	not set	35	not set	not set
BATCH_GROUP	not set	10	not set	not set
ORA$DIAGNOSTICS	not set	3	not set	not set
ORA$AUTOTASK_SUB_PLAN	not set	3	not set	not set
OTHER_GROUPS	not set	3	not set	not set

5.1.7 About CDB Plans and Pluggable Databases

The container database (CDB) can have multiple workloads within multiple PDBs competing for resources.

The Oracle Multitenant container database (CDB) supports many user-defined pluggable databases (PDBs). In a CDB, resources are managed at the following levels:

- **CDB level**: Oracle Database Resource Manager manages the workloads for multiple PDBs that are contending for system and CDB resources. The administrator can specify how resources are allocated to PDBs, and can limit the resource utilization of specific PDBs.
- **PDB level**: Oracle Database Resource Manager manages the workloads within each PDB.

The following example outlines a CDB plan for three PDBs named SALES, SERVICES and HR. The PDBs have different shares and max utilization limits in the CDB plan.

PDB Name	Directive for Shares	Directive for Utilization Limit	Memory_min	Memory_limit
SALES	3	Unlimited	20	not set
SERVICES	3	Unlimited	20	not set
HR	1	70	not set	50

5.2 Administering IORM

You can perform various administrative tasks related to I/O Resource Management (IORM).

To perform the tasks, use the `DBMS_RESOURCE_MANAGER` package to define database resource plans on the database servers, and the CellCLI utility to specify the IORM plan on each cell.

- Setting the IORM Objective
  Use the CellCLI `ALTER IORMPLAN` command to set the IORM objective.
- Administering Database Resource Management
  To set up database resource management, you must use Oracle Database Resource Manager to configure the consumer groups, assign sessions to consumer groups, create a database resource plan, and enable the plan.
- Administering the IORM Plan
  You can administer the IORM plan by using the CellCLI `ALTER IORMPLAN` command.
- Listing an I/O Resource Management Plan
  You can view the current interdatabase plan for a storage server using the CellCLI `LIST IORMPLAN` command on the storage server.
- Managing Flash Cache Quotas for Databases and PDBs
  IORM enables you to control how you want the flash cache to be shared among different databases and pluggable databases (PDBs).
- Managing XRMEM Cache Quotas for Databases and PDBs
- Managing PMEM Cache Quotas for Databases and PDBs
- Using IORM Profiles
  I/O Resource Management (IORM) interdatabase plans support profiles to ease management, and configuration of interdatabase plans for hundreds of databases.
- Verifying the Configuration of I/O Resource Management
  Use this checklist to verify that I/O Resource Management (IORM) is configured correctly.

Related Topics

- Using the CellCLI Utility
  You use the Cell Control Command-Line Interface (CellCLI) utility to manage Oracle Exadata System Software.

5.2.1 Setting the IORM Objective

Use the CellCLI `ALTER IORMPLAN` command to set the IORM objective.

For example:

```
CellCLI> ALTER IORMPLAN objective=auto
```

Note:

The IORM objective is configured individually on every storage server using the CellCLI `ALTER IORMPLAN` command. To deliver consistent overall system performance, ensure every storage server in the storage cluster uses the same IORM configuration settings.

Related Topics

- About the IORM Objective
  The I/O Resource Management (IORM) `objective` option specifies the optimization mode for IORM.
- ALTER IORMPLAN

5.2.2 Administering Database Resource Management

To set up database resource management, you must use Oracle Database Resource Manager to configure the consumer groups, assign sessions to consumer groups, create a database resource plan, and enable the plan.

- Setting Up Consumer Groups and Categories
  Consumer groups and categories are set up with the procedures in the PL/SQL `DBMS_RESOURCE_MANAGER` package.
- Assigning Sessions to Consumer Groups
  You can assign a session to a resource consumer group manually or automatically using consumer group mapping rules.
- Creating a CDB Plan
  The CDB plan manages CPU resources on the database servers, and flash cache space and I/O bandwidth on the Exadata storage servers.
- 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()`.
- 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 an Oracle Scheduler window with a resource plan.
- Managing Fast File Creation
  Oracle Exadata System Software features fast file creation, allowing accelerated initialization of data files.
- Managing Data Import
  You can control the priority of ETL as well as the amount of I/O resources that ETL consumes using I/O Resource Management (IORM).
- Managing Oracle Recovery Manager Backups and Copies
  You can use I/O Resource Management (IORM) to control the resource consumption and priority of RMAN I/Os.

5.2.2.1 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()`

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 5-1 Setting Up Consumer Groups and Categories with PL/SQL in the Database

This example shows how to set up consumer groups and categories in a database. The `MAINTENANCE` category is predefined, and is not created in this example.

```
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;
/
```

Example 5-2   Consumer Groups and Categories in an Oracle Database

This example shows a query on the `DBA_RSRC_CONSUMER_GROUPS` view.

```
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
AUTO_TASK_CONSUMER_GROUP       OTHER
DEFAULT_CONSUMER_GROUP         OTHER

13 rows selected
```

Related Topics

- *Oracle Database Administrator’s Guide*
- *Oracle Database Reference*

5.2.2.2 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 5-3. Before running the `SET_CONSUMER_GROUP_MAPPING` procedure, you must first create a pending area.

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.

Example 5-3 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;
/
```

See Also:

Oracle Database Administrator’s Guide for additional information about the following:

- Assigning Sessions to Resource Consumer Groups
- Creating Consumer Group Mapping Rules

5.2.2.3 Creating a CDB Plan

The CDB plan manages CPU resources on the database servers, and flash cache space and I/O bandwidth on the Exadata storage servers.

CDB plans are created using the PL/SQL procedures `DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN()` and `CREATE_CDB_PLAN_DIRECTIVE()`. The CDB plan can only be configured from the root PDB.

Example 5-4 Using a CDB Plan to Distribute Resources Between PDBs

This example shows how to distribute resources between three PDBs named SALES, SERVICES and HR. SALES and SERVICES have higher priority and get three shares each compared to one share for HR. The limit on the HR PDB is set to 70% maximum utilization limit.

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();

DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN(
    plan    => ''NEWCDB_PLAN ',
    comment => 'CDB resource plan for newcdb');

  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => 'NEWCDB_PLAN',
    pluggable_database     => 'SALESPDB',
    shares                 => 3,
    memory_min             => 20,
    utilization_limit      => 100);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => ' NEWCDB_PLAN ',
    pluggable_database     => 'SERVICESPDB',
    shares                 => 3,
    memory_min             => 20,
    memory_limit           => 75);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => ' NEWCDB_PLAN ',
    pluggable_database     => 'HRPDB',
    shares                 => 1,
    memory_limit           => 50,
    utilization_limit      => 70);

DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

5.2.2.4 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 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.

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.

Example 5-5 Sharing Resources Across Applications

In this example, assume you have 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.

```
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;
/
```

Example 5-6   Sharing Resources Across Workloads

In this example, assume you want to prioritize one workload over another. For example, suppose that you load data into your data warehouse while also 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 (reporting and ad-hoc) 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 the query consumer groups do not use all of their allocation. You can use resource plan levels to specify the allocation priorities.

```
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;
/
```

Related Topics

- *Oracle Database Administrator’s Guide*

5.2.2.5 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 an Oracle Scheduler window with a resource plan.

When the Oracle Scheduler window opens, the resource plan is enabled. When the Oracle 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.

Related Topics

- *Oracle Database Administrator’s Guide*

5.2.2.6 Managing Fast File Creation

Oracle Exadata System Software features fast file creation, allowing accelerated initialization of data files.

This feature automatically runs whenever you create a new tablespace, add a data file to an existing tablespace, or autoextend an existing tablespace. Oracle Exadata System Software 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 (IORM), you can control the priority of fast file creations for creating a new tablespace or adding a data file 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`.

Because autoextending an existing tablespace is a brief and often time-critical operation, you cannot modify its priority using IORM.

Example 5-7   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;
/
```

5.2.2.7 Managing Data Import

You can control the priority of ETL as well as the amount of I/O resources that ETL consumes using I/O Resource Management (IORM).

Data import, or extract, transform, load (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.

To manage ETL, do the following:

- Map the ETL sessions to the `ETL_GROUP` consumer group.

  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.
- Include the `ETL_GROUP` group in your resource plans.

To import non-compressed data as compressed data,

Example 5-8 Mapping a Program to the ETL_GROUP Consumer Group

This example shows how to map 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.

Related Topics

- TRANSFORM

5.2.2.8 Managing Oracle Recovery Manager Backups and Copies

You can use I/O Resource Management (IORM) to control the resource consumption and priority of RMAN I/Os.

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. You can use IORM for a greater degree of control over the resource consumption and priority of RMAN I/Os. For example, you can map RMAN to a low priority consumer group. If the Oracle Exadata Storage Server is busy, then the RMAN operations run very slowly and not interfere with the other database operations. However, whenever the Oracle Exadata Storage Server 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 the following example.

Example 5-9 Using Consumer Groups to Manage Resources

This example shows how to map the `BACKUP` function to the `BATCH_GROUP` consumer group and the `COPY` function to the `MAINTENANCE_GROUP` consumer group.

```
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;
/
```

5.2.3 Administering the IORM Plan

You can administer the IORM plan by using the CellCLI `ALTER IORMPLAN` command.

- Setting the IORM Plan
  Use the CellCLI `ALTER IORMPLAN` command to set the IORM plan.
- Using Share-Based Resource Management
  A share value represents the relative importance of each entity.
- Using Allocation-Based Resource Management
  In IORM, an allocation value specifies the resource allocation as a percentage (0-100).
- Using the limit Attribute
  The `limit` attribute specifies the maximum flash I/O utilization limit as a percentage of the available resources.
- Controlling Access to Flash Cache and Flash Log
  You can use IORM to manage access to flash cache and flash log.
- Using Flash Cache Attributes
  You can use flash cache attributes in the IORM plan to guarantee space allocation in Exadata Smart Flash Cache.
- Controlling Access to XRMEM Resources
- Using XRMEM Cache Attributes
- Controlling Access to PMEM Cache and PMEM Log
- Using PMEM Cache Attributes
- Using the role Attribute
  The `role` attribute allows a different allocation to be specified, based on whether the database has the Oracle Data Guard `primary` or `standby` role.

- Using the asmcluster Attribute
  You can use the `asmcluster` attribute to uniquely identify databases with the same `DB_UNIQUE_NAME`.
- Resetting Default Values in an IORM Plan
  Use an empty string to reset an IORM plan.

5.2.3.1 Setting the IORM Plan

Use the CellCLI `ALTER IORMPLAN` command to set the IORM plan.

For example:

```
CellCLI> ALTER IORMPLAN                                                -
         dbplan=((name=sales01, share=4),                              -
                 (name=sales02, share=3),                              -
                 (name=dev01, share=1),                                -
                 (name=DEFAULT, share=2))
```

Because of the length and complexity of typical IORM plan definitions, consider putting the `ALTER IORMPLAN` command in a script file and run it by using the CellCLI `START` command.

Note:

Exadata I/O Resource Management (IORM) is configured individually on every storage server using the CellCLI `ALTER IORMPLAN` command. To deliver consistent overall system performance, ensure every storage server in the storage cluster uses the same IORM configuration settings.

Related Topics

- IORM Plans
  Exadata I/O Resource Management (IORM) manages resources using various IORM plans.
- ALTER IORMPLAN
- START and @

5.2.3.2 Using Share-Based Resource Management

A share value represents the relative importance of each entity.

With share-based resource allocation, a higher share value implies higher priority and more access to the I/O resources. For example, a database with a share value of 2 gets twice the resource allocation of a database with a share value of 1.

Valid share values are 1 to 32, with 1 being the lowest share, and 32 being the highest share. The sum of all share values in a plan cannot be greater than 32768.

Share-based resource allocation is the recommended method for the interdatabase plan (`dbplan`). For the cluster plan (`clusterplan`), share-based resource allocation is the only option.

The following example illustrates how to use share-based resource management in an interdatabase plan. Consider four databases sharing the same Oracle Exadata Storage Server resources. The four databases are:

- A critical 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 `PROD` 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 `PROD` and the `DW` databases.

To ensure a reasonable distribution of I/O resources, you can define a share-based interdatabase plan as follows:

```
CellCLI> ALTER IORMPLAN                                    -
         dbplan=((name=prod, share=16),                    -
                 (name=dw, share=4),                       -
                 (name=prod_test, share=2),                -
                 (name=DEFAULT, share=1))
```

By using the example interdatabase plan, the critical OLTP database (`PROD`) gets priority when there is contention for I/O resources. Specifically, the I/O share for `PROD` is 4 times greater than `DW`, 8 times greater than `PROD_TEST`, and a 16 times greater that the default share that is assigned to `PROD_DEV`.

At any time, you change the share allocations to adjust the relative priorities.

5.2.3.3 Using Allocation-Based Resource Management

In IORM, an allocation value specifies the resource allocation as a percentage (0-100).

With allocation-based resource management, each allocation is associated with a level. Valid level values are from 1 to 8, and the sum of allocation values cannot exceed 100 for each level. Resources are allocated to level 1 first, and then remaining resources are allocated to level 2, and so on.

Though not recommended, allocation-based resource management can be used in the interdatabase plan (`dbplan`). For the category plan (`catplan`), allocation-based resource management is the only option.

The following example illustrates how to use allocation-based resource management in an interdatabase plan. Consider four databases sharing the same Oracle Exadata Storage Server resources. The four databases are:

- A critical 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 `PROD` 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 `PROD` and the `DW` databases.

To ensure a reasonable distribution of I/O resources, you can define an allocation-based interdatabase plan as follows:

```
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))
```

By using the example interdatabase plan:

- The critical OLTP database (`PROD`) is guaranteed 80 percent of the I/O resources during periods of I/O resource contention.
- The `DW` database gets 80 percent any remaining unused I/O.
- Finally, the `PROD_TEST` and `PROD_DEV` databases get any unused I/O in the amount of 50 percent and 40 percent respectively. Also, in this example, a 10% allocation is reserved for `OTHER` databases that are not explicitly listed in the plan.

At any time, you change the allocations to adjust the resource assignments.

5.2.3.4 Using the limit Attribute

The `limit` attribute specifies the maximum flash I/O utilization limit as a percentage of the available resources.

You can use the `limit` attribute to restrict the flash I/O utilization for an entity in the interdatabase plan or the cluster plan. This attribute ensures that the specified entity never utilizes flash I/O resources beyond the specified limit. The attribute applies only to I/O on flash devices, which includes flash-based grid disks and Exadata Smart Flash Cache.

For example, if a production and test database are sharing Oracle Exadata Storage Server resources, then you could set a maximum flash utilization limit for the test database in the interdatabase plan as follows:

```
ALTER IORMPLAN dbplan=((name=prod),                   -
                       (name=test, limit=20),         -
                       (name=DEFAULT, limit=10))
```

If maximum utilization limits are specified, then excess capacity might not be used. As a result, it is possible for flash devices to run below full capacity when maximum utilization limits are specified.

Note:

Specifying low `limit` values can have a significant performance impact and is generally not advisable.

Resource management using limits is ideal for pay-for-performance use cases but should not be used to implement fairness. Instead, use the `share` attribute to ensure equitable distribution of I/O resources.

5.2.3.5 Controlling Access to Flash Cache and Flash Log

You can use IORM to manage access to flash cache and flash log.

You can use the `flashcache` and `flashlog` attributes in the interdatabase plan to control access to flash cache and flash log. When set in the interdatabase plan, these attributes control access by a specific database.

The following example shows how to disable flash cache and flash log for the `prod_test` and `prod_dev` databases. In the example, flash log is also disabled for the `dw_test` database.

```
CellCLI> ALTER IORMPLAN                                           -
         dbplan=((name=prod, flashcache=on, flashlog=on),         -
                (name=dw, flashcache=on, flashlog=on),            -
                (name=prod_test, flashcache=off, flashlog=off),   -
                (name=prod_dev, flashcache=off, flashlog=off),    -
                (name=dw_test, flashcache=on, flashlog=off))
```

You can also use these attributes in conjunction with other attributes. For example:

```
CellCLI> ALTER IORMPLAN                                                    -
         dbplan=((name=prod, share=8, flashcache=on, flashlog=on),         -
                (name=dw, share=6, flashcache=on, flashlog=on),            -
                (name=prod_test, share=2, flashcache=off, flashlog=off),   -
                (name=prod_dev, share=1, flashcache=off, flashlog=off),    -
                (name=dw_test, share=2, flashcache=on, flashlog=off),      -
                (name=other, share=1))
```

You do not need to explicitly set `flashcache=on` or `flashlog=on` because they are the default settings.

5.2.3.6 Using Flash Cache Attributes

You can use flash cache attributes in the IORM plan to guarantee space allocation in Exadata Smart Flash Cache.

Using flash cache attributes, IORM can reserve space for critical databases while preventing less important or rogue entities from consuming the entire flash cache. These attributes can only be specified in an interdatabase plan, and are configured using the CellCLI utility.

You can use the following attributes to set limits for flash cache resources. A hard limit means that the specified limit cannot be exceeded even when the memory cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `flashCacheMin` — Specifies the minimum amount of flash cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

  Because `flashCacheMin` is a guaranteed reservation, the sum of `flashCacheMin` across all directives should be less than the size of the flash cache to ensure that each database gets its respective quota.
- `flashCacheLimit` — Specifies the soft maximum amount of flash cache space that is available to the database. If the flash cache is not full, a database can exceed the `flashCacheLimit` value.
- `flashCacheSize` — Specifies the hard maximum amount of flash cache space that is available to the database. The `flashCacheSize` value cannot be exceeded at any time.

  However, if you set `flashCacheSize` to a value that is lower than the current space occupied by the database, then starting with Oracle Exadata System Software release 20.1.0 excess data is proactively cleared from the cache. Previously, excess data was only removed when overwritten by other data.

  Starting with Oracle Exadata System Software release 19.2.0, `flashCacheSize` is not a guaranteed reservation if the sum of the `flashCacheSize` across all directives is more than the size of the flash cache. In this case, you can also specify `flashCacheMin` to define a guaranteed minimum quota.

Example 5-10 Configuring an Interdatabase Plan with Flash Cache Attributes

This example shows how to create an interdatabase plan with flash cache attributes.

In the example, the `sales` and `test` databases are allocated an amount of space in the flash cache by using the `flashCacheSize` parameter. But, the databases cannot exceed the specified allocation, even if the flash cache has free space.

The `finance` and `dev` databases use `flashCacheMin` for guaranteed minimum quotas. These databases can also exceed the specified `flashCacheLimit` size when there is free space in the flash cache.

```
ALTER IORMPLAN                                                    -
 dbplan=((name=sales, share=8, flashCacheSize=10G),               -
         (name=finance, share=8, flashCacheLimit=10G, flashCacheMin=2G), -
         (name=dev, share=2, flashCacheLimit=4G, flashCacheMin=1G),      -
         (name=test, share=1, limit=10, flashCacheSize=1G))
```

Example 5-11 Configuring an Interdatabase Plan with Flash Cache Attributes

Starting with Oracle Exadata System Software release 19.3.0, you can specify both `flashCacheMin` and `flashCacheSize` for the same target.

```
ALTER IORMPLAN                                             -
 dbplan=((name=sales, share=8, flashCacheMin=3G, flashCacheSize=10G),    -
         (name=finance, share=8, flashCacheLimit=10G, flashCacheMin=2G), -
         (name=dev, share=2, flashCacheLimit=4G, flashCacheMin=1G),      -
         (name=test, share=1, limit=10, flashCacheSize=1G))
```

5.2.3.7 Controlling Access to XRMEM Resources

You can use IORM to manage access to Exadata RDMA Memory (XRMEM) resources.

You can use the `xrmemcache` and `xrmemlog` attributes in the interdatabase plan to control access to XRMEM cache and XRMEM log. When set in the interdatabase plan, these attributes control access by a specific database.

Note:

On Exadata X8M and X9M storage server models only, you can use `pmemcache` and `pmemlog` attributes interchangeably with `xrmemcache` and `xrmemlog`. However, you cannot have a mixture of PMEM and XRMEM attributes in the IORM plan.

The following example shows how to disable XRMEM cache and XRMEM log for the `prod_test` and `prod_dev` databases. In the example, XRMEM log is also disabled for the `dw_test` database.

```
CellCLI> ALTER IORMPLAN                                         -
         dbplan=((name=prod, xrmemcache=on, xrmemlog=on),         -
                (name=dw, xrmemcache=on, xrmemlog=on),            -
                (name=prod_test, xrmemcache=off, xrmemlog=off),   -
                (name=prod_dev, xrmemcache=off, xrmemlog=off),    -
                (name=dw_test, xrmemcache=on, xrmemlog=off))
```

You can also use these attributes in conjunction with other attributes. For example:

```
CellCLI> ALTER IORMPLAN                                                  -
         dbplan=((name=prod, share=8, xrmemcache=on, xrmemlog=on),         -
                (name=dw, share=6, xrmemcache=on, xrmemlog=on),            -
                (name=prod_test, share=2, xrmemcache=off, xrmemlog=off),   -
                (name=prod_dev, share=1, xrmemcache=off, xrmemlog=off),    -
                (name=dw_test, share=2, xrmemcache=on, xrmemlog=off),      -
                (name=other, share=1))
```

You do not need to explicitly set `xrmemcache=on` or `xrmemlog=on` because they are the default settings.

5.2.3.8 Using XRMEM Cache Attributes

You can use XRMEM cache attributes in the IORM plan to guarantee space allocation in the Exadata RDMA Memory Cache (XRMEM cache).

Using XRMEM cache attributes, IORM can reserve space for critical databases while preventing less important or rogue entities from consuming the entire XRMEM cache. These attributes can only be specified in an interdatabase plan, and are configured using the CellCLI utility.

You can use the following attributes to set limits for XRMEM cache resources. A hard limit means the database cannot exceed its quota even when the memory cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `xrmemCacheMin` — Specifies the minimum amount of XRMEM cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

Because `xrmemCacheMin` is a guaranteed reservation, the sum of `xrmemCacheMin` across all directives should be less than the size of the XRMEM cache to ensure that each database gets its respective quota.

- `xrmemCacheLimit` — Specifies the soft maximum amount of XRMEM cache space that is available to the database. If the XRMEM cache is not full, a database can exceed the `xrmemCacheLimit` value.
- `xrmemCacheSize` — Specifies the hard maximum amount of XRMEM cache space that is available to the database. The `xrmemCacheSize` value cannot be exceeded at any time.

  However, if you set `xrmemCacheSize` to a value that is lower than the current space occupied by the database, then excess data is proactively cleared from the cache.

  `xrmemCacheSize` is not a guaranteed reservation if the sum of the `xrmemCacheSize` across all directives is more than the size of the XRMEM cache. In this case, you can also specify `xrmemCacheMin` to define a guaranteed minimum quota.

> **Note:**
>
> On Exadata X8M and X9M storage server models only, you can interchangeably use attribute names starting with `pmem` instead of `xrmem`. For example, `pmemCacheMin` instead of `xrmemCacheMin`. However, you cannot have a mixture of PMEM and XRMEM attributes in the IORM plan.

Example 5-12 Configuring an Interdatabase Plan with XRMEM Cache Attributes

This example shows how to create an interdatabase plan with XRMEM cache attributes.

In the example, the `sales` and `test` databases are allocated an amount of space in the XRMEM cache by using the `xrmemCacheSize` parameter. But, the databases cannot exceed the specified allocation, even if the XRMEM cache has free space.

The `finc` and `dev` databases use `xrmemCacheMin` for guaranteed minimum quotas. These databases can also exceed the specified `xrmemCacheLimit` size when there is free space in the XRMEM cache.

The example plan also contains various flash cache attributes.

```
ALTER IORMPLAN
dbplan=
     -
((name=sales, share=8, xrmemCacheSize=2G,
flashCacheSize=10G),                                   -
(name=finc, share=8, xrmemCacheMin=1G, xrmemCacheLimit=2G,
flashCacheLimit=10G, flashCacheMin=2G), -
(name=dev, share=2, xrmemCacheMin=500M, xrmemCacheLimit=1G,
flashCacheLimit=4G, flashCacheMin=1G), -
(name=test, share=1, limit=10, xrmemCacheSize=200M))
```

5.2.3.9 Controlling Access to PMEM Cache and PMEM Log

Note:

This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see Controlling Access to XRMEM Resources.

You can use IORM to manage access to persistent memory (PMEM) cache and PMEM log.

You can use the `pmemcache` and `pmemlog` attributes in the interdatabase plan to control access to PMEM cache and PMEM log. When set in the interdatabase plan, these attributes control access by a specific database.

The following example shows how to disable PMEM cache and PMEM log for the `prod_test` and `prod_dev` databases. In the example, PMEM log is also disabled for the `dw_test` database.

```
CellCLI> ALTER IORMPLAN                                                   -
         dbplan=((name=prod, pmemcache=on, pmemlog=on),                   -
                (name=dw, pmemcache=on, pmemlog=on),                      -
                (name=prod_test, pmemcache=off, pmemlog=off),             -
                (name=prod_dev, pmemcache=off, pmemlog=off),              -
                (name=dw_test, pmemcache=on, pmemlog=off))
```

You can also use these attributes in conjunction with other attributes. For example:

```
CellCLI> ALTER IORMPLAN                                                            -
         dbplan=((name=prod, share=8, pmemcache=on, pmemlog=on),                   -
                (name=dw, share=6, pmemcache=on, pmemlog=on),                      -
                (name=prod_test, share=2, pmemcache=off, pmemlog=off),             -
                (name=prod_dev, share=1, pmemcache=off, pmemlog=off),              -
                (name=dw_test, share=2, pmemcache=on, pmemlog=off),                -
                (name=other, share=1))
```

You do not need to explicitly set `pmemcache=on` or `pmemlog=on` because they are the default settings.

5.2.3.10 Using PMEM Cache Attributes

Note:

This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see Using XRMEM Cache Attributes.

You can use PMEM cache attributes in the IORM plan to guarantee space allocation in the persistent memory (PMEM) cache.

Using PMEM cache attributes, IORM can reserve space for critical databases while preventing less important or rogue entities from consuming the entire PMEM cache. These attributes can only be specified in an interdatabase plan, and are configured using the CellCLI utility.

You can use the following attributes to set limits for PMEM cache resources. A hard limit means the database cannot exceed its quota even when the memory cache is not full. A soft limit means that the specified limit can be exceeded if there are available resources.

- `pmemCacheMin` — Specifies the minimum amount of PMEM cache space that is guaranteed for the specified database, even if the blocks are cold. This is a hard limit.

  Because `pmemCacheMin` is a guaranteed reservation, the sum of `pmemCacheMin` across all directives should be less than the size of the PMEM cache to ensure that each database gets its respective quota.
- `pmemCacheLimit` — Specifies the soft maximum amount of PMEM cache space that is available to the database. If the PMEM cache is not full, a database can exceed the `pmemCacheLimit` value.
- `pmemCacheSize` — Specifies the hard maximum amount of PMEM cache space that is available to the database. The `pmemCacheSize` value cannot be exceeded at any time.

  However, if you set `pmemCacheSize` to a value that is lower than the current space occupied by the database, then starting with Oracle Exadata System Software release 20.1.0 excess data is proactively cleared from the cache. Previously, excess data was only removed when overwritten by other data.

  `pmemCacheSize` is not a guaranteed reservation if the sum of the `pmemCacheSize` across all directives is more than the size of the PMEM cache. In this case, you can also specify `pmemCacheMin` to define a guaranteed minimum quota.

Example 5-13 Configuring an Interdatabase Plan with PMEM Cache Attributes

This example shows how to create an interdatabase plan with pmem cache attributes. In the example, the `sales` and `test` databases are guaranteed an amount of space in the PMEM cache by using the `pmemCacheSize` parameter. But, the databases cannot exceed the specified allocation, even if the PMEM cache has free space.

The `finc` and `dev` databases use `pmemCacheMin` for guaranteed minimum quotas. These databases can also exceed the specified `pmemCacheLimit` size when there is free space in the PMEM cache.

The example plan also contains various flash cache attributes.

```
ALTER IORMPLAN dbplan=                                                -
((name=sales, share=8, pmemCacheSize= 2G, flashCacheSize=10G), -
(name=finc, share=8, pmemCacheMin= 1G, pmemCacheLimit= 2G,
flashCacheLimit=10G, flashCacheMin=2G), -
(name=dev, share=2, pmemCacheMin= 500M, pmemCacheLimit= 1G,
flashCacheLimit=4G, flashCacheMin=1G), -
(name=test, share=1, limit=10, pmemCacheSize= 200M))
```

5.2.3.11 Using the role Attribute

The `role` attribute allows a different allocation to be specified, based on whether the database has the Oracle Data Guard `primary` or `standby` role.

By default, interdatabase plan directives apply when the database is in either role. If you want the directive to apply only when the database is in the `primary` role, then include role=primary. Similarly, if you want the directive to apply only when the database is in the standby role, then include role=standby.

For example:

```
ALTER IORMPLAN                                                  -
dbPlan=((name=prod, share=8, role=primary),                     -
        (name=prod, share=1, limit=25, role=standby)            -
        (name=default, share=2))
```

Related Topics

- Configuring Resource Manager for Oracle Active Data Guard (My Oracle Support Doc ID 1930540.1)

5.2.3.12 Using the asmcluster Attribute

You can use the asmcluster attribute to uniquely identify databases with the same DB_UNIQUE_NAME.

If you have databases with the same DB_UNIQUE_NAME but associated with different Oracle ASM clusters, then, starting with Oracle Exadata System Software release 19.1.0, you can use the asmcluster attribute to uniquely identify the databases.

The databases must be clients of different Oracle ASM clusters, and you must ASM-scoped security configured to facilitate cluster identification.

For example:

```
ALTER IORMPLAN                                                  -
dbplan=((name=pdb1, share=4, flashcachemin=5G, asmcluster=asm1),  -
        (name=pdb1, share=2, limit=80, asmcluster=asm2),  -
        (name=pdb2, share=2, flashcachelimit=2G, asmcluster=asm1),  -
        name=default, share=1, flashcachelimit=1G))
```

Related Topics

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

5.2.3.13 Resetting Default Values in an IORM Plan

Use an empty string to reset an IORM plan.

You can reset the entire IORM plan, or separately reset the catplan, dbplan, or clusterplan. For example:

```
CellCLI> ALTER IORMPLAN catplan="", dbplan="", clusterplan=""
CellCLI> ALTER IORMPLAN catplan=""
CellCLI> ALTER IORMPLAN dbplan=""
CellCLI> ALTER IORMPLAN clusterplan=""
```

5.2.4 Listing an I/O Resource Management Plan

You can view the current interdatabase plan for a storage server using the CellCLI LIST IORMPLAN command on the storage server.

Example 5-14 Displaying Interdatabase Plan Details

This example shows how to get a detailed list of the interdatabase plan attributes.

```
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=sales_prod, share=8, role=primary
                            name=sales_prod, share=1, limit=50, role=standby
                            name=sales_test, share=1, limit=25
                            name=default, share=2
   objective:               balanced
```

Related Topics

- LIST IORMPLAN

5.2.5 Managing Flash Cache Quotas for Databases and PDBs

IORM enables you to control how you want the flash cache to be shared among different databases and pluggable databases (PDBs).

This can be done using just the CDB resource plan or in conjunction with the I/O Resource Management (IORM) interdatabase plan.

Consider an example intradatabase resource plan for the `NEWCDB` CDB. The plan specifies `memory_min` and `memory_limit` for the 3 PDBs mentioned in the plan.

Note the following:

- The `memory_min` and `memory_limit` values are specified in percentages and range from 0 and 100. Because over-provisioning is supported, the sum of the percentages is not restricted to 100%. If the sum of these values is greater than 100%, then the values are normalized down to a percentage.
- If `memory_min` is not specified, then it defaults to 0.
- If `memory_limit` is not specified, then it defaults to 100.
- For `CDB$ROOT`, there is a 5% `memory_limit` value.

The following code shows how to create the example intradatabase plan. The sum of the `memory_min` values is 40%, and the sum of the `memory_limit` values is 175%, which must be normalized. If an interdatabase plan is not specified, then these percentages apply to the entire size of the flash cache. If an interdatabase plan is specified, then the quotas for the PDBs are computed as a percentage of the `flashcachemin` and `flashcachesize` values for the database as specified in the interdatabase plan directive.

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();

DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN(
    plan    => 'NEWCDB_PLAN',
    comment => 'CDB resource plan for newcdb');

  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'SALESPDB',
    memory_min            => 20);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'SERVICESPDB',
    memory_min            => 20,
    memory_limit          => 50);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'HRPDB',
    memory_limit          => 25);

DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

In the example above, if no interdatabase plan is specified and the size of the flash cache is 50 GB, then the following table shows the breakdown of the quotas after normalization of the limit (because the sum of the `memory_limit` values is greater than 100%). After normalization, if a minimum value is greater than the corresponding limit, then the minimum value is reduced to make it equal to the limit.

Table 5-4 Case 1: PDB Flash Cache Limits with No Interdatabase Plan

PDB	Flash Cache Min	FC Soft Limit	Normalized Soft Limit	FC Hard Limit
SALESPDB	20% = 10 GB	100 (default)	100 / 175 * 50 GB = 28.57 GB	n/a
SERVICESPDB	20% = 10 GB	50	50 / 175 * 50 GB = 14.28 GB	n/a
HRPDB	0	25	25 / 175 * 50 GB = 7.14 GB	n/a

The next example shows an interdatabase plan with flash cache quotas that includes the `NEWCDB` CDB.

```
ALTER IORMPLAN
dbplan=
                -
      ((name=newcdb, share=8,
flashCacheSize=20G),                                        -
       (name=finance, share=8, flashCacheLimit=10G, flashCacheMin=2G), -
       (name=dev, share=2, flashCacheLimit=4G, flashCacheMin=1G),    -
       (name=test, share=1))
```

In addition to the `NEWCDB` CDB, three other databases (`finance`, `dev`, and `test`) share the same storage servers. Flash cache quotas are only enforced if the directives specify the `flashcachesize`, `flashcachelimit`, or `flashcachemin` attributes. The database `test` does not specify any flash cache directive; so that database and its PDBs (if any exist) are not managed for any flash cache quotas.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies `flashcachemin` and `flashcachesize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `flashcachemin` setting, and the PDB-specific `memory_limit` values represent fractions of the `flashcachesize`.

However, if the interdatabase IORM plan directive specifies `flashcachesize` without `flashcachemin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `flashcachesize`.

So, because the example interdatabase IORM plan directive for `newcdb` specifies `flashcachesize` without `flashcachemin`, the PDB-specific `memory_min` quotas in the CDB plan are ignored. The following table lists the effective flash cache limits when applying the example CDB plan together with the example interdatabase IORM plan.

Table 5-5 Case 2: PDB Flash Cache Limits with an InterDatabase Plan

PDB	Flash Cache Min	FC Hard Limit	Normalized Hard Limit	FC Soft Limit
SALESPDB	0	100 (default)	100 / 175 * 20 GB = 11.43 GB	n/a
SERVICESPDB	0	50	50 / 175 * 20 GB = 5.71 GB	n/a
HRPDB	0	25	25 / 175 * 20 GB = 2.86 GB	n/a

For non-CDB databases, the `flashcachesize`, `flashcachemin`, and `flashcachelimit` values are specified in absolute terms and no additional normalization is required. Because `flashcachemin` is a guaranteed reservation, the sum of `flashcachemin` across all the directives should be less than the total size of the flash cache.

5.2.6 Managing XRMEM Cache Quotas for Databases and PDBs

I/O Resource Management (IORM) enables you to control how you want the Exadata RDMA Memory Cache (XRMEM cache) to be shared among different databases and pluggable databases (PDBs).

This can be done using just the CDB resource plan or in conjunction with the I/O Resource Management (IORM) interdatabase plan.

Consider an example intradatabase resource plan for the `NEWCDB` CDB. The plan specifies `memory_min` and `memory_limit` for the 3 PDBs mentioned in the plan.

Note the following:

- The `memory_min` and `memory_limit` values are specified in percentages and range from 0 and 100. Because over-provisioning is supported, the sum of the percentages is not restricted to 100%. If the sum of these values is greater than 100%, then the values are normalized down to a percentage.
- If `memory_min` is not specified, then it defaults to 0.
- If `memory_limit` is not specified, then it defaults to 100.

- For CDB$ROOT, there is a 5% memory_limit value.

The following code shows how to create the example intradatabase plan. The sum of the memory_min values is 40%, and the sum of the memory_limit values is 175%, which must be normalized. If an interdatabase plan is not specified, then these percentages apply to the entire size of the XRMEM cache. If an interdatabase plan is specified, then the quotas for the PDBs are computed as a percentage of the xrmemcachemin and xrmemcachesize values for the database as specified in the interdatabase plan directive.

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();

DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN(
    plan    => 'NEWCDB_PLAN',
    comment => 'CDB resource plan for newcdb');

  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => 'NEWCDB_PLAN',
    pluggable_database     => 'SALESPDB',
    memory_min             => 20);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => 'NEWCDB_PLAN',
    pluggable_database     => 'SERVICESPDB',
    memory_min             => 20,
    memory_limit           => 50);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                   => 'NEWCDB_PLAN',
    pluggable_database     => 'HRPDB',
    memory_limit           => 25);

DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

In the example above, if no interdatabase plan is specified and the size of the XRMEM cache is 1.25 TB, then the following table shows the breakdown of the quotas after normalization of the limit (because the sum of the memory_limit values is greater than 100%). After normalization, if a minimum value is greater than the corresponding limit, then the minimum value is reduced to make it equal to the limit.

Table 5-6 Case 1: PDB XRMEM Cache Limits with No Interdatabase Plan

PDB	XRMEM Cache Min	XRMEM Soft Limit	Normalized Soft Limit	XRMEM Hard Limit
SALESPDB	20% = 256 GB	100 (default)	100 / 175 * 1.25 TB = 731 GB	n/a
SERVICESPDB	20% = 256 GB	50	50 / 175 * 1.25 TB = 366 GB	n/a
HRPDB	0	25	25 / 175 * 1.25 TB = 183 GB	n/a

The next example shows an interdatabase plan with XRMEM cache quotas that includes the NEWCDB CDB.

```
ALTER IORMPLAN
dbplan=
                 -
      ((name=newcdb, share=8, xrmemCacheSize= 200G,
flashCacheSize=10G),                                    -
       (name=finance, share=8, xrmemCacheMin= 100G, xrmemCacheLimit= 200G,
flashCacheLimit=10G, flashCacheMin=2G), -
       (name=dev, share=2, xrmemCacheMin= 1G, xrmemCacheLimit= 10G,
flashCacheLimit=4G, flashCacheMin=1G),    -
       (name=test, share=1))
```

In addition to the NEWCDB CDB, three other databases (finance, dev, and test) share the same storage servers. XRMEM cache quotas are only enforced if the directives specify the xrmemcachesize, xrmemcachelimit, or xrmemcachemin attributes. The database test does not specify any XRMEM cache directive; so that database and its PDBs (if any exist) are not managed for any XRMEM cache quotas.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies xrmemcachemin and xrmemcachesize settings for a database, then the PDB-specific memory_min quotas in the CDB plan represent fractions of the xrmemcachemin setting, and the PDB-specific memory_limit values represent fractions of the xrmemcachesize.

However, if the interdatabase IORM plan directive specifies xrmemcachesize without xrmemcachemin, then the PDB-specific memory_min settings are ignored while the memory_limit settings continue to represent fractions of the xrmemcachesize.

So, because the example interdatabase IORM plan directive for newcdb specifies xrmemcachesize without xrmemcachemin, the PDB-specific memory_min quotas in the CDB plan are ignored. The following table lists the effective XRMEM cache limits when applying the example CDB plan together with the example interdatabase IORM plan.

Table 5-7 Case 2: PDB XRMEM Cache Limits with an InterDatabase Plan

PDB	XRMEM Cache Min	XRMEM Hard Limit	Normalized Hard Limit	XRMEM Soft Limit
SALESPDB	0	100 (default)	100 / 175 * 200 GB = 114.29 GB	n/a
SERVICESPDB	0	50	50 / 175 * 200 GB = 57.14 GB	n/a
HRPDB	0	25	25 / 175 * 200 GB = 28.57 GB	n/a

For non-CDB databases, the xrmemcachesize, xrmemcachemin, and xrmemcachelimit values are specified in absolute terms and no additional normalization is required. Because xrmemcachemin is a guaranteed reservation, the sum of xrmemcachemin across all the directives should be less than the total size of the XRMEM cache.

5.2.7 Managing PMEM Cache Quotas for Databases and PDBs

Note:

This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see Managing XRMEM Cache Quotas for Databases and PDBs.

I/O Resource Management (IORM) enables you to control how you want the PMEM cache to be shared among different databases and pluggable databases (PDBs).

This can be done using just the CDB resource plan or in conjunction with the I/O Resource Management (IORM) interdatabase plan.

Consider an example intradatabase resource plan for the `NEWCDB` CDB. The plan specifies `memory_min` and `memory_limit` for the 3 PDBs mentioned in the plan.

Note the following:

- The `memory_min` and `memory_limit` values are specified in percentages and range from 0 and 100. Because over-provisioning is supported, the sum of the percentages is not restricted to 100%. If the sum of these values is greater than 100%, then the values are normalized down to a percentage.
- If `memory_min` is not specified, then it defaults to 0.
- If `memory_limit` is not specified, then it defaults to 100.
- For `CDB$ROOT`, there is a 5% `memory_limit` value.

The following code shows how to create the example intradatabase plan. The sum of the `memory_min` values is 40%, and the sum of the `memory_limit` values is 175%, which must be normalized. If an interdatabase plan is not specified, then these percentages apply to the entire size of the PMEM cache. If an interdatabase plan is specified, then the quotas for the PDBs are computed as a percentage of the `pmemcachemin` and `pmemcachesize` values for the database as specified in the interdatabase plan directive.

```
BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();

DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN(
    plan    => 'NEWCDB_PLAN',
    comment => 'CDB resource plan for newcdb');

  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'SALESPDB',
    memory_min            => 20);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'SERVICESPDB',
    memory_min            => 20,
    memory_limit          => 50);
  DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan                  => 'NEWCDB_PLAN',
    pluggable_database    => 'HRPDB',
    memory_limit          => 25);

DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
END;
/
```

In the example above, if no interdatabase plan is specified and the size of the PMEM cache is 10 GB, then the following table shows the breakdown of the quotas after normalization of the limit (because the sum of the `memory_limit` values is greater than 100%). After normalization, if a minimum value is greater than the corresponding limit, then the minimum value is reduced to make it equal to the limit.

Table 5-8   Case 1: PDB PMEM Cache Limits with No Interdatabase Plan

PDB	PMEM Cache Min	PMEM Soft Limit	Normalized Soft Limit	PMEM Hard Limit
SALESPDB	20% = 2 GB	100 (default)	100 / 175 * 10 GB = 5.71 GB	n/a
SERVICESPDB	20% = 2 GB	50	50 / 175 * 10 GB = 2.85 GB	n/a
HRPDB	0	25	25 / 175 * 10 GB = 1.42 GB	n/a

The next example shows an interdatabase plan with PMEM cache quotas that includes the `NEWCDB` CDB.

```
ALTER IORMPLAN
dbplan=
                 -
      ((name=newcdb, share=8, pmemCacheSize= 2G,
flashCacheSize=10G),                                       -
       (name=finance, share=8, pmemCacheMin= 1G, pmemCacheLimit= 2G,
flashCacheLimit=10G, flashCacheMin=2G), -
       (name=dev, share=2, pmemCacheMin= 100M, pmemCacheLimit= 1G,
flashCacheLimit=4G, flashCacheMin=1G),    -
       (name=test, share=1))
```

In addition to the `NEWCDB` CDB, three other databases (`finance`, `dev`, and `test`) share the same storage servers. PMEM cache quotas are only enforced if the directives specify the `pmemcachesize`, `pmemcachelimit`, or `pmemcachemin` attributes. The database `test` does not specify any PMEM cache directive; so that database and its PDBs (if any exist) are not managed for any PMEM cache quotas.

Cache limits in an interdatabase IORM plan directive constrain the settings in any corresponding CDB plan. Consequently, if an interdatabase IORM plan directive specifies `pmemcachemin` and `pmemcachesize` settings for a database, then the PDB-specific `memory_min` quotas in the CDB plan represent fractions of the `pmemcachemin` setting, and the PDB-specific `memory_limit` values represent fractions of the `pmemcachesize`.

However, if the interdatabase IORM plan directive specifies `pmemcachesize` without `pmemcachemin`, then the PDB-specific `memory_min` settings are ignored while the `memory_limit` settings continue to represent fractions of the `pmemcachesize`.

So, because the example interdatabase IORM plan directive for `newcdb` specifies `pmemcachesize` without `pmemcachemin`, the PDB-specific `memory_min` quotas in the CDB plan are ignored. The following table lists the effective PMEM cache limits when applying the example CDB plan together with the example interdatabase IORM plan.

Table 5-9 Case 2: PDB PMEM Cache Limits with an InterDatabase Plan

PDB	PMEM Cache Min	PMEM Hard Limit	Normalized Hard Limit	PMEM Soft Limit
SALESPDB	0	100 (default)	100 / 175 * 2 GB = 1.14 GB	n/a
SERVICESPDB	0	50	50 / 175 * 2 GB = 0.57 GB	n/a
HRPDB	0	25	25 / 175 * 2 GB = 0.28 GB	n/a

For non-CDB databases, the `pmemcachesize`, `pmemcachemin`, and `pmemcachelimit` values are specified in absolute terms and no additional normalization is required. Because `pmemcachemin` is a guaranteed reservation, the sum of `pmemcachemin` across all the directives should be less than the total size of the PMEM cache.

5.2.8 Using IORM Profiles

I/O Resource Management (IORM) interdatabase plans support profiles to ease management, and configuration of interdatabase plans for hundreds of databases.

Profiles introduce a way to allocate I/O resources for groups of databases. Profiles are specified as directives for the interdatabase plan, and are configured using the CellCLI utility. A profile directive consists of an identifier (name), and a set of attributes. To differentiate between a database directive and a profile directive, a qualifier attribute called `type` is used. The `type` attribute can be set to either `database` or `profile`. The following is an example of the `type` attribute syntax:

```
CellCLI> ALTER IORMPLAN DBPLAN=((name=gold, share=10, type=profile),  -
                                (name=silver, share=5, type=profile), -
                                (name=bronze, share=1, type=profile))
```

The preceding example contains three directives for profiles, `GOLD`, `SILVER` and `BRONZE`. All databases with `db_performance_profile` set to `GOLD` would automatically get 10 shares of cell I/O resources. Likewise, in the example above, databases with the `SILVER` profile would get 5 shares, and databases with the `BRONZE` profile would get 1 share.

After your create your profiles, you then map new and existing databases to one of the profiles defined in the interdatabase plan. This is done by setting the `db_performance_profile` initialization parameter for each database to the name of the desired profile. You must then restart the database. As with Oracle Database Resource Manager plans, the IORM profile information is automatically pushed to all the storage servers (cells). The following SQL command displays how the initialization parameter can be set for a database:

```
SQL> ALTER SYSTEM SET db_performance_profile=gold SCOPE=spfile;

SQL> SHUTDOWN IMMEDIATE

SQL> STARTUP
```

When adding a new database, you can set the `db_performance_profile` parameter and restart the database. The database automatically inherits the profile attributes without having to modify the interdatabase plan. You can also create interdatabase plans with a mix of profile directives, and database directives.

You can view existing profiles using the `LIST IORMPROFILE` command.

Note the following when managing interdatabase profile plans:

- The `db_performance_profile` parameter is not a dynamic parameter, so profile updates require a database restart.
- If the `type` attribute is not specified, then the directive defaults to the `database` directive.
- An interdatabase plan can specify only 8 profile directives, and 1024 database directives.
- Level, allocation, and role cannot be specified with a profile directive.
- The words `OTHER`, and `DEFAULT` are reserved words. A profile name cannot be `OTHER` or `DEFAULT`.
- The `type` attribute cannot be specified with category plans.
- Profiles cannot be specified in conjunction with category plans.
- If multiple databases are mapped to the `OTHER` directive, then Oracle Exadata Storage Server does not use Oracle Database Resource Manager for these databases. All of the I/O requests are treated the same.

Related Topics

- ALTER IORMPLAN

5.2.9 Verifying the Configuration of I/O Resource Management

Use this checklist to verify that I/O Resource Management (IORM) is configured correctly.

- Verify that the following criteria are 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.
    - If Oracle Database Resource Manager is enabled using Scheduler Window, then the same plan is always enabled on all database instances.
    - If Oracle 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');
```

- 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 the `dbPlan` parameter.
- Verify that sessions are mapped to the correct consumer group.

  Run the following query 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;
    ```

    Use the following command to grant permission for any session to switch into the consumer group. This example shows to how grant the permission for `BATCH_GROUP`.

    ```
    EXEC dbms_resource_manager_privs.grant_switch_consumer_group -
      ('public', 'BATCH_GROUP', FALSE);
    ```
  - **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.

- Use CellCLI to list 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 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;
  ```

- 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.

Related Topics

- Monitoring IORM Utilization
  You can use metrics to monitor IORM utilization.

6
Monitoring Exadata

6.1 Introducing Exadata Monitoring Tools and Information Sources

You can monitor Exadata system performance by using various tools and information sources. The following topics introduce and compare the main Exadata performance monitoring tools and information sources:

- Automatic Workload Repository (AWR)
  AWR is the integrated performance diagnostic tool built into Oracle Database. AWR includes Exadata storage server configuration, health, and statistics information.
- Database Statistics and Wait Events
  Oracle Database provides performance information in a series of dynamic performance views (also known as `V$` views).
- Exadata Metrics
  Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.
- Exadata Alerts
  Alerts draw attention to potential and actual problems and other events of interest to an administrator.
- Real-Time Insight
  You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems.
- ExaWatcher
  ExaWatcher collects very detailed system statistics, primarily focused on the operating system and network interfaces.

6.1.1 Automatic Workload Repository (AWR)

AWR is the integrated performance diagnostic tool built into Oracle Database. AWR includes Exadata storage server configuration, health, and statistics information.

AWR collects information from various database dynamic performance views at regular intervals. Each data collection is an AWR snapshot. An AWR report shows statistics captured between two AWR snapshots.

AWR collects and reports a wealth of statistical data related to Oracle Database, including database statistics and wait events from `V$SYSSTAT`, `V$SYSTEM_EVENT`, `V$SEGMENT_STATISTICS`, `V$SQL`.

Commencing with Oracle Database release 12.1.0.2, AWR also collects performance statistics from Exadata storage servers and includes them in the HTML version of the AWR report. Consequently, AWR provides comprehensive and unified performance information that includes Oracle Database statistics and Exadata statistics, storage metrics, and alerts from all storage servers.

Note:

The Exadata sections are only available in the HTML version of the AWR report.

The AWR report contains the following Exadata-specific sections:

- Exadata Server Configuration
- Exadata Server Health Report
- Exadata Statistics

Key benefits include:

- AWR provides comprehensive and unified performance information that includes Oracle Database metrics and Exadata storage metrics and alerts from all storage servers.
- AWR enables simple outlier detection and analysis.
- The granularity of AWR information is customizable and based on the AWR collection interval, which is hourly by default.

It is important to note that the Exadata storage server statistics are collected and maintained by the storage servers, so the information is not restricted to a specific database or database instance. Consequently, the storage related statistics in AWR includes the I/O for all databases running on the storage servers, while the database statistics relate to the specific instance or database hosting AWR.

When using a container database (CDB), the Exadata-specific sections of the AWR report are only available in the root container.

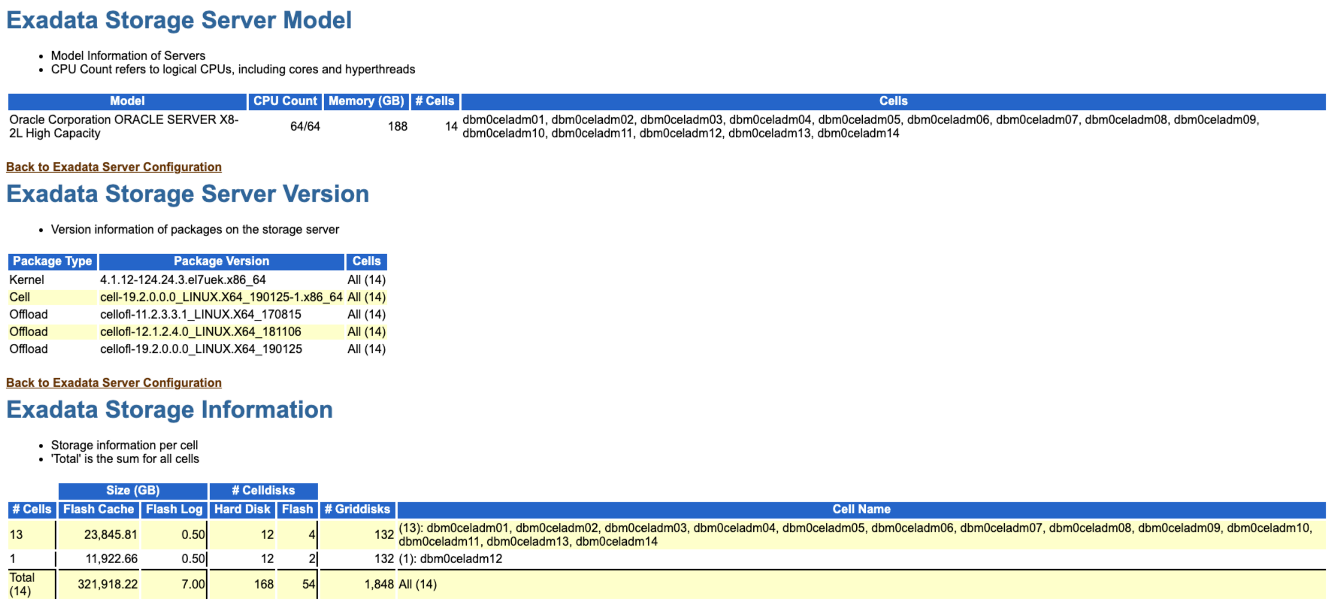
As part of Oracle Database, AWR is constantly being enhanced with new statistics and analysis. The availability of a specific section or statistic is subject to the version of Oracle Database being used.

Exadata Server Configuration

The Exadata Server Configuration section of the AWR report shows configuration information from all of the associated storage servers, including model information, software versions, and storage information relating to cell disks, grid disks, ASM disk groups, and the IORM objective. By examining this information, you can easily see any configuration differences across the storage servers that could potentially affect the behavior or performance of the system.

The following example shows part of the Exadata Server Configuration section. The example is based on a system with 14 X8-2 High Capacity storage servers. In the example, all of the servers have the same software version. However, the storage information shows that one cell contains 2 flash cell disks, while the others all have 4 flash cell disks.

Figure 6-1   AWR Report: Exadata Server Configuration



Exadata Storage Server Model

- Model Information of Servers
- CPU Count refers to logical CPUs, including cores and hyperthreads

Model	CPU Count	Memory (GB)	# Cells	Cells
Oracle Corporation ORACLE SERVER X8-2L High Capacity	64/64	188	14	dbm0celadm01, dbm0celadm02, dbm0celadm03, dbm0celadm04, dbm0celadm05, dbm0celadm06, dbm0celadm07, dbm0celadm08, dbm0celadm09, dbm0celadm10, dbm0celadm11, dbm0celadm12, dbm0celadm13, dbm0celadm14

Back to Exadata Server Configuration

Exadata Storage Server Version

- Version information of packages on the storage server

Package Type	Package Version	Cells
Kernel	4.1.12-124.24.3.el7uek.x86_64	All (14)
Cell	cell-19.2.0.0.0_LINUX.X64_190125-1.x86_64	All (14)
Offload	cellofl-11.2.3.3.1_LINUX.X64_170815	All (14)
Offload	cellofl-12.1.2.4.0_LINUX.X64_181106	All (14)
Offload	cellofl-19.2.0.0.0_LINUX.X64_190125	All (14)

Back to Exadata Server Configuration

Exadata Storage Information

- Storage information per cell
- 'Total' is the sum for all cells

	Size (GB)		# Celldisks			
# Cells	Flash Cache	Flash Log	Hard Disk	Flash	# Griddisks	Cell Name
13	23,845.81	0.50	12	4	132	(13): dbm0celadm01, dbm0celadm02, dbm0celadm03, dbm0celadm04, dbm0celadm05, dbm0celadm06, dbm0celadm07, dbm0celadm08, dbm0celadm09, dbm0celadm10, dbm0celadm11, dbm0celadm13, dbm0celadm14
1	11,922.66	0.50	12	2	132	(1): dbm0celadm12
Total (14)	321,918.22	7.00	168	54	1,848	All (14)

Exadata Server Health Report

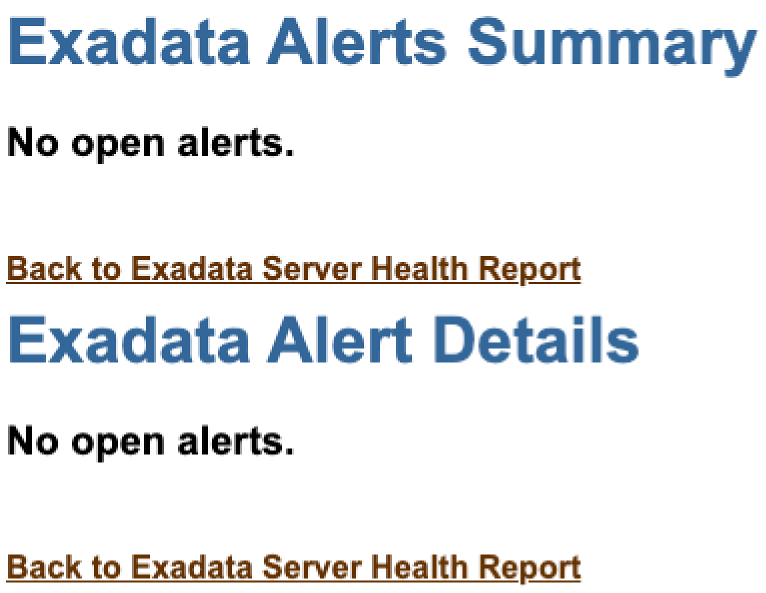
The Exadata Server Health Report section of the AWR report shows information about offline disks and open storage server alerts.

When a disk is offline, then the total I/O bandwidth is reduced, which may affect I/O performance.

Alerts represent events of importance on a storage server that could affect its functionality and performance. Open alerts are alerts that have not been marked as examined by an administrator. For more details, see Exadata Alerts.

Ideally, there should be no open alerts in the AWR report, as shown in the following example:

Figure 6-2   AWR Report: No Open Alerts



Exadata Alerts Summary

No open alerts.

Back to Exadata Server Health Report

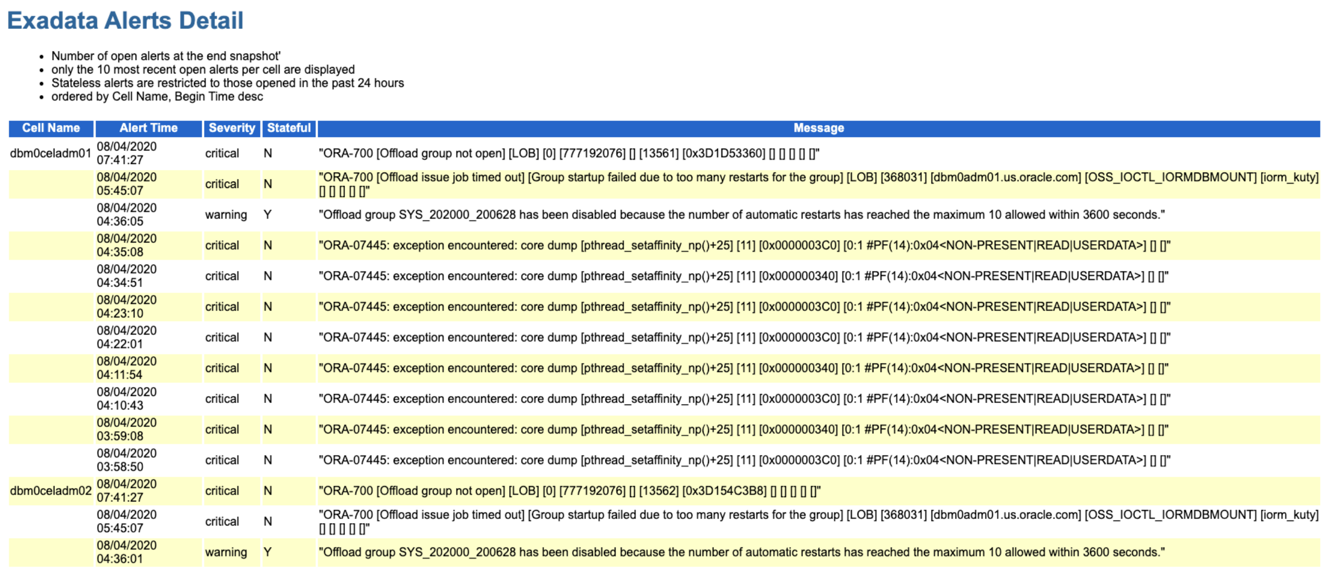
Exadata Alert Details

No open alerts.

Back to Exadata Server Health Report

The following example shows Exadata alert details indicating that the offload server was unable to start, which would affect smart scan performance on the system.

Figure 6-3 AWR Report: Exadata Alert Details



Exadata Alerts Detail

- Number of open alerts at the end snapshot'
- only the 10 most recent open alerts per cell are displayed
- Stateless alerts are restricted to those opened in the past 24 hours
- ordered by Cell Name, Begin Time desc

Cell Name	Alert Time	Severity	Stateful	Message
dbm0celadm01	08/04/2020 07:41:27	critical	N	"ORA-700 [Offload group not open] [LOB] [0] [777192076] [] [13561] [0x3D1D53360] [] [] [] [] []"
	08/04/2020 05:45:07	critical	N	"ORA-700 [Offload issue job timed out] [Group startup failed due to too many restarts for the group] [LOB] [368031] [dbm0adm01.us.oracle.com] [OSS_IOCTL_IORMDBMOUNT] [iorm_kuty] [] [] [] [] []"
	08/04/2020 04:36:05	warning	Y	"Offload group SYS_202000_200628 has been disabled because the number of automatic restarts has reached the maximum 10 allowed within 3600 seconds."
	08/04/2020 04:35:08	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x0000003C0] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 04:34:51	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x000000340] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 04:23:10	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x0000003C0] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 04:22:01	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x0000003C0] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 04:11:54	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x000000340] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 04:10:43	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x0000003C0] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 03:59:08	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x000000340] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
	08/04/2020 03:58:50	critical	N	"ORA-07445: exception encountered: core dump [pthread_setaffinity_np()+25] [11] [0x0000003C0] [0:1 #PF(14):0x04<NON-PRESENT|READ|USERDATA>] [] []"
dbm0celadm02	08/04/2020 07:41:27	critical	N	"ORA-700 [Offload group not open] [LOB] [0] [777192076] [] [13562] [0x3D154C3B8] [] [] [] [] []"
	08/04/2020 05:45:07	critical	N	"ORA-700 [Offload issue job timed out] [Group startup failed due to too many restarts for the group] [LOB] [368031] [dbm0adm01.us.oracle.com] [OSS_IOCTL_IORMDBMOUNT] [iorm_kuty] [] [] [] [] []"
	08/04/2020 04:36:01	warning	Y	"Offload group SYS_202000_200628 has been disabled because the number of automatic restarts has reached the maximum 10 allowed within 3600 seconds."

Exadata Statistics

AWR includes statistics from various Exadata storage server components, such as Smart I/O, Smart Flash Cache, Smart Flash Log, XRMEM cache, XRMEM log, and IO Resource Manager. It includes I/O statistics from both the operating system and the cell server software, and performs simple outlier analysis on these I/O statistics. Because Exadata typically distributes I/O operations evenly across all cells and all disks, outliers may indicate a potential problem and warrant further investigation. If any cell or disk is behaving differently or performing more work, this could potentially cause slower performance for the whole system.

The Exadata Statistics section of the AWR report includes the following sub-sections:

- Performance Summary
- Exadata Resource Statistics
  - Exadata Outlier Summary
  - Exadata OS Statistics Outliers
  - Exadata Cell Server Statistics Outliers
  - Exadata Outlier Details
  - Exadata OS Statistics Top
  - Exadata Cell Server Statistics Top
- Exadata Smart Statistics
  - Smart IO
  - Flash Log
  - XRMEM Log
  - Flash Cache

  - XRMEM Cache
- Exadata IO Reasons
  - Top IO Reasons by Requests
  - Top IO Reasons by MB
  - Internal IO Reasons
- Exadata Top Database Consumers
  - Top Databases by Requests
  - Top Databases by Requests - Details
  - Top Databases by Throughput
  - Top Databases by Requests per Cell
  - Top Databases by Requests per Cell - Details
  - Top Databases by Throughput per Cell
- Exadata IO Latency Capping
  - Cancelled IOs - Client
  - Cancelled IOs - Cells
- Exadata Flash Wear

Key Exadata statistics are discussed in detail in the corresponding topics under Monitoring Oracle Exadata System Software Components.

Related Topics

- Automatic Workload Repository
- About Automatic Workload Repository Compare Periods Reports

6.1.2 Database Statistics and Wait Events

Oracle Database provides performance information in a series of dynamic performance views (also known as `V$` views).

The dynamic performance views contain a wealth of database statistics and wait events. By using dynamic performance views, a database administrator can perform detailed and customized monitoring on individual SQL operations and database sessions. The dynamic performance views are also a major information source for Automatic Workload Repository (AWR) and other Oracle Database monitoring tools, such as SQL monitor.

The following sections introduce the main dynamic performance views that include Exadata-specific information.

- `V$SYSSTAT`
- `V$SESSION`
- `V$SESSION_EVENT`
- `V$SYSTEM_EVENT`
- `V$ACTIVE_SESSION_HISTORY`
- `V$SQL`
- `V$SEGMENT_STATISTICS`

V$SYSSTAT

V$SYSSTAT contains various system-wide statistics relating to the current database instance, including I/O statistics related to Exadata storage server.

For example, you can use the following query to display I/O statistics related to Exadata storage server:

```
SQL> SELECT name, value
     FROM v$sysstat
     WHERE name like '%cell%' or name like 'physical%'
     ORDER BY name;
```

These statistics are also in the Global Activity Statistics or Instance Activity Statistics section of the AWR report.

V$SESSION

V$SESSION contains various statistics and wait events for currently active database sessions.

For example, you can use the following query to display a summary that shows the number of sessions that are currently waiting on cell events:

```
SQL> SELECT event, count(*)
     FROM v$session
     WHERE event like 'cell%'
       AND wait_time = 0  -- session currently waiting
     GROUP BY event
     ORDER BY count(*) desc;
```

The following query displays event details for sessions that are currently waiting on cell events:

```
SQL> SELECT event, p1, p2, p3
     FROM v$session
     WHERE event like 'cell%'
       AND wait_time = 0;
```

For cell wait events in V$SESSION:

- The P1 column identifies the cell hash number. You can use the V$CELL view to identify the cell from the cell hash number.
- The P2 column identifies the disk hash number. You can use the V$ASM_DISK view to identify the disk name from the disk hash number.

V$SESSION_EVENT

V$SESSION_EVENT displays information on waits for an event by a session.

For example, you can use the following query to display a summary of the wait events for the current session. By examining the output from this query, you can compare the time spent waiting for Exadata storage server events with the time spent waiting for other events in the database.

```
SQL> SELECT event, total_waits, time_waited_micro, time_waited_micro/
decode(total_waits, 0, null, total_waits) avg_wait_time
     FROM v$session_event
     WHERE wait_class != 'Idle'
       AND sid = userenv('SID')
     ORDER BY time_waited_micro desc;
```

V$SYSTEM_EVENT

`V$SYSTEM_EVENT` displays system-wide information on total waits for an event.

You can use the following query to display a summary of the wait events since start-up of the current database instance:

```
SQL> SELECT event, total_waits, time_waited_micro, time_waited_micro/
decode(total_waits, 0, null, total_waits) avg_wait_time
     FROM v$system_event
     WHERE wait_class != 'Idle'
     ORDER by time_waited_micro desc;
```

Information from `V$SYSTEM_EVENT` is contained in the Wait Events Statistics section of the AWR report.

V$ACTIVE_SESSION_HISTORY

`V$ACTIVE_SESSION_HISTORY` keeps a history of the active sessions on the system, where active is defined as using CPU or waiting on a non-idle wait event. For each session, `V$ACTIVE_SESSION_HISTORY` includes a wealth of information on what the session was doing, such as SQL execution plans and wait event details, including information about Exadata storage interactions.

For example, you can use the following query to display cell block reads over the past 5 minutes with latency in excess of 10ms:

```
SQL> SELECT sample_time, p1, p2, time_waited
     FROM v$active_session_history
     WHERE event = 'cell single block physical read'
       AND wait_time = 0  -- currently waiting
       AND sample_time > sysdate - 5/1440  -- past 5 minutes
       AND time_waited > 10000
     ORDER by time_waited desc;
```

The Active Session History (ASH) Report section of the AWR report is based on `V$ACTIVE_SESSION_HISTORY`. Oracle Database also provides a more detailed Active Session History (ASH) report.

V$SQL

`V$SQL`, and related views like `V$SQLAREA`, `V$SQLAREA_PLAN_HASH`, `V$SQLSTATS`, and `V$SQLSTATS_PLAN_HASH`, contain information and statistics for SQL statements processed on the database instance.

V$SQL, and related views, include the following columns that contain information about Exadata storage interactions:

- PHYSICAL_READ_BYTES — number of bytes read by the SQL
- PHYSICAL_READ_REQUESTS — number of read requests issued by the SQL
- PHYSICAL_WRITE_BYTES — number of bytes written by the SQL
- PHYSICAL_WRITE_REQUESTS — number of write requests issued by the SQL
- IO_CELL_OFFLOAD_ELIGIBLE_BYTES — number of bytes eligible for predicate offload to Exadata storage server
- IO_CELL_OFFLOAD_RETURNED_BYTES — number of bytes returned by smart scans
- IO_INTERCONNECT_BYTES — number of bytes exchanged between the database and the storage servers
- IO_CELL_UNCOMPRESSED_BYTES — number of uncompressed bytes read, where uncompressed bytes is the size after decompression
- OPTIMIZED_PHY_READ_REQUESTS — number of read requests satisfied from Exadata Smart Flash Cache, or read requests avoided due to storage index or columnar cache

The following example shows a query that displays Exadata offload processing performance data. The output focuses on a query that scans the SALES table. The output shows that all of the data in the SALES table (approximately 5 GB) was eligible for offload processing. In this case, because of offload processing, even though the query performed 5385.75 MB of I/O, only 417.65 MB of data was delivered over the network to the database host.

```
SQL> SELECT sql_text,
       io_cell_offload_eligible_bytes/1024/1024 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 UPPER(sql_text) LIKE '%FROM SALES%';

SQL_TEXT                      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
```

Information from V$SQL is displayed in the SQL Statistics section of the AWR report.

V$SEGMENT_STATISTICS

V$SEGMENT_STATISTICS contains 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 Exadata storage.

The optimized physical read segment statistic records the number of read requests for objects that are read from Exadata Smart Flash Cache or reads that are avoided through the use of storage index or columnar cache. The optimized physical writes statistic records the number of write requests for an object that first went to Exadata Smart Flash Cache. Such write requests can be synchronized to the disk later in a lazy manner to free up cache space.

The following example query displays objects that are associated with more than 1000 optimized reads from Exadata storage. A similar query can be used to determine the objects that are associated with very few optimized reads.

```
SQL> SELECT object_name, value
     FROM v$segment_statistics
     WHERE statistic_name='optimized physical reads'
       AND value>1000
     ORDER BY value;
```

Information from `V$SEGMENT_STATISTICS` is displayed in the Segment Statistics section of the AWR report.

6.1.3 Exadata Metrics

Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.

Exadata metrics contain detailed statistics for most Exadata components. Most metrics relate to the storage server and its components, such as flash cache, cell disks, and grid disks. Storage server metrics enable detailed monitoring of Exadata storage server performance.

Metrics are of the following types:

- Cumulative metrics are statistics that accumulate over time since the metric was created or the server was restarted.

  For example, `CL_IO_RQ_NODATA` is the total number of I/O requests that didn't return data.

- Instantaneous metrics contain the current value at the time of the metric observation.

  For example, `CL_MEMUT` is the percentage of total physical memory currently used on the server.

- Rate metrics are computed statistics where the value is observed over time.

  For example, `N_NIC_KB_TRANS_SEC` is the number of kilobytes per second transmitted over the server's Ethernet interfaces.

Some metrics differentiate between small I/O and large I/O. For such metrics, small I/O means I/O that is less than or equal to 128 KB in size. Large I/O is greater than 128 KB in size.

By default, metric collections occur at 1-minute intervals. Metric observations in the default collection are initially recorded in memory and later written to a disk-based repository. For most metrics, historical observations are maintained for seven days by default. However, a subset of key metrics are retained for up to one year. Starting with Oracle Exadata System Software 24.1.0, you can view and control which metrics to maintain for up to one year. In all cases, Exadata automatically purges historical metric observations if the storage server detects a shortage of space for the disk-based metric history repository.

Commencing with Oracle Exadata System Software 22.1.0, you can optionally configure fine-grained metrics. To enable fine-grained metrics, you must specify a collection interval between 1 and 60 seconds. You can also choose the metrics to include in the fine-grained collection. Fine-grained metric collection is the foundation for real-time metric streaming. Consequently, fine-grained metrics are only recorded in memory to support efficient streaming to an external metric collection and visualization platform.

You can use the CellCLI `LIST` command as the primary means to display Exadata metric definitions and observations. See Using Metrics. However, because Exadata metrics are managed within each Exadata server, metric observations must be collected and correlated by some additional means to gain a system-wide view.

Related Topics

- Using Metrics
  You can use CellCLI commands to display, monitor, and manage storage server metrics.

6.1.4 Exadata Alerts

Alerts draw attention to potential and actual problems and other events of interest to an administrator.

There are three types of alerts; informational alerts, warning alerts, and critical alerts.

Metrics observations are automatically compared with stored thresholds. When a metric crosses a specified threshold, an alert is automatically generated. Metrics can be associated with warning and critical thresholds. When a metric value crosses a warning threshold, then a warning alert is generated. When a metric crosses a critical threshold, then a critical alert is generated.

Exadata administrators can specify the warning and critical threshold for most metrics. There are also some preset thresholds. For example, there are built-in thresholds for ambient temperature. If the temperature is too low or too high, an alert is automatically generated.

Some system conditions and state changes can generate informational alerts; for example, a server chassis is opened. More serious system errors and state changes can also generate warning or critical alerts; for example, a hard disk device is reporting disk errors.

This system of thresholds and alerts enables administrators to focus on the most important events, while maintaining awareness other interesting events.

You can use CellCLI commands to manage and display Exadata alerts and thresholds. You can also configure alert notification using email and SNMP. Alert information is also included in the AWR report.

Like Exadata metrics, alerts are managed separately on each server.

Related Topics

- Monitoring Alerts
  You can monitor and receive notifications for alerts.

6.1.5 Real-Time Insight

You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems.

Commencing with Oracle Exadata System Software 22.1.0, Real-Time Insight provides infrastructure to:

- Categorize specific metrics as fine-grained, and enable the collection of fine-grained metrics as often as every second.
- Stream metric observations to user-defined locations in real-time, using either push (upload) or pull (download) transmission models.

Related Topics

- Using Real-Time Insight
  You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems using an external metric collection platform.

6.1.6 ExaWatcher

ExaWatcher collects very detailed system statistics, primarily focused on the operating system and network interfaces.

ExaWatcher runs separately on each server and collects server and network-related data using operating system tools and utilities, such as `iostat`, `vmstat`, `mpstat`, `top`. It also collects storage server statistics using `cellsrvstat`.

The information is very granular, with most operating system statistics collected at 5-second intervals. To extract statistics from ExaWatcher for analysis, you can use the `GetExaWatcherResults.sh` script. The output from `GetExaWatcherResults.sh` also include charts for a small subset of key statistics.

ExaWatcher is typically used for detailed debugging. Consequently, it is recommended to keep baselines of ExaWatcher data for comparison purposes.

Related Topics

- System Diagnostics Data Gathering with sosreports and Oracle ExaWatcher
  You can use the sosreport utility and Oracle ExaWatcher to diagnose problems with your system.
- About ExaWatcher Charts

6.2 Guidelines for Exadata Monitoring

Regardless of the information source, it is considered good practice to maintain baseline collections of the various performance data sources covering the important periods of your workload, such as peak periods, month-end or year-end processing, or before patching or upgrades.

Baseline information enables a comparison with a known 'normal' state when a performance regression occurs. Such comparison helps to identify changes and guide further analysis.

Baseline data should include AWR data (using AWR exports or AWR baselines), Exadata metrics, and ExaWatcher collections. For data warehousing workloads that rely on smart scans, keeping a collection of SQL monitor reports for key SQL statements is also highly recommended.

6.3 Monitoring Oracle Exadata System Software Components

- Monitoring Exadata Smart Flash Cache
- Monitoring XRMEM Cache
- Monitoring PMEM Cache
- Monitoring Exadata Smart Flash Log
- Monitoring XRMEM Log
- Monitoring PMEM Log

- Monitoring Smart I/O
- Monitoring I/O Resource Management (IORM)
- Monitoring Cell Disk I/O
- Monitoring Grid Disks
- Monitoring Host Interconnect Metrics
  Host interconnect metrics provide information about the I/O transmission for hosts that access cell storage.

6.3.1 Monitoring Exadata Smart Flash Cache

Exadata Smart Flash Cache holds frequently accessed data in flash storage, while most data is kept in very cost-effective disk storage. Caching occurs automatically and requires no user or administrator effort. Exadata Smart Flash Cache can intelligently determine the data that is most useful to cache based on data usage, access patterns, and hints from the database that indicate the type of data being accessed.

Exadata Smart Flash Cache can operate in Write-Through mode or Write-Back mode. In Write-Through mode, database writes go to disk first and subsequently populate Flash Cache. If a flash device fails with Exadata Smart Flash Cache operating in Write-Through mode, there is no data loss because the data is already on disk.

In Write-Back mode, database writes go to Flash Cache first and later to disk. The contents of the Write-Back Flash Cache is persisted across reboots, eliminating any warm-up time needed to populate the cache. Write-intensive applications can benefit from Write-Back caching by taking advantage of the fast latencies provided by flash. The amount of disk I/O also reduces when the cache absorbs multiple writes to the same block before writing it to disk. However, if a flash device fails while using Write-Back mode, data that is not yet persistent to disk is lost and must be recovered from a mirror copy. For this reason, Write-Back mode is recommended in conjunction with using high redundancy (triple mirroring) to protect the database files.

Exadata Smart Flash Cache accelerates OLTP performance by providing fast I/O for frequently accessed data blocks. It also accelerates Decision Support System (DSS) performance by automatically caching frequently scanned data and temporary segments, providing columnar cache storage, and enabling other features that optimize the performance of large analytic queries and large loads.

On Extreme Flash (EF) storage servers, all of the data resides in flash. Consequently, Exadata Smart Flash Cache is not required for normal caching. However, in this case Exadata Smart Flash Cache is still used to host the Columnar Cache, which caches data in columnar format and optimizes various analytical queries.

Performance issues related to Exadata Smart Flash Cache typically exhibit increased `cell single block physical read` latencies in the database. However, occasional long latencies associated with `cell single block physical read` do not necessarily indicate a performance issue with Exadata Smart Flash Cache, and may simply indicate that the read request was not satisfied using Exadata Smart Flash Cache.

- Monitoring Exadata Smart Flash Cache Using AWR
- Monitoring Exadata Smart Flash Cache Using Database Statistics and Wait Events
- Monitoring Exadata Smart Flash Cache Using Exadata Metrics
- What to Look For When Monitoring Exadata Smart Flash Cache

6.3.1.1 Monitoring Exadata Smart Flash Cache Using AWR

Automatic Workload Repository (AWR) contains a wealth of information relating to Exadata Smart Flash Cache. Following are descriptions and examples of key sections in the AWR report that contain information about Exadata Smart Flash Cache. By reviewing these sections of the AWR report, administrators can understand how Exadata Smart Flash Cache operates.

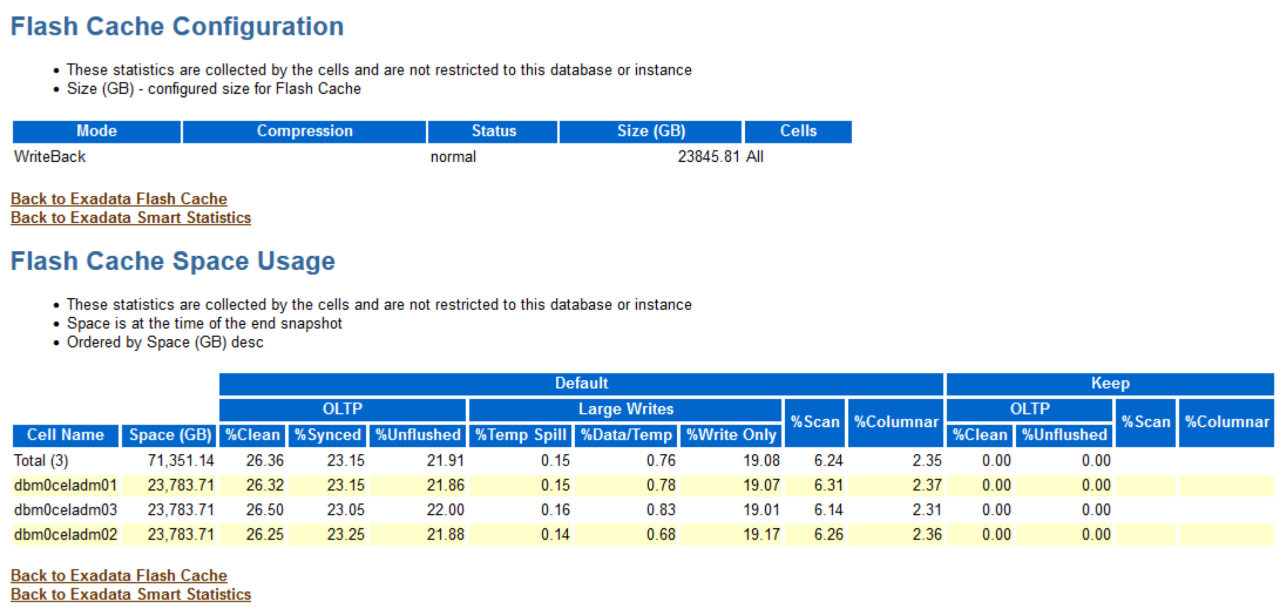
Flash Cache Configuration and Space Usage

The Flash Cache Configuration section contains summary information including the caching mode (Write-Through or Write-Back), status and overall size. The Flash Cache Space Usage section provides summary statistics on space usage in Exadata Smart Flash Cache.

The following example shows Exadata Smart Flash Cache configured in Write-Back mode on all cells, with a size of almost 24 TB on each cell. In the example, the Flash Cache Space Usage section shows that:

- Practically all of the Exadata Smart Flash Cache space is in use on each cell.
- Approximately 70% of the allocated space is being used for OLTP, which is typically data that is read into the database buffer cache using block mode reads. The OLTP data is further categorized as follows:
  - Clean data is data that is unchanged since being read into the cache. In the example output, approximately 26% of the space in each cache contains clean OLTP data.
  - If cached OLTP data is updated and the update is written back to the primary storage (usually a hard disk drive), the data is classified as synced. In the example output, approximately 23% of the space in each cache contains synced OLTP data.
  - When cached OLTP data is updated, but the update has not been written to the primary storage, the data is classified as unflushed. In the example output, approximately 22% of the space in each cache contains unflushed OLTP data.
- Approximately 20% of the cache space is being used to absorb large writes.
- Approximately 6% of the cache supports scan operations.
- Approximately 2% of the cache is being used to support the columnar cache.
- In this case, none of the cache space is occupied by keep objects. Keep objects are database segments (tables, partitions, and so on) that override the default caching policy for Exadata Smart Flash Cache by using the `CELL_FLASH_CACHE` storage clause option with the `KEEP` setting.

Figure 6-4 AWR Report: Flash Cache Configuration and Space Usage



Flash Cache Configuration

- These statistics are collected by the cells and are not restricted to this database or instance
- Size (GB) - configured size for Flash Cache

Mode	Compression	Status	Size (GB)	Cells
WriteBack		normal	23845.81	All

Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache Space Usage

- These statistics are collected by the cells and are not restricted to this database or instance
- Space is at the time of the end snapshot
- Ordered by Space (GB) desc

		Default								Keep			
		OLTP			Large Writes			%Scan	%Columnar	OLTP		%Scan	%Columnar
Cell Name	Space (GB)	%Clean	%Synced	%Unflushed	%Temp Spill	%Data/Temp	%Write Only			%Clean	%Unflushed		
Total (3)	71,351.14	26.36	23.15	21.91	0.15	0.76	19.08	6.24	2.35	0.00	0.00		
dbm0celadm01	23,783.71	26.32	23.15	21.86	0.15	0.78	19.07	6.31	2.37	0.00	0.00		
dbm0celadm03	23,783.71	26.50	23.05	22.00	0.16	0.83	19.01	6.14	2.31	0.00	0.00		
dbm0celadm02	23,783.71	26.25	23.25	21.88	0.14	0.68	19.17	6.26	2.36	0.00	0.00		

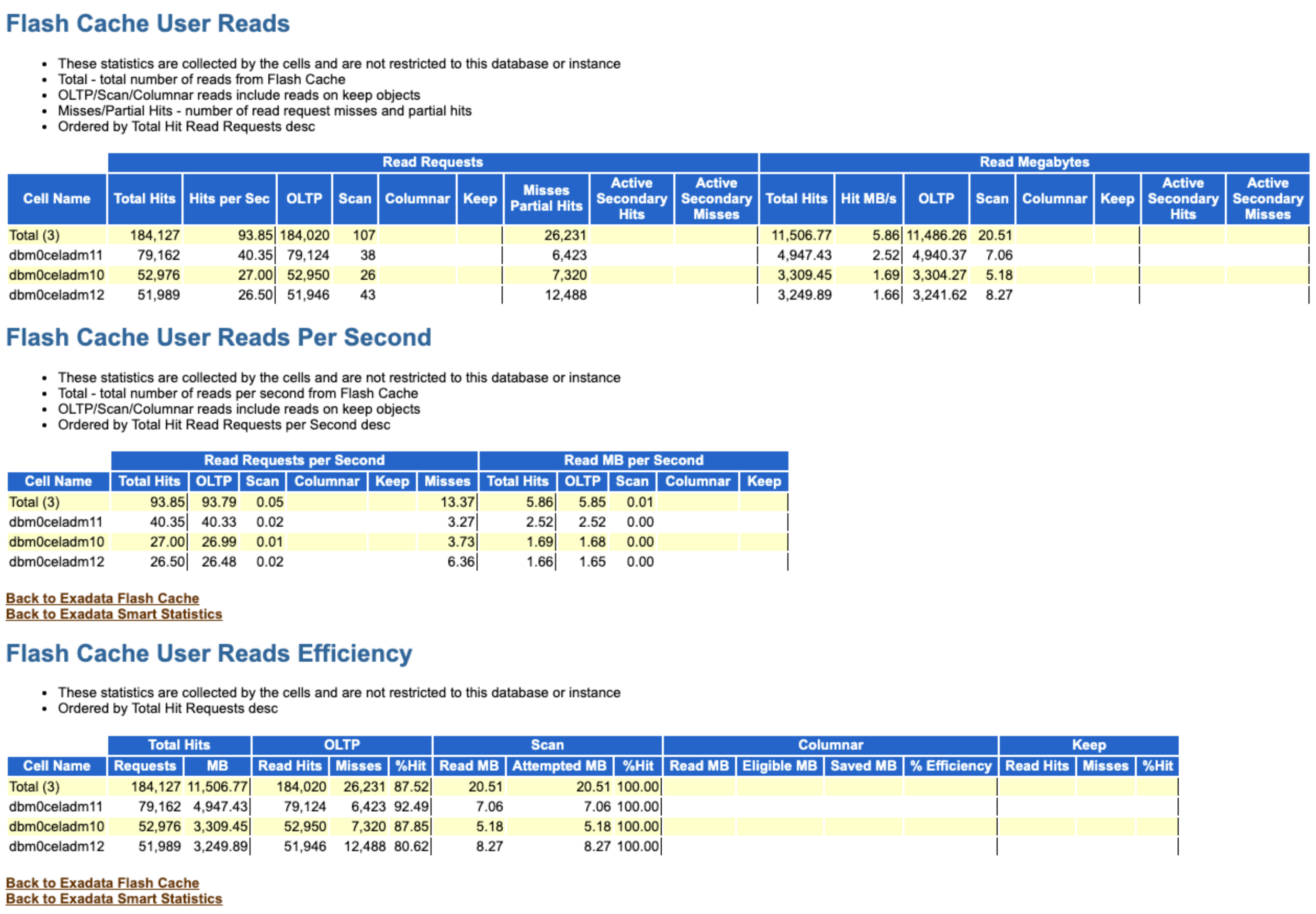
Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache User Reads

The Flash Cache User Reads sections show information about read requests, read throughput, and read efficiency from database clients. The statistics show the I/Os against different areas of Exadata Smart Flash Cache:

- OLTP - relates to block requests
- Scan - relates to scan requests
- Columnar - relates to columnar cache requests
- Keep - relates to read requests against the KEEP pool

Figure 6-5 AWR Report: Flash Cache User Reads



Flash Cache User Reads

- These statistics are collected by the cells and are not restricted to this database or instance
- Total - total number of reads from Flash Cache
- OLTP/Scan/Columnar reads include reads on keep objects
- Misses/Partial Hits - number of read request misses and partial hits
- Ordered by Total Hit Read Requests desc

	Read Requests									Read Megabytes							
Cell Name	Total Hits	Hits per Sec	OLTP	Scan	Columnar	Keep	Misses Partial Hits	Active Secondary Hits	Active Secondary Misses	Total Hits	Hit MB/s	OLTP	Scan	Columnar	Keep	Active Secondary Hits	Active Secondary Misses
Total (3)	184,127	93.85	184,020	107			26,231			11,506.77	5.86	11,486.26	20.51				
dbm0celadm11	79,162	40.35	79,124	38			6,423			4,947.43	2.52	4,940.37	7.06				
dbm0celadm10	52,976	27.00	52,950	26			7,320			3,309.45	1.69	3,304.27	5.18				
dbm0celadm12	51,989	26.50	51,946	43			12,488			3,249.89	1.66	3,241.62	8.27				

Flash Cache User Reads Per Second

- These statistics are collected by the cells and are not restricted to this database or instance
- Total - total number of reads per second from Flash Cache
- OLTP/Scan/Columnar reads include reads on keep objects
- Ordered by Total Hit Read Requests per Second desc

	Read Requests per Second						Read MB per Second				
Cell Name	Total Hits	OLTP	Scan	Columnar	Keep	Misses	Total Hits	OLTP	Scan	Columnar	Keep
Total (3)	93.85	93.79	0.05			13.37	5.86	5.85	0.01		
dbm0celadm11	40.35	40.33	0.02			3.27	2.52	2.52	0.00		
dbm0celadm10	27.00	26.99	0.01			3.73	1.69	1.68	0.00		
dbm0celadm12	26.50	26.48	0.02			6.36	1.66	1.65	0.00		

Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache User Reads Efficiency

- These statistics are collected by the cells and are not restricted to this database or instance
- Ordered by Total Hit Requests desc

	Total Hits		OLTP			Scan			Columnar				Keep		
Cell Name	Requests	MB	Read Hits	Misses	%Hit	Read MB	Attempted MB	%Hit	Read MB	Eligible MB	Saved MB	% Efficiency	Read Hits	Misses	%Hit
Total (3)	184,127	11,506.77	184,020	26,231	87.52	20.51	20.51	100.00							
dbm0celadm11	79,162	4,947.43	79,124	6,423	92.49	7.06	7.06	100.00							
dbm0celadm10	52,976	3,309.45	52,950	7,320	87.85	5.18	5.18	100.00							
dbm0celadm12	51,989	3,249.89	51,946	12,488	80.62	8.27	8.27	100.00							

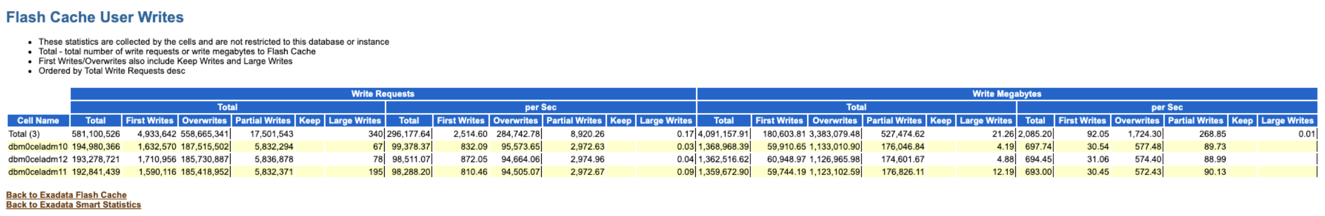
Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache User Writes

The Flash Cache User Writes section shows information about write requests and write throughput for Exadata Smart Flash Cache in Write-Back mode.

In this section, First Writes indicate new data being written to Exadata Smart Flash Cache, while Overwrites indicate data being overwritten in Exadata Smart Flash Cache. First Writes also require writing additional flash cache metadata. Overwrites represents the disk writes that were avoided by using Exadata Smart Flash Cache in Write-Back mode.

Figure 6-6 AWR Report: Flash Cache User Writes



Flash Cache User Writes

- These statistics are collected by the cells and are not restricted to this database or instance
- Total - total number of write requests or write megabytes to Flash Cache
- First Writes/Overwrites also include Keep Writes and Large Writes
- Ordered by Total Write Requests desc

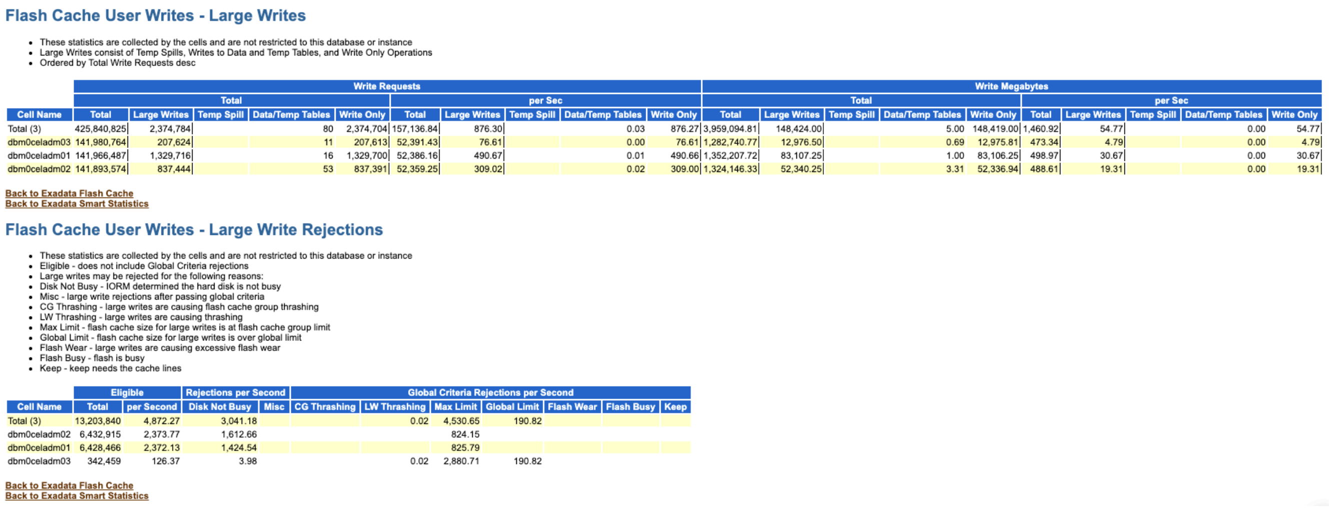
	Write Requests												Write Megabytes											
	Total						per Sec						Total						per Sec					
Cell Name	Total	First Writes	Overwrites	Partial Writes	Keep	Large Writes	Total	First Writes	Overwrites	Partial Writes	Keep	Large Writes	Total	First Writes	Overwrites	Partial Writes	Keep	Large Writes	Total	First Writes	Overwrites	Partial Writes	Keep	Large Writes
Total (3)	581,100,526	4,933,642	558,665,341	17,501,543		340	296,177.64	2,514.60	284,742.78	8,920.26		0.17	4,091,157.91	180,603.81	3,383,079.48	527,474.62		21.26	2,085.20	92.05	1,724.30	268.85		0.01
dbm0celadm10	194,980,366	1,632,570	187,515,502	5,832,294		67	99,378.37	832.09	95,573.65	2,972.63		0.03	1,368,968.39	59,910.65	1,133,010.90	176,046.84		4.19	697.74	30.54	577.48	89.73		
dbm0celadm12	193,278,721	1,710,956	185,730,887	5,836,878		78	98,511.07	872.05	94,664.06	2,974.96		0.04	1,362,516.62	60,948.97	1,126,965.98	174,601.67		4.88	694.45	31.06	574.40	88.99		
dbm0celadm11	192,841,439	1,590,116	185,418,952	5,832,371		195	98,288.20	810.46	94,505.07	2,972.67		0.09	1,359,672.90	59,744.19	1,123,102.59	176,826.11		12.19	693.00	30.45	572.43	90.13		

Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache User Writes - Large Writes

The Flash Cache User Writes - Large Writes sections show information about large write requests that are absorbed and rejected by Exadata Smart Flash Cache in Write-Back mode.

Figure 6-7 AWR Report: Flash Cache User Writes - Large Writes



Flash Cache User Writes - Large Writes

- These statistics are collected by the cells and are not restricted to this database or instance
- Large Writes consist of Temp Spills, Writes to Data and Temp Tables, and Write Only Operations
- Ordered by Total Write Requests desc

	Write Requests										Write Megabytes									
	Total					per Sec					Total					per Sec				
Cell Name	Total	Large Writes	Temp Spill	Data/Temp Tables	Write Only	Total	Large Writes	Temp Spill	Data/Temp Tables	Write Only	Total	Large Writes	Temp Spill	Data/Temp Tables	Write Only	Total	Large Writes	Temp Spill	Data/Temp Tables	Write Only
Total (3)	425,840,825	2,374,784		80	2,374,704	157,136.84	876.30		0.03	876.27	3,959,094.81	148,424.00		5.00	148,419.00	1,460.92	54.77		0.00	54.77
dbm0celadm03	141,980,764	207,624		11	207,613	52,391.43	76.61		0.00	76.61	1,282,740.77	12,976.50		0.69	12,975.81	473.34	4.79		0.00	4.79
dbm0celadm01	141,966,487	1,329,716		16	1,329,700	52,386.16	490.67		0.01	490.66	1,352,207.72	83,107.25		1.00	83,106.25	498.97	30.67		0.00	30.67
dbm0celadm02	141,893,574	837,444		53	837,391	52,359.25	309.02		0.02	309.00	1,324,146.33	52,340.25		3.31	52,336.94	488.61	19.31		0.00	19.31

Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache User Writes - Large Write Rejections

- These statistics are collected by the cells and are not restricted to this database or instance
- Eligible - does not include Global Criteria rejections
- Large writes may be rejected for the following reasons:
- Disk Not Busy - IORM determined the hard disk is not busy
- Misc - large write rejections after passing global criteria
- CG Thrashing - large writes are causing flash cache group thrashing
- LW Thrashing - large writes are causing thrashing
- Max Limit - flash cache size for large writes is at flash cache group limit
- Global Limit - flash cache size for large writes is over global limit
- Flash Wear - large writes are causing excessive flash wear
- Flash Busy - flash is busy
- Keep - keep needs the cache lines

	Eligible		Rejections per Second		Global Criteria Rejections per Second						
Cell Name	Total	per Second	Disk Not Busy	Misc	CG Thrashing	LW Thrashing	Max Limit	Global Limit	Flash Wear	Flash Busy	Keep
Total (3)	13,203,840	4,872.27	3,041.18			0.02	4,530.65	190.82			
dbm0celadm02	6,432,915	2,373.77	1,612.66				824.15				
dbm0celadm01	6,428,466	2,372.13	1,424.54				825.79				
dbm0celadm03	342,459	126.37	3.98			0.02	2,880.71	190.82			

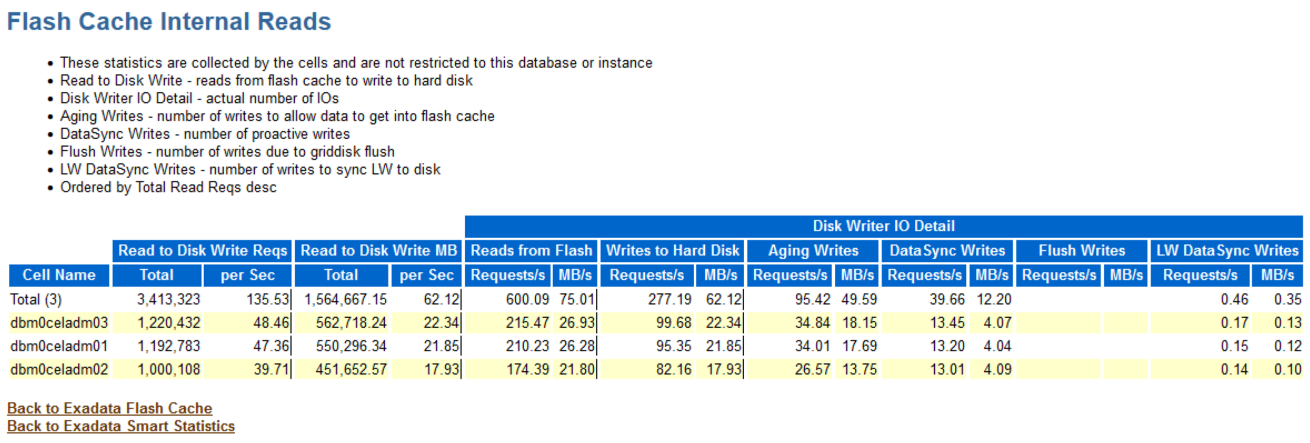
Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache Internal Reads

The Flash Cache Internal Reads section shows reads from Exadata Smart Flash Cache that are performed by Oracle Exadata System Software. These statistics are populated when Exadata Smart Flash Cache is in Write-Back mode.

The Disk Writer IO Detail columns relate to internal I/Os performed to persist data from Exadata Smart Flash Cache to hard disk devices. These columns show the reads from flash and the various categories of writes to hard disk.

Figure 6-8 AWR Report: Flash Cache Internal Reads



Flash Cache Internal Reads

- These statistics are collected by the cells and are not restricted to this database or instance
- Read to Disk Write - reads from flash cache to write to hard disk
- Disk Writer IO Detail - actual number of IOs
- Aging Writes - number of writes to allow data to get into flash cache
- DataSync Writes - number of proactive writes
- Flush Writes - number of writes due to griddisk flush
- LW DataSync Writes - number of writes to sync LW to disk
- Ordered by Total Read Reqs desc

					Disk Writer IO Detail											
	Read to Disk Write Reqs		Read to Disk Write MB		Reads from Flash		Writes to Hard Disk		Aging Writes		DataSync Writes		Flush Writes		LW DataSync Writes	
Cell Name	Total	per Sec	Total	per Sec	Requests/s	MB/s	Requests/s	MB/s	Requests/s	MB/s	Requests/s	MB/s	Requests/s	MB/s	Requests/s	MB/s
Total (3)	3,413,323	135.53	1,564,667.15	62.12	600.09	75.01	277.19	62.12	95.42	49.59	39.66	12.20			0.46	0.35
dbm0celadm03	1,220,432	48.46	562,718.24	22.34	215.47	26.93	99.68	22.34	34.84	18.15	13.45	4.07			0.17	0.13
dbm0celadm01	1,192,783	47.36	550,296.34	21.85	210.23	26.28	95.35	21.85	34.01	17.69	13.20	4.04			0.15	0.12
dbm0celadm02	1,000,108	39.71	451,652.57	17.93	174.39	21.80	82.16	17.93	26.57	13.75	13.01	4.09			0.14	0.10

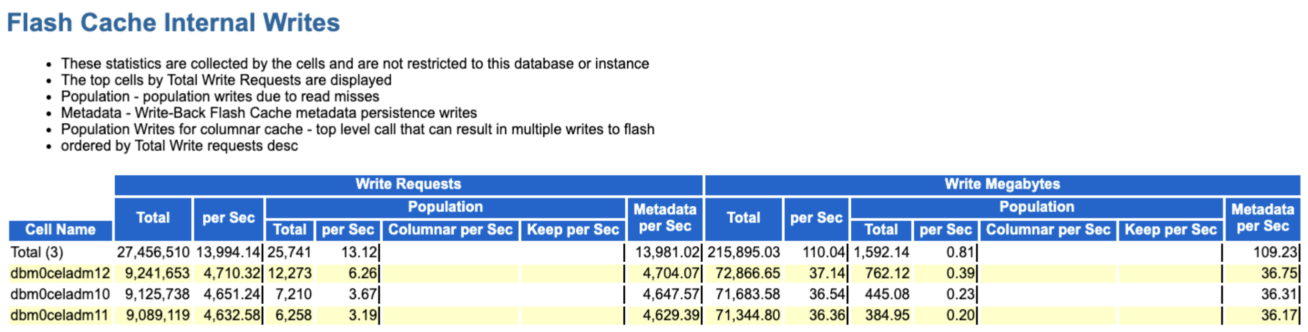
Back to Exadata Flash Cache
Back to Exadata Smart Statistics

Flash Cache Internal Writes

The Flash Cache Internal Writes section shows writes to Exadata Smart Flash Cache that are performed by Oracle Exadata System Software. The internal writes are I/Os that populate Exadata Smart Flash Cache in response to a cache read miss.

The statistics also include metadata writes that occur when Exadata Smart Flash Cache is in Write-Back mode. Metadata writes occur when a cacheline is used to cache new data.

Figure 6-9 AWR Report: Flash Cache Internal Writes



Flash Cache Internal Writes

- These statistics are collected by the cells and are not restricted to this database or instance
- The top cells by Total Write Requests are displayed
- Population - population writes due to read misses
- Metadata - Write-Back Flash Cache metadata persistence writes
- Population Writes for columnar cache - top level call that can result in multiple writes to flash
- ordered by Total Write requests desc

	Write Requests							Write Megabytes						
	Total	per Sec	Population				Metadata per Sec	Total	per Sec	Population				Metadata per Sec
Cell Name			Total	per Sec	Columnar per Sec	Keep per Sec				Total	per Sec	Columnar per Sec	Keep per Sec	
Total (3)	27,456,510	13,994.14	25,741	13.12			13,981.02	215,895.03	110.04	1,592.14	0.81			109.23
dbm0celadm12	9,241,653	4,710.32	12,273	6.26			4,704.07	72,866.65	37.14	762.12	0.39			36.75
dbm0celadm10	9,125,738	4,651.24	7,210	3.67			4,647.57	71,683.58	36.54	445.08	0.23			36.31
dbm0celadm11	9,089,119	4,632.58	6,258	3.19			4,629.39	71,344.80	36.36	384.95	0.20			36.17

6.3.1.2 Monitoring Exadata Smart Flash Cache Using Database Statistics and Wait Events

The following table describes various database statistics that are useful for monitoring Exadata Smart Flash Cache. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`cell flash cache read hits`	The number of read requests that were satisfied by the cache
`cell flash cache read hits for smart IO`	The number of read requests for smart IO that were satisfied by the cache
`cell flash cache read hits for temp IO`	The number of read requests for temp IO that were satisfied by the cache
`cell flash read hits for controlfile reads`	The number of read requests for a database control file that were satisfied by the cache
`cell overwrites in flash`	Total number of write requests that overwrote an existing cacheline in Exadata Smart Flash Cache that had not been written to disk. In effect, this is the amount of disk I/O saved by using Write-Back mode. This statistic is incremented once per mirror write.
`cell partial writes in flash`	Total number of write requests written to both Exadata Smart Flash Cache and disk. Part of the data was written to flash, and the rest was written to disk. This statistic is incremented once per mirror write.

Statistic	Description
`cell writes to flash cache`	Total number of write requests written entirely to Exadata Smart Flash Cache. This statistic is incremented once per mirror write.
`cell writes to flash cache for temp IO`	The number of write requests for temporary segments that were absorbed by Exadata Smart Flash Cache
`physical read IO requests`	Number of physical read requests issued by the database.
`physical read requests optimized`	Total number of read requests satisfied by using Exadata Smart Flash Cache, and read requests avoided by using storage index or columnar cache.
`physical read total bytes`	Total amount of I/O bytes for reads issued by the database, whether or not the read was offloaded to the storage servers.
`physical read total bytes optimized`	Total number of bytes read from Exadata Smart Flash Cache, and bytes avoided by using storage index or columnar cache.
`physical read total IO requests`	Total number of read requests issued by the database for all activity including application, backup, recovery, and other utilities.
`physical write IO requests`	Number of physical write requests issued by the database.
`physical write total bytes`	Total number of IO bytes for writes issued by the database for all activity.
`physical write total bytes optimized`	Total number of bytes written to Exadata Smart Flash Cache. These writes are synchronized to disk in a lazy manner.
`physical write total IO requests`	Total number of write requests issued by the database for all activity, including application, backup, recovery and other utilities.

The following table describes database wait events that are useful for monitoring Exadata Smart Flash Cache. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

The latency of the `cell single block physical read` event typically indicates if the read is satisfied from Exadata Smart Flash Cache. The higher the latency, the higher the likelihood that the request was not satisfied from Exadata Smart Flash Cache.

Wait Event	Description
`cell list of blocks physical read`	This wait event occurs during recovery or during buffer pre-fetching (rather than performing multiple single-block reads). It is used to monitor database blocks that must be changed as part of recovery and are read in parallel for the database. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file parallel read` for a cell.
`cell list of blocks read request`	This is a placeholder wait event associated with `cell list of blocks physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell list of blocks physical read`.
`cell multiblock physical read`	This wait event represents the time taken to perform all the I/Os for a multi-block database read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file scattered read` for a cell.
`cell multiblock read request`	This is a placeholder wait event associated with `cell multiblock physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell multiblock physical read`.

Wait Event	Description
`cell single block physical read`	This wait event represents the time taken to perform a single block database I/O. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file sequential read` for a cell. Commencing with the May 2022 Oracle Database release updates (versions 19.15.0.0.220419, 21.6.0.0.220419, and later), this wait event no longer includes I/O from Exadata Smart Flash Cache or database I/O using a Remote Direct Memory Access (RDMA) read.
`cell single block physical read: flash cache`	This wait event represents the time taken to perform a single block database I/O from Exadata Smart Flash Cache. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data (not the cache location), which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.
`cell single block read request`	This is a placeholder wait event associated with a single block database I/O that is visible only during the wait period. After the wait event ends, the placeholder is converted to the appropriate wait event, which is typically one of the `cell single block physical read` events.

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.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.1.3 Monitoring Exadata Smart Flash Cache Using Exadata Metrics

Exadata metrics that are related to Exadata Smart Flash Cache are identified in the Exadata storage server `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=FLASHCACHE`.

Example 6-1   Displaying Flash Cache Metric Definitions

This example shows how to display the flash cache metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
FLASHCACHE
         FC_BYKEEP_DIRTY                         "Number of megabytes
unflushed for keep objects on FlashCache"
         FC_BYKEEP_OLTP                          "Number of megabytes for
OLTP keep objects in flash cache"
         FC_BYKEEP_OVERWR                        "Number of megabytes pushed
out of the FlashCache because of space limit for keep objects"
         FC_BYKEEP_OVERWR_SEC                    "Number of megabytes per
second pushed out of the FlashCache because of space limit for keep objects"
         FC_BYKEEP_USED                          "Number of megabytes used
for keep objects on FlashCache"
         FC_BY_ALLOCATED                         "Number of megabytes
allocated in flash cache"
         FC_BY_ALLOCATED_DIRTY                   "Number of megabytes
allocated for unflushed data in flash cache"
         FC_BY_ALLOCATED_OLTP                    "Number of megabytes
allocated for OLTP data in flash cache"
         FC_BY_DIRTY                             "Number of unflushed
megabytes in FlashCache"
         FC_BY_STALE_DIRTY                       "Number of unflushed
megabytes in FlashCache which cannot be flushed because cached disks are not
accessible"
         FC_BY_USED                              "Number of megabytes used on
FlashCache"
         FC_COL_BYKEEP_USED                      "Number of megabytes used
for keep objects in Columnar FlashCache"
         FC_COL_BY_USED                          "Number of megabytes used in
Columnar FlashCache"
         FC_COL_IO_BYKEEP_R                      "Number of megabytes read
```

```
from Columnar FlashCache for keep objects"
         FC_COL_IO_BYKEEP_R_SEC                    "Number of megabytes read
per second from Columnar FlashCache for keep objects"
         FC_COL_IO_BY_R                            "Number of megabytes that
were read from Columnar FlashCache"
         FC_COL_IO_BY_R_ELIGIBLE                   "Number of megabytes
eligible to read from Columnar FlashCache"
         FC_COL_IO_BY_R_ELIGIBLE_SEC               "Number of megabytes per
second eligible to read from Columnar FlashCache"
         FC_COL_IO_BY_R_SEC                        "Number of megabytes per
second that were read from Columnar FlashCache"
         FC_COL_IO_BY_SAVED                        "Number of megabytes saved
by reads from Columnar FlashCache"
         FC_COL_IO_BY_SAVED_SEC                    "Number of megabytes saved
per second by reads from Columnar FlashCache"
         FC_COL_IO_BY_W_POPULATE                   "Number of megabytes that
are population writes into Columnar FlashCache due to read miss"
         FC_COL_IO_BY_W_POPULATE_SEC               "Number of megabytes per
second that are population writes into Columnar FlashCache due to read miss"
         FC_COL_IO_RQKEEP_R                        "Number of requests read for
keep objects from Columnar FlashCache"
         FC_COL_IO_RQKEEP_R_SEC                    "Number of requests read per
second for keep objects from Columnar FlashCache"
         FC_COL_IO_RQ_R                            "Number of requests that
were read from Columnar FlashCache"
         FC_COL_IO_RQ_R_ELIGIBLE                   "Number of reads eligible
for Columnar FlashCache"
         FC_COL_IO_RQ_R_ELIGIBLE_SEC               "Number of reads per second
eligible for Columnar FlashCache"
         FC_COL_IO_RQ_R_SEC                        "Number of requests per
second that were read from Columnar FlashCache"
         FC_COL_IO_RQ_W_POPULATE                   "Number of requests that are
population writes into Columnar FlashCache due to read miss"
         FC_COL_IO_RQ_W_POPULATE_SEC               "Number of requests per
second that are population writes into Columnar FlashCache due to read miss"
         FC_IO_BYKEEP_R                            "Number of megabytes read
from FlashCache for keep objects"
         FC_IO_BYKEEP_R_SEC                        "Number of megabytes read
per second from FlashCache for keep objects"
         FC_IO_BYKEEP_W                            "Number of megabytes written
to FlashCache for keep objects"
         FC_IO_BYKEEP_W_SEC                        "Number of megabytes per
second written to FlashCache for keep objects"
         FC_IO_BY_DISK_WRITE                       "Number of megabytes written
from flash cache to hard disks"
         FC_IO_BY_DISK_WRITE_SEC                   "Number of megabytes per
second written from flash cache to hard disks"
         FC_IO_BY_R                                "Number of megabytes of
small reads (OLTP) from flash cache"
         FC_IO_BY_R_ACTIVE_SECONDARY               "Number of megabytes for
active secondary reads satisfied from flash cache"
         FC_IO_BY_R_ACTIVE_SECONDARY_MISS          "Number of megabytes for
active secondary reads not satisfied from flash cache"
         FC_IO_BY_R_ACTIVE_SECONDARY_MISS_SEC      "Number of megabytes per
second for active secondary reads not satisfied from flash cache"
         FC_IO_BY_R_ACTIVE_SECONDARY_SEC           "Number of megabytes per
```

```
second for active secondary reads satisfied from flash cache"
         FC_IO_BY_R_DISK_WRITER                    "Number of megabytes read
from flash cache by disk writer"
         FC_IO_BY_R_DISK_WRITER_SEC                "Number of megabytes per
second read from flash cache by disk writer"
         FC_IO_BY_R_DW                             "Number of megabytes of
large reads (DW) from flash cache"
         FC_IO_BY_R_DW_SEC                         "Number of megabytes of
large reads (DW) per second from flash cache"
         FC_IO_BY_R_MISS                           "Number of megabytes of
small reads (OLTP) from disks because some of the requested data was not in
flash cache"
         FC_IO_BY_R_MISS_DW                        "Number of megabytes of
large reads (DW) from disks because some of the requested data was not in
flash cache"
         FC_IO_BY_R_MISS_DW_SEC                    "Number of megabytes of
large reads (DW) per second from disks because some of the requested data was
not in flash cache"
         FC_IO_BY_R_MISS_SEC                       "Number of megabytes of
small reads (OLTP) per second from disks because some of the requested data
was not in flash cache"
         FC_IO_BY_R_SEC                            "Number of megabytes of
small reads (OLTP) per second from flash cache"
         FC_IO_BY_R_SKIP                           "Number of megabytes read
from disks for IO requests that bypass FlashCache"
         FC_IO_BY_R_SKIP_FC_THROTTLE               "Number of megabytes read
from disk for IO requests that bypass FlashCache due to heavy load on
FlashCache"
         FC_IO_BY_R_SKIP_FC_THROTTLE_SEC           "Number of megabytes read
per second from disk for IO requests that bypass FlashCache due to heavy load
on FlashCache"
         FC_IO_BY_R_SKIP_LG                        "Number of megabytes read
from disk for IO requests that bypass FlashCache due to the large IO size"
         FC_IO_BY_R_SKIP_LG_SEC                    "Number of megabytes read
per second from disk for IO requests that bypass FlashCache due to the large
IO size"
         FC_IO_BY_R_SKIP_NCMIRROR                  "Number of megabytes read
from disk for IO requests that bypass FlashCache as the IO is on non-primary,
non-active secondary mirror"
         FC_IO_BY_R_SKIP_SEC                       "Number of megabytes read
from disks per second for IO requests that bypass FlashCache"
         FC_IO_BY_W                                "Number of megabytes written
to FlashCache"
         FC_IO_BY_W_DISK_WRITER                    "Number of megabytes written
to hard disks by disk writer"
         FC_IO_BY_W_DISK_WRITER_SEC                "Number of megabytes per
second written to hard disks by disk writer"
         FC_IO_BY_W_FIRST                          "Number of megabytes that
are first writes into flash cache"
         FC_IO_BY_W_FIRST_SEC                      "Number of megabytes per
second that are first writes into flash cache"
         FC_IO_BY_W_LG_CHINT                       "Number of megabytes used
for large writes to flash cache due to cache hint"
         FC_IO_BY_W_LG_DTAGE                       "Number of megabytes used
for data aging large writes to flash cache"
         FC_IO_BY_W_LG_MRCV                        "Number of megabytes used
```

```
for media recovery large writes to flash cache"
         FC_IO_BY_W_METADATA                     "Number of megabytes that
are flash cache metadata writes"
         FC_IO_BY_W_METADATA_SEC                 "Number of megabytes per
second that are flash cache metadata writes"
         FC_IO_BY_W_OVERWRITE                    "Number of megabytes that
are overwrites into flash cache"
         FC_IO_BY_W_OVERWRITE_SEC                "Number of megabytes per
second that are overwrites into flash cache"
         FC_IO_BY_W_POPULATE                     "Number of megabytes that
are population writes into flash cache due to read miss"
         FC_IO_BY_W_POPULATE_SEC                 "Number of megabytes per
second that are population writes into flash cache due to read miss"
         FC_IO_BY_W_SEC                          "Number of megabytes per
second written to FlashCache"
         FC_IO_BY_W_SKIP                         "Number of megabytes written
to disk for IO requests that bypass FlashCache"
         FC_IO_BY_W_SKIP_FC_THROTTLE             "Number of megabytes written
to disk for IO requests that bypass FlashCache due to heavy load on
FlashCache"
         FC_IO_BY_W_SKIP_FC_THROTTLE_SEC         "Number of megabytes written
per second to disk for IO requests that bypass FlashCache due to heavy load
on FlashCache"
         FC_IO_BY_W_SKIP_LG                      "Number of megabytes written
to disk for IO requests that bypass FlashCache due to the large IO size"
         FC_IO_BY_W_SKIP_LG_SEC                  "Number of megabytes written
per second to disk for IO requests that bypass FlashCache due to the large IO
size"
         FC_IO_BY_W_SKIP_NCMIRROR                "Number of megabytes written
to disk for IO requests that bypass FlashCache as the IO is on non-primary,
non-active secondary mirror"
         FC_IO_BY_W_SKIP_SEC                     "Number of megabytes written
to disk per second for IO requests that bypass FlashCache"
         FC_IO_ERRS                              "Number of IO errors on
FlashCache"
         FC_IO_RQKEEP_R                          "Number of read requests for
keep objects from FlashCache"
         FC_IO_RQKEEP_R_MISS                     "Number of read requests for
keep objects which did not find all data in FlashCache"
         FC_IO_RQKEEP_R_MISS_SEC                 "Number of read requests per
second for keep objects which did not find all data in FlashCache"
         FC_IO_RQKEEP_R_SEC                      "Number of read requests per
second for keep objects from FlashCache"
         FC_IO_RQKEEP_R_SKIP                     "Number of read requests for
keep objects that bypass FlashCache"
         FC_IO_RQKEEP_R_SKIP_SEC                 "Number of read requests per
second for keep objects that bypass FlashCache"
         FC_IO_RQKEEP_W                          "Number of requests for keep
objects which resulted in FlashCache being populated with data"
         FC_IO_RQKEEP_W_SEC                      "Number of requests per
second for keep objects which resulted in FlashCache being populated with
data"
         FC_IO_RQ_DISK_WRITE                     "Number of requests written
from flash cache to hard disks"
         FC_IO_RQ_DISK_WRITE_SEC                 "Number of requests per
second written from flash cache to hard disks"
```

```
         FC_IO_RQ_R                            "Number of small reads
(OLTP) satisfied from the flash cache"
         FC_IO_RQ_REPLACEMENT_ATTEMPTED        "Number of requests
attempted to find space in the flash cache"
         FC_IO_RQ_REPLACEMENT_DW_FAILED        "Number of times that client
DW IOs failed to find a replacement buffer"
         FC_IO_RQ_REPLACEMENT_DW_SUCCEEDED     "Number of times that client
DW IOs succeeded in finding a replacement buffer"
         FC_IO_RQ_REPLACEMENT_FAILED           "Number of requests failed
to find space in the flash cache"
         FC_IO_RQ_REPLACEMENT_OLTP_FAILED      "Number of times that client
OLTP IOs failed to find a replacement buffer"
         FC_IO_RQ_REPLACEMENT_OLTP_SUCCEEDED   "Number of times that client
OLTP IOs succeeded in finding a replacement buffer"
         FC_IO_RQ_R_ACTIVE_SECONDARY           "Number of requests for
active secondary reads satisfied from flash cache"
         FC_IO_RQ_R_ACTIVE_SECONDARY_MISS      "Number of requests for
active secondary reads not satisfied from flash cache"
         FC_IO_RQ_R_ACTIVE_SECONDARY_MISS_SEC  "Number of requests per
second for active secondary reads not satisfied from flash cache"
         FC_IO_RQ_R_ACTIVE_SECONDARY_SEC       "Number of requests per
second for active secondary reads satisfied from flash cache"
         FC_IO_RQ_R_DISK_WRITER                "Number of requests read
from flash cache by disk writer"
         FC_IO_RQ_R_DISK_WRITER_SEC            "Number of requests per
second read from flash cache by disk writer"
         FC_IO_RQ_R_DW                         "Number of large reads (DW)
satisfied from the flash cache"
         FC_IO_RQ_R_DW_SEC                     "Number of large reads (DW)
per second satisfied from the flash cache"
         FC_IO_RQ_R_MISS                       "Number of small reads
(OLTP) that did not find all data in flash cache"
         FC_IO_RQ_R_MISS_DW                    "Number of large reads (DW)
that did not find all data in flash cache"
         FC_IO_RQ_R_MISS_DW_SEC                "Number of large reads (DW)
per second that did not find all data in flash cache"
         FC_IO_RQ_R_MISS_SEC                   "Number of small reads
(OLTP) per second that did not find all data in flash cache"
         FC_IO_RQ_R_SEC                        "Number of small reads
(OLTP) per second satisfied from the flash cache"
         FC_IO_RQ_R_SKIP                       "Number of requests read
from disk that bypass FlashCache"
         FC_IO_RQ_R_SKIP_FC_THROTTLE           "Number of requests read
from disk that bypass FlashCache due to heavy load on FlashCache"
         FC_IO_RQ_R_SKIP_FC_THROTTLE_SEC       "Number of requests read
from disk per second that bypassed FlashCache due to heavy load on FlashCache"
         FC_IO_RQ_R_SKIP_LG                    "Number of requests read
from disk that bypass FlashCache due to the large IO size"
         FC_IO_RQ_R_SKIP_LG_SEC                "Number of requests read
from disk per second that bypass FlashCache due to the large IO size"
         FC_IO_RQ_R_SKIP_NCMIRROR              "Number of requests read
from disk that bypass FlashCache as the IO is on non-primary, non-active
secondary mirror"
         FC_IO_RQ_R_SKIP_SEC                   "Number of requests read
from disk per second that bypass FlashCache"
         FC_IO_RQ_W                            "Number of requests which
```

```
resulted in FlashCache being populated with data"
         FC_IO_RQ_W_DISK_WRITER                    "Number of requests written
to hard disks by disk writer"
         FC_IO_RQ_W_DISK_WRITER_SEC                "Number of requests per
second written to hard disks by disk writer"
         FC_IO_RQ_W_FIRST                          "Number of requests that are
first writes into flash cache"
         FC_IO_RQ_W_FIRST_SEC                      "Number of requests per
second that are first writes into flash cache"
         FC_IO_RQ_W_LG_CHINT                       "Number of large writes to
flash cache due to cache hint"
         FC_IO_RQ_W_LG_DTAGE                       "Number of data aging large
writes to flash cache"
         FC_IO_RQ_W_LG_MRCV                        "Number of media recovery
large writes to flash cache"
         FC_IO_RQ_W_METADATA                       "Number of requests that are
flash cache metadata writes"
         FC_IO_RQ_W_METADATA_SEC                   "Number of requests per
second that are flash cache metadata writes"
         FC_IO_RQ_W_OVERWRITE                      "Number of requests that are
overwrites into flash cache"
         FC_IO_RQ_W_OVERWRITE_SEC                  "Number of requests per
second that are overwrites into flash cache"
         FC_IO_RQ_W_POPULATE                       "Number of requests that are
population writes into flash cache due to read miss"
         FC_IO_RQ_W_POPULATE_SEC                   "Number of requests per
second that are population writes into flash cache due to read miss"
         FC_IO_RQ_W_SEC                            "Number of requests per
second which resulted in FlashCache being populated with data"
         FC_IO_RQ_W_SKIP                           "Number of requests written
to disk that bypass FlashCache"
         FC_IO_RQ_W_SKIP_FC_THROTTLE               "Number of requests written
to disk that bypass FlashCache due to heavy load on FlashCache"
         FC_IO_RQ_W_SKIP_FC_THROTTLE_SEC           "Number of requests written
to disk per second that bypass FlashCache due to heavy load on FlashCache"
         FC_IO_RQ_W_SKIP_LG                        "Number of requests written
to disk that bypass FlashCache due to the large IO size"
         FC_IO_RQ_W_SKIP_LG_SEC                    "Number of requests written
to disk per second that bypass FlashCache due to the large IO size"
         FC_IO_RQ_W_SKIP_NCMIRROR                  "Number of requests written
to disk that bypass FlashCache as the IO is on non-primary, non-active
secondary mirror"
         FC_IO_RQ_W_SKIP_SEC                       "Number of requests written
to disk per second that bypass FlashCache"
```

Note the following additional details:

- Space in Exadata Smart Flash Cache is internally managed in chunks known as cachelines. `FC_BY_ALLOCATED` represents the amount of space (in megabytes) that is allocated to cachelines in the flash cache. If the value is close to the flash cache size, then the flash cache is fully populated.
- `FC_BY_USED` represents amount of space (in megabytes) that is occupied by data in the flash cache. For some workloads, like OLTP, the `FC_BY_USED` value can be much less than the value of `FC_BY_ALLOCATED` because an OLTP write might only use a small fraction of a

cacheline. The total amount used by OLTP cachelines is tracked by `FC_BY_ALLOCATED_OLTP`.

- Various flash cache metrics having names ending with `_DW` track large read operations, which are typically associated with Data Warehouse workloads. Each of these metrics has a corresponding metric that tracks small read operations, which are typically associated with OLTP operations. To get a total count, add the large read metric value to the corresponding small read metric value.

  For example, to get the total number of reads satisfied from flash cache, add `FC_IO_RQ_R_DW` to `FC_IO_RQ_R`.

6.3.1.4 What to Look For When Monitoring Exadata Smart Flash Cache

Increased Read Latencies

Possible issues related to Exadata Smart Flash Cache tend to be visible in the database as increased read latencies, specifically in the `cell single block physical read` wait event. The increased latency is usually caused when reads are issued against the hard disks instead of being satisfied by Exadata Smart Flash Cache.

Read requests may not be satisfied from Exadata Smart Flash Cache when:

- The required data is not in Exadata Smart Flash Cache.

  In this case, requests are recorded as flash cache misses, which are visible as increased misses in the Flash Cache User Reads section of the AWR report. This is also typically accompanied by increased population writes, which are visible in the Flash Cache Internal Writes section of the AWR report.

  On the database, you may see a reduction in the number of `cell flash cache read hits` compared with the `physical read IO requests` or `physical read total IO requests`.

- The data is not eligible for Exadata Smart Flash Cache.

  To maximize cache efficiency, when an I/O request is sent from the database to the storage servers, it includes a hint that indicates whether or not the data should be cached in Exadata Smart Flash Cache.

  If data is not eligible for caching, the corresponding I/O is visible in the Flash Cache User Reads - Skips section of the AWR report, along with the reason why the read was not eligible. Possible reasons include:

  – The grid disk caching policy is set to `none`.

  – The database segment is configured with the `CELL_FLASH_CACHE NONE` storage option.

  – The I/O type and situation precludes caching. For example, the I/O is related to an RMAN backup operation and the hard disks are not busy. If this is a common occurrence, it will be evident in Top IO Reasons section of the AWR report

In some cases, the `cell single block physical read` latency may increase, with no apparent difference in Exadata Smart Flash Cache behavior. This can be caused by increased I/O load, especially on hard disks.

Occasional long latencies in `cell single block physical read`, usually visible as a long tail in the `cell single block physical read` histogram may simply indicate that the data is read from hard disk, and does not necessarily indicate a problem with Exadata Smart Flash Cache.

Skipped Writes

When using Exadata Smart Flash Cache in Write-Back mode, most database write requests are absorbed by the cache. However, some write requests may skip the cache. The most common reason for skipping the cache is when the request writes a large amount of data that will be read once, such as temp sorts, or it is not expected to be read at all in the foreseeable future, such as backups and archiving.

Another common reason for skipping large writes is `Disk Not Busy`, which typically means that there is no benefit to using Exadata Smart Flash Cache because the hard disks have sufficient capacity to handle the write requests.

If skipping Exadata Smart Flash Cache for large writes is causing a performance issue, then it is typically visible in the database with corresponding long latencies for the `direct path write` or `direct path write temp` wait events.

The reasons for rejecting large writes are visible in the Flash Cache User Writes - Large Write Rejections section of the AWR report.

Database Working Set Size

The database working set refers to the subset of most commonly accessed information in the database. In most cases, the database working set is fairly stable. However, if for any reason, the working set does not fit in Exadata Smart Flash Cache, you may see the following symptoms:

- Increased cache misses, indicating that data is not in Exadata Smart Flash Cache. This is visible in the Flash Cache User Reads section of the AWR report, or the `FC_IO_RQ_R_MISS_SEC` cell metric.
- Increased population activity to populate data not in Exadata Smart Flash Cache. This is visible in the Flash Cache Internal Writes section of the AWR report, or the `FC_IO_[BY|RQ]_W_POPULATE_SEC` cell metrics.
- Increased disk writer activity, indicating that dirty cachelines have to be written to disk, so that the cacheline can be reused to cache other data. Disk writer activity is visible in the Flash Cache Internal Reads section of the AWR report, or the `FC_IO_[RQ|BY]_[R|W]_DISK_WRITER_SEC` cell metrics.
- Increased first writes, indicating new data is being written to Exadata Smart Flash Cache. A large number of first writes with few overwrites means new data is being written to Exadata Smart Flash Cache. This is visible in the Flash Cache User Writes section of the AWR report, or in the `FC_IO_[RQ|BY]_W_FIRST_SEC` cell metrics.

In this case:

- Review the database access patterns for tuning opportunities that reduce the amount of data being accessed.
- Consider increasing the number of available storage servers to deliver more space for Exadata Smart Flash Cache.
- Review your I/O Resource Management (IORM) quotas for Exadata Smart Flash Cache and allocate space where it is most required.
- Consider using Extreme Flash storage servers to eliminate the disk I/Os.

Miscellaneous Issues

There are some cases when the increased `cell single block physical read` latency may not be due to cell performance, but may be caused by something else along the IO path, such as network contention or contention for database server CPU resources.

The histograms for `cell single block physical read` and small reads are available in the Single Block Reads and Small Read Histogram - Detail sections of the AWR report under Exadata Statistics > Performance Summary. The `cell single block physical read` histogram shows latencies measured by Oracle Database, while the small read histograms show latencies measured in the storage server.

A histogram with a significant number of occasional long latencies is said to have a long tail. When the histograms for `cell single block physical read` and small reads have long tails, then this is an indication of slow read times in the storage server, which would warrant further investigation of the other I/O performance statistics. See Monitoring Cell Disk I/O.

If the `cell single block physical read` histogram has a long tail that is not present in the small read histograms, then the cause is generally not in the storage server, but rather something else in the I/O path, such as bottlenecks in the network or contention for compute node CPU.

Related Topics

- Faster Performance for Large Analytic Queries and Large Loads

6.3.2 Monitoring XRMEM Cache

Exadata RDMA Memory Cache (XRMEM cache) provides direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling lower read latencies and faster response times. When database clients read from the XRMEM cache, the client software performs an RDMA read of the cached data, which bypasses the storage server software and results in much lower read latencies.

XRMEM cache works in conjunction with Exadata Smart Flash Cache. If available, data that is not in XRMEM cache may be retrieved from Exadata Smart Flash Cache.

Statistics from XRMEM cache are slightly different when compared with other Exadata components. Because clients issue RDMA I/O directly to XRMEM cache, the request does not go to `cellsrv`, so the storage server cannot account for the RDMA I/Os. For this reason, there are no cell metrics for XRMEM cache I/O. Instead, Oracle Database statistics account for the I/O that is performed using RDMA.

Performance issues related to XRMEM cache typically cause latency increases in the Oracle Database `cell single block physical read` wait events. However, bear in mind that Exadata Smart Flash Cache is still available to service the requests, and although requests from Exadata Smart Flash Cache generally experience higher read latencies compared to XRMEM cache, the requests still benefit from the fast I/O provided by flash.

- Monitoring XRMEM Cache Using AWR
- Monitoring XRMEM Cache Using Database Statistics and Wait Events
- Monitoring XRMEM Cache Using Exadata Metrics
- What to Look For When Monitoring XRMEM Cache

6.3.2.1 Monitoring XRMEM Cache Using AWR

Automatic Workload Repository (AWR) contains information relating to XRMEM cache. Following are descriptions and examples of the most commonly used sections in the AWR report that contain information about XRMEM cache. By reviewing these sections of the AWR report, administrators can understand the operation of XRMEM cache.

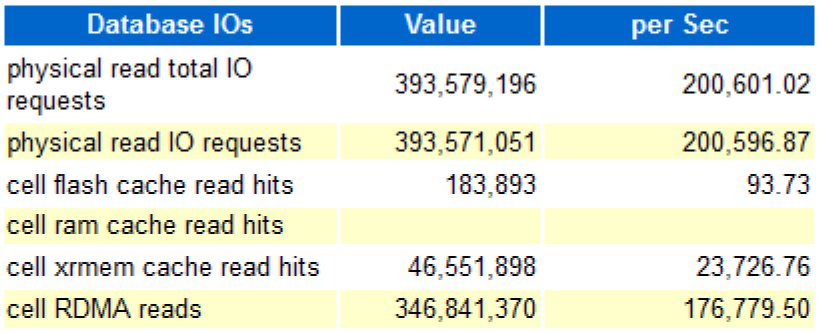
Note that because storage servers cannot account for the RDMA I/Os, the Oracle Database statistics are critical in understanding the use of XRMEM cache. The XRMEM cache statistics in the Exadata-specific sections of the AWR report only include I/Os that as serviced by `cellsrv`, which are not RDMA I/Os.

Database IOs

A summary of Database IOs may be found in the Single Block Reads section of the AWR report. This summary contains information about the effectiveness of XRMEM cache. The Single Block Reads section is located in the AWR report under Exadata Statistics > Performance Summary.

In the following example, the overwhelming majority of all read I/O requests (approximately 88%) are serviced using RDMA reads to XRMEM cache (`cell RDMA reads`), at a rate of more than 176,000 per second. Nearly all of the rest are satisfied using non-RDMA XRMEM cache reads (`cell xrmem cache read hits`), while the remaining reads are serviced by using Exadata Smart Flash Cache (`cell flash cache read hits`). Notice the massive difference in throughput for each different I/O type, which illustrates the power of XRMEM cache and RDMA.

Figure 6-10 AWR Report: Database IOs



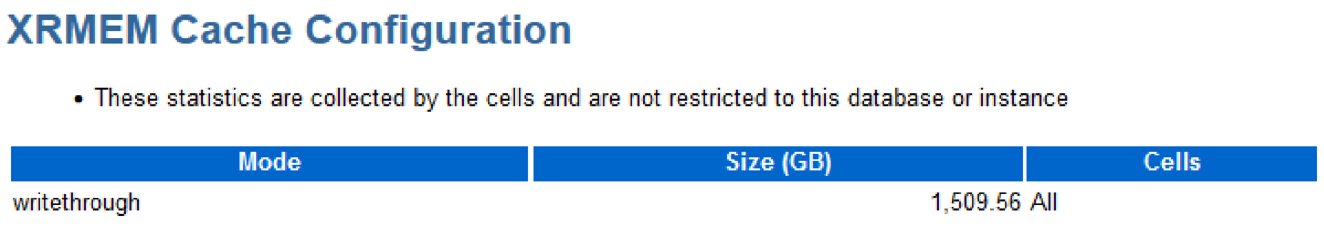
Database IOs	Value	per Sec
physical read total IO requests	393,579,196	200,601.02
physical read IO requests	393,571,051	200,596.87
cell flash cache read hits	183,893	93.73
cell ram cache read hits		
cell xrmem cache read hits	46,551,898	23,726.76
cell RDMA reads	346,841,370	176,779.50

XRMEM Cache Configuration and Space Usage

The XRMEM Cache Configuration section contains summary information including the caching mode (Write-Through or Write-Back) and overall size. The XRMEM Cache Space Usage section provides summary statistics on space usage in XRMEM cache.

The following example shows XRMEM cache configured in Write-Through mode on all cells, with a total size of approximately 1509 GB.

Figure 6-11 AWR Report: XRMEM Cache Configuration



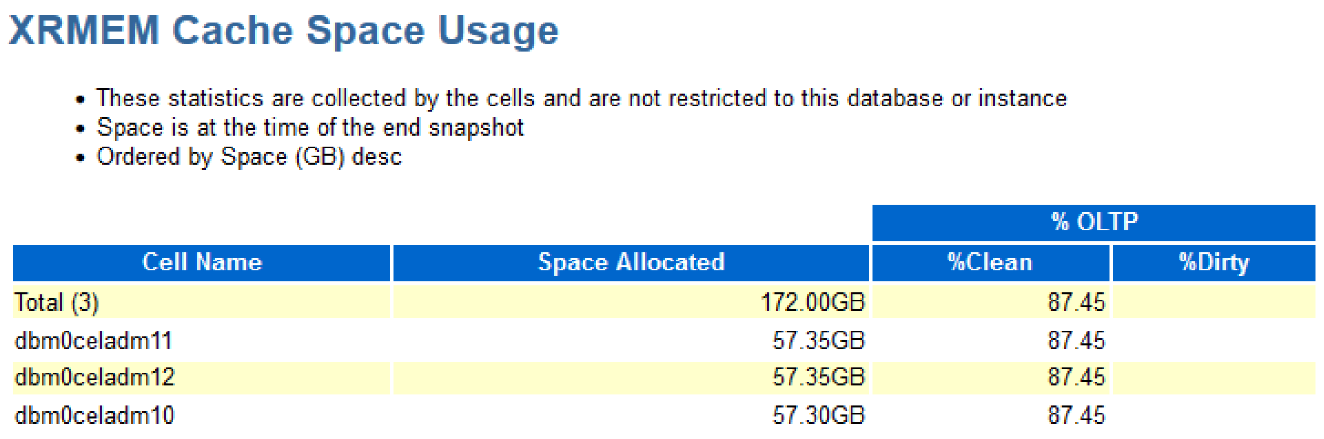
XRMEM Cache Configuration

- These statistics are collected by the cells and are not restricted to this database or instance

Mode	Size (GB)	Cells
writethrough	1,509.56	All

The following example of the XRMEM Cache Space Usage section shows 172 GB of XRMEM cache spread evenly across 3 cells.

Figure 6-12 AWR Report: XRMEM Cache Space Usage



XRMEM Cache Space Usage

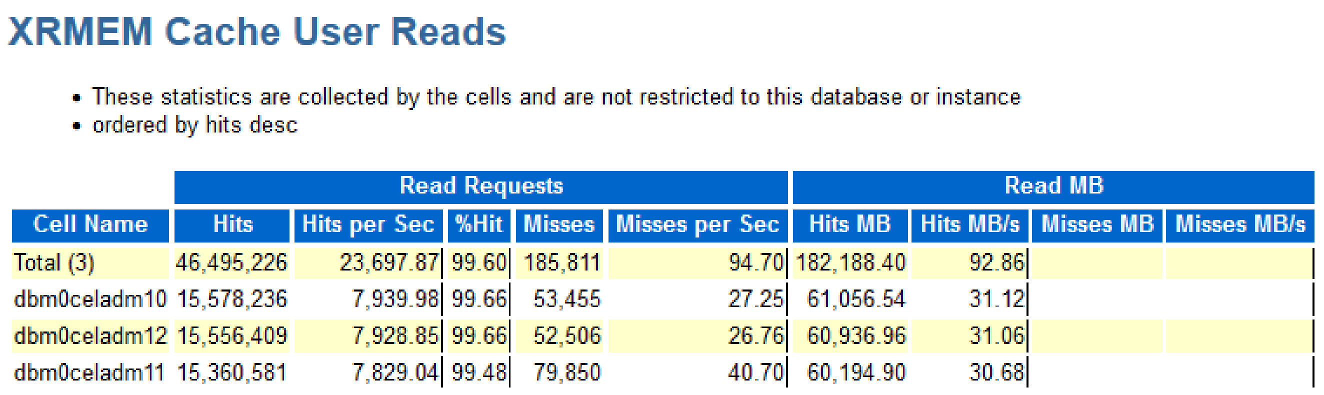
- These statistics are collected by the cells and are not restricted to this database or instance
- Space is at the time of the end snapshot
- Ordered by Space (GB) desc

		% OLTP	
Cell Name	Space Allocated	%Clean	%Dirty
Total (3)	172.00GB	87.45	
dbm0celadm11	57.35GB	87.45	
dbm0celadm12	57.35GB	87.45	
dbm0celadm10	57.30GB	87.45	

XRMEM Cache User Reads

The XRMEM Cache User Reads section shows information about read requests, read throughput, and read efficiency. Because storage servers cannot account for RDMA I/Os, the statistics only relate to non-RDMA reads processed by `cellsrv`. Hits are for I/Os satisfied using XRMEM cache, while misses indicate that the data was not in XRMEM cache.

Figure 6-13 AWR Report: XRMEM Cache User Reads



XRMEM Cache User Reads

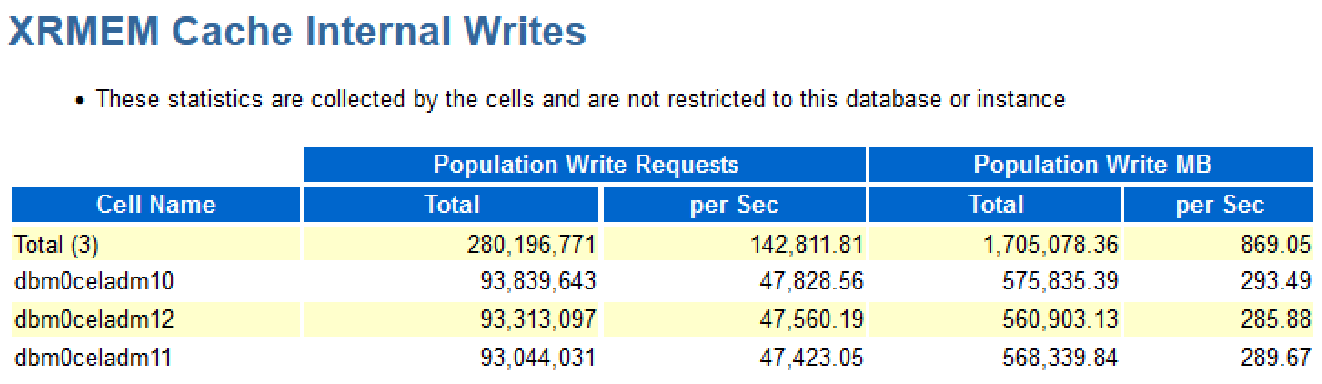
- These statistics are collected by the cells and are not restricted to this database or instance
- ordered by hits desc

	Read Requests					Read MB			
Cell Name	Hits	Hits per Sec	%Hit	Misses	Misses per Sec	Hits MB	Hits MB/s	Misses MB	Misses MB/s
Total (3)	46,495,226	23,697.87	99.60	185,811	94.70	182,188.40	92.86		
dbm0celadm10	15,578,236	7,939.98	99.66	53,455	27.25	61,056.54	31.12		
dbm0celadm12	15,556,409	7,928.85	99.66	52,506	26.76	60,936.96	31.06		
dbm0celadm11	15,360,581	7,829.04	99.48	79,850	40.70	60,194.90	30.68		

XRMEM Cache Internal Writes

The XRMEM Cache Internal Writes section shows writes to XRMEM cache that are performed by Oracle Exadata System Software. The internal writes are I/Os that populate XRMEM cache.

Figure 6-14 AWR Report: XRMEM Cache Internal Writes



XRMEM Cache Internal Writes

- These statistics are collected by the cells and are not restricted to this database or instance

	Population Write Requests		Population Write MB	
Cell Name	Total	per Sec	Total	per Sec
Total (3)	280,196,771	142,811.81	1,705,078.36	869.05
dbm0celadm10	93,839,643	47,828.56	575,835.39	293.49
dbm0celadm12	93,313,097	47,560.19	560,903.13	285.88
dbm0celadm11	93,044,031	47,423.05	568,339.84	289.67

6.3.2.2 Monitoring XRMEM Cache Using Database Statistics and Wait Events

Note:

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

The following table describes database statistics that are useful for monitoring XRMEM cache. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`cell RDMA reads`	The number of successful XRMEM cache read requests using RDMA
`cell RDMA reads eligible`	The number of database reads that meet basic eligibility criteria for RDMA
`cell RDMA read hash table probes`	The total number of RDMA hash table probes issued to determine the presence of data in XRMEM cache. Each eligible read is usually associated with a hash table probe.
`cell RDMA reads issued`	The total number of RDMA reads issued to retrieve data from XRMEM cache. An RDMA read is usually issued after each successful hash table probe.

Statistic	Description

`cell RDMA probe failures - hash table buffer allocation failed`
`cell RDMA probe failures - IPCDAT metadata allocation failed`
`cell RDMA probe failures - IPCDAT errors`
	The total number of RDMA hash table probes that failed. Each statistic covers a specific failure reason.
	The sum of these failures accounts for most of the difference between `cell RDMA read hash table probes` and `cell RDMA reads issued`.
	Some RDMA hash table probe failures are normal while XRMEM cache is being initialized. However, frequent failures or a large number of failures may indicate a problem requiring further investigation.

`cell RDMA read failures - lease expired`
`cell RDMA read failures - client registration errors`
	The total number of RDMA reads that failed. Each statistic covers a specific failure reason.
	The sum of these failures accounts for most of the difference between `cell RDMA reads issued` and `cell RDMA reads`.
	Normally, some RDMA read failures may occur. However, frequent failures or a large number of failures may indicate a problem requiring further investigation.

`cell RDMA reads rejected - ineligible`	The number of database reads that failed basic eligibility criteria for RDMA
`cell xrmem cache read hits`	The number of non-RDMA read requests processed by `cellsrv` resulting in a XRMEM cache hit

The following table describes database wait events that are useful for monitoring XRMEM cache. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`cell list of blocks physical read`	This wait event occurs during recovery or during buffer pre-fetching (rather than performing multiple single-block reads). It is used to monitor database blocks that must be changed as part of recovery and are read in parallel for the database. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file parallel read` for a cell.

Wait Event	Description
`cell list of blocks read request`	This is a placeholder wait event associated with `cell list of blocks physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell list of blocks physical read`.
`cell multiblock physical read`	This wait event represents the time taken to perform all the I/Os for a multi-block database read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file scattered read` for a cell.
`cell multiblock read request`	This is a placeholder wait event associated with `cell multiblock physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell multiblock physical read`.
`cell single block physical read`	This wait event represents the time taken to perform a single block database I/O, equivalent to `db file sequential read` for a cell. This wait event does not include I/O from Exadata Smart Flash Cache, I/O from XRMEM cache, or database I/O using a Remote Direct Memory Access (RDMA) read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation.

Wait Event	Description
`cell single block physical read: RDMA`	This wait event represents the time taken to perform a single block database I/O using a Remote Direct Memory Access (RDMA) read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.
`cell single block physical read: xrmem cache`	This wait event represents the time taken to perform a single block database I/O from XRMEM cache. Effective use of XRMEM cache coincides with extremely low latency for this wait event, which is typical for operations using RDMA. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data (not the cache location), which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation.
`cell single block read request`	This is a placeholder wait event associated with a single block database I/O that is visible only during the wait period. After the wait event ends, the placeholder is converted to the appropriate wait event, which is typically one of the `cell single block physical read` events.
`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` events. 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.

6.3.2.3 Monitoring XRMEM Cache Using Exadata Metrics

Exadata metrics that are related to XRMEM cache are identified in the Exadata storage server `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=XRMEMCACHE`.

Note that because reads from XRMEM cache are primarily performed using RDMA calls from Oracle Database, there are no Exadata metrics that tally XRMEM cache I/O.

Example 6-2 Displaying XRMEM Cache Metric Definitions

This example shows how to display the XRMEM cache metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
XRMEMCACHE
         XRM_BY_ALLOCATED         "Number of megabytes allocated in XRMEM
cache"
```

Note the following additional details:

- `XRM_BY_ALLOCATED` represents the number of megabytes allocated in XRMEM cache and tracks how many cachelines are used in the XRMEM cache. If the value is close to the XRMEM cache size, then the XRMEM cache is fully populated.

  This metric is also available per database (`DB_XRM_BY_ALLOCATED`), and per PDB (`PDB_XRM_BY_ALLOCATED`).

6.3.2.4 What to Look For When Monitoring XRMEM Cache

Database Working Set

Performance is maximized if the entire database working set fits into XRMEM cache. However, the XRMEM cache is smaller than Exadata Smart Flash Cache and it is likely that the database working set, or even the most frequently accessed portion of it, might not reside in XRMEM cache.

If the size of the working set increases you may observe more reads being satisfied by Exadata Smart Flash Cache or hard disk instead of XRMEM cache. This is magnified when access patterns on the data are more random. Consequently, you may observe increased read latencies, specifically in the `cell single block physical read` wait event.

Non-RDMA Reads

XRMEM cache is most effective when it is used in conjunction with RDMA. However, in various situations requests to `cellsrv` might still be satisfied using XRMEM cache. In such cases, you will see the associated reads as `cell xrmem cache read hits` rather than `cell RDMA reads`.

A non-RDMA read may occur for the following reasons:

- Based on the type of data being read, Oracle Database may not be able to perform an RDMA read. Direct reads and reads from the Oracle Database control file are examples of this. To identify the cause you may be need to correlate with other statistics or wait events. For example, significant activity against the control files is usually accompanied by the `control file sequential read` wait event, or visible in the IOStat by Filetype section of the AWR report.

- In the Oracle Database client, buffers may not be registered for RDMA. This normally occurs shortly after a client process starts or while XRMEM cache is populating.
- Inside XRMEM cache, the RDMA hash table may not contain the required metadata (or it may be marked as invalid). This normally occurs while XRMEM cache is populating.
- An RDMA read times out after exceeding the default lease time (20 ms).
- Memory limitations prevent the creation of memory structures that are required for RDMA.
- An RDMA read error occurs.

6.3.3 Monitoring PMEM Cache

> **Note:**
>
> This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see Monitoring XRMEM Cache.

Persistent Memory (PMEM) cache provides direct access to persistent memory on the storage servers using Remote Direct Memory Access (RDMA), enabling lower read latencies and faster response times. PMEM is available only in selected Exadata X8M and X9M storage server models. When database clients read from the PMEM cache, the client software performs an RDMA read of the cached data, which bypasses the storage server software and results in much lower read latencies.

PMEM cache works in conjunction with Exadata Smart Flash Cache. If available, data that is not in PMEM cache may be retrieved from Exadata Smart Flash Cache. Similarly, as data is written out of PMEM cache, it is written to Exadata Smart Flash Cache when using Write-Back mode.

Statistics from PMEM cache are slightly different when compared with other Exadata components. Because clients issue RDMA I/O directly to PMEM cache, the request does not go to `cellsrv`, so the storage server cannot account for the RDMA I/Os. For this reason, there are no cell metrics for PMEM cache I/O. Instead, Oracle Database statistics account for the I/O that is performed using RDMA.

Performance issues related to PMEM cache typically cause latency increases in the Oracle Database `cell single block physical read` wait events. However, bear in mind that Exadata Smart Flash Cache is still available to service the requests, and although requests from Exadata Smart Flash Cache generally experience higher read latencies compared to PMEM cache, the requests still benefit from the fast I/O provided by flash.

- Monitoring PMEM Cache Using AWR
- Monitoring PMEM Cache Using Database Statistics and Wait Events
- Monitoring PMEM Cache Using Exadata Metrics
- What to Look For When Monitoring PMEM Cache

6.3.3.1 Monitoring PMEM Cache Using AWR

Note:

In Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later, all references to PMEM cache are renamed to XRMEM cache (Exadata RDMA Memory Cache). However, these parts of the AWR report still correspond to PMEM cache when the underlying Oracle Exadata Storage Server contains PMEM cache.

Automatic Workload Repository (AWR) contains information relating to PMEM cache. Following are descriptions and examples of the most commonly used sections in the AWR report that contain information about PMEM cache. By reviewing these sections of the AWR report, administrators can understand the operation of PMEM cache.

Note that because storage servers cannot account for the RDMA I/Os, the Oracle Database statistics are critical in understanding the use of PMEM cache. The PMEM cache statistics in the Exadata-specific sections of the AWR report only include I/Os that as serviced by `cellsrv`, which are not RDMA I/Os.

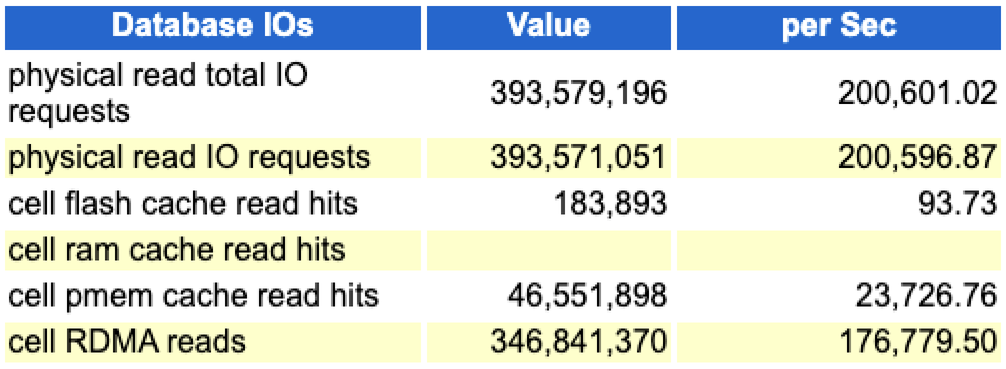
Apart from the sections described below, the AWR report also contains sections for PMEM Cache User Writes, and PMEM Cache Internal Reads. These sections are not described in detail because they relate to the use of PMEM cache in Write-Back mode, which is not generally recommended.

Database IOs

A summary of Database IOs may be found in the Single Block Reads section of the AWR report. This summary contains information about the effectiveness of PMEM cache. The Single Block Reads section is located in the AWR report under Exadata Statistics > Performance Summary.

In the following example, the overwhelming majority of all read I/O requests (approximately 88%) are serviced using RDMA reads to PMEM cache (`cell RDMA reads`), at a rate of more than 176,000 per second. Nearly all of the rest are satisfied using non-RDMA PMEM cache reads (`cell xrmem cache read hits` or `cell pmem cache read hits`), while the remaining reads are serviced by using Exadata Smart Flash Cache (`cell flash cache read hits`). Notice the massive difference in throughput for each different I/O type, which illustrates the power of PMEM cache and RDMA.

Figure 6-15 AWR Report: Database IOs



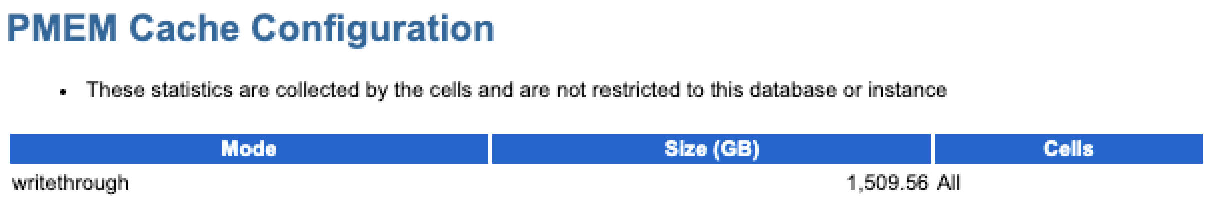
Database IOs	Value	per Sec
physical read total IO requests	393,579,196	200,601.02
physical read IO requests	393,571,051	200,596.87
cell flash cache read hits	183,893	93.73
cell ram cache read hits		
cell pmem cache read hits	46,551,898	23,726.76
cell RDMA reads	346,841,370	176,779.50

PMEM Cache Configuration and Space Usage

The PMEM Cache Configuration section contains summary information including the caching mode (Write-Through or Write-Back) and overall size. The PMEM Cache Space Usage section provides summary statistics on space usage in PMEM cache.

The following example shows PMEM cache configured in Write-Through mode on all cells, with a total size of approximately 1509 GB.

Figure 6-16 AWR Report: PMEM Cache Configuration



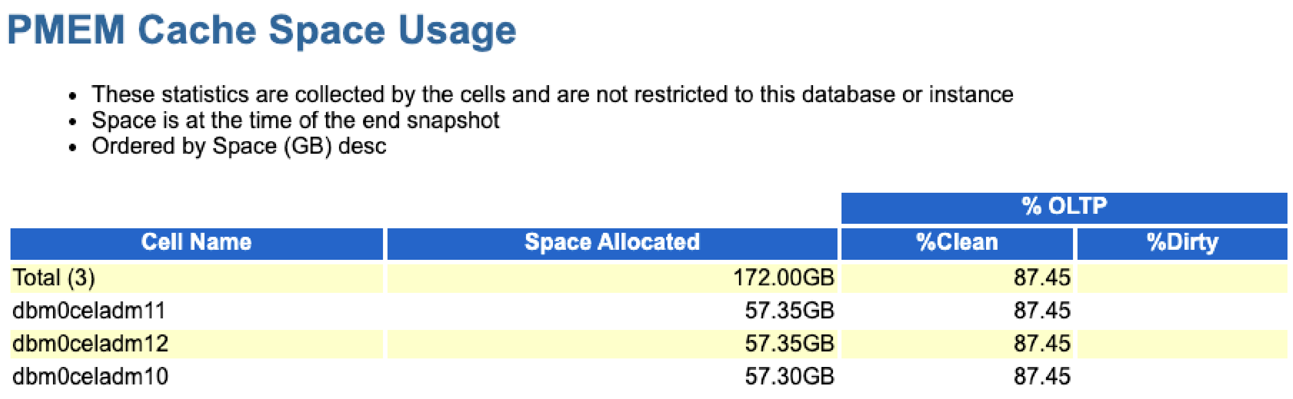
PMEM Cache Configuration

- These statistics are collected by the cells and are not restricted to this database or instance

Mode	Size (GB)	Cells
writethrough	1,509.56	All

The following example of the PMEM Cache Space Usage section shows 172 GB of PMEM cache spread evenly across 3 cells.

Figure 6-17 AWR Report: PMEM Cache Space Usage



PMEM Cache Space Usage

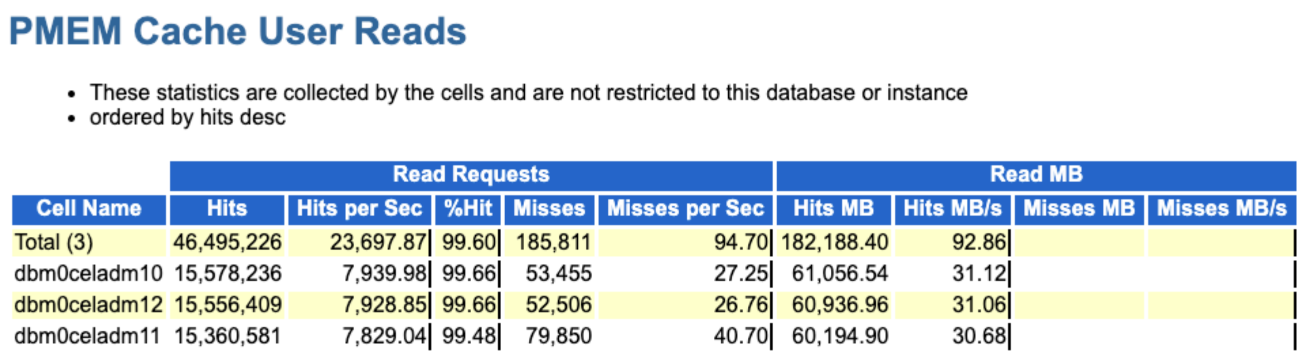
- These statistics are collected by the cells and are not restricted to this database or instance
- Space is at the time of the end snapshot
- Ordered by Space (GB) desc

		% OLTP	
Cell Name	Space Allocated	%Clean	%Dirty
Total (3)	172.00GB	87.45	
dbm0celadm11	57.35GB	87.45	
dbm0celadm12	57.35GB	87.45	
dbm0celadm10	57.30GB	87.45	

PMEM Cache User Reads

The PMEM Cache User Reads section shows information about read requests, read throughput, and read efficiency. Because storage servers cannot account for RDMA I/Os, the statistics only relate to non-RDMA reads processed by `cellsrv`. Hits are for I/Os satisfied using PMEM cache, while misses indicate that the data was not in PMEM cache.

Figure 6-18   AWR Report: PMEM Cache User Reads



PMEM Cache User Reads

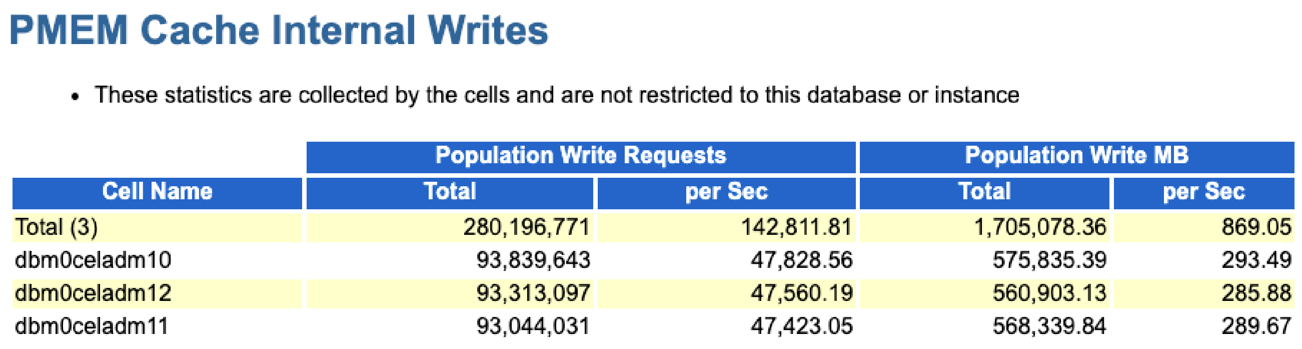
- These statistics are collected by the cells and are not restricted to this database or instance
- ordered by hits desc

	Read Requests					Read MB			
Cell Name	Hits	Hits per Sec	%Hit	Misses	Misses per Sec	Hits MB	Hits MB/s	Misses MB	Misses MB/s
Total (3)	46,495,226	23,697.87	99.60	185,811	94.70	182,188.40	92.86		
dbm0celadm10	15,578,236	7,939.98	99.66	53,455	27.25	61,056.54	31.12		
dbm0celadm12	15,556,409	7,928.85	99.66	52,506	26.76	60,936.96	31.06		
dbm0celadm11	15,360,581	7,829.04	99.48	79,850	40.70	60,194.90	30.68		

PMEM Cache Internal Writes

The PMEM Cache Internal Writes section shows writes to PMEM cache that are performed by Oracle Exadata System Software. The internal writes are I/Os that populate PMEM cache.

Figure 6-19   AWR Report: PMEM Cache Internal Writes



PMEM Cache Internal Writes

- These statistics are collected by the cells and are not restricted to this database or instance

	Population Write Requests		Population Write MB	
Cell Name	Total	per Sec	Total	per Sec
Total (3)	280,196,771	142,811.81	1,705,078.36	869.05
dbm0celadm10	93,839,643	47,828.56	575,835.39	293.49
dbm0celadm12	93,313,097	47,560.19	560,903.13	285.88
dbm0celadm11	93,044,031	47,423.05	568,339.84	289.67

6.3.3.2 Monitoring PMEM Cache Using Database Statistics and Wait Events

Note:

- The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.
- In Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later, all database statistics and wait events relating to PMEM cache are renamed to XRMEM cache (Exadata RDMA Memory Cache). However, these database statistics and wait events still correspond to PMEM cache when the underlying Oracle Exadata Storage Server contains PMEM cache.

The following table describes database statistics that are useful for monitoring PMEM cache. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`cell RDMA reads`	The number of successful PMEM cache read requests using RDMA
`cell RDMA reads eligible`	The number of database reads that meet basic eligibility criteria for RDMA
`cell RDMA read hash table probes`	The total number of RDMA hash table probes issued to determine the presence of data in PMEM cache. Each eligible read is usually associated with a hash table probe.
`cell RDMA reads issued`	The total number of RDMA reads issued to retrieve data from PMEM cache. An RDMA read is usually issued after each successful hash table probe.

`cell RDMA probe failures - hash table buffer allocation failed`
`cell RDMA probe failures - IPCDAT metadata allocation failed`
`cell RDMA probe failures - IPCDAT errors`
	The total number of RDMA hash table probes that failed. Each statistic covers a specific failure reason.
	The sum of these failures accounts for most of the difference between `cell RDMA read hash table probes` and `cell RDMA reads issued`.
	Some RDMA hash table probe failures are normal while PMEM cache is being initialized. However, frequent failures or a large number of failures may indicate a problem requiring further investigation.

`cell RDMA read failures - lease expired`
`cell RDMA read failures - client registration errors`
	The total number of RDMA reads that failed. Each statistic covers a specific failure reason.
	The sum of these failures accounts for most of the difference between `cell RDMA reads issued` and `cell RDMA reads`.
	Normally, some RDMA read failures may occur. However, frequent failures or a large number of failures may indicate a problem requiring further investigation.

`cell RDMA reads rejected - ineligible`	The number of database reads that failed basic eligibility criteria for RDMA
`cell RDMA writes`	The number of successful PMEM cache write requests using RDMA

`cell xrmem cache read hits`
`cell pmem cache read hits`
	The number of non-RDMA read requests processed by `cellsrv` resulting in a PMEM cache hit
	The statistic is named `cell xrmem cache read hits` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later. In earlier versions, it is named `cell pmem cache read hits`.

`cell xrmem cache writes`
`cell pmem cache writes`
	The number of non-RDMA write requests processed by `cellsrv` resulting in a PMEM cache write
	The statistic is named `cell xrmem cache writes` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later. In earlier versions, it is named `cell pmem cache writes`.

The following table describes database wait events that are useful for monitoring PMEM cache. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`cell list of blocks physical read`	This wait event occurs during recovery or during buffer pre-fetching (rather than performing multiple single-block reads). It is used to monitor database blocks that must be changed as part of recovery and are read in parallel for the database. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file parallel read` for a cell.
`cell list of blocks read request`	This is a placeholder wait event associated with `cell list of blocks physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell list of blocks physical read`.
`cell multiblock physical read`	This wait event represents the time taken to perform all the I/Os for a multi-block database read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file scattered read` for a cell.
`cell multiblock read request`	This is a placeholder wait event associated with `cell multiblock physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell multiblock physical read`.

Wait Event	Description
`cell single block physical read`	This wait event represents the time taken to perform a single block database I/O, equivalent to `db file sequential read` for a cell. Commencing with the May 2022 Oracle Database release updates (versions 19.15.0.0.220419, 21.6.0.0.220419, and later), this wait event no longer includes I/O from Exadata Smart Flash Cache, I/O from PMEM cache, or database I/O using a Remote Direct Memory Access (RDMA) read. With this change, another wait event (`cell single block physical read: xrmem cache` or `cell single block physical read: pmem cache`) represents the time taken to perform a single block database I/O from PMEM cache. Before the May 2022 Oracle Database release updates, effective use of PMEM cache coincides with extremely low latency for this wait event, which is typical for operations using RDMA. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation.

Wait Event	Description

`cell single block physical read: xrmem cache`
`cell single block physical read: pmem cache`
	This wait event represents the time taken to perform a single block database I/O from PMEM cache.
	Effective use of PMEM cache coincides with extremely low latency for this wait event, which is typical for operations using RDMA.
	This wait event was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.
	The wait event is named `cell single block physical read: xrmem cache` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later. In earlier versions, it is named `cell single block physical read: pmem cache`.
	In `V$SESSION`, records associated with this event include additional parameters:
	• `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`.
	• `P2` identifies the disk hash number for the grid disk that contains the data (not the cache location), which corresponds to `V$ASM_DISK.HASH_VALUE`.
	• `P3` specifies the number of bytes processed during the I/O read operation.

`cell single block read request`	This is a placeholder wait event associated with a single block database I/O that is visible only during the wait period. After the wait event ends, the placeholder is converted to the appropriate wait event, which is typically one of the `cell single block physical read` events.
`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.

6.3.3.3 Monitoring PMEM Cache Using Exadata Metrics

Exadata metrics that are related to PMEM cache are identified in the Exadata storage server `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=PMEMCACHE`.

Note that because reads from PMEM cache are primarily performed using RDMA calls from Oracle Database, there are no Exadata metrics that tally PMEM cache I/O.

Example 6-3 Displaying PMEM Cache Metric Definitions

This example shows how to display the PMEM cache metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
PMEMCACHE
         PC_BY_ALLOCATED         "Number of megabytes allocated in PMEM cache"
```

Note the following additional details:

- `PC_BY_ALLOCATED` represents the number of megabytes allocated in PMEM cache and tracks how many cachelines are used in the PMEM cache. If the value is close to the PMEM cache size, then the PMEM cache is fully populated.

  This metric is also available per database (`DB_PC_BY_ALLOCATED`), and per PDB (`PDB_PC_BY_ALLOCATED`).

6.3.3.4 What to Look For When Monitoring PMEM Cache

Database Working Set

Performance is maximized if the entire database working set fits into PMEM cache. However, the PMEM cache is much smaller than Exadata Smart Flash Cache and it is likely that the database working set, or even the most frequently accessed portion of it, might not reside in PMEM cache.

If the size of the working set increases you may observe more reads being satisfied by Exadata Smart Flash Cache or hard disk instead of PMEM cache. This is magnified when access patterns on the data are more random. Consequently, you may observe increased read latencies, specifically in the `cell single block physical read` wait event.

Non-RDMA Reads

PMEM cache is most effective when it is used in conjunction with RDMA. However, in various situations requests to `cellsrv` might still be satisfied using PMEM cache. In such cases, you will see the associated reads as `cell xrmem cache read hits` or `cell pmem cache read hits` rather than `cell RDMA reads`.

A non-RDMA read may occur for the following reasons:

- Based on the type of data being read, Oracle Database may not be able to perform an RDMA read. Direct reads and reads from the Oracle Database control file are examples of this. To identify the cause you may be need to correlate with other statistics or wait events. For example, significant activity against the control files is usually accompanied by the `control file sequential read` wait event, or visible in the IOStat by Filetype section of the AWR report.
- In the Oracle Database client, buffers may not be registered for RDMA. This normally occurs shortly after a client process starts or while PMEM cache is populating.
- Inside PMEM cache, the RDMA hash table may not contain the required metadata (or it may be marked as invalid). This normally occurs while PMEM cache is populating.
- An RDMA read times out after exceeding the default lease time (20 ms).
- Memory limitations prevent the creation of memory structures that are required for RDMA.
- An RDMA read error occurs.

6.3.4 Monitoring Exadata Smart Flash Log

Exadata Smart Flash Log reduces the average latency for performance-sensitive redo log write I/O operations, thereby eliminating performance bottlenecks that may occur due to slow redo log writes. Exadata Smart Flash Log removes latency spikes by simultaneously performing redo log writes to two media devices. The redo write is acknowledged as soon as the first write to either media device completes.

Originally, Exadata Smart Flash Log was used to perform simultaneous writes to disk and flash storage. However, Oracle Exadata System Software release 20.1 adds a further optimization, known as Smart Flash Log Write-Back, that uses Exadata Smart Flash Cache in Write-Back mode instead of disk storage.

- Monitoring Exadata Smart Flash Log Using AWR
- Monitoring Exadata Smart Flash Log Using Database Statistics and Wait Events
- Monitoring Exadata Smart Flash Log Using Exadata Metrics
- What to Look For When Monitoring Exadata Smart Flash Log

6.3.4.1 Monitoring Exadata Smart Flash Log Using AWR

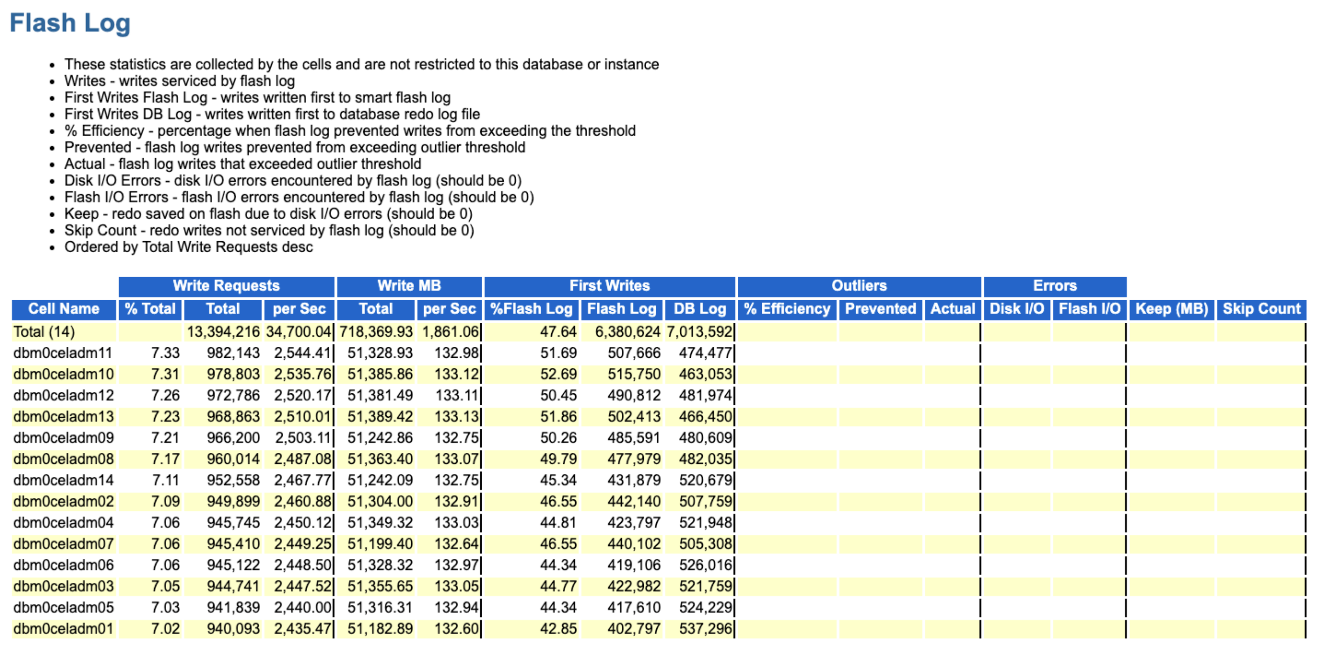
Following are descriptions and examples of the Flash Log Statistics sections in the AWR report. By reviewing these sections of the AWR report, administrators can understand the operation of Exadata Smart Flash Log.

Flash Log

The Flash Log section contains summary statistics about the operation of Exadata Smart Flash Log, including the number and of writes to Exadata Smart Flash Log, the number outliers prevented by Exadata Smart Flash Log, and the number of times when Exadata Smart Flash Log is skipped.

The following show an example of the Flash Log section. As shown in the example, when Exadata Smart Flash Log operates optimally, I/O is evenly distributed across all of the cells and there are no skips.

Figure 6-20 AWR Report: Flash Log



Flash Log

- These statistics are collected by the cells and are not restricted to this database or instance
- Writes - writes serviced by flash log
- First Writes Flash Log - writes written first to smart flash log
- First Writes DB Log - writes written first to database redo log file
- % Efficiency - percentage when flash log prevented writes from exceeding the threshold
- Prevented - flash log writes prevented from exceeding outlier threshold
- Actual - flash log writes that exceeded outlier threshold
- Disk I/O Errors - disk I/O errors encountered by flash log (should be 0)
- Flash I/O Errors - flash I/O errors encountered by flash log (should be 0)
- Keep - redo saved on flash due to disk I/O errors (should be 0)
- Skip Count - redo writes not serviced by flash log (should be 0)
- Ordered by Total Write Requests desc

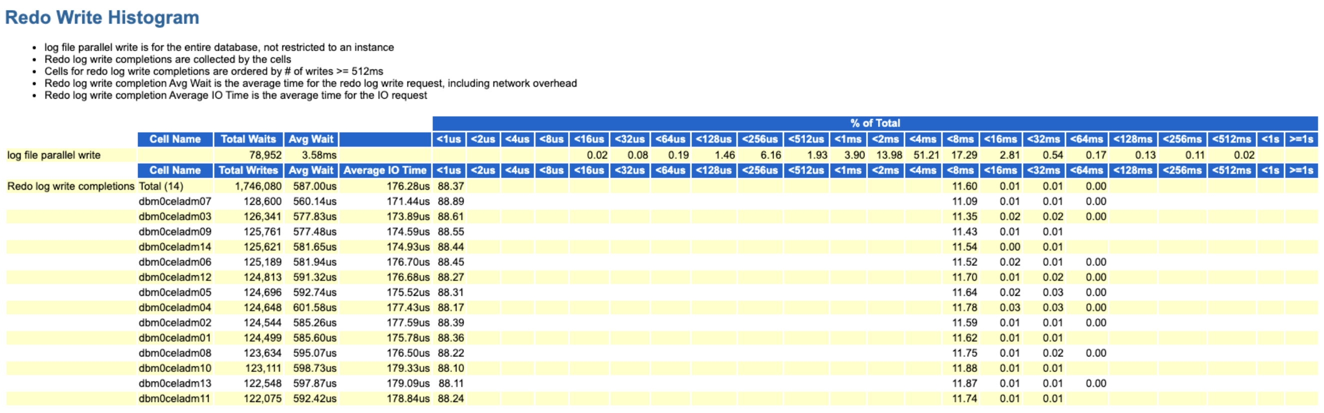
	Write Requests			Write MB		First Writes			Outliers			Errors			
Cell Name	% Total	Total	per Sec	Total	per Sec	%Flash Log	Flash Log	DB Log	% Efficiency	Prevented	Actual	Disk I/O	Flash I/O	Keep (MB)	Skip Count
Total (14)		13,394,216	34,700.04	718,369.93	1,861.06	47.64	6,380,624	7,013,592							
dbm0celadm11	7.33	982,143	2,544.41	51,328.93	132.98	51.69	507,666	474,477							
dbm0celadm10	7.31	978,803	2,535.76	51,385.86	133.12	52.69	515,750	463,053							
dbm0celadm12	7.26	972,786	2,520.17	51,381.49	133.11	50.45	490,812	481,974							
dbm0celadm13	7.23	968,863	2,510.01	51,389.42	133.13	51.86	502,413	466,450							
dbm0celadm09	7.21	966,200	2,503.11	51,242.86	132.75	50.26	485,591	480,609							
dbm0celadm08	7.17	960,014	2,487.08	51,363.40	133.07	49.79	477,979	482,035							
dbm0celadm14	7.11	952,558	2,467.77	51,242.09	132.75	45.34	431,879	520,679							
dbm0celadm02	7.09	949,899	2,460.88	51,304.00	132.91	46.55	442,140	507,759							
dbm0celadm04	7.06	945,745	2,450.12	51,349.32	133.03	44.81	423,797	521,948							
dbm0celadm07	7.06	945,410	2,449.25	51,199.40	132.64	46.55	440,102	505,308							
dbm0celadm06	7.06	945,122	2,448.50	51,328.32	132.97	44.34	419,106	526,016							
dbm0celadm03	7.05	944,741	2,447.52	51,355.65	133.05	44.77	422,982	521,759							
dbm0celadm05	7.03	941,839	2,440.00	51,316.31	132.94	44.34	417,610	524,229							
dbm0celadm01	7.02	940,093	2,435.47	51,182.89	132.60	42.85	402,797	537,296							

If the Skip Count is greater than zero, then the Flash Log Skip Details section contains the reasons for skipping Exadata Smart Flash Log.

Redo Write Histogram

The Redo Write Histogram section provides histograms that show the `log file parallel write` latency from the database, along with the `redo write request completion` latency from the storage servers. By comparing the database and cell latency histograms, you can determine if the high latency outliers are related to processing bottlenecks on the storage servers.

Figure 6-21 AWR Report: Redo Write Histogram



Redo Write Histogram

- log file parallel write is for the entire database, not restricted to an instance
- Redo log write completions are collected by the cells
- Cells for redo log write completions are ordered by # of writes >= 512ms
- Redo log write completion Avg Wait is the average time for the redo log write request, including network overhead
- Redo log write completion Average IO Time is the average time for the IO request

					% of Total																					
	Cell Name	Total Waits	Avg Wait		<1us	<2us	<4us	<8us	<16us	<32us	<64us	<128us	<256us	<512us	<1ms	<2ms	<4ms	<8ms	<16ms	<32ms	<64ms	<128ms	<256ms	<512ms	<1s	>=1s
log file parallel write		78,952	3.58ms						0.02	0.08	0.19	1.46	6.16	1.93	3.90	13.98	51.21	17.29	2.81	0.54	0.17	0.13	0.11	0.02		
	Cell Name	Total Writes	Avg Wait	Average IO Time	<1us	<2us	<4us	<8us	<16us	<32us	<64us	<128us	<256us	<512us	<1ms	<2ms	<4ms	<8ms	<16ms	<32ms	<64ms	<128ms	<256ms	<512ms	<1s	>=1s
Redo log write completions	Total (14)	1,746,080	587.00us	176.28us	88.37													11.60	0.01	0.01	0.00					
	dbm0celadm07	128,600	560.14us	171.44us	88.89													11.09	0.01	0.01	0.00					
	dbm0celadm03	126,341	577.83us	173.89us	88.61													11.35	0.02	0.02	0.00					
	dbm0celadm09	125,761	577.48us	174.59us	88.55													11.43	0.01	0.01						
	dbm0celadm14	125,621	581.65us	174.93us	88.44													11.54	0.00	0.01						
	dbm0celadm06	125,189	581.94us	176.70us	88.45													11.52	0.02	0.01	0.00					
	dbm0celadm12	124,813	591.32us	176.68us	88.27													11.70	0.01	0.02	0.00					
	dbm0celadm05	124,696	592.74us	175.52us	88.31													11.64	0.02	0.03	0.00					
	dbm0celadm04	124,648	601.58us	177.43us	88.17													11.78	0.03	0.03	0.00					
	dbm0celadm02	124,544	585.26us	177.59us	88.39													11.59	0.01	0.01	0.00					
	dbm0celadm01	124,499	585.60us	175.78us	88.36													11.62	0.01	0.01						
	dbm0celadm08	123,634	595.07us	176.50us	88.22													11.75	0.01	0.02	0.00					
	dbm0celadm10	123,111	598.73us	179.33us	88.10													11.88	0.01	0.01						
	dbm0celadm13	122,548	597.87us	179.09us	88.11													11.87	0.01	0.01	0.00					
	dbm0celadm11	122,075	592.42us	178.84us	88.24													11.74	0.01	0.01						

6.3.4.2 Monitoring Exadata Smart Flash Log Using Database Statistics and Wait Events

The following table describes various database statistics that are useful for monitoring redo write and Exadata Smart Flash Log performance. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	**Description**

`redo write size count (4KB)`
`redo write size count (8KB)`
`redo write size count (16KB)`
`redo write size count (32KB)`
`redo write size count (64KB)`
`redo write size count (128KB)`
`redo write size count (256KB)`
`redo write size count (512KB)`
`redo write size count (1024KB)`
`redo write size count (inf)`
	Number of redo writes where the redo write size is smaller than the size indicated in parenthesis. For example, `redo write size count (4KB)` includes redo writes smaller than 4 KB in size, and `redo write size count (inf)` includes redo writes over 1 MB in size.

`redo writes`	Total number of writes to the redo log files by the Oracle Database log writer (LGWR) process(es).

The following table describes database wait events that are useful for monitoring redo write and Exadata Smart Flash Log performance. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	**Description**
`log file parallel write`	The Oracle Database log writer (LGWR) process waits on this event when it is waiting for the completion of writes to the redo log file. Efficient use of Exadata Smart Flash Log is indicated by comparatively lower time waiting on this event.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.4.3 Monitoring Exadata Smart Flash Log Using Exadata Metrics

Exadata metrics that are related to Exadata Smart Flash Log provide information about flash log utilization, such as the number of megabytes written per second. Flash Log metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=FLASHLOG`.

Example 6-4   Displaying Flash Log Metric Definitions

This example shows how to display the flash log metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
FLASHLOG
         FL_ACTUAL_OUTLIERS              "The number of times redo writes to
flash and disk both exceeded the outlier threshold"
         FL_BY_KEEP                      "The amount of  redo data saved on
flash due to disk I/O errors"
         FL_DISK_FIRST                   "The number of times redo writes
first completed to disk"
         FL_DISK_IO_ERRS                 "The number of disk I/O errors
encountered by Smart Flash Logging"
         FL_EFFICIENCY_PERCENTAGE        "The efficiency of Smart Flash
Logging expressed as a percentage"
         FL_EFFICIENCY_PERCENTAGE_HOUR   "The efficiency of Smart Flash
Logging over the last hour expressed as a percentage"
         FL_FLASH_FIRST                  "The number of times redo writes
first completed to flash"
         FL_FLASH_IO_ERRS                "The number of flash I/O errors
encountered by Smart Flash Logging"
         FL_FLASH_ONLY_OUTLIERS          "The number of times redo writes to
flash exceeded the outlier threshold"
         FL_IO_DB_BY_W                   "The number of MB written to hard
disk by Smart Flash Logging"
         FL_IO_DB_BY_W_SEC               "The rate which is the number of MB
per second written to hard disk by Smart Flash Logging"
         FL_IO_FL_BY_W                   "The number of MB written to flash
by Smart Flash Logging"
         FL_IO_FL_BY_W_SEC               "The rate which is the number of MB
per second written to flash by Smart Flash Logging"
         FL_IO_TM_W                      "Cumulative latency of all redo log
writes"
         FL_IO_TM_W_RQ                   "Average latency of all redo log
writes"
         FL_IO_W                         "The number of writes serviced by
Smart Flash Logging"
         FL_IO_W_SKIP_BUSY               "The number of redo writes that
could not be serviced by Smart Flash Logging because too much data had not
yet been written to disk"
         FL_IO_W_SKIP_BUSY_MIN           "The number of redo writes during
the last minute that could not be serviced by Smart Flash Logging because too
much data had not yet been written to disk"
         FL_IO_W_SKIP_DISABLED_GD        "The number of redo writes that
could not be serviced by Smart Flash Logging because it was disabled for the
redo log's grid disk"
         FL_IO_W_SKIP_IORM_LIMIT         "The number of redo writes that
could not be serviced by Smart Flash Logging because the IORM limit had been
reached for the redo log's grid disk"
         FL_IO_W_SKIP_IORM_PLAN          "The number of redo writes that
could not be serviced by Smart Flash Logging because it was disabled by the
IORM plan"
         FL_IO_W_SKIP_LARGE              "The number of large redo writes
that could not be serviced by Smart Flash Logging because the size of the
```

```
data was larger than the amount of available space on any flash disk"
         FL_IO_W_SKIP_LOG_ON_FAST_DEV    "The number of redo writes that
bypassed Smart Flash Logging because the redo log resides on a fast device"
         FL_IO_W_SKIP_NO_BUFFER          "The number of redo writes that
could not be serviced by Smart Flash Logging because of lack of available
buffers"
         FL_IO_W_SKIP_NO_FL_DISKS        "The number of redo writes that
could not be serviced by Smart Flash Logging because there were no available
Flash Log disks"
         FL_PREVENTED_OUTLIERS           "The number of times redo writes to
disk exceeded the outlier threshold; these would have been outliers had it
not been for Smart Flash Logging"
         FL_RQ_TM_W                      "Cumulative latency of all redo log
write requests (includes network and other processing overhead)"
         FL_RQ_TM_W_RQ                   "Average latency of all redo log
write requests"
         FL_RQ_W                         "The number of redo log write
requests serviced (includes requests which were not handled by Smart Flash
Logging)"
         FL_SKIP_OUTLIERS                "The number of times redo writes to
disk exceeded the outlier threshold when Smart Flash Logging was not used"
```

Note the following additional details:

- `FL_RQ_TM_W` tracks the cumulative redo log write request latency, which includes networking and other overhead. To determine the overhead component, get the latency overhead due to factors such as network and processing, you can use subtract `FL_IO_TM_W` from `FL_RQ_TM_W`.
- `FL_SKIP_OUTLIERS` tracks the number of outliers when a redo log write skips using the flash log. An outlier is a redo log write that exceeds 0.5 seconds.

  Additionally, metrics starting with `FL_IO_W_SKIP` track other situations where the flash log is not used.
- For simultaneous writes to disk and flash storage, the disk controller write cache can absorb some writes quicker than flash. Consequently, it is normal for a significant proportion of redo log write operations to complete to disk before flash. In some cases, the value of `FL_DISK_FIRST` can exceed the value of `FL_FLASH_FIRST`. However, this does not mean that the Exadata Smart Flash Logging feature is ineffective or unnecessary.

6.3.4.4 What to Look For When Monitoring Exadata Smart Flash Log

General Performance

Performance issues related to redo logging typically exhibit high latency for the `log file sync` wait event in the Oracle Database user foreground processes, with corresponding high latency for `log file parallel write` in the Oracle Database log writer (LGWR) process. Because of the performance-critical nature of redo log writes, occasional long latencies for `log file parallel write` may cause fluctuations in database performance, even if the average `log file parallel write` wait time is acceptable.

If any of these are occurring, then it may be indicative of an issue with Exadata Smart Flash Log performance.

Redo Write Histograms

The `log file parallel write` wait event indicates the amount of time that the database waits on a redo log write. The `log file parallel write` histogram shows the number of times the redo write completed within a specified time range. Similarly, the `redo log write completions` statistic indicates the amount of time that the storage server spends processing redo write requests, and the `redo log write completions` histogram shows the number of times the redo write request was completed within a specified time range. Both histograms are shown in the Redo Write Histogram section of the AWR report.

A histogram with a significant number of occasional long latencies is said to have a long tail. When both of the histograms in the Redo Write Histogram section of the AWR report have long tails, then this is an indication of slow write times on the storage server, which would warrant further investigation of the other I/O performance statistics. See Monitoring Cell Disk I/O.

If the `log file parallel write` histogram has a long tail that is not present in the `redo log write completions` histogram, then the cause is generally not in the storage server, but rather something else in the I/O path, such as bottlenecks in the network or contention for compute node CPU.

Skipping

Increased redo write latencies can also occur when the Exadata Smart Flash Log is skipped, and the redo write goes only to disk. Both the AWR report and the storage server metrics show the number of redo log writes that skipped Exadata Smart Flash Log. Skipping may occur when Exadata Smart Flash Log contains too much data that has not yet been written to disk.

There are a few factors that can cause redo writes to skip Exadata Smart Flash Log:

- Flash disks with high write latencies.

  This can be observed in various IO Latency tables located in the Exadata Resource Statistics section of the AWR report, and in the Exadata `CELLDISK` metrics. This can also be identified by checking the `FL_FLASH_ONLY_OUTLIERS` metric. If the metric value is high, this indicates a flash disk performance issue.

- Hard disks with high latencies or high utilization.

  Prior to Oracle Exadata System Software release 20.1, redo log writes are written to both Exadata Smart Flash Log and hard disk. If the hard disks experience high latencies or high utilization, redo log write performance can be impacted.

  This can be observed in various IO Latency tables located in the Exadata Resource Statistics section of the AWR report, and in the Exadata `CELLDISK` metrics. This can also be identified by checking the Outliers columns in the Flash Log section of the AWR report, or the `FL_PREVENTED_OUTLIERS` storage server metric. A large number of prevented outliers may indicate that the hard disk writes are taking a long time.

  In this case, although Exadata Smart Flash Log prevents outliers, overall throughput may be limited due to the queue of redo log data that must be written to disk.

  Oracle Exadata System Software release 20.1 adds a further optimization, known as Smart Flash Log Write-Back, that uses Exadata Smart Flash Cache in Write-Back mode instead of disk storage, thereby eliminating the hard disks as a potential performance bottleneck. Depending on the system workload, this feature can improve overall log write throughput by up to 250%.

6.3.5 Monitoring XRMEM Log

Note:

Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies. In this release, the persistent memory commit accelerator, previously known as PMEM log, is now called XRMEM Log.

Redo log writes are critical database operations and need to complete in a timely manner to prevent load spikes or stalls. Exadata Smart Flash Log is designed to prevent redo write latency outliers. XRMEM log helps to further reduce redo log write latency by using Persistent Memory (PMEM) and Remote Direct Memory Access (RDMA). XRMEM Log is available only in selected Exadata X8M and X9M storage server models.

With XRMEM Log, database clients send I/O buffers directly to PMEM on the storage servers using RDMA, thereby reducing transport latency. The cell server (`cellsrv`) then writes the redo to Exadata Smart Flash Log (if enabled) and disk at a later time.

Reduced redo log write latency improves OLTP performance, resulting in higher transaction throughput. In cases where XRMEM log is bypassed, Exadata Smart Flash Log can still be used.

- Monitoring XRMEM Log Using Database Statistics and Wait Events
- What to Look For When Monitoring XRMEM Log

6.3.5.1 Monitoring XRMEM Log Using Database Statistics and Wait Events

The following table describes database statistics that are useful for monitoring XRMEM log performance. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description

`cell xrmem log writes`
`cell pmem log writes`
	The number of redo log write requests that used XRMEM log
	The statistic is named `cell xrmem log writes` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later. In earlier versions, it is named `cell pmem log writes`.

The following table describes database wait events that are useful for monitoring XRMEM log performance. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`log file parallel write`	The Oracle Database log writer (LGWR) process waits on this event when it is waiting for the completion of writes to the redo log file. Efficient use of XRMEM log is indicated by comparatively lower time waiting on this event.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.5.2 What to Look For When Monitoring XRMEM Log

General Performance

Performance issues related to redo logging typically exhibit high latency for the `log file sync` wait event in the Oracle Database user and foreground processes, with corresponding high latency for `log file parallel write` in the Oracle Database log writer (LGWR) process. Because of the performance-critical nature of redo log writes, occasional long latencies for `log file parallel write` may cause fluctuations in database performance, even if the average `log file parallel write` wait time is acceptable.

If any of these are occurring, then it may be indicative of an issue with XRMEM log performance.

Bypassing XRMEM Log

Increased redo write latencies when using XRMEM log can occur when XRMEM log is bypassed. When XRMEM log is bypassed, the request is sent to the `cellsrv`, and Exadata Smart Flash Log is still used (if available). However, when the bypass request is sent, it has to ensure that there is no conflict with a previous XRMEM log request. This conflict checking, which requires scanning XRMEM log, makes bypass writes more expensive to process, and can result in higher than expected redo log write latencies.

There are several possible causes that result in a small number of XRMEM log bypasses. Under normal circumstances, the number of bypasses should be substantially less than 1% of the total number of XRMEM log requests. A high number of XRMEM log bypasses is likely to be a symptom of another problem, such as congestion on the RoCE Network Fabric.

6.3.6 Monitoring PMEM Log

Note:

This topic applies only to Oracle Exadata System Software releases before 23.1.0. Otherwise, see Monitoring XRMEM Log.

Redo log writes are critical database operations and need to complete in a timely manner to prevent load spikes or stalls. Exadata Smart Flash Log is designed to prevent redo write latency outliers. PMEM Log helps to further reduce redo log write latency by using Persistent Memory (PMEM) and Remote Direct Memory Access (RDMA). PMEM Log is available only in selected Exadata X8M and X9M storage server models.

With PMEM Log, database clients send I/O buffers directly to PMEM on the storage servers using RDMA, thereby reducing transport latency. The cell server (`cellsrv`) then writes the redo to Exadata Smart Flash Log (if enabled) and disk at a later time.

Reduced redo log write latency improves OLTP performance, resulting in higher transaction throughput. In cases where PMEM Log is bypassed, Exadata Smart Flash Log can still be used.

- Monitoring PMEM Log Using Database Statistics and Wait Events
- What to Look For When Monitoring PMEM Log

6.3.6.1 Monitoring PMEM Log Using Database Statistics and Wait Events

The following table describes database statistics that are useful for monitoring PMEM Log performance. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description

`cell xrmem log writes`
`cell pmem log writes`
	The number of redo log write requests that used PMEM log
	The statistic is named `cell xrmem log writes` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later. In earlier versions, it is named `cell pmem log writes`.

The following table describes database wait events that are useful for monitoring PMEM Log performance. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`log file parallel write`	The Oracle Database log writer (LGWR) process waits on this event when it is waiting for the completion of writes to the redo log file. Efficient use of PMEM Log is indicated by comparatively lower time waiting on this event.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.6.2 What to Look For When Monitoring PMEM Log

General Performance

Performance issues related to redo logging typically exhibit high latency for the `log file sync` wait event in the Oracle Database user and foreground processes, with corresponding high latency for `log file parallel write` in the Oracle Database log writer (LGWR) process. Because of the performance-critical nature of redo log writes, occasional long latencies for `log file parallel write` may cause fluctuations in database performance, even if the average `log file parallel write` wait time is acceptable.

If any of these are occurring, then it may be indicative of an issue with PMEM Log performance.

Bypassing PMEM Log

Increased redo write latencies when using PMEM Log can occur when PMEM Log is bypassed. When PMEM Log is bypassed, the request is sent to the `cellsrv`, and Exadata Smart Flash Log is still used (if available). However, when the bypass request is sent, it has to ensure that there is no conflict with a previous PMEM Log request. This conflict checking, which requires scanning PMEM Log, makes bypass writes more expensive to process, and can result in higher than expected redo log write latencies.

There are several possible causes that result in a small number of PMEM Log bypasses. Under normal circumstances, the number of bypasses should be substantially less than 1% of the total number of PMEM Log requests. A high number of PMEM Log bypasses is likely to be a symptom of another problem, such as congestion on the RoCE Network Fabric.

6.3.7 Monitoring Smart I/O

Exadata Smart Scan offloads data search and retrieval processing to the Exadata storage servers. It is able to evaluate database predicates on the storage server to optimize efficiency and performance, especially for large queries and certain classes of bulk data processing.

Smart Scan also uses storage indexes, which are maintained automatically on the storage servers, to further optimize filtering by eliminating unnecessary I/O.

Columnar cache is another smart I/O optimization. The columnar cache is a section of Exadata Smart Flash Cache that stores data in columnar format. The columnar cache is automatically populated and maintained by the storage server when a Smart Scan is performed.

- Monitoring Smart I/O Using AWR
- Monitoring Smart I/O Using Database Statistics and Wait Events
- Monitoring Smart I/O Using SQL Monitor
- Monitoring Smart I/O Using SQL Explain Plan
- Monitoring Smart I/O Using Exadata Metrics
- What to Look For When Monitoring Smart I/O

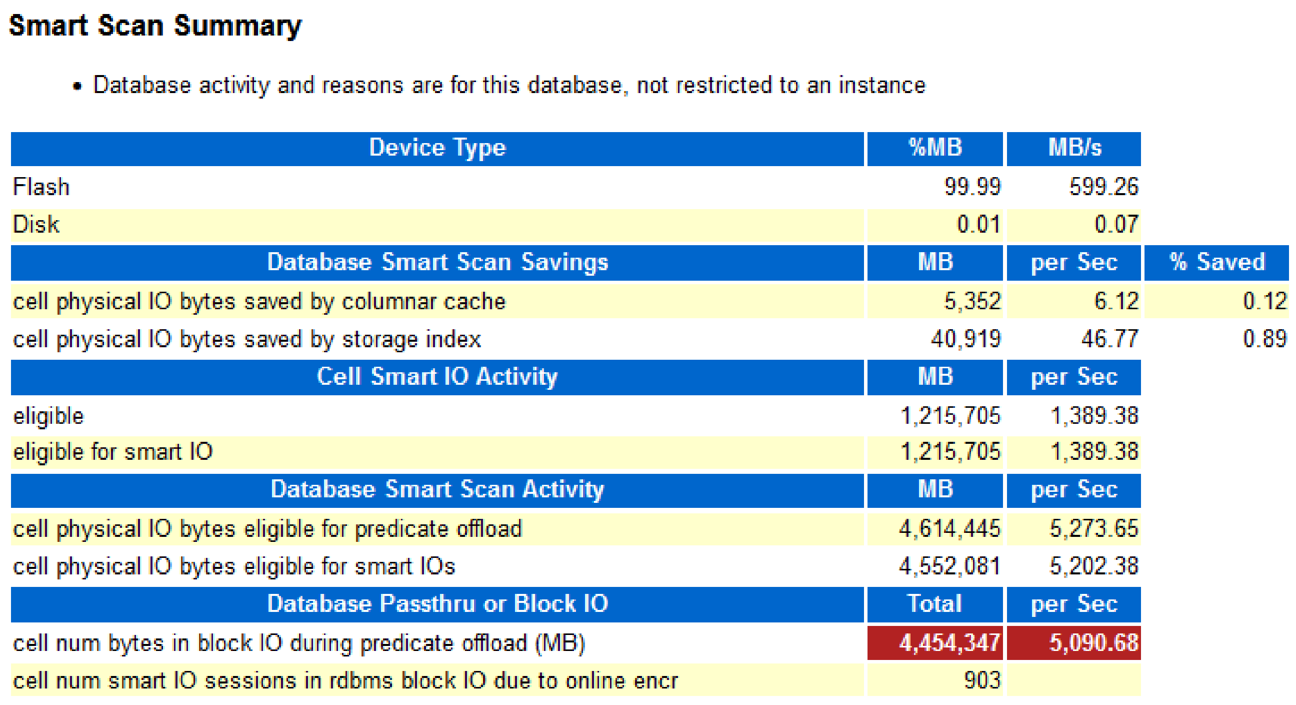
Related Topics

- Offloading Data Search and Retrieval Processing
  Exadata Smart Scan offloads search and retrieval processing to the storage servers.
- Storage Index
  Oracle Exadata Storage Servers maintain a storage index which contains a summary of the data distribution on the disk.
- In-Memory Columnar Format Support
  In an Oracle Exadata environment, the data is automatically stored in In-Memory columnar format in the flash cache when it will improve performance.

6.3.7.1 Monitoring Smart I/O Using AWR

The Smart Scan Summary is part of the Performance Summary section of the AWR report. This area contains a high-level summary of key statistics associated with smart I/O. The following example highlights a situation where some activity reverted to block I/O because of ongoing online encryption.

Figure 6-22 AWR Report: Performance Summary - Smart Scan Summary



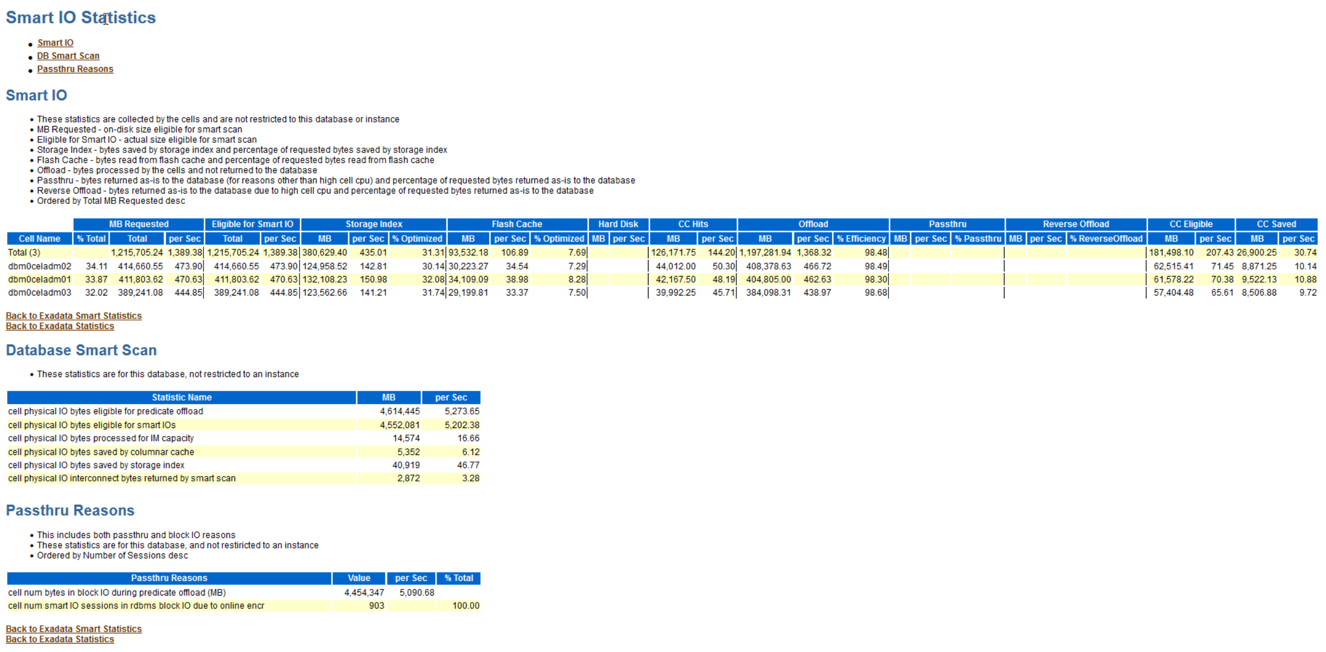
Smart Scan Summary

- Database activity and reasons are for this database, not restricted to an instance

Device Type	%MB	MB/s	
Flash	99.99	599.26	
Disk	0.01	0.07	
Database Smart Scan Savings	**MB**	**per Sec**	**% Saved**
cell physical IO bytes saved by columnar cache	5,352	6.12	0.12
cell physical IO bytes saved by storage index	40,919	46.77	0.89
Cell Smart IO Activity	**MB**	**per Sec**	
eligible	1,215,705	1,389.38	
eligible for smart IO	1,215,705	1,389.38	
Database Smart Scan Activity	**MB**	**per Sec**	
cell physical IO bytes eligible for predicate offload	4,614,445	5,273.65	
cell physical IO bytes eligible for smart IOs	4,552,081	5,202.38	
Database Passthru or Block IO	**Total**	**per Sec**	
cell num bytes in block IO during predicate offload (MB)	4,454,347	5,090.68	
cell num smart IO sessions in rdbms block IO due to online encr	903		

The Smart IO Statistics section of the AWR report contains more detailed summarized statistical information for smart I/O operations on the whole system and on each storage server. In particular, it tracks the amount of I/O that is eligible for optimization using Smart Scan, the savings from using storage indexes and columnar cache, the amount of data in Exadata Smart Flash Cache, and the amount of data filtered by the storage servers without transportation back to the database server. It also shows associated database statistics and more detailed information about situations where the storage servers are unable to perform predicate evaluation (also known as 'passthru').

Figure 6-23   AWR Report: Smart IO Statistics



Smart IO Statistics

- Smart IO
- DB Smart Scan
- Passthru Reasons

Smart IO

- These statistics are collected by the cells and are not restricted to this database or instance
- MB Requested - on-disk size eligible for smart scan
- Eligible for Smart IO - actual size eligible for smart scan
- Storage Index - bytes saved by storage index and percentage of requested bytes saved by storage index
- Flash Cache - bytes read from flash cache and percentage of requested bytes read from flash cache
- Offload - bytes processed by the cells and not returned to the database
- Passthru - bytes returned as-is to the database (for reasons other than high cell cpu) and percentage of requested bytes returned as-is to the database
- Reverse Offload - bytes returned as-is to the database due to high cell cpu and percentage of requested bytes returned as-is to the database
- Ordered by Total MB Requested desc

	MB Requested			Eligible for Smart IO		Storage Index			Flash Cache			Hard Disk		CC Hits		Offload			Passthru			Reverse Offload			CC Eligible		CC Saved	
Cell Name	% Total	Total	per Sec	Total	per Sec	MB	per Sec	% Optimized	MB	per Sec	% Optimized	MB	per Sec	MB	per Sec	MB	per Sec	% Efficiency	MB	per Sec	% Passthru	MB	per Sec	% ReverseOffload	MB	per Sec	MB	per Sec
Total (3)		1,215,705.24	1,389.38	1,215,705.24	1,389.38	380,629.40	435.01	31.31	93,532.18	106.89	7.69			126,171.75	144.20	1,197,281.94	1,368.32	98.48							181,498.10	207.43	26,900.25	30.74
dbm0celadm02	34.11	414,660.55	473.90	414,660.55	473.90	124,958.52	142.81	30.14	30,223.27	34.54	7.29			44,012.00	50.30	408,378.63	466.72	98.49							62,515.41	71.45	8,871.25	10.14
dbm0celadm01	33.87	411,803.62	470.63	411,803.62	470.63	132,108.23	150.98	32.08	34,109.09	38.98	8.28			42,167.50	48.19	404,805.00	462.63	98.30							61,578.22	70.38	9,522.13	10.88
dbm0celadm03	32.02	389,241.08	444.85	389,241.08	444.85	123,562.66	141.21	31.74	29,199.81	33.37	7.50			39,992.25	45.71	384,098.31	438.97	98.68							57,404.48	65.61	8,506.88	9.72

Back to Exadata Smart Statistics
Back to Exadata Statistics

Database Smart Scan

- These statistics are for this database, not restricted to an instance

Statistic Name	MB	per Sec
cell physical IO bytes eligible for predicate offload	4,614,445	5,273.65
cell physical IO bytes eligible for smart IOs	4,552,081	5,202.38
cell physical IO bytes processed for IM capacity	14,574	16.66
cell physical IO bytes saved by columnar cache	5,352	6.12
cell physical IO bytes saved by storage index	40,919	46.77
cell physical IO interconnect bytes returned by smart scan	2,872	3.28

Passthru Reasons

- This includes both passthru and block IO reasons
- These statistics are for this database, and not restiricted to an instance
- Ordered by Number of Sessions desc

Passthru Reasons	Value	per Sec	% Total
cell num bytes in block IO during predicate offload (MB)	4,454,347	5,090.68	
cell num smart IO sessions in rdbms block IO due to online encr	903		100.00

Back to Exadata Smart Statistics
Back to Exadata Statistics

Related Topics

- Monitoring Exadata Smart Flash Cache Using AWR

6.3.7.2 Monitoring Smart I/O Using Database Statistics and Wait Events

The following table describes various database statistics that are useful for monitoring smart I/O operations. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`cell IO uncompressed bytes`	The total size of uncompressed data that is processed on the cell. For operations on segments compressed using Exadata Hybrid Columnar Compression, this statistic is the size of the data after decompression.
`cell num bytes in block IO during predicate offload`	The number of bytes that were not offloaded because the client was in block I/O mode.
`cell num bytes in filter passthru due to low mem`	The number of bytes that were not offloaded and sent back to the database for processing due to a low memory on the cell.
`cell num bytes in filter passthru due to subheap size limit exc`	The number of bytes that were not offloaded and sent back to the database for processing due to a memory limit on the cell.
`cell num bytes in passthru due to quarantine`	The number of bytes that were not offloaded and sent back to the database for processing due to a quarantine on the cell.
`cell num bytes in passthru during predicate offload`	The number of bytes that were not offloaded and sent back to the database for processing.

Statistic	Description
`cell num smart IO sessions in rdbms block IO due to big payload`	The number of sessions in block I/O mode (not offloaded) due to excessively large metadata.
`cell num smart IO sessions in rdbms block IO due to no cell mem`	The number of sessions in block I/O mode (not offloaded) due to memory shortage on the storage servers.
`cell num smart IO sessions in rdbms block IO due to online encr`	The number of sessions in block I/O mode (not offloaded) due to an ongoing online encryption operation.
`cell num smart IO sessions in rdbms block IO due to open fail`	The number of sessions in block I/O mode (not offloaded) due to a failure in opening a connection to a cell.
`cell num smart IO sessions in rdbms block IO due to user`	The number of sessions in block I/O mode (not offloaded) due to a user setting.
`cell num smart IO sessions using passthru mode due to cellsrv`	The number of sessions in passthru mode (not offloaded) due to an issue with CELLSRV.
`cell num smart IO sessions using passthru mode due to timezone`	The number of sessions in passthru mode (not offloaded) due to an ongoing database timezone upgrade operation.
`cell num smart IO sessions using passthru mode due to user`	The number of sessions in passthru mode (not offloaded) due to a user setting.
`cell physical IO bytes added to storage index`	The number of bytes added to the storage index during a Smart Scan. This is an indication that the storage index is being built.
`cell physical IO bytes eligible for predicate offload`	The number of bytes on-disk eligible for predicate offload.
`cell physical IO bytes eligible for smart IOs`	The number of actual bytes eligible for predicate offload. For example, when using columnar cache, this is the size of columnar cache instead of the on-disk size.
`cell physical IO bytes processed for IM capacity`	The number of bytes read from the columnar cache in `memcompress for capacity` format.
`cell physical IO bytes processed for IM query`	The number of bytes read from the columnar cache in `memcompress for query` format.
`cell physical IO bytes processed for no memcompress`	The number of bytes read from the columnar cache in `no memcompress` format.
`cell physical IO bytes processed for XrCC`	The number of bytes read from the columnar cache on Exadata RDMA Memory (XRMEM).
`cell physical IO bytes saved by columnar cache`	The number of bytes saved by columnar cache; that is, the number of bytes of reading that was avoided.
`cell physical IO bytes saved by storage index`	The number of bytes saved by storage index.
`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 the cells. This statistic shows the benefit of optimized file creation operations.
`cell physical IO bytes saved during optimized RMAN restore`	The number of I/O bytes saved by the database host by offloading the RMAN file restore operation to the cells. This statistic shows the benefit of optimized RMAN file restore operations.

Statistic	Description
`cell physical IO bytes sent directly to DB node to balance CPU usage`	The number of I/O bytes sent back to the database server for processing due to high storage server CPU usage.
`cell physical IO interconnect bytes`	The number of I/O bytes exchanged over the interconnect between the database host and the cells.
`cell physical IO interconnect bytes returned by smart scan`	The number of I/O bytes that are returned by the cell for Smart Scan operations. It does not include bytes for other database I/O.

The following table describes database wait events that are useful for monitoring smart I/O operations. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`cell external table smart scan`	This wait event appears when the database is waiting for an external table scan on a cell. The cell hash number in the `P1` column in `V$SESSION` can help to identify a slow cell compared to the rest of the cells.
`cell smart file creation`	This wait event appears when the database is waiting for the completion of a file creation on a cell. The cell hash number in the `P1` column in `V$SESSION` can help to 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 `V$SESSION` can help to identify a slow cell compared to the rest of the cells.
`cell smart index scan`	This wait event appears when the database is waiting for an index fast full scan. The cell hash number in the `P1` column in `V$SESSION` can help to identify a slow cell when compared to the rest of the cells.
`cell smart index scan: db timezone upgrade`	The wait event appears when the cells are unable to offload because a database timezone upgrade is in progress.
`cell smart index scan: disabled by user`	The wait event appears when the cells are unable to offload due to a user setting.
`cell smart index scan: pass through`	The wait event appears when the cells are unable to offload the Smart Scan.

Wait Event	Description
`cell smart index scan request`	This is a placeholder wait event associated with `cell smart index scan`, which is only visible during the wait period. After the wait event ends, the placeholder is typically converted to `cell smart index scan`. However, to better describe the wait outcome, the placeholder may be converted to `cell smart index scan: db timezone upgrade`, `cell smart index scan: disabled by user`, or `cell smart index scan: pass through`.
`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 `V$SESSION` can help to identify a slow cell when compared to the rest of the cells.
`cell smart table scan`	This wait event appears when the database is waiting for smart scans to complete on a cell. The cell hash number in the `P1` column in `V$SESSION` can help to identify a slow cell when compared to the rest of the cells.
`cell smart table scan: db timezone upgrade`	The wait event appears when the cells are unable to offload because a database timezone upgrade is in progress.
`cell smart table scan: disabled by user`	The wait event appears when the cells are unable to offload due to a user setting.
`cell smart table scan: pass through`	The wait event appears when the cells are unable to offload the Smart Scan.
`cell smart table scan request`	This is a placeholder wait event associated with `cell smart table scan`, which is only visible during the wait period. After the wait event ends, the placeholder is typically converted to `cell smart table scan`. However, to better describe the wait outcome, the placeholder may be converted to `cell smart table scan: db timezone upgrade`, `cell smart table scan: disabled by user`, or `cell smart table scan: pass through`.

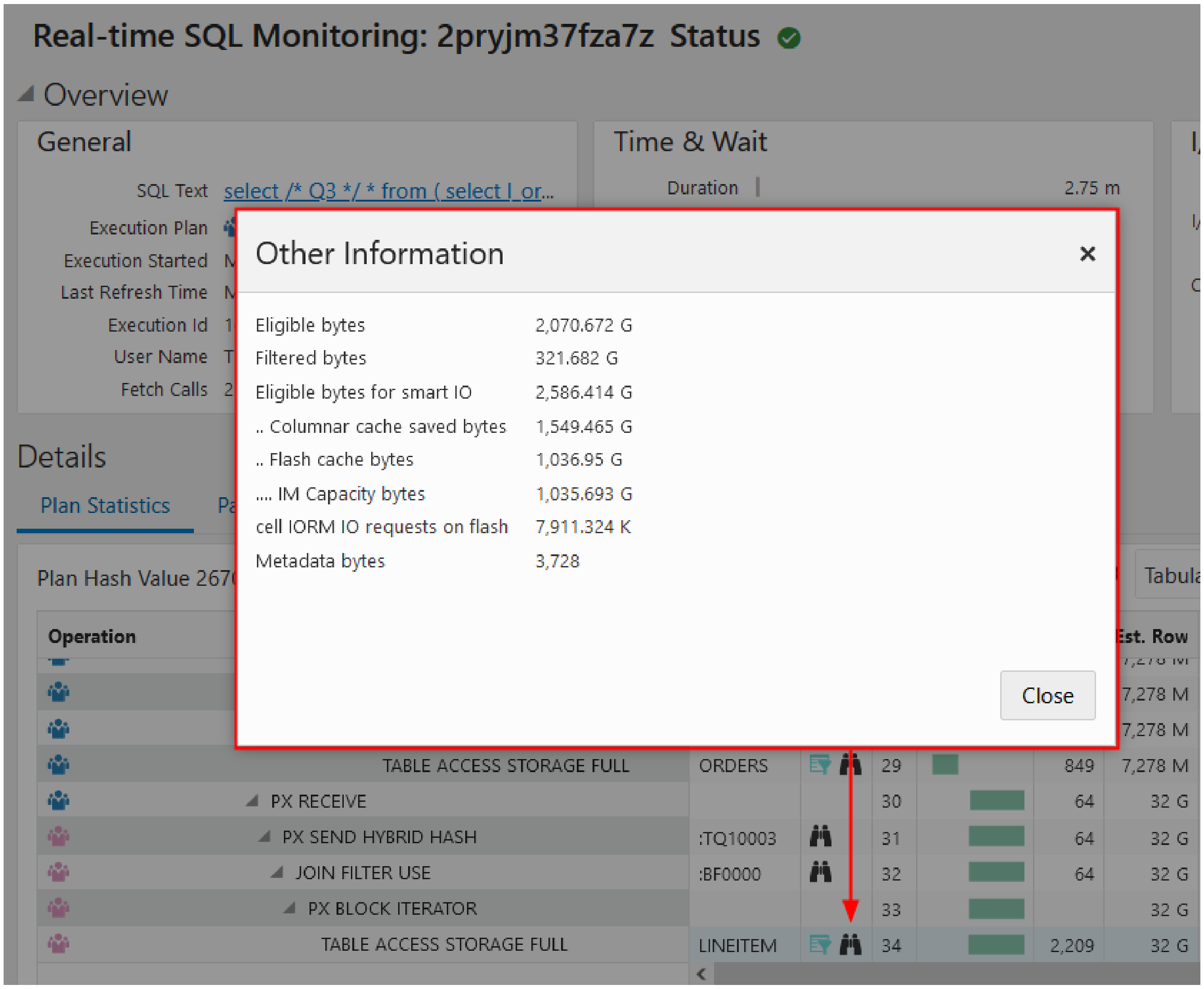
The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.7.3 Monitoring Smart I/O Using SQL Monitor

In addition to dynamic performance views and wait events, Oracle Database provides SQL monitor, which enables you to monitor the execution of individual SQL statements.

The SQL monitor report includes detailed statistics for the row sources, which includes additional information for smart I/O operations. The row source statistics can be viewed in the Enterprise Manager active SQL monitor report by clicking on the binoculars in the row source. For example:

Figure 6-24   SQL Monitor Report: Smart I/O Row Source Statistics


Real-time SQL Monitoring: 2pryjm37fza7z Status

Overview

General

SQL Text select /* Q3 */ * from (select l_or...

Execution Plan

Execution Started

Last Refresh Time

Execution Id

User Name

Fetch Calls

Time & Wait

Duration 2.75 m

Details

Plan Statistics

Plan Hash Value 267

Other Information

Eligible bytes	2,070.672 G
Filtered bytes	321.682 G
Eligible bytes for smart IO	2,586.414 G
.. Columnar cache saved bytes	1,549.465 G
.. Flash cache bytes	1,036.95 G
.... IM Capacity bytes	1,035.693 G
cell IORM IO requests on flash	7,911.324 K
Metadata bytes	3,728

Close

Operation					Est. Row
					7,278 M
					7,278 M
TABLE ACCESS STORAGE FULL	ORDERS	29		849	7,278 M
PX RECEIVE		30		64	32 G
PX SEND HYBRID HASH	:TQ10003	31		64	32 G
JOIN FILTER USE	:BF0000	32		64	32 G
PX BLOCK ITERATOR		33			32 G
TABLE ACCESS STORAGE FULL	LINEITEM	34		2,209	32 G

The following table describes various row source statistics provided by SQL monitor, which are useful for monitoring smart I/O.

Statistic	Description
`Filtered bytes`	The number of bytes returned by the cell.
`Cell passthru IO bytes`	The number of bytes that are not offloaded and sent back to the database for processing.
`Cell passthru IO bytes due to quarantine`	The number of bytes that are not offloaded and sent back to the database for processing due to a quarantine on the cell.
`Eligible bytes for smart IO`	The number of actual bytes eligible for predicate offload. For example, when using columnar cache, this is the size of columnar cache instead of the on-disk size.

Statistic	Description
`SI saved bytes`	The number of bytes saved by storage index; that is, the number of bytes that did not have to be read.
`Columnar cache saved bytes`	The number of bytes saved by columnar cache; that is, the number of bytes that did not have to be read.
`Partial flash cache and disk bytes`	The number of bytes that read from both Exadata Smart Flash Cache and disk.
`Flash cache bytes`	The number of bytes read from Exadata Smart Flash Cache.
`IM Capacity bytes`	The number of bytes read from the columnar cache in `memcompress for capacity` format.
`IM Query bytes`	The number of bytes read from the columnar cache in `memcompress for query` format.
`No memcompress bytes`	The number of bytes read from the columnar cache in `no memcompress` format.
`XRMEM Columnar Cache bytes`	The number of bytes read from the columnar cache on Exadata RDMA Memory (XRMEM).
`Bytes added to storage index`	The number of bytes added to the storage index during a Smart Scan. This is an indication that the storage index is being built.
`cell IORM IO requests on flash`	The number of physical I/O requests to flash storage.
`cell IORM wait time on flash (us)`	The amount of time (in microseconds) IORM queued the flash request. The `cell IORM wait time on flash (us)` / `cell IORM IO requests on flash` gives an indication of how much time, on average, is spent in the IORM queue.
`cell IORM IO requests on disk`	The number of physical I/O requests to disk storage.
`cell IORM wait time on disk (us)`	The amount of time (in microseconds) IORM queued the disk request. The `cell IORM wait time on disk (us)` / `cell IORM IO requests on disk` gives an indication of how much time, on average, is spent in the IORM queue.
`Block IO bytes`	The number of bytes in block I/O mode.

`Slow metadata bytes`
`Metadata bytes`
	The size of the query metadata sent from the database compute node to a cell.

`Eligible bytes`	The number of bytes on-disk eligible for predicate offload.

The availability of a specific statistic is subject to the version of Oracle Database being used.

Related Topics

- Real-Time SQL Monitoring and Real-Time Database Operations
- REPORT_SQL_MONITOR Function

6.3.7.4 Monitoring Smart I/O Using SQL Explain Plan

The SQL `EXPLAIN PLAN` command displays information about smart I/O optimizations in the SQL execution plan.

You can use the `EXPLAIN PLAN` command to identify parts of a SQL query that can be offloaded to the storage server. The database parameter `CELL_OFFLOAD_PLAN_DISPLAY` must be set to `AUTO` or `ALWAYS` for the `EXPLAIN PLAN` command to display the smart I/O optimizations in the SQL execution plan.

Example 6-5 Monitoring Smart I/O Using SQL Explain Plan

This example shows how to use the `EXPLAIN PLAN` command to display the smart I/O optimizations in the SQL execution plan.

In the query plan, the `TABLE ACCESS STORAGE FULL` operation indicates that the corresponding full table scan is offloaded to the storage server. The predicate information further describes the query predicates that are offloaded to the storage server.

```
SQL> ALTER SESSION SET CELL_OFFLOAD_PLAN_DISPLAY = ALWAYS;

Session altered.

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> 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)
```

6.3.7.5 Monitoring Smart I/O Using Exadata Metrics

Smart I/O metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=SMARTIO`.

For cumulative metrics, the metric value for a specific time period can be determined by subtracting values from different `collectionTime` periods.

Example 6-6 Displaying Smart I/O Metric Definitions

This example shows how to display the smart I/O metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
SMARTIO
         SIO_IO_EL_OF            "Cumulative number of megabytes eligible for
smart IO offload"
         SIO_IO_EL_OF_SEC        "Number of megabytes per second eligible for
smart IO offload"
         SIO_IO_OF_RE            "Cumulative number of interconnect megabytes
returned by smart IO"
         SIO_IO_OF_RE_SEC        "Number of interconnect megabytes per second
returned by smart IO"
         SIO_IO_PA_TH            "Cumulative number of megabytes of passthru
IOs by smart IO"
         SIO_IO_PA_TH_SEC        "Number of megabytes per second of passthru
IOs by smart IO"
         SIO_IO_RD_FC            "Cumulative number of megabytes read from
flash cache by smart IO"
         SIO_IO_RD_FC_HD         "Cumulative number of megabytes read from
both flash cache and hard disk by smart IO"
         SIO_IO_RD_FC_HD_SEC     "Number of megabytes per second read from
both flash cache and hard disk by smart IO"
         SIO_IO_RD_FC_SEC        "Number of megabytes per second read from
flash cache by smart IO"
         SIO_IO_RD_HD            "Cumulative number of megabytes read from
hard disk by smart IO"
         SIO_IO_RD_HD_SEC        "Number of megabytes per second read from
hard disk by smart IO"
         SIO_IO_RD_RQ_FC         "Cumulative number of read IO requests from
flash cache by smart IO"
         SIO_IO_RD_RQ_FC_HD      "Cumulative number of read IO requests from
both flash cache and hard disk by smart IO"
         SIO_IO_RD_RQ_FC_HD_SEC  "Number of read IO requests per second from
both flash cache and hard disk by smart IO"
```

```
         SIO_IO_RD_RQ_FC_SEC     "Number of read IO requests per second from
flash cache by smart IO"
         SIO_IO_RD_RQ_HD         "Cumulative number of read IO requests from
hard disk by smart IO"
         SIO_IO_RD_RQ_HD_SEC     "Number of read IO requests per second from
hard disk by smart IO"
         SIO_IO_RV_OF            "Cumulative number of megabytes sent to DB
node to balance CPU by smart IO"
         SIO_IO_RV_OF_SEC        "Number of megabytes per second sent to DB
node to balance CPU by smart IO"
         SIO_IO_SI_SV            "Cumulative number of megabytes saved by
storage index"
         SIO_IO_SI_SV_SEC        "Number of megabytes per second saved by
storage index"
         SIO_IO_WR_FC            "Cumulative number of megabytes of flash
cache population writes by smart IO"
         SIO_IO_WR_FC_SEC        "Number of megabytes per second of flash
cache population writes by smart IO"
         SIO_IO_WR_HD            "Cumulative number of megabytes written to
hard disk by smart IO"
         SIO_IO_WR_HD_SEC        "Number of megabytes per second written to
hard disk by smart IO"
         SIO_IO_WR_RQ_FC         "Cumulative number of IO requests for flash
cache population writes by smart IO"
         SIO_IO_WR_RQ_FC_SEC     "Number of IO requests per second for flash
cache population writes by smart IO"
         SIO_IO_WR_RQ_HD         "Cumulative number of write IO requests to
hard disk by smart IO"
         SIO_IO_WR_RQ_HD_SEC     "Number of write IO requests per second to
hard disk by smart IO"
```

Columnar cache metrics are identified as having `name like 'FC_COL.*'`

Example 6-7   Displaying Columnar Cache Metric Definitions

This example shows how to display the columnar cache metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE NAME LIKE
'FC_COL.*'
         FC_COL_BYKEEP_USED          "Number of megabytes used for keep
objects in Columnar FlashCache"
         FC_COL_BY_USED              "Number of megabytes used in Columnar
FlashCache"
         FC_COL_IO_BYKEEP_R          "Number of megabytes read from Columnar
FlashCache for keep objects"
         FC_COL_IO_BYKEEP_R_SEC      "Number of megabytes read per second
from Columnar FlashCache for keep objects"
         FC_COL_IO_BY_R              "Number of megabytes that were read
from Columnar FlashCache"
         FC_COL_IO_BY_R_ELIGIBLE     "Number of megabytes eligible to read
from Columnar FlashCache"
         FC_COL_IO_BY_R_ELIGIBLE_SEC  "Number of megabytes per second
eligible to read from Columnar FlashCache"
         FC_COL_IO_BY_R_SEC          "Number of megabytes per second that
were read from Columnar FlashCache"
```

```
         FC_COL_IO_BY_SAVED           "Number of megabytes saved by reads
from Columnar FlashCache"
         FC_COL_IO_BY_SAVED_SEC       "Number of megabytes saved per second
by reads from Columnar FlashCache"
         FC_COL_IO_BY_W_POPULATE      "Number of megabytes that are
population writes into Columnar FlashCache due to read miss"
         FC_COL_IO_BY_W_POPULATE_SEC  "Number of megabytes per second that
are population writes into Columnar FlashCache due to read miss"
         FC_COL_IO_RQKEEP_R           "Number of requests read for keep
objects from Columnar FlashCache"
         FC_COL_IO_RQKEEP_R_SEC       "Number of requests read per second for
keep objects from Columnar FlashCache"
         FC_COL_IO_RQ_R               "Number of requests that were read from
Columnar FlashCache"
         FC_COL_IO_RQ_R_ELIGIBLE      "Number of reads eligible for Columnar
FlashCache"
         FC_COL_IO_RQ_R_ELIGIBLE_SEC  "Number of reads per second eligible
for Columnar FlashCache"
         FC_COL_IO_RQ_R_SEC           "Number of requests per second that
were read from Columnar FlashCache"
         FC_COL_IO_RQ_W_POPULATE      "Number of requests that are population
writes into Columnar FlashCache due to read miss"
         FC_COL_IO_RQ_W_POPULATE_SEC  "Number of requests per second that are
population writes into Columnar FlashCache due to read miss"
```

6.3.7.6 What to Look For When Monitoring Smart I/O

Smart I/O Not Performing As Expected

Smart I/O operations typically occur when a full table scan or index fast full scan is done on a row source. If smart I/O operations do not work as expected, then users tend to experience noticeable increases in query elapsed times. In some cases, the database shows increased `cell smart table scan` wait times. However, when wait events such as `cell multiblock physical read` or `direct path read` are present instead of `cell smart table scan`, this is an indicator that smart I/O operations are not being performed.

The following are symptoms and reasons that explain why smart I/O operations might not perform as expected:

- Direct reads are a prerequisite for Smart Scan, and Smart Scan cannot occur without direct reads.

  The `cell multiblock physical read` wait event occurs when blocks are read into the buffer cache. A common reason for reading into the buffer cache, rather than using direct reads, is the size of the segment. For small segments, the optimizer favors the buffer cache over direct reads.

  Also, by design, Oracle Database shared server sessions do not use direct reads. Consequently, serial queries issued by a shared server session are not eligible for Smart Scan. For parallel queries issued in a shared server session, the parallel worker processes can use direct reads (and Smart Scan), but the query blocks the shared server for its entire duration.

- The `direct path read` wait event occurs when direct reads are performed, but predicates are not offloaded to the storage servers. This may occur when there is a resource shortage on the storage servers. For example, there might be a memory shortage caused by a large number of concurrent parallel queries on the system.

Such resource shortages are typically evident in the `ExaWatcher` data. In particular, you can review statistics such as `Number of low memory threshold failures` and `Number of no memory threhsold failures` in `cellsrvstat`. ExaWatcher also includes `cellmem` collection, which shows how memory is consumed on the storage servers, and is visible in the ExaWatcher charts produced using `GetExaWatcherResults.sh`.

To address this issue, you can review the use of parallel query and potentially reduce the number of active parallel query servers.

- Predicate offload is not possible when there is uncommitted data. This typically becomes an issue if large batch operations are modifying the data, and you attempt to run a large query against a large amount of uncommitted data.

  When Smart Scan encounters uncommitted data, predicate filtering cannot be offloaded to the storage servers, and additional data must be transported back to the database servers, which appears as increased bytes returned by the Smart Scan. Extra processing is also required on the database server to construct a read-consistent copy of the data, which is manifested in the following ways:

  – In the best-case scenario, additional `buffer gets` or `session logical reads` are required to construct a read-consistent copy of the data.

  – If the undo buffers reside in another database instance, then Oracle RAC-related wait events may also be observed. The Oracle RAC-related wait events are prefixed by `gc`.

  – If the undo blocks do not reside in the buffer cache of any database instance, then additional `cell single block physical read` waits are observed in conjunction with the single block I/O that is required for read-consistency. The additional I/O can significantly impact the performance of the operation.

- When the storage server CPU utilization is high, the storage server sends data back to the database for processing, rather than consuming even more storage server CPU to perform predicate evaluation. This is known as 'reverse offload'.

  When this occurs, it is evident in the `Reverse Offload` column in the Smart IO section of the AWR report, and the database statistic `cell physical IO bytes sent directly to DB node to balance CPU usage`.

  The high CPU utilization on the storage servers may be due to the type of predicates being offloaded. For example, case-insensitive searches, or the use of `REGEXP_LIKE` utilize more CPU than simpler predicates.

  Increased storage server CPU and I/O load may also stem from SQL execution plan changes in the database, in which case, reviewing the execution plans and tuning the affected SQL statements may help to resolve the issue.

- When the storage server is unable to perform predicate evaluation, it will send the data back to the database for processing. This is also known as passthough (or 'passthru'). The following all indicate that passthrough is occurring:

  – A large value in the `Passthru` column in the Smart IO section of the AWR report, when compared to eligible bytes.

  – A large value in the database statistic `cell num bytes in passthru during predicate offload`, when compared to `cell physical IO bytes eligible for smart IO`.

  – A large `Cell passthru IO bytes` value in the SQL monitor row source statistic, compared to the `Eligible bytes for Smart IO` value.

  Possible causes include:

  - Quarantines — To confirm, review the database statistic `cell num bytes in passthru due to quarantine`, or the SQL monitor row source statistic `Cell passthru IO bytes due to quarantine`.
  - Database timezone upgrade — Smart Scan is disabled when a database timezone upgrade is ongoing. Review the database statistic `cell num smart IO sessions using passthru mode due to timezone` from the Global Activity Statistics or Instance Activity Statistics section, or the Passthru Reasons in the Smart IO section of the AWR report. Depending on the database release, you may also observe the `cell smart table scan: db timezone upgrade` or `cell smart index scan: db timezone upgrade` wait event.
  - User setting — A user or application may set `cell_offload_processing=false`, which disables smart scans. To confirm, review the database statistic `cell num smart IO sessions using passthru mode due to user` from the Global Activity Statistics or Instance Activity Statistics section, or the Passthru Reasons in the Smart IO section of the AWR report. Depending on the database release, you may also observe the `cell smart table scan: disabled by user` or `cell smart index scan: disabled user` wait event.
  - Operation cannot be offloaded — There are other reasons why the storage servers may be unable to perform predicate offload. Instances of this occurrence would be visible in the database statistic `cell num smart IO sessions using passthru mode due to cellsrv`, or in wait events `cell smart table scan: passthrough` or `cell smart index scan: passthrough`. The following section describes the reasons in detail.
- In some situations, Oracle Database reverts to block I/O mode for an operation typically performed using smart I/O. When this occurs, it is evident in the database statistic `cell num bytes in block IO during predicate offload`.

  There are additional related statistics that provide insight into the underlying cause. These statistics have names beginning with `cell num smart IO sessions in rdbms block IO due to`. For example, `cell num smart IO sessions in rdbms block IO due to online encr` counts sessions that revert to block I/O mode because of an ongoing online encryption operation that prevents using smart I/O.

Operation Not Being Offloaded

A smart I/O operation cannot be offloaded to the Exadata storage servers in the following cases:

- A scan is performed on a clustered table
- A scan is performed on an index-organized table
- 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 `CREATE INDEX` using `nosort`
- A `LONG` column is being selected or queried.
- The query contains a compressed or out-of-line `LOB`. An out-of-line `LOB` stores `LOB` data apart from the other row data and is typically larger than 4 KB in size.
- A `SELECT ... VERSIONS` query is done on a table

- A query that has more than 255 columns referenced, and the heap table is uncompressed, or Basic or OLTP compressed. However, such queries on tables compressed using Exadata Hybrid Columnar Compression are offloaded.
- The tablespace is encrypted, and the `CELL_OFFLOAD_DECRYPTION` parameter is set to `FALSE`. In order for the Oracle Exadata System Software to perform decryption, Oracle Database needs to send the decryption key to the storage server. This feature is typically disabled, if there are security concerns about keys being shipped across the network to the storage server
- The tablespace is not completely stored on Oracle Exadata Storage Server.
- The predicate evaluation is on a virtual column.
- Although offloading is supported for most SQL operators and functions, Oracle Exadata System Software does not support offloading for some SQL operators and functions. The dynamic performance view `V$SQLFN_METADATA` includes information about whether offloading is supported for a SQL operator or function. If the `OFFLOADABLE` column contains `YES`, then offloading is supported for the corresponding operator or function. `NO` indicates that offloading is not supported for the corresponding operator or function.

Storage Index Not Performing as Expected

The storage index resides in storage server memory. Before the persistent storage index feature (introduced in Oracle Exadata System Software release 21.2.0), the storage index is lost whenever `cellsrv` stops and must be rebuilt after `cellsrv` starts. Consequently, on systems not enabled with the persistent storage index feature, the storage index is not immediately available every time `cellsrv` starts.

On systems enabled with the persistent storage index feature, the storage index resides in shared memory, which is maintained across `cellsrv` restarts. Furthermore, storage index data is automatically saved to persistent storage during a graceful server shutdown. During server restart, the storage index is automatically restored but is unavailable during the restoration process. Finally, the storage index is lost and must be completely rebuilt if the storage server suffers from an ungraceful shutdown (such as a power outage or kernel panic).

For segments in an unencrypted tablespace, the storage index is maintained when DML occurs. However, for segments in an encrypted tablespace, DML invalidates the portion of the storage index associated with each changed data chunk (1 MB). The invalidated chunks are rebuilt during the next scan of the segment. However, the overall efficiency of the storage index is not optimal while portions are invalid.

To monitor storage index performance, monitor:

- The `Storage Index` column in the Smart IO section of the AWR report.
- The database statistic `cell physical IO bytes saved by storage index`.
- The SQL monitor row source statistic `SI saved bytes`.

You may see reduced storage index savings if the storage index is not yet built (or rebuilt). While the storage index is building, you will see increases in the database statistic `cell physical IO bytes added to storage index` and the SQL monitor row source statistic `Bytes added to storage index`.

Columnar Cache Not Performing as Expected

Like storage index, if the columnar cache is not yet built (or rebuilt), you may see reduced savings associated with columnar cache. To monitor columnar cache performance, monitor the Columnar Cache sections of the AWR report, or the database statistic `cell physical IO` `bytes saved by columnar cache`, or the SQL monitor row source statistic `Columnar cache saved bytes`.

Similar to storage index, the columnar cache is rebuilt every time `cellsrv` starts. Consequently, the columnar cache cannot benefit operations immediately after `cellsrv` starts.

The columnar cache is automatically populated and maintained by the storage server when a Smart Scan is performed. For Smart Scan operations on uncompressed segments, segments compressed using OLTP compression, and segments compressed using Exadata Hybrid Columnar Compression, data is automatically converted into the columnar cache format (`no memcompress`) as part of the Smart Scan.

However, if you are using Oracle Database In-Memory, data is rewritten into Oracle Database In-Memory columnar format (`memcompress for query` or `memcompress for capacity`) by using a background process. Consequently, operations on that data do not benefit from the Oracle Database In-Memory optimizations until the cache is repopulated. Information about the population jobs is available in the Columnar Cache Population sections of the AWR report.

If you are reading less from columnar cache, then it will be evident by lower values in the database statistics: `cell physical IO bytes processed for IM Query`, `cell physical IO bytes processed for IM Capacity`, or `cell physical IO bytes processed for no memcompress`. The equivalent SQL monitor row source statistics are: `IM Query bytes`, `IM Capacity bytes`, and `No memcompress bytes`.

6.3.8 Monitoring I/O Resource Management (IORM)

Exadata I/O Resource Management (IORM) enables multiple databases and pluggable databases (PDBs) to share the same storage while ensuring that I/O resources are allocated appropriately across the various databases that share the system. This prevents one database from utilizing the entire I/O bandwidth and degrading the performance of the other databases. IORM also works in conjunction with Oracle Database Resource Manager to manage I/O resources across multiple consumer groups within a single database.

- Monitoring I/O Resource Management (IORM) Using AWR
- Monitoring I/O Resource Management (IORM) Using Database Statistics
- Monitoring I/O Resource Management (IORM) Using Exadata Metrics
- What to Look For When Monitoring I/O Resource Management (IORM)

6.3.8.1 Monitoring I/O Resource Management (IORM) Using AWR

The AWR report includes information about the top databases running on the Exadata system. By using this information, you can monitor system-wide I/O resource consumption regardless of the database that hosts the AWR report. Additionally, the AWR report includes per-cell statistics, which enables you to easily determine if there is any skew in the I/O resource usage across the cells.

The AWR report includes:

- Top Databases by IO Requests — provides statistics for the databases with the greatest number of I/O requests across all cells. It also breaks down the requests by device type (flash or disk) and by size (small or large).
- Top Databases by Requests - Details — provides additional statistics regarding the I/O requests. Specifically, it shows the average latency and the average queue time for I/O requests. The queue time is the amount of time that an I/O requests spends in relevant I/O queue. Large queue times indicate that IORM is throttling the I/O.

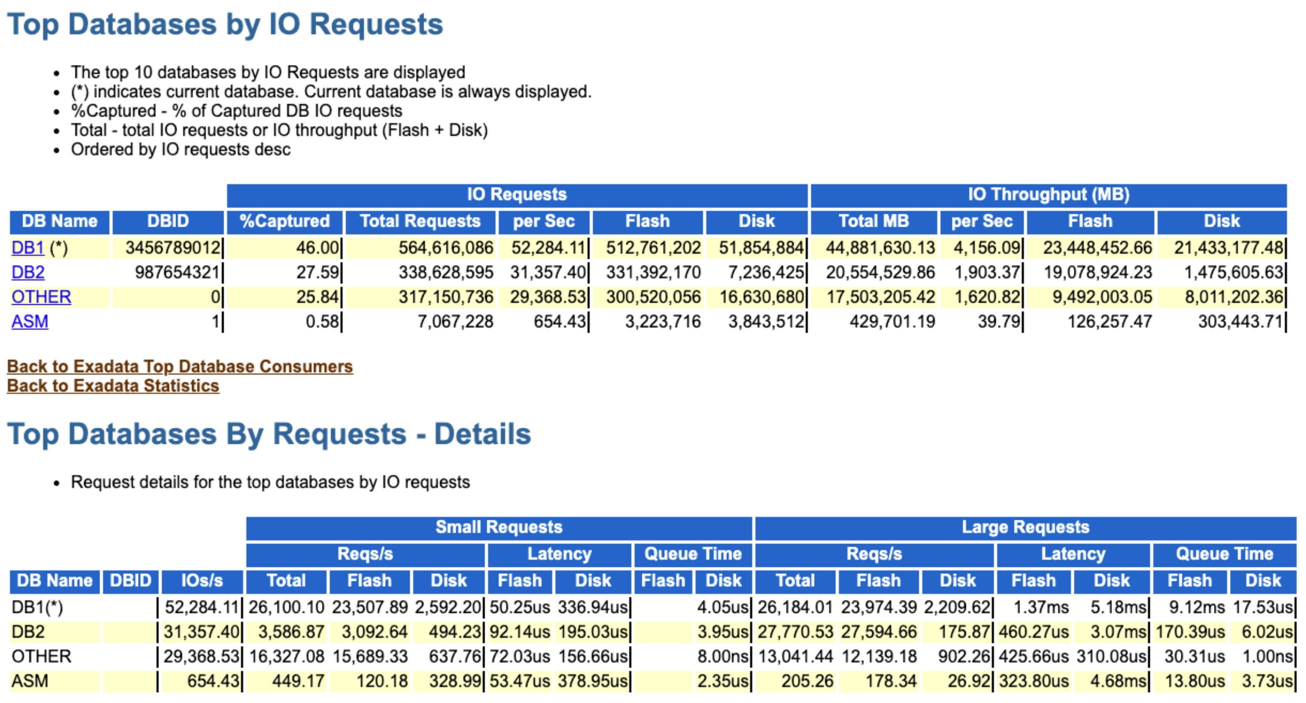
- Top Databases by IO Requests per Cell — provides statistics for the databases with the greatest number of I/O requests on each cell. It also breaks down the requests by device type (flash or disk) and by size (small or large). By using this information, you can easily determine if any cell behaves differently when servicing the I/O requests from the database(s).
- Top Databases by IO Requests per Cell - Details — provides additional statistics regarding the I/O requests on each cell. Specifically, it shows the average latency and the average queue time for I/O requests on each cell. By using this information, you can easily determine if IORM behaves differently on any cell.
- Top Databases by IO Throughput — provides statistics for the databases with the greatest I/O throughput across all cells. It also breaks down the throughput by device type (flash or disk) and by request size (small or large).
- Top Databases by Throughput per Cell — provides statistics for the databases with the greatest I/O throughput on each cell. It also breaks down the throughput by device type (flash or disk) and by request size (small or large). By using this information, you can easily determine if any cell behaves differently when servicing the I/O from the database(s).

The AWR report may not be show information about all of the databases running on the Exadata system. Rather, it is designed to focus on the databases that are responsible for the majority of I/O on the storage servers. If more detailed information is required, then IORM cell metrics should be used.

For multitenant container databases (CDBs), the statistics in the AWR report include all of the I/Os associated with the database, including all of the pluggable databases (PDBs).

The following example shows the Top Databases by IO Requests and Top Databases by Requests - Details sections in the AWR report. The Top Databases by IO Requests section shows that the DB1 database generates 46% of the I/O requests captured in AWR. It also shows that approximately 10% of the I/O requests associated with the DB1 database are I/O requests to disk devices (51,854,884 disk I/O requests of the 564,616,086 total I/O requests). The Top Databases by Requests - Details section mostly shows minimal IORM wait time (queue time < 1 millisecond) across all databases. The exception is large I/O requests to flash for the DB1 database, which each have an average queue time of 9.12 milliseconds.

Figure 6-25 AWR Report: Top Databases by IO Requests



Top Databases by IO Requests

- The top 10 databases by IO Requests are displayed
- (*) indicates current database. Current database is always displayed.
- %Captured - % of Captured DB IO requests
- Total - total IO requests or IO throughput (Flash + Disk)
- Ordered by IO requests desc

		IO Requests					IO Throughput (MB)			
DB Name	DBID	%Captured	Total Requests	per Sec	Flash	Disk	Total MB	per Sec	Flash	Disk
DB1 (*)	3456789012	46.00	564,616,086	52,284.11	512,761,202	51,854,884	44,881,630.13	4,156.09	23,448,452.66	21,433,177.48
DB2	987654321	27.59	338,628,595	31,357.40	331,392,170	7,236,425	20,554,529.86	1,903.37	19,078,924.23	1,475,605.63
OTHER	0	25.84	317,150,736	29,368.53	300,520,056	16,630,680	17,503,205.42	1,620.82	9,492,003.05	8,011,202.36
ASM	1	0.58	7,067,228	654.43	3,223,716	3,843,512	429,701.19	39.79	126,257.47	303,443.71

Back to Exadata Top Database Consumers
Back to Exadata Statistics

Top Databases By Requests - Details

- Request details for the top databases by IO requests

			Small Requests							Large Requests						
			Reqs/s			Latency		Queue Time		Reqs/s			Latency		Queue Time	
DB Name	DBID	IOs/s	Total	Flash	Disk	Flash	Disk	Flash	Disk	Total	Flash	Disk	Flash	Disk	Flash	Disk
DB1(*)		52,284.11	26,100.10	23,507.89	2,592.20	50.25us	336.94us		4.05us	26,184.01	23,974.39	2,209.62	1.37ms	5.18ms	9.12ms	17.53us
DB2		31,357.40	3,586.87	3,092.64	494.23	92.14us	195.03us		3.95us	27,770.53	27,594.66	175.87	460.27us	3.07ms	170.39us	6.02us
OTHER		29,368.53	16,327.08	15,689.33	637.76	72.03us	156.66us		8.00ns	13,041.44	12,139.18	902.26	425.66us	310.08us	30.31us	1.00ns
ASM		654.43	449.17	120.18	328.99	53.47us	378.95us		2.35us	205.26	178.34	26.92	323.80us	4.68ms	13.80us	3.73us

6.3.8.2 Monitoring I/O Resource Management (IORM) Using Database Statistics

The following table describes database statistics that are useful for monitoring I/O Resource Management (IORM). The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`Session total flash IO requests`	Total number of physical I/O requests on flash.
`Session IORM flash wait time`	Total amount of IORM wait time for flash I/O requests (in microseconds). To determine the average IORM wait time for flash I/O requests, divide `Session IORM flash wait time` by `Session total flash IO requests`.
`PDB total disk IO requests`	Total number of physical I/O requests on disk.
`PDB IORM disk wait time`	Total amount of IORM wait time for disk I/O requests (in microseconds). To determine the average IORM wait time for disk I/O requests, divide `PDB IORM disk wait time` by `PDB total disk IO requests`.

The availability of a specific statistic is subject to the version of Oracle Database being used.

6.3.8.3 Monitoring I/O Resource Management (IORM) Using Exadata Metrics

You can monitor I/O Resource Management (IORM) by using Oracle Exadata System Software metrics.

IORM uses the database unique name, not the database identifier, to collect statistics and display output. Starting with Oracle Exadata System Software release 19.1.0, if you configured ASM-scoped security for the Oracle Automatic Storage Management (Oracle ASM) cluster used by the database, then the database name is prefixed with the Oracle ASM cluster name.

- Monitoring IORM with Database Metrics
  Database metrics provide information about the I/O load from each database listed in the IORM interdatabase plan.
- Monitoring IORM with PDB Metrics
  Pluggable Database (PDB) metrics provide information about the I/O load from each PDB hosted by a container database (CDB) listed in the IORM interdatabase plan.
- Monitoring IORM with Consumer Group Metrics
  Consumer group metrics provide information about the I/O load from each consumer group specified in a database resource plan.
- Monitoring IORM with Category Metrics
  Category metrics provide information about the I/O load from each category specified in the current IORM category plan.
- Monitoring IORM Utilization
  You can use metrics to monitor IORM utilization.

6.3.8.3.1 Monitoring IORM with Database Metrics

Database metrics provide information about the I/O load from each database listed in the IORM interdatabase plan.

Database metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=IORM_DATABASE`.

Example 6-8 Displaying Category Metric Definitions

This example shows how to display the database metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
IORM_DATABASE
         DB_FC_BY_ALLOCATED      "Number of megabytes allocated in flash
cache for this database"
         DB_FC_IO_BY_SEC         "Number of megabytes of I/O per second for
this database to flash cache"
         DB_FC_IO_RQ             "Number of IO requests issued by a database
to flash cache"
         DB_FC_IO_RQ_LG          "Number of large IO requests issued by a
database to flash cache"
         DB_FC_IO_RQ_LG_SEC      "Number of large IO requests issued by a
database to flash cache per second"
         DB_FC_IO_RQ_SEC         "Number of IO requests issued by a database
to flash cache per second"
         DB_FC_IO_RQ_SM          "Number of small IO requests issued by a
database to flash cache"
         DB_FC_IO_RQ_SM_SEC      "Number of small IO requests issued by a
database to flash cache per second"
         DB_FD_IO_BY_SEC         "Number of megabytes of I/O per second for
this database to flash disks"
         DB_FD_IO_LOAD           "Average I/O load from this database for
flash disks"
```

```
         DB_FD_IO_RQ_LG          "Number of large IO requests issued by a
database to flash disks"
         DB_FD_IO_RQ_LG_SEC      "Number of large IO requests issued by a
database to flash disks per second"
         DB_FD_IO_RQ_SM          "Number of small IO requests issued by a
database to flash disks"
         DB_FD_IO_RQ_SM_SEC      "Number of small IO requests issued by a
database to flash disks per second"
         DB_FD_IO_TM             "The cumulative latency of reading or
writing blocks by a database from flash disks"
         DB_FD_IO_TM_LG          "The cumulative latency of reading or
writing large blocks by a database from flash disks"
         DB_FD_IO_TM_LG_RQ       "The rate which is the average latency of
reading or writing large blocks per request by a database from flash disks"
         DB_FD_IO_TM_RQ          "The rate which is the average latency of
reading or writing blocks per request by a database from flash disks"
         DB_FD_IO_TM_SM          "The cumulative latency of reading or
writing small blocks by a database from flash disks"
         DB_FD_IO_TM_SM_RQ       "The rate which is the average latency of
reading or writing small blocks per request by a database from flash disks"
         DB_FD_IO_UTIL           "Percentage of flash resources utilized by
this database"
         DB_FD_IO_UTIL_LG        "Percentage of flash resources utilized by
large requests from this database"
         DB_FD_IO_UTIL_SM        "Percentage of flash resources utilized by
small requests from this database"
         DB_FD_IO_WT_LG          "IORM wait time for large IO requests issued
to flash disks by a database"
         DB_FD_IO_WT_LG_RQ       "Average IORM wait time per request for
large IO requests issued to flash disks by a database"
         DB_FD_IO_WT_SM          "IORM wait time for small IO requests issued
to flash disks by a database"
         DB_FD_IO_WT_SM_RQ       "Average IORM wait time per request for
small IO requests issued to flash disks by a database"
         DB_FL_IO_BY             "The number of MB written to the Flash Log"
         DB_FL_IO_BY_SEC         "The number of MB written per second to the
Flash Log"
         DB_FL_IO_RQ             "The number of I/O requests issued to the
Flash Log"
         DB_FL_IO_RQ_SEC         "The number of I/O requests per second
issued to the Flash Log"
         DB_IO_BY_SEC            "Number of megabytes of I/O per second for
this database to hard disks"
         DB_IO_LOAD              "Average I/O load from this database for
hard disks"
         DB_IO_RQ_LG             "Number of large IO requests issued by a
database to hard disks"
         DB_IO_RQ_LG_SEC         "Number of large IO requests issued by a
database to hard disks per second"
         DB_IO_RQ_SM             "Number of small IO requests issued by a
database to hard disks"
         DB_IO_RQ_SM_SEC         "Number of small IO requests issued by a
database to hard disks per second"
         DB_IO_TM_LG             "The cumulative latency of reading or
writing large blocks by a database from hard disks"
         DB_IO_TM_LG_RQ          "The rate which is the average latency of
```

```
reading or writing large blocks per request by a database from hard disks"
         DB_IO_TM_SM             "The cumulative latency of reading or
writing small blocks by a database from hard disks"
         DB_IO_TM_SM_RQ          "The rate which is the average latency of
reading or writing small blocks per request by a  database from hard disks"
         DB_IO_UTIL_LG           "Percentage of disk resources utilized by
large requests from this database"
         DB_IO_UTIL_SM           "Percentage of disk resources utilized by
small requests from this database"
         DB_IO_WT_LG             "IORM wait time for large IO requests issued
to hard disks by a database"
         DB_IO_WT_LG_RQ          "Average IORM wait time per request for
large IO requests issued to hard disks by a database"
         DB_IO_WT_SM             "IORM wait time for small IO requests issued
to hard disks by a database"
         DB_IO_WT_SM_RQ          "Average IORM wait time per request for
small IO requests issued to hard disks by a database"
         DB_XRM_BY_ALLOCATED      "Number of megabytes allocated in XRMEM
cache for this database"
```

Note the following additional details:

- The database for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects.
- For metrics that relate to I/O load (for example, `DB_FD_IO_LOAD` and `DB_IO_LOAD`), see the additional information relating to `CD_IO_LOAD`.
- Starting with Oracle Exadata System Software release 19.1.0, if you configured ASM-scoped security for the Oracle Automatic Storage Management (Oracle ASM) cluster used by the database, then the database name is prefixed with the Oracle ASM cluster name.
- 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 metric descriptions, small I/O requests are less than or equal to 128 KB, and large I/O requests are larger than 128 KB.
- 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 inter-database plan, use the following CellCLI command:

  ```
  CellCLI> LIST METRICHISTORY WHERE objectType = 'IORM_DATABASE' AND
  metricValue != 0 ATTRIBUTES name, metricObjectName, metricValue,
  collectionTime
  ```

- For multitenant container databases (CDBs), the database metric observations include all of the I/Os associated with the database, including all of the associated pluggable databases (PDBs). For example, the value for `DB_FC_IO_BY_SEC` includes the sum of the `PDB_FC_IO_BY_SEC` values for all of the PDBs hosted by the CDB.
- Observations for Oracle ASM and all other databases not listed in the interdatabase plan are grouped together using `_OTHER_DATABASE_` as the `metricObjectName` value.

6.3.8.3.2 Monitoring IORM with PDB Metrics

Pluggable Database (PDB) metrics provide information about the I/O load from each PDB hosted by a container database (CDB) listed in the IORM interdatabase plan.

PDB metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=IORM_PLUGGABLE_DATABASE`.

Example 6-9   Displaying PDB Metric Definitions

This example shows how to display the PDB metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE = 
IORM_PLUGGABLE_DATABASE
         PDB_FC_BY_ALLOCATED     "Number of megabytes allocated in flash 
cache for this pluggable database"
         PDB_FC_IO_BY_SEC        "Number of megabytes of I/O per second for 
this pluggable database to flash cache"
         PDB_FC_IO_RQ            "Number of IO requests issued by this 
pluggable database to flash cache"
         PDB_FC_IO_RQ_LG         "Number of large IO requests issued by this 
pluggable database to flash cache"
         PDB_FC_IO_RQ_LG_SEC     "Number of large IO requests issued by this 
pluggable database to flash cache per second"
         PDB_FC_IO_RQ_SEC        "Number of IO requests issued by this 
pluggable database to flash cache per second"
         PDB_FC_IO_RQ_SM         "Number of small IO requests issued by this 
pluggable database to flash cache"
         PDB_FC_IO_RQ_SM_SEC     "Number of small IO requests issued by this 
pluggable database to flash cache per second"
         PDB_FD_IO_BY_SEC        "Number of megabytes of I/O per second for 
this pluggable database to flash disks"
         PDB_FD_IO_LOAD          "Average I/O load from this pluggable 
database for flash disks"
         PDB_FD_IO_RQ_LG         "Number of large IO requests issued by this 
pluggable database to flash disks"
         PDB_FD_IO_RQ_LG_SEC     "Number of large IO requests issued by this 
pluggable database to flash disks per second"
         PDB_FD_IO_RQ_SM         "Number of small IO requests issued by this 
pluggable database to flash disks"
         PDB_FD_IO_RQ_SM_SEC     "Number of small IO requests issued by this 
pluggable database to flash disks per second"
         PDB_FD_IO_TM            "The cumulative latency of reading or 
writing blocks by this pluggable database from flash disks"
         PDB_FD_IO_TM_LG         "The cumulative latency of reading or 
writing large blocks by this pluggable database from flash disks"
         PDB_FD_IO_TM_LG_RQ      "The rate which is the average latency of 
reading or writing large blocks per request by this pluggable database from 
flash disks"
         PDB_FD_IO_TM_RQ         "The rate which is the average latency of 
reading or writing blocks per request by this pluggable database from flash 
disks"
         PDB_FD_IO_TM_SM         "The cumulative latency of reading or 
writing small blocks by this pluggable database from flash disks"
         PDB_FD_IO_TM_SM_RQ      "The rate which is the average latency of
```

```
reading or writing small blocks per request by this pluggable database from 
flash disks"
         PDB_FD_IO_UTIL          "Percentage of flash resources utilized by 
this pluggable database"
         PDB_FD_IO_UTIL_LG       "Percentage of flash resources utilized by 
large requests from this pluggable database"
         PDB_FD_IO_UTIL_SM       "Percentage of flash resources utilized by 
small requests from this pluggable database"
         PDB_FD_IO_WT_LG         "IORM wait time for large IO requests issued 
to flash disks by this pluggable database"
         PDB_FD_IO_WT_LG_RQ      "Average IORM wait time per request for 
large IO requests issued to flash disks by this pluggable database"
         PDB_FD_IO_WT_SM         "IORM wait time for small IO requests issued 
to flash disks by this pluggable database"
         PDB_FD_IO_WT_SM_RQ      "Average IORM wait time per request for 
small IO requests issued to flash disks by this pluggable database"
         PDB_IO_BY_SEC           "Number of megabytes of I/O per second for 
this pluggable database to hard disks"
         PDB_IO_LOAD             "Average I/O load from this pluggable 
database for hard disks"
         PDB_IO_RQ_LG            "Number of large IO requests issued by this 
pluggable database to hard disks"
         PDB_IO_RQ_LG_SEC        "Number of large IO requests issued by this 
pluggable database to hard disks per second"
         PDB_IO_RQ_SM            "Number of small IO requests issued by this 
pluggable database to hard disks"
         PDB_IO_RQ_SM_SEC        "Number of small IO requests issued by this 
pluggable database to hard disks per second"
         PDB_IO_TM_LG            "The cumulative latency of reading or 
writing large blocks by this pluggable database from hard disks"
         PDB_IO_TM_LG_RQ         "The rate which is the average latency of 
reading or writing large blocks per request by this pluggable database from 
hard disks"
         PDB_IO_TM_SM            "The cumulative latency of reading or 
writing small blocks by this pluggable database from hard disks"
         PDB_IO_TM_SM_RQ         "The rate which is the average latency of 
reading or writing small blocks per request by this pluggable database from 
hard disks"
         PDB_IO_UTIL_LG          "Percentage of disk resources utilized by 
large requests from this pluggable database"
         PDB_IO_UTIL_SM          "Percentage of disk resources utilized by 
small requests from this pluggable database"
         PDB_IO_WT_LG            "IORM wait time for large IO requests issued 
to hard disks by this pluggable database"
         PDB_IO_WT_LG_RQ         "Average IORM wait time per request for 
large IO requests issued to hard disks by this pluggable database"
         PDB_IO_WT_SM            "IORM wait time for small IO requests issued 
to hard disks by this pluggable database"
         PDB_IO_WT_SM_RQ         "Average IORM wait time per request for 
small IO requests issued to hard disks by this pluggable database"
         PDB_XRM_BY_ALLOCATED    "Number of megabytes allocated in XRMEM 
cache for this pluggable database"
```

Note the following additional details:

- The PDB for the metric is specified by the `metricObjectName` attribute of the `METRICCURRENT` and `METRICHISTORY` objects. The PDB name is a concatenation of the CDB name with the PDB name.
- For metrics that relate to I/O load (for example, `PDB_FD_IO_LOAD` and `PDB_IO_LOAD`), see the additional information relating to `CD_IO_LOAD`.
- Starting with Oracle Exadata System Software release 19.1.0, if you configured ASM-scoped security for the Oracle Automatic Storage Management (Oracle ASM) cluster used by the database, then the database name is prefixed with the Oracle ASM cluster name.
- 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 metric descriptions, small I/O requests are less than or equal to 128 KB, and large I/O requests are larger than 128 KB.

6.3.8.3.3 Monitoring IORM with Consumer Group Metrics

Consumer group metrics provide information about the I/O load from each consumer group specified in a database resource plan.

Consumer group metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=IORM_CONSUMER_GROUP`.

Example 6-10 Displaying Consumer Group Metric Definitions

This example shows how to display the consumer group metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
IORM_CONSUMER_GROUP
         CG_FC_IO_BY_SEC         "Number of megabytes of I/O per second for
this consumer group to flash cache"
         CG_FC_IO_RQ             "Number of IO requests issued by a consumer
group to flash cache"
         CG_FC_IO_RQ_LG          "Number of large IO requests issued by a
consumer group to flash cache"
         CG_FC_IO_RQ_LG_SEC      "Number of large IO requests issued by a
consumer group to flash cache per second"
         CG_FC_IO_RQ_SEC         "Number of IO requests issued by a consumer
group to flash cache per second"
         CG_FC_IO_RQ_SM          "Number of small IO requests issued by a
consumer group to flash cache"
         CG_FC_IO_RQ_SM_SEC      "Number of small IO requests issued by a
consumer group to flash cache per second"
         CG_FD_IO_BY_SEC         "Number of megabytes of I/O per second for
this consumer group to flash disks"
         CG_FD_IO_LOAD           "Average I/O load from this consumer group
for flash disks"
         CG_FD_IO_RQ_LG          "Number of large IO requests issued by a
consumer group to flash disks"
         CG_FD_IO_RQ_LG_SEC      "Number of large IO requests issued by a
consumer group to flash disks per second"
         CG_FD_IO_RQ_SM          "Number of small IO requests issued by a
consumer group to flash disks"
         CG_FD_IO_RQ_SM_SEC      "Number of small IO requests issued by a
```

```
consumer group to flash disks per second"
         CG_FD_IO_TM            "The cumulative latency of reading or
writing blocks by a consumer group from flash disks"
         CG_FD_IO_TM_LG         "The cumulative latency of reading or
writing large blocks by a consumer group from flash disks"
         CG_FD_IO_TM_LG_RQ      "The rate which is the average latency of
reading or writing large blocks per request by a consumer group from flash
disks"
         CG_FD_IO_TM_RQ         "The rate which is the average latency of
reading or writing blocks per request by a consumer group from flash disks"
         CG_FD_IO_TM_SM         "The cumulative latency of reading or
writing small blocks by a consumer group from flash disks"
         CG_FD_IO_TM_SM_RQ      "The rate which is the average latency of
reading or writing small blocks per request by a consumer group from flash
disks"
         CG_FD_IO_UTIL          "Percentage of flash resources utilized by
this consumer group"
         CG_FD_IO_UTIL_LG       "Percentage of flash resources utilized by
large requests from this consumer group"
         CG_FD_IO_UTIL_SM       "Percentage of flash resources utilized by
small requests from this consumer group"
         CG_FD_IO_WT_LG         "IORM wait time for large IO requests issued
to flashdisks by a consumer group"
         CG_FD_IO_WT_LG_RQ      "Average IORM wait time per request for
large IO requests issued to flash disks by a consumer group"
         CG_FD_IO_WT_SM         "IORM wait time for small IO requests issued
to flashdisks by a consumer group"
         CG_FD_IO_WT_SM_RQ      "Average IORM wait time per request for
small IO requests issued to flash disks by a consumer group"
         CG_IO_BY_SEC           "Number of megabytes of I/O per second for
this consumer group to hard disks"
         CG_IO_LOAD             "Average I/O load from this consumer group
for hard disks"
         CG_IO_RQ_LG            "Number of large IO requests issued by a
consumer group to hard disks"
         CG_IO_RQ_LG_SEC        "Number of large IO requests issued by a
consumer group to hard disks per second"
         CG_IO_RQ_SM            "Number of small IO requests issued by a
consumer group to hard disks"
         CG_IO_RQ_SM_SEC        "Number of small IO requests issued by a
consumer group to hard disks per second"
         CG_IO_TM_LG            "The cumulative latency of reading or
writing large blocks by a consumer group from hard disks"
         CG_IO_TM_LG_RQ         "The rate which is the average latency of
reading or writing large blocks per request by a consumer group from hard
disks"
         CG_IO_TM_SM            "The cumulative latency of reading or
writing small blocks by a consumer group from hard disks"
         CG_IO_TM_SM_RQ         "The rate which is the average latency of
reading or writing small blocks per request by a consumer group from hard
disks"
         CG_IO_UTIL_LG          "Percentage of disk resources utilized by
large requests from this consumer group"
         CG_IO_UTIL_SM          "Percentage of disk resources utilized by
small requests from this consumer group"
         CG_IO_WT_LG            "IORM wait time for large IO requests issued
```

```
to hard disks by a consumer group"
         CG_IO_WT_LG_RQ          "Average IORM wait time per request for
large IO requests issued to hard disks by a consumer group"
         CG_IO_WT_SM             "IORM wait time for small IO requests issued
to hard disks by a consumer group"
         CG_IO_WT_SM_RQ          "Average IORM wait time per request for
small IO requests issued to hard disks by a consumer group"
```

Note the following additional details:

- 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 metrics that relate to I/O load (for example, `CG_FD_IO_LOAD` and `CG_IO_LOAD`), see the additional information relating to `CD_IO_LOAD`.
- Starting with Oracle Exadata System Software release 19.1.0, if you configured ASM-scoped security for the Oracle Automatic Storage Management (Oracle ASM) cluster used by the database, then the database name is prefixed with the Oracle ASM cluster name.
- 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 metric descriptions, small I/O requests are less than or equal to 128 KB, and large I/O requests are larger than 128 KB.
- 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

6.3.8.3.4 Monitoring IORM with Category Metrics

Category metrics provide information about the I/O load from each category specified in the current IORM category plan.

Category metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=IORM_CATEGORY`.

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.

Example 6-11   Displaying Category Metric Definitions

This example shows how to display the category metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
IORM_CATEGORY
         CT_FC_IO_BY_SEC          "Number of megabytes of I/O per second for
this category to flash cache"
         CT_FC_IO_RQ              "Number of IO requests issued by an IORM
category to flash cache"
         CT_FC_IO_RQ_LG           "Number of large IO requests issued by an
IORM category to flash cache"
         CT_FC_IO_RQ_LG_SEC       "Number of large IO requests issued by an
IORM category to flash cache per second"
         CT_FC_IO_RQ_SEC          "Number of IO requests issued by an IORM
category to flash cache per second"
         CT_FC_IO_RQ_SM           "Number of small IO requests issued by an
IORM category to flash cache"
         CT_FC_IO_RQ_SM_SEC       "Number of small IO requests issued by an
IORM category to flash cache per second"
         CT_FD_IO_BY_SEC          "Number of megabytes of I/O per second for
this category to flash disks"
         CT_FD_IO_LOAD            "Average I/O load from this category for
flash disks"
         CT_FD_IO_RQ_LG           "Number of large IO requests issued by an
IORM category to flash disks"
         CT_FD_IO_RQ_LG_SEC       "Number of large IO requests issued by an
IORM category to flash disks per second"
         CT_FD_IO_RQ_SM           "Number of small IO requests issued by an
IORM category to flash disks"
         CT_FD_IO_RQ_SM_SEC       "Number of small IO requests issued by an
IORM category to flash disks per second"
         CT_FD_IO_TM              "The cumulative latency of reading or
writing blocks for this category from flash disks"
         CT_FD_IO_TM_LG           "The cumulative latency of reading or
writing large blocks for this category from flash disks"
         CT_FD_IO_TM_LG_RQ        "The rate which is the average latency of
reading or writing large blocks per request for this category from flash
disks"
         CT_FD_IO_TM_RQ           "The rate which is the average latency of
reading or writing blocks per request for this category from flash disks"
         CT_FD_IO_TM_SM           "The cumulative latency of reading or
writing small blocks for this category from flash disks"
         CT_FD_IO_TM_SM_RQ        "The rate which is the average latency of
reading or writing small blocks per request for this category from flash
disks"
         CT_FD_IO_UTIL            "Percentage of flash resources utilized by
this category"
         CT_FD_IO_UTIL_LG         "Percentage of flash resources utilized by
```

```
large requests from this category"
         CT_FD_IO_UTIL_SM        "Percentage of flash resources utilized by
small requests from this category"
         CT_FD_IO_WT_LG          "IORM wait time for large IO requests issued
to flash disks by an IORM category"
         CT_FD_IO_WT_LG_RQ       "Average IORM wait time per request for
large IO requests issued to flash disks by an IORM category"
         CT_FD_IO_WT_SM          "IORM wait time for small IO requests issued
to flash disks by an IORM category"
         CT_FD_IO_WT_SM_RQ       "Average IORM wait time per request for
small IO requests issued to flash disks by an IORM category"
         CT_IO_BY_SEC            "Number of megabytes of I/O per second for
this category to hard disks"
         CT_IO_LOAD              "Average I/O load from this category for
hard disks"
         CT_IO_RQ_LG             "Number of large IO requests issued by an
IORM category to hard disks"
         CT_IO_RQ_LG_SEC         "Number of large IO requests issued by an
IORM category to hard disks per second"
         CT_IO_RQ_SM             "Number of small IO requests issued by an
IORM category to hard disks"
         CT_IO_RQ_SM_SEC         "Number of small IO requests issued by an
IORM category to hard disks per second"
         CT_IO_TM_LG             "The cumulative latency of reading or
writing large blocks for this category from hard disks"
         CT_IO_TM_LG_RQ          "The rate which is the average latency of
reading or writing large blocks per request for this category from hard disks"
         CT_IO_TM_SM             "The cumulative latency of reading or
writing small blocks for this category from hard disks"
         CT_IO_TM_SM_RQ          "The rate which is the average latency of
reading or writing small blocks per request for this category from hard disks"
         CT_IO_UTIL_LG           "Percentage of disk resources utilized by
large requests from this category"
         CT_IO_UTIL_SM           "Percentage of disk resources utilized by
small requests from this category"
         CT_IO_WT_LG             "IORM wait time for large IO requests issued
to hard disks by an IORM category"
         CT_IO_WT_LG_RQ          "Average IORM wait time per request for
large IO requests issued to hard disks by an IORM category"
         CT_IO_WT_SM             "IORM wait time for small IO requests issued
to hard disks by an IORM category"
         CT_IO_WT_SM_RQ          "Average IORM wait time per request for
small IO requests issued to hard disks by an IORM category"
```

Note the following additional details:

- In the metric descriptions, small I/O requests are less than or equal to 128 KB, and large I/O requests are larger than 128 KB.
- The unit of measurement for the wait metrics is milliseconds. The wait metrics have metric names starting with `CD_IO_WT_`.
- For metrics that relate to I/O load (for example, `CT_FD_IO_LOAD` and `CT_IO_LOAD`), see the additional information relating to `CD_IO_LOAD`.
- 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
  ```

- Category metrics are also 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

6.3.8.3.5 Monitoring IORM Utilization

You can use metrics to monitor IORM utilization.

When OLTP and DSS workloads share Oracle Exadata Storage Servers, IORM determines whether to optimize for low latency or high throughput. To optimize for low latency, the concurrency of large I/O requests is reduced to ensure that I/O bandwidth is not saturated. To optimize for high throughput, each Oracle Exadata Storage Server must handle many concurrent large I/O requests, allowing the storage to be fully utilized while applying optimization algorithms. However, when a cell has many concurrent large I/O requests, average latency may increase because each I/O is queued behind other I/Os.

The utilization metrics for I/O requests from a database, pluggable database (PDB), or consumer group corresponds to the amount of time that the database, PDB, or consumer group utilized the storage server. Large I/O requests utilize the storage server more than small I/O requests. The following are the utilization metrics for determining IORM optimization:

- `CG_IO_UTIL_LG`
- `CG_IO_UTIL_SM`
- `PDB_IO_UTIL_LG`
- `PDB_IO_UTIL_SM`
- `CT_IO_UTIL_LG`
- `CT_IO_UTIL_SM`
- `DB_IO_UTIL_LG`
- `DB_IO_UTIL_SM`

By comparing the I/O resource consumption with the I/O resource allocations, the database administrator can determine if IORM should be tuned for latency or throughput, or if a balanced approach is optimal. 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:

Note:

If the current plan has the IORM attribute `objective` set to `BASIC`, then `IORM_MODE` has no meaning and should be ignored.

- 1 means the cell IORM objective is set to `low_latency`.
- 2 means the cell IORM objective is set to `balanced`.
- 3 means the cell IORM objective is set to `high_throughput`.

A value in between 1 and 2, or between 2 and 3, indicates that the IORM objective changed in the metric period, and the precise value indicates proximity to a given objective. It is also indicative of a constantly-changing mix of workloads.

6.3.8.4 What to Look For When Monitoring I/O Resource Management (IORM)

I/O Latency

Issues with IORM typically result in increased I/O latency. This is usually characterized by higher latency in the `cell single block physical read` database wait event, and in some cases the `cell smart table scan` database wait event. If these database wait events are significant, and there is no corresponding latencies in the storage servers that are associated with the flash or disk devices, then this may indicate that IORM is throttling the workload.

To confirm that IORM throttling is occurring:

- In the Top Databases by Requests - Details section of the AWR report, review the Queue Time columns, which show the average amount of time that IORM spent throttling IO requests for the database.
- Review the cell metrics for IORM wait times: `DB_IO_WT_SM_RQ`, `DB_IO_WT_LG_RQ`, `PDB_IO_WT_SM_RQ`, `PDB_IO_WT_LG_RQ` and `CG_IO_WT_SM_RQ`.

You can use IORM database statistics and the AWR report to understand the I/O workload as a whole. You can use Exadata metrics to further understand I/O consumption by each category, database, or consumer group. By analyzing statistics and metrics, you can understand which category, database, pluggable database (PDB), or consumer group is not using its resource allocation and which is exceeding its resource allocation.

If the wait times are small or zero, then the plan allocation is sufficient. If the wait times are large, then the plan allocation is insufficient. If the wait times due to high I/O latency result in unacceptable performance, then the IORM plan can be adjusted to give a larger allocation, or additional storage servers may be required to deliver the required I/O resources.

Using IOSTAT

Device utilization and I/O service time statistics monitored using `iostat` (which is collected by ExaWatcher) are unreliable.

The following is stated in the Linux `man` page for `iostat`:

```
svctm - The average service time (in milliseconds) for I/O requests that were
issued to the device. Warning! Do not trust this field any more. This field will
be removed in a future sysstat version.
```

Since the utilization computation depends upon the I/O service time, it is also unreliable.

To monitor the actual I/O utilization for a cell disk, database, pluggable database or consumer group, use the corresponding IORM metrics.

6.3.9 Monitoring Cell Disk I/O

When database performance issues are related to I/O load on the Exadata storage servers, typically there will be increased latencies in the I/O-related wait events, and increased database time in the User I/O or System I/O wait classes. If the increased database latencies are due to the performance of the Exadata storage servers, then the increased latencies will also be visible in the cell-side statistics. If you have comprehensive baseline statistics for periods when the system is performing well, then you can compare the baseline statistics with other observations to identify differences and explain the situation.

For example, if Oracle Database reports increased `cell single block physical read` latencies, you can then check the statistics from the storage servers and compare them to a baseline to determine if the cause is increased latency on flash devices, more disk I/O, or some other cause. If the statistics show an increase in disk I/O requests, that may be related to a change in Exadata Smart Flash Cache, which would prompt you to review the Exadata Smart Flash Cache statistics. On the other hand, if the statistics show a latency increase for small reads from flash, then this may be caused by a change in the I/O pattern, and understanding the type of I/O that has increased (small reads, small writes, large reads, or large writes) can help to drive further investigation.

Typically, in an Exadata environment, an even workload distribution is expected across all cells and all disks. If one cell or one disk is doing more work than the others, then you should investigate further, as that cell or disk has the potential of slowing down the entire system.

- Monitoring Cell Disk I/O Using AWR
- Monitoring Cell Disk I/O Using Database Statistics and Wait Events
- Monitoring Cell Disk I/O Using Exadata Metrics
- What to Look For When Monitoring Cell Disk I/O

6.3.9.1 Monitoring Cell Disk I/O Using AWR

The following sections in the Automatic Workload Repository (AWR) report are particularly useful for understanding I/O load on Exadata:

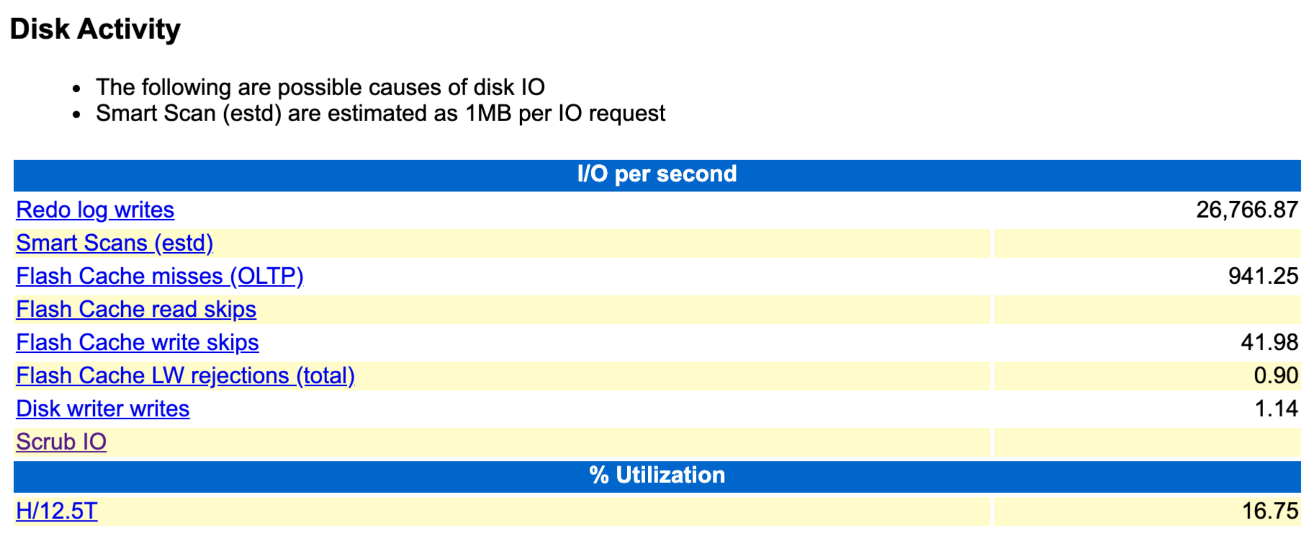
- Disk Activity
- Exadata Resource Statistics
- Exadata IO Reasons
- Internal IO Reasons

Often, to better understand characteristics about the I/O load, the statistics from these sections can be correlated with other sections in the AWR report.

Disk Activity

The Disk Activity section provides a high-level summary for potential sources of disk activity. The Disk Activity section is located in the AWR report under Exadata Statistics > Performance Summary.

Figure 6-26 AWR Report: Disk Activity



Disk Activity

- The following are possible causes of disk IO
- Smart Scan (estd) are estimated as 1MB per IO request

I/O per second	
Redo log writes	26,766.87
Smart Scans (estd)	
Flash Cache misses (OLTP)	941.25
Flash Cache read skips	
Flash Cache write skips	41.98
Flash Cache LW rejections (total)	0.90
Disk writer writes	1.14
Scrub IO	
% Utilization	
H/12.5T	16.75

High I/O load or a substantial change in the pattern of disk activity may prompt further investigation. Possible causes include:

- Redo log write — result in disk writes when redo is written to disk. When using Exadata Smart Flash Log, note that redo is written to both Exadata Smart Flash Log and the online redo log file. Also, Oracle Exadata System Software release 20.1 adds a further optimization, known as Smart Flash Log Write-Back, that uses Exadata Smart Flash Cache in Write-Back mode instead of disk storage. For further details, review Database Redo Activity and Smart Flash Log in the AWR report.
- Smart Scans — result in disk reads for requests that are not satisfied using Exadata Smart Flash Cache. These are typically large reads. For further details, review Smart IO in the AWR report.
- Flash Cache misses — result in disk reads when requested data does not exist in Exadata Smart Flash Cache. These are typically small reads. For further details, review Flash Cache Misses in the AWR report.
- Flash Cache read skips — result in disk reads when requested data is not eligible for Exadata Smart Flash Cache. For further details, review Flash Cache User Reads - Skips in the AWR report.
- Flash Cache write skips or Flash Cache LW rejections — results in disk writes when data is not eligible for Exadata Smart Flash Cache. For further details, review Flash Cache User Writes - Skips and Flash Cache User Writes - Large Write Rejects in the AWR report.
- Disk writer writes — result in disk writes when data from Exadata Smart Flash Cache in Write-Back mode is persisted to disk. For further details, review Flash Cache Internal Writes in the AWR report.
- Scrub IO — occurs when Oracle Exadata System Software automatically inspects and repairs the hard disks. Scrub I/O is performed periodically when the hard disks are idle, and mostly results in large disk reads, which should be throttled automatically if the disk becomes I/O bound.

The specific causes listed in this section are subject to the version of Oracle Database being used.

Exadata Resource Statistics

The Exadata Resource Statistics section contains many statistics and is organized into several sub-sections. Primarily, the statistics enumerate the I/O occurring on the storage servers using information from the storage server operating system (OS) and Oracle Exadata System Software. From the OS, it includes statistics relating to I/Os per second (IOPS), throughput, utilization, service time and wait time. These statistics are equivalent to the statistics shown by the `iostat` command. From the Oracle Exadata System Software, it includes statistics relating to IOPS, throughput, and latency, which are also broken down by small reads, small writes, large reads, and large writes. These statistics are based on the cell disk metrics.

The statistics are aggregated by device type, and then by cell or disk. The device type aggregation ensures comparison across the same device type, as different device types are expected to have different performance characteristics. The statistics are presented two ways. Firstly, they are presented to enable outlier analysis. They are also organized to show the 'Top N' cells or disks for a specific statistic.

The outlier analysis presentation allows you to quickly see the statistics aggregated across all storage servers, by cell, and by disk. The display also includes the statistical mean, standard deviation, and normal range. The normal range is based on the mean and standard deviation, not the observed low and high values. For cells, the normal range is the range of values that are plus or minus one standard deviation from the mean. For disks, the normal range is the range of values that are plus or minus three standard deviations from the mean. If there are cells or disks that fall outside the normal range, then they are reported as outliers. This simple outlier analysis is designed to highlight areas for further investigation. However, based on the number of cells or disks in use, and the value of the standard deviation, the outlier analysis may not identify outliers in all cases.

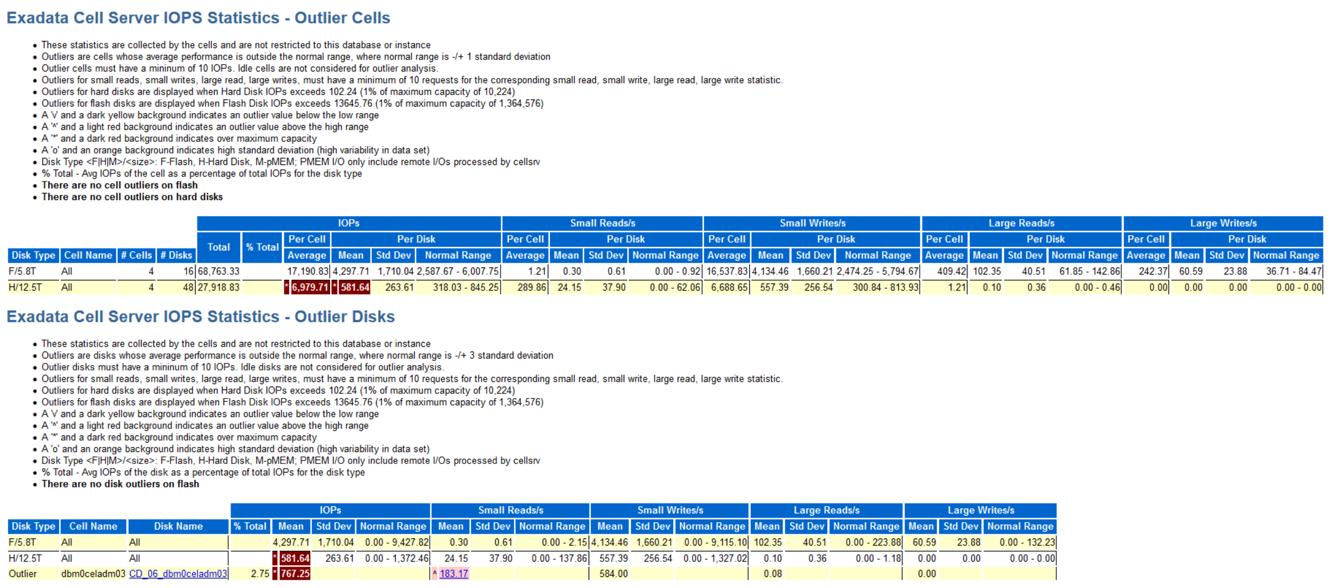
The 'Top N' presentation simply shows the top ranked cells or disks for a specific statistic. This presentation enables you to identify cells or disks that perform more or less work than the others. By using this presentation, you can potentially identify outliers that are not identified by the outlier analysis. Also highlighted in these sections are cells or disks that exceed the expected maximum IOPS for the device or the expected maximum throughput for the device.

The following list outlines the sub-sections in the Exadata Resource Statistics section of the AWR report:

- Exadata Outlier Summary — displays a summary of outliers identified in the various outlier sub-sections.
- Exadata OS Statistics Outliers — contains outlier sub-sections for cells and disks based on OS statistics including, IOPS, throughput (MB/s), utilization percentage, service time, wait time, and CPU utilization per cell.
- Exadata Cell Server Statistics — contains outlier sub-sections for cells and disks based on cell disk metrics, including IOPS, throughput (MB/s), and latency. The statistics are further categorized by I/O type; that is, small read, small write, large read, or large write.
- Exadata Outlier Details — displays detailed information for the identified outliers, along with other statistics related to the outlier.
- Exadata OS Statistics Top — contains 'Top N' sub-sections for cells and disks based on OS statistics, including IOPS, latency, and CPU utilization.
- Exadata Cell Server Statistics Top — contains 'Top N' sub-sections for cells and disks based on cell disk metrics, including IOPS, throughput (MB/s), and latency. The statistics are further categorized by I/O type; that is, small read, small write, large read, or large write.

The following example shows two of the Exadata Cell Server Statistics outlier sub-sections in the AWR report. The example highlights that hard disk throughput (IOPS) exceeds the expected maximum, and that a specific disk that is performing substantially more small reads than other disks.

Figure 6-27 AWR Report: Exadata Cell Server IOPS Statistics Outliers



Exadata Cell Server IOPS Statistics - Outlier Cells

- These statistics are collected by the cells and are not restricted to this database or instance
- Outliers are cells whose average performance is outside the normal range, where normal range is -/+ 1 standard deviation
- Outlier cells must have a mininum of 10 IOPs. Idle cells are not considered for outlier analysis.
- Outliers for small reads, small writes, large read, large writes, must have a minimum of 10 requests for the corresponding small read, small write, large read, large write statistic.
- Outliers for hard disks are displayed when Hard Disk IOPs exceeds 102.24 (1% of maximum capacity of 10,224)
- Outliers for flash disks are displayed when Flash Disk IOPs exceeds 13645.76 (1% of maximum capacity of 1,364,576)
- A 'V' and a dark yellow background indicates an outlier value below the low range
- A '^' and a light red background indicates an outlier value above the high range
- A '*' and a dark red background indicates over maximum capacity
- A 'o' and an orange background indicates high standard deviation (high variability in data set)
- Disk Type <F|H|M>/<size>: F-Flash, H-Hard Disk, M-pMEM; PMEM I/O only include remote I/Os processed by cellsrv
- % Total - Avg IOPs of the cell as a percentage of total IOPs for the disk type
- **There are no cell outliers on flash**
- **There are no cell outliers on hard disks**

				IOPs						Small Reads/s				Small Writes/s				Large Reads/s				Large Writes/s			
				Total	% Total	Per Cell	Per Disk			Per Cell	Per Disk			Per Cell	Per Disk			Per Cell	Per Disk			Per Cell	Per Disk		
Disk Type	Cell Name	# Cells	# Disks			Average	Mean	Std Dev	Normal Range	Average	Mean	Std Dev	Normal Range	Average	Mean	Std Dev	Normal Range	Average	Mean	Std Dev	Normal Range	Average	Mean	Std Dev	Normal Range
F/5.8T	All	4	16	68,763.33		17,190.83	4,297.71	1,710.04	2,587.67 - 6,007.75	1.21	0.30	0.61	0.00 - 0.92	16,537.83	4,134.46	1,660.21	2,474.25 - 5,794.67	409.42	102.35	40.51	61.85 - 142.86	242.37	60.59	23.88	36.71 - 84.47
H/12.5T	All	4	48	27,918.83		* 6,979.71	* 581.64	263.61	318.03 - 845.25	289.86	24.15	37.90	0.00 - 62.06	6,688.65	557.39	256.54	300.84 - 813.93	1.21	0.10	0.36	0.00 - 0.46	0.00	0.00	0.00	0.00 - 0.00

Exadata Cell Server IOPS Statistics - Outlier Disks

- These statistics are collected by the cells and are not restricted to this database or instance
- Outliers are disks whose average performance is outside the normal range, where normal range is -/+ 3 standard deviation
- Outlier disks must have a mininum of 10 IOPs. Idle disks are not considered for outlier analysis.
- Outliers for small reads, small writes, large read, large writes, must have a minimum of 10 requests for the corresponding small read, small write, large read, large write statistic.
- Outliers for hard disks are displayed when Hard Disk IOPs exceeds 102.24 (1% of maximum capacity of 10,224)
- Outliers for flash disks are displayed when Flash Disk IOPs exceeds 13645.76 (1% of maximum capacity of 1,364,576)
- A 'V' and a dark yellow background indicates an outlier value below the low range
- A '^' and a light red background indicates an outlier value above the high range
- A '*' and a dark red background indicates over maximum capacity
- A 'o' and an orange background indicates high standard deviation (high variability in data set)
- Disk Type <F|H|M>/<size>: F-Flash, H-Hard Disk, M-pMEM; PMEM I/O only include remote I/Os processed by cellsrv
- % Total - Avg IOPs of the disk as a percentage of total IOPs for the disk type
- **There are no disk outliers on flash**

			IOPs				Small Reads/s			Small Writes/s			Large Reads/s			Large Writes/s		
Disk Type	Cell Name	Disk Name	% Total	Mean	Std Dev	Normal Range	Mean	Std Dev	Normal Range	Mean	Std Dev	Normal Range	Mean	Std Dev	Normal Range	Mean	Std Dev	Normal Range
F/5.8T	All	All		4,297.71	1,710.04	0.00 - 9,427.82	0.30	0.61	0.00 - 2.15	4,134.46	1,660.21	0.00 - 9,115.10	102.35	40.51	0.00 - 223.88	60.59	23.88	0.00 - 132.23
H/12.5T	All	All		* 581.64	263.61	0.00 - 1,372.46	24.15	37.90	0.00 - 137.86	557.39	256.54	0.00 - 1,327.02	0.10	0.36	0.00 - 1.18	0.00	0.00	0.00 - 0.00
Outlier	dbm0celadm03	CD_06_dbm0celadm03	2.75	* 767.25			^ 183.17			584.00			0.08			0.00		

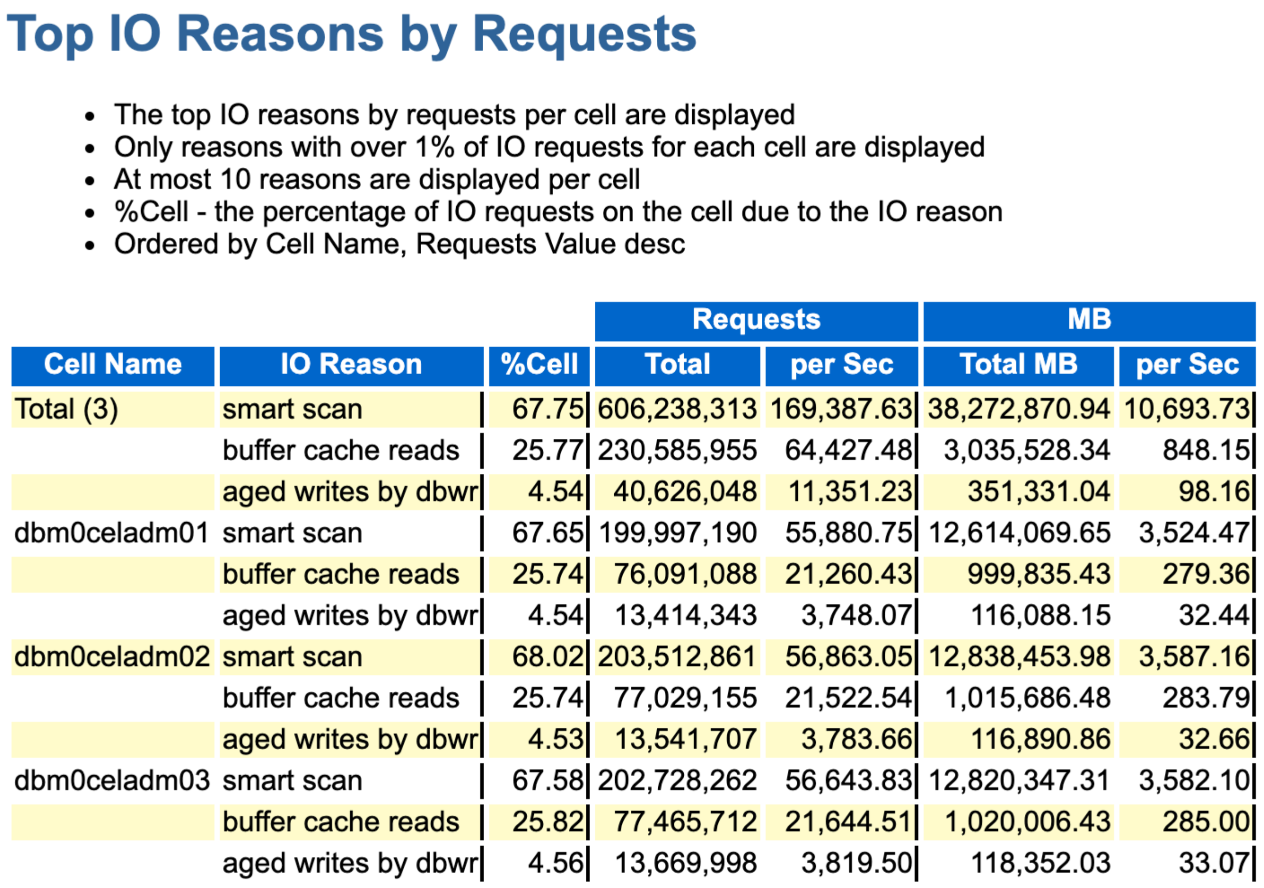
Exadata IO Reasons

When the database sends an I/O request to Exadata, the request is tagged with information that includes the reason for the I/O. This information is aggregated in the Exadata IO Reasons sections of the AWR report and allows you to understand the reasons for performing I/O.

The AWR report contains sub-sections that display the Top IO Reasons by Requests and the Top IO Reasons by MB (throughput). Later versions of the AWR report, further break down the Top IO Reasons by Requests to categorize read requests from flash, write requests to flash, read requests from disk, and write requests to disk.

The following example shows Top IO Reasons by Requests. The example output is typical of a well-performing system, with a high proportion of I/O that is associated with Smart Scan, and similar I/O profiles across all of the storage servers.

Figure 6-28   AWR Report: Top IO Reasons by Request



Top IO Reasons by Requests

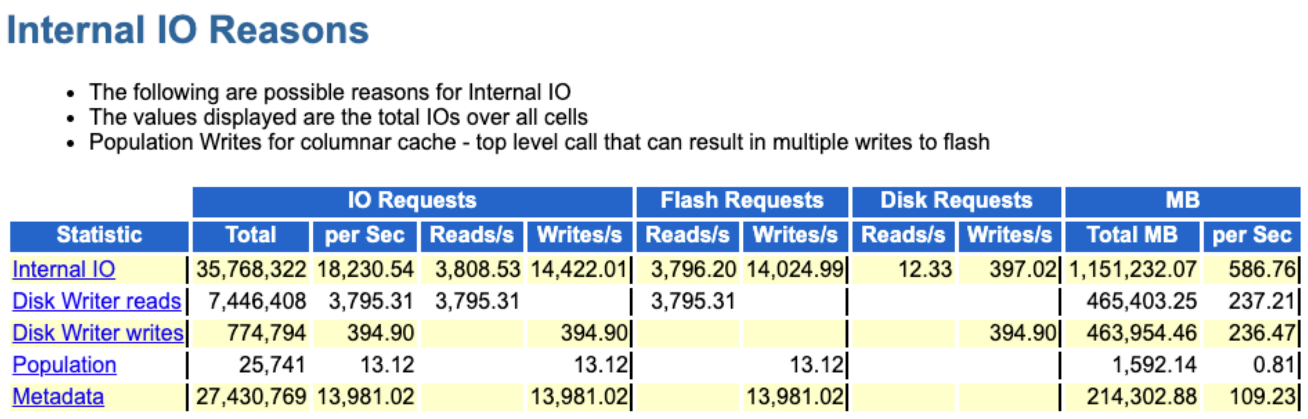
- The top IO reasons by requests per cell are displayed
- Only reasons with over 1% of IO requests for each cell are displayed
- At most 10 reasons are displayed per cell
- %Cell - the percentage of IO requests on the cell due to the IO reason
- Ordered by Cell Name, Requests Value desc

			Requests		MB	
Cell Name	IO Reason	%Cell	Total	per Sec	Total MB	per Sec
Total (3)	smart scan	67.75	606,238,313	169,387.63	38,272,870.94	10,693.73
	buffer cache reads	25.77	230,585,955	64,427.48	3,035,528.34	848.15
	aged writes by dbwr	4.54	40,626,048	11,351.23	351,331.04	98.16
dbm0celadm01	smart scan	67.65	199,997,190	55,880.75	12,614,069.65	3,524.47
	buffer cache reads	25.74	76,091,088	21,260.43	999,835.43	279.36
	aged writes by dbwr	4.54	13,414,343	3,748.07	116,088.15	32.44
dbm0celadm02	smart scan	68.02	203,512,861	56,863.05	12,838,453.98	3,587.16
	buffer cache reads	25.74	77,029,155	21,522.54	1,015,686.48	283.79
	aged writes by dbwr	4.53	13,541,707	3,783.66	116,890.86	32.66
dbm0celadm03	smart scan	67.58	202,728,262	56,643.83	12,820,347.31	3,582.10
	buffer cache reads	25.82	77,465,712	21,644.51	1,020,006.43	285.00
	aged writes by dbwr	4.56	13,669,998	3,819.50	118,352.03	33.07

Internal IO Reasons

If Internal IO is among the Top IO Reasons reported, then the AWR report will include a section that summarizes the Internal IO Reasons:

Figure 6-29   AWR Report: Internal IO Reasons



Internal IO Reasons

- The following are possible reasons for Internal IO
- The values displayed are the total IOs over all cells
- Population Writes for columnar cache - top level call that can result in multiple writes to flash

	IO Requests				Flash Requests		Disk Requests		MB	
Statistic	Total	per Sec	Reads/s	Writes/s	Reads/s	Writes/s	Reads/s	Writes/s	Total MB	per Sec
Internal IO	35,768,322	18,230.54	3,808.53	14,422.01	3,796.20	14,024.99	12.33	397.02	1,151,232.07	586.76
Disk Writer reads	7,446,408	3,795.31	3,795.31		3,795.31				465,403.25	237.21
Disk Writer writes	774,794	394.90		394.90				394.90	463,954.46	236.47
Population	25,741	13.12		13.12		13.12			1,592.14	0.81
Metadata	27,430,769	13,981.02		13,981.02		13,981.02			214,302.88	109.23

Possible causes of internal I/O include:

- Disk Writer reads — results in flash reads when the disk writer reads from Exadata Smart Flash Cache in Write-Back mode to persist data to disk. For further details, review Flash Cache Internal Reads in the AWR report.
- Disk Writer writes — results in disk writes when the disk writer persists data to disk from Exadata Smart Flash Cache in Write-Back mode. For further details, review Flash Cache Internal Reads in the AWR report.
- Population — results in flash writes when requested data is read into Exadata Smart Flash Cache. When the data is read from disk, it also populates Exadata Smart Flash Cache. This is often correlated with flash cache misses. For further details, review Flash Cache User Reads and Flash Cache Internal Writes in the AWR report.
- Metadata — results in flash writes when new data is written to Exadata Smart Flash Cache in Write-Back mode. This is often due to first writes. For further details, review Flash Cache User Writes in the AWR report.

The specific causes listed in this section are subject to the version of Oracle Database being used.

6.3.9.2 Monitoring Cell Disk I/O Using Database Statistics and Wait Events

The following table describes various database statistics that are useful for monitoring cell disk I/O. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	Description
`physical read IO requests`	Number of physical read requests issued by the database.
`physical read requests optimized`	Total number of read requests satisfied by using Exadata Smart Flash Cache, and read requests avoided by using storage index or columnar cache.
`physical read total bytes`	Total amount of I/O bytes for reads issued by the database, whether or not the read was offloaded to the storage servers.
`physical read total bytes optimized`	Total number of bytes read from Exadata Smart Flash Cache, and bytes avoided by using storage index or columnar cache.
`physical read total IO requests`	Total number of read requests issued by the database for all activity including application, backup, recovery, and other utilities.
`physical write IO requests`	Number of physical write requests issued by the database.
`physical write total bytes`	Total number of IO bytes for writes issued by the database for all activity.
`physical write total bytes optimized`	Total number of bytes written to Exadata Smart Flash Cache. These bytes are synchronized to disk in a lazy manner.
`physical write total IO requests`	Total number of write requests issued by the database for all activity, including application, backup, recovery and other utilities.

Additional database I/O statistics are also contained in the following `V$IOSTAT` views, which are also used in the IOStat sections of the AWR report.

- `V$IOSTAT_FILE` — displays information about disk I/O statistics for database files, including data files, temp files, and other types of database files.
- `V$IOSTAT_FUNCTION` — displays disk I/O statistics for database functions, such as the LGWR and DBWR.
- `V$IOSTAT_FUNCTION_DETAIL` — displays disk I/O statistics for database functions, broken down by file type.

The following table describes database wait events that are useful for monitoring cell disk I/O. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`cell list of blocks physical read`	This wait event occurs during recovery or during buffer pre-fetching (rather than performing multiple single-block reads). It is used to monitor database blocks that must be changed as part of recovery and are read in parallel for the database. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file parallel read` for a cell.
`cell list of blocks read request`	This is a placeholder wait event associated with `cell list of blocks physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell list of blocks physical read`.
`cell multiblock physical read`	This wait event represents the time taken to perform all the I/Os for a multi-block database read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file scattered read` for a cell.

Wait Event	Description
`cell multiblock read request`	This is a placeholder wait event associated with `cell multiblock physical read`, which is visible only during the wait period. After the wait event ends, the placeholder is typically converted to `cell multiblock physical read`.
`cell single block physical read`	This wait event represents the time taken to perform a single block database I/O. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event is equivalent to `db file sequential read` for a cell. Commencing with the May 2022 Oracle Database release updates (versions 19.15.0.0.220419, 21.6.0.0.220419, and later), this wait event no longer includes I/O from Exadata Smart Flash Cache or database I/O using a Remote Direct Memory Access (RDMA) read.
`cell single block physical read: flash cache`	This wait event represents the time taken to perform a single block database I/O from Exadata Smart Flash Cache. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data (not the cache location), which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.

Wait Event	Description

`cell single block physical read: xrmem cache`
 	
`cell single block physical read: pmem cache`
	This wait event represents the time taken to perform a single block database I/O from XRMEM cache.
	
	In `V$SESSION`, records associated with this event include additional parameters:
	• `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`.
	• `P2` identifies the disk hash number for the grid disk that contains the data (not the cache location), which corresponds to `V$ASM_DISK.HASH_VALUE`.
	• `P3` specifies the number of bytes processed during the I/O read operation.
	
	`cell single block physical read: pmem cache` was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.
	
	The wait event is named `cell single block physical read: xrmem cache` in Oracle Database versions that support Oracle Exadata System Software release 23.1.0 and later.

`cell single block physical read: RDMA`	This wait event represents the time taken to perform a single block database I/O using a Remote Direct Memory Access (RDMA) read. In `V$SESSION`, records associated with this event include additional parameters: • `P1` contains the relevant storage server hash number, which corresponds to `V$CELL.CELL_HASHVAL`. • `P2` identifies the disk hash number for the grid disk that contains the data, which corresponds to `V$ASM_DISK.HASH_VALUE`. • `P3` specifies the number of bytes processed during the I/O read operation. This wait event was introduced in the May 2022 Oracle Database release updates and is present in Oracle Database versions 19.15.0.0.220419, 21.6.0.0.220419, and later. Previously, `cell single block physical read` included these waits.
`cell single block read request`	This is a placeholder wait event associated with a single block database I/O that is visible only during the wait period. After the wait event ends, the placeholder is converted to the appropriate wait event, which is typically one of the `cell single block physical read` events.

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.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

6.3.9.3 Monitoring Cell Disk I/O Using Exadata 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.

Cell disk metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=CELLDISK`.

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 is determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute.

Example 6-12 Displaying Cell Disk Metric Definitions

This example shows how to display the cell disk metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
CELLDISK
         CD_BY_FC_DIRTY            "Number of unflushed megabytes cached in
FLASHCACHE on a cell disk"
         CD_IO_BY_R_LG             "Number of megabytes read in large blocks
from a cell disk"
         CD_IO_BY_R_LG_SEC         "Number of megabytes read in large blocks
per second from a cell disk"
         CD_IO_BY_R_SCRUB          "Number of megabytes read from a cell disk
by the scrubbing job"
         CD_IO_BY_R_SCRUB_SEC      "Number of megabytes read per second from a
cell disk by the scrubbing job"
         CD_IO_BY_R_SM             "Number of megabytes read in small blocks
from a cell disk"
         CD_IO_BY_R_SM_SEC         "Number of megabytes read in small blocks
per second from a cell disk"
         CD_IO_BY_W_LG             "Number of megabytes written in large blocks
to a cell disk"
         CD_IO_BY_W_LG_SEC         "Number of megabytes written in large blocks
per second to a cell disk"
         CD_IO_BY_W_SM             "Number of megabytes written in small blocks
```

```
to a cell disk"
         CD_IO_BY_W_SM_SEC       "Number of megabytes written in small blocks
per second to a cell disk"
         CD_IO_ERRS              "Number of IO errors on a cell disk"
         CD_IO_ERRS_MIN          "Number of IO errors on a cell disk per
minute"
         CD_IO_ERRS_SCRUB        "Number of IO errors hit by the scrubbing
job on a cell disk"
         CD_IO_LOAD              "Average I/O load for the cell disk"
         CD_IO_RQ_R_LG           "Number of requests to read large blocks
from a cell disk"
         CD_IO_RQ_R_LG_SEC       "Number of requests to read large blocks per
second from a cell disk"
         CD_IO_RQ_R_SCRUB        "Number of requests to read from a cell disk
by the scrubbing job"
         CD_IO_RQ_R_SCRUB_SEC    "Number of requests to read per second from
a cell disk by the scrubbing job"
         CD_IO_RQ_R_SM           "Number of requests to read small blocks
from a cell disk"
         CD_IO_RQ_R_SM_SEC       "Number of requests to read small blocks per
second from a cell disk"
         CD_IO_RQ_W_LG           "Number of requests to write large blocks to
a cell disk"
         CD_IO_RQ_W_LG_SEC       "Number of requests to write large blocks
per second to a cell disk"
         CD_IO_RQ_W_SM           "Number of requests to write small blocks to
a cell disk"
         CD_IO_RQ_W_SM_SEC       "Number of requests to write small blocks
per second to a cell disk"
         CD_IO_ST_RQ             "Average service time per request for small
IO requests to a cell disk"
         CD_IO_TM_R_LG           "Cumulative latency of reading large blocks
from a cell disk"
         CD_IO_TM_R_LG_RQ        "Average latency of reading large blocks per
request to a cell disk"
         CD_IO_TM_R_SM           "Cumulative latency of reading small blocks
from a cell disk"
         CD_IO_TM_R_SM_RQ        "Average latency of reading small blocks per
request from a cell disk"
         CD_IO_TM_W_LG           "Cumulative latency of writing large blocks
to a cell disk"
         CD_IO_TM_W_LG_RQ        "Average latency of writing large blocks per
request to a cell disk"
         CD_IO_TM_W_SM           "Cumulative latency of writing small blocks
to a cell disk"
         CD_IO_TM_W_SM_RQ        "Average latency of writing small blocks per
request to a cell disk"
         CD_IO_UTIL              "Percentage of disk resources utilized for
the cell disk"
         CD_IO_UTIL_LG           "Percentage of disk resources utilized by
large requests for the cell disk"
         CD_IO_UTIL_SM           "Percentage of disk resources utilized by
small requests for the cell disk"
```

Note the following additional details:

- `CD_IO_LOAD` represents the average I/O load for the cell disk.

  I/O load specifies the length of the disk queue. It is similar to iostat's `avgqu-sz`, but I/O load is a weighted value depending on the type of disk:

  – For hard disks, a large I/O has three times the weight of a small I/O.

  – For flash disks, large and small I/Os have the same weight.

  Corresponding metrics are also available for each database (`DB_IO_LOAD`), pluggable database (PDB) (`PDB_IO_LOAD`), IORM category (`CT_IO_LOAD`), and consumer group (`CG_IO_LOAD`).

- `CD_IO_ST_RQ` cannot be used for flash devices, as explained in Using IOSTAT.

- `CD_IO_UTIL` is similar to `%util` of `iostat` for hard disks, but not for flash devices as mentioned in Using IOSTAT. For flash devices, it is a percentage of the maximum I/O bandwidth for your system, as specified in the product data sheet.

  Because this metric is computed by IORM, it is also available per database, PDB, and consumer group.

- The unit of measurement for the latency metrics is microseconds. The latency metrics have metric names starting with `CD_IO_TM_`.

6.3.9.4 What to Look For When Monitoring Cell Disk I/O

Imbalances

In an Exadata environment, an even load distribution is expected across all cells or disks. However, there are situations that may cause an imbalance.

A load imbalance may by due to characteristics of the workload, such as:

- Repeated small-table scans — This is often caused by a table on the right side of a nested loop join. Since the small table may only reside on a few disks or cells, repeated access means reading from a small set of devices, which may be flash devices if the data resides in Exadata Smart Flash Cache. To address the imbalance, you can identify affected SQL statements and review their execution plans. You can also consider the physical organization of the affected segments.

- Repeated control file reads — Control file reads may be caused by queries against some database dynamic performance views. The control file is small and may only reside on a few disks or cells, so repeated access means reading from a small set of devices, usually flash. Repeated control file reads are evident in the IOStat by File Type section of the AWR report and using the `control file sequential read` wait event.

  To understand the cause of repeated control file reads, you can use the Exadata IO Reasons section of the AWR report to identify cells servicing many control file reads and correlate this with statistics from the Top Databases section to identify databases issuing the control file reads. You can also review Active Session History (ASH) to identify SQL statements that are experiencing waits for `control file sequential read`.

  Beware that control file reads may account for a disproportionate amount of small I/O on a lightly-loaded system. In this case, you can safely ignore the imbalance.

- Repeated reads of a small segment, such as a LOB segment — To address this kind of imbalance, review database statistics for segment, or examine ASH to identify the responsible SQL commands. Then, examine the SQL and the surrounding application logic to see if the repeated reads can be avoided.

- Repeated ASM metadata access — This is often caused by database queries related to space usage, which require access to ASM metadata. The ASM metadata is small and may only reside on a few disks or cells, so repeated access may show up as an imbalance. This can show up in the Exadata IO Reasons sections of the AWR report with reasons that are prefixed with ASM, such as ASM CACHE IO. You can use the Exadata IO Reasons section to identify affected cells and correlate this with statistics from the Top Databases sections to confirm ASM as the source of the I/O. To address the imbalance, review the need for and frequency of the ASM space usage queries that access the metadata.

Apart from workload-related causes, imbalances may also be caused by:

- Uneven data distribution — Hot spots may occur when some disks contain more or less data than others. To check for balanced data distribution, you can query the `V$ASM_DISK_IOSTAT` view before and after running a SQL query that contains large full-table scans. In this case, the statistics in the `read` column and the `read_bytes` column should be approximately the same for all disks in the disk group. You can also check for balanced data distribution by using the script available in My Oracle Support document 367445.1.
- Asymmetric grid disk configurations — If the grid disks are sized differently on different cells, then different amounts of data may reside on each cell, resulting in imbalance due to uneven data distribution. For this reason, symmetric configuration is recommended across all storage servers in a system.
- Failures or recovery from failures — Some processing cannot spread evenly across the remaining disks or cells. For example, if a flash card fails, the cell containing the card has less available flash cache space, and this may show up as an imbalance when the flash cache statistics are compared with the other cells.

An imbalance often shows up in different statistics across the AWR report. By correlating different sections in the AWR report you may gain a deeper understanding of the imbalance.

High I/O Load

When database performance issues are related to I/O load on the Exadata storage servers, typically there will be increased latencies in the I/O related wait events, and increased database time in the user I/O or system I/O wait classes. When dealing with high I/O load, first understand the composition of the I/O load. Based on the composition of the I/O load you my be directed to different statistics to gain a deeper understanding.

In the AWR report, a good starting point is the Disk Activity section, which provides a high-level summary for potential sources of disk activity. The Disk Activity section is located in the AWR report under Exadata Statistics > Performance Summary. See Disk Activity.

Internal I/O

In the AWR report, if internal I/O is reported among the Top IO Reasons, then the AWR report also includes a section that summarizes the Internal IO Reasons. Use this section to understand the composition of internal I/O. Based on the composition of the internal I/O load you my be directed to different statistics to gain a deeper understanding. See Internal IO Reasons.

6.3.10 Monitoring Grid Disks

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.

Grid disk metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=GRIDDISK`.

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 is determined by subtracting values from different `collectionTime` periods. For rate metrics, the time period for the metric value is over the previous minute.

Example 6-13   Displaying Grid Disk Metric Definitions

This example shows how to display the grid disk metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
GRIDDISK
         GD_BY_FC_DIRTY                  "Number of unflushed megabytes
cached in FLASHCACHE for a grid disk"
         GD_IO_BY_R_LG                   "Number of megabytes read in large
blocks from a grid disk"
         GD_IO_BY_R_LG_SEC               "Number of megabytes read in large
blocks per second from a grid disk"
         GD_IO_BY_R_SCRUB                "Number of megabytes read from a
grid disk by the scrubbing job"
         GD_IO_BY_R_SCRUB_SEC            "Number of megabytes read per second
from a grid disk by the scrubbing job"
         GD_IO_BY_R_SM                   "Number of megabytes read in small
blocks from a grid disk"
         GD_IO_BY_R_SM_SEC               "Number of megabytes read in small
blocks per second from a grid disk"
         GD_IO_BY_W_LG                   "Number of megabytes written in
large blocks to a grid disk"
         GD_IO_BY_W_LG_SEC               "Number of megabytes written in
large blocks per second to a grid disk"
         GD_IO_BY_W_SM                   "Number of megabytes written in
small blocks to a grid disk"
         GD_IO_BY_W_SM_SEC               "Number of megabytes written in
small blocks per second to a grid disk"
         GD_IO_ERRS                      "Number of IO errors on a grid disk"
         GD_IO_ERRS_MIN                  "Number of IO errors on a grid disk
per minute"
         GD_IO_ERRS_SCRUB                "Number of IO errors hit by the
scrubbing job on a grid disk"
         GD_IO_RQ_R_LG                   "Number of requests to read large
blocks from a grid disk"
         GD_IO_RQ_R_LG_SEC               "Number of requests to read large
blocks per second from a grid disk"
         GD_IO_RQ_R_SCRUB                "Number of requests to read from a
grid disk by the scrubbing job"
         GD_IO_RQ_R_SCRUB_SEC            "Number of requests to read per
second from a grid disk by the scrubbing job"
         GD_IO_RQ_R_SM                   "Number of requests to read small
blocks from a grid disk"
         GD_IO_RQ_R_SM_SEC               "Number of requests to read small
blocks per second from a grid disk"
         GD_IO_RQ_W_LG                   "Number of requests to write large
```

```
blocks to a grid disk"
         GD_IO_RQ_W_LG_SEC                "Number of requests to write large
blocks per second to a grid disk"
         GD_IO_RQ_W_SM                    "Number of requests to write small
blocks to a grid disk"
         GD_IO_RQ_W_SM_SEC                "Number of requests to write small
blocks per second to a grid disk"
         GD_NVM_READ_RETRIES              "Number of read retries for a non-
volatile memory (NVM) grid disk"
         GD_SP_BY_ALLOCATED               "Allocated physical space for grid
disk in bytes"
         GD_SP_IO_BY_PARTIAL              "Bytes returned by partial IOs"
         GD_SP_IO_BY_PARTIAL_SEC          "Bytes returned by partial IOs per
second"
         GD_SP_IO_BY_REDIRECTED           "Sparse bytes redirected to original
data block"
         GD_SP_IO_BY_REDIRECTED_SEC       "Sparse bytes redirected to original
data block per second"
         GD_SP_IO_RQ_PARTIAL              "Number of IO requests that returned
partial data"
         GD_SP_IO_RQ_PARTIAL_SEC          "Number of IO requests that returned
partial data per second"
         GD_SP_PRCT_ALLOCATED             "Allocated physical space for grid
disk by percentage"
```

Note the following additional details:

- `GD_SP_PRCT_ALLOCATED` tracks the allocated physical space for the grid disk, and is presented as a percentage. By default, a warning alert is generated when space usage is more than 95%, and a critical alert is generated when space usage reaches 99%. To lower the space usage, you can increase the size of the grid disk, or drop some of the data on the grid disk.

6.3.11 Monitoring Host Interconnect Metrics

Host interconnect metrics provide information about the I/O transmission for hosts that access cell storage.

Host interconnect metrics are identified in the `METRICCURRENT`, `METRICDEFINITION`, and `METRICHISTORY` objects as having `objectType=HOST_INTERCONNECT`.

The host interconnect 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.

Example 6-14 Displaying Host Interconnect Metric Definitions

This example shows how to display the host interconnect metric definitions that are available in the Oracle Exadata System Software.

```
CellCLI> LIST METRICDEFINITION ATTRIBUTES NAME,DESCRIPTION WHERE OBJECTTYPE =
HOST_INTERCONNECT
         N_MB_DROP                "Number of megabytes droped during
transmission  to  a particular host"
         N_MB_DROP_SEC            "Number of megabytes droped during
```

```
transmission  per second  to  a particular host"
         N_MB_RDMA_DROP          "Number of megabytes dropped during RDMA
transmission to  a particular host"
         N_MB_RDMA_DROP_SEC      "Number of megabytes dropped during RDMA
transmission per second  to  a particular host"
         N_MB_RECEIVED           "Number of megabytes received from a
particular host"
         N_MB_RECEIVED_SEC       "Number of megabytes per second received
from a particular host"
         N_MB_RESENT             "Number of megabytes resent  to  a
particular host"
         N_MB_RESENT_SEC         "Number of megabytes resent per second to  a
particular host"
         N_MB_SENT               "Number of megabytes transmitted to  a
particular host"
         N_MB_SENT_SEC           "Number of megabytes transmitted per second
to  a particular host"
         N_RDMA_RETRY_TM         "Latency of the retry actions during RDMA
transmission to a particular host"
```

6.4 Using Exadata Monitoring Objects

- Using Metrics
  You can use CellCLI commands to display, monitor, and manage storage server metrics.
- Monitoring Alerts
  You can monitor and receive notifications for alerts.
- Displaying Active Requests
  Use the `LIST ACTIVEREQUEST` command to display the active requests for the storage server.
- Using Real-Time Insight
  You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems using an external metric collection platform.

6.4.1 Using Metrics

You can use CellCLI commands to display, monitor, and manage storage server metrics.

- Displaying Metric Definitions
  Use the `LIST METRICDEFINITION` command to display the metric definitions for a storage server.
- Displaying Current Metrics
  Use the `LIST METRICCURRENT` command to display the current metric values for a storage server.
- Using Metric History
  The metric history is a collection of past metric observations.

6.4.1.1 Displaying Metric Definitions

Use the `LIST METRICDEFINITION` command to display the metric definitions for a storage server.

A metric definition listing shows the configuration of a metric.

Example 6-15 Displaying Metric Definitions

This example shows how to display attributes for the `METRICDEFINITION` object.

```
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
```

Related Topics

- LIST METRICDEFINITION

6.4.1.2 Displaying Current Metrics

Use the `LIST METRICCURRENT` command to display the current metric values for a storage server.

A current metric listing shows a set of observations on the current value of an individual metric.

Example 6-16 Displaying Current Metric Values

This example shows how to display attributes for the `METRICCURRENT` object.

```
CellCLI> LIST METRICCURRENT CL_TEMP DETAIL

         name:                   CL_TEMP
         alertState:             normal
         collectionTime:         2009-12-17T15:32:25-08:00
         metricObjectName:       abcd2x3
         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_abcd2x3    1.9 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_01_abcd2x3    1.0 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_02_abcd2x3    4.1 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_03_abcd2x3    9.5 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_04_abcd2x3    0.1 MB  2009-12-17T15:46:52-08:00
         CD_IO_BY_R_LG   CD_05_abcd2x3    0.4 MB  2009-12-17T15:46:52-08:00
         ...
```

Related Topics

- DESCRIBE METRICCURRENT
- LIST METRICCURRENT

6.4.1.3 Using Metric History

The metric history is a collection of past metric observations.

The retention period for most metric history files is specified by the `metricHistoryDays` attribute associated with the storage server (`CELL`) or database server (`DBSERVER`). The default retention period is seven days. On each storage server, you can modify this setting with the CellCLI `ALTER CELL`. Likewise, you can change the default retention period on each database server with the DBMCLI `ALTER DBSERVER` command.

In addition to the metrics governed by the `metricHistoryDays` attribute, a subset of metric observations are retained for up to one year to support longer-term analysis and planning.

Starting with Oracle Exadata System Software 24.1.0, you can view and control which metrics to maintain for up to one year using the `retentionPolicy` attribute associated with each metric definition. When the `retentionPolicy` is `Default`, the retention period for the associated metric is governed by the `metricHistoryDays` attribute. If the `retentionPolicy` is `Annual`, the associated metric has a one-year retention period.

To view all metrics that may have up to one year of observation history maintained in the server, you can use the following command in CellCLI or DBMCLI:

```
LIST METRICDEFINITION WHERE retentionPolicy='Annual'
```

To modify the retention policy for a specific metric, you can use the `ALTER METRICDEFINITION` command and set the `retentionPolicy` attribute to `Annual` or `Default`. For example:

```
ALTER METRICDEFINITION CD_IO_BY_R_LG retentionPolicy='Annual'
```

To restore a metric to the original `retentionPolicy` setting defined in Oracle Exadata System Software, set the `retentionPolicy` attribute to an empty string (`''`). For example:

```
ALTER METRICDEFINITION CD_IO_BY_R_LG retentionPolicy=''
```

In all cases, regardless of the `retentionPolicy` and `metricHistoryDays` settings, historical metric observations are purged automatically if the server detects a shortage of storage space for the metric history repository.

To display the metric history, you can use the `LIST METRICHISTORY` command. Following are some examples:

Example 6-17 Displaying Metric History Values

The following command displays observations in the metric history for the `CD_IO_RQ_R_LG` cell metric when the metric was in a `critical` alert state. The output contains all of the metric attributes.

```
CellCLI> LIST METRICHISTORY CD_IO_RQ_R_LG WHERE alertState='critical' DETAIL
```

The following command displays a subset of metric attributes for all metric observations that are associated with cell disks. The output is limited to observations with non-zero metric values that were observed after the specified time.

```
CellCLI> LIST METRICHISTORY WHERE objectType = 'CELLDISK' AND metricValue != 
0   -
         AND collectionTime > '2009-08-12T09:10:51-07:00' ATTRIBUTES 
name,       -
         metricObjectName, metricValue, collectionTime
```

The following command displays all cell metric observations within a specified time period that are associated with the specified object types.

```
CellCLI> LIST METRICHISTORY WHERE objectType LIKE 'CELLDISK|FLASHCACHE|
FLASHLOG|SMARTIO|IORM_DATABASE' -
         AND collectionTime > '2020-07-15T08:00:00-07:00' -
         AND collectionTime < '2020-07-15T09:00:00-07:00'
```

Related Topics

- DESCRIBE METRICHISTORY
- LIST METRICHISTORY
- LIST METRICDEFINITION
- ALTER METRICDEFINITION
- ALTER CELL
- DESCRIBE CELL

6.4.2 Monitoring Alerts

You can monitor and receive notifications for alerts.

Alerts represent events of importance occurring within the cell. Typically, alerts indicate that Exadata server functionality is compromised or in danger of failure.

To receive notifications, use the `ALTER CELL` command.

- Displaying Alert Definitions
  Use the `LIST ALERTDEFINITION` command to display the alert definitions for the storage server.
- Receiving Alert Notifications
  Administrators for Oracle Exadata System Software can receive alert notifications by email or by Simple Network Management Protocol (SNMP) trap alerts.

- Displaying Alert History
  Use the `LIST ALERTHISTORY` command to display the alert history that has occurred on a cell.
- Modifying Alert History
  Use the `ALTER ALERTHISTORY` command to update the alert history for the cell.

Related Topics

- Exadata Alerts
  Alerts draw attention to potential and actual problems and other events of interest to an administrator.
- ALTER CELL

6.4.2.1 Displaying Alert Definitions

Use the `LIST ALERTDEFINITION` command to display the alert definitions for the storage server.

An alert definition provides a definition for every alert that can be produced on a storage server.

Example 6-18 Listing Alert Definition Attributes

This example shows how to display a detailed list of attributes for the alert `ADRAlert`.

```
CellCLI> LIST ALERTDEFINITION ADRAlert DETAIL
         name:                   ADRAlert
         alertShortName:         ADR
         alertSource:            "Automatic Diagnostic Repository"
         description:            "Incident Alert"
```

Example 6-19 Listing Alert Definition Name and Description Attributes

You can display a list of specified attributes for an alert definition. This example shows how to display 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.

```
CellCLI> LIST ALERTDEFINITION ATTRIBUTES name, metricName, description
    ADRAlert                                        "Incident Alert"
    HardwareAlert                                   "Hardware Alert"
    MetricAlert                                     "Threshold Alert"
    SoftwareAlert                                   "Software 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"
```

Related Topics

- Displaying Metric Definitions
  Use the `LIST METRICDEFINITION` command to display the metric definitions for a storage server.
- DESCRIBE ALERTDEFINITION
- LIST ALERTDEFINITION

6.4.2.2 Receiving Alert Notifications

Administrators for Oracle Exadata System Software can receive alert notifications by email or by Simple Network Management Protocol (SNMP) trap alerts.

By using the `ALTER CELL` command, you can set the alert notification attributes to configure Oracle Exadata Storage Server to send notification email messages or SNMP trap alerts.

Use of SNMP alerts allows Oracle Exadata Storage Servers to be monitored by a management application, such as Oracle Enterprise Manager Cloud Control.

Note:

The SNMP alerts conform to a MIB (management information base) which is included with each installation of Oracle Exadata System Software. The MIB file on Oracle Exadata Storage Server is available at `/opt/oracle/cell/cellsrv/deploy/config/cell_alert.mib`.

Related Topics

- ALTER CELL
- Alert Notification Attributes

6.4.2.3 Displaying Alert History

Use the `LIST ALERTHISTORY` command to display the alert history that has occurred on a cell.

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.

Example 6-20 Listing Alert History Attributes

This example 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.

```
CellCLI> LIST ALERTHISTORY WHERE severity = 'critical' AND examinedBy = ''
DETAIL
```

Related Topics

- DESCRIBE ALERTHISTORY
- LIST ALERTHISTORY
- ALTER CELL
- DESCRIBE CELL

6.4.2.4 Modifying Alert History

Use the `ALTER ALERTHISTORY` command to update the alert history for the cell.

Example 6-21 Altering Alert History Attributes

This example shows how to set the `examinedBy` attribute to the user ID of the administrator that examined the alert.

```
CellCLI> ALTER ALERTHISTORY 1671443714 examinedBy="jdoe"
```

The `examinedBy` attribute is the only `ALERTHISTORY` attribute that can be modified.

Related Topics

- ALTER ALERTHISTORY
- DESCRIBE ALERTHISTORY

6.4.3 Displaying Active Requests

Use the `LIST ACTIVEREQUEST` command to display the active requests for the storage server.

To view the `ACTIVEREQUEST` attributes, use the `DESCRIBE ACTIVEREQUEST` command.

Example 6-22   Listing Active Request Attributes

This example shows how to display a detailed list of attributes for a specified request I/O type.

```
CellCLI> LIST ACTIVEREQUEST WHERE IoType = 'predicate pushing' DETAIL
```

Related Topics

- LIST ACTIVEREQUEST
- DESCRIBE ACTIVEREQUEST

6.4.4 Using Real-Time Insight

You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems using an external metric collection platform.

- Using Fine-Grained Metrics
- Uploading Metric Observations to a Collection Platform
- Downloading Metric Observations from an Exadata Server
- Tagging Metric Observations
- Understanding the Metric Stream Format

6.4.4.1 Using Fine-Grained Metrics

Traditionally, Exadata metric collections occur at 1-minute intervals. However, real-time monitoring requires more timely metrics. Commencing with Oracle Exadata System Software 22.1.0, you can configure fine-grained metric collection.

Fine-grained metric collection is the foundation for the Real-Time Insight feature. Fine-grained metric collection works in conjunction with an external metric collection platform, where the fine-grained metric observations are collected and processed for visualization. Fine-grained metric observations reside only in server memory and are not persisted on the server. Consequently, no fine-grained metric history is maintained on each server.

- Controlling Fine-Grained Metric Collection
- Customizing Fine-Grained Metric Collection

6.4.4.1.1 Controlling Fine-Grained Metric Collection

The `metricFGCollIntvlInSec` attribute controls fine-grained metric collection.

- To enable fine-grained metric collection, you must set the collection interval to a value between 1 and 60 seconds.

  For example:

  ```
  CellCLI> ALTER CELL metricFGCollIntvlInSec=1
  ```

  The `metricFGCollIntvlInSec` setting is related to the automatic upload frequency specified in the `metricStreamIntvlInSec` attribute. When automatic metric upload and fine-grained collection are both enabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec>0`), `metricStreamIntvlInSec` must be between 5 and 30 times `metricFGCollIntvlInSec`. For example, if `metricStreamIntvlInSec=60`, then `metricFGCollIntvlInSec` must be between 2 and 12.

- To disable fine-grained metric collection on a server, set `metricFGCollIntvlInSec=0`.

  For example:

  ```
  CellCLI> ALTER CELL metricFGCollIntvlInSec=0
  ```

  Fine-grained metric collection can be disabled only when automatic metric upload is disabled (`metricStreamIntvlInSec=0`) or the automatic upload frequency is between 5 and 30 minutes (`metricStreamIntvlInSec` is between 300 and 1800).

6.4.4.1.2 Customizing Fine-Grained Metric Collection

By default, a set of key performance metrics is automatically enabled for fine-grained collection. But, you can customize fine-grained metric collection by enabling or disabling specific metrics.

- To enable a metric for fine-grained collection, use the `ALTER METRICDEFINITION` command and specify `finegrained=enabled`. For example:

  ```
  CellCLI> ALTER METRICDEFINITION N_NIC_KB_TRANS_SEC finegrained=enabled
  ```

  ```
  CellCLI> ALTER METRICDEFINITION N_MB_SENT,N_MB_RECEIVED finegrained=enabled
  ```

  ```
  CellCLI> ALTER METRICDEFINITION finegrained=enabled WHERE name LIKE
  'N_NIC.*'
  ```

- To disable a metric for fine-grained collection, use the `ALTER METRICDEFINITION` command and specify `finegrained=disabled`. For example:

  ```
  CellCLI> ALTER METRICDEFINITION N_MB_SENT finegrained=disabled
  ```

- At any time, to view the metrics enabled for fine-grained collection, use the following command:

  ```
  CellCLI> LIST METRICDEFINITION WHERE finegrained=enabled
  ```

- At any time, to view all of the metric definition details, including each metric description and whether the metric is enabled for fine-grained collection, use the following command:

  ```
  CellCLI> LIST METRICDEFINITION DETAIL
  ```

6.4.4.2 Uploading Metric Observations to a Collection Platform

You can enable an Exadata server to automatically upload (push) metric observations to an external metric collection platform.

- Controlling the Automatic Metric Upload Frequency
- Customizing the Metric Stream
- Configuring the Endpoints for Automatic Metric Upload

6.4.4.2.1 Controlling the Automatic Metric Upload Frequency

The `metricStreamIntvlInSec` attribute sets the upload interval (in seconds) for automatic uploads to the metric streaming endpoints specified by the `metricStreamEndPoint` attribute.

- To enable automatic metric uploads, set the `metricStreamIntvlInSec` attribute to a non-zero value.

  For example:

  ```
  CellCLI> ALTER CELL metricStreamIntvlInSec=25
  ```

  The `metricStreamIntvlInSec` setting is related to the fine-grained collection frequency specified in the `metricFGCollIntvlInSec` attribute:

  – When automatic metric upload and fine-grained collection are both enabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec>0`), `metricStreamIntvlInSec` must be between 5 and 30 times `metricFGCollIntvlInSec`. For example, if `metricFGCollIntvlInSec` is set to 5, then `metricStreamIntvlInSec` must be between 25 and 150.

  – When automatic metric upload is enabled and fine-grained collection is disabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec=0`), the automatic upload frequency must be between 5 and 30 minutes (`metricStreamIntvlInSec` must be between 300 and 1800).

- To disable automatic metric uploads, set `metricStreamIntvlInSec=0`.

  For example:

  ```
  CellCLI> ALTER CELL metricStreamIntvlInSec=0
  ```

6.4.4.2.2 Customizing the Metric Stream

By default, a set of key performance metrics is automatically enabled for streaming. But, you can customize the metric stream by enabling or disabling specific metrics.

- To include a metric in the metric stream, use the `ALTER METRICDEFINITION` command and specify `streaming=enabled`. For example:

  ```
  CellCLI> ALTER METRICDEFINITION N_NIC_KB_TRANS_SEC streaming=enabled
  ```

  ```
  CellCLI> ALTER METRICDEFINITION N_MB_SENT,N_MB_RECEIVED streaming=enabled
  ```

  ```
  CellCLI> ALTER METRICDEFINITION streaming=enabled WHERE name LIKE 'N_NIC.*'
  ```

- To remove a metric from the metric stream, use the `ALTER METRICDEFINITION` command and specify `streaming=disabled`. For example:

  ```
  CellCLI> ALTER METRICDEFINITION N_MB_SENT streaming=disabled
  ```

- At any time, to view the metrics that are included in the metric stream, use the following command:

  ```
  CellCLI> LIST METRICDEFINITION WHERE streaming=enabled
  ```

- At any time, to view all of the metric definition details, including each metric description and whether the metric is included in the metric stream, use the following command:

  ```
  CellCLI> LIST METRICDEFINITION DETAIL
  ```

6.4.4.2.3 Configuring the Endpoints for Automatic Metric Upload

You can automatically upload (push) the metric stream to one or more collection endpoints by setting the `metricStreamEndPoint` attribute as follows:

```
metricStreamEndPoint[+]=((host="endpoint-URL"[,type="stream-format"]
[,token="authentication-token"][,{httpProxy|httpsProxy}="proxy-server"])
                              [,(host="endpoint-URL"[,type="stream-format"]
[,token="authentication-token"][,{httpProxy|httpsProxy}="proxy-server"])]...)
```

In the `metricStreamEndPoint` definition:

- `host`: Specifies the URL for the collection endpoint. The URL can use HTTP or HTTPS.
- `type`: Optionally specifies the format of the stream. Supported values are:
  - `json`: Provides the stream in a JSON format
  - `plaintext`: Provides the stream in a plain text format

  The default value is `json`.
- `token`: Optionally specifies the authentication token for the collection endpoint. Consult the metric collection platform for details about generating the token.
- `httpProxy` or `httpsProxy`: Optionally specifies a proxy server to facilitate network connectivity to the collection endpoint. A proxy server is required if a firewall resides between the Exadata system and the collection endpoint.

You can use the optional `+=` operator to add collection endpoints to an existing `metricStreamEndPoint` definition. Otherwise, the `=` operator overwrites the previous attribute value.

Example 6-23 Setting Up a JSON Stream

This example shows how to set up a JSON stream. In the example command, the `host` and `token` values come from the collection platform and we assume that network connectivity is through the specified proxy.

```
CellCLI> ALTER CELL metricStreamEndPoint=((host="https://
ingest.stream.example.com/v2/
datapoint",type="json",token="wcfA_**********Z58QpKg",httpProxy="www-
proxy.example.com:80"))
```

Example 6-24 Adding a Plain Text Endpoint

This example shows how to add a plain text endpoint to the existing `metricStreamEndPoint`. In the example command, the `host` and `token` values come from the collection platform. The example also assumes that the collection platform is within the corporate network requiring no network proxy.

```
CellCLI> ALTER CELL metricStreamEndPoint+=((host="http://
idbsrv.example.com:8086/api/v2/write?
org=Exadata&bucket=Metrics&precision=ns",type="plaintext",token="6unif********
**rOXwtfkG0gWGENyePd6uN6OLR_deTZL4IuG9VTfDWwvpB-QvJcCcFs_NVjmpsyANz0Q8psA=="))
```

6.4.4.3 Downloading Metric Observations from an Exadata Server

You can download (pull) the metric stream from an Exadata server by using the provided REST endpoint. This includes all metrics enabled for streaming (`streaming=enabled`), regardless of whether the system is enabled for fine-grained metric collection or configured for automatic metric upload.

On each storage server, the REST endpoint URL is:

```
https://server-name/metricstream/list?stream=true
```

On each database server, the endpoint uses port 7879. Consequently, the URL is:

```
https://server-name:7879/metricstream/list?stream=true
```

For maximum efficiency, the download interval should be a multiple of the metric collection interval. If fine-grained metric collection is enabled (`metricFGCollIntvlInSec>0`) on the server, then coordinate the download frequency with the fine-grained collection interval. Otherwise, coordinate the download frequency with the standard 1-minute collection interval.

To facilitate access to the metric stream, you should use a dedicated user account, which only has access to the stream. You can use the following command sequence to appropriately configure a user account in CellCLI, which you can then use for authentication to the REST endpoint. In the command sequence, substitute your own user and role names.

```
CREATE ROLE metric_collector_role
GRANT PRIVILEGE LIST ON METRICSTREAM ALL ATTRIBUTES WITH ALL OPTIONS TO ROLE
metric_collector_role
CREATE USER metric_collector PASSWORD=<password>
GRANT ROLE metric_collector_role TO USER metric_collector
```

6.4.4.4 Tagging Metric Observations

In each Exadata database server and storage server you can define a set of metric tags, which are included in every observation in the metric stream. These tags can help you to organize and group observations generated by numerous Exadata servers.

You can configure metric tags by setting the `metricStreamTags` attribute to a valid JSON string containing tag and value pairs as follows:

```
metricStreamTags='{"tag1":"value1"[,"tag2":"value2"]...}'
```

For example:

```
CellCLI> ALTER CELL
metricStreamTags='{"application":"personnel","department":"HR"}'
```

6.4.4.5 Understanding the Metric Stream Format

Real-time metric observations contain a core set of attributes. However, the format of the metric stream depends on the mode of access.

If you are automatically uploading the metric stream to a metric collection platform, you can specify the metric stream format in the metric endpoint configuration. The available formats are JSON or plain text.

Following is an example of the JSON format:

```
json: {
    "gauge": [{
            "metric": "OS_NET_RX_BY_SEC",
            "value": "0.0012989044189453125",
            "timestamp": 1652473286000,
            "unit": "MB/sec",
            "dimensions": {
                "server": "celadm09.example.com",
                "objectName": "eth0",
                "nodeType": "STORAGE",
                "fleet": "example-fleet",
                "pod": "dbm01",
                "cluster": "c01"
            }
        }, {
            "metric": "SIO_IO_RD_FC_HD_SEC",
            "value": "0.0",
            "timestamp": 1652473286000,
            "unit": "MB/sec",
            "dimensions": {
                "server": "celadm09.example.com",
                "objectName": "SMARTIO",
                "nodeType": "STORAGE",
                "fleet": "example-fleet",
                "pod": "dbm01",
                "cluster": "c01"
            }
        }
    ]
}
```

The plain text format contains essentially the same information as the JSON stream. However, with the plain text format, each metric observation is presented on a separate line. Following is an example of the plain text format:

```
metrics,name=OS_NET_RX_BY_SEC,objectName=eth0,server=celadm09.example.com,unit
=MB/sec,nodeType=STORAGE,fleet=example-fleet,pod=dbm01,cluster=c01
value=9.441184615324398E-4 1652473456000000000
metrics,name=OS_NET_RX_BY_SEC,objectName=eth0,server=celadm09.example.com,unit
```

```
=MB/sec,nodeType=STORAGE,fleet=example-fleet,pod=dbm01,cluster=c01
value=0.002647613311980988 1652473457000000000
```

If you are downloading the metric stream by using the provided REST endpoint, the data is presented in a format similar to the plan text upload format where each metric observation is presented on a separate line. Following is an example of the download format:

```
DS_CPUT{objectName="dbadm05",unit="%",server="dbadm05.example.com",nodeType="K
VMHOST",fleet="example-fleet",pod="dbm01",cluster="c01"} 23.10906363831155
1652485449597
DS_MEMUT{objectName="dbadm05",unit="%",server="dbadm05.example.com",nodeType="
KVMHOST",fleet="example-fleet",pod="dbm01",cluster="c01"} 99 1652485449597
DS_MEMUT_MS{objectName="dbadm05",unit="%",server="dbadm05.example.com",nodeTyp
e="KVMHOST",fleet="example-fleet",pod="dbm01",cluster="c01"}
0.12396045794483294 1652485449597
```

The following list describes the attributes contained in the metric stream:

- The metric name is identified as follows:
  - In the JSON upload format, the metric name follows the `metric` tag.
  - In the plain text upload format, the metric name is the value following `name=`.
  - In the download format, the metric name is the first element on each line, preceding the left braces (`{`).
- The metric value is located as follows:
  - In the JSON upload format, the metric value follows the `value` tag.
  - In the plain text upload format, the metric value follows `value=`.
  - In the download format, the metric value is the second last element on each line, following the right braces (`}`).
- The time of the metric observation is located as follows:
  - In the JSON upload format, the timestamp follows the `timestamp` tag. The timestamp is expressed as the number of milliseconds (1 x 10^{-3} sec) since January 1, 1970, 00:00:00 GMT.
  - In the plain text upload format, the timestamp is the last element on each line. The timestamp is expressed as the number of nanoseconds (1 x 10^{-9} sec) since January 1, 1970, 00:00:00 GMT.
  - In the download format, the timestamp is the last element on each line. The timestamp is expressed as the number of milliseconds (1 x 10^{-3} sec) since January 1, 1970, 00:00:00 GMT.
- The `unit` value describes the unit of measurement for the metric observation.
- The `server` value contains the name of the Exadata server that generated the metric observation.
- The `objectName` value contains the name of the Exadata object associated with the metric.
- The `nodeType` value contains the type of the Exadata server that generated the metric observation.

- The `fleet`, `pod`, and `cluster` attributes are examples of user-defined metric tags, which you can use to organize and group observations generated by numerous Exadata servers. You can tag metrics by setting the `metricStreamTags` CELL attribute.

Related Topics

- Tagging Metric Observations

7

Using the CellCLI Utility

You use the Cell Control Command-Line Interface (CellCLI) utility to manage Oracle Exadata System Software.

CellCLI provides many of the features that are provided with SQL*Plus, including the use of script files.

- Overview of the CellCLI Utility
  The Cell Control Command-Line Interface (CellCLI) utility is the command-line administration tool for Oracle Exadata System Software.
- About CellCLI Administration Commands
  CellCLI administrative commands do not act directly on objects.
- About CellCLI Object Commands
  This topic describes the CellCLI object commands, object types, and object attributes.
- About CellCLI Object Types
  There are several Oracle Exadata System Software object types that can be used with CellCLI object commands.
- About Quoting Object Names
  If an object name has characters that cause parsing errors enclose the object name in quotes.
- About CellCLI Object Attributes
  Each CellCLI object has a set of attributes that are assigned when the object is created or altered.
- CellCLI Command Reference
  CellCLI has both administrative and object commands.

Related Topics

- Using the dcli Utility
  The dcli utility facilitates centralized management across Oracle Exadata Database Machine.

7.1 Overview of the CellCLI Utility

The Cell Control Command-Line Interface (CellCLI) utility is the command-line administration tool for Oracle Exadata System Software.

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 Oracle Exadata Storage Server is shipped.

- 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).
- Understanding Command Syntax and Options for CellCLI
  This topic describes the syntax and command options for CellCLI.

- Reserved Words
  If any of the reserved keywords are used as values in commands, then they must be enclosed in quotation marks.
- CellCLI Command-Line Editing
  Most of the command editing features of CellCLI are similar to modern shells, such as `bash` and `tcsh`.
- CellCLI Input and Output Options
  Oracle Exadata System Software command-line utilities read commands from standard input and write output to standard output.
- Comments in CellCLI Scripts
  You can add single-line comments to CellCLI scripts using multiple formats.
- Line Continuation in CellCLI Commands
  To continue a long command on to the next line, insert a hyphen (-) at the end of the line.

7.1.1 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).

Syntax

The command-line syntax is as follows:

```
cellcli [-n] [-m] [-x] [ --xml | --json ] [ -e command ]
```

Options

The following options can be used with the CellCLI command:

- `-n` — Runs the CellCLI utility in non-interactive mode. This option suppresses the command prompt and disables the command-line editing features.
- `-m` — Runs CellCLI in monitor (read-only) mode.
- `-x` — Suppresses the banner.
- `--xml` — Causes command output to be displayed in XML format.
- `--json` — Causes command output to be displayed in JSON format.
- `-e` *`command`* — Runs the specified CellCLI command. For example:

  ```
  $ cellcli -e list cell detail
  $ cellcli -e "list celldisk attributes name where name   -
    like '.*cell01'"
  ```

  CellCLI exits after running the command.

Authentication

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.

7.1.2 Understanding Command Syntax and Options for CellCLI

This topic describes the syntax and command 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. If you are using hyphens in names or to denote negative values, it must be immediately followed by an alphanumerical value.

7.1.3 Reserved Words

If any of the reserved keywords are used as values in commands, then they must be enclosed in quotation marks.

The following are CellCLI reserved words:

```
ABORT
ACTIVE
ACTIVEREQUEST
ALERTDEFINITION
ALERTHISTORY
ALL
ALTER
ASSIGN
BBU
BMC
CALIBRATE
CATPLAN
CELL
CONFIGUREBMC
CREATE
```

```
DBPLAN
DESCRIBE
DETAIL
DROP
EXPORT
FLASHCACHE
FLASHCACHECONTENT
FOR REPLACEMENT
FORCE
GRIDDISK
IGNORE REDUNDANCY
IMPORT
INACTIVE
IORMPLAN
KEY
LED
LIST
LUN
MAIL
MEMORY
METRICDEFINITION
METRICCURRENT
METRICHISTORY
MS
NULL
OFF
ON
PHYSICALDISK
PRIVILEGE
REALM
RESTART
RS
RULE
SHUTDOWN
SNMP
STARTUP
THRESHOLD
USER
VALIDATE
```

7.1.4 CellCLI Command-Line Editing

Most of the command editing features of CellCLI are similar to modern shells, such as `bash` and `tcsh`.

The CellCLI utility supports command-line history and editing, similar to Berkeley Software Distribution (BSD) `editline` and GNU `readline` functionality.

7.1.5 CellCLI Input and Output Options

Oracle Exadata System Software 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.

7.1.6 Comments in CellCLI Scripts

You can add single-line comments to CellCLI scripts using multiple 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
```

7.1.7 Line Continuation in CellCLI Commands

To continue a long command on to the next line, insert a hyphen (-) at the end of the line.

After inserting the hyphen, press **Enter**, and continue typing the command.

For example:

```
CellCLI> LIST CELLDISK WHERE name LIKE 'CD_04.*' -
         ATTRIBUTES name, status, comment
```

7.2 About CellCLI Administration Commands

CellCLI administrative commands do not act directly on objects.

The following administration commands are available with CellCLI:

- EXIT
- HELP
- QUIT
- SET
- SPOOL
- START and @

Note:

The `celladmin` user should be used to run all services on the cell.The `cellmonitor` user is for monitoring purposes. The `cellmonitor` user can run the following commands:

- `DESCRIBE`
- `EXIT`
- `HELP`
- `LIST`
- `REMARK`
- `SET`
- `START`

7.3 About CellCLI Object Commands

This topic describes the CellCLI object commands, object types, and object attributes.

The following CellCLI commands operate on Oracle Exadata System Software objects:

- ALTER
- ASSIGN KEY
- CALIBRATE
- CREATE
- DESCRIBE
- DROP
- EXPORT CELLDISK
- GRANT
- IMPORT CELLDISK
- LIST
- REVOKE

7.4 About CellCLI Object Types

There are several Oracle Exadata System Software object types that can be used with CellCLI object commands.

- `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.
- `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.
- `CLUSTER` — An Oracle Grid Infrastructure (GI) cluster, which may contain various Oracle databases.
- `DATABASE` — Database refers to an active database instance.
- `DIAGPACK` — A diagpack represents a compressed file under `$LOG_HOME` and contains log files and trace files.
- `FLASHCACHE` — Flash storage allocated to Exadata Smart Flash Cache.
- `FLASHCACHECONTENT` — List of all objects currently cached in Exadata Smart Flash Cache.
- `FLASHLOG` — The portion of flash storage allocated for storing the Exadata Smart Flash Log.
- `GRIDDISK` — A grid disk is a logical partition of a cell disk. It is exposed on the Oracle Exadata Storage Server network to the database hosts, where it can be consumed by Oracle ASM as part of a disk group.
- `IBPORT` — The InfiniBand Network Fabric ports for Oracle Exadata Storage Server. Does not apply to Oracle Exadata X8M systems.
- `IORMPLAN` — An I/O Resource Management (IORM) interdatabase plan is a set of directives that determines allocation of I/O resources to database clients. There is one plan for the cell.
- `IORMPROFILE` — IORM interdatabase plans support profiles to ease management, and configuration of interdatabase plans for hundreds of databases. Profiles introduce a way to allocate I/O resources for a database.
- `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` — Describes the most recent (most current) metric observation.
- `METRICDEFINITION` — Describes the configuration of a metric.
- `METRICHISTORY` — Describes the collection of past (historical) metric observations.
- `METRICSTREAM` — Describes the set of metrics configured for fine-grained collection and Real-Time Insight.
- `OFFLOADGROUP` — An object that contains modifiable attributes of offload groups, and can be used to restart, start up, and shut down services.
- `OFFLOADPACKAGE` — An object that contains attributes describing offload support on the cell, including the installation time.
- `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.
- `PLUGGABLEDATABASE` — This object describes tenant databases known to the cell.

- `PMEMCACHE` — The portion of persistent memory (PMEM) storage allocated for use as a cache.

  > **Note:**
  >
  > Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies. In this release, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache.

- `PMEMLOG` — The portion of persistent memory (PMEM) storage allocated for caching redo log data.

  > **Note:**
  >
  > Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies. In this release, the persistent memory commit accelerator, previously known as PMEM log, is now called XRMEM log.

- `PRIVILEGE` — A right or permission assigned to a role.
- `QUARANTINE` — A quarantine stops faulty SQL statements from performing a Smart Scan. This reduces software crashes, and improves storage availability.
- `ROLE` — A named group of related privileges.
- `SOFTWAREHISTORY` — A list of final states for past software updates.
- `SOFTWAREUPDATE` — An object that contains the software location and time parameters for scheduling software updates
- `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.
- `USER` — A person allowed access to the storage servers.
- `XRMEMCACHE` — Memory allocated for use as the Exadata RDMA Memory Cache (XRMEM cache).

  > **Note:**
  >
  > Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies.

- XRMEMLOG — The portion of persistent memory (PMEM) storage allocated for caching redo log data.

> **Note:**
>
> Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies. In this release, the persistent memory commit accelerator, previously known as PMEM log, is now called XRMEM log.

Not all possible command-object combinations are valid. For valid command-object combinations, review the syntax for the specific object command.

Related Topics

- About CellCLI Object Commands
  This topic describes the CellCLI object commands, object types, and object attributes.
- Oracle Exadata System Software Components
  This section provides a summary of the following Oracle Exadata System Software components.

7.5 About Quoting Object Names

If an object name has characters that cause parsing errors enclose the object name in quotes.

The object name can be a number or string in commands that have the following format:

```
<verb> <object_type> <object_name>
```

For example, consider the following command:

```
CellCLI> list pluggabledatabase  CDB$ROOT
                                    ^
CELL-01504: Invalid command syntax.
```

LIST is the verb, PLUGGABLEDATABASE is the object type, and CDB$ROOT is the object name. In the above example, the parser throws an error when confronted by the $ character in the object name.

For object names that contain invalid characters, you need to surround the object name with either single or double quotes to force the parser to treat the object name as a string.

To avoid this error, surround the object name with either single or double quotes:

```
CellCLI> LIST PLUGGABLEDATABASE 'CDB$ROOT'
         CDB$ROOT
```

7.6 About 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.

- Restrictions on Values of Common Attributes
  Review the following restrictions for the values of attributes used by multiple CellCLI objects.
- Attribute Lists in LIST Command
  You can specify which attributes to display for the `LIST` command with the optional `ATTRIBUTES` clause.
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

Related Topics

- LIST
- DESCRIBE

7.6.1 Restrictions on Values of Common Attributes

Review the following restrictions for the values of attributes used by multiple CellCLI objects.

- The value of the `name` attribute must be less than 256 characters and composed only of the following ASCII characters:
  - Lowercase alphabetic characters (`a` to `z`)
  - Uppercase alphabetic characters (`A` to `Z`)
  - Numbers (`0` to `9`)
  - Underscore (_)
  - Hyphen (-)

  Note:

  On Oracle Exadata System Software release 19.2.0, or earlier, you must enclose the string in double quotes. For example: `"hyphenated-string"`

- 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.

7.6.2 Attribute Lists in LIST Command

You can specify which attributes to display for the `LIST` command with the optional `ATTRIBUTES` clause.

This syntax of the `ATTRIBUTES` clause is:

```
ATTRIBUTES { ALL | attribute1 [, attribute2] ... }
```

`ALL` displays all possible object attributes for the `LIST` object.

Example 7-1 Listing METRICHISTORY for Specific Attributes

This example shows the `LIST METRICHISTORY` command with the `name` and `metrictype` attributes specified, and the output.

```
LIST METRICHISTORY ATTRIBUTES name, metrictype
         CL_CPUT            Instantaneous
         CL_FANS            Instantaneous
         CL_RUNQ            Instantaneous
         CL_TEMP            Instantaneous
         N_NIC_RCV_SEC      Rate
         N_NIC_TRANS_SEC    Rate
...
```

7.6.3 Attribute Filters in LIST and ALTER Commands

You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

This syntax of the *attribute_filters* clause is:

```
WHERE attribute_filter1 [ AND attribute_filter2 ... ]
```

Each *attribute_filterN* has the following syntax:

```
attribute [ NOT | ! ] operator comparison_value
```

The *attribute* placeholder represents the name of the attribute to use for filtering. The supported types of *operator* are listed in the following table. These operators can be combined with `NOT` or `!`.

Table 7-1 Supported Operators in Attribute Filters

Operator	Description
=	Tests for equality between string, status, or numeric attributes. For example: `status NOT = normal`

Table 7-1 (Cont.) Supported Operators in Attribute Filters

Operator	Description
`>`	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.*'`

When used with the supported operators, *comparison_value* is one of the following data 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

7.7 CellCLI Command Reference

CellCLI has both administrative and object commands.

The following commands are available with the CellCLI utility:

- ALTER
- ASSIGN KEY
- CALIBRATE
- CREATE
- DESCRIBE
- DROP
- EXIT
- EXPORT CELLDISK
- GRANT
- HELP
- IMPORT CELLDISK
- LIST
- QUIT
- REVOKE
- SET

- SPOOL
- START and @

7.7.1 ALTER

Purpose

The `ALTER` command performs an action on or changes attributes of a single cell object or multiple Oracle Exadata System Software 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

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.

- ALTER ALERTHISTORY
- ALTER CELL
- ALTER CELLDISK
- ALTER FLASHCACHE
- ALTER GRIDDISK
- ALTER IBPORT
- ALTER IORMPLAN
- ALTER LUN
- ALTER METRICDEFINITION
- ALTER OFFLOADGROUP
- ALTER PHYSICALDISK
- ALTER PMEMCACHE
- ALTER QUARANTINE
- ALTER SOFTWAREUPDATE
- ALTER THRESHOLD
- ALTER USER
- ALTER XRMEMCACHE

Related Topics

- About CellCLI Object Types
  There are several Oracle Exadata System Software object types that can be used with CellCLI object commands.

7.7.1.1 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.

Example 7-2   Altering ALERTHISTORY Attributes

This example 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.

```
CellCLI> ALTER ALERTHISTORY 1671443714 -
                            examinedBy="jdoe"

CellCLI> ALTER ALERTHISTORY ALL examinedBy="jdoe"
```

7.7.1.2 ALTER CELL

Purpose

The `ALTER CELL` command changes the attributes of the cell.

Syntax

```
ALTER CELL  {
  | SHUTDOWN SERVICES { RS | MS | CELLSRV | ALL } [IGNORE REDUNDANCY]
  | RESTART SERVICES  { RS | MS | CELLSRV | ALL } [IGNORE REDUNDANCY]
  | RESTART BMC
  | STARTUP SERVICES  { RS | MS | CELLSRV | ALL }
  | LED {ON | OFF}
  | DONOTSERVICELED {ON | OFF [FORCE]}
  | VALIDATE { MAIL | SNMP | CONFIGURATION }
  | VALIDATE SYSLOGCONF selector.node
  | CONFIGUREBMC
  | BBU { DROP FOR REPLACEMENT | REENABLE }
  | attribute_name = attribute_value
        [, attribute_name = attribute_value]...
  }
```

- ALTER CELL Commands for Managing Services
- ALTER CELL Commands for Managing Exadata Storage Server Hardware

- ALTER CELL Commands for Configuration Validation
- ALTER CELL Commands for Setting Attributes

7.7.1.2.1 ALTER CELL Commands for Managing Services

Syntax

```
ALTER CELL {
    SHUTDOWN SERVICES { RS | MS | CELLSRV | ALL } [IGNORE REDUNDANCY]
  | RESTART SERVICES  { RS | MS | CELLSRV | ALL } [IGNORE REDUNDANCY]
  | STARTUP SERVICES  { RS | MS | CELLSRV | ALL }
}
```

Usage Notes

The following table lists the arguments and options for the `ALTER CELL` commands that perform service management operations:

Argument	Description
`SHUTDOWN SERVICES {RS | MS}`	Shuts down the Restart Server or Management Server service.
`SHUTDOWN SERVICES CELLSRV [IGNORE REDUNDANCY]`	Shuts down the Cell Server service. If you include `IGNORE REDUNDANCY`, then the service is immediately stopped without waiting on the redundancy checks from Oracle ASM.
`SHUTDOWN SERVICES ALL`	Shuts down all services (Restart Server, Management Server, and Cell Server). If you include `IGNORE REDUNDANCY`, then the Cell Server service is immediately stopped without waiting on the redundancy checks from Oracle ASM.
`RESTART SERVICES {RS | MS}`	Stops and then starts the Restart Server or Management Server service.
`RESTART SERVICES CELLSRV [IGNORE REDUNDANCY]`	Stops and then starts the Cell Server service. If you include `IGNORE REDUNDANCY`, then the service is stopped without waiting on the redundancy checks from Oracle ASM.
`RESTART SERVICES ALL [IGNORE REDUNDANCY]`	Stops and then starts all services (Restart Server, Management Server, and Cell Server). If you include `IGNORE REDUNDANCY`, then the Cell Server service is stopped without waiting on the redundancy checks from Oracle ASM.
`STARTUP SERVICES {RS | MS | CELLSERV | ALL}`	Starts the specified service. If you use the keyword `ALL`, then all services are started.

The following are additional usage notes for the `ALTER CELL` commands that perform service management:

- During a shutdown operation affecting CELLSRV, the system first checks the status of the grid disks to ensure that it is safe to proceed. Specifically, the `asmDeactivationOutcome` attribute is checked for all of the grid disks. If the attribute value is `yes`, then the grid disk can be deactivated without data loss.

Depending on the command, the following occurs if the `asmDeactivationOutcome` attribute value is `yes` for all of the grid disks:

- For the `ALTER CELL SHUTDOWN SERVICES CELLSRV` and `ALTER CELL SHUTDOWN SERVICES ALL` commands:
    1. The grid disks are deactivated on the cell.
    2. Oracle ASM takes the corresponding ASM disks offline.
    3. Finally, the applicable services are shutdown.
- For the `ALTER CELL RESTART SERVICES CELLSRV` and `ALTER CELL RESTART SERVICES ALL` commands, the CELLSRV service is restarted immediately, followed by a restart of the MS and RS services, if applicable.

Otherwise, if the `asmDeactivationOutcome` attribute value is not `yes` for any grid disk, then the `CELL-01548` error message is displayed and the status of the services remains unchanged.

- The `IGNORE REDUNDANCY` option bypasses the `asmDeactivationOutcome` attribute checks. Using the `IGNORE REDUNDANCY` option results in immediate execution of the command. Consequently, if the command shuts down the only online copy of a grid disk, then the corresponding Oracle ASM disk group is dismounted.
- If the Restart Server (RS) service is not running, then you must run either `ALTER CELL STARTUP SERVICES RS` or `ALTER CELL RESTART SERVICES RS` before you can start other services individually, or you can run the `ALTER CELL STARTUP ALL` command.

Example 7-3 Starting Up and Shutting Down Cell Services

This example shows how to start up and shut down cell services.

```
CellCLI> ALTER CELL STARTUP SERVICES CELLSRV
CellCLI> ALTER CELL STARTUP SERVICES ALL

CellCLI> ALTER CELL SHUTDOWN SERVICES MS
CellCLI> ALTER CELL SHUTDOWN SERVICES CELLSRV IGNORE REDUNDANCY
CellCLI> ALTER CELL SHUTDOWN SERVICES ALL

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

7.7.1.2.2 ALTER CELL Commands for Managing Exadata Storage Server Hardware

Syntax

```
ALTER CELL {
    RESTART BMC
  | LED {ON | OFF}
  | DONOTSERVICELED {ON | OFF [FORCE]}
  | CONFIGUREBMC
  | BBU { DROP FOR REPLACEMENT | REENABLE }
  | attribute_name = attribute_value [, attribute_name = attribute_value]...
  }
```

Usage Notes

The following table lists the arguments and options for the `ALTER CELL` commands that perform Exadata storage server hardware management operations:

Argument	Options	Description
`RESTART BMC`	none	Restarts the Baseboard Management Controller (BMC).
`LED`	`ON` `OFF`	`LED ON` and `LED OFF` operations turn on and off the Fault-Service Required LED. You can manually light the LED to indicate that a cell requires maintenance. The LED also turns on automatically if a component fails.
`DONOTSERVICELED`	`ON` `OFF`	Turns the Do Not Service LED on and off. This LED is available with Oracle Exadata Database Machine X7 and later models.
`CONFIGUREBMC`	none	Configures the BMC for hardware alerts to the local cell so that Management Server (MS) can pick up the alerts.

`BBU`	`DROP FOR REPLACEMENT`
	`REENABLE`
		`BBU DROP FOR REPLACEMENT` drops the hard disk controller battery-backed unit (BBU).
		`BBU REENABLE` re-enables the BBU.

The following are additional usage notes for the `ALTER CELL` commands that perform storage server hardware management:

- The `ALTER CELL BBU DROP FOR REPLACEMENT` command is run prior to replacement of a hard disk controller battery. The command changes the caching policy from `writeback` to `writethrough`, and turns on the locator LED. The new battery is enabled automatically.
- The `ALTER CELL BBU REENABLE` command is run when a battery is removed and then the same battery is re-inserted. The command changes the caching policy from `writethrough` to `writeback`, and turns off the locator LED.

Attributes Related to Hardware Management

- 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.
- The `bbuLearnSchedule` attribute is used to set the next battery learn cycle. The following parameters are used with the `bbuLearnSchedule` attribute:
  - `month`: Values are `1` through `12`. The month entered must be within the current month and the next three months. For example, if the `bbuLearnSchedule` attribute is set in February, then the months could be February, March, April or May.
  - `week`: Values are `1` through `5`. The value `1` represents the first week of the month, `2` represents the second week, and so on. The `week` value must be specified when specifying `month` and `day`.
  - `day`: Values are `1` through `7`. The value `1` represents Sunday, `2` represents Monday, and so on. The `day` value must be specified when specifying `month` and `week`.

  – `date`: Values are `1` through `31`. The values represent the days of the month. The default date is `17`.
  – `hour`: Values are `0` through `23`. The value `0` represents 12:00 a.m., `1` represents 1:00 a.m., and so on.
  – `minute`: Values are `0` to `59`. The values represent the minutes in an hour.
  – `second`: Values are `0` to `59`. The values represent the seconds in a minute.
- The `ALTER CELL eighthRack=true` command enables or disables an Eighth Rack configuration on Oracle Exadata Database Machine X3-2 Quarter Racks or later. The options are `true` to enable the Eighth Rack configuration, and `false` to disable the Eighth Rack configuration. The `ALTER CELL eighthRack=true` command requires that there are no cell disks because enabling the Eighth Rack configures only half of the hard disks and flash capacity. After using this command you must restart Cell Server (CELLSRV) to make the new changes effective and prevent unexpected results.

Examples

Example 7-4 Setting the Cell LED Off and On

This example shows how to set the Fault-Service Required LED on and off for the cell.

```
CellCLI> ALTER CELL LED OFF
CellCLI> ALTER CELL LED ON
```

Example 7-5 Setting the Battery Learn Cycle

This example shows how to schedule for the battery learn cycle. In this example, the command sets the battery learn cycle to occur January 17 3:00:59, and then the following learn cycles are April 17 3:00:59, July 17 3:00:59, and October 17 3:00:59. The default setting is "`MONTH 1 DATE 17 HOUR 2 MINUTE 0`."

```
CellCLI> ALTER CELL bbuLearnSchedule = "MONTH 1 HOUR 3 SECOND 59"
```

7.7.1.2.3 ALTER CELL Commands for Configuration Validation

Syntax

```
ALTER CELL {
    VALIDATE { MAIL | SNMP | CONFIGURATION }
  | VALIDATE SYSLOGCONF selector.node
  }
```

Usage Notes

The following table lists the arguments and options for the `ALTER CELL` commands that perform configuration validation operations:

Argument	Description
`VALIDATE MAIL`	The `VALIDATE MAIL` operation sends a test message using the e-mail attributes configured for the cell.

Argument	Description
`VALIDATE SNMP`	The `VALIDATE SNMP` operation sends a test message using the SNMP attributes configured for the cell. The `VALIDATE SNMP TYPE=ASR` operation validates Oracle ASR on Oracle Exadata Storage Server.
`VALIDATE CONFIGURATION`	The `VALIDATE CONFIGURATION` operation validates the configuration. When the validation is complete and correct, the system responds with `Cell` *`cell_name`* `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.

Usage Notes

For more information about SYSLOG configuration, see SYSLOG Attributes.

Examples

Example 7-6 shows how to validate the e-mail setup on a cell.

Example 7-7 shows how to validate the Oracle ASR e-mail setup on a cell.

Example 7-8 shows how to validate the SNMP setup on a cell.

Example 7-9 shows how to validate the configuration on a cell.

Example 7-10 shows a sample error message when configuration on a cell is incorrect.

Example 7-6 Validating E-mail on a Cell

```
CellCLI> ALTER CELL VALIDATE MAIL
```

Example 7-7 Validating Oracle ASR E-mail on a Cell

```
CellCLI> ALTER CELL VALIDATE SNMP type=asr
```

Example 7-8 Validating SNMP on a Cell

```
CellCLI> ALTER CELL VALIDATE SNMP
```

Example 7-9 Validating Configuration on a Cell

```
CellCLI> ALTER CELL VALIDATE CONFIGURATION

Cell CD_01_cell01 successfully altered
```

Example 7-10 Checking an Incorrect Configuration on a Cell

```
CellCLI> ALTER CELL VALIDATE CONFIGURATION

CELL-02827: Cell configuration check for hardware and firmware encountered the
following issues:

ILOM check has detected the following issue(s):
    Attribute Name : ILOMVersion
    Required       : 3.0.6.10.a r49240
    Found          : 3.0.6.10.a r49385
```

7.7.1.2.4 ALTER CELL Commands for Setting Attributes

Syntax

```
ALTER CELL
    attribute_name = attribute_value
        [, attribute_name = attribute_value]...
```

Usage Notes

The attributes that can be changed using the `ALTER CELL` command are shown as `modifiable` in Example 7-96 or described below.

- Caching Attributes
- Alert Notification Attributes
- Alert Summary Attributes
- SYSLOG Attributes
- Disk Scrubbing Attributes
- Security Certificate Attributes
- Persistence Attributes
- Real-Time Insight Attributes
- Miscellaneous Attributes

7.7.1.2.4.1 Caching Attributes

Flash Cache Mode

The `flashCacheMode` attribute is used to display and set the current value for flash cache. The values are `writethrough` (the default) or `writeback`. Note the following about the `flashCacheMode` attribute:

- If the attribute is modified from `writeback` to `writethrough` and there is existing flash cache, then an error is displayed. The flash cache must be flushed and dropped before changing the attribute to `writethrough`.
- If the attribute is to be modified from `writethrough` to `writeback`, then flash cache must be dropped before modifying the attribute.
- Write back caching can be disabled on the grid disks that do not need caching, such as the grid disks in the RECO disk group. This allows other objects to use the cache space.

See Also:

- ALTER GRIDDISK for information about disabling caching on grid disks
- *Oracle Exadata Database Machine Maintenance Guide* for information about enabling and disabling flash cache

Flash Cache Compression

Flash cache compression is first available in Oracle Exadata System Software release 11.2.3.3.0, and only on Oracle Exadata Database Machine X3 and X4 storage servers.

The `ALTER CELL flashCacheCompress` command enables or disables flash cache compression. The options are `true` to enable flash cache compression, and `false` to disable flash cache compression. To enable flash cache compression on Oracle Exadata Database Machine X3 and X4 storage servers, use the following command:

```
CellCLI> ALTER CELL flashCacheCompress=true
```

Note:

- Oracle Advanced Compression Option is required to enable flash cache compression.
- The `flashCacheCompress` cell attribute is no longer available starting with Oracle Exadata System Software release 24.1.0.

RAM Cache Mode

The `ramCacheMode` attribute can be set to `on`, `off`, or `auto`. The default value is `auto`, which means the RAM Cache feature is not enabled. When you modify this attribute, you must restart CELLSRV for the change to take effect.

Note:

Commencing with the September 2022 Oracle Exadata System Software release updates (versions 22.1.3, 21.2.16, and later), the storage server RAM Cache feature is deprecated.

7.7.1.2.4.2 Alert Notification Attributes

Configuring Alert Notifications

To set up the cell to send notifications about alerts, you can configure the following cell attributes:

- `mailServer`: Fully qualified domain name of the email relay server used to send alert notifications. This attribute only requires specification in cases where DNS returns an unreachable or invalid mail exchange (MX) record for the email server specified in `smtpToAddr`.

- `smtpPort`: Email server port used to send alert notifications
- `smtpUseSSL`: Specification to use Secure Socket Layer (SSL) encryption for alert notifications.
- `smtpFrom`: User name that appears in the `From:` header of the alert notifications
- `smtpFromAddr`: Email address that appears in the `From:` header of the alert notifications. This email address is not authenticated with the email server.
- `smtpToAddr`: Address to which email is sent. It can be a comma-delimited list in quotation marks to allow multiple subscribers to alerts.
- `snmpSubscriber`: List of hosts that subscribe to the SNMP alert notifications
- `snmpUser`: Defines users who receives SNMP alerts
- `snmpEngineID`: An identifier used by the SNMP managers to subscribe to alerts from the storage cells
- `notificationMethod`: Notification method for alerts
- `notificationPolicy`: Indicator for severity alerts to be sent to subscribers
- `emailFormat`: File format for email messages
- `emailSubscriber`: List of names that subscribe to the alert notifications

Usage Notes

- **`mailServer`**

  The `mailServer` attribute identifies the email relay server used to send alert notifications. When you modify the `mailServer` attribute value, the Exadata Management Server (MS) automatically configures and restarts the sendmail service. You can clear the `mailServer` attribute and remove the email relay server from the sendmail configuration by setting `mailServer` to an empty string enclosed by quotation marks (`mailServer=''`).

- **`smtpPort`**

  The `smtpPort` attribute can be reset to the default value by setting it to an empty string enclosed by quotation marks (`smtpPort=''`).

- **`smtpUseSSL`**

  The `smtpUseSSL` attribute enables Secure Socket Layer (SSL) encryption on the email notifications when the attribute is set to `true`.

- **`smtpToAddr`**

  The `smtpToAddr` attribute can be used to set a list of comma-delimited email addresses that are the recipients of the alert notification. The list must be enclosed in quotation marks.

- **`snmpSubscriber`**

  The `snmpSubscriber` attribute can be set to a list of SNMP targets to which the SNMP alert notification is sent. The targets are specified as follows:

  ```
  snmpSubscriber[-|+]=(
    (host=host[,port=port][,type=subscriber_type][,community=community]
  [,snmpuser=snmp_user_name][,fromIP="ip"][,asrmPort="ASRManager_port"])
  ```

```
[,(host=host[,port=port][,type=subscriber_type][,community=community]
[,snmpuser=snmp_user_name][,fromIP="ip"]
[,asrmPort="ASRManager_port"])] ...)
```

The `snmpSubscriber` attribute uses the following values:

- The `host` must be specified as either a host name or an IP address. Enclose the host name or IP address in quotation marks if it contains non-alphanumeric characters.
- The default value for `port` is `162`. This value is optional.
- The valid `type` values are `v1`, `ASR`, `v3`, and `v3ASR`.
  - Starting with Oracle Exadata System Software release 24.1.0, you must specify the `type` value.

    Previously, setting the `type` is optional, and the default value is `v1`.
  - The `type=v3` and `type=v3ASR` options use SNMP V3. SNMP V3 is considered more secure than earlier SNMP versions, and should be used where possible.
  - The `snmpSubscriber` with `type=ASR` or `type=v3ASR` should only be configured to point to Oracle ASR Manager.
  - The `type=ASR` and `type=v3ASR` options set the Oracle ASR destination for Oracle Exadata Storage Server, and its ILOM. Removing all `snmpSubscriber` entries with `type=ASR` and `type=v3ASR` from the SNMP subscriber list disables the Oracle ASR trap mechanism for Oracle Exadata Storage Server and its ILOM.
  - For the `v3ASR` type, the user must be defined with `authProtocol=SHA`, and `privProtocol=AES`. These are the only protocols supported by Oracle ASR Manager. Setting the `snmpSubscriber` as type `v3ASR` also sets the ILOM properties and rules for traps sent by ILOM.
- Starting with Oracle Exadata System Software release 24.1.0, you must specify the `community` value for subscribers with `type=v1` or `type=ASR`. Also, common default values such as `public` and `private` are discouraged for security reasons.

  Previously, setting the `community` is optional, and the default value is `public`.
- For subscribers with `type=v3` or `type=v3ASR`, you must specify an SNMP user name (`snmpuser=`*`snmp_user_name`*), which is already configured within the server.

  For example:

  ```
  CellCLI> ALTER CELL snmpuser.snmpuser1=(authprotocol=SHA,authpassword=*)
  ...

  CellCLI> ALTER CELL
  snmpSubscriber=((host=newhost,port=162,type=v3,snmpuser=snmpuser1))
  ```
- The `fromIP` field enables you to specify an IP address from which the trap is sent. If this field is not specified, it defaults to the IP address associated with eth0. Use this field if the default IP address is not registered with Oracle ASR Manager. Oracle ASR Manager only processes SNMP traps that are sent from IP addresses that it recognizes.

  The `fromIP` field is allowed only for SNMP subscribers whose type is either `ASR` or `v3ASR`.

For example:

```
CellCLI> ALTER CELL
snmpSubscriber=((host=asrhost,port=162,community=asrcommunity,fromIP="1.
1.1.1",type=ASR))
```

The following example returns an error because the type is not `ASR` or `v3ASR`.

```
CellCLI> ALTER CELL
snmpSubscriber=((host=localhost,port=162,community=asrcommunity,fromIP="
1.1.1.1",type=v1))
CELL-00068: The fromIP field is only supported for ASR SNMP subscribers.
```

  - The `asrmPort` field enables you to specify the port number on an Oracle ASR Manager machine that MS uses to communicate with Oracle ASR Manager. This port must be the same as the HTTP port of Oracle ASR Manager's HTTP Receiver. You can check this by running `asr show_http_receiver` on the Oracle ASR Manager machine.

    The `asrmPort` field is allowed only for SNMP subscribers whose type is either `ASR` or `v3ASR`. The default value for this port is 16161.

By default, `ALTER CELL snmpSubscriber=(`*`SNMPtargets`*`)` replaces the existing `snmpSubscriber` value. However, starting with Oracle Exadata System Software release 21.2.0, you can add to the existing list of SNMP targets by using `snmpSubscriber+=(`*`SNMPtarget`*`)`. For example:

```
CellCLI> ALTER CELL
snmpSubscriber+=((host=newhost,port=162,community=snmpcommunity,type=v1))
```

Also, starting with Oracle Exadata System Software release 22.1, you can remove an entry from the existing list of SNMP targets by using `snmpSubscriber-=(`*`SNMPtarget`*`)`. For example:

```
CellCLI> ALTER CELL snmpSubscriber-
=((host=myhost,port=162,community=snmpcommunity,type=v1))
```

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 set for the new ILOM. If the entries are removed from the ILOM, then they must be manually added to the ILOM using the `ALTER CELL ... snmpUser=` command.

- **`snmpUser`**

  The `snmpUser` attribute defines users who receives SNMP alerts. This command can only be run in interactive mode. There are two methods for configuring this attribute.

  ```
  snmpuser=((user_clause1)[,(user_clauseN)]...)

  snmpuser.name=(user_clause)
  ```

  - If you specify `snmpuser`, then you must provide a *user_clause* for every configured user. If you omit a user, then that user will no longer receive SNMP alerts. The `((user_clause1)[,(user_clauseN)]...)` string that you provide overwrites the previous string used for the `snmpuser` attribute.

- If you specify `snmpuser.name`, then you must provide a *user_clause* for only the specified user. This allows you to add, delete, or modify each user individually, without having to supply the entire `snmpuser` attribute string each time.
- If you use `snmpuser=''`, then all SNMP users are removed. If you use `snmpuser.name=''`, then only the specified user is removed. You cannot remove an SNMP user while it is still referenced by a V3 SnmpSubscriber.

Each method uses a *user_clause*, which has the following general format:

```
([name=user1,] authProtocol=auth_type, authPassword=*
   [, privProtocol=priv_type, privPassword=*])
[,(name=userN, authProtocol=auth_type, authPassword=*
   [, privProtocol=priv_type, privPassword=*] )]...
```

If updating a single user using the `snmpuser.name` notation, do not include the phrase `name=user1` in the *user_clause*.

- `name` is the user name.
- Only `*` is allowed for the password values in the command. Passwords are not stored or displayed. Secure hash keys are computed and used for trap authentication and encryption.
- `authProtocol` specifies the authentication protocol.

  Options include `MD5` and `SHA`. Additionally, Oracle Exadata System Software release 24.1.0 introduces the following SHA2 authentication protocols for SNMP V3 subscribers: `SHA-224`, `SHA-256`, `SHA-384`, and `SHA-512`.

  The `authProtocol` must be specified for the `snmpUser` attribute.

  The system prompts for the authentication password. The authentication password must have 8 to 12 alphanumeric characters.
- `privProtocol` is encryption protocol. Options are `none`, `AES`, or `DES`. The default is `none` when the `privProtocol` attribute is not specified.

  The system prompts for an encryption password if the encryption protocol is specified. The password is exactly 8 alphanumeric characters, and they are case sensitive.

- **`snmpEngineID`**

  The `ALTER CELL snmpEngineID` command is used by the SNMP managers to subscribe to alerts from the storage cells. The `snmpEngineID` parameter can be up to 20 characters. It should be unique for each target within a data center. The default is the cell name. This default is used if the `snmpEngineID` attribute is not set before the SNMP users are defined.

  > **Note:**
  >
  > The engine identifier should not be changed after SNMP users are defined. Any change to an engine identifier causes the user keys to be re-computed, and user passwords must be re-entered.

- **`notificationMethod`**

The `notificationMethod` attribute value can be `mail`, `snmp`, `none`, or a combination of `mail` and `snmp`, such as `notificationMethod='mail,snmp'`. The default value is `mail`.

- **`notificationPolicy`**

  The `notificationPolicy` attribute value can be `none` or a combination of `critical`, `warning`, or `clear`, such as `notificationPolicy='warning,clear.'`

  – 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 `maintenance` value refers to all hardware-related errors. The hardware errors are reported as "Maintenance" in email message subject lines.

- **`emailFormat`**

  The `emailFormat` attribute can be `html` or `text`. By default, email notifications are sent in HTML format. Change the value to `text` to receive plain text email notifications.

- **`emailSubscriber`**

  The `ALTER CELL emailSubscriber` command sets a list of comma-delimited email addresses that are the recipients of alert notifications for specific alert types. The syntax for this command is:

```
ALTER CELL emailSubscriber = ((email="email_address1",                \
           alertType="alert_type")                                 \
          [, (email="email_address2",alertType="alert_type"), ...])
```

  – The email address must be a valid email address. The `email` parameter is mandatory.

  – The `alertType` parameter specifies the type of alert, and is optional. The alert types are `HARDWARE`, `SOFTWARE`, `METRIC` or `ADR`. If the alert type is not specified, then the subscription is for all alert types.

  – An empty input string removes the current set of subscribers.

  – The notification policy must be set before alert notifications can be received. The policy applies to all email subscribers. The notification policy for these alerts are the same as for `snmpSubscriber` alerts.

  To validate that email messages are successfully sent for cell alerts or events, use the `ALTER` command with the `VALIDATE MAIL` option. The validation process sends a test email message to the configured recipient. If that test email message is not received, then an email configuration setting is not valid.

Examples

Example 7-11 shows how to set the `asrmPort` field for an `snmpSubscriber`.

Example 7-12 shows how to set up email notifications for the cell.

Example 7-13 shows how to modify the SNMP user.

Example 7-14 shows how to modify a single SNMP user.

Example 7-15 shows how to specify the type of email alerts. In the example, one subscriber gets hardware and software alerts, and the other subscriber gets ADR alerts.

Example 7-16 shows how to change the format of email messages.

Example 7-17 shows how to unsubscribe from email alerts.

Example 7-18 shows how to reset the `notificationPolicy` attribute to its default value.

Example 7-11 Setting the asrmPort for an snmpSubscriber

```
CellCLI> ALTER CELL
snmpSubscriber=((host=host1,port=162,community=asrcommunity,type=asr,asrmPort=
16161))
```

Example 7-12 Configuring Email Notifications for a Cell

The example includes multiple SNMP subscribers. Note that because `host2` is an SNMP v3 subscriber, there is no `community` specification. Instead, for SNMP v3 subscribers, you must specify an existing SNMP user. See also the following for examples for modifying an SNMP user.

```
CellCLI> ALTER CELL
mailServer='my_mail_relay.example.com',                                  -

smtpFromAddr='john.doe@example.com',                                     -
                    smtpFrom='John
Doe',                                                         -

smtpToAddr='jane.smith@example.com',                                     -

snmpSubscriber=((host=host1,port=162,community=snmpcommunity,type=v1),   -

(host=host2,port=162,snmpuser=user2,type=v3)),                          -

notificationPolicy='clear',                                              -
                    notificationMethod='mail,snmp'
```

Example 7-13 Modifying the SNMP User

This example shows SNMP user configuration, where the administrator is prompted to enter the passwords. The example contains one user definition, but the same approach can be expanded to define multiple SNMP users in the same command.

```
CellCLI> ALTER CELL snmpuser = ((name=ASR, authprotocol=md5,
authpassword=*,    \
                    privprotocol=AES, privpassword=*))
snmpUser ASR authpassword: password
Confirm snmpUser ASR authpassword: password
snmpUser ASR privpassword: password
Confirm snmpUser ASR privpassword: password
```

Example 7-14   Modifying an SNMP User

The following code examples show adding an SNMP user, changing that user's password, and then removing that user.

```
## adding users individually
CellCLI> ALTER CELL snmpuser.user2=(authprotocol=SHA,authpassword=*)

snmpUser user2 authpassword: password
Confirm snmpUser user2 authpassword: password

snmpUser ((name=user1, authProtocol=SHA, privProtocol=AES)) has been replaced
with
((name=user1, authProtocol=SHA, privProtocol=AES),(name=user2,
authProtocol=SHA)).
...

## changing a password of an existing user
CellCLI> ALTER CELL snmpuser.user2 = (authprotocol=SHA,authpassword=password)

## delete a user individually
CellCLI> ALTER CELL snmpuser.user2=''

snmpUser ((name=user1, authProtocol=SHA, privProtocol=AES),(name=user2,
authProtocol=SHA)) has
 been replaced with ((name=user1, authProtocol=SHA, privProtocol=AES)).
...
```

Example 7-15   Specifying the Type of Email Alert

```
ALTER CELL emailSubscriber=                                            \
           ((email="email1@example.com",alertType="HARDWARE,SOFTWARE"), \
           (email="email2@example.com",alertType="ADR"))
```

Example 7-16   Changing the Format of Email Messages

```
CellCLI> ALTER CELL emailFormat='text'
CellCLI> ALTER CELL emailFormat='html'
```

Example 7-17   Unsubscribing from Email Alerts

```
ALTER CELL emailSubscriber=""
```

Example 7-18   Setting the Default Value for the notificationPolicy Attribute

This example shows how to set the default value for the `notificationPolicy` attribute.

```
CellCLI> alter cell notificationPolicy=""
```

7.7.1.2.4.3 Alert Summary Attributes

Configuring Alert Summaries

- The `alertSummaryInterval` attribute sets the frequency of the open alerts summary e-mail message. The open alerts e-mail message is an HTML document that provides a concise summary of all open issues on a cell even without access to the cell. Valid options are `daily`, `weekly`, `biweekly` and `none`.The default value is `weekly`.
- The `alertSummaryStartTime` attribute sets the delivery time for the open alerts summary e-mail message. The command accepts any valid time stamp.

Example 7-19 Setting the Frequency for the Open Alerts Summary E-mail Message

This example shows how to set the frequency for the open alerts summary e-mail message to weekly.

```
CellCLI> ALTER CELL alertSummaryInterval=weekly
```

Example 7-20 Setting the Time for Open Alerts Message Delivery

This example shows how to set the delivery time for the open alerts summary e-mail message.

```
CellCLI> ALTER CELL alertSummaryStartTime="2013-04-23T12:57:00-06:00"
```

7.7.1.2.4.4 SYSLOG Attributes

Configuring SYSLOG Attributes: `syslogconf` and `syslogFormat`

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 syntax for configuring this attribute is:

```
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 `facility` option for the `syslogconf` attribute must be one of the following: `auth`, `authpriv`, `cron`, `daemon`, `ftp`, `kern`, `lpr`, `mail`, `mark`, `news`, `security`, `syslog`, `user`, `uucp`, `local0`, `local1`, `local2`, `local3`, `local4`, `local5`, `local6`, `local7`, `none`, and `*`.
- The `priority` option for the `syslogconf` attribute must be one of the following: `alert`, `crit`, `debug`, `emerg`, `err`, `error`, `info`, `notice`, `panic`, `warn`, `warning`, `none`, and `*` (asterisk).

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.

Starting with Oracle Exadata System Software release 19.1.0, you can use the `syslogFormat` attribute to change the standard format for syslog to any format by setting the value to the desired format string. Setting the `syslogFormat` attribute to an empty string removes the format change, reverting the syslog format to the default format. If the format string contains a control character, it must be preceded by a backslash when entering the command.

See Example 7-24 for examples of the syntax.

Starting with Oracle Exadata System Software release 19.3.0, you can use the `syslogFormat` attribute to enable sending syslog in an encrypted format. For the complete configuration steps, refer to Encrypting System Log Information.

Example 7-21 Using the syslogconf Attribute

This example shows how to add a rule using the syslogconf attribute.

```
CellCLI> ALTER CELL syslogconf=('*.err;authpriv.none @loghost', -
         '*.emerg @loghost')
```

Example 7-22 Adding and Validating a Rule

This example shows how to add and validate a rule with test message.

```
CellCLI> ALTER CELL syslogconf=('kern.crit @loghost')
CellCLI> ALTER CELL VALIDATE syslogconf   'kern.crit'
```

Example 7-23 Removing All `syslog.conf` Rules

This example shows how to remove the `syslog.conf` rule.

```
CellCLI> ALTER CELL syslogconf=''
```

Example 7-24 Setting the Syslog Format to a Custom String Then Reverting to the Default Format

This example shows how to specify a customized format for syslog.

```
CellCLI> ALTER CELL syslogformat="%TIMESTAMP:::date-rfc3339% %HOSTNAME%
%syslogtag%
%syslogseverity-text%:%msg:::sp-if-no-1st-sp%%msg:::drop-last-lf%\\n"

CellCLI> ALTER CELL syslogformat="%TIMESTAMP% %HOSTNAME% %msg%\\n"

CellCLI> ALTER CELL syslogformat=""
```

Configuring additional log forwarding: `syslogInput`

Starting with Oracle Exadata System Software release 22.1, the `syslogInput` attribute enables syslog on the local host (database server or storage server) to forward additional logs to remote log servers.

The syntax for configuring the `syslogInput` attribute is:

```
syslogInput = ('selector @[@]node[:remote_port]' [, 'selector
@[@]node[:remote_port]']... )
```

In the preceding syntax, *selector* specifies the additional logs being forwarded. The *selector* value can contain the following entries:

- `audit` - Specifies the audit log at `/var/log/audit/audit.log`.
- `aide` - Specifies the Advanced Intrusion Detection Environment (AIDE) log at `/var/log/aide/aide.log`.
- `yum` - Specifies the YUM log at `/var/log/yum.log`.

Multiple *selector* entries must be separated by a semicolon (`;`) character.

Each *node* is specified using the hostname or IP address preceded by one or two ampersand (`@`) characters. You can specify one ampersand (`@`) character to use UDP for communications or specify two ampersand (`@@`) characters to use TCP.

By default, the remote system receives communications on port 514, which is the default rsyslogd port. You can specify another port number by appending a colon (`:`) character and remote port number to the *node* specification

In the following example, `loghost1` is configured to receive audit and AIDE logs using UDP on the default rsyslogd port (514). Also, `loghost2` is configured to receive YUM logs using TCP on port 10514.

```
CellCLI> ALTER CELL syslogInput=('audit;aide @loghost1','yum
@@loghost2:10514')
```

To stop and remove additional log forwarding, set `syslogInput` to an empty string. For example:

```
CellCLI> ALTER CELL syslogInput=''
```

Configuring the ILOM SYSLOG: `ilomSyslogClients`

Starting with Oracle Exadata System Software release 21.2.0, the `ilomSyslogClients` attribute specifies the remote destination to forward syslog messages from the Integrated Lights Out Manager (ILOM) service processor (SP).

The `ilomSyslogClients` attribute accepts a comma-separated list of up to two loghost servers. For each loghost server, you must specify a valid hostname or IP address.

For example:

```
CellCLI> ALTER CELL ilomSyslogClients="192.0.2.101,192.0.2.201"
```

Note:

The specified `ilomSyslogClients` must listen on port 514 to receive the ILOM syslog messages.

7.7.1.2.4.5 Disk Scrubbing Attributes

Configuring Disk Scrubbing Attributes

Disk scrubbing proactively inspects and repairs hard disks. If a bad sector is detected on a hard disk, Oracle Exadata System Software automatically orchestrates the repair using data from another mirror copy.

Disk scrubbing is a long-running operation. The time required to complete a scrubbing operation depends mainly on disk size and workload. As a guide, allow between 1 and 2 hours for each terabyte of disk capacity to scrub an otherwise idle disk.

On an idle system, disk scrubbing can drive disk utilization to 100%. However, Exadata I/O Resource Management (IORM) throttles disk scrubbing so that user workloads are prioritized and should not be affected by disk scrubbing.

The `ALTER CELL hardDiskScrubInterval` command activates (or deactivates) automatic scrubbing and sets the scrubbing interval. Valid options are `daily`, `weekly`, `biweekly`, and `none`. Using the `none` option stops all active disk scrubbing operations and deactivates automatic scrubbing.

The `ALTER CELL hardDiskScrubStartTime` command sets the start time for the scrubbing schedule. Valid options are a specific date and time or `now`.

In the alert log, you may see messages such as `Begin scrubbing celldisk` and `Finished scrubbing celldisk`. These are expected informational messages, and no action is necessary.

Example 7-25   Setting the Disk Scrubbing Interval to Weekly

This example shows how to activate automatic weekly disk scrubbing.

```
CellCLI> ALTER CELL hardDiskScrubInterval=weekly
```

Example 7-26   Setting the Start Time for Proactive Disk Scrubbing

This example shows how to set a specific start time for disk scrubbing.

```
CellCLI> ALTER CELL hardDiskScrubStartTime='2013-08-07T21:19:22-07:00'
```

7.7.1.2.4.6 Security Certificate Attributes

Configuring CA-Certified Security Certificate Attributes

To set up CA-certified security certificates on the cell for use with ExaCLI, use the following attributes:

> **Note:**
>
> The following attributes can be used only if you are running the `ALTER CELL` command from ExaCLI.

- `securityPubKey` - Specifies the URL to the public key file.
- `securityPrivKey` - Specifies the URL to the private key file.
- `securityPrivKeyPW` - Specifies the password to use if the private key file is encrypted.

After you upload the CA-certified security certificate, you must restart MS before the new security certificate is visible.

```
CellCLI> ALTER CELL RESTART SERVICES MS
```

See Also:

Using a CA-Certified Security Certificate in *Oracle Exadata Database Machine Maintenance Guide*

Example 7-27   Configuring the Security Keys for a Storage Server

This example shows how to configure the security keys for a storage server, including supplying the password after entering the command.

```
exacli -e 'ALTER CELL securityPubKey="http://www.example.com/security/
newkey.crt",  -
                   securityPrivKey="http://www.example.com/security/
newkey.key", -
                   securityPrivKeyPW=*'

password=***************
```

7.7.1.2.4.7 Persistence Attributes

Persistent Columnar Cache

Starting with Oracle Exadata System Software release 21.2.0, the `columnarCachePersMode` attribute controls the persistent columnar cache feature. The valid attribute setting are:

- `on` - Enables the persistent columnar cache feature.
- `off` - Disables the persistent columnar cache feature.
- `auto` - Oracle Exadata System Software decides whether to enable or disable the persistent columnar cache feature. If the `columnarCachePersMode` attribute is not set, then the `auto` setting is implied.

  Starting with Oracle Exadata System Software release 21.2.11, the `auto` setting enables the persistent columnar cache feature (equivalent to `columnarCachePersMode=on`). Previously, the `auto` setting disabled the persistent columnar cache feature (equivalent to `columnarCachePersMode=off`).

After you alter the `columnarCachePersMode` attribute, you must restart the cell server to implement the change.

Persistent Storage Index

Starting with Oracle Exadata System Software release 21.2.0, the `storageIndexPersMode` attribute controls the persistent storage index feature. The valid attribute setting are:

- `on` - Enables the persistent storage index feature.
- `off` - Disables the persistent storage index feature.
- `auto` - Oracle Exadata System Software decides whether to enable or disable the persistent storage index feature. If the `storageIndexPersMode` attribute is not set, then the `auto` setting is implied.

  Starting with Oracle Exadata System Software release 21.2.11, the `auto` setting enables the persistent storage index feature (equivalent to `storageIndexPersMode=on`). Previously, the `auto` setting disabled the persistent storage index feature (equivalent to `storageIndexPersMode=off`).

After you alter the `storageIndexPersMode` attribute, you must restart the cell server to implement the change.

7.7.1.2.4.8 Real-Time Insight Attributes

Commencing with Oracle Exadata System Software 22.1.0, you can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems.

Fine-Grained Metric Collection

The `metricFGCollIntvlInSec` attribute controls fine-grained metric collection.

- To enable fine-grained metric collection, you must set the collection interval to a value between 1 and 60 seconds.

  For example:

  ```
  CellCLI> ALTER CELL metricFGCollIntvlInSec=1
  ```

  The `metricFGCollIntvlInSec` setting is related to the automatic upload frequency specified in the `metricStreamIntvlInSec` attribute. When automatic metric upload and fine-grained collection are both enabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec>0`), `metricStreamIntvlInSec` must be between 5 and 30 times `metricFGCollIntvlInSec`. For example, if `metricStreamIntvlInSec=60`, then `metricFGCollIntvlInSec` must be between 2 and 12.
- To disable fine-grained metric collection on a server, set `metricFGCollIntvlInSec=0`.

  For example:

  ```
  CellCLI> ALTER CELL metricFGCollIntvlInSec=0
  ```

  Fine-grained metric collection can be disabled only when automatic metric upload is disabled (`metricStreamIntvlInSec=0`) or the automatic upload frequency is between 5 and 30 minutes (`metricStreamIntvlInSec` is between 300 and 1800).

Automatic Metric Upload

The `metricStreamIntvlInSec` attribute sets the upload interval (in seconds) for automatic uploads to the metric streaming endpoints specified by the `metricStreamEndPoint` attribute.

- To enable automatic metric uploads, set the `metricStreamIntvlInSec` attribute to a non-zero value.

  For example:

  ```
  CellCLI> ALTER CELL metricStreamIntvlInSec=25
  ```

  The `metricStreamIntvlInSec` setting is related to the fine-grained collection frequency specified in the `metricFGCollIntvlInSec` attribute:

  – When automatic metric upload and fine-grained collection are both enabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec>0`), `metricStreamIntvlInSec` must be between 5 and 30 times `metricFGCollIntvlInSec`.

For example, if `metricFGCollIntvlInSec` is set to 5, then `metricStreamIntvlInSec` must be between 25 and 150.

  – When automatic metric upload is enabled and fine-grained collection is disabled (`metricStreamIntvlInSec>0` and `metricFGCollIntvlInSec=0`), the automatic upload frequency must be between 5 and 30 minutes (`metricStreamIntvlInSec` must be between 300 and 1800).

- To disable automatic metric uploads, set `metricStreamIntvlInSec=0`.

  For example:

  ```
  CellCLI> ALTER CELL metricStreamIntvlInSec=0
  ```

Metric Upload Endpoints

The `metricStreamEndPoint` attribute specifies one or more collection endpoints that automatically receive the metric stream. You can set `metricStreamEndPoint` as follows:

```
metricStreamEndPoint[+]=((host="endpoint-URL"[,type="stream-format"]
[,token="authentication-token"][,{httpProxy|httpsProxy}="proxy-server"])
                             [,(host="endpoint-URL"[,type="stream-format"]
[,token="authentication-token"][,{httpProxy|httpsProxy}="proxy-server"])]...)
```

In the `metricStreamEndPoint` definition:

- `host`: Specifies the URL for the collection endpoint. The URL can use HTTP or HTTPS.
- `type`: Optionally specifies the format of the stream. Supported values are:
  – `json`: Provides the stream in a JSON format
  – `plaintext`: Provides the stream in a plain text format

  The default value is `json`.
- `token`: Optionally specifies the authentication token for the collection endpoint. Consult the metric collection platform for details about generating the token.
- `httpProxy` or `httpsProxy`: Optionally specifies a proxy server to facilitate network connectivity to the collection endpoint. A proxy server is required if a firewall resides between the Exadata system and the collection endpoint.

You can use the optional `+=` operator to add collection endpoints to an existing `metricStreamEndPoint` definition. Otherwise, the `=` operator overwrites the previous attribute value.

Metric Tags

The `metricStreamTags` attribute defines a set of metric tags, which are included in every metric observation generated by the server. These tags can help you to organize and group observations generated by numerous Exadata servers.

You can set the `metricStreamTags` attribute to a valid JSON string containing tag and value pairs as follows:

```
metricStreamTags='{"tag1":"value1"[,"tag2":"value2"]...}'
```

For example:

```
CellCLI> ALTER CELL
metricStreamTags='{"application":"personnel","department":"HR"}'
```

7.7.1.2.4.9 Miscellaneous Attributes

> **Note:**
>
> For a complete list of cell attributes, see DESCRIBE CELL.

dbPerfDataSuppress

Use the `dbPerfDataSuppress` attribute to hide performance output information for specific databases. Specify the databases to mask as a comma-delimited list of names. The performance information for the specified databases is still collected, but is only visible when queried from that database. If you query `V$CELL_DB` from a different database, then the performance information for the hidden databases appears in the category of `OTHER`.

diagPackEmailAttach

Use the `diagPackEmailAttach` attribute to turn on and off adding the diagnostic pack attachment to emails, for example:

```
alter cell diagPackEmailAttach=FALSE
```

See CREATE DIAGPACK for information about diagnostic packs.

diagPackUploadEnabled

Use the `diagPackUploadEnabled` attribute to enable or disable automatically uploading diagnostic data to a service request using Oracle ASR.

Example 7-28   Enabling/Disabling Auto Diagpack Upload

You can enable or disable this feature by setting the `diagPackUploadEnabled` attribute on the `cell` object.

Set the attribute to `false` to disable this feature, `true` to enable it. The default is `true`.

```
CellCLI> ALTER CELL diagPackUploadEnabled=FALSE
```

enableSmartStorage

The `enableSmartStorage` attribute can be set to `TRUE` to enable the use of Oracle Exadata System Software capabilities such as Smart Scan and Storage Index on Exadata Extended (XT) Storage Server after you have procured the necessary software licenses.

httpsAccess

Starting with Oracle Exadata System Software release 19.1.0, the `httpsAccess` attribute can be used to specify a list of IP addresses or IP subnet masks that control who can access the

RESTful service via HTTPs. The value you specify for `httpsAccess` overwrites any previous value. You can use the following values for `httpsAccess`:

- `ALL` — to allow access to all hosts (Default)
- `NONE` — to disable the HTTPs port completely
- `IP1, IP2,..., IPn` — to only allow access to hosts with IP addresses IP1, IP2,..., IP*n* where IP*n* is a valid IP address in IPv4, IPv4 subnet, IPv6 or IPv4-embedded IPv6 format. You can specify a maximum of 512 IP addresses for the access control list.

Additionally, instead of a single IP address, you can use the `/` character to specify a range of IP addresses using a subnet mask. For example the range `'192.168.10.0/24'` corresponds to hosts having IP addresses from 192.168.10.1 to 192.168.10.255. If you specify an IP address range, you need to enclose the IP address string in quotes.

Example 7-29   Restricting HTTPS Access to the Exadata RESTful Service

This example shows how to configure an access control list for HTTPs access to the Exadata RESTful service. The following command allows HTTPs port access to hosts having IP addresses in the range from 192.168.10.1 to 192.168.10.255.

```
CellCLI> ALTER CELL httpsAccess="192.168.10.0/24"
```

interconnectN

The `ALTER CELL interconnectN=""` command removes the RDMA Network Fabric configuration information for the cell for the specified interface (*N*).

If the IP address to an RDMA Network Fabric 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.

Example 7-30   Setting RDMA Network Fabric Interconnections

This example shows how to set the RDMA Network Fabric interconnections.

For systems that use InfiniBand Network Fabric, use a command such as the following:

```
CellCLI> ALTER CELL interconnect1='ib0', interconnect2='ib1'
```

For systems that use RoCE Network Fabric, use a command such as the following:

```
CellCLI> ALTER CELL interconnect1='re0', interconnect2='re1'
```

After making the updates, restart all services in the storage server:

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

iotimeoutthreshold

Use the `iotimeoutthreshold` attribute to change the timeout threshold. If cell I/O takes longer than the defined threshold, then the I/O is canceled, and Oracle ASM redirects the I/O to another mirror copy of the data. Any I/Os issued to the last valid mirror copy of the data are not canceled, even if the timeout threshold is exceeded.

The default value for `iotimeoutthreshold` is 1000s. The command takes a value, such as `5`, and a unit. The valid unit is `s`, for seconds.

Caution:

Setting the timeout threshold too low can negatively impact system performance. Oracle recommends reviewing the Automatic Workload Repository (AWR) reports of peak I/O loads, and setting the threshold value to a value higher than the peak I/O latency with sufficient safety margin.

Example 7-31 Setting the iotimeoutthreshold Value

This example demonstrates how to set the `iotimeoutthreshold` to 5 seconds.

```
CellCLI> ALTER CELL iotimeoutthreshold = '5s'
```

To reset the `iotimeoutthreshold` to the default value, use the following command:

```
CellCLI> ALTER CELL iotimeoutthreshold = ""
```

listeningInterface

Starting with Oracle Exadata System Software release 24.1.0, the `listeningInterface` attribute specifies the network interfaces that listen for commands using the Exadata RESTful service. The value you specify for `listeningInterface` overwrites any previous value. You can use the following values for `listeningInterface`:

- `ALL` — to allow access on all network interfaces (Default)
- `NONE` — to disable access on all network interfaces
- `IP1, IP2, ..., IPn` — to allow access only through the network interfaces associated with the specified IP addresses

The `listeningInterface` attribute complements the `httpsAccess` attribute. The `listeningInterface` attribute specifies which server network interfaces accept REST requests, while the `httpsAccess` attribute restricts the source of requests to the Exadata RESTful service.

Example 7-32 Restricting Access to the Exadata RESTful Service

This example shows how to identify a specific network interface on the server that listens for commands using the Exadata RESTful service.

```
CellCLI> ALTER CELL listeningInterface="192.168.10.11"
```

name

The `name` attribute contains the host name of the storage server, for example, `dm01celladm01`.

Example 7-33 Altering Cell Name

This example shows the `ALTER` command with the `CELL` object.

```
CellCLI> ALTER CELL name=cell02
```

traceLevel

The level for which trace messages are written. The default is `FINE`. The value can be:

- A valid Java logging level
  - `SEVERE`
  - `WARNING`
  - `INFO`
  - `CONFIG`
  - `FINE`
  - `FINER`
  - `FINEST`
- 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`

To reset this attribute to its default value, use a value of `""`.

Examples

Example 7-34 Setting the traceLevel Value to its Default Value

This example shows how to set the `traceLevel` value to its default value.

```
CellCLI> ALTER CELL traceLevel=""
```

7.7.1.3 ALTER CELLDISK

Purpose

The `ALTER CELLDISK` command changes the attributes of all cell disks or the specified cell disks.

Syntax

```
ALTER CELLDISK { ALL [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] FLASHDISK
| HARDDISK | PMEM ] | cdisk_name [, cdisk_name]... }
   {{FLUSH [NOWAIT] | CANCEL FLUSH} |
   { 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 7-97.

- The `FLASHDISK` option limits the `ALTER CELLDISK` command to cell disks that are flash disks.

  Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to alter cell disks matching only the specified flash media type.
- The `HARDDISK` option limits the `ALTER CELLDISK` command to cell disks that are hard disks.
- The `PMEM` option limits the `ALTER CELLDISK` command to all cell disks of type PMEM.
- The `FLUSH` option synchronizes dirty data associated with the specified cell disks. Dirty data is updated data in an Exadata cache (flash cache, PMEM cache, or XRMEM cache) that has not been synchronized with the underlying grid disk. Synchronization of dirty data can be a lengthy process, depending on the number of bytes to be synchronized. Use the following command to check the progress:

  ```
  CellCLI> LIST CELLDISK ATTRIBUTES name, flushstatus, flusherror
  ```
- The `ALTER CELLDISK ... FLUSH` command must be run before exporting a cell disk to ensure that the dirty data is flushed to the grid disks on the specified cell disk.
- The `FLUSH` option stops new data from being cached on the flash cache until CELLSRV restarts, or the flush operation is canceled.
- The `CANCEL FLUSH` option terminates an earlier flush operation, and reinstates caching.
- When the `ALTER CELLDISK ... FLUSH` command is run for a flash-based cell disk, it synchronizes dirty data from the flash cache located on the specified FDOM to cached grid disks on the specified cell disk. When the command is run for a hard disk-based cell disk, it synchronizes dirty data from the flash cache located on all FDOMs to the grid disks located on the specified cell disk.

Example 7-35 Altering Cell Disk Attributes

This example shows how to change 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'

CellCLI> ALTER CELLDISK ALL HARDDISK FLUSH NOWAIT
```

```
CellCLI> ALTER CELLDISK c9datafile1 CANCEL FLUSH
```

Related Topics

- Restrictions on Values of Common Attributes
  Review the following restrictions for the values of attributes used by multiple CellCLI objects.
- CREATE CELLDISK

7.7.1.4 ALTER FLASHCACHE

Purpose

The `ALTER FLASHCACHE` command stops new data from being cached on the flash cache and then flushes data not synchronized with the grid disks (dirty data) from flash cache to the specified disks.

Syntax

```
ALTER FLASHCACHE { ALL [size=fc_size] | CELLDISK="cdisk1 [,cdisk2] ..." [,
size=fc_size] [FORCE] }
       { FLUSH [NOWAIT] | CANCEL FLUSH }
```

Usage Notes

Note:

The `FLUSH` option stops new data from being cached on the flash cache until CELLSRV restarts, or the flush operation is canceled with the `ALTER FLASHCACHE CANCEL FLUSH` command.

- The `ALL` option affects all available flash cell disks.
- The `CELLDISK` option allows specific cell disks to be flushed.
- The `FORCE` option allows you to forcefully change the set of cell disks used by the flash cache.
- If specified, the `size` attribute re-sizes the flash cache. The value is validated.

  If a valid size is specified in conjunction with the `ALL` option, then the flash cache is dropped and re-created on all of the cell disks using the specified size.

  If a valid size is specified in conjunction with the `CELLDISK` option, then the flash cache is dropped and re-created on the specified cell disks using the specified size.
- The `FLUSH` option synchronizes dirty data from the flash cache to the grid disks. Dirty data is data that has not been synchronized with the grid disk. Synchronization of dirty data can be a lengthy process, depending on the number of bytes to be synchronized. Use the following command to check the progress:

  ```
  LIST CELLDISK ATTRIBUTES name, flushstatus, flusherror
  ```

- The `ALTER FLASHCACHE CELLDISK= ... FLUSH` command does not flush the dirty data when the data cannot be read from the flash cache or written to disk. To flush the dirty data from the flash disk to grid disks use the `ALTER GRIDDISK ... FLUSH` command.
- The `ALTER FLASHCACHE ... FLUSH` command stops new data from being written to the flash cache and then synchronizes all data in the flash cache with the hard disks. As a result, all data is removed from the flash cache. When the flash cache is re-enabled, the flash cache activity metrics are reset.
- The `CANCEL FLUSH` option terminates an earlier flush operation, and reinstates flash caching.
- The `NOWAIT` option allows the `ALTER` command to complete while the flush operation is in progress.
- By default, 5 percent of space on Extreme Flash storage servers is used for write-back flash cache.

Example 7-36 Flushing Dirty Blocks from Flash Cell Disks

This example shows how to flush dirty blocks from all flash cell disks.

```
CellCLI> ALTER FLASHCACHE ALL FLUSH
Flash cache on FD_00_scac01cel07 successfully altered
Flash cache on FD_01_scac01cel07 successfully altered
Flash cache on FD_02_scac01cel07 successfully altered
...
Flash cache on FD_14_scac01cel07 successfully altered
Flash cache on FD_15_scac01cel07 successfully altered
```

7.7.1.5 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 [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] FLASHDISK
| HARDDISK ] | gdisk_name1 [,gdisk_name2] ... }
      { [ attribute_filters ] [ ACTIVE | INACTIVE ] [ FLUSH [NOWAIT] | CANCEL
FLUSH ] } |
        attribute_name = attribute_value [, attribute_name =
attribute_value] ...
              [ attribute_filters ] [NOWAIT] }
```

Command Options

- Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to alter grid disks on cell disks associated with the specified flash media type.

  If you do not fully specify the flash media type, `ALL FLASHDISK` is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK` on Extreme Flash (EF) storage servers containing capacity-optimized flash devices. On all other storage servers, `ALL FLASHDISK` is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.

- If the `ALL` option is specified with a media type qualifier (`FLASHDISK` or `HARDDISK`), the command only alters grid disks on cell disks associated with the specified media type.

  Starting with Oracle Exadata System Software release 24.1.0, if you specify the `ALL` option without a media type qualifier (`FLASHDISK` or `HARDDISK`), then the command alters grid disks on the cell disks associated with the primary storage media on the storage server. Specifically:

  – On Extreme Flash (EF) storage servers containing capacity-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK`.

  – On Extreme Flash (EF) storage servers containing only performance-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.

  – On all storage servers containing hard disk drives (HDDs), `ALL` with no media type qualifier is equivalent to `ALL HARDDISK`.

  Before Oracle Exadata System Software release 24.1.0, if the `ALL` option is specified without a media type qualifier, the command alters all matching grid disks regardless of media type.

- The `ACTIVE` option notifies CELLSRV to accept I/O as normal for the specified grid disks. The grid disks are visible to the database clients.

- The `INACTIVE` option makes the grid disks visible to the cell administrator, but not visible to the database clients. CELLSRV treats the grid disks as if they were offline. 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:

  If the grid disk is used by Oracle ASM and is made `INACTIVE` when currently in use by a database client, 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 `FLUSH` option synchronizes dirty data from the flash cache to the grid disks. Dirty data is data that has not been synchronized with the grid disk.

  Synchronization of dirty data can be a lengthy process, depending on the number of bytes to be synchronized. Use the following command to check the progress:

  ```
  LIST GRIDDISK ATTRIBUTES name, flushstatus
  ```

- The `CANCEL FLUSH` option terminates an earlier flush operation.
- The `NOWAIT` option allows the `ALTER` command to complete while a resize or flush operation continues, for example:
- Starting with Oracle Exadata System Software release 20.1.0, the `ALTER GRIDDISK` command accepts a `WHERE` clause, which allows for a specific subset of grid disks to be altered. For example, if you have two sets of grid disks, one for `DATA` and the other for `RECO`, you can use the `WHERE` clause to resize the grid disks from one disk group only, for example:

  ```
  ALTER GRIDDISK size=5T WHERE name like 'RECO.*'
  ```

  You can use the `WHERE` clause with either a list of grid disks or with the `ALL {FLASHDISK|HARDDISK}` option. If you use the `WHERE` clause and do not specify either a list of grid disks or the `ALL {FLASHDISK|HARDDISK}` option, then the `WHERE` clause acts against all disks. Some examples of this syntax are:

  ```
  ALTER GRIDDISK WHERE name like 'DATA.*' INACTIVE

  ALTER GRIDDISK ALL FLASHDISK FLUSH NOWAIT
  ```

Usage Notes

The attributes that can be changed with the `ALTER GRIDDISK` command are shown as `modifiable` in Example 7-103.

- The length of a grid disk name is limited to 30 characters.
- The `FLUSH` option stops new data from being cached on the specified grid disks until CELLSRV restarts, or the flush operation is canceled.
- The `FLUSH` option is valid for write back disks, not write through disks.
- The `size` attribute can be specified to expand or reduce space allocated to a grid disk.

  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.

  If the grid disk is used by Oracle ASM, the corresponding Oracle ASM disk must be resized separately.
- A grid disk should not be renamed when the grid disk is being accessed.

  If you try to rename a grid disk when it is being accessed, then the operation fails. If the grid disk is used by Oracle ASM, you can make the grid disk inactive or dismount the Oracle ASM disk group to stop access to the grid disk before renaming it.
- 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.

> **Note:**
>
> Interleaved grid disks are deprecated in Oracle Exadata System Software release 19.1.0.

- The `cachingPolicy` attribute specifies the flash caching policy for the grid disk.

  Flash cache is effectively disabled for database files located in a disk group composed of grid disks having `cachingPolicy` set to `none`. You may choose this because:

  – You want specific database files persisted directly to disk instead of utilizing write-back flash cache. This is particularly useful in situations where the durability of disk storage outweighs the performance of flash storage.

  – You want to dedicate flash cache resources to objects in other database files.

  Use the following commands to set the `cachingPolicy` attribute to `none`:

  ```
  ALTER GRIDDISK grid_disk_name CACHINGPOLICY="none"
  ALTER GRIDDISK grid_disk_name FLUSH
  -- Wait for the FLUSH to complete.
  ALTER GRIDDISK grid_disk_name CANCEL FLUSH
  ```

  To re-enable caching on the grid disk:

  ```
  ALTER GRIDDISK grid_disk_name CACHINGPOLICY="default"
  ```

  By default, OEDA configures the RECO disk group grid disks with `cachingPolicy` set to `none`. Consequently, flash cache is not used for any database files placed in RECO.

Example 7-37 Altering Grid Disk Attributes

This example shows the `ALTER` command with the `GRIDDISK` object.

```
CellCLI> ALTER GRIDDISK data1_CD_01_cell01, data2_CD_01_cell01
         comment = "This grid disk is on cell01"

CellCLI> ALTER GRIDDISK ALL INACTIVE
```

Related Topics

- Restrictions on Values of Common Attributes
  Review the following restrictions for the values of attributes used by multiple CellCLI objects.
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

7.7.1.6 ALTER IBPORT

Purpose

The `ALTER IBPORT` command performs an action on all InfiniBand Network Fabric ports, or specified InfiniBand Network Fabric ports.

Note:

This command does not apply to Oracle Exadata X8M systems.

Syntax

```
ALTER IBPORT {ALL | ibport_name [, ibport_name] ...} RESET COUNTERS
```

Usage Notes

The `RESET COUNTERS` option resets all counters on the InfiniBand Network Fabric port.

Example 7-38 Altering IBPORT Attributes

This example shows the `ALTER` command with the `IBPORT` object.

```
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.
```

7.7.1.7 ALTER IORMPLAN

Purpose

The `ALTER IORMPLAN` command updates the I/O Resource Management (IORM) plan for the cell.

The `ALTER IORMPLAN` command clauses control the IORM objective and specify directives that control access to I/O resources.

Syntax

```
ALTER IORMPLAN [ objective = iorm_objective ]
      [ catplan = { ( directive [, directive] ... ) | "" } ]
      [ dbplan = { ( directive [, directive] ... ) | "" } ]
      [ clusterplan = { ( directive [, directive] ... ) | ""  } ]
```

Parameters

- `objective`: Specifies the optimization mode for IORM. The valid `objective` values are:
  - `auto` - Use this setting for IORM to determine the best mode based on active workloads and resource plans. IORM continuously and dynamically determines the optimization objective, based on the observed workloads and enabled resource plans. This is the recommended value for most use cases, and starting with Oracle Exadata System Software release 21.2.0, this is the default setting.
  - `high_throughput` - Use this setting to optimize critical DSS workloads that require high throughput. This setting improves throughput at the cost of I/O latency.

  - `low_latency` - Use this setting to optimize critical OLTP workloads that require extremely good disk latency. This setting provides the lowest possible latency at the cost of throughput by limiting disk utilization.
  - `balanced` - Use this setting for a mixture of 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 latency and throughput.
  - `basic`: Use this setting to limit the maximum small I/O latency and otherwise disable I/O prioritization. This is the default setting in Oracle Exadata System Software release 20.1.0 and earlier.
- `catplan`: Specifies the category plan, allowing you to allocate resources primarily by the category of the work being done. If no `catplan` directives are set, then every category has an equal share of the resources by default.

  Note:

  Starting with Oracle Exadata System Software release 21.2.0, the category plan is deprecated and a warning message is issued when a category plan is set.

- `dbplan`: Specifies the interdatabase plan, allowing you to manage resource allocations among databases. If no `dbplan` directives are set, then every database has an equal share of the resources by default.
- `clusterplan`: Specifies the cluster plan, allowing you to manage resource allocations among Oracle Grid Infrastructure clusters. If no `clusterplan` directives are set, then every cluster has an equal share of the resources by default.

  Note:

  The cluster plan is first introduced in Oracle Exadata System Software release 21.2.0.

Usage Notes

- To fully enable any user-defined IORM plans (`catplan`, `dbplan`, or `clusterplan`), the IORM `objective` must be set to a value other than `basic`.
- The cluster plan (`clusterplan`) uses ASM-scoped security for cluster identification. The value of the `name` attribute must match the `asm` field in the `cellkey.ora` file, which is part of the ASM-scoped security definition for the cluster. If ASM-scoped security is not configured, then all databases in the cluster are associated with the `DEFAULT` cluster.
- Different user-defined IORM plans inter-operate as follows:
  - The `catplan` and `dbplan` can be used in combination only if the `dbplan` does not contain any directives having `type=profile`.

    In this case, directives from both plans are applied to determine the share of resources.
  - The `catplan` and `clusterplan` cannot be used in combination.

    You cannot set `clusterplan` directives when `catplan` directives exist. Likewise, you cannot set `catplan` directives when `clusterplan` directives exist.

  - The `clusterplan` and `dbplan` can be used in combination only if the `dbplan` does not contain any `allocation` or `level` directives.

    In this case, directives from both plans are applied to determine the share of resources.

- To remove the current directives and reset a `catPlan`, `dbPlan`, or `clusterplan` parameter, set the parameter to an empty string by using a pair of single or double quotation marks. The quotation marks must match. For example, `""` is correct, but `"'` is incorrect.
- Because of the command length and complexity, consider running `ALTER IORMPLAN` commands by using scripts.
- IORM is configured individually on every storage server using the `ALTER IORMPLAN` command. To deliver consistent overall system performance, ensure every storage server in the storage cluster uses the same IORM configuration settings.

Example 7-39 Setting the IORMPLAN Objective

This example shows the `ALTER IORMPLAN` command being used to set the IORM optimization mode.

```
CellCLI> ALTER IORMPLAN objective=low_latency


CellCLI> ALTER IORMPLAN objective=auto
```

Example 7-40 Resetting IORMPLAN Plans

This example shows how to reset the `IORMPLAN dbplan` and `catplan`. The first command resets the `dbplan` and `catplan` using one command. The other commands reset the `dbplan` and `catplan` individually.

```
CellCLI> ALTER IORMPLAN dbplan="", catplan=""


CellCLI> ALTER IORMPLAN dbplan=""


CellCLI> ALTER IORMPLAN catplan=""
```

- Directives for a Category Plan
- Directives for a Database Plan
- Directives for a Cluster Plan
- name Attribute
- share Attribute
- allocation and level Attributes
- limit Attribute
- flashcache Attribute
- xrmemcache Attribute
- pmemcache Attribute
- flashlog Attribute

- xrmemlog Attribute
- pmemlog Attribute
- flashcachelimit Attribute
- flashcachemin Attribute
- flashcachesize Attribute
- xrmemcachelimit Attribute
- xrmemcachemin Attribute
- xrmemcachesize Attribute
- pmemcachelimit Attribute
- pmemcachemin Attribute
- pmemcachesize Attribute
- asmcluster Attribute
- role Attribute
- type Attribute

7.7.1.7.1 Directives for a Category Plan

The directives for a category plan (`catplan`) use the following syntax:

```
( name=category_name , level=number, allocation=number )
```

Usage Notes

- The `name` attribute must be the first attribute listed in each directive. Otherwise, the order of attributes is not important.
- The category plan can have a maximum of 32 directives.
- You cannot have multiple directives with the same category name.

See the following topics for details about the attributes (`name`, `level`, and `allocation`) that are defined in each `catplan` directive.

7.7.1.7.2 Directives for a Database Plan

The directives for a database plan (`dbplan`) use the following syntax:

```
( name={ db_name | profile_name }
  [, { share=number | level=number, allocation=number }]
  [, limit=number]
  [, flashcache={on|off}]
  [, xrmemcache={on|off}]
  [, pmemcache={on|off}]
  [, flashlog={on|off}]
  [, xrmemlog={on|off}]
  [, pmemlog={on|off}]
  [, flashcachelimit=number]
  [, flashcachemin=number]
  [, flashcachesize=number]
  [, xrmemcachelimit=number]
  [, xrmemcachemin=number]
  [, xrmemcachesize=number]
  [, pmemcachelimit=number]
  [, pmemcachemin=number]
  [, pmemcachesize=number]
  [, asmcluster=asm_cluster_name]
  [, type={database|profile}]
  [, role={primary|standby}] )
```

Usage Notes

- The `name` attribute must be the first attribute listed in each directive. Otherwise, the order of attributes is not important.
- The database plan cannot contain a mixture of resource allocation directives, with some using the `share` attribute and others using the `level` and `allocation` attributes. The resource allocation directives must all use the `share` attribute, or they must all use the `level` and `allocation` attributes.
- If you use the `share` attribute to allocate I/O resources, then the database plan can have a maximum of 1024 directives. If you use the `level` and `allocation` attributes to allocate I/O resources, then the database plan can have a maximum of 32 directives.
- Only one active directive is allowed for each database name and each profile name.

See the following topics for details about the attributes that can be defined in each `dbplan` directive.

7.7.1.7.3 Directives for a Cluster Plan

The directives for a cluster plan (`clusterplan`) use the following syntax:

```
( name=cluster_name [, share=number] [, limit=number] )
```

Usage Notes

- The `name` attribute must be the first attribute listed in each directive. Otherwise, the order of attributes is not important.
- The cluster plan can have a maximum of 1024 directives.
- You cannot have multiple directives with the same cluster name.

See the following topics for details about the attributes that can be defined in each `clusterplan` directive.

7.7.1.7.4 name Attribute

Purpose

The `name` attribute identifies the entity that is the subject of the directive.

Syntax

```
ALTER IORMPLAN
   catplan = (( name=category_name, ... ) ... )


ALTER IORMPLAN
   dbplan = (( name={ db_name | profile_name }, ... ) ... )


ALTER IORMPLAN
   clusterplan = (( name=cluster_name, ... ) ... )
```

Usage Notes

- For directives in a category plan (`catplan`), the `name` attribute specifies the category name. Oracle Database manages intra-database resources using Database Resource Manager (DBRM). DBRM manages resources across consumer groups, and each consumer group is associated with a category. The `catplan` category name is associated with any DBRM category having the same name.
- For directives in a database plan (`dbplan`), the `name` attribute usually identifies the database that is associated with the directive. However, when the directive includes `type=profile`, the `name` attribute specifies the profile name.

  In directives that identify a database, the `name` value usually matches with the value of the `DB_UNIQUE_NAME` database parameter. The exception is where the directive uses the `role` attribute to manage an Oracle Data Guard configuration. For further details, see role Attribute.
- For directives in a cluster plan (`clusterplan`), the `name` attribute identifies the Oracle Grid Infrastructure cluster that is associated with the directive. The cluster plan (`clusterplan`) uses ASM-scoped security for cluster identification. The value of the `name` attribute must match the `asm` field in the `cellkey.ora` file, which is part of the ASM-scoped security definition for the cluster.
- The `name` attribute must be the first attribute in a directive.
- The `name` attribute value cannot start with an underscore (_).
- Each `name` must be followed by at least one other attribute, for example:
  - `(name=sales, share=8)`
  - `(name=oltpdg, limit=80)`
  - `(name=dwh, flashcachesize=50G)`
- There are two special `name` values:
  - `OTHER`: names a special directive that defines the resource allocation for all other entities that are not specified in the plan. All entities that are not explicitly named in the plan share the resources associated with the `OTHER` directive.

    The `OTHER` directive is used in database plans and category plans that use allocation-based resource management; that is, resource allocation defined using the `level` and `allocation` attributes. Plans with allocation-based directives must also include an `OTHER` directive.

    In a database plan, the `limit` attribute can also be defined in the `OTHER` directive.

  - `DEFAULT`: names a special directive that defines the resource allocation for each entity that is not specified in the plan. Every entity that is not explicitly named in the plan receives the resources in the `DEFAULT` directive.

    The `DEFAULT` directive is available in database plans and cluster plans that use share-based resource allocation; that is, resource allocation defined using the `share` attribute.

    If ASM-scoped security is not configured, then all databases in the cluster are associated with the `DEFAULT` cluster.

Example 7-41 Using the name Attribute in a Database Plan

```
CellCLI> ALTER IORMPLAN                                           -
         dbplan=((name=db1, limit=50),                            -
                 (name=db2, limit=50),                            -
                 (name=OTHER, level=1, allocation=25))
```

Example 7-42 Setting a Database Plan with a DEFAULT Directive

This example shows how to use the `DEFAULT` directive to set the default share allocation for all databases except `dev01` and `dev02`.

```
CellCLI> ALTER IORMPLAN                                                -
         dbplan=((name=dev01, share=1, limit=50, flashlog=off),        -
                 (name=dev02, share=1, limit=25, flashcache=off),      -
                 (name=DEFAULT, share=4))
```

7.7.1.7.5 share Attribute

Purpose

The `share` attribute controls share-based resource allocation, which specifies the relative priority for a database in a `dbplan` or a cluster in a `clusterplan`. A higher `share` value implies higher priority and more access to the I/O resources.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... share=number ... ) ... )


ALTER IORMPLAN
  clusterplan=(( name=cluster_name, ... share=number ... ) ... )
```

Usage Notes

- `share`: Specifies the resource allocation share.

  Valid values are 1 to 32, with 1 being the lowest share, and 32 being the highest share. The share value represents the relative importance of each entity. A higher `share` value implies higher priority and more access to resources. The sum of all `share` values in a plan cannot be greater than 32768.

- For share-based resource allocation, `name=DEFAULT` is used to define the default share for each entity that is not specified in the IORM plan.

- Share-based resource allocation is the recommended method for a database plan (`dbplan`). For a cluster plan (`clusterplan`), share-based resource allocation is the only option.

Example 7-43 Setting a Database Plan Using the share Attribute

This example shows how to configure `dbPlan` using the `share` attribute.

```
CellCLI> ALTER IORMPLAN                                                   -
         dbplan=((name=sales01, share=4),                                 -
                 (name=sales02, share=4),                                 -
                 (name=fin01, share=3),                                   -
                 (name=fin02, share=2),                                   -
                 (name=dev01, share=1, limit=50, flashLog=off),           -
                 (name=dev02, share=1, limit=25, flashCache=off),         -
                 (name=DEFAULT, share=2))
```

7.7.1.7.6 allocation and level Attributes

Purpose

The `level` and `allocation` attributes control allocation-based resource management. You can use allocation-based resource management to control I/O distribution for a database in a `dbplan` or a workload category in a `catplan`.

Syntax

```
 ALTER IORMPLAN
   catplan=(( name=category_name, level=number, allocation=number ) ... )


 ALTER IORMPLAN
   dbplan=(( name=db_name, ... level=number, allocation=number ... ) ... )
```

Usage Notes

- `level`: Specifies the allocation level.

  Valid values are from 1 to 8. Resources are allocated to level 1 first, and then remaining resources are allocated to level 2, and so on.
- `allocation`: Specifies the resource allocation as a percentage (0-100) within the level.

  For each `level`, the sum of `allocation` values cannot exceed 100.
- For allocation-based resource management, `name=OTHER` is used to define the resource allocation to share across all entities that are not specified in the IORM plan. Plans (`dbplan` or `catplan`) with allocation-based directives must also include a directive with `name=OTHER`.

Example 7-44 Using the level and allocation Attributes

These examples show the `ALTER` command with the `level` and `allocation` 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                                          -
         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))
```

7.7.1.7.7 limit Attribute

Purpose

The `limit` attribute specifies the maximum flash I/O utilization limit.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... limit=number ... ) ... )


 ALTER IORMPLAN
   clusterplan=(( name=cluster_name, ... limit=number ... ) ... )
```

Usage Notes

- `limit`: Specifies the maximum flash I/O utilization limit as a percentage of the available resources. The attribute applies only to I/O on flash devices, which includes flash-based grid disks and Exadata Smart Flash Cache.

  Valid values are 1 to 100. If a `limit` is specified, then excess capacity is never used by the associated database or cluster. Consequently, it is possible for flash devices to run below full capacity when limits are specified.

  > **Note:**
  >
  > Specifying low `limit` values can have a significant performance impact and is generally not advisable.

- Resource management using limits is ideal for pay-for-performance use cases but should not be used to implement fairness. Instead, use the `share` attribute to ensure equitable distribution of I/O resources.

Example 7-45 Using the limit Attribute with a Database Plan

```
CellCLI> ALTER IORMPLAN                                          -
         dbplan=((name=db1, limit=40),                           -
                 (name=db2, limit=40),                           -
                 (name=DEFAULT, limit=20))
```

7.7.1.7.8 flashcache Attribute

Purpose

The `flashcache` attribute controls use of Exadata Smart Flash Cache by a database. This ensures that cache space is reserved for mission-critical databases.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... flashcache={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use Exadata Smart Flash Cache unless it is affected by a directive that specifies `flashcache=off`.
- `flashcache=off` is invalid in a directive that contains the `flashcachemin`, `flashcachelimit`, or `flashcachesize` attributes.

Example 7-46 Setting Flash Cache Use in a Database Plan

This example shows how to enable Flash Cache use in a database plan.

```
CellCLI> ALTER IORMPLAN                                            -
         dbplan=((name=sales_prod, flashcache=on),                 -
                 (name=sales_dev, flashcache=on),                  -
                 (name=sales_test, flashcache=off),                -
                 (name=DEFAULT, flashcache=off))
```

7.7.1.7.9 xrmemcache Attribute

Purpose

The `xrmemcache` attribute controls use of the Exadata RDMA Memory Cache (XRMEM cache) by a database. This ensures that cache space is reserved for mission-critical databases.

Syntax

```
ALTER IORMPLAN
   dbplan=(( name=db_name, ... xrmemcache={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use the XRMEM cache unless it is affected by a directive that specifies `xrmemcache=off`.
- `xrmemcache=off` is invalid in a directive that contains the `xrmemcachemin`, `xrmemcachelimit`, or `xrmemcachesize` attributes.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache.

  For backward compatibility, on Exadata X8M and X9M systems, you can use `pmemcache` instead of `xrmemcache` in the `ALTER IORMPLAN` command. However, starting with Oracle

Exadata System Software release 23.1.0, output from the `LIST IORMPLAN` command only displays `xrmemcache`.

Example 7-47 Setting XRMEM Cache Use in a Database Plan

This example shows how to enable XRMEM cache use in a database plan.

```
CellCLI> ALTER IORMPLAN                                         -
         dbplan=((name=sales_prod, xrmemcache=on),              -
                 (name=sales_dev, xrmemcache=off),              -
                 (name=sales_test, xrmemcache=off),             -
                 (name=DEFAULT, xrmemcache=off))
```

7.7.1.7.10 pmemcache Attribute

Purpose

The `pmemcache` attribute controls use of the persistent memory (PMEM) cache by a database. This ensures that cache space is reserved for mission-critical databases.

Note:

This attribute applies to Oracle Exadata System Software releases before 23.1.0. Otherwise, see xrmemcache Attribute.

Syntax

```
ALTER IORMPLAN
   dbplan=(( name=db_name, ... pmemcache={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use the PMEM cache unless it is affected by a directive that specifies `pmemcache=off`.
- `pmemcache=off` is invalid in a directive that contains the `pmemcachemin`, `pmemcachelimit`, or `pmemcachesize` attributes.

Example 7-48 Setting PMEM Cache Use in a Database Plan

This example shows how to enable PMEM cache use in a database plan.

```
CellCLI> ALTER IORMPLAN                                         -
         dbplan=((name=sales_prod, pmemcache=on),             -
                 (name=sales_dev, pmemcache=off),             -
                 (name=sales_test, pmemcache=off),            -
                 (name=DEFAULT, pmemcache=off))
```

7.7.1.7.11 flashlog Attribute

Purpose

The `flashlog` attribute controls use of Exadata Smart Flash Log by a database. This ensures that Exadata Smart Flash Log is reserved for mission-critical databases.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... flashlog={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use Exadata Smart Flash Log unless it is affected by a directive that specifies `flashlog=off`.

Example 7-49 Setting Flash Log Use in a Database Plan

This example shows how to control Flash Log use in a database plan.

```
CellCLI> ALTER
IORMPLAN
 -
         dbplan=((name=oltp, level=1, allocation=80, flashcache=on,
flashlog=on),             -
                 (name=dss, level=1, allocation=20, limit=50, flashcache=off,
flashlog=off),  -
                 (name=OTHER, level=2,
allocation=100),                                       -
                 (name=DEFAULT, flashcache=off, flashlog=off))
```

7.7.1.7.12 xrmemlog Attribute

Purpose

The `xrmemlog` attribute controls use of the XRMEM log by a database. This ensures that commit acceleration is reserved for mission-critical databases.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... xrmemlog={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use XRMEM log unless it is affected by a directive that specifies `xrmemlog=off`.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory commit accelerator, previously known as PMEM log, is now called XRMEM log.

  For backward compatibility, on Exadata X8M and X9M systems, you can use `pmemlog` instead of `xrmemlog` in the `ALTER IORMPLAN` command. However, starting with Oracle

Exadata System Software release 23.1.0, output from the `LIST IORMPLAN` command only displays `xrmemlog`.

Example 7-50 Setting XRMEM Log Use in a Database Plan

This example shows how to control XRMEM Log use in a database plan.

```
CellCLI> ALTER
IORMPLAN
-
         dbplan=((name=oltp, level=1, allocation=80, xrmemcache=on,
xrmemlog=on),              -
                 (name=dss, level=1, allocation=20, limit=50, xrmemcache=off,
xrmemlog=off),   -
                 (name=OTHER, level=2,
allocation=100),                                      -
                 (name=DEFAULT, xrmemcache=off, xrmemlog=off))
```

7.7.1.7.13 pmemlog Attribute

Purpose

The `pmemlog` attribute controls use of the persistent memory commit accelerator (PMEM log) by a database. This ensures that commit acceleration is reserved for mission-critical databases.

Note:

This attribute applies to Oracle Exadata System Software releases before 23.1.0. Otherwise, see xrmemlog Attribute.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... pmemlog={on|off}  ... ) ... )
```

Usage Notes

- By default, any database can use PMEM log unless it is affected by a directive that specifies `pmemlog=off`.

Example 7-51 Setting PMEM Log Use in a Database Plan

This example shows how to control PMEM Log use in a database plan.

```
CellCLI> ALTER
IORMPLAN
-
         dbplan=((name=oltp, level=1, allocation=80, pmemcache=on,
pmemlog=on),               -
                 (name=dss, level=1, allocation=20, limit=50, pmemcache=off,
pmemlog=off),    -
                 (name=OTHER, level=2,
allocation=100),                                        -
                     (name=DEFAULT, pmemcache=off, pmemlog=off))
```

7.7.1.7.14 flashcachelimit Attribute

Purpose

The `flashcachelimit` attribute defines a soft limit for space usage in Exadata Smart Flash Cache. If the cache is not full, the limit can be exceeded.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... flashcachelimit=number ... ) ... )
```

Usage Notes

- You specify the value for `flashcachelimit` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `flashcachelimit` must be at least 4 MB.
- The `flashcachelimit` and `flashcachesize` attributes cannot be specified in the same directive.
- The value for `flashcachelimit` cannot be smaller than `flashcachemin`, if it is specified.

Example 7-52 Specifying Flash Cache Quotas in a Database Plan

This example shows how to configure flash cache quotas in a database plan.

```
CellCLI> ALTER
IORMPLAN                                                              -
           dbplan=((name=prod, share=8,
flashCacheMin=400M),                        -
                     (name=dev,  share=2, flashCacheMin=100M,
flashCacheLimit=200M), -
                     (name=test, share=1, limit=40, flashCacheLimit=20M))
```

7.7.1.7.15 flashcachemin Attribute

Purpose

The `flashcachemin` attribute specifies a minimum guaranteed space allocation in Exadata Smart Flash Cache.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... flashcachemin=number ... ) ... )
```

Usage Notes

- You specify the value for `flashcachemin` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `flashcachemin` must be at least 4 MB.
- In any plan, the sum of all `flashcachemin` values cannot exceed the size of Exadata Smart Flash Cache.
- If `flashcachelimit` is specified, then the value for `flashcachemin` cannot exceed `flashcachelimit`.
- If `flashcachesize` is specified, then the value for `flashcachemin` cannot exceed `flashcachesize`.

7.7.1.7.16 flashcachesize Attribute

Purpose

The `flashcachesize` attribute defines a hard limit for space usage in Exadata Smart Flash Cache. The limit cannot be exceeded, even if the cache is not full.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... flashcachesize=number ... ) ... )
```

Usage Notes

- You specify the value for `flashcachesize` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `flashcachesize` must be at least 4 MB.
- The `flashcachelimit` and `flashcachesize` attributes cannot be specified in the same directive.
- The value for `flashcachesize` cannot be smaller than `flashcachemin`, if it is specified.
- In an IORM plan, if the size of Exadata Smart Flash Cache can accommodate all of the `flashcachemin` and `flashcachesize` allocations, then each `flashcachesize` definition represents a guaranteed space allocation.

  However, starting with Oracle Exadata System Software release 19.2.0 you can use the `flashcachesize` attribute to over-provision space in Exadata Smart Flash Cache. Consequently, if the size of Exadata Smart Flash Cache cannot accommodate all of the `flashcachemin` and `flashcachesize` allocations, then only `flashcachemin` is guaranteed.

7.7.1.7.17 xrmemcachelimit Attribute

Purpose

The `xrmemcachelimit` attribute defines a soft limit for space usage in the Exadata RDMA Memory Cache (XRMEM cache). If the cache is not full, the limit can be exceeded.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... xrmemcachelimit=number ... ) ... )
```

Usage Notes

- You specify the value for `xrmemcachelimit` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `xrmemcachelimit` must be at least 4 MB.
- The `xrmemcachelimit` and `xrmemcachesize` attributes cannot be specified in the same directive.
- The value for `xrmemcachelimit` cannot be smaller than `xrmemcachemin`, if it is specified.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache.

  For backward compatibility, on Exadata X8M and X9M systems, you can use `pmemcachelimit` instead of `xrmemcachelimit` in the `ALTER IORMPLAN` command. However, starting with Oracle Exadata System Software release 23.1.0, output from the `LIST IORMPLAN` command only displays `xrmemcachelimit`.

7.7.1.7.18 xrmemcachemin Attribute

Purpose

The `xrmemcachemin` attribute specifies a minimum guaranteed space allocation in the Exadata RDMA Memory Cache (XRMEM cache).

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... xrmemcachemin=number ... ) ... )
```

Usage Notes

- You specify the value for `xrmemcachemin` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `xrmemcachemin` must be at least 4 MB.
- In any plan, the sum of all `xrmemcachemin` values cannot exceed the size of the XRMEM cache.
- If `xrmemcachelimit` is specified, then the value for `xrmemcachemin` cannot exceed `xrmemcachelimit`.
- If `xrmemcachesize` is specified, then the value for `xrmemcachemin` cannot exceed `xrmemcachesize`.

- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache.

  For backward compatibility, on Exadata X8M and X9M systems, you can use `pmemcachemin` instead of `xrmemcachemin` in the `ALTER IORMPLAN` command. However, starting with Oracle Exadata System Software release 23.1.0, output from the `LIST IORMPLAN` command only displays `xrmemcachemin`.

7.7.1.7.19 xrmemcachesize Attribute

Purpose

The `xrmemcachesize` attribute defines a hard limit for space usage in the Exadata RDMA Memory Cache (XRMEM cache). The limit cannot be exceeded, even if the cache is not full.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... xrmemcachesize=number ... ) ... )
```

Usage Notes

- You specify the value for `xrmemcachesize` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `xrmemcachesize` must be at least 4 MB.
- The `xrmemcachelimit` and `xrmemcachesize` attributes cannot be specified in the same directive.
- The value for `xrmemcachesize` cannot be smaller than `xrmemcachemin`, if it is specified.
- In an IORM plan, if the size of the XRMEM cache can accommodate all of the `xrmemcachemin` and `xrmemcachesize` allocations, then each `xrmemcachesize` definition represents a guaranteed space allocation.

  However, you can use the `xrmemcachesize` attribute to over-provision space in the XRMEM cache. Consequently, if the XRMEM cache size cannot accommodate all of the `xrmemcachemin` and `xrmemcachesize` allocations, then only `xrmemcachemin` is guaranteed.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache.

  For backward compatibility, on Exadata X8M and X9M systems, you can use `pmemcachesize` instead of `xrmemcachesize` in the `ALTER IORMPLAN` command. However, starting with Oracle Exadata System Software release 23.1.0, output from the `LIST IORMPLAN` command only displays `xrmemcachesize`.

7.7.1.7.20 pmemcachelimit Attribute

Purpose

The `pmemcachelimit` attribute defines a soft limit for space usage in the persistent memory (PMEM) cache. If the cache is not full, the limit can be exceeded.

> **Note:**
>
> This attribute applies to Oracle Exadata System Software releases before 23.1.0. Otherwise, see xrmemcachelimit Attribute.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... pmemcachelimit=number ... ) ... )
```

Usage Notes

- You specify the value for `pmemcachelimit` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `pmemcachelimit` must be at least 4 MB.
- The `pmemcachelimit` and `pmemcachesize` attributes cannot be specified in the same directive.
- The value for `pmemcachelimit` cannot be smaller than `pmemcachemin`, if it is specified.

7.7.1.7.21 pmemcachemin Attribute

Purpose

The `pmemcachemin` attribute specifies a minimum guaranteed space allocation in the persistent memory (PMEM) cache.

> **Note:**
>
> This attribute applies to Oracle Exadata System Software releases before 23.1.0. Otherwise, see xrmemcachemin Attribute.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... pmemcachemin=number ... ) ... )
```

Usage Notes

- You specify the value for `pmemcachemin` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `pmemcachemin` must be at least 4 MB.
- In any plan, the sum of all `pmemcachemin` values cannot exceed the size of the PMEM cache.

- If `pmemcachelimit` is specified, then the value for `pmemcachemin` cannot exceed `pmemcachelimit`.
- If `pmemcachesize` is specified, then the value for `pmemcachemin` cannot exceed `pmemcachesize`.

7.7.1.7.22 pmemcachesize Attribute

Purpose

The `pmemcachesize` attribute defines a hard limit for space usage in the persistent memory (PMEM) cache. The limit cannot be exceeded, even if the cache is not full.

Note:

This attribute applies to Oracle Exadata System Software releases before 23.1.0. Otherwise, see xrmemcachesize Attribute.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... pmemcachesize=number ... ) ... )
```

Usage Notes

- You specify the value for `pmemcachesize` in bytes. You can also use the suffixes `M` (megabytes), `G` (gigabytes), or `T` (terabytes) to specify larger values. For example, `300M`, `150G`, or `1T`.
- The value for `pmemcachesize` must be at least 4 MB.
- The `pmemcachelimit` and `pmemcachesize` attributes cannot be specified in the same directive.
- The value for `pmemcachesize` cannot be smaller than `pmemcachemin`, if it is specified.
- In an IORM plan, if the size of the PMEM cache can accommodate all of the `pmemcachemin` and `pmemcachesize` allocations, then each `pmemcachesize` definition represents a guaranteed space allocation.

  However, you can use the `pmemcachesize` attribute to over-provision space in the PMEM cache. Consequently, if the PMEM cache size cannot accommodate all of the `pmemcachemin` and `pmemcachesize` allocations, then only `pmemcachemin` is guaranteed.

7.7.1.7.23 asmcluster Attribute

Purpose

Starting with Oracle Exadata System Software release 19.1.0, you can use the `asmcluster` attribute to distinguish between databases with the same name running in different Oracle ASM clusters.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... asmcluster=asm_cluster_name ... ) ... )
```

Usage Notes

- To use the `asmcluster` attribute, ASM-scoped security must be configured.
- The value of the `asmcluster` attribute must match the `asm` field in the `cellkey.ora` file, which is part of the ASM-scoped security definition for the cluster.
- You cannot use the `asmcluster` attribute in conjunction with allocation-based resource management (using the `level` and `allocation` attributes).

Example 7-53 Using the asmcluster Attribute

This example shows how to use the `asmcluster` attribute to distinguish between databases with the same name.

```
CellCLI> ALTER
IORMPLAN                                                          -
         dbPlan=((name=prod1, share=4, flashcachemin=5G,
asmcluster=cluster1),    -
                 (name=prod1, share=2, limit=80,
asmcluster=cluster2),            -
                 (name=prod2, share=2, flashcachelimit=2G,
asmcluster=cluster1),  -
                 (name=DEFAULT, share=1, flashcachelimit=1G))
```

7.7.1.7.24 role Attribute

Purpose

The `role` attribute enables you specify different plan directives based on the Oracle Data Guard database role. The directive for a database is applied only when the database is in the specified role. New directives are automatically applied by IORM when a database changes roles because of an Oracle Data Guard switchover or fail-over.

Syntax

```
ALTER IORMPLAN
  dbplan=(( name=db_name, ... role={primary|standby} ... ) ... )
```

Usage Notes

- Directives using the `role` attribute must be defined in matched pairs, using the same `name` value. That is, for each directive that specifies `role=primary`, you must have a corresponding directive that specifies `role=standby`. Likewise, each standby directive must have a matching primary directive.
- You must use the same `name` value to identify the database in both the primary directive and the standby directive. To achieve this, you can:
  - Set the `name` attribute to the value of the `DB_UNIQUE_NAME` parameter in the standby database, and set the `DB_NAME` parameter in the primary database to the same value.

This option allows you to define specific directives to manage multiple standby databases.

  – Set the `name` attribute to the value of the `DB_NAME` database parameter, which will be the same in the primary and standby databases. This option is not recommended for cases supporting multiple standby databases because all of the cell metrics relating to the standby databases are aggregated under one name.

- If the `role` attribute is not specified, then the directive applies regardless of the database role.
- For allocation-based resource management (using the `level` and `allocation` attributes), the sum of the allocation values (including `OTHER`) cannot exceed 100 for every combination of `level` and `role`.
- The `role` attribute cannot be specified in `DEFAULT` or `OTHER` directives.

Example 7-54 Using the role Attribute with Allocation-Based Resource Management

```
CellCLI> ALTER IORMPLAN                                                     -
         dbplan=((name=sales_prod, level=1, allocation=30, role=primary),   -
                 (name=sales_prod, level=1, allocation=20, role=standby),   -
                 (name=sales2, level=1, allocation=20),                     -
                 (name=other, level=3, allocation = 50))
```

Example 7-55 Using the role Attribute with Share-Based Resource Allocation

```
CellCLI> ALTER IORMPLAN                                               -
         dbplan=((name=salesprod, share=4, role=primary),             -
                 (name=salesprod, share=1, limit=50, role=standby),   -
                 (name=finance, share=4),                             -
                 (name=hr, share=2))
```

7.7.1.7.25 type Attribute

Purpose

The `type` attribute enables you to create a profile, or template, to ease management and configuration of resource plans in environments with many databases.

Syntax

```
 ALTER IORMPLAN
   dbplan=(( name=db_name, ... type={database|profile} ... ) ... )
```

Usage Notes

- `type`: Specifies the directive type. Valid values are `database` or `profile`:
  – `type=database`: Specifies a directive that applies to a specific database. If `type` in not specified, then the directive defaults to the `database` type.
  – `type=profile`: Specifies a directive that applies to a profile rather than a specific database. To associate a database with an IORM profile, you must set the database initialization parameter `db_performance_profile` to the value of the profile `name`. Databases that map to a profile inherit the settings specified in the profile.

A profile directive can contain any attributes except `level`, `allocation`, `asmcluster`, and `role`.

A profile name cannot be `OTHER` or `DEFAULT`.

- The `dbplan` can contain a combination of profile and database directives.

Example 7-56 Creating a Profile

This example shows how to specify profiles as part of a database plan.

```
CellCLI> ALTER IORMPLAN                                    -
         dbplan=((name=gold, share=10, type=profile),   -
                 (name=silver, share=5, type=profile), -
                 (name=bronze, share=1, type=profile))
```

7.7.1.8 ALTER LUN

Purpose

The `ALTER LUN` command re-enables 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.

Example 7-57 Re-enabling a LUN

This example shows the `ALTER` command with the `LUN` object.

```
CellCLI> ALTER LUN 'x:7' REENABLE FORCE

CellCLI> ALTER LUN ALL REENABLE FORCE
```

7.7.1.9 ALTER METRICDEFINITION

Purpose

The `ALTER METRICDEFINITION` command controls various metric attributes.

Syntax

```
ALTER METRICDEFINITION metric_name [, metric_name ]... attribute_name =
attribute_value [, attribute_name = attribute_value]...


ALTER METRICDEFINITION attribute_name = attribute_value [, attribute_name =
attribute_value]... attribute_filters
```

Usage Notes

- In the command:
  - *metric_name*: Identifies a specific metric definition to alter.
  - *attribute_filters*: Identifies the metric definitions to alter by filtering according to attribute values.
  - *attribute_name=attribute_value*: Specifies an attribute setting:
    - The `finegrained` attribute controls whether the metric is enabled for fine-grained collection.

      Specify `finegrained=enabled` to enable the metric for fine-grained collection, or specify `finegrained=disabled` to disable fine-grained collection for the metric.
    - The `streaming` attribute controls whether the metric is included in the metric stream.

      Specify `streaming=enabled` to include the metric in the metric stream, or specify `streaming=disabled` to exclude the metric from the metric stream.
    - The `retentionPolicy` attribute specifies the retention policy for metric observations.

      When `retentionPolicy=Default`, the retention period for the associated metric is governed by the `metricHistoryDays` cell attribute. If `retentionPolicy=Annual`, the associated metric has a one-year retention period.

      By default, a subset of key metrics are retained for up to one year (`retentionPolicy=Annual`).

      To restore a metric to the original `retentionPolicy` setting defined in Oracle Exadata System Software, set the `retentionPolicy` attribute to an empty string (`retentionPolicy=''`).

      However, regardless of the `retentionPolicy` setting, historical metric observations are purged automatically if the server detects a shortage of storage space for the metric history repository.
- By default, a set of key performance metrics is automatically enabled for fine-grained collection and automatically included in the metric stream. However, you can customize the `finegrained` and `streaming` attributes independently. In other words, you can enable a metric for fine-grained collection without including it in the metric stream. Likewise, you can include a metric in the metric stream when it is not in the fine-grained collection.

Example 7-58 Alter a Specific Metric Definition

This example shows the command to enable fine grained metric collection for the metric that shows the Ethernet network interface transfer rate (`N_NIC_KB_TRANS_SEC`).

```
CellCLI> ALTER METRICDEFINITION N_NIC_KB_TRANS_SEC finegrained=enabled
```

Example 7-59 Alter a List of Metric Definitions

This example shows the command to disable fine grained metric collection for a listed set of metrics.

```
CellCLI> ALTER METRICDEFINITION N_MB_SENT,N_MB_RECEIVED finegrained=disabled
```

Example 7-60 Alter Metric Definitions Using a Filter

This example shows the command to include in the metric stream all of the metrics specified by the attribute filter.

```
CellCLI> ALTER METRICDEFINITION streaming=enabled WHERE name LIKE 'N_NIC.*'
```

Related Topics

- Real-Time Insight
  You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems.
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

7.7.1.10 ALTER OFFLOADGROUP

Purpose

The `ALTER OFFLOADGROUP` command enables you to alter modifiable attributes of offload groups, and also to restart, start up, and shut down services.

Syntax

```
ALTER OFFLOADGROUP { offloadgroup1 [,offloadgroup2, ...] }
{attribute_name = attribute_value [, attribute_name = attribute_value ...]]
| STARTUP | RESTART | SHUTDOWN }
```

Usage Notes

- The *offloadgroupN* (where *N* is a number) parameters specify the names of the offload groups whose attributes you want to modify, or that you want to start, shut down, or restart.
- The *attribute_name* and *attribute_value* parameters specify the name and value of the attribute you want to modify.
- The `STARTUP` parameter specifies that the offload group(s) is to be started.
- The `RESTART` parameter specifies that the offload group(s) is to be shut down, then started.
- The `SHUTDOWN` parameter specifies that the offload group(s) is to be shut down.

Examples

Example 7-61 Updating the "Comment" Attribute

```
ALTER OFFLOADGROUP offloadgroup1 comment='System group'
```

Example 7-62 Starting up the Offload Group Named "offloadgroup1"

```
ALTER OFFLOADGROUP offloadgroup1 startup
```

Related Topics

- LIST OFFLOADGROUP

7.7.1.11 ALTER PHYSICALDISK

Purpose

The `ALTER PHYSICALDISK` command prepares a disk for replacement.

Syntax

```
ALTER PHYSICALDISK { ALL [ HARDDISK ] | disk_id1 [,disk_id2]  ...  }
 { DROP FOR REPLACEMENT [ MAINTAIN REDUNDANCY [ NOWAIT ] | FORCE ] |
REENABLE }
```

Usage Notes

- The `DROP FOR REPLACEMENT` option:
  - Is supported only for hot-pluggable disks
  - Checks if it is safe to proactively replace the specified disks. For example, if you attempt to drop the last good system disk, then replacing it would cause the system to crash.
  - If used without the `MAINTAIN REDUNDANCY` option, the `DROP FOR REPLACEMENT` option off-lines any data grid disks that exist on the physical disks.
  - Flushes the disk controller cache for physical disks based on hard disk drives
  - Prepares devices so that they can be removed online. For example, for flash devices this option powers off the associated PCIe slot.
- The following options are available in conjunction with the `DROP FOR REPLACEMENT` option:
  - `MAINTAIN REDUNDANCY` maintains data redundancy by rebalancing data before dropping the corresponding ASM disks. Without this option, the specified grid disks are off-lined immediately, and data redundancy is affected until the physical disks are re-enabled.
  - In addition to the `MAINTAIN REDUNDANCY` option, `NOWAIT` allows the `ALTER PHYSICALDISK` command to complete immediately while the `DROP FOR REPLACEMENT MAINTAIN REDUNDANCY` operation runs asynchronously in the background.
  - `FORCE` ignores inbuilt safety checks and performs the command even if it is deemed unsafe. This option cannot be used in conjunction with the `MAINTAIN REDUNDANCY` option.

- `REENABLE` re-enables a normal physical disk that was dropped for replacement.
- `SERVICELED` is now obsolete. If you use this option, you will get the error message: `CELL-04591`.

Examples

Example 7-63 Dropping a Normal, Functioning Physical Disk

This example shows how to drop a physical disk.

```
CellCLI> ALTER PHYSICALDISK FLASH_5_1 DROP FOR REPLACEMENT
```

Example 7-64 Re-enabling a Physical Disk

This example shows how to re-enable a physical disk.

```
CellCLI> ALTER PHYSICALDISK 12:3 REENABLE
```

Related Topics

- LIST PHYSICALDISK

7.7.1.12 ALTER PMEMCACHE

Purpose

The `ALTER PMEMCACHE` command can alter the set of cell disks used by PMEM cache, flush dirty blocks from PMEM cache, or cancel a previous flush operation on the specified cell disks to re-enable caching.

> **Note:**
>
> The `ALTER PMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
ALTER PMEMCACHE { ALL | CELLDISK="cdisk1 [,cdisk2] ..." [FORCE]}}
      {FLUSH [NOWAIT] | CANCEL FLUSH}
```

Usage Notes

- The `ALL` option affects all available PMEM cell disks.
- The `CELLDISK` option allows you to specify individual cell disks. *cdiskn* represents a cell disk name.
- The `FLUSH` option synchronizes dirty data from the PMEM cache to the cell disks. Dirty data is data that has been modified in the cache but not yet synchronized with the data on disk. Synchronization of dirty data can be a lengthy process, depending on the number of bytes

to be synchronized. Use the following command to check the progress of the flush operation:

```
LIST CELLDISK ATTRIBUTES name, flushstatus, flusherror
```

> **Note:**
>
> The `FLUSH` option stops new data from being cached on the PMEM cache until CELLSRV restarts, or the flush operation is canceled with the `ALTER PMEMCACHE CANCEL FLUSH` command.

- The `CANCEL FLUSH` option terminates an earlier flush operation, and reinstates PMEM caching.
- The `NOWAIT` option allows the `ALTER` command to complete while the flush operation is in progress.
- The `FORCE` option can be used to forcefully change the set of cell disks used by the PMEM cache.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `ALTER XRMEMCACHE` instead of `ALTER PMEMCACHE`.

Example 7-65 Using the ALTER PMEMCACHE Command

The following command specifies that the PMEM cache uses all PMEM cell disks.

```
CellCLI> ALTER PMEMCACHE ALL
```

The following command specifies that the PMEM cache uses only two PMEM cell disks, ignoring any errors or warnings.

```
CellCLI> ALTER PMEMCACHE CELLDISK='PM_01_mycell, PM_03_mycell' FORCE
```

The following command specifies initiates a flush operation for all PMEM cell disks, and returns the prompt before the operation completes.

```
CellCLI> ALTER PMEMCACHE ALL FLUSH NOWAIT
```

7.7.1.13 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

The following example shows the `ALTER` command with the `QUARANTINE` object.

Example 7-66 Altering a Quarantine

```
CELLCLI> ALTER QUARANTINE 12 comment='bugX'
```

7.7.1.14 ALTER SOFTWAREUPDATE

Purpose

You can schedule software updates by setting `SOFTWAREUPDATE` attributes. The `ALTER SOFTWAREUPDATE` command enables you to alter modifiable Software Update attributes, to validate the pre-requirements for the software update, or to start the upgrade immediately.

You can also run the `ALTER SOFTWAREUPDATE` command using `exacli`.

Syntax

```
ALTER SOFTWAREUPDATE {VALIDATE PREREQ | UPGRADE [FORCE] | attribute =
attribute value [,attribute = attribute value...]}
```

Options for the ALTER SOFTWAREUPDATE Command

- `VALIDATE PREREQ`

  Run software update check pre-requirement steps now. This will download the software update pre-requirement code for the update specified by the `store` attribute. These checks are run automatically as part of update. Use this option only if you want to run prerequisite checks explicitly. Any error found will be displayed. A stateful alert will be raised if any error is found by the `VALIDATE PREREQ` command.

- `UPGRADE [FORCE]`

  Run the software update (including the pre-requirement steps) now, using the software location specified by the Software Update `store` attribute. Use this command if you want to perform the update now rather than wait for the time specified by the Software Update `time` attribute.

  If `FORCE` is specified, then the upgrade continues despite any pre-requirement check errors.

- *attribute* = *attribute value*

  Modify the specified Software Update attributes to the values provided.

Attributes for SOFTWAREUPDATE

The following attributes for the `ALTER SOFTWAREUPDATE` command are modifiable:

- `frequency`: Storage server updates can automatically be done periodically by setting the `frequency` attribute to the desired frequency. You can specify one of the following values: `{none | daily | weekly | biweekly }`. If the value specified for `frequency` is `''` or `none`, then the scheduled update is only done once. The value `none` can be used for the `frequency` in Oracle Exadata System Software release 19.1.0 or later.
- `name`: The name of the patch to use in the update, which includes the software version string such as 18.1.1.0.0.171018. If you use the `ALTER SOFTWAREUPDATE UPGRADE` or `ALTER SOFTWAREUPDATE VALIDATE PREREQ` command, then the software store is checked and the

`name` attribute is set automatically (if not already set) to the latest available version in the software store. Otherwise, the `name` attribute has a value of `unknown`. If there are multiple software versions at the store site, then this attribute can be used to specify which version should be used.

Patches downloaded from My Oracle Support use a different name format, for example `p26875767_181100_Linux-x86-64.zip`. If you are using Oracle Exadata System Software 18c (18.1.0) or 18c (18.1.1), then you must rename the downloaded patch file so it has a name like `18.1.1.0.0.171018`. Starting with Oracle Exadata System Software release 18.1.2, the `ALTER SOFTWAREUPDATE` command accepts patch names of the form `p26875767_181100`.

- `store`: A URL for the location of the software store.

  If you specify a network store location using the `HTTP` or `HTTPS` protocol:

  – The URL must be accessible using the management network or RDMA Network Fabric.

  – MS automatically finds and downloads the required software update files.

  – The local staging location for the downloaded software update files is `/root/swupdate` or `/var/swupdate`.

  Alternatively, you can use the `FILE` protocol to specify a URL for a local store. When using a local store:

  – MS does not need to download the patch zip files or perform the associated space checks.

  – MS does not manage the local software store content. You must download the required patch files before patching and remove them afterward to free up space.

  – The file URL must use one of the following formats:

    * `file:///`*`localpath`*

    * `file:/`*`localpath`*

- `time`: A future date and time at which the software update should be performed. The time can be specified as a local informal date and time, for example `"1 AM, next Tuesday"`. If the date and time is valid then the output from setting this attribute shows the interpreted time in standard format with timezone offset, such as `2017-08-22T01:00:00-08:00`.

  If you set this attribute to the empty string, `""`, then it cancels the scheduled software update.

- `timeLimitInMinutes`: An update may wait for other storage servers to complete their updates in order to preserve disk group redundancy. By default, there is no limit on the amount of time which can be spent waiting to update. This attribute may be set to a positive, maximum integer which represents the number of minutes a storage server will spend waiting to update. If an update does not start within the time specified by the limit, then the update is canceled and an update alert is reported.

Usage Notes

- The storage server software is initially updated on the unused system partition. Then, according to the desired schedule, the storage server activates the update by booting from the updated partition.
- Software download and the prerequisite check will begin up to a week before the scheduled update time.

- The update progress can be monitored by displaying the non-modifiable Software Update `status` attribute.
- Software updates do not occur if the upgrade software is already installed
- You can use `dcli` or `exacli` to schedule and install updates using the `ALTER SOFTWAREUPDATE` command.
- The Software Update feature supports using HTTPS transport for software downloads. When using HTTPS, TLS certificate checks are performed by default. If the remote server's certificate cannot be validated then the following error is reported:

  ```
  CELL-00076: An error occurred during download of software update:
  source https://hostname:port is not available.
  CELL-00092: The store's TLS certificate cannot be authenticated with known
  CA certificates.
  ```

  This can happen if the remote server uses a self-signed certificate or if the remote server uses a certificate signed by a certificate authority (CA) that is not included in the storage server's CA store. You can use the following procedure to add a CA certificate to the storage server's CA store. This is a security setup step which requires shell access as `root` on the storage server.

  1. Get a CA certificate that can verify the remote server. The certificate should be stored in PEM or DER file format.
  2. Copy the file to the storage server at this directory: `/etc/pki/ca-trust/source/anchors/`
  3. Run following commands:

     ```
     update-ca-trust enable

     update-ca-trust extract
     ```

  Use the operating system `man` utility to get more information about the `update-ca-trust` command.

Examples

Example 7-67 Modifying the Software Update time Attribute

Modify the scheduled time of the next software update to 1 a.m. on Thursday.

```
CellCLI> ALTER SOFTWAREUPDATE time = "1 AM Thursday"
CELL update is scheduled to begin at 2017-08-24T01:00:00-08:00
```

Example 7-68 Setting the Software Update store Attribute

This example shows how to set the store attribute to a location that uses HTTPS protocol.

```
ALTER SOFTWAREUPDATE store="https://my-exa-store/cell"
```

Example 7-69   Starting a Software Update Immediately

This example shows how to immediately start a software update using the attribute values already specified.

```
ALTER SOFTWAREUPDATE UPGRADE FORCE
```

7.7.1.15 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 Describing the THRESHOLD Object.

Examples

The following example shows how to alter threshold attributes.

Example 7-70   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
```

Related Topics

- CREATE THRESHOLD

7.7.1.16 ALTER USER

Purpose

The `ALTER USER` command changes the attributes of a user role.

Syntax

```
ALTER USER user1 attribute_name1 = attribute_value1      \
[, attribute_name2 = attribute_value2, ...]
```

Usage Notes

- The user name cannot be `root`, `celladmin` or `cellmonitor`. Those are reserved.
- The user name should be unique.

- The system prompts for a password for the new user. The password must have 12 to 40 alphanumeric characters or special characters !@#$%^&*() with at least one digit, one lowercase letter, and one uppercase letter. Starting with Oracle Exadata System Software release 18.1.0.0.0, the password can be 8 to 40 characters in length and can also utilize the special characters - and _.
- The new password cannot be the same as the current password for the user.

Example 7-71 Using the ALTER USER Command

This example shows how to change a user's password.

```
CellCLI> ALTER USER sjones password=TOPsecret2345
```

7.7.1.17 ALTER XRMEMCACHE

Purpose

Starting with Oracle Exadata System Software release 23.1.0, the `ALTER XRMEMCACHE` command is equivalent to the `ALTER PMEMCACHE` command.

> **Note:**
>
> The `ALTER XRMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

7.7.2 ASSIGN KEY

Purpose

The `ASSIGN KEY` command assigns or removes a security key to or from a client.

Syntax

```
ASSIGN KEY FOR [ASMCLUSTER] 'client_name1' = 'key-value1' [, 'client_name2' =
'key-value2'...]

ASSIGN KEY FOR CELL 'key-value'

ASSIGN KEY FOR [REMOTE | LOCAL] CELL 'client_name1' = 'key-value1' [,
'client_name2' = 'key-value2'...]
```

Options

- *client_name* is an alias that is the unique name (`DB_UNIQUE_NAME`) for a database client or Oracle ASM cluster.

  > **Note:**
  >
  > The client name or Oracle ASM cluster name not case-sensitive. For example, `ASM1` and `asm1` are treated as the same value.

- *key-value* is a hexadecimal string key that is assigned to the client as a security key. The key value is generated with the `CREATE KEY` command. The key values assigned with the `ASSIGN` command must match the key in the client `cellkey.ora` file on the database servers. The key value can be the same for multiple clients that need the same access. An empty string for the *key-value* removes a previously assigned key.
- Starting with Oracle Exadata System Software release 12.2.1.1.0, you can use the optional keyword `ASMCLUSTER` to indicate that the client is an Oracle ASM cluster. The Oracle ASM cluster alias must not be longer than 15 characters, and only alphanumeric and hyphen characters are allowed.
- Starting with Oracle Exadata System Software release 12.2.1.1.0, the use of the `CELL` keyword can be used to assign a single key to all storage servers to enable cell-to-cell direct operations. You specify only a single *key-value*; you do not specify a *client_name*. You cannot use a list of values with the `CELL` keyword.
- Starting with Oracle Exadata System Software release 12.2.1.1.0, the `FOR LOCAL CELL` clause assigns a cell key to the local (current) cell. If you specify `FOR LOCAL CELL`, there can be only one key; a list of values is not supported. The *client_name* is a unique identifier for each cell.
- Starting with Oracle Exadata System Software release 12.2.1.1.0, the `FOR REMOTE CELL` clause specifies the cell keys that the current cell will accept. The *client_name* is the unique identifier for the cell assigned the *key-value*. You can specify a single client and key, or a list of values.

Usage Notes

- For ASM-scoped security or DB-scoped security, the client aliases must be entered in the `availableTo` attribute of the `GRIDDISK` object.
- When using the `ASMCLUSTER` keyword in Oracle Exadata System Software release 12.2.1.1.0 or later, if you specify a client name and key that already exists (that is a key was already specified for an Oracle ASM client prior to Oracle Exadata System Software release 12.2.1.1.0), then the client will be changed to be an Oracle ASM cluster client. In this case, the name and key will be removed from the ASM-scoped security list, and added as an Oracle ASM cluster client. Grid disks with this Oracle ASM client in their ACL can remain online for this operation.

Examples

Example 7-72 Assigning Keys to Clients

This example shows how to use the `ASSIGN KEY` command to assign keys to one or multiple 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 LOCAL CELL mykey='fa292e11b31b210c4b7a24c5f1bb4d32'

CellCLI> ASSIGN KEY FOR REMOTE CELL -
```

```
          'cellkey1'='b67d5587fe728118af47c57ab8da650a', -
          'cellkey2'='118af47c57ab8da650ab67d5587fe728'


CellCLI> ASSIGN KEY FOR CELL '4839deff903625aab394df7638e7b29a'

CellCLI> ASSIGN KEY FOR ASMCLUSTER asm1='118af47c57ab8da650ab67d5587fe728'
```

Example 7-73 Removing Keys from Clients

This example shows how to use the `ASSIGN KEY` command to remove keys from clients.

```
CellCLI> ASSIGN KEY FOR 'db1'='', 'db2'='', 'db3'='', '+asm'=''

CellCLI> ASSIGN KEY FOR ASMCLUSTER asm1=''
```

Related Topics

- Enabling Cell-to-Cell Operations
- CREATE KEY
- About Security Keys

7.7.3 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 | LUN1 [, LUN2]] [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.

Use the `LUN`*`n`* option to specify a list of LUNs by name be calibrated.

Examples

Example 7-74 Output from CALIBRATE Command with FORCE Option on Oracle Exadata Storage Server

This example shows the output when using `CALIBRATE` 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.
```

Example 7-75 Calibrating LUNs by Name

```
CALIBRATE '2_1', '2_3' FORCE
```

Related Topics

- ALTER CELL

7.7.4 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

- 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.
- CREATE CELL
- CREATE CELLDISK
- CREATE DIAGPACK
- CREATE FLASHCACHE
- CREATE FLASHLOG

- CREATE GRIDDISK
- CREATE KEY
- CREATE PMEMCACHE
- CREATE PMEMLOG
- CREATE QUARANTINE
- CREATE ROLE
- CREATE THRESHOLD
- CREATE USER
- CREATE XRMEMCACHE
- CREATE XRMEMLOG

Related Topics

- About CellCLI Object Types
  There are several Oracle Exadata System Software object types that can be used with CellCLI object commands.

7.7.4.1 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 ...]
   [, attributeName = attributeValue  ...]
```

Usage Notes

The attributes that can be set are shown as `modifiable` in Example 7-96.

- This command can be used to assign the `ASR` value to the `snmpSubscriber` attribute.

  When specifying the `snmpSubscriber` attribute, the community name cannot contain spaces or the following characters: = ' " \ / < >
- 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.
- By default, the `CREATE CELL` command creates Exadata Smart Flash Cache and the associated cell disks.

  You can specify `FLASHCACHE=0` to bypass default creation of Exadata Smart Flash Cache. A non-zero value specifies the total size for creating Exadata Smart Flash Cache. The size is divided evenly across the flash LUNs.

- By default, the `CREATE CELL` command creates Exadata Smart Flash Log and the associated cell disks.

  You can specify `FLASHLOG=0` to bypass default creation of Exadata Smart Flash Log. A non-zero value specifies the total size for creating Exadata Smart Flash Log. The size is divided evenly across the flash LUNs.
- On Exadata X8M and X9M storage server models that are equipped with Persistent Memory (PMEM), the `CREATE CELL` command automatically creates PMEM cache, PMEM log, and the associated cell disks.

  > **Note:**
  >
  > Oracle Exadata System Software release 23.1.0 introduces Exadata RDMA Memory (XRMEM). XRMEM represents the collection of Exadata software technologies that provide direct access to storage server memory using Remote Direct Memory Access (RDMA), enabling faster response times and lower read latencies. In this release, the persistent memory data accelerator, previously known as PMEM cache, is now called XRMEM cache. Likewise, the persistent memory commit accelerator, previously known as PMEM log, is now called XRMEM Log.

- Starting with Exadata X10M, on storage server models equipped to support Exadata RDMA Memory Cache (XRMEM cache), the `CREATE CELL` command automatically creates the XRMEM cache.
- The `CREATE CELL eighthRack` command enables or disables an Eighth Rack configuration on Oracle Exadata Database Machine X3-2 Quarter Racks or later. The options are `true` to enable the Eighth Rack configuration, and `false` to disable the Eighth Rack configuration. The `CREATE CELL eighthRack=true` command requires that there are no cell disks because enabling the Eighth Rack configures only half of the hard disks and flash capacity.
- Starting with Oracle Exadata System Software release 19.1.0, the `httpsAccess` attribute can be used to specify a list of IP addresses or IP subnet masks that control who can access the RESTful service via HTTPs. The value you specify for `httpsAccess` overwrites any previous value. You can use the following values for `httpsAccess`:
  - `ALL` — to allow access to all hosts (Default)
  - `NONE` — to disable the HTTPs port completely
  - `IP1, IP2,..., IP`*`n`* — to only allow access to hosts with IP addresses IP1, IP2,..., IP*n* where IP*n* is a valid IP address in IPv4, IPv4 subnet, IPv6 or IPv4-embedded IPv6 format. You can specify a maximum of 512 IP addresses for the access control list.

  Additionally, instead of a single IP address, you can use the `/` character to specify a range of IP addresses using a subnet mask. For example the range `'192.168.10.0/24'` corresponds to hosts having IP addresses from 192.168.10.1 to 192.168.10.255. If you specify an IP address range, you need to enclose the IP address string in quotes.

Examples

Example 7-76 Creating a Cell

This example shows how to create a cell. In the example, the interconnections are set to an existing InfiniBand Network Fabric connection.

```
CellCLI> CREATE CELL cell22 interconnect1=bondib0
```

Example 7-77 Creating an Eighth Rack Configuration

This example shows how to create a cell in an Eighth Rack configuration.

```
CellCLI> CREATE CELL eighthRack=true
```

Example 7-78 Creating a Cell with Restricted HTTPs Access

This example shows how to create a cell that allows HTTPs port access only from hosts having IP addresses in the range 192.168.10.1 to 192.168.10.255.

```
CellCLI> CREATE CELL httpsAccess='192.168.10.0/24'
```

Related Topics

- Restrictions on Values of Common Attributes
  Review the following restrictions for the values of attributes used by multiple CellCLI objects.
- *Oracle Exadata Database Machine Maintenance Guide*

7.7.4.2 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 on multiple devices.

Syntax

```
CREATE CELLDISK  {
   ALL [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] 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`, `size`, 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.
- You can use the `size` attribute when adding a new disk that is a different size than the existing disks. You must specify a value for `size` that is less than or equal to the maximum allowed cell disk size.
- `CREATE CELLDISK ALL` is a shortcut command to create all candidate cell disks for the cell. This operation occurs in two steps:
  - LUNs are configured for all physical disks that are not already configured. 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.
  - You can include `size` to create all cell disks with the specified size, instead of attempting to use the entire disk. If you do not specify `size` with the `CREATE CELLDISK ALL` command, then:
    - On disks without a partition containing the Exadata System Software, Management Server (MS) creates the cell disks with a size that is equal to the smallest physical disk size in the cell (which is also the maximum cell disk size allowed in the cell).
    - On disks containing a system partition, MS creates the cell disk with a size that is equal to the maximum allowed cell disk size minus the size of the system partition.
- The `FLASHDISK` option limits the `CREATE CELLDISK` command to cell disks that are flash disks.

  Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to create cell disks only on the specified flash media type.
- 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 has been deprecated. Starting with Oracle Exadata System Software release 19.1.0, attempts to create interleaving grid disks will be automatically converted to normal grid disk creation. Interleaving grid disks created in earlier Oracle Exadata System Software releases will continue to operate normally.

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 cell disk metadata partition. On a subsequent restart, Cell Server (CELLSRV) 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.

Example 7-79 Creating a Cell Disk

```
CellCLI> CREATE CELLDISK ALL

CellCLI> CREATE CELLDISK cdisk03 lun=0_3

CellCLI> CREATE CELLDISK cdisk04 physicalDisk='I2:1:2'
CellCLI> CREATE CELLDISK CD_08_cell06 lun=0_8, size=300M

CellCLI> CREATE CELLDISK CD_03_cell04 lun=0_3

CellCLI> CREATE CELLDISK CD_05_cell09 physicalDisk='2I:1:2'
```

Related Topics

- ALTER CELL
- Restrictions on Values of Common Attributes
  Review the following restrictions for the values of attributes used by multiple CellCLI objects.

7.7.4.3 CREATE DIAGPACK

Purpose

The `CREATE DIAGPACK` command creates a diagnostic package, which contains logs and traces that you can use to troubleshoot problems in your system. You can also send the generated package to Oracle Support Services, as needed.

Syntax

```
CREATE DIAGPACK packStartTime=time, [durationInHrs=duration]
```

or

```
CREATE DIAGPACK alertName=alertName
```

Usage Notes

When an alert occurs, a diagnostic package is created automatically. This package contains logs and traces related to the alert.

The `CREATE DIAGPACK` command enables you to generate diagnostic packages manually.

- The `packStartTime` parameter specifies when to start collecting the logs and traces. The format of `packStartTime` is *`yyyy_MM_ddTHH_mm_ss`*, For example: `2015_07_07T09_00_00`.

  You can also specify the keyword `now`for `packStartTime`. The `packStartTime` cannot be in the future and cannot be older than 7 days. The value of `packStartTime` is used as part of the name of the diagnostic package.
- The `durationInHrs` parameter specifies the number of hours of logs and traces to include in the diagnostic package. Valid values are from `1` (default) to `6`.

  Every diagnostic package includes logs 1 hour before and 1 hour after the `packStartTime`. For example, if you specify a time of 12_00_00, then logs will collected from 11_00_00 to 13_00_00, unless the end time is in the future.
- The `alertName` parameter specifies the alert name for which to create the diagnostic package. You can run the LIST ALERTHISTORY command to view the alert names.

Name of Diagnostic Packages

The name of the diagnostic package is formed as: *`hostname`*`_diag_`*`packStartTime`*`_`*`unique package ID`*

For example: `testcell1_diag_2015_07_07T09_00_00_3`

For alerts, the name of the diagnostic package is formed as: *`hostname_timestamp of when the package was created_alert ID`*. For example: `testcell1_2015_09_30T13_13_00_2_1`

Location of Diagnostic Packages

The location of the diagnostic packages is `$LOG_HOME`.

Status of Diagnostic Packages

You can run the LIST DIAGPACK command to get a list of diagnostic packages in your system, and their status.

Privileges Needed to Create, List, and Download Diagnostic Packages

There are certain privileges that are needed for working with diagnostic packages. Use CellCLI to grant the following privileges to a role:

- Privilege to create diagnostic packages:

  ```
  grant privilege CREATE ON DIAGPACK to ROLE role
  ```
- Privilege to list diagnostic packages and check their status:

  ```
  grant privilege LIST ON DIAGPACK to ROLE role
  ```
- Privilege to download diagnostic packages:

  ```
  grant privilege DOWNLOAD ON DIAGPACK to ROLE role
  ```

You can then grant the role to users. For example, if you named your role `diagpack_role`, the following command grants the role to `fred`.

```
CellCLI> GRANT ROLE diagpack_role TO USER fred
```

During deployment, Oracle Exadata Deployment Assistant (OEDA) creates an Exadata storage software user called `CELLDIAG`. You can use this user to connect to a cell remotely using ExaCLI or REST API. This user has privileges to create, list, and download diagnostic packages.

Downloading Diagnostic Packages

You can download diagnostic packages using any of the following methods. Note that you need the `DOWNLOAD ON DIAGPACK` privilege before you can download diagnostic packages.

- Using the REST API
  - To download the diagnostic package by name, use the following URL, where *hostname* specifies the host name of the cell and *diagpackname* specifies the name of the diagnostic package:

    ```
    https://hostname/diagpack/download?name=diagpackname
    ```

    If the user is not already logged in, the URL will prompt for a user name and password.

    Diagnostic packages can also be accessed at `https://hostname/diagpack` . For example: `https://cell1.example.com/diagpack`.

    The page then prompts the user to log in:

    ```
    User: fred
    Password: ********
    ```

    Based on the user's privileges, various sections of this page could be hidden:

    * The form to create a new diagpack will not be shown if the user does not have the `CREATE ON DIAGPACK` privilege.
    * Similarly, the list of alerts and their diagnostic packages will not be shown if the user does not have the `LIST ON DIAGPACK` privilege.
  - To download the diagnostic package by alert name, use the following URL, where *hostname* specifies the host name of the cell and *alertName* specifies the alert name of the diagnostic package:

    ```
    https://hostname/diagpack/download?alert=alertName
    ```

    The alert name is the same alert name that is used in AlertHistory. It looks like 1, 2, 3 for stateless alerts, and 1_1, 2_1, 3_1, 3_2 for stateful alerts.
- Using the `download` ExaCLI command

  ExaCLI enables you to run CellCLI commands on storage nodes remotely from compute nodes.

  To run the download command, run the following commands on a compute node:

  1. Start up ExaCLI and connect to the cell containing the diagnostic pack. For example, use a command similar to the following where *hostname* specifies the host name of the cell:

     ```
     exacli -l celladministrator -c hostname
     Password=********
     ```

2. Run the download command using a command similar to the following where *name* specifies the name of the diagnostic package to download and *destinationFolder* specifies the directory where you want to save the downloaded diagnostic package:

   ```
   exacli> download diagpack name destinationFolder
   ```

- Getting the diagnostic package from the alert emails

  The alert emails include diagnostic packages for all alerts except INFO, CLEAR, and WARNING. Diagnostic packages are generated for critical alerts only.

Re-triggering Package Creation from the Web Page

You can use the following URL to re-trigger package creation:

`https://hostname/diagpack`

If the diagnostic package for an alert does not exist on disk, then the web page shows a **Create Package** link instead of a **Download** link.

Click the **Create Package** link to add the alert to the list for creating a diagnostic package. Once the diagnostic package has been created, and the web page is refreshed, the page will display a **Download** link that you can use to download the newly created diagnostic package.

Turning Off the Diagnostic Pack Attachment in Emails

To turn off the diagnostic pack attachment in emails, run `ALTER CELL diagPackEmailAttach=FALSE`. The diagnostic packs are still generated and stored on the storage servers. To download the diagnostic packs, see "Downloading Diagnostic Packages".

Examples

Example 7-80 Using "now" for packStartTime

This example creates a diagnostic package using `now` as the start time and the default duration of 1 hour.

The output is 1 compressed file under `$LOG_HOME`.

```
CellCLI> CREATE DIAGPACK packStartTime="now"
    Processing: scab01cel11_diag_2015_07_08T17_53_58_1
    Use 'list diagPack' to check its status.
```

Example 7-81 Specifying a duration

This example creates 3 diagnostic packages under `$LOG_HOME`:

The first package has a start time of `2015_07_07T09_00_00`.

The second package has a start time of `2015_07_07T10_00_00`.

The third package has a start time of `2015_07_07T11_00_00`.

```
CellCLI> CREATE DIAGPACK packStartTime="2015_07_07T09_00_00", durationInHrs=3
    Processing: scab01cel11_diag_2015_07_07T09_00_00_1
    scab01cel11_diag_2015_07_07T10_00_00_1 (In queue...)
    scab01cel11_diag_2015_07_07T11_00_00_1 (In queue...)
    Use 'list diagPack' to check its status.
```

Related Topics

- About Oracle Auto Service Request
- About Automatic Diagnostic Repository
- Using the ExaCLI Utility

7.7.4.4 CREATE FLASHCACHE

Purpose

The `CREATE FLASHCACHE` command creates Exadata Smart Flash Cache on a cell for I/O requests.

Syntax

```
CREATE FLASHCACHE { ALL [size=fc_size] | CELLDISK="cdisk1 [,cdisk2] ..." [,
size=fc_size] }
```

Usage Notes

Cell disks defined on Exadata Smart Flash Cache cannot be exported.

The `ALL` argument creates Exadata Smart Flash Cache on all flash cell disks. If the `ALL` argument is not specified, then the `CELLDISK` argument must be specified.

Using the `CELLDISK` argument, you can specify a list of flash cell disks to be used for flash cache. The names of the flash cell disks are comma-delimited. The `FLASHDISK` argument is not required.

The `size` argument specifies the total space used for flash cache. Similar to space in grid disks and flash logs, flash cache space is allocated in 16 MB units, referred to as allocation units. If the `size` attribute is specified when creating a flash cache, 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 flash cache 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 flash cell disk with the `LIST FLASHCACHE` command. For example, `LIST FLASHCACHE ATTRIBUTES freespace`. If the `size` attribute is not specified, then the maximum size is allocated.

If the `size` attribute is not specified, then all available space on each cell disk in the list is used for Exadata Smart Flash Cache.

By default, 5 percent of space on Extreme Flash storage servers is used for write-back flash cache.

Examples

Example 7-82 Creating Exadata Smart Flash Cache

This example shows how to create Exadata Smart Flash Cache on a cell.

```
CellCLI> CREATE FLASHCACHE ALL

CellCLI> CREATE FLASHCACHE ALL SIZE=250g

CellCLI> CREATE FLASHCACHE CELLDISK='fd_01,fd_02,fd_03,fd_04'

CellCLI> CREATE FLASHCACHE CELLDISK='fd_01_mycell,fd_02_mycell', size = 64G
```

Related Topics

- ALTER FLASHCACHE

7.7.4.5 CREATE FLASHLOG

Purpose

The `CREATE FLASHLOG` command creates the Oracle Exadata Smart Flash Log on a cell for redo log write requests.

Syntax

```
CREATE FLASHLOG { ALL [ FLASHDISK ] [ size=log_size ] |
          CELLDISK='cdisk1[,cdisk2]...' [ , size=log_size ] }
```

Usage Notes

The `CREATE FLASHLOG` 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 will total the specified size. If a size is not specified, then a default size of 512 MB is used.

The size of Oracle Exadata Smart Flash Log space on each flash disk must be less than 4 GB. If all 16 flash disks are available, then the total size of Oracle Exadata Smart Flash Log must be less than 64 GB.

Similar to space in grid disks and flash cache, flash log space is allocated in 16 MB units, referred to as allocation units. If the size attribute is specified when creating a flash log, 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 flash log is 16 MB. Any size value less than 16 MB is rounded up to 16 MB.

The `ALL FLASHDISK` argument creates Oracle Exadata Smart Flash Log on all flash cell disks. If the `ALL` argument is not specified, then the `CELLDISK` argument must be specified. The `FLASHDISK` argument is not required.

> **Note:**
>
> The `CREATE FLASHCACHE` command, by default, uses all available space on each flash disk. Therefore, use the `CREATE FLASHLOG` command before creating the flash cache to ensure both objects consume the correct amount of flash disk space.

To change the size of the flash log, use the `DROP FLASHLOG` command to drop the flash log, and then use the `CREATE FLASHLOG` command to create a flash log with the new size.

Examples

The following example shows how to create Oracle Exadata Smart Flash Log on a cell.

Example 7-83 Creating Oracle Exadata Smart Flash Log

```
CellCLI> CREATE FLASHLOG ALL

CellCLI> CREATE FLASHLOG ALL SIZE=1g

CellCLI> CREATE FLASHLOG ALL FLASHDISK

CellCLI> CREATE FLASHLOG CELLDISK='fd1,fd2,fd3,fd4'
```

7.7.4.6 CREATE GRIDDISK

Purpose

The `CREATE GRIDDISK` command creates a grid disk object on a specified cell disk or creates one or more grid disks on each cell disk in the storage server. The command also assigns initial attributes to each new grid disk.

Syntax

```
CREATE GRIDDISK
   { ALL [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] FLASHDISK | HARDDISK ]
PREFIX={[']gdisk_name_prefix[']|'gdisk_name_prefix1[,gdisk_name_prefix2]...'}

[,multi_attribute_name={[']attribute_value[']|'attribute_value1[,attribute_val
ue2]...'} ]...
   | gdisk_name CELLDISK=cell_disk_name }
   [,attribute_name=[']attribute_value[']]...
```

Usage Notes

- If an individual grid disk name (*gdisk_name*) is specified, then the grid disk is created on the cell disk specified by the `CELLDISK` argument. You must ensure that the grid disk name is unique across all storage servers. 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.
- Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to create grid disks on cell disks associated with the specified flash media type.

If you do not fully specify the flash media type, `ALL FLASHDISK` is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK` on Extreme Flash (EF) storage servers containing capacity-optimized flash devices. On all other storage servers, `ALL FLASHDISK` is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.

- If the `ALL` option is specified with a media type qualifier (`FLASHDISK` or `HARDDISK`), the command only creates grid disks on cell disks associated with the specified media type.

  Starting with Oracle Exadata System Software release 24.1.0, if you specify the `ALL` option without a media type qualifier (`FLASHDISK` or `HARDDISK`), then the command creates grid disks on the cell disks associated with the primary storage media on the storage server. This behavior ensures that grid disks created using the `ALL` option use the same media type, making it much less likely that you can mistakenly attempt to configure an Oracle ASM disk group with a mixture of storage media. Specifically:

  – On Extreme Flash (EF) storage servers containing capacity-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK`.

  – On Extreme Flash (EF) storage servers containing only performance-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.

  – On all storage servers containing hard disk drives (HDDs), `ALL` with no media type qualifier is equivalent to `ALL HARDDISK`.

  Before Oracle Exadata System Software release 24.1.0, if the `ALL` option is specified without a media type qualifier, the command creates grid disks on all available cell disks regardless of media type.

- The `PREFIX` option must be specified when `ALL` is used.

  The `PREFIX` option specifies one or more comma-separated prefix strings. Each prefix is used to create a grid disk on the underlying cell disk. The generated grid disk names are composed of the grid disk prefix followed by an underscore (_) and then the cell disk name.

  For example, `CREATE GRIDDISK ALL PREFIX=data01` creates one grid disk on each available cell disk. In this case, if the cell disks are named `CD_01_cell01`, `CD_02_cell01`, `CD_03_cell01`, and so on, then the corresponding grid disk names are `data01_CD_01_cell01`, `data01_CD_02_cell01`, `data01_CD_03_cell01`, and so on.

  Use prefixes that match the Oracle ASM disk groups that consume the grid disks. This helps you to identify where the grid disks are being used. Also, use short prefix values to ensure that the generated grid disk names stay within the 30-character limit.

- `multi_attribute_name` specifies one of the following attributes, which may accept a comma-separated list of values:

  – `size`

  – `offset`

  – `virtualSize`

  – `comment`

  If the command specifies multiple comma-separated `PREFIX` values, then for each `multi_attribute_name` you must specify only one value or the same number of values as in the prefix list. If one value is specified, it applies to all values in the prefix list. Otherwise, each entry in the list of attribute values applies to the corresponding value in the prefix list.

  For example, `CREATE GRIDDISK ALL PREFIX='data01,data02', size='500G,300G'` creates two grid disks on each available cell disk. In this case, `size=500G` is associated with `PREFIX=data01`, and `size=300G` is associated with `PREFIX=data02`.

All other attribute settings require only one value at all times.

- The `size` attribute determines the amount of storage space that is allocated to the grid disk. If the `size` attribute is not specified, then the grid disk consumes all of the available space on the cell disk.

  Grid disk space is allocated in 16 MB units, referred to as allocation units. If the `size` attribute is specified, then the allocated size is the largest multiple of allocation units that is 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 first determine the available free space on each target cell disk by using the `LIST CELLDISK` command. For example, `LIST CELLDISK` *`cdisk`* `ATTRIBUTES freespace`.

- The `offset` attribute determines the position on the hard disk where the grid disk is allocated. The outermost tracks have lowest offset values. If the `offset` attribute is not specified, then the lowest available offset is automatically chosen in order of grid disk creation.

> **Note:**
>
> The `offset` attribute does not apply to flash storage.

- The `size` and `offset` attributes are specified as a number of bytes, unless the suffix `M` (megabytes), `G` (gigabytes), or `T` (terabytes) is included with the number, such as `size=300M`, or `size=150G`.
- The `CREATE GRIDDISK ALL ...` command skips disks which do not have enough free space for a minimum sized grid disk (16 MB). In this case, the skipped grid disks are identified in the command output and the command continues.
- The value of the `availableTo` attribute is set to the names of the clients registered for DB-scoped security. The value for each client name is the database 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 access the grid disk.
- Do not edit the value of `idp.type` or `idp.boundary`. Oracle Exadata System Software passes a hint to Oracle ASM about the type of interleaved grid disk, either normal redundancy or high redundancy. Oracle ASM sets the default value for `idp.type` to `static` and `idp.boundary` to the type of redundancy used in the underlying grid disks. The default value of the `idp.type` attribute is `static` for Oracle Exadata Storage Server disk groups created on interleaved grid disks.

> **Note:**
>
> Interleaved grid disks are deprecated in Oracle Exadata System Software release 19.1.0.

- The cachingPolicy attribute can be set to default or none. The default option allows the data in the grid disk to be cached in the flash cache. Flash cache is not used for data stored on grid disks with cachingPolicy=none.

  Oracle Exadata Deployment Assistant (OEDA) configures the grid disks supporting the RECO Oracle ASM disk group to use cachingPolicy=none. Consequently, flash cache is not used for any data files placed in the RECO disk group.
- The virtualSize attribute is used to create grid disks used in an Oracle ASM SPARSE disk group. The maximum virtual size for a sparse grid disk is approximately 100 TB. Sparse grid disks are available for Oracle Exadata Database Machine X3-2 and later.

Note:

The Oracle Database and Oracle Grid Infrastructure software must be release 12.1.0.2.0 BP5 or later when using sparse grid disks.

Example 7-84 Creating Grid Disks

This examples shows how to create grid disks.

```
CellCLI> CREATE GRIDDISK ALL HARDDISK PREFIX='data01,data02', size='500G,300G'

CellCLI> CREATE GRIDDISK ALL HARDDISK PREFIX=data03

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=data10, availableTo='+asm,db1,db2'

CellCLI> CREATE GRIDDISK hr7_CD_07_cell01 CELLDISK=CD_07_cell01,
availableTo='asm_hr,hrdb0'

CellCLI> CREATE GRIDDISK GD123 CELLDISK=RECO_CD123, size=100G,
cachingPolicy=none
```

Example 7-85 Creating a SPARSE Grid Disk

```
CellCLI> CREATE GRIDDISK spar01 celldisk=CD_01_cel01, size=10G,
virtualsize=100G
```

7.7.4.7 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.

Example 7-86 Creating a Key

This example shows the `CREATE` command with the `KEY` object.

```
CellCLI> CREATE KEY
         3452c64fec9a5800bbe48d4093269400
```

Related Topics

- About Security Keys
- ASSIGN KEY

7.7.4.8 CREATE PMEMCACHE

Purpose

The `CREATE PMEMCACHE` command creates a persistent memory (PMEM) cache for I/O requests.

Note:

The `CREATE PMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
CREATE PMEMCACHE { ALL [ size = cache_size ] |
          CELLDISK='cdisk1[,cdisk2]...' [ , size=cache_size ] }
```

Usage Notes

- The `ALL` argument creates the PMEM cache on all PMEM cell disks. If the `ALL` argument is not specified, then specific PMEM cell disks must be specified by using the `CELLDISK` argument. The PMEM cache is spread as evenly as possible across the PMEM cell disks.
- PMEM cache space is allocated in 16 MB chunks, referred to as allocation units, with a minimum of 1 allocation unit. Consequently, when the `size` argument is specified, the actual space allocated to the PMEM cache is at least 16 MB or the largest multiple of 16

MB that is less than or equal to the specified size. For example, if `size=500M` is specified, then 496 MB is allocated using 31 allocation units.

- Before specifying the `size` attribute, use the following command to check the available free space on the PMEM cell disks:

  ```
  LIST CELLDISK ATTRIBUTES freespace WHERE disktype=PMEM
  ```

- If the `size` attribute is not specified, then the PMEM cache consumes all of the available space (allocation units).
- Cell disks used by the PMEM cache cannot be exported.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `CREATE XRMEMCACHE` instead of `CREATE PMEMCACHE`.

Examples

Example 7-87   Creating PMEM Cache

The following command demonstrates how to create a 64 GB PMEM cache on specific PMEM cell disks.

```
CREATE PMEMCACHE celldisk='PM_01_mycell,PM_02_mycell', size = 64G
```

The following command demonstrates how to create a 64 GB PMEM cache using all PMEM cell disks.

```
CREATE PMEMCACHE ALL size = 64G
```

7.7.4.9 CREATE PMEMLOG

Purpose

The `CREATE PMEMLOG` command creates the persistent memory (PMEM) log on a cell for redo log write requests.

Note:

The `CREATE PMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
CREATE PMEMLOG { ALL [ size=log_size ] |
          CELLDISK='cdisk1[,cdisk2]...' [ , size=log_size ] }
```

Usage Notes

- The `ALL` argument creates the PMEM log on all PMEM cell disks. If the `ALL` argument is not specified, then specific PMEM cell disks must be specified by using the `CELLDISK` argument. The PMEM log is spread as evenly as possible across the PMEM cell disks.

- PMEM log space is allocated in 16 MB chunks, referred to as allocation units, with a minimum of 1 allocation unit. Consequently, when the `size` argument is specified, the actual space allocated to the PMEM log is at least 16 MB or the largest multiple of 16 MB that is less than or equal to the specified size. For example, if `size=300M` is specified, then 288 MB is allocated using 18 allocation units.
- If a size is not specified, then a default size is used. Commencing with Oracle Exadata System Software version 20.1.0, the default size is 10176 MB (9.9375 GB). Previously, the default size was 960 MB.
- To change the size of the PMEM log, use the `DROP PMEMLOG` command to drop the PMEM log, and then use the `CREATE PMEMLOG` command to create a PMEM log with the new size.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `CREATE XRMEMLOG` instead of `CREATE PMEMLOG`.

Note:

By default, the `CREATE PMEMCACHE` command uses all available space on each PMEM cell disk. Therefore, it is recommended to create the PMEM log before the PMEM Cache to ensure that both objects are accommodated.

Examples

The following example shows how to create the PMEM log on a cell.

Example 7-88 Creating the PMEM Log

To create the PMEM log using the default size spread across all available PMEM cell disks, use the following command:

```
CellCLI> CREATE PMEMLOG ALL
```

To create the PMEM log with a size of 1 GB spread across all available PMEM cell disks, use the following command:

```
CellCLI> CREATE PMEMLOG ALL SIZE=1g
```

To create the PMEM log using the default size spread across specific PMEM cell disks, use the following command:

```
CellCLI> CREATE PMEMLOG
CELLDISK='PM_01_mycell,PM_02_mycell,PM_03_mycell,PM_04_mycell'
```

To create the PMEM log with a size of 1 GB spread across specific PMEM cell disks, use the following command:

```
CellCLI> CREATE PMEMLOG
CELLDISK='PM_01_mycell,PM_02_mycell,PM_03_mycell,PM_04_mycell', size=1G
```

7.7.4.10 CREATE QUARANTINE

Purpose

The `CREATE QUARANTINE` command allows a quarantine to be created manually.

Syntax

```
CREATE QUARANTINE quarantineType=value quarantinePlan="SYSTEM",
                  dbUniqueName=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 that are known to cause problems.

- `quarantineType` specifies the type of quarantine to be created, such as SQLID and SQL_PLAN.
- `quarantinePlan` must be set to `SYSTEM`. Oracle Support Services may specify other values.
- `dbUniqueName` specifies the name of the database that has the quarantine.

Example 7-89 Creating a Quarantine

This example shows the `CREATE` command with the `QUARANTINE` object.

```
CELLCLI> CREATE QUARANTINE  DBUG comment='For debugging quarantines"

CellCLI> CREATE QUARANTINE quarantineType="SQLID", quarantinePlan="SYSTEM", -
         dbUniqueName="DB1", sqlid="5xnjp4cutc1s7";
```

Related Topics

- DESCRIBE QUARANTINE

7.7.4.11 CREATE ROLE

Purpose

The `CREATE ROLE` command creates a role for a user accessing a cell.

Syntax

```
CREATE ROLE  role_name1 [, role_name2, ...]
```

Usage Notes

The role name should be unique.

Example 7-90 Creating a Role

This example shows how to create a role named `gd_monitor`.

```
CellCLI>CREATE ROLE gd_monitor
```

7.7.4.12 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`. Use the `LIST METRICCURRENT` *`metric`* command to display the available object name for metric. The object name is optional.
- When a object name is not specified, then the threshold is applied to all metric objects for the given metric.
- 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.
- The `GD_SP_PRCT_ALLOCATED` metric has a built-in threshold, and automatically sends alerts. Create thresholds for other metrics to receive alerts for those metrics.

When specifying occurrences and observations, you need the specified number of consecutive occurrences of sample averages over the number of observations to cause an alert. For example, if the following five observations (`observations=5`) happen on a cell, then the average sample would be 10 because the number of consecutive occurrences (`occurrences=2`) had values of 5 and 15.

```
Observation 1: 0
Observation 2: 30
Observation 3: 0
Observation 4: 5
Observation 5: 15
```

Example 7-91 Creating a Threshold

This example shows how to create a threshold.

```
CellCLI> CREATE THRESHOLD db_io_rq_sm_sec.db123 comparison='>', critical=120
CellCLI> CREATE THRESHOLD ct_io_wt_sm.interactive warning=10, critical=20, -
         comparison='=', occurrences=2, observation=5
```

Related Topics

- Exadata Alerts
  Alerts draw attention to potential and actual problems and other events of interest to an administrator.

7.7.4.13 CREATE USER

Purpose

The `CREATE USER` command creates a user.

Syntax

```
CREATE USER name PASSWORD = *
```

Usage Notes

- The user name should be unique.
- `celladmin`, `cellmonitor`, and `root` are reserved user names that cannot be used with the CREATE USER command.
- The system prompts for a password for the new user. The password must have 12 to 40 alphanumeric characters or special characters !@#$%^&*() with at least one digit, one lowercase letter, and one uppercase letter. Starting with Oracle Exadata System Software release 18.1.0.0.0, the password can be 8 to 40 characters in length and can also utilize the special characters - and _.
- The new password cannot be the same as the current password for the user.

Example 7-92 Creating a User

This example shows how to create a user.

```
CellCLI> CREATE USER agarcia PASSWORD = *
password:
Confirm password: password
User agarcia successfully created.
```

7.7.4.14 CREATE XRMEMCACHE

Purpose

Starting with Exadata X10M, the `CREATE XRMEMCACHE` command creates the Exadata RDMA Memory Cache (XRMEM cache).

Note:

On Exadata X8M and X9M systems, the `CREATE XRMEMCACHE` command is equivalent to the `CREATE PMEMCACHE` command.

Syntax

```
CREATE XRMEMCACHE
```

Usage Notes

Starting with Exadata X10M, the XRMEM cache is created automatically during cell creation on supported storage servers. The XRMEM cache is also re-created automatically during Cell Server (CELLSRV) startup if it does not exist.

7.7.4.15 CREATE XRMEMLOG

Purpose

Starting with Oracle Exadata System Software release 23.1.0, the `CREATE XRMEMLOG` command is equivalent to the `CREATE PMEMLOG` command.

> **Note:**
>
> The `CREATE XRMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

7.7.5 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 object types supported by CellCLI.
- The list of attributes can be used as arguments in the `LIST` command.
- `DESCRIBE` does not display all of the attributes for the objects.

- DESCRIBE ACTIVEREQUEST
- DESCRIBE ALERTDEFINITION
- DESCRIBE ALERTHISTORY
- DESCRIBE CELL
- DESCRIBE CELLDISK
- DESCRIBE DATABASE
- DESCRIBE DISKMAP
- DESCRIBE FLASHCACHE

- DESCRIBE FLASHCACHECONTENT
- DESCRIBE FLASHLOG
- DESCRIBE GRIDDISK
- DESCRIBE IBPORT
- DESCRIBE IORMPLAN
- DESCRIBE KEY
- DESCRIBE LUN
- DESCRIBE METRICCURRENT
- DESCRIBE METRICDEFINITION
- DESCRIBE METRICHISTORY
- DESCRIBE OFFLOADGROUP
- DESCRIBE PHYSICALDISK
- DESCRIBE PLUGGABLEDATABASE
- DESCRIBE PMEMCACHE
- DESCRIBE PMEMLOG
- DESCRIBE QUARANTINE
- DESCRIBE ROLE
- DESCRIBE SOFTWAREHISTORY
- DESCRIBE SOFTWAREUPDATE
- DESCRIBE THRESHOLD
- DESCRIBE USER
- DESCRIBE XRMEMCACHE
- DESCRIBE XRMEMCACHECONTENT
- DESCRIBE XRMEMLOG

7.7.5.1 DESCRIBE ACTIVEREQUEST

Purpose

The `DESCRIBE ACTIVEREQUEST` command displays a list of attributes for the `ACTIVEREQUEST` object type.

Syntax

```
DESCRIBE ACTIVEREQUEST
```

Usage Notes

The attributes for the `DESCRIBE ACTIVEREQUEST` command can include the following:

- `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
- `pdbID`: Identifier of the pluggable database
- `requestState`: State of the active request, such as
  - 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

Example 7-93 Describing the ACTIVEREQUEST Object

This example shows the `DESCRIBE` command with 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
         pdbID
         requestState
         sessionID
         sessionSerNumber
         sqlID
         tableSpaceNumber
```

7.7.5.2 DESCRIBE ALERTDEFINITION

Purpose

The `DESCRIBE ALERTDEFINITION` command displays a list of attributes for the `ALERTDEFINITION` object type.

Syntax

```
DESCRIBE ALERTDEFINITION
```

Usage Notes

The attributes for the `DESCRIBE ALERTDEFINITION` command can include the following:

- `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

Example 7-94   Describing the ALERTDEFINITION Object

```
CellCLI> DESCRIBE ALERTDEFINITION

         name
         alertShortName
         alertSource
         alertType
         description
         metricName
```

Related Topics

- Displaying Alert Definitions
  Use the `LIST ALERTDEFINITION` command to display the alert definitions for the storage server.

7.7.5.3 DESCRIBE ALERTHISTORY

Purpose

The `DESCRIBE ALERTHISTORY` command displays a list of attributes for the `ALERTHISTORY` object type.

Syntax

```
DESCRIBE ALERTHISTORY
```

Usage Notes

The attributes for the `DESCRIBE ALERTHISTORY` command can include the following:

- `alertAction`: Recommended action to perform for this alert
- `alertDescription`: Description for the 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, or "non-deliverable": Never tried
  - 1, or "sent": Sent successfully
  - 2, or "attempting delivery": Retrying, up to 5 times
  - 3, or "delivery failed": Retry failed 5 times
  - 4, or "creating diagpack": The diagnostic packaging and the outgoing alert email are still pending for this alert. It is used by MS to keep track of the alerts that have not been processed or are being processed. This enables MS, in the event of a restart, to continue processing alerts with state 4 and send the alert email.
- `sequenceBeginTime`: Time stamp when an alert sequence ID is first created
- `serviceRequestLink`: The URL to the service request associated with the alert
- `serviceRequestNumber`: The service request number associated with the alert
- `severity`: Severity level. Values are:
  - `clear`
  - `info`
  - `warning`
  - `critical`

Example 7-95 Describing the ALERTHISTORY Object

```
CellCLI> DESCRIBE ALERTHISTORY

         name
         alertAction
         alertDescription
         alertMessage
         alertSequenceID
         alertShortName
         alertType
         beginTime
         endTime
         examinedBy                modifiable
         failedMail
         failedSNMP
         metricObjectName
```

```
metricValue
notificationState
sequenceBeginTime
serviceRequestLink
serviceRequestNumber
severity
```

Related Topics

- DESCRIBE METRICDEFINITION

7.7.5.4 DESCRIBE CELL

Purpose

The `DESCRIBE CELL` command displays a list of attributes for the `CELL` object type.

Syntax

```
DESCRIBE CELL
```

Usage Notes

The following list contains the attributes for the `DESCRIBE CELL` command.

- `accessLevelPerm`: Specifies the access level at which the cell runs by default. Value is either `remoteLoginEnabled` or `remoteLoginDisabled`.
- `accessLevelTemp`: Duration of time in which the access level is temporarily changed from the setting of `accessLevelPerm`
- `accountLockInDays`: Number of days after a password expires before a user account is locked. Available with Oracle Exadata System Software release 19.1.0 or later.
- `bbuStatus`: Status of hard disk controller battery-backed unit (BBU)
- `cellsrvStatus`: Status of Cell Server
- `cellVersion`: Release number of the cell software
- `columnarCachePersMode`: Controls the persistent columnar cache feature
- `comment`: User-supplied text string
- `cpuCount`: Number of CPUs on the cell
- `dbPerfDataSuppress`: Specifies which databases should not have their statistics reported in Automatic Workload Repository (AWR) reports
- `diagHistoryDays`: Number of days ADR files are retained. The default is 7 days.
- `diagPackEmailAttach`: Whether a diagpack is included as an attachment in the alert email or not. The default is `true`.
- `diagPackUploadEnabled`: Whether the auto diagpack upload feature is enabled or not. The default is `true`.
- `doNotServiceLEDStatus`: Status of the cell DoNotService LED. The value can be `on` or `off`.

- `eighthRack`: Specifies whether Oracle Exadata Database Machine Eighth Rack configuration for storage cells is enabled or disabled
- `emailFormat`: File format for email messages. The value can be `html` or `text`.
- `emailSubscriber`: List of names that subscribe to the alert notifications
- `events`: String for *events*++ that is passed to Cell Server for debugging and trace information purposes
- `exacliEnabled`: Specifies whether `exacli` is enabled or disabled. The default value is true (enabled).
- `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`.
- `flashCacheMode`: Setting for flash cache. The value can be `writethrough` or `writeback`. The default is `writethrough`.
- `httpsAccess`: Control list of IP addresses for HTTPs port access to the Exadata RESTful Service. Available with Oracle Exadata System Software release 19.1.0 or later.
- `id`: Global unique identifier (GUID) supplied by the hardware vendor
- `interconnect[1-8]`: Interconnect1 to interconnect8 for the cell. For example: `bondeth0` or `bondib0`.
- `interconnectCount`: Number of active InfiniBand network interconnects
- `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.
- `IOTimeoutThreshold`: Specifies the timeout threshold. If cell I/O takes longer than the defined threshold, then the I/O is canceled, and Oracle ASM redirects the I/O to another mirror copy of the data.
- `ipaddress[1-8]`: `ipaddress1` to `ipaddress8` for the cell
- `kernelVersion`: Version of the host kernel software
- `location`: Physical location of the cell hardware supplied by the user
- `locatorLEDStatus`: Status of cell LOCATE LED. The value can be `on` or `off`.
- `mailServer`: Fully qualified domain name of the email relay server used to send alert notifications
- `makeModel`: Make and model of the cell hardware supplied by the vendor
- `memoryGB`: The memory in gigabytes for the cell
- `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 regular metric history 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`, `none`, or both `mail` and `snmp`.

- `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`.
- `offloadGroupEvents`: Used only under the guidance of Oracle Support
- `pmemCacheMode`: PMEM cache mode.

  The default is `writethrough`. Commencing with Oracle Exadata System Software release 23.1.0, `writethrough` is the only supported mode for PMEM cache. Previously, `writeback` mode was available but was not generally recommended.
- `powerCount`: Count of power supplies, displayed as working/total
- `powerStatus`: Status of the power. The value can be `normal`, `warning`, or `critical`.
- `pwdExpInDays`: The number of days before a user's password expires. Available with Oracle Exadata System Software release 19.1.0 or later.
- `pwdExpWarnInDays`: The number of days before a user's password expires that a warning message is issued during login attempts. Available with Oracle Exadata System Software release 19.1.0 or later.
- `rackName`: The name of the rack
- `releaseImageStatus`: Indicator for knowing whether imaging is successful
- `releaseTrackingBug`: Patch number for the cell software, such as 30441371
- `releaseVersion`: Release number for the cell software, such as 19.3.1.0.0.191018
- `remotePwdChangeAllowed`: Whether or not a user password can be changed remotely through REST services. Available with Oracle Exadata System Software release 19.1.0 or later.
- `rescuePlan`: A list of commands that you can run after a server rescue to restore settings, such as IORM plans, thresholds, and notifications, to the last known values.
- `rollbackVersion`: The inactive image version that the cell maintains. If `patchmgr rollback` is invoked for the cell, the value displayed by `rollbackVersion` is the software version that will be reinstated.
- `rpmVersion`: The RPM version of the cell
- `rsStatus`: Status of Restart Server
- `securityCert`: The certified identity of the cell. Either CA-certified identity or the default self-certified identity.
- `siteName`: The site name for the cell
- `smtpFrom`: User name that appears in the `From:` header of the alert notifications
- `smtpFromAddr`: Email address that appears in the `From:` header of the alert notifications. This email address is not authenticated with the email server.
- `smtpPort`: Email server port used to send alert notifications
- `smtpToAddr`: Address to which email is sent. It can be a comma-delimited list in quotation marks to allow multiple subscribers to alerts.
- `smtpUseSSL`: Specification to use Secure Socket Layer (SSL) encryption for alert notifications.
- `snmpSubscriber`: List of hosts that subscribe to the SNMP alert notifications
- `snmpUser`: Defines the user who receives SNMP alerts

- `status`: Status of the cell
- `storageIndexPersMode`: Controls the persistent storage index feature
- `syslogConf`: Designates syslog messages that should be forwarded to a specified management server. Uses the following syntax for the attribute, where *selector* is the message type, and *node* is the specified server:

  ```
  syslogconf = (selector @node' [, 'selector @node']... )
  ```

  Both *selector* and *node* follow `syslog.conf` standard syntax rules.
- `syslogFormat`: A string representing the format for syslog messages. Available with Oracle Exadata System Software release 19.1.0 or later.
- `temperatureReading`: Current temperature (Celsius) of the cell obtained from the BMC
- `temperatureStatus`: Status of the temperature. The value can be `normal`, `warning`, or `critical`.
- `traceLevel`: The level for which trace messages are written. The default is `FINE`. The value can be:
  - A valid Java logging level
    * `SEVERE`
    * `WARNING`
    * `INFO`
    * `CONFIG`
    * `FINE`
    * `FINER`
    * `FINEST`
  - 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
- `usbStatus`: Status of the USB device

Examples

The following example shows the `DESCRIBE` command with the `CELL` object.

Example 7-96   Describing the CELL Object

```
CellCLI> DESCRIBE CELL

    name                    modifiable
    accessLevelPerm         modifiable
    accessLevelTemp         modifiable
    accountLockInDays       modifiable
    bbuStatus
    cellsrvStatus
    cellVersion
    comment                 modifiable
    cpuCount
    dbPerfDataSuppress      modifiable
    diagHistoryDays         modifiable
    diagPackEmailAttach     modifiable
    diagPackUploadEnabled   modifiable
    doNotServiceLEDStatus
    eighthRack              modifiable
    emailFormat             modifiable
    emailSubscriber         modifiable
    events                  modifiable
    exacliEnabled           modifiable
    fanCount
    fanStatus
    flashCacheMode          modifiable
    httpsAccess             modifiable
    id
    interconnect1           modifiable
    interconnect2           modifiable
    interconnect3           modifiable
    interconnect4           modifiable
    interconnect5           modifiable
    interconnect6           modifiable
    interconnect7           modifiable
    interconnect8           modifiable
    interconnectCount
    iormBoost
    IOTimeoutThreshold      modifiable
    ipaddress1
    ipaddress2
    ipaddress3
    ipaddress4
    ipaddress5
    ipaddress6
    ipaddress7
    ipaddress8
    kernelVersion
    location                modifiable
    locatorLEDStatus
    mailServer              modifiable
    makeModel
    memoryGB
    metricCollection        modifiable
    metricHistoryDays       modifiable
    msStatus
    notificationMethod     modifiable
    notificationPolicy     modifiable
    offloadGroupEvents     modifiable
    pmemCacheMode          modifiable
    powerCount
    powerStatus
    pwdExpInDays           modifiable
    pwdExpWarnInDays       modifiable
    rackName               modifiable
    releaseImageStatus
    releaseTrackingBug
    releaseVersion
    remotePwdChangeAllowed modifiable
    rescuePlan             hidden
    rollbackVersion
    rpmVersion
    rsStatus
    securityCert           modifiable
    siteName               modifiable
    smtpFrom               modifiable
    smtpFromAddr           modifiable
    smtpPort               modifiable
    smtpToAddr             modifiable
    smtpUseSSL             modifiable
    snmpSubscriber         modifiable
    snmpUser               modifiable
    status
    syslogConf             modifiable
    syslogFormat           modifiable
    temperatureReading
    temperatureStatus
    traceLevel             modifiable
    upTime
    usbStatus
```

Related Topics

- ALTER CELL
- CREATE CELL

7.7.5.5 DESCRIBE CELLDISK

Purpose

The `DESCRIBE CELLDISK` command displays a list of attributes for the `CELLDISK` object type.

Syntax

```
DESCRIBE CELLDISK
```

Usage Notes

The attributes displayed by the `DESCRIBE CELLDISK` command can include:

- `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.
- `flushError`: Errors reported by while flushing the flash cache.
- `flushStatus`: The current status of the flash cache flush operation.
- `freeSpace`: Amount of unused space available on the cell disk.
- `id`: Global unique identifier (GUID) that is generated when the cell disk is created.
- `name`: Unique name of the cell disk.
- `physicalDisk`: Name of the physical disk on which the cell disk is located.
- `size`: Total size of the cell disk.
- `status`: Current status of the cell disk, such as `normal` or `importRequired`.

Example 7-97 Describing the CELLDISK Object

This shows the `DESCRIBE` command with the `CELLDISK` object.

```
CellCLI> DESCRIBE CELLDISK

         name                     modifiable
         comment                  modifiable
         creationTime
         deviceName
         devicePartition
         diskType
         errorCount
         flushError               hidden
         flushStatus              hidden
         freeSpace
         id
         physicalDisk
         size
         status
```

Related Topics

- ALTER CELLDISK
- CREATE CELLDISK

7.7.5.6 DESCRIBE DATABASE

Purpose

Displays the specified attributes for active databases.

Syntax

```
DESCRIBE DATABASE
```

Usage Notes

The attributes for the `DESCRIBE DATABASE` command include the following:

- `name`: The database name.
- `asmClusterName`: The name of the associated ASM cluster. This attribute is populated when ASM-scoped security is configured.
- `databaseID`: The unique identifier for the database.
- `flashCacheLimit`: Specifies the amount of flash cache space that is available to the database. The value is a soft limit, so the database may consume more space if the flash cache is not full.
- `flashCacheMin`: Specifies the minimum amount of flash cache space that is guaranteed for the database.
- `flashCacheSize`: Specifies the maximum amount of flash cache space that is available to the database. The value is a hard limit that cannot be exceeded at any time.
- `iormDiskLimit`: Specifies the I/O utilization limit for the database, expressed as a percentage of the available disk resources.
- `iormDiskShare`: Specifies the relative share of disk I/O resources that is available to the database. A higher share value implies higher priority and more access to the I/O resources.
- `iormFlashLimit`: Specifies the I/O utilization limit for the database, expressed as a percentage of the available flash resources.
- `iormFlashShare`: Specifies the relative share of flash I/O resources that is available to the database. A higher share value implies higher priority and more access to the I/O resources.
- `lastRequestTime`: The time stamp of the last request from the database.
- `pmemCacheLimit`: Specifies the amount of PMEM cache space that is available to the database. The value is a soft limit, so the database may consume more space if the PMEM cache is not full.
- `pmemCacheMin`: Specifies the minimum amount of PMEM cache space that is guaranteed for the database.
- `pmemCacheSize`: Specifies the maximum amount of PMEM cache space that is available to the database. The value is a hard limit that cannot be exceeded at any time.
- `profile`: The IORM profile associated with the database.
- `xrmemCacheLimit`: Specifies the amount of XRMEM cache space that is available to the database. The value is a soft limit, so the database may consume more space if the XRMEM cache is not full.
- `xrmemCacheMin`: Specifies the minimum amount of XRMEM cache space that is guaranteed for the database.
- `xrmemCacheSize`: Specifies the maximum amount of XRMEM cache space that is available to the database. The value is a hard limit that cannot be exceeded at any time.

Examples

The following example shows the `DESCRIBE` command with the `DATABASE` object.

Example 7-98   Describing the DATABASE Object

```
CellCLI> DESCRIBE DATABASE
        name
        asmClusterName
        databaseID
        flashCacheLimit
        flashCacheMin
        flashCacheSize
        iormDiskLimit
        iormDiskShare
        iormFlashLimit
        iormFlashShare
        lastRequestTime
        pmemCacheLimit
        pmemCacheMin
        pmemCacheSize
        profile
        xrmemCacheLimit
        xrmemCacheMin
        xrmemCacheSize
```

Related Topics

- LIST DATABASE
- LIST IORMPROFILE

7.7.5.7 DESCRIBE DISKMAP

Purpose

Displays the grid disk attributes for a physical disk.

Syntax

```
DESCRIBE DISKMAP
```

Usage Notes

The attributes displayed by the `DESCRIBE DISKMAP` command can include:

- `celldisk`: The cell disk name
- `devicePartition`: The disk partition name
- `gridDisks`: The name of the grid disks associated with the disk
- `name`: The disk name.
- `physicalSerial`: The serial number of the disk
- `physicalSize`: The size of the disk

- `slotNumber`: The slot number of the disk
- `status`: The disk status

Examples

The following example shows the `DESCRIBE` command with the `DISKMAP` object.

Example 7-99 Describing the DISKMAP Object

```
CellCLI> DESCRIBE CELLDISK

         name
         celldisk
         devicePartition
         gridDisks
         physicalSerial
         physicalSize
         slotNumber
         status
```

7.7.5.8 DESCRIBE FLASHCACHE

Purpose

The `DESCRIBE FLASHCACHE` command displays a list of attributes for the `FLASHCACHE` object type.

Syntax

```
DESCRIBE FLASHCACHE
```

Usage Notes

The attributes displayed by the `DESCRIBE FLASHCACHE` command can include:

- `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 cell disks.
- `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

The following example shows the `DESCRIBE` command with the `FLASHCACHE` object.

Example 7-100 Describing the FLASHCACHE Object

```
CellCLI> DESCRIBE FLASHCACHE

         name
         cellDisk                 modifiable
         creationTime
         degradedCelldisks
         effectiveCacheSize
         id
         size                     modifiable
         status
```

7.7.5.9 DESCRIBE FLASHCACHECONTENT

Purpose

The `DESCRIBE FLASHCACHECONTENT` command displays a list of attributes for the `FLASHCACHECONTENT` object type.

Syntax

```
DESCRIBE FLASHCACHECONTENT
```

Usage Notes

The attributes displayed by the `DESCRIBE FLASHCACHECONTENT` command can include:

- `cachedKeepSize`: Size, in bytes, cached in `keep` mode for this object.
- `cachedSize`: Size, in bytes, cached for this object.
- `cachedWriteSize`: Size, in bytes, of cached data for this object in write-back flash cache that has not yet been written to hard disk.
- `columnarCacheSize`: Size, in bytes, cached in Hybrid Columnar Compression (HCC) format for this object.
- `columnarKeepSize`: Size, in bytes, cached in Hybrid Columnar Compression (HCC) format that is 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`: Mostly specifies the Oracle Database dictionary object number of the database object (table, index, partition, and so on) that is associated with the `FLASHCACHECONTENT` object.

  Additionally, the following values have special meaning:

  – `0` (zero) indicates that the object number is undefined. This value is often used in conjunction with internal I/O performed by ASM.

  - `4294967292` indicates that the `FLASHCACHECONTENT` object contains ASM Dynamic Volume Manager (ADVM) data.
  - `4294967293` indicates cached redo log data.
  - `4294967294` and `4294967295` indicate data from internal database objects, such as temporary segments, rollback segments, control file data, and so on, which is not associated with a specific data object (table, index, partition, and so on).
- `tableSpaceNumber`: Tablespace number associated with the database object.

Examples

The following example shows the `DESCRIBE` command with the `FLASHCACHECONTENT` object.

Example 7-101   Describing the FLASHCACHECONTENT Object

```
CellCLI> DESCRIBE FLASHCACHECONTENT

         cachedKeepSize
         cachedSize
         cachedWriteSize
         columnarCacheSize
         columnarKeepSize
         dbID
         dbUniqueName
         hitcount
         hoursToExpiration
         missCount
         objectNumber
         tableSpaceNumber
```

7.7.5.10 DESCRIBE FLASHLOG

Purpose

The `DESCRIBE FLASHLOG` command displays a list of attributes for the `FLASHLOG` object type.

Syntax

```
DESCRIBE FLASHLOG
```

Usage Notes

The attributes displayed by the `DESCRIBE FLASHLOG` command can include:

- `cellDisk`: Names of the cell disks that contain Oracle Exadata Smart Flash Log.
- `creationTime`: Timestamp when Oracle Exadata Smart Flash Log was created.
- `degradedCelldisks`: List of cell disks configured for Oracle Exadata Smart Flash Log, but not currently available.
- `effectiveSize`: Size of available Oracle Exadata Smart Flash Log after deducting space on unavailable cell disks.
- `efficiency`: Efficiency of Oracle Exadata Smart Flash Log expressed as a percentage.

- `id`: Global unique identifier (GUID) generated when Oracle Exadata Smart Flash Log is created.
- `name`: Unique name of Oracle Exadata Smart Flash Log.
- `size`: Total size of Oracle Exadata Smart Flash Log.
- `status`: Current status of Oracle Exadata Smart Flash Log, such as `normal`, `warning` or `critical`. Status `normal` indicates all flash disks are available. Status `warning` indicates some flash disks are not available. Status `critical` indicates all flash disks are unavailable.

Examples

The following example shows the `DESCRIBE` command with the `FLASHLOG` object.

Example 7-102 Describing the FLASHLOG Object

```
CellCLI> DESCRIBE FLASHLOG

         name
         cellDisk
         creationTime
         degradedCelldisks
         effectiveSize
         efficiency
         id
         size
         status
```

7.7.5.11 DESCRIBE GRIDDISK

Purpose

The `DESCRIBE GRIDDISK` command displays a list of attributes for the `GRIDDISK` object type.

Syntax

```
DESCRIBE GRIDDISK
```

Usage Notes

The attributes for the `DESCRIBE GRIDDISK` command include the following:

- `asmDeactivationOutcome`: Indicator whether a grid disk can be deactivated without loss of data. A value of `YES` indicates the grid disk can be deactivated without data loss.
- `asmDiskgroupName`: Name of the Oracle ASM disk group.
- `asmDiskName`: Name of the Oracle ASM disk.
- `asmDiskRepairTime`: The amount of time the grid disk can remain offline before it is dropped by Oracle ASM.
- `asmDiskSize`: Size of the Oracle ASM disk.

  This attribute is available in Oracle Exadata System Software release 12.1.2.3.0 and later.
- `asmFailGroupName`: Name of the Oracle ASM failure group.

- `asmModeStatus`: Indicator shows the current Oracle ASM usage of a grid disk. Statuses are `ONLINE`, `OFFLINE`, `DROPPED`, `UNUSED`, `SYNCING`, or `UNKNOWN`.
- `availableTo`: Names of the clients that can access this grid disk.
- `cachedBy`: The name of the flash disks that are currently caching data for this grid disk for write-back flash cache.
- `cachingPolicy`: The flash caching policy for this grid disk. Values are `default` or `none`.
  - `default` means data for this grid disk uses the flash cache.
  - `none` means data for this grid disk do not use flash cache.

  The caching policy can be set when creating a grid disk, or using the `ALTER GRIDDISK` command.
- `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.
- `size`: Total size of the grid disk.
- `sizeAllocated`: Total size of materialized space used by data in a sparse grid disk.

  This attribute is available in Oracle Exadata System Software 22.1.0 and later. It applies to sparse grid disks only.
- `sparse`: Whether the grid disk is a sparse disk.
- `status`: Current status of the grid disk, such as `active`, `inactive`, `not present` or `importRequired`.
- `virtualSize`: The size of the disk group for the sparse grid disks.

> **Note:**
>
> The `asmDeactivationOutcome` and `asmModeStatus` attributes must be explicitly specified when using the `LIST GRIDDISK` command.

Example 7-103 Describing the GRIDDISK Object

This example shows the `DESCRIBE` command with the `GRIDDISK` object.

```
CellCLI> DESCRIBE GRIDDISK

          name                         modifiable
          asmDeactivationOutcome       hidden
          asmDiskgroupName
          asmDiskName
          asmDiskRepairTime            hidden
          asmDiskSize                  hidden
```

```
         asmFailGroupName
         asmModeStatus              hidden
         availableTo                modifiable
         cachedBy
         cachingPolicy              modifiable
         cellDisk
         comment                    modifiable
         creationTime
         diskType
         errorCount
         id
         size                       modifiable
         sizeAllocated
         sparse
         status
         virtualSize                modifiable
```

Related Topics

- CREATE GRIDDISK
- ALTER GRIDDISK

7.7.5.12 DESCRIBE IBPORT

Purpose

The `DESCRIBE IBPORT` command displays a list of attributes for the IBPORT object type on systems that use InfiniBand Network Fabric.

Note:

This command does not apply to Oracle Exadata X8M systems.

Syntax

```
DESCRIBE IBPORT
```

Usage Notes

The attributes for the `DESCRIBE IBPORT` command can include the following:

- `activeSlave`: Indicator whether the port is currently the active port for the bonded IP.
- `dataRate`: The data rate of the InfiniBand Network Fabric port.
- `hcaFWVersion`: The version of the host channel adapter firmware.
- `id`: The Global unique identifier (GUID) of the InfiniBand Network Fabric port.
- `lid`: The local identifier of the InfiniBand Network Fabric 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 Network Fabric port.
- `physLinkState`: The physical link state.
- `portNumber`: The port number of the InfiniBand Network Fabric port.
- `rcvConstraintErrs`: The number of received constraint errors experienced by the InfiniBand Network Fabric port.
- `rcvData`: The number of 32-bit data words received by the InfiniBand Network Fabric port.
- `rcvErrs`: The number of packets received at the InfiniBand Network Fabric port containing an error.
- `rcvRemotePhysErrs`: The number of physical errors experienced at the InfiniBand Network Fabric port.
- `status`: The link status.
- `symbolErrs`: The number of minor link errors experienced at the InfiniBand Network Fabric port.
- `vl15Dropped`: The number of incoming VL15 packets dropped at the InfiniBand Network Fabric port due to resource limitations, such as lack of buffers.
- `xmtConstraintErrs`: The number of transmitted constraint errors experienced at the InfiniBand Network Fabric port.
- `xmtData`: The number of 32-bit data words transmitted on the InfiniBand Network Fabric port.
- `xmtDiscards`: The number of outbound packets discarded by the InfiniBand Network Fabric port because the port was down or congested.

Example 7-104 Describing the IBPORT Object

The following example shows the `DESCRIBE` command with the `IBPORT` object on systems that use InfiniBand Network Fabric.

```
CellCLI> DESCRIBE IBPORT

        name
        activeSlave
        dataRate
        hcaFWVersion
        id
        lid
        linkDowned
        linkIntegrityErrs
        linkRecovers
        physLinkState
        portNumber
        rcvConstraintErrs
        rcvData
        rcvErrs
        rcvRemotePhysErrs
        status
        symbolErrs
        vl15Dropped
        xmtConstraintErrs
        xmtData
        xmtDiscards
```

7.7.5.13 DESCRIBE IORMPLAN

Purpose

The `DESCRIBE IORMPLAN` command displays a list of attributes for the `IORMPLAN` object type.

Syntax

```
DESCRIBE IORMPLAN
```

Usage Notes

The attributes for the `DESCRIBE IORMPLAN` command can include the following:

- `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

The following example shows the `DESCRIBE` command with the `IORMPLAN` object.

Example 7-105   Describing the IORMPLAN Object

```
CellCLI> DESCRIBE IORMPLAN

         name
         catPlan                 modifiable
         dbPlan                  modifiable
         objective               modifiable
         status
```

Related Topics

- ALTER IORMPLAN

7.7.5.14 DESCRIBE KEY

Purpose

The `DESCRIBE KEY` command displays a list of attributes for the `KEY` object type.

Syntax

```
DESCRIBE KEY
```

Usage Notes

The attributes for the DESCRIBE KEY command can include the following:

- key: Random hexadecimal string used to assign client keys.
- name: Name of the key. The value of this field is not displayed with LIST.
- type: The type of key.

Examples

The following example shows the DESCRIBE command with the KEY object.

Example 7-106 Describing the KEY Object

```
CellCLI> DESCRIBE KEY

         name
         key                      modifiable
         type                     modifiable
```

Related Topics

- CREATE KEY

7.7.5.15 DESCRIBE LUN

Purpose

The DESCRIBE LUN command displays a list of attributes for the LUN object type.

Syntax

```
DESCRIBE LUN
```

Usage Notes

The attributes for the DESCRIBE LUN command can include the following:

- cellDisk: The name of the flash disk, for example FD_02_rack1celadm10. Not used for hard disks.
- 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.
- 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 over-provisioned 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

The following example shows the `DESCRIBE` command with the `LUN` object.

Example 7-107 Describing the LUN Object

```
CellCLI> DESCRIBE LUN

         name
         cellDisk
         deviceName
         diskType
         errorCount
         id
         isSystemLun
         lunSize
         lunUID
         lunWriteCacheMode
         overProvisioning
         physicalDrives
         raidLevel
         status
```

Related Topics

- LIST DISKMAP

7.7.5.16 DESCRIBE METRICCURRENT

Purpose

The `DESCRIBE METRICCURRENT` command displays a list of attributes for the `METRICCURRENT` object type.

Syntax

```
DESCRIBE METRICCURRENT
```

Usage Notes

The attributes for the `DESCRIBE METRICCURRENT` command can include the following:

- `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`
  - `CELL_FILESYSTEM`
  - `CELLDISK`
  - `FLASHCACHE`
  - `FLASHLOG`
  - `GRIDDISK`
  - `IBPORT`
  - `IORM_CATEGORY`
  - `IORM_CONSUMER_GROUP`
  - `IORM_DATABASE`
  - `IORM_PLUGGABLE_DATABASE`
  - `HOST_INTERCONNECT`
  - `SMARTIO`

Examples

The following example shows the `DESCRIBE` command with the `METRICCURRENT` object.

Example 7-108 Describing the METRICCURRENT Object

```
CellCLI> DESCRIBE METRICCURRENT

         name
         alertState
         collectionTime
         metricObjectName
         metricType
         metricValue
         objectType
```

Related Topics

- DESCRIBE METRICDEFINITION
- Exadata Metrics
  Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.

7.7.5.17 DESCRIBE METRICDEFINITION

Purpose

The `DESCRIBE METRICDEFINITION` command displays a list of attributes for the `METRICDEFINITION` object type.

Syntax

```
DESCRIBE METRICDEFINITION
```

Usage Notes

The attributes for the `DESCRIBE METRICDEFINITION` command can include the following:

- `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:

  - `CD_` (cell disk)
  - `CG_` (IORM consumer group, database-qualified)
  - `CL_` (cell)
  - `CT_` (IORM category)
  - `DB_` (IORM database-level)
  - `FC_` (flash cache)
  - `FL_` (flash log)
  - `GD_` (grid disk)
  - `IORM`
  - `N_` (network, IBPORT, HOST_INTERCONNECT)
  - `PDB_` (IORM pluggable database)
  - `SIO_` (Smart IO)

  After the abbreviation of the object type, most of the `name` attributes contain one of the following combinations to identify the operation:

  - `IO_BY` (I/O amount)
  - `IO_RQ` (number of I/O requests)
  - `IO_TM` (I/O latency)
  - `IO_WT` (I/O wait time)
  - `FC_IO_BY` (Flash cache I/O amount)
  - `FC_IO_RQ` (Flash cache I/O requests)
  - `FD_IO_BY` (Flash disk I/O amount)
  - `FD_IO_RQ` (Flash disk I/O requests)
  - `FD_IO_TM` (Flash disk latency)

  – FD_IO_UTIL (Flash disk utilization percentage)

  Next, in the name could be _R (for read) or _W (for 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.

- description: Description of the metric.
- fineGrained: Specifies if the metric is enabled for fine-grained collection.
- 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.
- objectType: Type of object being measured. Values are:
  - CELL
  - CELL_FILESYSTEM
  - CELLDISK
  - FLASHCACHE
  - FLASHLOG
  - GRIDDISK
  - IBPORT
  - IORM_CATEGORY
  - IORM_CONSUMER_GROUP
  - IORM_DATABASE
  - IORM_PLUGGABLE_DATABASE
  - HOST_INTERCONNECT
  - SMARTIO
- retentionPolicy: Specifies the retention policy for historical metric observations.

  When retentionPolicy=Default, the retention period for the associated metric is governed by the metricHistoryDays cell attribute. If retentionPolicy=Annual, the associated metric has a one-year retention period.
- streaming: Specifies if the metric is enabled for streaming to a collection endpoint.
- unit: Unit for the metric explicitly, and is related to the metric collected:
  - Number

  - % (percentage)
  - F (Fahrenheit)
  - C (Celsius)
  - IO/sec
  - “IO requests”
  - KB
  - KB/sec
  - MB
  - MB/sec
  - /min
  - ms
  - ms/request
  - ms/sec
  - us (*microseconds*)
  - us/request
  - us/sec

Examples

The following example shows the `DESCRIBE` command with the `METRICDEFINITION` object.

Example 7-109 Describing the METRICDEFINITION Object

```
CellCLI> DESCRIBE METRICDEFINITION

         name
         description
         metricType
         objectType
         persistencePolicy
         unit
```

Related Topics

- LIST METRICDEFINITION
- Exadata Metrics
  Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.

7.7.5.18 DESCRIBE METRICHISTORY

Purpose

The `DESCRIBE METRICHISTORY` command displays a list of attributes for the `METRICHISTORY` object type.

Syntax

```
DESCRIBE METRICHISTORY
```

Usage Notes

The attributes for the `DESCRIBE METRICHISTORY` command can include the following:

- `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
- `metricValueAvg`: Average value of the metric
- `metricValueMax`: Maximum value of the metric
- `metricValueMin`: Minimum value of the metric
- `name`: Name of the current metric
- `objectType`: Type of object being measured. Values are:
  - `CELL`
  - `CELL_FILESYSTEM`
  - `CELLDISK`
  - `FLASHCACHE`
  - `FLASHLOG`
  - `GRIDDISK`
  - `IBPORT`
  - `IORM_CATEGORY`
  - `IORM_CONSUMER_GROUP`
  - `IORM_DATABASE`
  - `IORM_PLUGGABLE_DATABASE`
  - `HOST_INTERCONNECT`
  - `SMARTIO`

Examples

The following example shows the `DESCRIBE` command with the `METRICHISTORY` object.

Example 7-110 Describing the METRICHISTORY Object

```
CellCLI> DESCRIBE METRICHISTORY

         name
         alertState
```

```
        collectionTime
        metricObjectName
        metricType
        metricValue
        metricValueAvg
        metricValueMax
        metricValueMin
        objectType
```

7.7.5.19 DESCRIBE OFFLOADGROUP

Purpose

The `DESCRIBE OFFLOADGROUP` command displays a list of attributes for the `OFFLOADGROUP` object type.

Syntax

```
DESCRIBE OFFLOADGROUP
```

Usage Notes

The attributes for the `DESCRIBE OFFLOADGROUP` command can include the following:

- `autoStart`: Whether the offload server associated with the offload group is dynamically started. Value can be either `true` or `false`.
- `comment`: An optional comment
- `creationTime`: The time when the offload group was created
- `id`: An identifier for the offload group
- `isSystemGroup`: Whether the offload group was created by the system software. The value can be either `true` or `false`.
- `name`: The name of the offload group
- `package`:
- `runtimeState`: The current state of the offload group process. The value can be `running` or `stopped`.

Example 7-111 Describing the OFFLOADGROUP Object

The following example shows ....

```
CellCLI> DESCRIBE OFFLOADGROUP
        name
        autoStart
        comment                 modifiable
        creationTime
        id
        isSystemGroup
        package                 modifiable
        runtimeState
```

7.7.5.20 DESCRIBE PHYSICALDISK

Purpose

The `DESCRIBE PHYSICALDISK` command displays a list of attributes for the `PHYSICALDISK` object type.

Syntax

```
DESCRIBE PHYSICALDISK
```

Usage Notes

The attributes for the `DESCRIBE PHYSICALDISK` command can include the following:

- `ctrlFirmware`: The hard disk controller software version
- `ctrlHwVersion`: The hard disk controller hardware version
- `deviceID`: The ID for the physical disk
- `deviceName`: The name of the physical disk device, for example `/dev/sdx`
- `diskType`: Type of the disk, whether it is a `HardDisk`, `FlashDisk`, or `M2Disk`.
- `enclosureDeviceId`: Identifier for the hard disk enclosure. This attribute is only applicable to Oracle Exadata System Software on Oracle Exadata Storage Server.
- `errCmdTimeoutCount`: The count of execution of commands related to physical disks that timed out, for example, disk firmware upgrade, listing physical disks, and so on.
- `errHardReadCount`: Total count of read errors on a physical disk
- `errHardWriteCount`: Total count of write errors on a physical disk
- `errorCount`: The sum of all known error counts for a physical disk
- `errOtherCount`: Total error count of all other (unknown) errors for a physical disk
- `errSeekCount`: Total number of disk seek errors
- `flashLifeLeft`: The percentage of flash disk life left for a disk
- `hotPlugCount`: Total number of times a disk has been pulled out and reinserted (hot plugged)
- `lastFailureReason`: The reason for the last physical disk failure
- `luns`: List of LUNs converted from this disk. M.2 disks do not have LUNs.
- `makeModel`: Model description provided by the system
- `name`: Unique name of the physical disk
- `notPresentSince`: Date at which the disk was no longer detected
- `physicalFirmware`: The version of the firmware
- `physicalInsertTime`: Time that the disk was inserted
- `physicalInterface`: Interface type used by the hard disk. For example, `SAS`
- `physicalPort`: (Only appplicable for HP models) The physical disk port value

- `physicalRPM`: The RPM value of a physical hard disk. This attribute is also used to determine the disk type (SATA or SAS).
- `physicalSerial`: System-assigned unique ID
- `physicalSize`: Size of the disk in bytes
- `physicalUseType`: Intended use of the disk, for example, `Data Drive`
- `sectorRemapCount`: Total number of physical disk sectors that have been remapped because of sector failures
- `slotNumber`: Physical location of disk. This attribute is only applicable to Oracle Exadata System Software on Oracle Exadata Storage Server.
- `status`: Status of the physical disk. Values can be:
  - `failed`: The disk has failed. In earlier releases, this status was called `critical`.
  - `normal`: The disk is functioning normally
  - `not present`: The disk has been removed
  - `peer failure`: Flash disk failure only
  - `poor performance`: The disk is performing poorly
  - `predictive failure`: The disk is expected to fail
  - `write-through caching`: Flash disk caching only.

Example 7-112 Describing the PHYSICALDISK Object in Oracle Exadata Storage Server

```
CellCLI> DESCRIBE PHYSICALDISK
        name
        ctrlFirmware
        ctrlHwVersion
        deviceId
        deviceName
        diskType
        enclosureDeviceId
        errCmdTimeoutCount
        errHardReadCount
        errHardWriteCount
        errorCount
        errOtherCount
        errSeekCount
        flashLifeLeft
        hotPlugCount
        lastFailureReason
        luns
        makeModel
        notPresentSince
        physicalFirmware
        physicalInsertTime
        physicalInterface
        physicalPort
        physicalRPM
        physicalSerial
        physicalSize
        physicalUseType
        sectorRemapCount
        slotNumber
        status
```

7.7.5.21 DESCRIBE PLUGGABLEDATABASE

Purpose

The `DESCRIBE PLUGGABLEDATABASE` command displays a list of attributes for the `PLUGGABLEDATABASE` object type.

Syntax

```
DESCRIBE PLUGGABLEDATABASE
```

Usage Notes

The attributes for the `DESCRIBE PLUGGABLEDATABASE` command can include the following:

- `asmClusterName`: The Oracle ASM cluster name or alias. Available with Oracle Exadata System Software release 19.1.0 or later.
- `flashCacheLimit`: The specified limit on the Flash cache for this pluggable database (PDB)
- `flashCacheMin`: The specified minimum size of the Flash cache for this PDB
- `flashCacheSize`: The size of the Flash cache specified for this PDB
- `iormLimit`: The disk I/O utilization limit for the PDB. Available with Oracle Exadata System Software release 19.1.0 or later.
- `iormShare`: The IORM share number for the PDB. Available with Oracle Exadata System Software release 19.1.0 or later.
- `name`: The name of the PDB
- `pdbID`: The ID for the PDB
- `pmemCacheLimit`: The specified limit on the PMEM cache for this pluggable database (PDB)
- `pmemCacheMin`: The specified minimum size of the PMEM cache for this PDB
- `pmemCacheSize`: The size of the PMEM cache specified for this PDB
- `name`: The pluggable database (PDB) name.
- `asmClusterName`: The name of the associated ASM cluster. This attribute is populated when ASM-scoped security is configured.
- `containerName`: The name of the container database (CDB)
- `flashCacheLimit`: Specifies the amount of flash cache space that is available to the PDB. The value is a soft limit, so the PDB may consume more space if the flash cache is not full.
- `flashCacheMin`: Specifies the minimum amount of flash cache space that is guaranteed for the PDB.
- `flashCacheSize`: Specifies the maximum amount of flash cache space that is available to the PDB. The value is a hard limit that cannot be exceeded at any time.

- `iormDiskLimit`: Specifies the I/O utilization limit for the PDB, expressed as a percentage of the available disk resources.
- `iormFlashLimit`: Specifies the I/O utilization limit for the PDB, expressed as a percentage of the available flash resources.
- `iormShare`: Specifies the relative share of I/O resources that is available to the PDB. A higher share value implies higher priority and more access to the I/O resources.
- `pdbID`: The unique identifier for the PDB.
- `pmemCacheLimit`: Specifies the amount of PMEM cache space that is available to the PDB. The value is a soft limit, so the PDB may consume more space if the PMEM cache is not full.
- `pmemCacheMin`: Specifies the minimum amount of PMEM cache space that is guaranteed for the PDB.
- `pmemCacheSize`: Specifies the maximum amount of PMEM cache space that is available to the PDB. The value is a hard limit that cannot be exceeded at any time.
- `xrmemCacheLimit`: Specifies the amount of XRMEM cache space that is available to the PDB. The value is a soft limit, so the PDB may consume more space if the XRMEM cache is not full.
- `xrmemCacheMin`: Specifies the minimum amount of XRMEM cache space that is guaranteed for the PDB.
- `xrmemCacheSize`: Specifies the maximum amount of XRMEM cache space that is available to the PDB. The value is a hard limit that cannot be exceeded at any time.

Example 7-113 Describing the PLUGGABLEDATABASE Object

```
CellCLI> DESCRIBE PLUGGABLEDATABASE
        name
        asmClusterName
        containerName
        flashCacheLimit
        flashCacheMin
        flashCacheSize
        iormDiskLimit
        iormFlashLimit
        iormShare
        pdbID
        pmemCacheLimit
        pmemCacheMin
        pmemCacheSize
        xrmemCacheLimit
        xrmemCacheMin
        xrmemCacheSize
```

7.7.5.22 DESCRIBE PMEMCACHE

Purpose

The `DESCRIBE PMEMCACHE` command displays a list of attributes for the `PMEMCACHE` object type.

Note:

The `DESCRIBE PMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
DESCRIBE PMEMCACHE
```

Usage Notes

The attributes displayed by the `DESCRIBE PMEMCACHE` command can include:

- `cellDisk`: Cell disk names used by the PMEM cache.
- `creationTime`: Time stamp when the PMEM cache was created.
- `degradedCelldisks`: List of cell disks configured for the cache but not currently available.
- `effectiveCacheSize`: Usable PMEM cache size after deducting space on unavailable cell disks.
- `id`: Global unique identifier (GUID) that is generated when the PMEM cache is created.
- `name`: Unique name of the PMEM cache.
- `size`: Total size of the PMEM cache.
- `status`: Current status of the PMEM cache, such as `normal`, `warning` or `critical`.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `DESCRIBE XRMEMCACHE` instead of `DESCRIBE PMEMCACHE`.

Examples

The following example shows the `DESCRIBE` command with the `PMEMCACHE` object.

Example 7-114 Describing the PMEMCACHE Object

```
CellCLI> DESCRIBE PMEMCACHE

         name
         cellDisk                 modifiable
         creationTime
         degradedCelldisks
         effectiveCacheSize
         id
         size                     modifiable
         status
```

7.7.5.23 DESCRIBE PMEMLOG

Purpose

The `DESCRIBE PMEMLOG` command displays a list of attributes for the `PMEMLOG` object type.

Note:

The `DESCRIBE PMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
DESCRIBE PMEMLOG
```

Usage Notes

The attributes displayed by the `DESCRIBE PMEMLOG` command can include:

- `cellDisk`: Names of the cell disks that contain PMEMLOG.
- `creationTime`: Timestamp when PMEMLOG was created.
- `degradedCelldisks`: List of cell disks configured for PMEMLOG that are not currently available.
- `effectiveSize`: Size of available PMEMLOG after deducting space on unavailable cell disks.
- `efficiency`: Efficiency of PMEMLOG expressed as a percentage.
- `id`: Global unique identifier (GUID) generated when PMEMLOG is created.
- `name`: Unique name of PMEMLOG.
- `size`: Total size of the PMEMLOG.
- `status`: Current status of PMEMLOG.
  - `normal`: All PMEM cell disks are available.
  - `warning`: Some PMEM cell disks are not available.
  - `critical`: All PMEM cell disks are unavailable.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `DESCRIBE XRMEMLOG` instead of `DESCRIBE PMEMLOG`.

Example 7-115 Describing the PMEMLOG Object

This example shows the `DESCRIBE` command with the `PMEMLOG` object.

```
CellCLI> DESCRIBE PMEMLOG
     name
     cellDisk
     creationTime
     degradedCelldisks
     effectiveSize
     efficiency
     id
     size
     status
```

7.7.5.24 DESCRIBE QUARANTINE

Purpose

The `DESCRIBE QUARANTINE` command displays a list of attributes for the `QUARANTINE` object type.

Syntax

```
DESCRIBE QUARANTINE
```

Usage Notes

The attributes for the `DESCRIBE QUARANTINE` command can include the following:

- `asmClusterId`: Identifier of the ASM cluster. This attribute is available in Exadata software 12.2.1.1.0 and later.
- `catDBPlan`: The name of the category plan
- `cellsrvChecksum`: Checksum of the CELLSRV binary
- `clientPID`: The process identifier for the client process which crashed the cell
- `comment`: Comment for the quarantine
- `conDbUniqueID`: The container database unique ID for the quarantine
- `conDbUniqueName`: The container database unique name 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
- `fineGrainControl`:
- `fineGrainValue`:
- `incidentID`: The incident identifier of the crash that caused the quarantine creation
- `interDBPlan`: The name of the interdatabase resource plan
- `intraDBPlan`: The name of the intradatabase resource plan
- `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.
- `name`: Identifier of the quarantine
- `objectID`:
- `planLineID`: The SQL Plan Line identifier. This is applicable to SQL Plan quarantine only.
- `quarantineMode`:

- `quarantinePlan`: This is usually SYSTEM
- `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.

Example 7-116 Describing the QUARANTINE Object

```
CellCLI> DESCRIBE QUARANTINE
         name
         asmClusterId
         catDBPlan
         cellsrvChecksum
         clientPID
         comment                   modifiable
         conDbUniqueID
         conDbUniqueName
         crashReason
         creationTime
         dbUniqueID
         dbUniqueName
         fineGrainControl
         fineGrainValue
         incidentID
         interDBPlan
         intraDBPlan
         ioBytes
         ioGridDisk
         ioOffset
         objectID
         planLineID
         quarantineMode
         quarantinePlan
         quarantineReason
         quarantineType
         remoteHostName
         rpmVersion
         sqlID
         sqlPlanHashValue
```

7.7.5.25 DESCRIBE ROLE

Purpose

The `DESCRIBE ROLE` command displays a list of attributes for the `ROLE` object type.

Syntax

```
DESCRIBE ROLE
```

Usage Notes

The attributes for the `DESCRIBE ROLE` command can include the following:

- `name`: Unique name of the user assigned the role
- `privileges`: Privileges granted to the role

Example 7-117 Describing the ROLE Object

```
CellCLI> DESCRIBE ROLE
        name
        privileges
```

7.7.5.26 DESCRIBE SOFTWAREHISTORY

Purpose

The `DESCRIBE SOFTWAREHISTORY` command displays a list of attributes for the `ALERTHISTORY` object type.

Syntax

```
DESCRIBE SOFTWAREHISTORY
```

Usage Notes

The attributes for the `DESCRIBE SOFTWAREHISTORY` command can include the following:

- `name`: The name of the software update
- `status`: The status of the software update

Example 7-118 Describing the SOFTWAREHISTORY Object

```
CellCLI> DESCRIBE SOFTWAREHISTORY
        name
        status
```

7.7.5.27 DESCRIBE SOFTWAREUPDATE

Purpose

The `DESCRIBE SOFTWAREUPDATE` command displays a list of attributes for the `SOFTWAREUPDATE` object type.

Syntax

```
DESCRIBE SOFTWAREUPDATE
```

Usage Notes

The attributes for the `DESCRIBE SOFTWAREUPDATE` command can include the following:

- `frequency`: The time period in which this software update is performed automatically. The value can be `none`, `daily`, `weekly`, or `biweekly`. The value `none` is available in Oracle Exadata System Software release 19.1.0 or later.
- `name`: The name of the patch to use in the update, or `unknown`. If the name defaults to `unknown`, then when the software update is performed, the most recent patch is chosen for the upgrade.
- `status`: The status of this software update.
- `store`: The URL for the location of the software update file
- `time`: The specified date and time at which the software update should be performed
- `timeLimitInMinutes`: The number of minutes a cell will spend waiting to update the software before canceling and issuing an alert.

Example 7-119 Describing the SOFTWAREUPDATE Object

```
CellCLI> DESCRIBE SOFTWAREUPDATE
        name                    modifiable
        status
        store                   modifiable
        time                    modifiable
        timeLimitInMinutes      modifiable
```

7.7.5.28 DESCRIBE THRESHOLD

Purpose

The `DESCRIBE THRESHOLD` command displays a list of attributes for the `THRESHOLD` object type.

Syntax

```
DESCRIBE THRESHOLD
```

Usage Notes

The attributes displayed by the `DESCRIBE THRESHOLD` command can include:

- `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

Example 7-120 Describing the THRESHOLD Object

```
CellCLI> DESCRIBE THRESHOLD

         name
         comparison               modifiable
         critical                 modifiable
         observation              modifiable
         occurrences              modifiable
         warning                  modifiable
```

7.7.5.29 DESCRIBE USER

Purpose

The `DESCRIBE USER` command displays a list of attributes for the `USER` object type.

Syntax

```
DESCRIBE USER
```

Usage Notes

The attributes displayed by the `DESCRIBE USER` command can include:

- `name`: Unique name of the user
- `roles`: Roles assigned to the user

Example 7-121 Describing the USER Object

```
CellCLI> DESCRIBE USER
         name
         roles
```

7.7.5.30 DESCRIBE XRMEMCACHE

Purpose

Starting with Exadata X10M, the `DESCRIBE XRMEMCACHE` command displays a list of attributes for the Exadata RDMA Memory Cache.

Note:

On Exadata X8M and X9M systems, the `DESCRIBE XRMEMCACHE` command is equivalent to the `DESCRIBE PMEMCACHE` command.

Syntax

```
DESCRIBE XRMEMCACHE
```

Usage Notes

Starting with Exadata X10M, the attributes displayed by the `DESCRIBE XRMEMCACHE` command include:

- `name`: Unique name of the Exadata RDMA Memory Cache (XRMEM cache).
- `creationTime`: Time stamp when the XRMEM cache was created.
- `effectiveCacheSize`: Usable XRMEM cache size.

Examples

The following example shows the `DESCRIBE` command with the `XRMEMCACHE` object.

Example 7-122   Describing the XRMEMCACHE Object

```
CellCLI> DESCRIBE XRMEMCACHE

         name
         creationTime
         effectiveCacheSize
```

7.7.5.31 DESCRIBE XRMEMCACHECONTENT

Purpose

The `DESCRIBE XRMEMCACHECONTENT` command displays a list of attributes for the `XRMEMCACHECONTENT` object type.

Syntax

```
DESCRIBE XRMEMCACHECONTENT
```

Usage Notes

The attributes displayed by the `DESCRIBE XRMEMCACHECONTENT` command can include:

- `cachedKeepSize`: Size, in bytes, cached in `keep` mode for this object.
- `cachedSize`: Size, in bytes, cached for this object.
- `cachedWriteSize`: Size, in bytes, of cached data for this object in XRMEM cache cache that has not yet been written to hard disk.
- `clusterName`: Name of the cluster associated with the cache object.
- `columnarCacheSize`: Size, in bytes, cached in Hybrid Columnar Compression (HCC) format for this object.
- `columnarKeepSize`: Size, in bytes, cached in Hybrid Columnar Compression (HCC) format that is in `keep` mode for this object.
- `dbID`: Database unique name identifier.

- `dbUniqueName`: Database unique name.
- `hitCount`: Number of I/Os which read data from XRMEM cache for this object.
- `missCount`: Number of I/Os which read data from disk for this object.
- `objectNumber`: Mostly specifies the Oracle Database dictionary object number of the database object (table, index, partition, and so on) that is associated with the `XRMEMCACHECONTENT` object.

  Additionally, the following values have special meaning:

  – `0` (zero) indicates that the object number is undefined. This value is often used in conjunction with internal I/O performed by ASM.

  – `4294967292` indicates that the `XRMEMCACHECONTENT` object contains ASM Dynamic Volume Manager (ADVM) data.

  – `4294967293` indicates cached redo log data.

  – `4294967294` and `4294967295` indicate data from internal database objects, such as temporary segments, rollback segments, control file data, and so on, which is not associated with a specific data object (table, index, partition, and so on).
- `tableSpaceNumber`: Tablespace number associated with the database object.
- `vaultID`: Vault identifier associated with the cache object.

Examples

The following example shows the `DESCRIBE` command with the `XRMEMCACHECONTENT` object.

Example 7-123 Describing the XRMEMCACHECONTENT Object

```
CellCLI> DESCRIBE XRMEMCACHECONTENT

         cachedKeepSize
         cachedSize
         cachedWriteSize
         clusterName
         columnarCacheSize
         columnarKeepSize
         dbID
         dbUniqueName
         hitCount
         missCount
         objectNumber
         tableSpaceNumber
         vaultID
```

7.7.5.32 DESCRIBE XRMEMLOG

Purpose

Starting with Oracle Exadata System Software release 23.1.0, the `DESCRIBE XRMEMLOG` command is equivalent to the `DESCRIBE PMEMLOG` command.

> **Note:**
>
> The `DESCRIBE XRMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

7.7.6 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

- 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.
- DROP ALERTHISTORY
- DROP CELL
- DROP CELLDISK
- DROP FLASHCACHE
- DROP FLASHLOG
- DROP GRIDDISK
- DROP PMEMCACHE
- DROP PMEMLOG
- DROP QUARANTINE
- DROP ROLE
- DROP SOFTWAREHISTORY
- DROP THRESHOLD
- DROP USER
- DROP XRMEMCACHE
- DROP XRMEMLOG

7.7.6.1 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

The following example shows the `DROP ALERTHISTORY` command.

Example 7-124 Dropping a Cell Alert History

```
CellCLI> DROP ALERTHISTORY 1, 2_1, 2_2
```

7.7.6.2 DROP CELL

Purpose

The `DROP CELL` command resets a cell to its original state.

Syntax

```
DROP CELL [ERASE = value] [FORCE]
```

Usage Notes

- This command is run from within the cell.
- 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.
- 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.
- Flash cache compression must be disabled before securely erasing a drive.
- 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
  ```

- Starting with Oracle Exadata System Software release 19.1.0, if you specify to erase hard disks or flash disks using `1pass`, `3pass`, or `7pass` method on Oracle Exadata Database Machine X5 or later, Oracle Exadata System Software automatically invokes Secure Eraser to erase the disks. Secure Eraser determines whether or not the disks can be erased using the better and faster cryptographic erasure method. If some of the disks are eligible, then the cryptographic erasure method is used to erase those disks, and the originally requested method (1/3/7 pass) is used on the other disks. This feature is not used on system disks.

See Table 7-3 for a list of the erasure methods available for each type of device.

The following table shows approximate time needed to securely erase a drive using the supported algorithms. When multiple grid disks or cell disks are dropped with the `ERASE` option, the command runs in parallel on all disks and flash drives. However, the recommended method of erasing data from a cell is to use Secure Erase. See Securely Erasing Database Servers and Storage Servers in *Oracle Exadata Database Machine Security Guide*.

Table 7-2 Estimated Erasure Times for Devices by Erasure Method

Type of Device	`1pass`	`3pass`	`7pass`	Cryptographic
600 GB hard drive	1 hour	3 hours	7 hours	not available
1.2 TB hard drive	1.67 hours	5 hours	11.67 hours	not available
2 TB hard drive	5 hours	15 hours	35 hours	not available
3 TB hard drive	7 hours	21 hours	49 hours	not available
4 TB hard drive	8 hours	24 hours	56 hours	not available
8 TB hard drive	13.17 hours	39.5 hours	92.17 hours	1 min
10 TB hard drive	14 hours	42 hours	98 hours	1 min
14 TB hard drive	18 hours	54 hours	126 hours	1 min
22.875 GB flash drive	not available	not available	21 minutes	not available
93 GB flash drive	not available	not available	32 minutes	not available
186 GB flash drive	not available	not available	36 minutes	not available
1.6 TB flash drive	not available	not available	5.5 hours	1 min
3.2 TB flash drive	not available	not available	8 hours	1 min
Persistent Memory (PMEM) device	not available	not available	not available	1 min
4 GB Internal USB	not available	30 minutes	not available	not available
8 GB Internal USB	not available	1 hour	not available	not available
150 GB M.2 device	not available	not available	not available	1 minute
ILOM	not available	not available	not available	1 minute

Example 7-125 Dropping a Cell

```
CellCLI> DROP CELL FORCE
```

Related Topics

- ALTER CELL

7.7.6.3 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 a cell disk, you should remove the corresponding grid disks from any associated Oracle ASM disk groups and then drop the grid disks.

Syntax

```
DROP CELLDISK { ALL [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] 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 `LUN` associated with the `CELLDISK` is flagged as automatically created, then that `LUN` is deleted along with the cell disk.
- 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.

  Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to drop cell disks matching only the specified flash media type.
- 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.
- If the specified cell disk includes flash cache, and that flash cache is in `writeback` mode, then the cell disk cannot be dropped.
- Starting with Oracle Exadata System Software release 19.1.0, if you specify to erase hard disks or flash disks using `1pass`, `3pass`, or `7pass` method on Oracle Exadata Database Machine X5 or later, Oracle Exadata System Software automatically invokes Secure Eraser to erase the disks. Secure Eraser determines whether or not the disks can be erased using the better and faster cryptographic erasure method. If some of the disks are eligible, then the cryptographic erasure method is used to erase those disks, and the originally requested method (1/3/7 pass) is used on the other disks. This feature is not used on system disks.
- 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.
  - `3pass`: Three passes, and the content is overwritten with set data patterns. This option follows the recommendations from the National Nuclear Security Administration (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 the United States Department of Defense (DOD).

  See Table 7-2 for the approximate erasure times for each disk and erasure method.
- Use the `NOWAIT` option with the `ERASE` option to run the command asynchronously.

- When dropping all cell disks using the `1pass` or `3pass` option, you must drop the flash disks first using the `7pass` option, and then drop the cell disks, for example:

  ```
  CellCLI> DROP CELLDISK ALL FLASHDISK ERASE=7pass
  CellCLI> DROP CELLDISK ALL ERASE=1pass
  ```

The following table gives a summary of the secure erasure methods used for each device type. Hard drives, flash devices, and internal USBs are securely erased in parallel: the time required to erase one device is the same as that required for erasing multiple devices of the same kind.

Table 7-3 Methods Used to Securely Erase Various Devices

Component	Make or Model	Erasure Method
Hard drive	• 8 TB hard drives on Oracle Exadata X5 • All hard drives on Oracle Exadata X6 or later	Cryptographic erase
Hard drive	All other hard drives	1/3/7-Pass erase
Flash device	Flash devices on Oracle Exadata X5 or later	Cryptographic erase
Flash device	All other flash devices	7-pass erase
M.2 device	Oracle Exadata Database Machine X7-2 or later	Cryptographic erase
Persistent Memory (PMEM) device	Oracle Exadata X8M or X9M	Cryptographic erase

Example 7-126 Examples of 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
```

Related Topics

- DROP CELL
- Securely Erasing Database Servers and Storage Servers

7.7.6.4 DROP FLASHCACHE

Purpose

The `DROP FLASHCACHE` command removes Exadata Smart Flash Cache from a cell.

Syntax

```
DROP FLASHCACHE
```

Usage Notes

Before dropping flash cache, the data not synchronized with the grid disk (dirty data) must be flushed from flash cache to the grid disks. Not flushing dirty data may cause data loss.

Examples

The following example shows how to remove Exadata Smart Flash Cache from a cell.

Example 7-127   Removing Exadata Smart Flash Cache

```
CellCLI> DROP FLASHCACHE
```

Related Topics

- ALTER FLASHCACHE

7.7.6.5 DROP FLASHLOG

Purpose

The `DROP FLASHLOG` command removes Oracle Exadata Smart Flash Log from a cell.

Syntax

```
DROP FLASHLOG [FORCE]
```

Usage Notes

The `DROP FLASHLOG` command can be run at runtime, but the command does not complete until all redo data on the flash disk is written to hard disk.

If `FORCE` is not specified, then the `DROP FLASHLOG` command fails if there is any saved redo. If `FORCE` is specified, then all saved redo is purged, and Oracle Exadata Smart Flash Log is removed.

> **Caution:**
>
> If `DROP FLASHLOG` fails due to the existence of saved redo, then do not use the `FORCE` option unless you are sure that all saved redo is no longer needed for any databases to perform recovery. Contact Oracle Support Services for additional information.

Examples

The following example shows how to remove Exadata Smart Flash Cache from a cell.

Example 7-128   Removing Oracle Exadata Smart Flash Log from a Cell

```
CellCLI> DROP FLASHLOG

CellCLI> DROP FLASHLOG FORCE
```

7.7.6.6 DROP GRIDDISK

Purpose

The `DROP GRIDDISK` command removes the named grid disks or removes all the grid disks specified by the `ALL` option.

> **Caution:**
>
> Before dropping a grid disk, ensure that it is not part of any Oracle ASM disk group.

Syntax

```
DROP GRIDDISK
   { ALL [[ CAPACITYOPTIMIZED | PERFORMANCEOPTIMIZED ] FLASHDISK | HARDDISK ]
PREFIX={[']gdisk_name_prefix[']|'gdisk_name_prefix1[,gdisk_name_prefix2]...'}
   | gdisk_name1[,gdisk_name2]... }
   [ERASE=value [NOWAIT]] [FORCE]
```

Usage Notes

- If one or more grid disk names (*gdisk_name1*, *gdisk_name1*, and so on) are specified, then each name identifies the individual grid disk to be removed.
- Starting with Oracle Exadata System Software release 24.1.0, you can optionally specify `CAPACITYOPTIMIZED` or `PERFORMANCEOPTIMIZED` before `FLASHDISK` to drop grid disks on cell disks associated with the specified flash media type.

  If you do not fully specify the flash media type, `ALL FLASHDISK` is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK` on Extreme Flash (EF) storage servers containing capacity-optimized flash devices. On all other storage servers, `ALL FLASHDISK` is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.
- If the `ALL` option is specified with a media type qualifier (`FLASHDISK` or `HARDDISK`), the command only drops grid disks on cell disks associated with the specified media type.

  Starting with Oracle Exadata System Software release 24.1.0, if you specify the `ALL` option without a media type qualifier (`FLASHDISK` or `HARDDISK`), then the command drops grid disks on the cell disks associated with the primary storage media on the storage server. Specifically:
  - On Extreme Flash (EF) storage servers containing capacity-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL CAPACITYOPTIMIZED FLASHDISK`.
  - On Extreme Flash (EF) storage servers containing only performance-optimized flash devices, `ALL` with no media type qualifier is equivalent to `ALL PERFORMANCEOPTIMIZED FLASHDISK`.
  - On all storage servers containing hard disk drives (HDDs), `ALL` with no media type qualifier is equivalent to `ALL HARDDISK`.

  Before Oracle Exadata System Software release 24.1.0, if the `ALL` option is specified without a media type qualifier, the command drops all matching grid disks regardless of media type.
- The `PREFIX` option must be specified when `ALL` is used.

The `PREFIX` option specifies one or more comma-separated prefix strings, which are used to identify the grid disks being dropped.

- 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 you drop a flash-based grid disk, the space is not automatically allocated to `FLASHCACHE`. You can use the `CREATE FLASHCACHE` command to reuse 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 value is not available for flash drives.
  – `3pass`: Three passes, and the disk is overwritten with set data patterns. This option follows the recommendations from NSA. 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.
- `DROP GRIDDISK ERASE` cannot be used for PMEM grid disks.
- 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.

Example 7-129 Examples of 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
```

Related Topics

- DROP CELL
- Dropping a Disk from an Oracle ASM Disk Group
  You can drop a grid disk from a disk group.
- ALTER GRIDDISK

7.7.6.7 DROP PMEMCACHE

Purpose

The `DROP PMEMCACHE` command removes the PMEM Cache from a cell.

Note:

The `DROP PMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
DROP PMEMCACHE
```

Usage Notes

- If the PMEM Cache is in write-back mode, then before dropping the PMEM cache, any data not synchronized with the grid disk (dirty data) must be flushed from PMEM cache to the disks. Failure to flush dirty data can cause data loss.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `DROP XRMEMCACHE` instead of `DROP PMEMCACHE`.

Example 7-130   Removing PMEM Cache from a Storage Server

```
DROP PMEMCACHE
```

7.7.6.8 DROP PMEMLOG

Purpose

The `DROP PMEMLOG` command removes PMEM log from a cell disk.

Note:

The `DROP PMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
DROP PMEMLOG [FORCE]
```

Usage Notes

- You can use the `DROP PMEMLOG` command at run-time, but the command does not complete until all redo data on PMEM is flushed to disk.

- If `FORCE` is not specified, then the `DROP PMEMLOG` command fails if there is any saved redo. If `FORCE` is specified, then all data is purged, and PMEM log is removed.

> **Caution:**
>
> If you are unable to drop the PMEM log due to existing redo information in the log, then retry the command first, before using the `FORCE` option. The `FORCE` option should only be used in extreme circumstances because it may cause redo log copies to be out of sync, possibly resulting in database redo data corruption. Contact Oracle Support Services for additional information.

- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `DROP XRMEMLOG` instead of `DROP PMEMLOG`.

Example 7-131 Removing PMEM Log from a Storage Server

This example shows how to remove PMEM log from a storage server.

```
CellCLI> DROP PMEMLOG
```

7.7.6.9 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

The following example shows the `DROP QUARANTINE` command.

Example 7-132 Dropping Quarantines

```
CellCLI> DROP QUARANTINE 1
```

7.7.6.10 DROP ROLE

Purpose

The `DROP ROLE` command removes user roles from the cell.

Syntax

```
DROP ROLE  { ALL | role_name1 [, role_name2, ...]} [FORCE]
```

Usage Notes

The `FORCE` option drops the role when the role has been granted to a user.

Examples

The following example shows how to drop a role.

Example 7-133 Dropping a Role

```
CellCLI>DROP ROLE gd_monitor
```

7.7.6.11 DROP SOFTWAREHISTORY

Purpose

The `DROP SOFTWAREHISTORY` command removes all history or individual update history.

Syntax

```
DROP SOFTWAREHISTORY { ALL | 'update_name[,update_name...]'}
```

Example 7-134 Dropping the History of Scheduled Software Updates

```
CellCLI> DROP SOFTWAREHISTORY '12.2.1.2.0.170509,12.2.1.2.0.17052'

CellCLI> DROP SOFTWAREHISTORY ALL
```

7.7.6.12 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

The following example shows the `DROP THRESHOLD` command.

Example 7-135 Dropping Thresholds

```
CellCLI> DROP THRESHOLD ct_io_wt_rq.interactive

CellCLI> DROP THRESHOLD ALL
```

Related Topics

- DESCRIBE THRESHOLD

7.7.6.13 DROP USER

Purpose

The `DROP USER` command removes a user from a cell.

Syntax

```
DROP USER { ALL | user1 [, user2]... }
```

Examples

The following example shows how to drop a user.

Example 7-136   Dropping a User

```
CellCLI>DROP USER agarcia
```

7.7.6.14 DROP XRMEMCACHE

Purpose

Starting with Exadata X10M, the `DROP XRMEMCACHE` command removes the Exadata RDMA Memory Cache (XRMEM cache).

Note:

On Exadata X8M and X9M systems, the `DROP XRMEMCACHE` command is equivalent to the `DROP PMEMCACHE` command.

Syntax

```
DROP XRMEMCACHE
```

Usage Notes

Starting with Exadata X10M, the XRMEM cache is re-created automatically during Cell Server (CELLSRV) startup if it does not exist on supported storage servers.

7.7.6.15 DROP XRMEMLOG

Purpose

Starting with Oracle Exadata System Software release 23.1.0, the `DROP XRMEMLOG` command is equivalent to the `DROP PMEMLOG` command.

Note:

The `DROP XRMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

7.7.7 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.

7.7.8 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.

Using cell disk export and import does not preserve the security configuration details associated with ASM-scoped or DB-scoped security. Consequently, after a successful cell disk export and import, you must remove and reconfigure ASM-scoped or DB-scoped security.

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.

Before exporting a cell disk, the data not synchronized with the grid disk (dirty data) must be flushed from flash cache to the grid disks. Not flushing dirty data may cause data loss.

Examples

The following example shows the `EXPORT CELLDISK` command.

Example 7-137 Exporting a Cell Disk

```
CellCLI> EXPORT CELLDISK CD_3_cell01

CellCLI> EXPORT CELLDISK ALL
```

Related Topics

- ALTER CELLDISK

7.7.9 GRANT

Purpose

The `GRANT` command sets attributes for privileges and roles.

Syntax

```
GRANT object_type [name] TO sub_object_type [sub_object_name]
```

Usage Notes

- *object_type* can be as follows:
  - `PRIVILEGE`
  - `ROLE`
- The following values can be used for `PRIVILEGE` object type:
  - *name* is in the following format:

    ```
    { ALL ACTIONS | action } ON { ALL OBJECTS | object }                \
    [{ ALL ATTRIBUTES | ATTRIBUTES attribute1 [, attribute2, ...] }]    \
    [{ WITH ALL OPTIONS | WITH OPTIONS option1 [, option2, ...] }]
    ```

  - The *sub_object_type* must be `ROLE`.
  - The *sub_object_name* is the name of the role.
- The following can be used for the `ROLE` object type:
  - *name* is the role name.
  - The *sub_object_type* must be `USER`.
  - The *sub_object_name* is the name of the user.
- GRANT PRIVILEGE

- GRANT ROLE

7.7.9.1 GRANT PRIVILEGE

Purpose

The `GRANT PRIVILEGE` command sets the access privileges for a role.

Syntax

```
GRANT PRIVILEGE { ALL ACTIONS | action } ON { ALL OBJECTS | object }    \
{ ALL ATTRIBUTES | ATTRIBUTES attribute1 [, attribute2, ...] }          \
{ WITH ALL OPTIONS | WITH OPTIONS option1 [, option2, ...] }            \
TO ROLE { ALL | role1 [, role2, ...] }
```

Usage Notes

- *action* is the command. Examples: `ALTER`, `CREATE`, `DESCRIBE`, `DROP`, `EXPORT`, `IMPORT`, `LIST`.

  Notes:

  - The `GRANT` and `REVOKE` commands cannot be granted.
  - `CREATE USER` and `DROP USER` cannot be granted.
  - `CREATE ROLE` and `DROP ROLE` cannot be granted.
- *object* is object type for the action. It can be any CellCLI object. Examples: `CELL`, `THRESHOLD`, `PHYSICALDISK`, `ALERTHISTORY`, `ROLE`.
- *attribute* are the attributes for the object. To get a list of attributes for an object, run the `LIST object_type` command.
- *option* are the options for the object. Examples: `DETAIL`, `LIMIT`, `ORDER BY`, `WHERE`.
- *role* is the name of the role to grant privileges.
- The `ALL ACTIONS` argument grants privileges for all actions.
- The `ALL OBJECTS` argument grants privileges for all objects.
- The `ALL ATTRIBUTES` argument grants privileges for all attributes.
- The `WITH ALL OPTIONS` argument grants privileges for all options.
- Specifying attributes and `WITH OPTIONS` is optional. If they are not specified, then all attributes and options are granted with the privilege.

Examples

Example 7-138 Granting Privileges to a Role

This example shows how to grant privileges to a role.

```
CellCLI> GRANT PRIVILEGE list on alerthistory ATTRIBUTES
alertAction,alertMessage  \
        WITH OPTIONS detail TO ROLE cellmonitor
```

Example 7-139 Granting All Attributes and Options to a Role

This example shows how to grant all attributes and options for a specified action and object to a role.

```
CellCLI> GRANT PRIVILEGE { ALL ACTIONS | action } ON { ALL OBJECTS | object }
to ROLE role1
```

Example 7-140 Granting All Options with Specified Action, Object and Attributes

This example shows how to grant all options with a specified action, object and attributes to a role.

```
CellCLI> GRANT PRIVILEGE { ALL ACTIONS | action } ON { ALL OBJECTS |
object }  \
ATTRIBUTES attribute1 [, attribute2, ...] to ROLE role1
```

Example 7-141 Granting All Attributes with Specified Action, Object and Options

This example shows how to grant all attributes with a specified action, object, and options to a role.

```
CellCLI> GRANT PRIVILEGE { ALL ACTIONS | action } ON { ALL OBJECTS |
object }   \
WITH OPTIONS option1 [, option, ...] to ROLE role1
```

7.7.9.2 GRANT ROLE

Purpose

The `GRANT ROLE` command sets the role for a user.

Syntax

```
GRANT ROLE { ALL | role1 [, role2, ...] } TO USER { ALL | user1 [, user2...] }
```

Usage Notes

- *role* is the name of the role.
- The `ALL` argument grants all roles to the user.
- The `TO USER ALL` argument grants the role to all users.

Examples

Example 7-142 Granting a Role to a User

This example shows how to grant a role to a user.

```
CellCLI> GRANT ROLE gd_monitor TO USER agarcia
```

7.7.10 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.

The following example shows examples of the `HELP` command.

Example 7-143 Display Help Text with the HELP Command

```
CellCLI> HELP
CellCLI> HELP ALTER
CellCLI> HELP ALTER CELL
```

7.7.11 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.

Using cell disk export and import does not preserve the security configuration details associated with ASM-scoped or DB-scoped security. Consequently, after a successful cell disk export and import, you must remove and reconfigure ASM-scoped or DB-scoped security.

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.
- The `LIST CELLDISK` command shows which cell disks need to be imported. The command displays output similar to the following:

  ```
  CellCLI> list celldisk
           CD_01_cell00     normal
           CD_01_cell01     normal
           CD_01_cell02     importRequired
           CD_01_cell03     importForceRequired
           CD_01_cell04     importRequired
           CD_01_flash0     normal
           CD_01_flash1     normal
           CD_01_flash2     normal
           CD_01_log00      normal
           CD_01_log01      normal
  ```
- 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. Oracle recommends contacting Oracle Support Services before using the `FORCE` option.

- 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 the `IMPORT` command to reconstruct the grid disks on the disk, and make them visible to Oracle ASM.

Example 7-144 Importing a Cell Disk

This example shows the `IMPORT CELLDISK` command. 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.

```
CellCLI> IMPORT CELLDISK CD_7_cell04 lun=3

CellCLI> IMPORT CELLDISK ALL
```

7.7.12 LIST

Purpose

The `LIST` command displays attributes for Oracle Exadata System Software objects. Displayed objects may be identified by name or by filters.

Syntax

```
LIST object_type  [ name | attribute_filters] [attribute_list] [DETAIL]  \
[ORDER BY attribute [ASC| DESC][, attribute [ASC| DESC], ...] \
[LIMIT integer]
```

Usage Notes

- 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 may be specified using the attribute list.

  You can use the `DESCRIBE` command to view the complete list of attributes for any object type.
- 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. Attribute values that are character strings with embedded blank spaces or tabs must be enclosed in quotation marks.
- 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. Blank lines separate each object in the display.
- 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.
- The `ORDER BY` option orders attributes in ascending or descending order. The default is `ASC`.
- The `LIMIT` option sets a limit on the number of displayed attributes. The maximum value is `100` when `LIMIT` is used with the `ORDER BY` option.

- LIST ACTIVEREQUEST
- LIST ALERTDEFINITION
- LIST ALERTHISTORY
- LIST CELL
- LIST CELLDISK
- LIST DATABASE
- LIST DIAGPACK
- LIST DISKMAP
- LIST FLASHCACHE
- LIST FLASHCACHECONTENT
- LIST FLASHLOG
- LIST GRIDDISK
- LIST IBPORT
- LIST IORMPLAN
- LIST IORMPROFILE
- LIST KEY
- LIST LUN
- LIST METRICCURRENT
- LIST METRICDEFINITION
- LIST METRICHISTORY
- LIST METRICSTREAM
- LIST OFFLOADGROUP
- LIST OFFLOADPACKAGE
- LIST PHYSICALDISK
- LIST PLUGGABLEDATABASE
- LIST PMEMCACHE
- LIST PMEMLOG
- LIST QUARANTINE
- LIST ROLE
- LIST SOFTWAREHISTORY

- LIST SOFTWAREUPDATE
- LIST THRESHOLD
- LIST USER
- LIST XRMEMCACHE
- LIST XRMEMCACHECONTENT
- LIST XRMEMLOG

Related Topics

- Attribute Lists in LIST Command
  You can specify which attributes to display for the `LIST` command with the optional `ATTRIBUTES` clause.
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

7.7.12.1 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

You can use the DESCRIBE ACTIVEREQUEST command to view the complete list of `ACTIVEREQUEST` attributes.

Example 7-145 Listing ACTIVEREQUEST Attributes

This example shows the `LIST` command with the `ACTIVEREQUEST` object.

```
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
```

7.7.12.2 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

You can use the DESCRIBE ALERTDEFINITION command to view the complete list of `ALERTDEFINITION` attributes.

Example 7-146   Listing ALERTDEFINITION Attributes

This example shows the `LIST` command with the `ALERTDEFINITION` object.

```
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
```

7.7.12.3 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

- You can use the DESCRIBE ALERTHISTORY command to view the complete list of `ALERTHISTORY` attributes.
- A `WHERE` clause can include the `ageInMInutes` attribute to specify that the output is limited to those alerts that 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 7-147 shows the `LIST` command with the `ALERTHISTORY` object.

Example 7-148 shows open stateful and stateless alerts.

Example 7-149 shows only open stateful alerts. The closed stateful alerts are filtered out.

Example 7-150 shows alerts that have not been cleared.

Example 7-147 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] [] [] [] [] [ ] []"

CellCLI> LIST ALERTHISTORY 7 DETAIL
         name:                   7
         alertMessage:           "Flash cache mode is set to WriteBack because
                                 there is dirty data in the flash cache."
         alertSequenceID:        7
         alertShortName:         Software
         alertType:              Stateless
         beginTime:              2012-09-10T13:22:38-07:00
         examinedBy:
         metricObjectName:       FlashCache
         notificationState:      0
         sequenceBeginTime:      2012-09-10T13:22:38-07:00
         severity:               info
         alertAction:            "If the newly-assigned mode for flash cache
is
                                 not wanted, then change it using the ALTER
CELL
                                 command as described in the Oracle Exadata
```

```
user's
                                 guide."
```

Example 7-148   Listing Open Stateful and Stateless Alerts

```
CellCLI> LIST ALERTHISTORY ATTRIBUTES alertsequenceid,name,alerttype    \
         WHERE endtime=null

         1       1       Stateless
         3       3       Stateless
         11      11_1    Stateful
```

Example 7-149   Listing Open Stateful Alerts

```
CellCLI> LIST ALERTHISTORY WHERE endtime=null AND alerttype=stateful
```

Example 7-150   Listing Alerts That Have Not Cleared

```
CellCLI> LIST ALERTHISTORY WHERE endtime=null

1       2014-11-11T11:08:15-08:00  info      "Factory defaults restored for
 Adapter 0"
3       2014-11-11T11:27:06-08:00  critical   "RS-700 [No IP found in Exadata
 config file] [Check cellinit.ora]
                                               [] [] [] [] [] [] [] [] [] []"
11_1    2014-12-19T12:01:06-08:00  critical    "The HDD disk controller
battery
 has failed. All disk drives have been placed in WriteThrough caching mode.
Disk
 write performance may be reduced. The flash drives are not affected. Battery
 Serial Number : 1142  Battery Type         : ibbu08  Battery
Temperature   : 39
 C  Full Charge Capacity  : 773 mAh  Relative Charge       : 83%  Ambient
 Temperature   : 32 C"
```

7.7.12.4 LIST CELL

Purpose

The `LIST CELL` command displays specified attributes of the cell.

Syntax

```
LIST CELL [ ATTRIBUTES attribute_list ] [ DETAIL ]
```

Usage Notes

- You can use the DESCRIBE CELL command to view the complete list of `CELL` attributes.
- `LIST CELL` only displays information about the local cell.
- To monitor the status of cell components, use the `LIST` command to verify the value of `status`, `fanStatus`, `temperatureStatus`, and `powerStatus`.

Examples

Example 7-151 shows the `LIST` command with the `CELL` object, and the corresponding output.

Example 7-152 shows how to display the status of cell components.

Example 7-153 shows how to display the values of the `snmpSubscriber` attribute.

Example 7-154 shows how to display the value of the `emailFormat` attribute.

Example 7-155 shows how to display the value of the `locatorLEDStatus` attribute.

Example 7-156 shows how to display the value of the `doNotServiceLEDStatus` attribute.

Example 7-157 shows how to display the value of the `bbuLearnCycleTime` attribute.

Example 7-158 shows how to display the value of the `rescuePlan` attribute.

Example 7-159 shows how to retrieve the value of the `httpsAccess` attribute.

Example 7-151   Listing Cell Information

```
CellCLI> LIST CELL

         cell01        online
```

Example 7-152   Displaying the Status of Cell Components

```
CellCLI> LIST CELL ATTRIBUTES name, status, location, -
         fanStatus, temperatureStatus, powerStatus

         cell01      online  rack5:shelf1     normal  normal  normal
```

Example 7-153   Displaying the snmpSubscriber Attribute

```
CellCLI> LIST CELL ATTRIBUTES snmpSubscriber

((host=server1.example.com,port=3873,community=public, type=asr))
```

Example 7-154   Displaying E-mail Format

```
CellCLI> LIST CELL ATTRIBUTES emailFormat
         html
```

Example 7-155   Displaying locatorLEDStatus

```
CellCLI> LIST CELL ATTRIBUTES locatorLEDStatus
         off
```

Example 7-156   Displaying doNotServiceLEDStatus

```
CellCLI> LIST CELL ATTRIBUTES doNotServiceLEDStatus
         on
```

Example 7-157 Listing the bbuLearnCycleTime Attribute

```
CellCLI> LIST CELL ATTRIBUTES bbuLearnCycleTime
```

Example 7-158 Displaying rescuePlan

```
CellCLI> LIST CELL ATTRIBUTES rescuePlan

CREATE ROLE "admin"

GRANT PRIVILEGE all actions ON diagpack all attributes WITH all options TO
ROLE "admin"

CREATE ROLE "diagRole"

GRANT PRIVILEGE download ON diagpack all attributes WITH all options TO ROLE
"diagRole"

GRANT PRIVILEGE create ON diagpack all attributes WITH all options TO ROLE
"diagRole"

GRANT PRIVILEGE list ON diagpack all attributes WITH all options TO ROLE
"diagRole"

ALTER CELL accessLevelPerm="remoteLoginEnabled", diagHistoryDays="7",
metricHistoryDays="7", notificationMethod="mail,snmp",
notificationPolicy="warning,critical,clear",
snmpSubscriber=((host="localhost", port=162, community="public", type=asr)),
bbuLearnCycleTime="2016-10-17T02:00:00-07:00", bbuLearnSchedule="MONTH 1 DATE
17 HOUR 2 MINUTE 0", alertSummaryStartTime="2016-09-21T17:00:00-07:00",
alertSummaryInterval=weekly, hardDiskScrubInterval=biweekly,
hardDiskScrubFollowupIntervalInDays="14"

ALTER IORMPLAN objective=basic
```

Example 7-159 Displaying the HTTPs Access Control List

This example shows how to view the HTTPs access control list for the Exadata RESTful service.

```
CellCLI> LIST CELL ATTRIBUTES httpsAccesss
         ALL
```

The value of `ALL` is the default value and allows access to all hosts.

7.7.12.5 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

You can use the DESCRIBE CELLDISK command to view the complete list of `CELLDISK` attributes.

Examples

The following example shows the `LIST` command with the `CELLDISK` object, and the corresponding output.

Example 7-160 Listing Cell Disk Attributes

```
CellCLI> LIST CELLDISK CD_01_cell05 ATTRIBUTES size

         557.859375G

CellCLI> LIST CELLDISK WHERE status!=normal ATTRIBUTES name

         CD_01_cell03

CellCLI> LIST CELLDISK WHERE DEVICENAME LIKE '/dev/c0d[2-5]' -
         ATTRIBUTES name, size

         CD_01_cell05             557.859375G

CellCLI> LIST CELLDISK CD_01_cell05 DETAIL

         name:                   CD_01_cell05
         comment:
         creationTime:           2018-03-21T13:39:15-04:00
         deviceName:             /dev/sdi
         devicePartition:        /dev/sdi
         diskType:               HardDisk
         errorCount:             0
         freespace:              0
         id:                     00000117-84d2-ed2c-0000-000000000000
         physicalDisk:           K7N5JJ
         size:                   557.859375G
         status:                 normal
```

7.7.12.6 LIST DATABASE

Purpose

Displays the specified attributes for active databases.

Syntax

```
LIST DATABASE [name | attribute_filters] [attribute_list]  [DETAIL]
```

Usage Notes

You can use the DESCRIBE DATABASE command to view the complete list of `DATABASE` attributes.

Examples

The following example shows the `LIST` command with the `DATABASE` object, and the corresponding output.

Example 7-161   Listing Database Attributes

```
CellCLI> LIST DATABASE
         DB01


CellCLI> LIST DATABASE DETAIL
         name:                   DB01
         asmClusterName:         SALESDBS_ASMCLUSTER
         databaseID:             1234567656
         lastRequestTime:        2016-10-27T07:46:36-07:00
         profile:                GOLD
         flashCacheMin:          4.00390625G
         flashCacheLimit:        4.19921875G
         flashCacheSize:         0
         pmemCacheMin:           2.001953125G
         pmemCacheLimit:         2.099609375G
         pmemCacheSize:          0


CellCLI> LIST DATABASE ATTRIBUTES NAME, PROFILE
         ASM
         TEST50          GOLD
         TEST100         GOLD
         TEST150         SILVER
         TEST20          GOLD
         TEST200         BRONZE
         TEST180         SILVER
         TEST175         SILVER
         TEST225         BRONZE
         TEST230         BRONZE
         TEST300
         TEST280
         TEST245         BRONZE

CellCLI> LIST DATABASE ATTRIBUTES NAME, DATABASEID WHERE PROFILE = 'GOLD'
         TEST50          50
         TEST100         100
         TEST20          20
```

7.7.12.7 LIST DIAGPACK

Purpose

The `LIST DIAGPACK` command lists the diagnostic packages in your system, along with their status.

Syntax

```
LIST DIAGPACK [DETAIL]
```

Usage Notes

The location of the diagnostic packages is $LOG_HOME.

Examples

Example 7-162 Output of the "list diagpack" Command

This example shows the output of the LIST DIAGPACK command.

```
CellCLI> LIST DIAGPACK
scab01cel04_diag_2015_09_30T13_29_06_1.tar.bz2    (7 minutes ago)
scab01cel04_2015_09_30T13_13_00_2_1.tar.bz2    (23 minutes ago for alert: 2_1)
scab01cel04_2015_09_30T13_07_10_1_1.tar.bz2    (28 minutes ago for alert: 1_1)
```

Example 7-163 Output of the "list diagpack" command with the DETAIL option

This example shows the output of the LIST DIAGPACK command with the DETAIL option.

```
CellCLI> LIST DIAGPACK DETAIL
Name:                scab01cel04_diag_2015_09_30T13_29_06_1.tar.bz2
Time:                Wed, 30 Sep 2015 13:29:06 (7 minutes ago)
Type:                Custom package

Name:                scab01cel04_2015_09_30T13_13_00_2_1.tar.bz2
Time:                Wed, 30 Sep 2015 13:13:00 (23 minutes ago)
Alert ID:            2_1
Alert description:   InfiniBand Port HCA-1:2 indicates invalid state.

Name:                scab01cel04_2015_09_30T13_07_10_1_1.tar.bz2
Time:                Wed, 30 Sep 2015 13:07:10 (28 minutes ago)
Alert ID:            1_1
Alert description:   File system "/" is 84% full
```

Related Topics

- CREATE DIAGPACK

7.7.12.8 LIST DISKMAP

Purpose

Displays the specified grid disk attributes for a physical disk.

Syntax

```
LIST DISKMAP
```

Usage Notes

You can use the DESCRIBE DISKMAP command to view the complete list of `DISKMAP` attributes.

Examples

The following example shows the `LIST` command with the `DISKMAP` object, and the corresponding output.

Example 7-164 Listing Grid Disk Attributes for a Physical Disk

```
CELLCLI> LIST DISKMAP

Name   PhysicalSerial  SlotNumber    Status  PhysicalSize  CellDisk  DevicePartition  GridDisks
27:0   E0XH34          0             normal   559G         CD_00_sgrcel2   /dev/sda3
"DATA_CD_00_sgrcel2, RECO_CD_00_sgrcel2"

27:1   E0XH2S          1             normal   559G         CD_01_sgrcel2   /dev/sdb3
"DATA_CD_01_sgrcel2, RECO_CD_01_sgrcel2"

27:2   E0Z0CS           2            normal   559G         CD_02_sgrcel2   /dev/sdc
"DATA_CD_02_sgrcel2, DBFS_CD_02_sgrcel2, RECO_CD_02_sgrcel2"
.
.
.
```

7.7.12.9 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

You can use the DESCRIBE FLASHCACHE command to view the complete list of `FLASHCACHE` attributes.

Examples

The following example shows the `LIST` command with the `FLASHCACHE` object, and the corresponding output.

Example 7-165 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:              2012-08-04T15:42:42-07:00
```

```
degradedCelldisks:
effectiveCacheSize:     192M
id:                     8a0adc84-9088-4c4e-8e1c-b6bcbd5cb1ba
size:                   192M
status:                 normal
```

7.7.12.10 LIST FLASHCACHECONTENT

Purpose

The `LIST FLASHCACHECONTENT` command displays attributes for the Exadata Smart Flash Cache entries determined by the specified attributes.

Syntax

```
LIST FLASHCACHECONTENT [attribute_filters] [attribute_list] [DETAIL]
```

Usage Notes

You can use the DESCRIBE FLASHCACHECONTENT command to view the complete list of `FLASHCACHECONTENT` attributes.

Examples

Example 7-166 Listing Exadata Smart Flash Cache Content Attributes

This example shows the `LIST` command with the `FLASHCACHECONTENT` object, and the corresponding output.

```
CellCLI> LIST FLASHCACHECONTENT DETAIL

         cachedKeepSize:              8192
         cachedSize:                  16384
         cachedWriteSize:             16384
         clusterName:                 CLUSTER-C1
         columnarCacheSize:           0
         columnarKeepSize:            0
         dbID:                        3557170052
         dbUniqueName:                TEST1
         hitCount:                    4
         missCount:
         objectNumber:                23102
         tableSpaceNumber:            1

         cachedKeepSize:              0
         cachedSize:                  983040
         cachedWriteSize:             983040
         clusterName:                 CLUSTER-C1
         columnarCacheSize:           0
         columnarKeepSize:            0
         dbID:                        4325252357
         dbUniqueName:                MYODB
         hitCount:                    1
         missCount:                   1
```

```
           objectNumber:                4294967295
           tableSpaceNumber:            4
```

Example 7-167   Listing Exadata Smart Flash Cache Content by Database Object

This example 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 Oracle Exadata Storage Server is then queried.

```
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 WHERE objectNumber=63197 DETAIL
           cachedKeepSize:              0
           cachedSize:                  983040
           cachedWriteSize:             983040
           clusterName:                 CLUSTER-C1
           columnarCacheSize:           0
           columnarKeepSize:            0
           dbID:                        3557170052
           dbUniqueName:                TEST1
           hitCount:                    1
           missCount:                   1
           objectNumber:                63197
           tableSpaceNumber:            4


  CellCLI> LIST FLASHCACHECONTENT WHERE objectNumber=63198 DETAIL
         cachedKeepSize:              0
         cachedSize:                  16384
         cachedWriteSize:             16384
         clusterName:                 CLUSTER-C1
         columnarCacheSize:           0
         columnarKeepSize:            0
         dbID:                        3557170052
         dbUniqueName:                TEST1
         hitCount:                    0
         missCount:                   2
         objectNumber:                63198
         tableSpaceNumber:            4
```

Example 7-168 Listing Exadata Smart Flash Cache Content for an ASMCLUSTER Client

Starting with Oracle Exadata System Software release 19.1.0, the `dbUniqueName` attribute is qualified with the ASMCLUSTER client name if ASM-scoped security is configured. This example shows the partial output of the `LIST FLASHCACHECONTENT` command for database instances associated with Oracle ASM clusters.

```
  CellCLI> LIST FLASHCACHECONTENT WHERE dbuniquename LIKE 'ASM.*' DETAIL

         cachedKeepSize:         0
         cachedSize:             65536
         cachedWriteSize:        65536
         columnarCacheSize:      0
         columnarKeepSize:       0
         dbID:                   3334479949
         dbUniqueName:           ASM1.DB1.CDB$ROOT
         hitCount:               0
         missCount:              0
         objectNumber:           75307
         tableSpaceNumber:       1

...

         cachedKeepSize:         0
         cachedSize:             2957312
         cachedWriteSize:        0
         columnarCacheSize:      0
         columnarKeepSize:       0
         dbID:                   1238079488
         dbUniqueName:           ASM1.DB1.PDB1
         hitCount:               4
         missCount:              47
         objectNumber:           4294967294
         tableSpaceNumber:       1

...

         cachedKeepSize:         0
         cachedSize:             17326080
         cachedWriteSize:        0
         columnarCacheSize:      0
         columnarKeepSize:       0
         dbID:                   1757889862
         dbUniqueName:           ASM2.DB2
         hitCount:               9
         missCount:              255
         objectNumber:           4294967294
         tableSpaceNumber:       5
```

Example 7-169 Listing Exadata Smart Flash Cache Content for Cached Redo Logs

Starting with Oracle Exadata System Software release 20.1.0, Exadata Smart Flash Cache can contain cached redo logs. Cached redo log entries are identified by `objectNumber =`

4294967293. This example shows a `LIST FLASHCACHECONTENT` command that displays details for cached redo logs.

```
  CellCLI> LIST FLASHCACHECONTENT -
           ATTRIBUTES dbUniqueName, dbID, tableSpaceNumber, objectNumber, 
cachedSize -
           WHERE objectNumber = 4294967293

         ACME1           3557170052      0         4294967293      140795904
         MYODB           4325252357      0         4294967293      142036992
```

7.7.12.11 LIST FLASHLOG

Purpose

The `LIST FLASHLOG` command displays attributes for the Oracle Exadata Smart Flash Log entries determined by the specified attributes.

Syntax

```
LIST FLASHLOG
```

Usage Notes

You can use the DESCRIBE FLASHLOG command to view the complete list of `FLASHLOG` attributes.

Examples

The following example shows the `LIST` command with the `FLASHLOG` object, and the corresponding output.

Example 7-170 Listing Oracle Exadata Smart Flash Log Attributes

```
CellCLI> LIST FLASHLOG

         raw_FLASHLOG normal

CellCLI> LIST FLASHLOG DETAIL

         name:                   raw_FLASHLOG
         cellDisk:               c9FLASH0,FD_FLASH1_raw,FD_FLASH2_raw
         creationTime:           2011-01-23T12:34:56-05:00
         degradedCelldisks:
         effectiveSize:          512M
         efficiency:             100.0
         id:                     8a0aadc84-908804c4e08e1c-b6bcbd5cb1ba
         size:                   512M
         status:                 normal
```

7.7.12.12 LIST GRIDDISK

Purpose

The `LIST GRIDDISK` command displays attributes for one or more Oracle Exadata Storage Server grid disks determined by the specified attributes and filters.

Syntax

```
LIST GRIDDISK [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Usage Notes

- You can use the DESCRIBE GRIDDISK command to view the complete list of `GRIDDISK` attributes.
- The `asmDeactivationOutcome` attribute can be used to determine if a grid disk can 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.
  - `DROPPED`: Oracle ASM has dropped this grid disk.
  - `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.
- When the `cachingPolicy` attribute is set to none, the associated flash cache is used for write I/O latency capping and logging. It is not used for caching.

Examples

Example 7-171
Example 7-172
Example 7-173
Example 7-174
Example 7-175
Example 7-176

Example 7-171 Listing Grid Disk Attributes

This example shows the `LIST` command with the `GRIDDISK` object, and the corresponding output.

```
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:            CLUSTER-C1
         size:                   10G
         errorCount:             0
         diskType:               HardDisk
         cachedBy:               FD_01_FLASH, FD02_FLASH, FD03_FLASH
         cachingPolicy:          default

CellCLI> LIST GRIDDISK DATA_CD_01_cell05 ATTRIBUTES size

         136.640625G

CellCLI> LIST GRIDDISK WHERE status!=active ATTRIBUTES name

         data_CD_01_1_abcd2x3

CellCLI> LIST GRIDDISK data4_CD_09_cell01 DETAIL

         name:                   data4_CD_09_cell01
         availableTo:
         cellDisk:               CD_09_cell01
         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 7-172   Determining if a Grid Disk can be Deactivated

This example shows the `asmDeactivationOutcome` attribute being used to determine if a grid disk can be deactivated.

```
CellCLI> LIST GRIDDISK ATTRIBUTES name, asmDeactivationOutcome

         QUAL_CD_00_cell01       Yes
         PROD_CD_02_cell01       Cannot de-activate due to other offline
disks in
                                 the diskgroup
         TEST_CD_03_cell01       Yes
         DATA_CD_04_cell01       Yes
         DATA_CD_05_cell01       Yes
         DATA_CD_06_cell01       Yes
         RECO_CD_01_cell01       Cannot de-activate due to other offline
disks in
                                 the diskgroup
         DATA_CD_08_cell01       Yes
         DATA_CD_09_cell01       Yes
         DATA_CD_10_cell01       Yes
         DATA_CD_11_cell01       Yes
```

Example 7-173 Viewing the Current Usage of a Grid Disk

This example shows the `asmModeStatus` attribute being used to check the current usage of a grid disk.

```
CellCLI> LIST GRIDDISK ATTRIBUTES name, asmModeStatus

         QUAL_CD_00_cell01        UNUSED
         RECO_CD_01_cell01        OFFLINE
         PROD_CD_02_cell01        SYNCING
         TEST_CD_03_cell01        UNKNOWN
         DATA_CD_04_cell01        ONLINE
         DATA_CD_05_cell01        ONLINE
         DATA_CD_06_cell01        ONLINE
         DATA_CD_07_cell01        ONLINE
         DATA_CD_08_cell01        ONLINE
         DATA_CD_09_cell01        ONLINE
         DATA_CD_10_cell01        ONLINE
         DATA_CD_11_cell01        ONLINE
```

Example 7-174 Checking the Status Using the LIST GRIDDISK Command

This example 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.

```
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 7-175 Checking the Status of Secure Erase

This example shows the `LIST GRIDDISK` command being used to check the status of grid disks that are being erased.

```
CellCLI> LIST GRIDDISK
         DATA_CD_00_cell01     active
         DATA_CD_05_cell01     active
         DATA_CD_06_cell01     erase in progress
         DATA_CD_07_cell01     erase in progress
```

Example 7-176 Listing Devices Used to Cache Grid Disks

This example shows the devices that are currently being used to cache the grid disks.

```
CellCLI> LIST GRIDDISK ATTRIBUTES name, diskType, size, cachedby

   C_DATA_CD_00_dm01celadm01   HardDisk  3.41796875T   FD_02_dm01celadm01
   C_DATA_CD_01_dm01celadm01   HardDisk  3.41796875T   FD_01_dm01celadm01
   C_DATA_CD_02_dm01celadm01   HardDisk  3.41796875T   FD_00_dm01celadm01
   C_DATA_CD_03_dm01celadm01   HardDisk  3.41796875T   FD_02_dm01celadm01
   C_DATA_CD_04_dm01celadm01   HardDisk  3.41796875T   FD_01_dm01celadm01
```

7.7.12.13 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

- You can use the DESCRIBE IBPORT command to view the complete list of `IBPORT` attributes.
- If the `activeSlave` attributes for both InfiniBand ports on a server are not listed, then active-active bonding is being used. If an `activeSlave` attribute is `TRUE`, then active-passive bonding is being used.

Examples

The following example shows the `LIST` command with the `IBPORT` object, and the corresponding output.

Example 7-177 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
```

7.7.12.14 LIST IORMPLAN

Purpose

The `LIST IORMPLAN` command lists the current plan of the local cell.

Syntax

```
LIST IORMPLAN [attribute_list]  [DETAIL]
```

Usage Notes

You can use the DESCRIBE IORMPLAN command to view the complete list of `IORMPLAN` attributes.

Examples

The following example shows the `LIST` command with the `IORMPLAN` object, and the corresponding output.

Example 7-178   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
```

7.7.12.15 LIST IORMPROFILE

Purpose

The `LIST IORMPROFILE` command enables you to list IORM profiles. To see which databases are associated with an IORM profile, use the LIST DATABASE command with the new `PROFILE` attribute.

Syntax

```
LIST IORMPROFILE [name | filters]
```

Usage Notes

name specifies the IORM profile to display.

filters specifies an expression that determines which IORM profiles to display.

If *name* and *filters* are omitted, all the IORM profiles are displayed.

Examples

The following example shows the `LIST IORMPROFILE` command, and the corresponding output.

Example 7-179 LIST IORMPROFILE

```
CellCLI> LIST IORMPROFILE
GOLD
SILVER
BRONZE
```

Related Topics

- Using IORM Profiles
  I/O Resource Management (IORM) interdatabase plans support profiles to ease management, and configuration of interdatabase plans for hundreds of databases.

7.7.12.16 LIST KEY

Purpose

The `LIST KEY` command displays key values for clients determined by the specified attributes and filters.

Syntax

```
LIST KEY [name | filters] [attribute_list] [DETAIL]

LIST KEY [FOR {ASMCLUSTER | CELL | LOCAL CELL | REMOTE CELL}] [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.
- The output of the command might show a `type` attribute, which was introduced in Oracle Exadata System Software release 12.2.1.1.0. Values for this attribute include `ASMCLUSTER`, `LOCAL-CELL`, `REMOTE-CELL`, and `CELL`. See the examples below.
- The `FOR [LOCAL | REMOTE] CELL` and `FOR ASMCLUSTER` clauses were also introduced in Oracle Exadata System Software release 12.2.1.1.0. These clauses indicate that the command is to display only the keys with the specified type.
- *`name`* specifies the key to display.

Examples

The following example shows the `LIST` command with the `KEY` object, and the corresponding output.

Example 7-180 Listing KEY Attributes

```
CellCLI>  LIST KEY db1 DETAIL
         name:                    db1
         key:                     b67d5587fe728118af47c57ab8da650a


CellCLI>  LIST KEY
         db1     b67d5587fe728118af47c57ab8da650a
         db456   118af47c57ab8da650ab67d5587fe728
         asm1    118af47c57ab8da650ab67d5587fe728   ASMCLUSTER


CellCLI>  LIST KEY asm1 DETAIL
         name:          asm1
         key:           b67d5587fe728118af47c57ab8da650a
         type:          ASMCLUSTER


CellCLI>  LIST KEY FOR CELL DETAIL
         key: fa292e11b31b210c4b7a24c5f1bb4d32
         type: CELL
```

Related Topics

- About Security Keys

7.7.12.17 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

You can use the DESCRIBE LUN command to view the complete list of `LUN` attributes.

Examples

The following example shows the `LIST` command with the `LUN` object, and the corresponding output.

Example 7-181 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
```

7.7.12.18 LIST METRICCURRENT

Purpose

The `LIST METRICCURRENT` command displays the most current metric observations.

Syntax

```
LIST METRICCURRENT [ metric_name |  attribute_filters ]  [attribute_list]
[DETAIL]
```

Usage Notes

- You can use the DESCRIBE METRICCURRENT command to view the complete list of `METRICCURRENT` attributes.
- Running the `LIST METRICCURRENT` command with no other options displays the most current metric observation for all metrics, which results in hundreds of lines of output. To generate more manageable output, use an attribute filter or identify a specific metric name.
- By default, the `LIST METRICCURRENT` command does not display fine-grained metric observations. To display fine-grained metric observations, run `LIST METRICCURRENT WHERE collection = finegrained`.

Examples

Example 7-182 shows the `LIST` command with filters to display information about the `METRICCURRRENT` object, and the corresponding output.

Example 7-183 shows the `LIST METRICCURRENT` command with the `ORDER BY` and `LIMIT` options.

Example 7-182 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

CellCLI> LIST METRICCURRENT CG_IO_UTIL_LG
         CG_IO_UTIL_LG   RDB1.BATCH_GROUP                                 0 %
         CG_IO_UTIL_LG   RDB1.INTERACTIVE_GROUP                           0 %
         CG_IO_UTIL_LG   RDB1.OTHER_GROUPS                                0 %
         CG_IO_UTIL_LG   RDB2.BATCH_GROUP                                0 %
         CG_IO_UTIL_LG   RDB2.INTERACTIVE_GROUP                          0 %
         CG_IO_UTIL_LG   RDB2.OTHER_GROUPS                               0 %
```

Example 7-183 Listing METRICCURRENT Attributes Using ORDER BY and LIMIT

```
CellCLI> LIST METRICCURRENT attributes name, metricObjectName, alertState,   \
         metricValue ORDER BY metricValue desc, metricObjectName asc,        \
         name desc LIMIT 3

CD_IO_TM_R_LG    c9FLASH0    normal    160,514,088   us
CD_IO_TM_R_LG    c9FLASH1    normal    156,659,463   us
DB_IO_TM_SM      ASM         normal    33,111,890    us
```

Related Topics

- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.
- Exadata Metrics
  Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.

7.7.12.19 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

You can use the DESCRIBE METRICDEFINITION command to view the complete list of `METRICDEFINITION` attributes.

Examples

The following example shows the `LIST` command with the `METRICDEFINITION` object, and the corresponding output.

Example 7-184   Listing the Metric Definitions for a Specific Object

```
CellCLI> LIST metricDefinition WHERE objectType=cell
         CL_BBU_CHARGE
         CL_BBU_TEMP
         CL_CPUT
         CL_CPUT_CS
         CL_CPUT_MS
         CL_FANS
         CL_IO_RQ_NODATA
         CL_IO_RQ_NODATA_SEC
         CL_MEMUT
         CL_MEMUT_CS
         CL_MEMUT_MS
         CL_RUNQ
         CL_SWAP_IN_BY_SEC
         CL_SWAP_OUT_BY_SEC
         CL_SWAP_USAGE
         CL_TEMP
         CL_VIRTMEM_CS
         CL_VIRTMEM_MS
         IORM_MODE
         N_HCA_MB_RCV_SEC
         N_HCA_MB_TRANS_SEC
         N_NIC_KB_RCV_SEC
         N_NIC_KB_TRANS_SEC
```

Example 7-185   Listing the Metric Definition Detail for a Specific Metric

```
CellCLI> LIST metricDefinition WHERE name=cl_swap_in_by_sec DETAIL
         name:                   CL_SWAP_IN_BY_SEC
         description:            "Amount of swap pages read in KB per second"
         fineGrained:            Disabled
         metricType:              Instantaneous
         objectType:              CELL
         streaming:               Disabled
         unit:                    KB/sec
```

Related Topics

- Exadata Metrics
  Exadata metrics are recorded observations of important properties or values relating to the Exadata system software.

7.7.12.20 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

- You can use the DESCRIBE METRICHISTORY command to view the complete list of `METRICHISTORY` attributes.
- The retention period for most metric history files is specified by the `metricHistoryDays` cell attribute. The default retention period is 7 days. You can modify this setting with the CellCLI `ALTER CELL` command.

  In addition to the metrics governed by the `metricHistoryDays` cell attribute, a subset of key metric observations are retained for up to one year. In all cases, historical metric observations are purged automatically if the server detects a storage space shortage.
- 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 7-186   Listing METRICHISTORY Attributes

This example 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.

```
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 7-187   Listing METRICHISTORY Using the OVER and MEMORY Attributes

This example shows the `LIST METRICHISTORY` command with the `OVER` and `MEMORY` options.

```
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
%
```

Example 7-188   Listing METRICHISTORY Attributes Using ORDER BY and LIMIT

This example shows the `LIST METRICHISTORY` command with the `ORDER BY` and `LIMIT` options.

```
CellCLI> LIST METRICHISTORY WHERE name like '.*IO_RQ.*' DETAIL ORDER BY  -
         metricValue desc, metricObjectName desc LIMIT 4

name:                   CT_FD_IO_RQ_SM
alertState:             normal
collectionTime:         2014-05-23T10:59:06-07:00
metricObjectName:       OTHER
metricType:             Cumulative
metricValue:            3,211,568 IO requests
objectType:             IORM_CATEGORY

name:                   CT_FD_IO_RQ_SM
```

```
alertState:             normal
collectionTime:         2014-05-23T10:58:06-07:00
metricObjectName:       OTHER
metricType:             Cumulative
metricValue:            3,211,568 IO requests
objectType:             IORM_CATEGORY

name:                   CT_FD_IO_RQ_SM
alertState:             normal
collectionTime:         2014-05-22T17:23:45-07:00
metricObjectName:       OTHER
metricType:             Cumulative
metricValue:            3,211,568 IO requests
objectType:             IORM_CATEGORY

name:                   CT_FD_IO_RQ_SM
alertState:             normal
collectionTime:         2014-05-22T17:21:41-07:00
metricObjectName:       OTHER
metricType:             Cumulative
metricValue:            3,211,568 IO requests
objectType:             IORM_CATEGORY
```

Example 7-189 Listing METRICHISTORY Attributes with ASM-scoped security configured

This example for Oracle Exadata System Software release 19.1.0 shows the `LIST METRICHISTORY` command with the `DETAIL` option for a system with ASM-scoped security configured.

```
CellCLI> LIST METRICHISTORY WHERE name like 'DB_IO_RQ_SM' DETAIL ORDER BY  -
         metricObjectName LIMIT 4

name:                   DB_IO_RQ_SM
alertState:             normal
collectionTime:         2018-08-23T07:34:12-05:00
metricObjectName:       ASM
metricType:             Cumulative
metricValue:            *.
objectType:             IORM_DATABASE

name:                   DB_IO_RQ_SM
alertState:             normal
collectionTime:         2018-08-23T07:33:38-05:00
metricObjectName:       ASM1.PRODDB
metricType:             Cumulative
metricValue:            *.
objectType:             IORM_DATABASE

name:                   DB_IO_RQ_SM
alertState:             normal
collectionTime:         2018-08-22T14:27:45-05:00
metricObjectName:       ASM1.PRODDB
metricType:             Cumulative
metricValue:            *.
objectType:             IORM_DATABASE
```

```
name:                   DB_IO_RQ_SM
alertState:             normal
collectionTime:         2018-08-22T14:25:06-05:00
metricObjectName:       _OTHER_DATABASE_
metricType:             Cumulative
metricValue:            *.
objectType:             IORM_DATABASE
```

Related Topics

- ALTER CELL
- DESCRIBE CELL
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.

7.7.12.21 LIST METRICSTREAM

Purpose

The `LIST METRICSTREAM` command displays metrics in the metric stream.

Syntax

```
LIST METRICSTREAM [ name |  attribute_filters ]  [attribute_list]  [DETAIL]
```

Usage Notes

- The `LIST METRICSTREAM` command is functionally equivalent to the `LIST METRICCURRENT` command, except that `LIST METRICSTREAM` only displays metrics that are included in the metric stream.

Example 7-190 Listing METRICSTREAM Attributes

This example shows the `LIST METRICSTREAM` command with a filter to display cell disk attributes.

```
CellCLI> LIST METRICSTREAM WHERE name LIKE 'N_NIC.*'
```

Related Topics

- Real-Time Insight
  You can use the Real-Time Insight feature to enable real-time monitoring of your Exadata systems.
- Attribute Filters in LIST and ALTER Commands
  You can use the *attribute_filters* clause to specify the objects to display in `LIST` commands. Some `ALTER` commands also support the *attribute_filters* clause.
- LIST METRICCURRENT

7.7.12.22 LIST OFFLOADGROUP

Purpose

The `LIST OFFLOADGROUP` command displays the attributes for offload groups.

Syntax

```
LIST OFFLOADGROUP [name | filters] [attribute_list] [DETAIL]
```

Usage Notes

- The *name* and *filters* parameters specify the offload groups for which you want to display the attributes.
  - *name* specifies the name of an offload group.
  - *filters* specifies an expression to match one or more offload groups.
  - If neither *name* nor *filters* is specified, the command lists the attributes for all offload groups.
- The *attribute_list* parameter specifies one or more attributes for which you want to view. The *attribute_list* begins with the `ATTRIBUTES` keyword. If specifying more than one attribute, separate the attributes with a comma:

  ```
  ATTRIBUTES { attr1 [, attr2]... }
  ```

  If the *attribute_list* parameter is omitted, the command displays all attributes.
- If the `DETAIL` parameter is used, the command displays an attribute descriptor for each attribute.

Examples

Example 7-191 Displaying All Offload Groups in Detail

The following command lists all the attributes for all offload groups.

```
LIST OFFLOADGROUP DETAIL
```

Example 7-192 Displaying the Attributes for an Offload Group

The following command lists all the attributes for an offload group named "offloadgroup1".

```
LIST OFFLOADGROUP offloadgroup1
```

Example 7-193 Displaying Specific Attributes

The following command lists the name and package attributes for all offload groups.

```
LIST OFFLOADGROUP ATTRIBUTES name, package
```

Related Topics

- ALTER OFFLOADGROUP

7.7.12.23 LIST OFFLOADPACKAGE

Purpose

The `LIST OFFLOADPACKAGE` command displays the attributes for offload packages. The offload packages provide information on the offload server versions that are installed on the cell. This indirectly shows the various database versions that are supported on the cell.

Syntax

```
LIST OFFLOADPACKAGE [name | filters] [attribute_list] [DETAIL]
```

Usage Notes

- The *name* and *filters* parameters specify the offload packages for which you want to display the attributes.
  - *name* The name of the offload package to be displayed.
  - *filters* an expression which determines which offload packages should be displayed.
  - If neither *name* nor *filters* is specified, the command lists the attributes for all offload packages.
- The *attribute_list* parameter specifies one or more attributes for which you want to view. The *attribute_list* begins with the `ATTRIBUTES` keyword. If specifying more than one attribute, separate the attributes with a comma:

  ```
  ATTRIBUTES { attr1 [, attr2]... }
  ```

  If the *attribute_list* parameter is omitted, the command displays all attributes.
- If the `DETAIL` parameter is used, the command displays an attribute descriptor for each attribute.

Examples

Example 7-194 Displaying All Offload Packages in Detail

The following command lists all the attributes for all offload packages.

```
LIST OFFLOADPACKAGE DETAIL
```

Example 7-195 Displaying the Attributes for an Offload Package

The following command lists all the attributes for an offload package named 'cellofl-12.1.1.1.1_LINUX.X64_130211'.

```
LIST OFFLOADPACKAGE 'cellofl-12.1.1.1.1_LINUX.X64_130211'
```

Example 7-196 Displaying Specific Attributes

The following command lists the name and ispublic attributes for all offload packages.

```
LIST OFFLOADPACKAGE ATTRIBUTES name, ispublic
```

Example 7-197 Displaying Packages with Specific Attribute Values

The following command lists the name of all packages where the ispublic atrribute.

```
LIST OFFLOADPACKAGE where ispublic = 'true'


CellCLI> describe offloadpackage
        name
        installationTime
        isPublic


CellCLI> list offloadpackage
        cellofl-12.1.2.4.0_LINUX.X64_200526
        cellofl-21.1.0.0.0_LINUX.X64_200912
        cellofl-11.2.3.3.1_LINUX.X64_200526
```

Related Topics

- ALTER OFFLOADGROUP

7.7.12.24 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

- You can use the DESCRIBE PHYSICALDISK command to view the complete list of `PHYSICALDISK` attributes.
- When a physical disk is performing a power cycle, the status of the disk is included in the output. The status options are `Normal-DiskPoweredOn`, `Normal-DiskPoweredOff`, `ProactiveFailure-DiskPoweredOn`, and `ProactiveFailure-DiskPoweredDown`. If physical disk status shows as failed, perform a power cycle on the disk to verify the status of the physical disk.

Example 7-198 Listing Physical Disk Attributes

This example shows the `LIST` command with the `PHYSICALDISK` object, and the corresponding output.

```
CellCLI> LIST PHYSICALDISK
         20:0            K68DWJ          normal
         20:1            K7YXUJ          normal
         20:2            K7TYEJ          normal
         20:3            K7BJMJ          normal
         20:4            K5B4SM          normal
         20:5            KEBTDJ          normal
         20:6            K4URJJ          normal
         20:7            K5E1DM          normal
         20:8            K7VL6J          normal
         20:9            K7N5NJ          normal
         20:10           K7Z3KJ          normal
         20:11           K504ZM          normal
         FLASH_1_0       1030M03RK1      normal
         FLASH_1_1       1030M03RJN      normal
         FLASH_1_2       1030M03RJH      normal
         FLASH_1_3       1030M03RJD      normal
         FLASH_2_0       1027M03N6X      normal
         FLASH_2_1       1027M03NMN      normal
         FLASH_2_2       1027M03N6Y      normal
         FLASH_2_3       1027M03N6W      normal
         FLASH_4_0       1025M03EJ3      normal
         FLASH_4_1       1025M03EJ2      normal
         FLASH_4_2       1025M03EHU      normal
         FLASH_4_3       1025M03EKE      normal
         FLASH_5_0       1028M03QP8      normal
         FLASH_5_1       1028M03QNA      normal
         FLASH_5_2       1028M03QKU      normal
         FLASH_5_3       1028M03QHN      normal

CellCLI> LIST PHYSICALDISK 20:0 DETAIL
         name:                   20:0
         deviceId:               46
         deviceName:             /dev/sda
         diskType:               HardDisk
         enclosureDeviceId:      20
         errOtherCount:          0
         luns:                   0_0
         makeModel:              "HITACHI HUS1560SCSUN600G"
         physicalFirmware:       A8C0
         physicalInsertTime:     2017-07-27T07:03:00-04:00
         physicalInterface:      sas
         physicalSerial:         K68DWJ
         physicalSize:           558.9120712280273G
         slotNumber:             0
         status:                 normal

CellCLI> LIST PHYSICALDISK FLASH_5_3 DETAIL
         name:                   FLASH_5_3
         deviceName:             /dev/sdx
         diskType:               FlashDisk
         luns:                   5_3
         makeModel:              "Sun Flash Accelerator F20 PCIe Card"
         physicalFirmware:       D21Y
         physicalInsertTime:     2017-07-27T07:03:01-04:00
         physicalSerial:         1028M03QHN
         physicalSize:           22.8880615234375G
         slotNumber:             "PCI Slot: 5; FDOM: 3"
         status:                 normal
```

Related Topics

- ALTER PHYSICALDISK

- About Quoting Object Names
  If an object name has characters that cause parsing errors enclose the object name in quotes.

7.7.12.25 LIST PLUGGABLEDATABASE

Purpose

Displays the specified attributes for active pluggable databases.

Syntax

```
LIST PLUGGABLEDATABASE [name | attribute_filters] [attribute_list]  [DETAIL]
```

Usage Notes

You can use the DESCRIBE PLUGGABLEDATABASE command to view the complete list of `PLUGGABLEDATABASE` attributes.

Examples

The following example shows the `LIST PLUGGABLEDATABASE` command and the corresponding output.

Example 7-199   Listing Pluggable Database Attributes

```
CellCLI> LIST PLUGGABLEDATABASE
         PDB$SEED
         CDB1_PDB1
         NEWPDB1
         NEWPDB2

CellCLI> LIST PLUGGABLEDATABASE DETAIL
         name:                   PDB$SEED
         asmClusterName:         SALESDBS_ASMCLUSTER
         containerName:          CDB1
         flashCacheLimit:        515M
         flashCacheMin:          0
         flashCacheSize:         0
         iormLimit:              0.0
         iormShare:              2
         pdbID:                  385656752
         pmemCacheMin:           0
         pmemCacheLimit:         256M
         pmemCacheSize:          0


         name:                   CDB1_PDB1
         asmClusterName:         SALESDBS_ASMCLUSTER
         containerName:          CDB1
         flashCacheLimit:        315G
         flashCacheMin:          200G
         flashCacheSize:         0
         iormLimit:              0.0
         iormShare:              5
         pdbID:                  2850864136
         pmemCacheMin:           0
         pmemCacheLimit:         256M
         pmemCacheSize:          0


         name:                   NEWPDB1
         asmClusterName:         SALESDBS_ASMCLUSTER
         containerName:          CDB1
         flashCacheLimit:        157G
         flashCacheMin:          20G
         flashCacheSize:         0
         iormLimit:              80.0
         iormShare:              1
         pdbID:                  167491455
         pmemCacheMin:           0
         pmemCacheLimit:         256M
         pmemCacheSize:          0

         name:                   NEWPDB2
         asmClusterName:         SALESDBS_ASMCLUSTER
         containerName:          CDB1
         flashCacheLimit:        157G
         flashCacheMin:          20G
         flashCacheSize:         0
         iormLimit:              60.0
         iormShare:              2
         pdbID:                  2392787216
         pmemCacheMin:           0
         pmemCacheLimit:         256M
         pmemCacheSize:          0
```

7.7.12.26 LIST PMEMCACHE

Purpose

The `LIST PMEMCACHE` command displays the specified attributes for the PMEM cache.

> **Note:**
>
> The `LIST PMEMCACHE` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
LIST PMEMCACHE [attribute_list] [DETAIL]
```

Usage Notes

- You can use the DESCRIBE PMEMCACHE command to view the complete list of `PMEMCACHE` attributes.

- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `LIST XRMEMCACHE` instead of `LIST PMEMCACHE`.

Examples

The following example shows the `LIST` command with the `PMEMCACHE` object, and the corresponding output.

Example 7-200 Listing PMEM Cache Attributes

```
CellCLI> LIST PMEMCACHE
         dbm01celadm08_PMEMCACHE      normal

CellCLI> LIST PMEMCACHE DETAIL
         name:                    dbm01celadm08_PMEMCACHE
         cellDisk:
PM_10_dbm01celadm08,PM_11_dbm01celadm08,PM_06_dbm01celadm08,PM_01_dbm01celadm0
8,

PM_00_dbm01celadm08,PM_03_dbm01celadm08,PM_08_dbm01celadm08,PM_02_dbm01celadm0
8,

PM_09_dbm01celadm08,PM_07_dbm01celadm08,PM_04_dbm01celadm08,PM_05_dbm01celadm0
8
         creationTime:         2019-09-12T11:37:00-07:00
         degradedCelldisks:
         effectiveCacheSize:   1.474365234375T
         id:                   0ba69a26-b02d-46a8-a5a6-82e699a6ac88
         size:                 1.474365234375T
         status:               normal
```

7.7.12.27 LIST PMEMLOG

Purpose

The `LIST PMEMLOG` command displays the specified attributes for the PMEM log.

Note:

The `LIST PMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

Syntax

```
LIST PMEMLOG [attribute_list] [DETAIL]
```

Usage Notes

- You can use the DESCRIBE PMEMLOG command to view the complete list of `PMEMLOG` attributes.
- On Exadata X8M and X9M systems with Oracle Exadata System Software release 23.1.0, you can interchangeably use `LIST XRMEMLOG` instead of `LIST PMEMLOG`.

Examples

The following example shows the `LIST` command with the `PMEMLOG` object, and the corresponding output.

Example 7-201 Listing the Current PMEMLOG

The following example shows the `LIST PMEMLOG` command with the default values.

```
CellCLI> LIST PMEMLOG

         raw_PMEMLOG    normal
```

Example 7-202 Listing All PMEMLOG Attributes

```
CellCLI> LIST PMEMLOG DETAIL

         name:                    raw_PMEMLOG
         cellDisk:                NV_00_raw,NV_01_raw,NV_02_raw,NV_03_raw
         creationTime:            2019-11-23T12:34:56-05:00
         degradedCelldisks:
         effectiveSize            1G
         efficiency:              100.0
         id:                      8a0adc84-9088-4c4e-8e1c-b6bcbd5cb1ba
         size:                    1G
         status:                  normal
```

7.7.12.28 LIST QUARANTINE

Purpose

The `LIST QUARANTINE` command displays specified attributes for quarantines.

Syntax

```
LIST QUARANTINE [ name |  attribute_filters ]  [attribute_list] [DETAIL]
```

Examples

The following example shows the `LIST` command with the `QUARANTINE` object.

Example 7-203 Listing QUARANTINE Attributes

```
CellCLI> LIST QUARANTINE  DETAIL

CellCLI> LIST QUARANTINE where comment like 'added.*'
```

Related Topics

- DESCRIBE QUARANTINE

7.7.12.29 LIST ROLE

Purpose

The `LIST ROLE` command displays the specified attributes for a role.

Syntax

```
LIST ROLE [name | filters]
          [ATTRIBUTES {ALL | attr1 [,attr2]...}] [DETAIL]
```

Usage Notes

- Use *name* to display information about a specific role.
- Use an expression in place of *filters* to display information about the roles that satisfy the expression.
- Use the `ATTRIBUTES` keyword to display information about one or more attributes of the roles. The `ALL` option can be used to display all attributes.
- Use the keyword `DETAIL` to format the output as an attribute on each line, with an attribute descriptor preceding each value.

Example 7-204 Displaying Roles

This example shows how to display detailed information about all roles.

```
CellCLI> LIST ROLE DETAIL
         name:                   admin
         privileges:             object=all objects, verb=all actions,
attributes=all attributes, options=all options

         name:                   gd_monitor
         privileges:             object=griddisk, verb=0, attributes=all
attributes, options=all options
                                 object=griddisk, verb=list, attributes=all
attributes, options=all options
```

Example 7-205 Displaying Roles Using an Expression

This example shows how to display roles using an expression.

```
CellCLI> LIST ROLE WHERE name>'ad' AND name<'ba'
         admin
```

7.7.12.30 LIST SOFTWAREHISTORY

Purpose

The `LIST SOFTWAREHISTORY` command displays a list of final states for past software updates.

Syntax

```
LIST SOFTWAREHISTORY [attribute_filters] [attribute_list] [DETAIL]
```

Example 7-206   Displaying the History of a Scheduled Software Update

By default, only the update name and status are shown.

```
CellCLI> LIST SOFTWAREHISTORY
  12.2.1.2.0.170509  Last update completed at: 2017-05-20T08:00:57-07:00
  12.2.1.2.0.170520  Last update completed at: 2017-05-21T06:39:54-07:00
  12.2.1.2.0.17052   Last update completed at: 2017-06-08T08:56:45-07:00
  12.2.1.2.0.170603  Last update completed at: 2017-06-08T16:03:17-07:00
```

Example 7-207   Displaying the Detailed History of a Specific Software Update

This example shows the detailed software update history for a particular update which is referenced by its name.

```
CellCLI> LIST SOFTWAREHISTORY WHERE name='12.2.1.2.0.170808.1' DETAIL
         name:                   12.2.1.2.0.170808.1
         status:                 Upgrade failed. See alerts at:
2017-08-10T10:56:15-07:00
```

7.7.12.31 LIST SOFTWAREUPDATE

Purpose

The `LIST SOFTWAREUPDATE` command displays the status of the most recently scheduled update.

Syntax

```
LIST SOFTWAREUPDATE [attribute_list] [DETAIL]
```

Usage Notes

The possible states for the software update are:

- `Ready to update at: `*`update_time`*
- `Downloading`
- `Checking prerequisites`
- `Prerequisites failed. See alerts.`
- `Last update completed at: `*`update_time`*
- `Running`
- `Upgrade failed. See alerts.`

Example 7-208   Displaying the Status of a Scheduled Software Update

By default, only the update name and status are shown.

```
CellCLI> LIST SOFTWAREUPDATE
  12.2.1.2.0.170603   Last update completed at: 2017-06-08 16:03:17 -0700
```

Example 7-209 Displaying the Detailed Status of a Scheduled Software Update

By default, only the update name and status are shown.

```
CellCLI> LIST SOFTWAREUPDATE DETAIL
         name:                   12.2.1.2.0.170808.1
         status:                 Upgrade failed. See alerts
         store:                  https://mystore_url:4443
         time:                   2017-08-10T10:35:00-07:00
```

7.7.12.32 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

You can use the DESCRIBE THRESHOLD command to view the complete list of `THRESHOLD` attributes.

Examples

The following example shows the `LIST` command with the `THRESHOLD` object, and the corresponding output.

Example 7-210 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
```

7.7.12.33 LIST USER

Purpose

The `LIST USER` command displays the specified attributes for a user.

Syntax

```
LIST USER [name | filters] [attribute_list] [DETAIL]
```

Usage Notes

- *name* is the user name.
- *filters* is an expression that determines which users are displayed.
- attribute_list is the attributes to display. The `ALL` option can be used to display all attributes.
- The `DETAIL` option formats the output as an attribute on each line, with an attribute descriptor preceding each value.

Examples

The following example shows the `LIST USER` command.

Example 7-211   Using the LIST USER Command

```
CellCLI> LIST USER DETAIL

CellCLI> LIST USER where name like 'agarcia' DETAIL
         name:                   agarcia
         roles:                  role=gd_monitor
         Privileges:             object=griddisk
                                 verb=list
                                 attributes=all attributes
                                 options= all options
```

7.7.12.34 LIST XRMEMCACHE

Purpose

Starting with Exadata X10M, the `LIST XRMEMCACHE` command displays attributes of the Exadata RDMA Memory Cache (XRMEM cache).

> **Note:**
>
> On Exadata X8M and X9M systems, the `LIST XRMEMCACHE` command is equivalent to the `LIST PMEMCACHE` command.

Syntax

```
LIST XRMEMCACHE [attribute_list] [DETAIL]
```

Usage Notes

You can use the DESCRIBE XRMEMCACHE command to view the complete list of `XRMEMCACHE` attributes.

Examples

The following example shows the `LIST` command with the `XRMEMCACHE` object, and the corresponding output.

Example 7-212 Listing XRMEM Cache Attributes

```
CellCLI> LIST XRMEMCACHE
         dbm01celadm08_XRMEMCACHE

CellCLI> LIST XRMEMCACHE DETAIL
         name:                   dbm01celadm08_XRMEMCACHE
         effectiveCacheSize:     1.25T
         creationTime:           2022-08-09T23:54:25-07:00
```

7.7.12.35 LIST XRMEMCACHECONTENT

Purpose

Starting with Exadata X10M and Oracle Exadata System Software release 24.1.0, the `LIST XRMEMCACHECONTENT` command displays information about the contents of the Exadata RDMA Memory Cache.

Syntax

```
LIST XRMEMCACHECONTENT [attribute_filters] [attribute_list] [DETAIL]
```

Usage Notes

You can use the LIST XRMEMCACHECONTENT command to view the complete list of `XRMEMCACHECONTENT` attributes.

Examples

Example 7-213 Listing XRMEM Cache Content Attributes

This example shows the `LIST` command with the `XRMEMCACHECONTENT` object, and a corresponding output sample.

```
  CellCLI> LIST XRMEMCACHECONTENT DETAIL

         cachedKeepSize:         0
         cachedSize:             13330546688
         cachedWriteSize:        0
         clusterName:
         columnarCacheSize:      0
         columnarKeepSize:       0
         dbID:                   0
         dbUniqueName:           UNKNOWN
         hitCount:               0
         missCount:              0
         objectNumber:           0
         tableSpaceNumber:       0
         vaultID:                0
```

7.7.12.36 LIST XRMEMLOG

Purpose

Starting with Oracle Exadata System Software release 23.1.0, the `LIST XRMEMLOG` command is equivalent to the `LIST PMEMLOG` command.

Note:

The `LIST XRMEMLOG` command can only be used on Exadata X8M and X9M storage server models.

7.7.13 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.

7.7.14 REVOKE

Purpose

The `REVOKE` command removes privileges and roles.

Syntax

```
REVOKE object_type [name] FROM sub_object_type [sub_object_name]
```

Usage Notes

- *object_type* can be as follows:
  - `PRIVILEGE`
  - `ROLE`
- The following can be used for `PRIVILEGE` object type:
  - *name* is in the following format:

```
{ ALL ACTIONS | action } ON { ALL OBJECTS | object }  { ALL ATTRIBUTES | \
ATTRIBUTES attribute1 [, attribute2, ...] }  { WITH ALL OPTIONS |        \
WITH OPTIONS option1 [, option2, ...] }
```

  - The *sub_object_type* must be `ROLE`.
  - The *sub_object_name* is the name of the role.
- The following can be used for the `ROLE` object type:
  - *name* is the role name.
  - The *sub_object_type* must be `USER`.
  - The *sub_object_name* is the name of the user.
- REVOKE PRIVILEGE
- REVOKE ROLE

Related Topics

- REVOKE PRIVILEGE
- REVOKE ROLE

7.7.14.1 REVOKE PRIVILEGE

Purpose

The `REVOKE PRIVILEGE` command revokes privileges from a role.

Syntax

```
REVOKE PRIVILEGE { ALL ACTIONS | action } ON { ALL OBJECTS | object } { ALL
ATTRIBUTES | ATTRIBUTES attribute1 [, attribute2...] } { WITH ALL OPTIONS |
WITH OPTIONS option1 [, option2, ...] } FROM ROLE { ALL | role1 [,
role2, ...] }
```

Usage Notes

- *action* is the command.
- *object* is object type for the action.
- *attribute* are the attributes for the object.
- *option* are the options for the object.
- *role* is the name of the role from which to revoke privileges.
- The `ALL ACTIONS` argument revokes privileges for all actions.
- The `ALL OBJECTS` argument revokes privileges for all objects.
- The `ALL ATTRIBUTES` argument revokes privileges for all attributes.
- The `WITH ALL OPTIONS` argument revokes privileges for all options.

Examples

The following example shows the `REVOKE PRIVILEGE` command.

Example 7-214 Revoking a Privilege

```
CellCLI> REVOKE PRIVILEGE ALL ACTIONS ON ALL OBJECTS ALL ATTRIBUTES -
         WITH ALL OPTIONS FROM ROLE ALL

CellCLI> REVOKE PRIVILEGE list on griddisk ATTRIBUTES name,size     -
         WITH OPTIONS detail FROM ROLE gd_monitor
```

7.7.14.2 REVOKE ROLE

Purpose

The `REVOKE ROLE` command revokes the role for a user.

Syntax

```
REVOKE ROLE { ALL | role1 [, role2, ...] } FROM USER { ALL | user1 [,
user2...] }
```

Usage Notes

- *role* is the name of a role.
- *user* is the name of a user.
- The `ALL` argument revokes all roles from the user.
- The `FROM USER ALL` argument revokes the role from all users.

Examples

The following example shows how to revoke a role from a user.

Example 7-215 Revoking a Role From a User

```
CellCLI> REVOKE ROLE gd_monitor FROM USER jdoe
```

7.7.15 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 7-216 Setting the Date Format with the SET Command

This example shows an example of the `SET` command.

```
SET DATEFORMAT STANDARD
```

7.7.16 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 the following table.

Table 7-4 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.

7.7.17 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 `@` both require the option *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, the commands in Example 7-3 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.

8

Using the dcli Utility

The dcli utility facilitates centralized management across Oracle Exadata Database Machine.

dcli automates the execution of operating system commands on a set of servers and returns the output to the centralized management location where the dcli utility was run.

- Overview of the dcli Utility
  The dcli utility runs operating system commands on multiple servers in parallel threads. The target servers can be Exadata Storage Servers or Exadata database servers. However, the dcli utility does not support an interactive session with a remote application.
- dcli Syntax
  This topic describes the syntax for the dcli utility.
- dcli Examples
  This section contains examples of the using the dcli utility.
- Setting Up SSH User-Equivalence on Oracle Exadata Storage Server
  Setting user-equivalence enables you to issue commands to remote cells without having to enter the password for the cell.

8.1 Overview of the dcli Utility

The dcli utility runs operating system commands on multiple servers in parallel threads. The target servers can be Exadata Storage Servers or Exadata database servers. However, the dcli utility does not support an interactive session with a remote application.

You can run the dcli utility on an Exadata Storage Server or Exadata database server, or you can copy the utility to another host computer that acts as a central management server. You can issue a command to be run on multiple servers, or use files that can be copied to servers and then run. The servers are referenced by their host name or IP address.

Starting with Oracle Exadata System Software release 24.1.0, the dcli utility requires Python version 2.5 or later. Previous releases require Python version 2.3 or later. You can determine the version of Python by running the `python -V` command. In addition, the utility requires prior setup of SSH user-equivalence for connection to the target servers. You can use the dcli utility initially with the `-k` option to set up SSH user-equivalence. Also, you can manually set up SSH user-equivalence to servers.

Command output (`stdout` and `stderr`) is collected and displayed after the copy and command execution is finished on the specified servers. The dcli options allow command output to be abbreviated to minimize non-error output, such as messages showing normal status.

Related Topics

- Setting Up SSH User-Equivalence on Oracle Exadata Storage Server
  Setting user-equivalence enables you to issue commands to remote cells without having to enter the password for the cell.

8.2 dcli Syntax

This topic describes the syntax for the dcli utility.

Syntax

```
dcli [options] [command]
```

Command Arguments

- *options*: command options
- *command:* Any command that can be run from an operating system prompt.

Command Options

Table 8-1 dcli Options

Option	Description
`--batchsize=`*`MAXTHDS`*	Limits the number of parallel execution threads, which limits the number of target servers on which to run the operation in parallel.
`-c` *`CELLS`*	Specifies a comma-delimited list of target servers to which commands are sent.
`--ctimeout=`*`CTIMEOUT`*	Specifies the maximum time in seconds for initial connection to a target server.
`-d` *`DESTFILE`*	Specifies the target destination directory or file on remote servers to be used when copying files or directories using the `-f` option.
`-f` *`FILE`*	Specifies the files or file template to be copied to the servers. 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 server.
`-g` *`GROUPFILE`*	Specifies a file containing a list of target servers to which commands are sent. The servers can be identified by host names or IP addresses.
`-h`, `--help`	Displays help text and then exits.
`--hidestderr`	Hide standard error messages (STDERR) for commands run remotely using SSH.
`-k`	Sets up SSH user-equivalence for the current user to the servers specified with the `-c` or `-g` option by appending public key files to the `authorized_keys` file on servers.
`--key-with-one-password`	This option simplifies SSH user-equivalence setup by prompting once for the user password and using the specified password for all the configured servers. If not specified, then a password prompt occurs for each server.
`-l` *`USERID`*	Identifies the user to log in as on remote servers. The default is the `celladmin` user.

Table 8-1 (Cont.) dcli Options

Option	Description
`--root-exadatatmp`	This option uses the `root` user, similar to `-l root`. However, instead of using `/root` as the default working directory, this option uses `/var/log/exadatatmp`, which avoids alerts generated by Advanced Intrusion Detection Environment (AIDE) for files written to `/root`.
`--maxlines=`*`MAXLINES`*	Limits output from each target server to the specified number of lines. By default, the output limit from each target server is 100000 lines.
`-n`	Abbreviates nonerror output. Servers that return normal output (return code of `0`) only have the server 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 servers 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 Oracle Exadata Storage Servers.
`--showbanner`, `--sh`	Show the banner of the remote node when using SSH.
`-t`	Displays the target servers 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 Oracle Exadata Storage Servers.
`-v`	Prints the verbose version of messages to `stdout`.
`--version`	Shows the version number of the program and then exits.
`--vmstat=`*`VMSTATOPS`*	Runs the `vmstat` utility on the target servers with the specified command options.
`-x` *`EXECFILE`*	Specifies the command file to be copied and run on the servers. 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 server. The file is copied to the default home directory of the user on the target server.

Usage Notes

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.

Troubleshooting

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 servers.
- 1: One or more servers could not be reached or remote execution returned a nonzero status.
- 2: A local error prevented any command execution.

If any servers are down or do not respond, then a message is written to `stderr` listing the unresponsive servers. The operations continue on the other servers, and the return code after completion is 1.

8.3 dcli Examples

This section contains examples of the using the dcli utility.

- Using dcli to Set up SSH User-equivalence for a Current User
  This example shows how to set SSH user-equivalence for a current user using the `-k` option.
- Using dcli with the -n Option
  This example shows how to run the CellCLI command `ALTER IORMPLAN OBJECTIVE='basic'`, and abbreviates non-error output.
- Using dcli with the -r Option
  This example shows how to run the CellCLI command `LIST GRIDDISK`, and deletes the lines in the output that contain `normal`.
- Using dcli with the -v Option
  This example shows how to use the verbose (`-v`) option with SSH.
- Using dcli with the -t Option
  This example shows how to use the `-t` option to list target cells.
- Using dcli with the -f Option
  This example shows how to use the `-f` option.
- Using dcli with the --vmstat Option
  This example shows how to use the `--vmstat` option of dcli.

- Using dcli with the --hidestderr Option
  This example shows how to use the `--hidestderr` option.
- Using dcli with the --showbanner Option
  This example shows how to use the `--showbanner` option.
- Using dcli to Change an IORM Plan
  This example shows a CellCLI command that changes the IORMPLAN to active on the target cells in the `mycells` group file.
- Using dcli with a Script
  This example shows the CellCLI commands in the `reConfig.scl` file on the target cells.
- Using dcli to List Grid Disk Status
  This example shows how to run a CellCLI command that lists the name and status of grid disks on the target cells.
- Using dcli to List Alert History Information
  This example shows a CellCLI command that lists alert history name, examined by, severity on the target cells.
- Using dcli to List Alert History Where examinedby is Not Set
  This example shows a CellCLI command that lists alert history where examined by has not been set on the target cells.
- Using dcli to List Current Metric Alert State
  This example shows a CellCLI command that retrieves metric current objects for the number of MB read in large blocks on a grid disk for a group of cells.
- Using dcli to List Specific Metric Current Objects in a Group
  This example shows a CellCLI command that retrieves metric current objects for the number of requests to read or write blocks on a grid disk.
- Using dcli to List Specific Metric Current Objects
  This example shows a CellCLI command that lists metric current objects with name equal to `cl_put` (cell CPU utilization) on the target cells.
- Using dcli to List Physical Disks
  This example shows a CellCLI command that lists physical disks where status is not equal to normal on the target cells.
- Using dcli to List Cell Disks with Free Space
  This example shows a CellCLI command that lists cell disks where free space is less than 100 MB on the target cells.
- Using dcli to View Alert History
  This example shows a CellCLI command to view the alert history from a particular period.

8.3.1 Using dcli to Set up SSH User-equivalence for a Current User

This example shows how to set SSH user-equivalence for a current user using the `-k` option.

Example 8-1 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.

8.3.2 Using dcli with the -n Option

This example shows how to run the CellCLI command `ALTER IORMPLAN OBJECTIVE='basic'`, and abbreviates non-error output.

Example 8-2 Using the -n Option

```
$ ./dcli -g mycells -l celladmin -n "cellcli -e alter iormplan \
objective=\'basic\'"
```

The abbreviated output would be similar to the following:

```
OK: ['abcd2x3']
stsd2s2:
stsd2s2: CELL-02619: Current IORMPLAN state is not 'active'.
```

8.3.3 Using dcli with the -r Option

This example 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 8-3 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', 'abcd2x3']
abcd2x3: CD_06_abcd2x3    importRequired
```

8.3.4 Using dcli with the -v Option

This example shows how to use the verbose (`-v`) option with SSH.

Example 8-4 Using the -v Option

```
$ ./dcli -s "-v" -c mycell date
```

8.3.5 Using dcli with the -t Option

This example shows how to use the `-t` option to list target cells.

The `-t` option should be used with `-c` or `-g` option.

Example 8-5   Using the -t Option

```
$ ./dcli -t -c exa01celadm09 date
Target cells: ['exa01celadm09']
exa01celadm09: Fri Jul 17 15:37:31 PDT 2019
```

8.3.6 Using dcli with the -f Option

This example shows how to use the `-f` option.

Example 8-6   Using the -f Option

In this example, a set of files is copied to the storage servers listed in the `mycells` file.

```
dcli -g mycells -f '*.bin'
```

8.3.7 Using dcli with the --vmstat Option

This example shows how to use the `--vmstat` option of dcli.

Example 8-7   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
abcd2x1: 2  0      0  22656 178512 792272    0     0     1    21     7     2
2
abcd2x2: 0  0 452304  21432 108760 867712    0     0   178   269     2     0
2
abcd2x3: 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
```

8.3.8 Using dcli with the --hidestderr Option

This example shows how to use the `--hidestderr` option.

This option can only be used when SSH is used for remotely run commands.

Example 8-8   Using the --hidestderr Option

This first command does not use the `--hidestderr` option, so the errors are shown.

```
$ ./dcli -l root -g cell "ls -1 unknown_file; cellcli -e list cell"

exam08cel01: ls: unknown_file: No such file or directory
exam08cel01: exam08cel01         online
```

```
exam08cel02: ls: unknown_file: No such file or directory
exam08cel02: exam08cel02         online
```

This second command uses the `--hidestderr` option, so the errors are not shown.

```
$ ./dcli -l root -g cell --hidestderr -l root "ls -l unknown_file;\
cellcli -e list cell"

exam08cel01: exam08cel01         online
exam08cel02: exam08cel02         online
```

8.3.9 Using dcli with the --showbanner Option

This example shows how to use the `--showbanner` option.

The banner of the remote cell replaces `******BANNER******` shown in the example.

Example 8-9 Using the --showbanner Option

```
$ ./dcli --showbanner -l root -g cell "cellcli -e list cell"
exam08cel01: ******BANNER******
exam08cel01:
exam08cel01: ******BANNER******
exam08cel01: exam08cel01         online
exam08cel02: ******BANNER******
exam08cel02:
exam08cel02: ******BANNER******
exam08cel02: exam08cel02         online
```

8.3.10 Using dcli to Change an IORM Plan

This example shows a CellCLI command that changes the IORMPLAN to active on the target cells in the `mycells` group file.

The `-t` option displays the cells that are in the `mycells` group file.

Example 8-10 Using dcli to Change an IORM Plan

```
$ ./dcli -g mycells -l root -t "cellcli -e alter iormplan active"
```

8.3.11 Using dcli with a Script

This example shows the CellCLI commands in the `reConfig.scl` file on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-11 Using dcli with a Script

```
$ ./dcli -g mycells -x reConfig.scl
```

8.3.12 Using dcli to List Grid Disk Status

This example shows how to run a CellCLI command that lists the name and status of grid disks on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user. Output lines that contain `active` as the status are deleted.

Example 8-12 Using dcli to List Grid Disk Status

```
$ ./dcli -r '.*active' -g mycells "cellcli -e list griddisk"
```

8.3.13 Using dcli to List Alert History Information

This example shows a CellCLI command that lists alert history name, examined by, severity on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user. Output lines that contain `clear` for severity are deleted.

Example 8-13 Using dcli to List Alert History Information

```
$ ./dcli -r '.*clear' -g mycells \
   "cellcli -e list alerthistory attributes name, examinedby, severity"
```

8.3.14 Using dcli to List Alert History Where examinedby is Not Set

This example shows a CellCLI command that lists alert history where examined by has not been set on the target cells.

The target cells are contained in the `allcells` group file. The command is run as the default `celladmin` user.

Example 8-14 Using dcli to List Alert History where examinedby is not Set

```
$ ./dcli -g allcells -l celladmin \
   "cellcli -e list alerthistory where examinedby=\'\' "
```

8.3.15 Using dcli to List Current Metric Alert State

This example shows a CellCLI command that retrieves metric current objects for the number of MB read in large blocks on a grid disk for a group of cells.

The command lists the current metric alert state and metric value for the metric `GD_IO_BY_R_LG` on the target cells. The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-15 Using dcli to List Current Metric Alert State

```
$ ./dcli -g mycells "cellcli -e list metriccurrent GD_IO_BY_R_LG \
  attributes alertstate, metricvalue"
```

8.3.16 Using dcli to List Specific Metric Current Objects in a Group

This example shows a CellCLI command that retrieves metric current objects for the number of requests to read or write blocks on a grid disk.

The CellCLI command lists metric current objects for names that begin with `GD_IO_RQ` on the target cells. The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-16   Using dcli to List Specific Metric Current Objects in a Group

```
$ ./dcli -g mycells "cellcli -e list metriccurrent where name like
\'GD_IO_RQ.*\'"
```

8.3.17 Using dcli to List Specific Metric Current Objects

This example shows a CellCLI command that lists metric current objects with name equal to `cl_put` (cell CPU utilization) on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-17   Using dcli to List Specific Metric Current Objects

```
$ ./dcli -g mycells "cellcli -e list metriccurrent cl_cput"
```

8.3.18 Using dcli to List Physical Disks

This example shows a CellCLI command that lists physical disks where status is not equal to normal on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-18   Using dcli to List Physical Disks

```
$ ./dcli -g allcells "cellcli -e list physicaldisk where status not = normal"
```

8.3.19 Using dcli to List Cell Disks with Free Space

This example shows a CellCLI command that lists cell disks where free space is less than 100 MB on the target cells.

The target cells are contained in the `mycells` group file. The command is run as the default `celladmin` user.

Example 8-19   Using dcli to List Cell Disks with Free Space

In this example, the backslash (\) is an escape character that allows the greater than character (>) to be passed to the CellCLI utility without being interpreted by the remote shell.

```
$ ./dcli -g allcells "cellcli -e list celldisk where freespace \> 100M"
```

8.3.20 Using dcli to View Alert History

This example shows a CellCLI command to view the alert history from a particular period.

Example 8-20 Using dcli to View Alert History

In this 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.

```
dcli -g lab.cells "cellcli -e  list alerthistory where begintime \> \'Aug 4,
2009 12:06:38 PM\'"
```

8.4 Setting Up SSH User-Equivalence on Oracle Exadata Storage Server

Setting user-equivalence enables you to issue commands to remote cells without having to enter the password for the cell.

- To set up SSH user-equivalence for use with the `dcli` utility, use the `-k` option.

  See Using dcli to Set up SSH User-equivalence for a Current User.

9 Setting up Oracle Exadata Storage Snapshots

9.1 Before You Begin With Exadata Snapshots

Understand the optimal use-case before you begin with Exadata Snapshots.

In addition to production database workloads, many customers use Oracle Exadata for non-production workloads, including application development, testing, quality assurance, and other uses.

Exadata Snapshots are ideal for creating space-efficient read-only or read-write clones of an Oracle database that you can use for development, testing, or other non-production purposes. Exadata Snapshots are particularly useful for non-production databases that leverage Exadata performance and availability features. For example, application functionality testing that validates the use of Exadata Smart Scan features.

However, before you begin with Exadata Snapshots, be aware that some non-production database needs may be better satisfied using alternative technologies or approaches. Consider the following in conjunction with your business needs:

- If you require full end-to-end performance and high availability (HA) testing, Oracle recommends a matching test environment that mirrors the production environment. This is the only solution to fully evaluate performance gains or regressions in response to hardware, software, database, or application changes.

- If your requirements emphasize dynamic snapshot capabilities without requiring Exadata smart storage features, then consider using Oracle Advanced Cluster File System (Oracle ACFS) snapshots on Exadata.

  For information about Oracle ACFS snapshots, see the following related links.

Related Topics

- About Oracle ACFS Snapshots
- Oracle ACFS Snapshot Use Cases on Exadata (Doc ID 2761360.1)

9.2 Overview of Exadata Snapshots

Traditionally, to clone databases in a production system, you would create test master and snapshot databases on a non-Exadata system (Figure 9-1). In some cases, these databases are a full copy that consumes as much storage as its source (Figure 9-2).

Figure 9-1 Traditional Clone to a Non-Exadata System

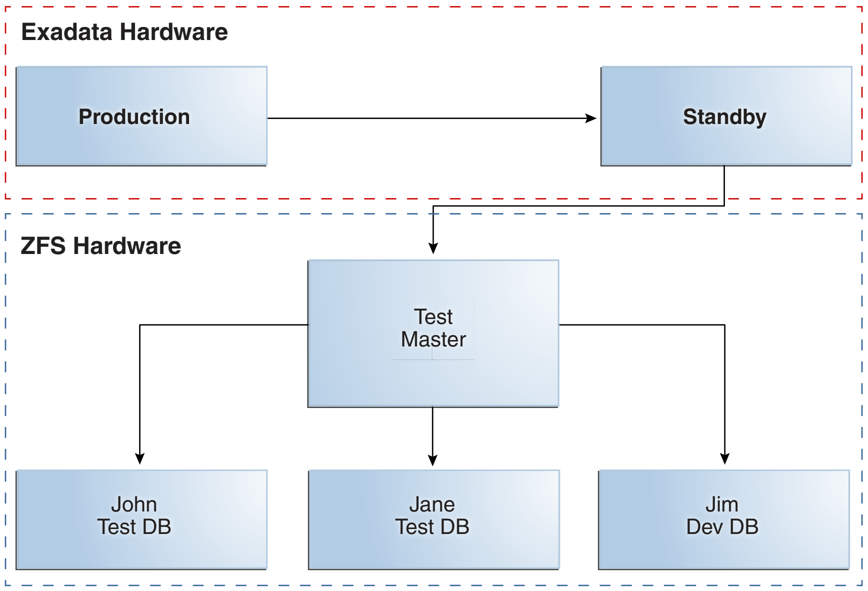
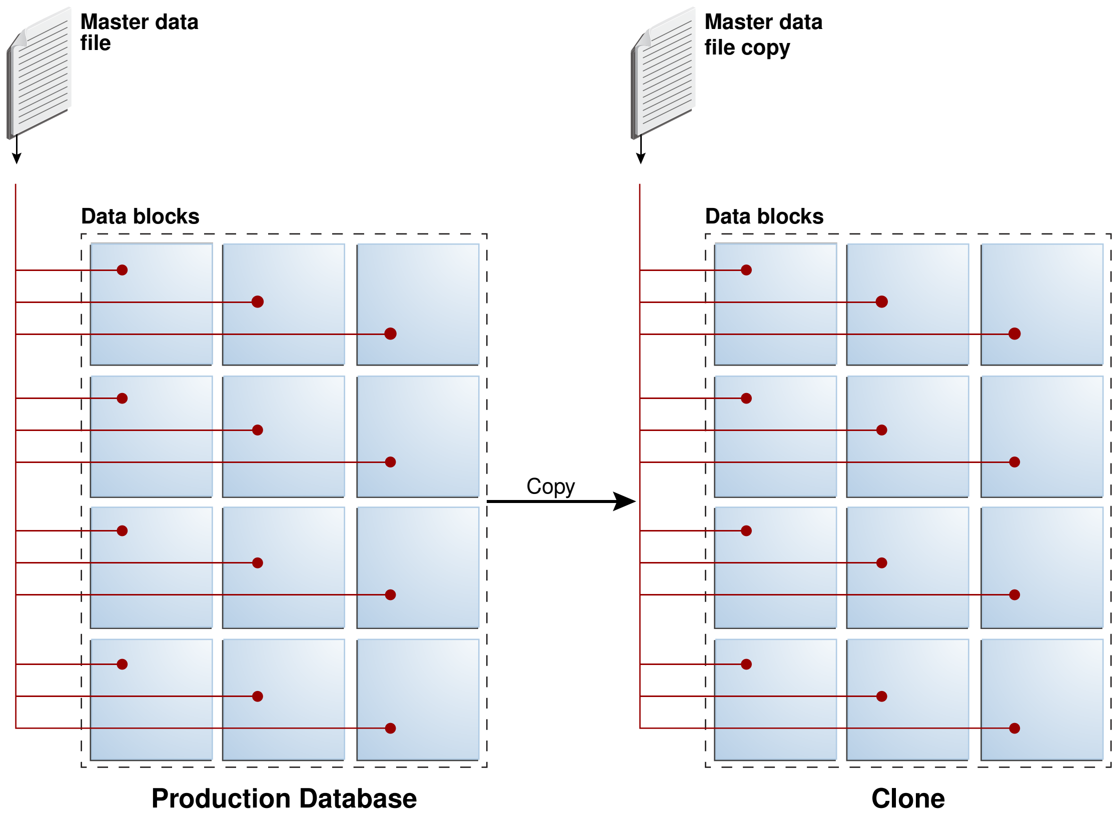


Figure 9-2 Database Clone That Is a Full Copy of Its Source



If the clones are a full copy of the production database, this is expensive in terms of the amount of storage consumed and the time it takes to create the clones. Imagine creating ten clones for a multi-terabyte database and it is easy to see why this approach does not scale.

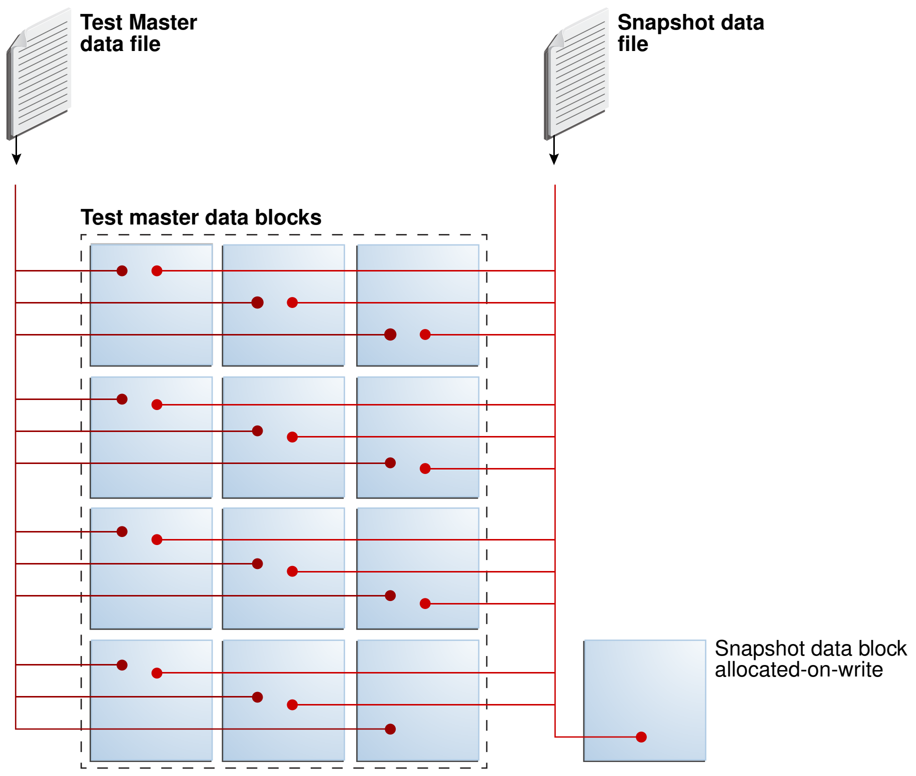
Another drawback to this approach is that Oracle Exadata System Software features such as Smart Scan, Smart Logging, and Smart Flash are not available on non-Exadata systems.

To solve these problems, you can use *Exadata Snapshots*. Exadata Snapshots are ideal for creating space-efficient read-only or read-write clones of an Oracle database that you can use for development, testing, or other non-production purposes, and when multiple clones are required because of disk space and time savings. The following image depicts the space required for an Exadata Snapshot.

Note:

Exadata Snapshots should be used only for development and testing purposes. They should not be used in production environments.

Figure 9-3   An Exadata Snapshot



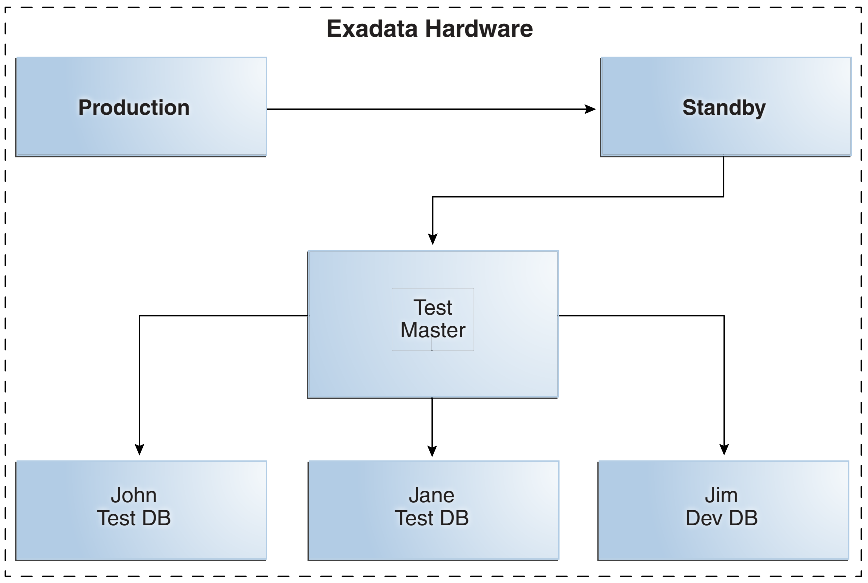
An Exadata Snapshot is based on a *test master*, which is a full clone of the source database. The test master is the only full copy of the source database. From a single test master you can create multiple Exadata Snapshots with minimal additional storage and minimal effort. Each Exadata Snapshot uses a small fraction of the disk space required for the test master and can be created or dropped in seconds. Each Exadata Snapshot is a logical copy of the test master.

Before creating Exadata Snapshots from the test master, you can modify the data in the test master, if required. For example, you can delete or mask sensitive data in the test master before making it available to non-privileged users.

Creating an Exadata Snapshot from the test master is as simple as recording the parent file name in the child file header, an operation that completes in seconds and requires minimal disk space. Additional disk space is consumed only when the user of the snapshot begins to change data. Only new data is written to data blocks that are allocated to the snapshot on write. All requests for data that has not changed are serviced by the data blocks of the test master.

Multiple users can create independent snapshots from the same test master. This enables multiple development and test environments to share space while maintaining independent databases for each user. The following image shows an Exadata environment with three Exadata Snapshots that use the same test master.

Figure 9-4 An Exadata Environment with 3 Exadata Snapshots from the Same Test Master



Hierarchical and Read-Write Snapshots

Oracle Exadata System Software release 12.2.1.1.0 introduces hierarchical and read-write snapshots. Hierarchical snapshots enable you to create snapshots from snapshots. You might want to do this if you are working on your snapshot and wish to save a copy before you make additional changes to it. In hierarchical snapshots, you can make a snapshot at any level descended from the test master. Exadata Snapshots are writable. A snapshot points to the parent's blocks for the data. If you edit a snapshot, then the snapshot will point to the new data. For the unchanged data, it will point to the parent's blocks.

If you have taken a snapshot from a snapshot, and you edit the parent snapshot, then you have to delete all snapshots that are dependent on that snapshot.

- Exadata Snapshot Concepts
  There are various object you need when creating and using Exadata Snapshots.
- Exadata Snapshot Support of Exadata Features
  In addition to space and time savings, Exadata Snapshots provide cost-effective development, quality assurance and test environments on Oracle Exadata.
- Separate Test/Development and Production Environments
  Oracle recommends that test and development environments be hosted on a separate physical Oracle Exadata Rack from the rack hosting the production database.
- Types of Exadata Snapshots
  You can create two types of Exadata Snapshots, depending on the current setup of your environment.
- Hierarchical Snapshot Databases
  Hierarchical snapshots enable you to create snapshot databases from other snapshot databases.
- Sparse Test Masters
  With the introduction of hierarchical snapshots, you can now create sparse test masters.

9.2.1 Exadata Snapshot Concepts

There are various object you need when creating and using Exadata Snapshots.

- Sparse Grid Disks
  A sparse grid disk has a virtual size attribute as well as physical size.
- Sparse Disk Groups
  Exadata Snapshots utilize Oracle ASM sparse disk groups.
- Sparse Database and Sparse Files
  In a sparse database, such as an Exadata Snapshot database, its data files are sparse files.

9.2.1.1 Sparse Grid Disks

A sparse grid disk has a virtual size attribute as well as physical size.

A sparse Oracle ASM disk group is composed of sparse grid disks.

The maximum physical size that can be allocated to sparse grid disks from a single cell disk is 4 TB. The maximum allowed virtual size is 100 TB.

You create sparse grid disks before creating a sparse Oracle ASM disk group. If you are deploying a new system or re-deploying an existing system, set aside some space from each cell disk for use by the disk group just as you would for any other disk group.

If you want to use an existing system without reimaging, see My Oracle Support note 1467056.1 for instructions on resizing existing grid disks.

Related Topics

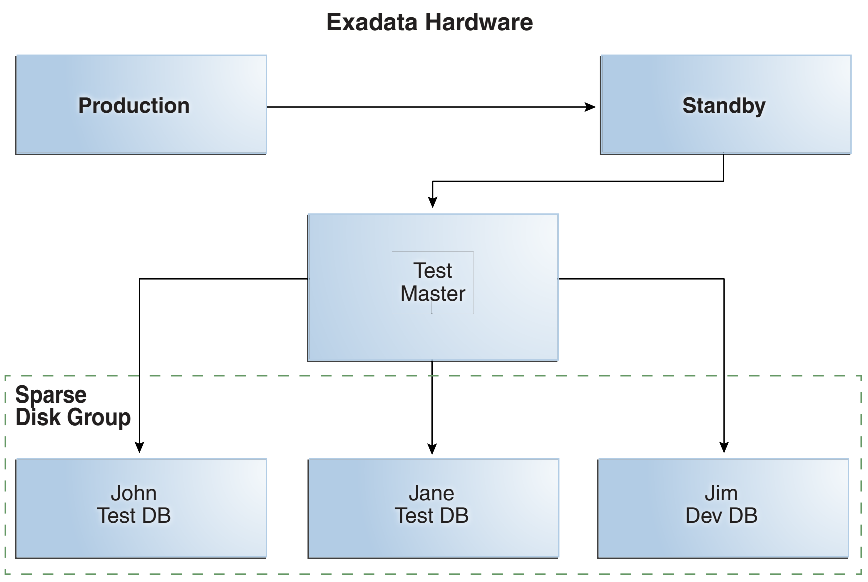
- Creating Sparse Grid Disks
- Resizing Grid Disks in Exadata: Examples (My Oracle Support Doc ID 1467056.1)

9.2.1.2 Sparse Disk Groups

Exadata Snapshots utilize Oracle ASM sparse disk groups.

Sparse data files can be created only in Oracle Automatic Storage Management (Oracle ASM) sparse disk groups. The following figure shows a sparse disk group containing three Exadata Snapshots.

Figure 9-5   Sparse Disk Group Contains Exadata Snapshots



A sparse disk group has the following attributes:

- `compatible.asm` must be set to `12.1.0.2` or higher.
- `compatible.rdbms` must be set to `12.1.0.2` or higher.
- `cell.smart_scan_capable` must be set to `true`.
- `cell.sparse_dg` must be set to `allsparse`. This attribute identifies the disk group to Oracle ASM as being made up of sparse grid disks.
- As is the case for all Oracle ASM disk groups on Exadata, the recommended allocation unit (AU) size is `4M`.

For example, the following SQL command creates a sparse disk group:

```
SQL> CREATE DISKGROUP SPARSE
  NORMAL REDUNDANCY
  DISK 'o/*/SPARSE_*'
  ATTRIBUTE
    'compatible.asm'          = '12.1.0.2',
    'compatible.rdbms'        = '12.1.0.2',
    'cell.smart_scan_capable' = 'true',
    'cell.sparse_dg'          = 'allsparse',
    'au_size'                 = '4M';
```

A sparse Oracle ASM disk group can store both sparse and non-sparse files, which makes it possible to store a test master database and Exadata Snapshots in the same disk group. However, because sparse grid disks are limited in size (to a maximum of 4 TB on a cell disk) and because there is no benefit in placing non-sparse files in a sparse disk group, you should reserve space in sparse disk groups for the sparse files associated with Exadata Snapshots and place the test master database in a regular (non-sparse) disk group.

9.2.1.3 Sparse Database and Sparse Files

In a sparse database, such as an Exadata Snapshot database, its data files are sparse files.

A database consists of the following files:

- control files
- online redo logs
- temp files
- data files

A **sparse file** contains only changes made to blocks from the parent file (the parent file remains unchanged) and maintains a pointer to the parent file for access to unchanged data.

The Exadata Snapshot has its own copy of the other database files (control files, online redo logs, and temp files). These other database files are not sparse files.

Related Topics

- Overview of Exadata Snapshots

9.2.2 Exadata Snapshot Support of Exadata Features

In addition to space and time savings, Exadata Snapshots provide cost-effective development, quality assurance and test environments on Oracle Exadata.

Exadata Snapshots can be used by developers and testers who need to validate functionality. You can also use Exadata Snapshot to practice maintenance and operational steps in a fully functional Exadata environment (for example, Exadata Smart Flash Cache, Exadata Smart Scan Offload, and Exadata Hybrid Columnar Compression).

9.2.3 Separate Test/Development and Production Environments

Oracle recommends that test and development environments be hosted on a separate physical Oracle Exadata Rack from the rack hosting the production database.

An Oracle Exadata system dedicated to development and test is ideal. Test masters and their associated Exadata Snapshots would be hosted on this system. Alternatively it may be an Oracle Exadata system that hosts Oracle Data Guard standby databases for high availability, disaster recovery, or other purposes as permitted by capacity. Test masters and their snapshots may reside in either physical or virtual machines on an Oracle Exadata system.

Related Topics

- Types of Exadata Snapshots
  You can create two types of Exadata Snapshots, depending on the current setup of your environment.

9.2.4 Types of Exadata Snapshots

You can create two types of Exadata Snapshots, depending on the current setup of your environment.

- **Using a test master from a pluggable database (PDB).**

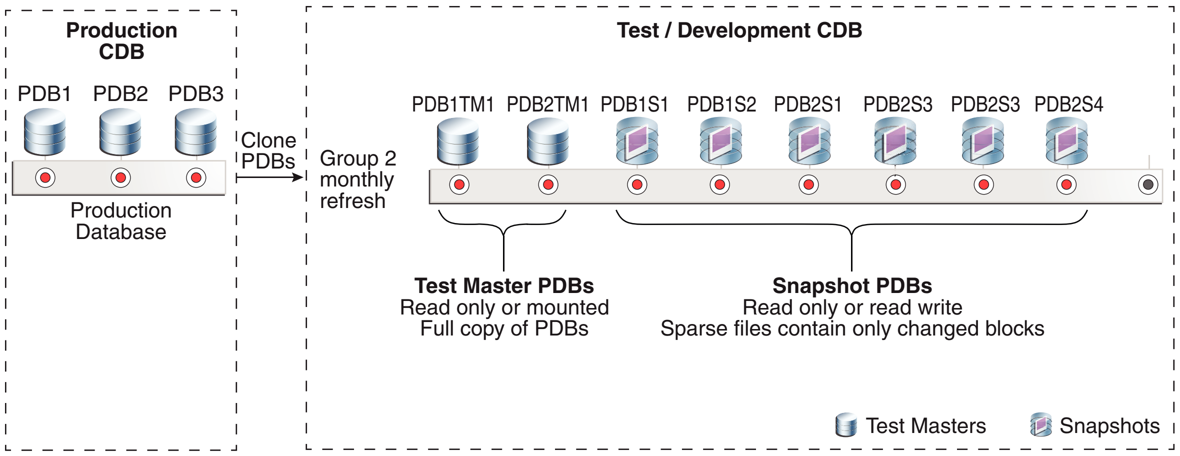
You can create **Exadata Snapshot PDBs** from individual PDBs within a container database (CDB). You can clone individual PDBs to create the test master PDB, from which Exadata Snapshot PDBs are generated.

You can move Exadata Snapshot PDBs from one CDB to another in the same Exadata cluster. You can also create an Exadata Snapshot PDB in one container from a test master PDB residing in another container as long as both CDBs are in the same Exadata cluster.

The test master database and their Exadata Snapshots must be in the same Oracle Automatic Storage Management (Oracle ASM) cluster environment.

The following image shows a production CDB with three PDBs. Two of the PDBs (PDB1 and PDB2) have been cloned to create test master PDBs, which were then unplugged and plugged into another CDB on the test Oracle Exadata. In this figure, six Exadata Snapshot PDBs have been created from the test master PDBs.

Figure 9-6 Exadata Snapshot PDBs



- **Using a test master from a whole database (CDB or non-CDB)**

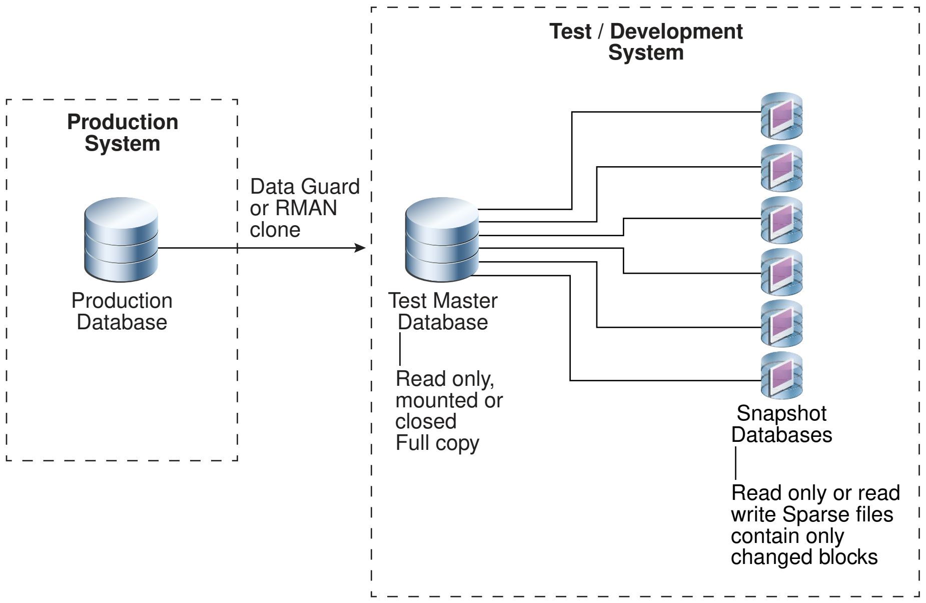
  An *Exadata Snapshot database* can be a snapshot of a complete container database (CDB) or non-container database (non-CDB). A snapshot of a complete CDB includes all of the PDBs in the CDB.

  > **Note:**
  >
  > To create a snapshot of a complete CDB, you must apply Patch 32233739 to the Oracle home directory that supports the Exadata Snapshot database.

  The next figure depicts a full clone of the production database. The clone, which is the test master database, is hosted on a separate test/development system, and it is associated with six Exadata Snapshots. The test master database is created using either an Oracle Data Guard standby database (recommended if the test master will be refreshed on a regular basis) or Oracle Recovery Manager (RMAN).

  The test master database and their Exadata Snapshots must be in the same Oracle ASM cluster environment.

Figure 9-7 Exadata Snapshot Databases


Production System
Data Guard or RMAN clone
Production Database
Test / Development System
Test Master Database
Read only, mounted or closed Full copy
Snapshot Databases
Read only or read write Sparse files contain only changed blocks

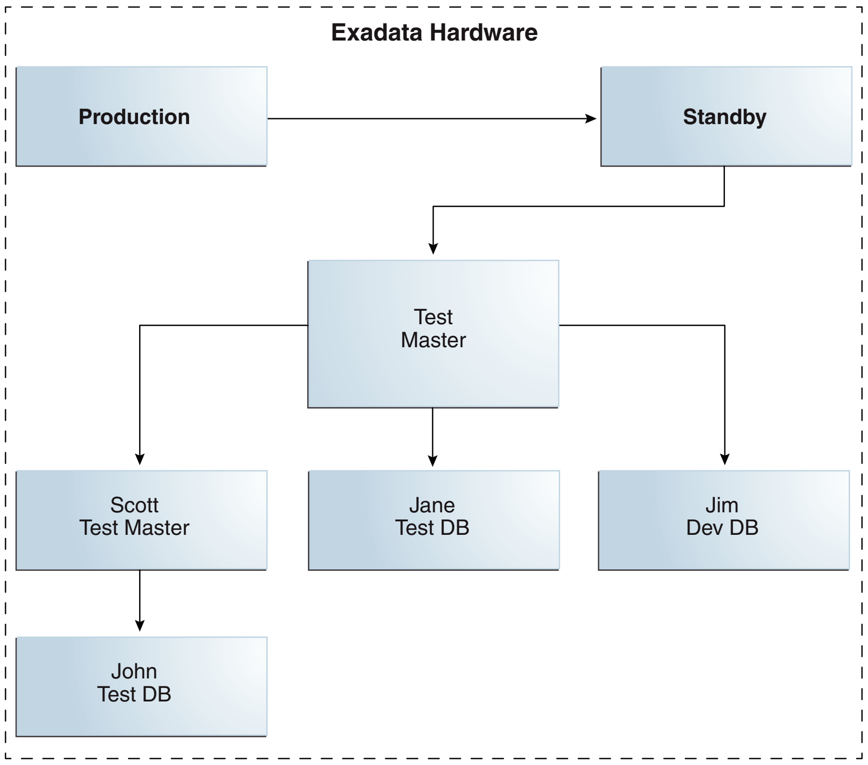
9.2.5 Hierarchical Snapshot Databases

Hierarchical snapshots enable you to create snapshot databases from other snapshot databases.

Oracle Exadata System Software release 12.2.1.1.0 introduced hierarchical snapshots. You might want to use hierarchical snapshots if you are working on your snapshot database and you want to save a copy before you make additional changes to it.

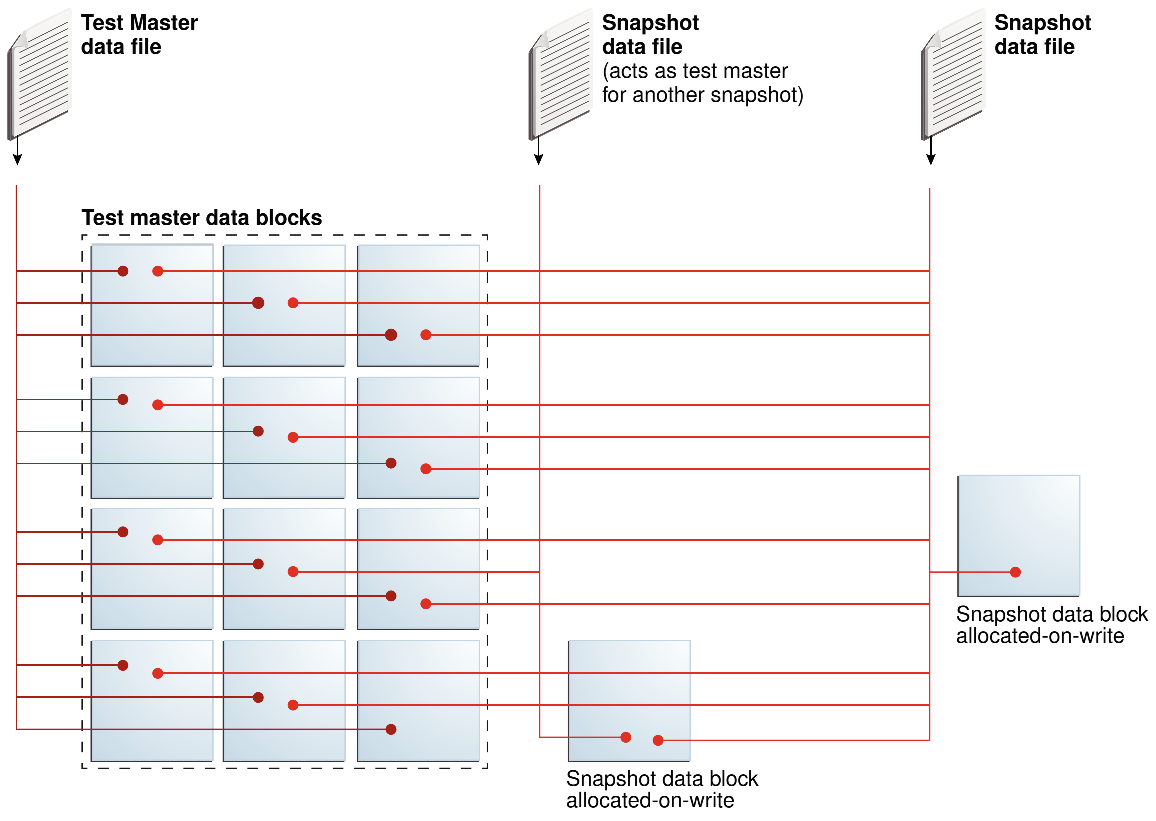
There is no set limit to the number of levels allowed in the hierarchy, but for performance and management reasons, a practical limit of 10 levels is recommended.

Figure 9-8 Hierarchical Snapshot Databases



A snapshot database points to its parent’s blocks for data. If you make a change to a snapshot database, then the snapshot database allocates a new block for the changed data. For the unchanged data, the snapshot points to the parent’s blocks. A snapshot that is several levels descended from the original test master will retrieve its data by traversing up the tree starting with the snapshot database from which it was created.

Figure 9-9 Allocation of Blocks in Hierarchical Snapshot Databases


Test Master data file
Snapshot data file (acts as test master for another snapshot)
Snapshot data file
Test master data blocks
Snapshot data block allocated-on-write
Snapshot data block allocated-on-write

If you have taken a snapshot database from another snapshot database, and you make a change to the parent snapshot database, then you have to drop all snapshot databases that depend on that snapshot database. The parent snapshot database becomes read-only when child snapshots are created from it. If you want to write to the parent snapshot again, you must drop all child snapshots.

9.2.6 Sparse Test Masters

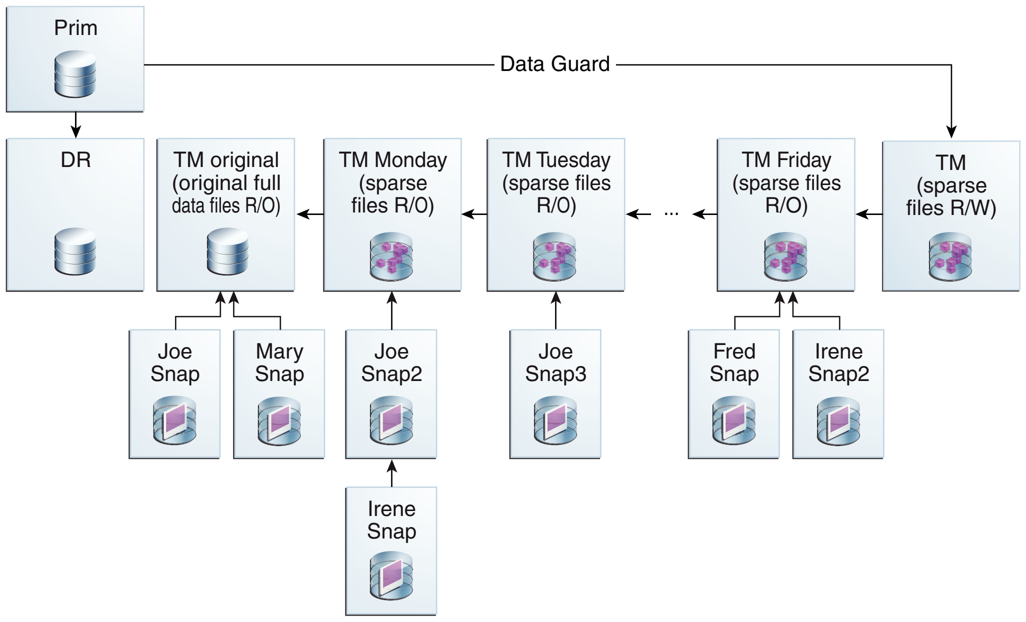
With the introduction of hierarchical snapshots, you can now create sparse test masters.

Hierarchical snapshot databases provide the capability to create sparse test masters. With a sparse test master there is one full copy of data to provide the parent data for multiple levels of snapshots. When you are ready to create snapshots, you create an additional set of sparse files to receive updates from your production database via a replication technology such as Oracle Data Guard.

When you need to create a new test master of current data, rather than cloning the full production database again as of the new point in time, you mark the previous additional sparse files read only creating a sparse test master and create a new set of sparse files to be kept current. Additional snapshots on this sparse test master could then be created to be used by application developers for testing. You can repeat this process up to 9 times total to allow one full copy of the production database to support snapshots from multiple points in time. Creating the new sparse test master is very quick (5 minutes or less) allowing you to easily keep current with production.

The sparse test master database can be a container database (CDB) or non-container database (non-CDB).

Figure 9-10 Example configuration with Sparse Test Masters


Prim
Data Guard
DR
TM original (original full data files R/O)
TM Monday (sparse files R/O)
TM Tuesday (sparse files R/O)
...
TM Friday (sparse files R/O)
TM (sparse files R/W)
Joe Snap
Mary Snap
Joe Snap2
Joe Snap3
Fred Snap
Irene Snap2
Irene Snap

Related Topics

- Creating Sparse Test Masters from a Single Full Database Copy
  You can create multiple sparse test masters from a single copy of a full database.

9.3 Prerequisites for Exadata Snapshot Databases

Before creating Exadata snapshot databases, check that your environment meets these requirements.

- Storage servers must be Oracle Exadata Database Machine X3-2 or later
- Oracle Exadata System Software 12.1.2.1.0 or later for Oracle Exadata Storage Servers and Oracle Exadata Database Servers

  You cannot downgrade to an earlier version with sparse grid disks on a cell.
- Oracle Grid Infrastructure software release 12.1.0.2.0 BP5 or later

  The Oracle ASM disk group that contains the sparse Oracle ASM grid disks must have both `COMPATIBLE.RDBMS` and `COMPATIBLE.ASM` set to 12.1.0.2 or later.

  The parent disk group can be 11.2 compatible.
- For disk groups hosting test master files, `COMPATIBLE.RDBMS` must be set to 11.2.0.0.0 or later.
- Oracle Database software release 12.1.0.2.0 BP5 or later

  The parent database and the snapshot database must be 12.1.0.2 compatible.
- The data files for the snapshot database and the parent database must be on the same Oracle ASM cluster.

- The `db_block_size` database initialization parameter must be at least 4096 and be a multiple of 4096.
- If you are using hierarchical snapshot databases, sparse test master databases, the new `--sparse` option for the Oracle ASM `cp` command, or the new `setsparseparent` Oracle ASM command, then you need Oracle Database and Oracle Grid Infrastructure software release 12.2.0.1.0 and Oracle Exadata System Software release 12.2.1.1.0 or later.
- Exadata sparse griddisks and hence Oracle ASM Sparse diskgroups cannot be created on Exadata XT storage servers.
- To create a snapshot of a complete container database (CDB), you must apply Patch 32233739 to the Oracle home directory that supports the Exadata Snapshot database.

Related Topics

- setsparseparent Command

9.4 Sparse Disk Sizing and Allocation Methodology

To create Exadata Snapshots, you must have sparse grid disks created with an Oracle ASM disk group created based on those disks.

It is not possible to directly modify an existing disk group and convert it into a sparse disk group; you must create the grid disks with the sparse attributes. If you want to use an existing disk group, then the disk group and associated grid disks must be dropped and re-created.

If you want to create a new sparse disk group and all of your space is currently allocated to existing disk groups, then you must resize one or more existing disk groups to free up space. Use the following steps to perform this process.

- Sizing Steps for New Sparse Disk Groups
  These steps describe how to determine the space needed for sparse disk groups and how to allocate that space.

9.4.1 Sizing Steps for New Sparse Disk Groups

These steps describe how to determine the space needed for sparse disk groups and how to allocate that space.

The maximum physical size that can be allocated to sparse grid disks from a single cell disk is 4 TB.

The maximum virtual size that can be allocated per sparse grid disk is 100 TB. However, to avoid additional performance impact Oracle does not recommend creating large virtual sizes unless it is truly required.

Remember also that a sparse disk group can house non-sparse database files, in which case the physical storage utilization matches the virtual file size.

1. Determine the amount of physical space for your snapshots. Use the formula found in Calculating the Physical Size for Grid Disks.

   Note that a good maximum starting point for physical space allocation for sparse grid disk is 15% of your total space. This can be tuned up or down as required based on utilization patterns and future needs.
2. Determine the amount of virtual space you will require for your snapshots. Use the formula found in Calculating the Virtual Size for Grid Disks.

Note that the virtual size can be modified online by modifying the Oracle ASM disk group without making any changes to the underlying grid disks.

3. Using the number of snapshots you plan on having at any one time and how much changes you expect to make, use the process defined in Determine the Amount of Available Space to determine which existing disk group(s) you want to shrink to release space for the sparse disk group. Based on your current allocations, you may need to shrink multiple disk groups to get the space you require for the sparse disk group. Leave a minimum of 15% free space in the original disk groups to allow for rebalancing during the resize operation.

   If there is not enough free space available in your current disk groups to provide desired space for the sparse disk group:

   - Rethink your sparse usage in this environment and downsize accordingly
   - Relocate objects and databases from this environment to another with available free space
   - Add storage

4. Once you have decided which disk group to resize, follow the steps in Shrink the Oracle ASM Disks in the Donor Disk Group and Shrink the Grid Disks in the Donor Disk Group to resize the disks.

   This process allows you to resize disks while they are online and only require a single rebalance operation per disk group. Oracle ASM may run more than one rebalance operation concurrently, depending on the number of Oracle ASM instances available in the cluster.

5. Create the sparse grid disk from the space just released using the commands outlined in Creating Sparse Grid Disks.
6. Create the sparse Oracle ASM disk group using the commands outlined in Resizing the Physical Space.
7. To monitor sparse grid disk activity see Monitoring Sparse Disk Group Utilization.
8. If you find that you need to resize your sparse grid disks:
   - If more physical space is required, follow steps 1 through 4 in this section to free up space and then follow the steps in Resizing the Physical Space. The maximum physical size that can be allocated to sparse grid disks from a single cell disk is 4 TB.
   - If more virtual space is required, follow the steps in Resizing the Virtual Space. The maximum virtual size that can be allocated to sparse grid disks from a single cell disk is 100 TB.

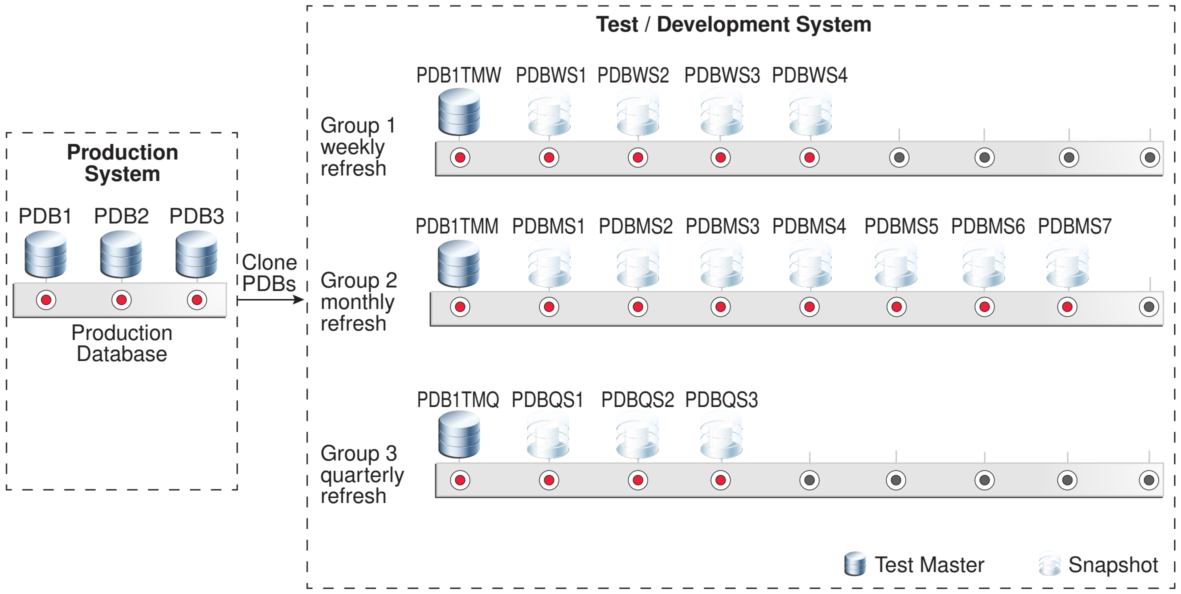
9.5 Refresh Considerations, or Lifecycle of Exadata Snapshots

Refresh cycles can influence how you use and create Exadata Snapshots.

The test master must remain in either `READONLY` state (if open) or closed while it has Exadata Snapshots associated with it. If the test master needs to be refreshed, all Exadata Snapshots dependent on the test master must be dropped and re-created.

If different groups of Exadata Snapshot users have different refresh cycle requirements, you may need to maintain multiple test masters. The following figure shows three test masters, each with its own refresh schedule.

Figure 9-11 Three Developer Groups for PDB1, Each With a Different Refresh Cycle



9.6 Using an Oracle Data Guard Standby Database as the Test Master

If the test master is a complete database that needs to be refreshed regularly, Oracle recommends creating the test master database as an Oracle Data Guard physical standby dedicated to this purpose.

There are multiple benefits when using this approach:

- Easier refresh

  Oracle Data Guard is a proven solution for synchronizing multiple physical replicas of a production database used for disaster recovery, read-only offload from production, backup offload, and test. This same functionality can be used to maintain a copy of the production database dedicated to serving as a test master database that is more easily refreshed on a periodic basis. The benefit of using an Oracle Data Guard physical standby increases as the size of the test master database and the number of times it must be refreshed increases. These benefits far outweigh the small incremental effort required to create an Oracle Data Guard replica compared to simply cloning a database from an Oracle Recovery Manager (RMAN) backup.

- Minimal impact to primary

  During the time the Oracle Data Guard replica is being used as the test master database, Oracle Data Guard redo transport and apply are disabled. There is zero impact to the production database. When it is time to refresh, only the deltas generated since the Oracle Data Guard replica was converted to a test master database are taken from the production database and used to resynchronize the test master database.

> **Note:**
>
> Since transport and apply for this Oracle Data Guard replica will be stopped while it functions as a test master, it should not be used for disaster recovery or for any purpose other than as test master. If you are already using Oracle Data Guard for high availability or disaster protection, Oracle recommends creating an Oracle Data Guard replica(s) to use as test master databases for Exadata Snapshot databases.

- Easy to scrub prior to creating snapshot clones

  Oracle Data Guard makes it easy to modify the test master database before making it available to create Exadata Snapshots. For example, an Oracle Data Guard replica can be opened read-write and data can be masked or scrubbed prior to creating Exadata Snapshots. Later, when testing is complete, the test master database can be converted back into an Oracle Data Guard replica discarding any modifications made to the original copy and refreshing it using only the deltas from the production database. Note that after refreshing the Oracle Data Guard replica, you need to re-scrub the database before you can use it as a test master again.

  If you are using an RMAN backup database, and you mask or scrub the data, when you need to refresh the test master, you have to create another backup as the test master and rescrub it to make it current.

Related Topics

- Refreshing the (Read-only) Test Master Database
  To refresh a read-only test master database, it must be converted temporarily to a read-write test master.
- Creating a Physical Standby Database

9.7 Managing Exadata Snapshots

To create and manage Exadata Snapshots, you need to perform these procedures:

- Creating Sparse Grid Disks
- Creating an ASM Disk Group for the Sparse Grid Disks
- Setting Up the Test Master
  You can create the test master using one of two methods:
- Creating Snapshots
  You can create an Exadata Snapshot of a pluggable database (PDB) or a complete database.
- Refreshing the (Read-only) Test Master Database
  To refresh a read-only test master database, it must be converted temporarily to a read-write test master.
- Creating a Snapshot Database from Another Snapshot Database
- Creating Sparse Test Masters from a Single Full Database Copy
  You can create multiple sparse test masters from a single copy of a full database.
- Creating Sparse Test Masters for PDBs
  This procedure creates a hierarchical snapshot tree or sparse test masters manually for a pluggable database (PDB) in an Oracle Multitenant database.

- Determining All Snapshots Associated with a Test Master
  Use this query to discover all of the children associated with a test master.
- Doing a Sparse Copy

9.7.1 Creating Sparse Grid Disks

When creating sparse grid disks, you need to specify the physical size and the virtual size.

- Calculating the Physical Size for Grid Disks
- Calculating the Virtual Size for Grid Disks
- Creating a Sparse Grid Disk

9.7.1.1 Calculating the Physical Size for Grid Disks

You can use the following formula to get a rough estimate of the total physical space to set aside for a sparse ASM disk group:

```
Total physical space =
   (SUM(size of all test masters in the sparse ASM disk group) +
    SUM(approximate size of all updates to the snapshot databases))
   * ASM Redundancy
```

In the formula above, ASM redundancy takes into account ASM mirroring of extents. Exadata requires ASM redundancy set to either normal redundancy (double mirror the extents) or high redundancy (triple mirror the extents). If the sparse ASM disk group will use normal redundancy, expect to double the space used. If using high redundancy, expect to triple the space used.

For example, if you want 2 test masters in the sparse ASM disk group created with normal redundancy with a combined total space of 500 GB (250 GB each) and each test master will have 5 Exadata snapshots with the expectation that each snapshot will modify 20% of the blocks, then the total physical space that will be needed can be calculated as follows:

```
Space for 2 test masters:   2 * 250 GB =                               500 GB
Space for 5 snapshots per test master, for a total of 10 snapshots:
    10 * 250 GB * 20% =                                                500 GB
Subtotal                                                              1000 GB
Normal redundancy: 2 * 1000 GB =                                      2000 GB
```

Divide this value by the number of disks to determine the `size` parameter for each disk. ASM grid disks should be allocated on 16 MB boundaries. If the `size` parameter in MB for each grid disk is not evenly divisible by 16, adjust up to a 16 MB boundary.

Note that you should set aside additional space to use for multiple projects and through multiple iterations.

Also to accommodate any disk rebalancing operations, you should add a 15% space cushion on top of the space used for snapshots and test masters.

9.7.1.2 Calculating the Virtual Size for Grid Disks

You can use the following formula to get a rough estimate of the virtual size to assign for a sparse ASM disk group:

```
Virtual size required for sparse disks =
   (SUM(full virtual size of all Exadata snapshots) + Physical space allocated)
   * ASM Redundancy
```

To continue with the example from the previous section, you have 10 Exadata snapshots. If they were full copies of the test master, they would be 250 GB each.

The following shows the calculation for the total virtual space:

```
Full size for 5 snapshots per test master, for a total of 10 snapshots:
    10 * 250 GB =                                                      2500 GB
Size of the 2 test masters: 2 * 250 GB =                                500 GB
Subtotal                                                               3000 GB
Normal redundancy: 2 * 3000 GB =                                       6000 GB
```

Divide this value by the number of disks to determine the `virtualsize` parameter for each disk. The virtual size for each grid disk should be allocated on 16 MB boundaries. If the `virtualSize` parameter in MB for each grid disk is not evenly divisible by 16, adjust up to a 16 MB boundary.

Note that you should set aside additional space to use for multiple projects and through multiple iterations.

9.7.1.3 Creating a Sparse Grid Disk

To create the sparse ASM grid disk, log in to each cell (or use `dcli`) in `CellCLI` and run a command similar to the following, changing size values where appropriate:

```
CellCLI> create griddisk all harddisk prefix=SPARSE, size=56G, virtualsize=560G
```

This creates a grid disk of physical size 56 GB but presents to ASM as a 560 GB grid disk. The `size` parameter should match the actual physical size of the ASM grid disk while the `virtualsize` parameter should be at least the physical size of the ASM grid disk.

Attributes for the ASM grid disk created above would look like the following from the "`LIST GRIDDISK DETAIL`" command:

```
CellCLI> LIST GRIDDISK DETAIL
size:        56G
sparse:      TRUE
virtualSize: 560G
```

`size` displays the actual physical size of the grid disk.

`sparse` has a value of `TRUE`.

`virtualSize` displays the virtual size of the grid disk.

9.7.2 Creating an ASM Disk Group for the Sparse Grid Disks

After you have created the sparse grid disks, you create an ASM disk group to enable those disks to be accessible to a database on the cluster. To create a disk group, log in to an ASM instance using SQL*Plus as `sysasm` and run a command similar to the following:

```
SQL> create diskgroup SPARSE high redundancy disk 'o/*/SPARSE_*' attribute
'compatible.asm'='12.1.0.2',
'compatible.rdbms'='12.1.0.2',
'au_size'='4M',
'cell.smart_scan_capable'='true',
'cell.sparse_dg'='allsparse',
'appliance.mode' = 'TRUE';
```

`compatible.asm` must be set to 12.1.0.2 or higher.

`compatible.rdbms` must be set to 12.1.0.2 or higher.

`cell.sparse_dg` must be set to "`allsparse`". This identifies the disk group to ASM as being made up of sparse grid disks.

`appliance.mode` must be set to `true`.

9.7.3 Setting Up the Test Master

You can create the test master using one of two methods:

After creating the test master, perform the tasks described in Setting the Ownership of the Test Master Data Files.

- Create a New Test Master - Full Clone on a Disk Group with ASM ACL Enabled
- Converting an Existing Full Clone or Standby Database to a Test Master
  If you already have a full clone or a standby database that you want to repurpose as a test master, then you can convert that database to a test master.
- Setting the Ownership of the Test Master Data Files
  After you have cloned the database to create a test master database, configure permissions on the disk group and data files.

9.7.3.1 Create a New Test Master - Full Clone on a Disk Group with ASM ACL Enabled

You can create a full clone of your database using RMAN backup/restore, data pump, or any of the other methods typically used to create a full clone of a database.

You should place the clone in a regular (non-sparse) Oracle ASM disk group.

After creating the full clone, make the data files read-only to help prevent accidental overwrite.

For example:

```
SQL> ALTER DISKGROUP DATA SET PERMISSON OWNER=READ ONLY, GROUP=READ ONLY, OTHER=NONE FOR
FILE 'FILENAME';
```

9.7.3.2 Converting an Existing Full Clone or Standby Database to a Test Master

If you already have a full clone or a standby database that you want to repurpose as a test master, then you can convert that database to a test master.

Standby databases cannot be running redo apply while serving as a test master.

1. If you are using an Oracle Data Guard standby database, perform the following steps:
   a. For initial creation of the Oracle Data Guard replica, use the steps outlined in My Oracle Support note 1617946.1.

      The Data Guard copy must have enough redo applied that it can be opened in a READ ONLY state.
   b. If the test master is a physical standby database and you need to make any modifications to the test master, for example, deleting or masking sensitive data, then perform the following steps:
      i. Convert the standby database into a snapshot standby.

> **Note:**
>
> An Oracle Data Guard snapshot standby is different from an Oracle Exadata snapshot. An Oracle Data Guard snapshot standby is a complete copy of the source database that is open read-write. Conversion to an Oracle Data Guard snapshot standby is a simple operation using a single command. Oracle Data Guard snapshot standby facilitates making modifications to the test master and refreshing it for subsequent rounds of testing. See *Oracle Data Guard Concepts and Administration* (referenced at the end of this topic) for more information on Oracle Data Guard snapshot standby databases.

ii. Modify the standby database as required.

c. When the standby database is at a consistent state and can be opened in `READ ONLY` mode, stop log transport to the standby and disable redo apply on the standby.

```
DGMGRL> edit database TESTMASTER set property logshipping=OFF;
Property "logshipping" updated
```

If you have not converted the physical standby database into a snapshot standby, then stop redo apply.

```
DGMGRL> edit database TESTMASTER set state=APPLY-OFF;
Succeeded
```

2. If access control is not already enabled on the disk group that contains the test master's data files, then enable access control on the disk group.

   The disk group must be on Oracle Exadata storage servers.

```
SQL> ALTER DISKGROUP DATA SET ATTRIBUTE 'ACCESS_CONTROL.ENABLED' = 'TRUE';
```

3. Grant ownership to all data files.

   See Setting the Ownership of the Test Master Data Files for details.

4. Remove write permissions on all the data files to help prevent accidental overwrite.

   SQL commands in an Oracle ASM instance only allow you to set file permissions to read only. You cannot remove write permissions in SQL.

```
SQL> ALTER DISKGROUP DATA set permission owner=read ONLY, group=read ONLY,
other=none for file 'FILENAME';
```

   This allows snapshots to be created and owned by users other than the owner of the base files.

Related Topics

- Creating a Physical Standby using RMAN Duplicate (RAC or Non-RAC) (My Oracle Support Doc ID 1617946.1)
- Managing a Snapshot Standby Database

9.7.3.3 Setting the Ownership of the Test Master Data Files

After you have cloned the database to create a test master database, configure permissions on the disk group and data files.

Set an operating system user as the owner of the disk group, and make the operating system user the owner of the test master's data files.

You can do this by running SQL commands manually in SQL*Plus, or by running a script:

- Running Commands Manually
  You can use SQL*Plus to manually run the commands to set the ownership of the test master data files.
- Running from a Script
  You can also set the ownership of the test master data files using a SQL script.

9.7.3.3.1 Running Commands Manually

You can use SQL*Plus to manually run the commands to set the ownership of the test master data files.

The following commands are run in SQL*Plus.

1. If the operating system user you are granting access to is not added as a user on the disk group, then add the user.

   Note:

   When enabling access control, all software owners that are running databases must be added as a user to the disk group.

   For example, to add the user SCOTT as an owner of the DATA disk group, use the following command:

   ```
   SQL> ALTER DISKGROUP DATA ADD USER 'scott';
   ```

2. Make the operating system user the owner of the test master's data files:

   ```
   SQL> ALTER DISKGROUP DATA SET OWNERSHIP OWNER='scott' FOR FILE
     '+DATA/TESTMASTER/DATAFILE/system.257.865863315';

   SQL> ALTER DISKGROUP DATA SET OWNERSHIP OWNER='scott' FOR FILE
     '+DATA/TESTMASTER/DATAFILE/sysaux.258.865863317';

   SQL> ALTER DISKGROUP DATA SET OWNERSHIP OWNER='scott' FOR FILE
     '+DATA/TESTMASTER/DATAFILE/sysext.259.865863317';

   SQL> ALTER DISKGROUP DATA SET OWNERSHIP OWNER='scott' FOR FILE
     '+DATA/TESTMASTER/DATAFILE/tbs_1.256.865863315';
   ```

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

9.7.3.3.2 Running from a Script

You can also set the ownership of the test master data files using a SQL script.

The following procedure is equivalent to the commands in the previous topic, but it queries `V$DATAFILE` for the filenames:

1. Add an operating system user as owner of the disk group.

   ```
   SQL> ALTER DISKGROUP DATA ADD USER 'scott';
   ```

2. Generate a script called `set_owner.sql` to set the owner of the test master's data files.

   - If the test master is a full database, run the following in the test master database:

     ```
     set newpage 0
     set linesize 999
     set pagesize 0
     set feedback off
     set heading off
     set echo off
     set space 0
     set tab off
     set trimspool on
     spool set_owner.sql
     select 'ALTER DISKGROUP DATA set ownership
     owner='||''''||'scott'||''''||' for file '||''''||name||''''||';' from
     v$datafile;
     exit
     ```

   - If the test master is a PDB, run the following in the CDB$ROOT of the test master PDB:

     In the `select` statement below, the example assumes the test master PDB has a con_id of `10`.

     ```
     set newpage 0
     set linesize 999
     set pagesize 0
     set feedback off
     set heading off
     set echo off
     set space 0
     set tab off
     set trimspool on
     spool set_owner.sql
     select 'ALTER DISKGROUP DATA set ownership
     owner='||''''||'scott'||''''||' for file '||''''||name||''''||';' -
     from v$datafile where con_id=10;
     exit
     ```

3. Remove extra lines in `set_owner.sql`.

   ```
   sed -i '/SQL/d' set_owner.sql
   ```

4. Run the script in the ASM instance.

```
SQL> @set_owner
```

9.7.4 Creating Snapshots

You can create an Exadata Snapshot of a pluggable database (PDB) or a complete database.

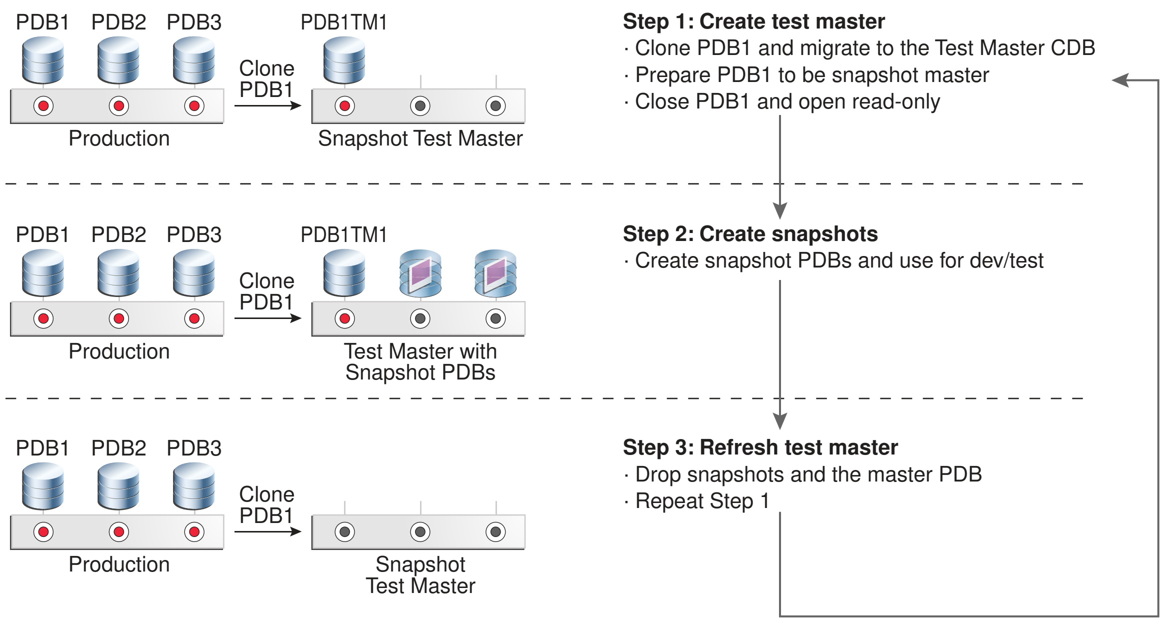
- Creating a Snapshot of a Pluggable Database
- Creating a Snapshot of a Full Database
  You create an Exadata snapshot of a full database.

9.7.4.1 Creating a Snapshot of a Pluggable Database

Creating an Exadata snapshot of a pluggable database (PDB) is the simplest method for creating a snapshot because it requires no additional manual steps. Two new clauses in the `CREATE PLUGGABLE DATABASE` statement identify the PDB as an Exadata snapshot.

Creating individual Exadata snapshot PDBs is best used when creating snapshots for a smaller number of PDBs within a given CDB. The following figure shows a high-level example of a typical lifecycle for a PDB with two PDB snapshots.

Figure 9-12 Lifecycle of Exadata Snapshots Using PDBs


PDB1
PDB2
PDB3
Clone PDB1
PDB1TM1
Production
Snapshot Test Master
Step 1: Create test master
· Clone PDB1 and migrate to the Test Master CDB
· Prepare PDB1 to be snapshot master
· Close PDB1 and open read-only
PDB1
PDB2
PDB3
Clone PDB1
PDB1TM1
Production
Test Master with Snapshot PDBs
Step 2: Create snapshots
· Create snapshot PDBs and use for dev/test
PDB1
PDB2
PDB3
Clone PDB1
Production
Snapshot Test Master
Step 3: Refresh test master
· Drop snapshots and the master PDB
· Repeat Step 1

One of the benefits of Oracle Multitenant and PDBs is the ability to easily clone an existing PDB to create a test master and move it from one CDB to another. Oracle recommends that you clone your source PDB to create the test master and then migrate it to your test environment where you can perform any data scrubbing that may be needed. Once complete, the test master PDB can be used to create numerous Exadata snapshot PDBs.

An Exadata snapshot PDB is created as a PDB in the CDB. Like all other PDBs, an Exadata snapshot PDB is subject to the general limit for the maximum number of PDBs in a single

CDB. For Oracle Database 12c release 1 (12.1), the limit is 252 PDBs in a single CDB. For Oracle Database 12c release 2 (12.2) and later, the limit is 4096 PDBs in a single CDB.

When creating an Exadata snapshot PDB, the command changes file permissions on the test master PDB, marking the files as `READONLY` in Oracle ASM. Consequently, if you want to delete the test master PDB, you must first drop all the snapshot copies and change the test master PDB file permissions back to `READ WRITE`.

Note:

If you drop the test master PDB before changing the file permissions to `READ WRITE`, the command completes without error. However, the underlying database files remain, and an entry is written to the database alert log file. In this case, you must manually delete the files in Oracle ASM to free up the space occupied by the files.

After creating the test master PDB, perform the following steps to create an Exadata snapshot PDB:

1. In SQL*Plus, connect to the CDB root container (`cdb$root`).
2. Close the test master PDB in all instances.

   ```
   SQL> alter pluggable database PDB1TM1 close instances=all;
   ```

3. Open the test master PDB in the local instance in read only mode.

   ```
   SQL> alter pluggable database PDB1TM1 open read only;
   ```

4. Create an Exadata snapshot PDB of the test master.

   ```
   SQL> create pluggable database PDB1S1 from PDB1TM1 tempfile reuse
   create_file_dest='+SPARSE' snapshot copy;
   ```

   The `create_file_dest` argument must specify the name of a sparse disk group. The `snapshot copy` clause creates the PDB as a snapshot rather than a full PDB clone.

   If you need to create more PDBs than can fit in a single CDB, you can create the Exadata snapshot PDB in another CDB by creating a remote clone using a database link. For example:

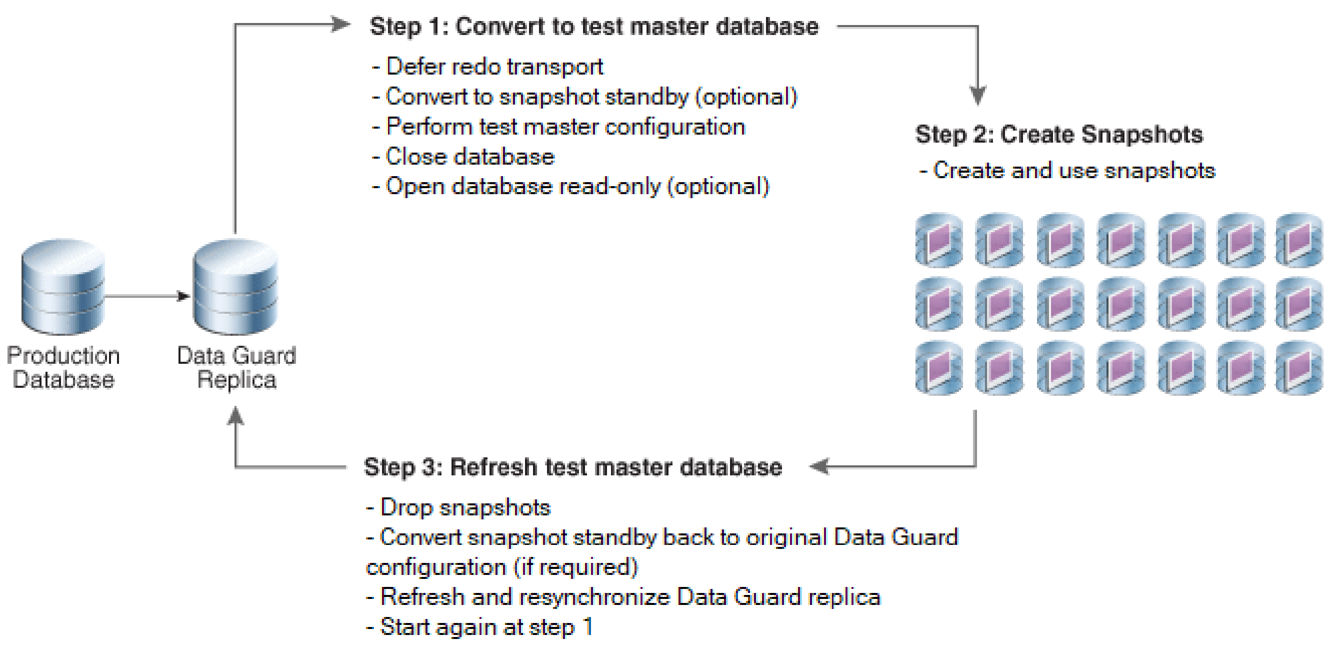
   ```
   SQL> create pluggable database PDB1S1 from PDB1TM1@cdb1_dblink tempfile
   reuse create_file_dest='+SPARSE' snapshot copy;
   ```

9.7.4.2 Creating a Snapshot of a Full Database

You create an Exadata snapshot of a full database.

The following figure shows the lifecycle of an Exadata test master database and snapshot databases where the test master is based on an Oracle Data Guard replica.

Figure 9-13 Lifecycle of Test Master and Snapshot Databases


Step 1: Convert to test master database
- Defer redo transport
- Convert to snapshot standby (optional)
- Perform test master configuration
- Close database
- Open database read-only (optional)
Step 2: Create Snapshots
- Create and use snapshots
Production Database
Data Guard Replica
Step 3: Refresh test master database
- Drop snapshots
- Convert snapshot standby back to original Data Guard configuration (if required)
- Refresh and resynchronize Data Guard replica
- Start again at step 1

Note that the test master database cannot be used as a failover target to provide high availability or disaster recovery (a Data Guard configuration may have multiple replicas that can each serve different purposes). Similar to test master PDBs, test master databases cannot be modified while Exadata snapshots exist against them.

The test master database cannot be a read-only physical standby database that is in recovery mode (for example, Active Data Guard in Real Time Apply).

The test master database and their Exadata snapshots must be in the same Oracle ASM cluster environment.

Note:

If the test master is an Oracle RAC database, you have to do one of the following:

- Create redo logs for all threads in the `CREATE CONTROLFILE` statement for the sparse clone, OR
- Specify Oracle Managed Files for the redo logs using the `ONLINE_LOG_CREATE_DEST_1` initialization parameter in the SPFILE of the sparse clone to have the redo logs created automatically.

1. In the test master database, create a sample control file script to use for your Exadata snapshot databases by backing up the existing control file to trace.

   Connect to the test master database via SQL*Plus as SYSDBA and do the following:

   a. Determine name and location of any trace file to be generated by your session.

For example:

```
SQL> SELECT value FROM v$diag_info WHERE name = 'Default Trace File';

VALUE
--------------------------------------------------------------------------
---------
/u01/app/oracle/diag/rdbms/TESTMASTER/TESTMASTER1/trace/
TESTMASTER1_ora_26756.trc
```

b. Run the BACKUP CONTROLFILE TO TRACE command to place the CREATE CONTROLFILE command into the trace file.

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO TRACE;
```

c. Retrieve the file shown for the Default Trace File.

2. In the test master database, determine the existing file names for the rename that will happen in step 11.

For example, log into SQL*Plus as SYSDBA and run the following:

```
SET newpage 0
SET linesize 999
SET pagesize 0
SET feedback off
SET heading off
SET echo off
SET space 0
SET tab off
SET trimspool on
SPOOL rename_files.sql
SELECT 'EXECUTE dbms_dnfs.clonedb_renamefile -
('||''''||name||''''||','||''''||REPLACE(REPLACE(REPLACE(name,'.','_'),-
'TESTMASTER','SNAPTEST'),'+DATA','+SPARSE')||''''||');' FROM v$datafile;
EXIT
```

The example script builds a file named `rename_files.sql` that contains statements for each data file similar to the following:

```
EXECUTE dbms_dnfs.clonedb_renamefile (
'+DATA/TESTMASTER/DATAFILE/system.257.865863315',
'+SPARSE/SNAPTEST/DATAFILE/system_257_865863315');
```

In the example script, the REPLACE function:

- Replaces periods with underscores in the original file name
- Replaces the original database name of `TESTMASTER` with `SNAPTEST`
- Replaces the original disk group name of `+DATA` with `+SPARSE`

If you modify the example to suite your environment, ensure that you maintain consistency throughout the rest of the procedure.

3. Create an `init.ora` file with the contents of the test master SPFILE.

   ```
   SQL> CREATE PFILE = 'init_TestMaster.ora' FROM SPFILE;
   ```

4. Shut down the test master.

   ```
   SQL> shutdown;
   ```

5. Create an `init.ora` file for the Exadata snapshot database.

   You can use the `init.ora` file of the test master as a template, but make sure to change the `db_name` and `control_files` entries.

   For this procedure, the `init.ora` file for the Exadata snapshot database is referenced as `snap_init.ora` in commands and examples.

   ```
   $ cp init_TestMaster.ora snap_init.ora
   ```

   Modify `snap_init.ora` with the new database name, new control file name, and audit file destination, for example:

   ```
   db_name = SNAPTEST
   control_files = '+DATA/SNAPTEST/control1.f'
   audit_file_dest=/u01/app/oracle/admin/snaptest/adump
   ```

   You can save the modified `snap_init.ora` file as a template for creating additional snapshot copies of the test master.

6. Create a script to create a control file for the Exadata snapshot database.

   You will use this script later in step 10.

   a. Examine the trace file generated in step 1 and locate the `CREATE CONTROLFILE` command that includes the `RESETLOGS` clause.

      The required command is located in the trace file immediately after the following comments:

      ```
      --     Set #2. RESETLOGS case
      --
      -- The following commands will create a new control file and use it
      -- to open the database.
      -- Data used by Recovery Manager will be lost.
      -- The contents of online logs will be lost and all backups will
      -- be invalidated. Use this only if online logs are damaged.
      ```

   b. Copy the `CREATE CONTROLFILE` command into a new script file.

      Copy only the complete `CREATE CONTROLFILE` command and discard all surrounding comments and commands.

      For example, the script file may be named `crt_ctlfile.sql`.

   c. Modify the `CREATE CONTROLFILE` command to create a control file for the Exadata snapshot database.

      The control file should be created with the Exadata snapshot database name, new log file names, and the data file names of the test master. The new log files can be in any disk group that has enough space, but they should not be created in the sparse Oracle ASM disk group.

The following shows an example `CREATE CONTROLFILE` command. In the example, `SNAPTEST` is the Exadata snapshot database. The `LOGFILE` lines specify the new log file locations, and the `DATAFILE` lines specify the data file locations for the test master database.

```
CREATE CONTROLFILE REUSE SET DATABASE SNAPTEST RESETLOGS ARCHIVELOG
      MAXLOGFILES 32
      MAXLOGMEMBERS 2
      MAXINSTANCES 1
      MAXLOGHISTORY 908
  LOGFILE
      GROUP 1 '+DATA/SNAPTEST/t_log1.f' SIZE 100M BLOCKSIZE 512,
      GROUP 2 '+DATA/SNAPTEST/t_log2.f' SIZE 100M BLOCKSIZE 512
  DATAFILE
      '+DATA/TESTMASTER/DATAFILE/system.257.865863315',
      '+DATA/TESTMASTER/DATAFILE/sysaux.258.865863317',
      '+DATA/TESTMASTER/DATAFILE/sysext.259.865863317',
      '+DATA/TESTMASTER/DATAFILE/tbs_1.256.865863315'
  CHARACTER SET WE8DEC;
```

You can save the modified script file as a template for creating additional snapshot copies of the test master.

7. Create the `audit_file_dest` directory on all nodes on which the snapshot will be running.

```
$ mkdir -p /u01/app/oracle/admin/snaptest/adump
```

8. Create the directories in Oracle ASM for the snapshot data files.

For example:

```
$ asmcmd -p
ASMCMD > cd +SPARSE
ASMCMD [+sparse] > mkdir SNAPTEST
ASMCMD [+sparse] > cd SNAPTEST
ASMCMD [+sparse/snaptest] > mkdir DATAFILE
```

If the test master is a container database (CDB), create additional data file sub-directories for each pluggable database (PDB). Use the globally unique identifier (GUID) for each PDB, which is available in the `V$PDBS` view.

For example:

```
$ sqlplus / as sysdba
SQL> ALTER SESSION set container=PDB1;
SQL> SELECT guid FROM v$pdbs;

                            GUID
--------------------------------
A839E00B5E9E7820E053412E850A5F18


$ asmcmd -p
ASMCMD > cd +SPARSE
ASMCMD > cd +SPARSE/SNAPTEST
ASMCMD [+sparse/snaptest] > mkdir A839E00B5E9E7820E053412E850A5F18
ASMCMD [+sparse/snaptest] > cd A839E00B5E9E7820E053412E850A5F18
ASMCMD [+sparse/snaptest/A839E00B5E9E7820E053412E850A5F18] > mkdir DATAFILE
```

9. Start a database instance pointing to the Exadata snapshot database `init.ora` file (`snap_init.ora`) using the following commands:

   ```
   $ sqlplus / as sysdba
   SQL> startup nomount pfile=snap_init.ora
   ```

10. Create the Exadata snapshot control file using the script created in step 6.

    In the following example the script is named `crt_ctlfile.sql`.

    ```
    SQL> @crt_ctlfile
    ```

11. Run the script you modified in step 2.

    All the files must be renamed prior to opening the Exadata snapshot database.

    Connect using SQL*Plus as SYSDBA to the Exadata snapshot database and run the following command:

    ```
    SQL> @rename_files
    ```

    This script modifies the permissions of the test master database files stored in Oracle ASM, marking them as READONLY.

    The `dbms_dnfs.clonedb_renamefile` procedure, which is called by `rename_files.sql`, sets up the parent-child relationship between the test master database and the snapshot database, and renames the file names in the snapshot database's control file.

12. Open the Exadata snapshot database with the RESETLOGS option:

    ```
    SQL> ALTER DATABASE OPEN RESETLOGS;
    ```

13. Confirm that the Exadata snapshot files are child files of the test master database. Connect using SQL*Plus as SYSDBA to the Exadata snapshot, and run the following command:

    ```
    SQL> SELECT filenumber num, clonefilename child, snapshotfilename parent
    FROM V$CLONEDFILE;
    ```

    The following is an example of the output from the query:

    ```
    NUM  CHILD
    ---- ---------------------------------------------
    PARENT
    -----------------
    1     +SPARSE/SNAPTEST/DATAFILE/system_257_865863315
    +DATA/TESTMASTER/DATAFILE/system.257.865863315

    2     +SPARSE/SNAPTEST/DATAFILE/sysaux_258_865863317
    +DATA/TESTMASTER/DATAFILE/sysaux.258.865863317

    3     +SPARSE/SNAPTEST/DATAFILE/sysext_259_865863317
    ```

```
+DATA/TESTMASTER/DATAFILE/sysext.259.865863317

4     +SPARSE/SNAPTEST/DATAFILE/tbs_1_256_865863315
 +DATA/TESTMASTER/DATAFILE/tbs_1.256.865863315
```

14. Log in using SQL*Plus to the Exadata snapshot database, and add temp files to the TEMP tablespace. This is a full size temp file, not a sparse temp file.

    ```
    SQL> ALTER TABLESPACE temp ADD TEMPFILE '+DATA' SIZE 10G;
    ```

    Additionally, for a snapshot of a complete CDB, connect to each PDB and add a temp file to the PDB-specific TEMP tablespace.

    For example:

    ```
    SQL> ALTER SESSION set container=PDB1;
    SQL> ALTER TABLESPACE temp ADD TEMPFILE '+DATA' SIZE 10G;
    ```

9.7.5 Refreshing the (Read-only) Test Master Database

To refresh a read-only test master database, it must be converted temporarily to a read-write test master.

To refresh the (read-only) test master database, the main steps are:

- Drop the Snapshot Databases
  Delete the Exadata snapshot databases that are children of the test master database you want to refresh.
- Change the Permissions on the Test Master to Read-Write
  To modify the data files from read-only to read-write you can use SQL to generate a script of SQL commands.
- Convert the Test Master Database Back to a Data Guard Replica
  Now that the data files for the test master are writable, convert the read-only test master database into an Oracle Data Guard replica.
- Update the Test Master Database
- Close the Test Master and Make All Test Master Data Files Read-Only
  After the Test Master has been updated, you can revert it to a read-only test master.
- Re-create All Snapshots
  After the Test Master has been updated and made read-only again, re-create all the snapshot databases to get the latest updates.

9.7.5.1 Drop the Snapshot Databases

Delete the Exadata snapshot databases that are children of the test master database you want to refresh.

You can delete the snapshot databases using RMAN.

1. Connect to the Exadata snapshot database using RMAN with the Exadata snapshot as the target.

Connect as a user that has the necessary privileges to start and drop a database, such as SYSBACKUP or SYSDBA.

```
RMAN> CONNECT TARGET "user@snapdb_name AS SYSDBA"
```

2. Start the snapshot database in restricted mode.

```
RMAN> STARTUP FORCE MOUNT DBA
```

3. Delete the snapshot database.

```
RMAN> DROP DATABASE;
```

Note:

- Failure to drop an Exadata snapshot database has no impact on the state of a test master database. However, an Exadata snapshot can behave unpredictably if its test master database is dropped or refreshed.
- If you have a snapshot hierarchy, you must delete all of the snapshots at all levels of the snapshot hierarchy that are children of the test master database you want to refresh.

9.7.5.2 Change the Permissions on the Test Master to Read-Write

To modify the data files from read-only to read-write you can use SQL to generate a script of SQL commands.

Before starting this part of refreshing the read-only test master database, you must first drop the snapshot databases.

1. After all Exadata snapshot databases have been deleted, start up the test master database in mount mode.
2. Create a script to reset permissions on the data files for the test master database.

   Connect to the test master using SQL*Plus and run the following script to create a new SQL script that contains the commands to reset permissions on the data files belonging to the test master database.

```
set newpage 0
set linesize 999
set pagesize 0
set feedback off
set heading off
set echo off
set space 0
set tab off
set trimspool on
spool change_perm.sql
select 'ALTER DISKGROUP DATA set permission owner=read write,
group=read write, other=none for file '||''''||name||''''||';' from
v$datafile;
exit
```

3. Remove the extra lines from the generated script.

   Run the following `sed` command to remove extra lines from the `change_perm.sql` script that you created in the previous step. Run this command from an operating system prompt, in the directory that contains the `change_perm.sql` script.

   ```
   $ sed -i '/SQL/d' change_perm.sql
   ```

4. Use the generated script to change the file permissions.

   Use SQL*Plus to connect to an Oracle ASM instance as a SYSASM user. Run the `change_perm.sql` script. This script changes the permissions of the test master's data files to make them writable.

   ```
   SQL> @change_perm
   ```

9.7.5.3 Convert the Test Master Database Back to a Data Guard Replica

Now that the data files for the test master are writable, convert the read-only test master database into an Oracle Data Guard replica.

If you had originally prepared the test master database using Oracle Data Guard snapshot standby, then convert it back to its original state as an Oracle Data Guard replica using the `CONVERT` command. This command discards any changes previously made to the replica to prepare it to be the test master. It also makes it possible to refresh the test master using just incremental changes from the source database instead of a complete restore from a current backup.

9.7.5.4 Update the Test Master Database

You have two options for refreshing the test master database:

- Allow Oracle Data Guard to refresh the test master database

  If the Oracle Data Guard replica has been used as a test master database for only a short period of time and you have all the redo generated during this time in archive logs on disk at the source database, then you can enable redo shipping and start redo apply. The test master database will use regular Oracle Data Guard protocols to retrieve archive logs and apply the logs until it is caught up with the primary database. Once the Oracle Data Guard replica is as current as you need it to be, disable redo shipping, stop redo apply and repeat the test master and snapshot creation cycle described in Setting Up the Test Master and Creating Snapshots.

  This option has the benefit of being able to stop redo apply at some intermediate point rather than bringing the test master database totally current.

  To let Oracle Data Guard refresh the standby, enable log shipping to the standby and redo apply on the standby:

  ```
  DGMGRL> edit database TESTMASTER set property logshipping=ON;
  Property "logshipping" updated
  DGMGRL> edit database TESTMASTER set state=apply-on;
  Succeeded
  ```

- Use RMAN `RECOVER...FROM SERVICE` to roll forward the test master database

  If the Oracle Data Guard replica has been used as a test master database for a long period of time or if you no longer have the redo available on disk to enable Oracle Data Guard to automatically refresh the test master database, use RMAN to perform live incremental apply over the network.

A major advantage to using this method is that no additional disk space is required. RMAN will bring changed blocks to the standby from the primary over the network and apply them directly. Also RMAN greatly simplifies the process by determining which blocks need to be retrieved based on the SCN of the data files on the test master. With this method you cannot recover to an intermediate point in time; the refresh will bring the test master database current with the primary. For more information on this method refer to Performing RMAN Recovery: Advanced Scenarios in *Oracle Database Backup and Recovery User's Guide*.

To refresh the test master database using RMAN Network Incrementals:

1. Prepare Oracle Net Services for the RMAN connections.

   These steps need to be performed only once.

   a. Create a `listener.ora` entry for the test master database (the Oracle Data Guard replica).

      The listener entry allows RMAN to connect to the target using the SID because the service is not started when the database is opened in `NOMOUNT` mode. The following is an example of the entry:

      ```
      SID_LIST_LISTENER =
        (SID_LIST =
          (SID_DESC =
            (SID_NAME = TESTMASTER1)
            (ORACLE_HOME = /u01/app/oracle/product/12.1.0.2/dbhome_1)
          )
        )
      ```

   b. Reload the listener to pick up the changes to the `listener.ora`.

      ```
       $ lsnrctl reload listener
      ```

   c. Create a TNS entry on the test master environment pointing to the SID of the local test master instance.

      The entry should use the local host name rather than the SCAN name to ensure the connection request goes to the correct host.

      ```
      TESTMASTER1 =
        (DESCRIPTION =
          (ADDRESS = (PROTOCOL = TCP)(HOST = standbydb01.example.com)(PORT =
      1521))
          (CONNECT_DATA =
            (SERVER = DEDICATED)
            (SID = TESTMASTER1)
            (UR=A)
          )
        )
      ```

2. Connect via RMAN to the test master database and save the `CURRENT_SCN` for later.

This value will be used to determine if newly created files since the last refresh need to be restored from the source database.

```
RMAN> select current_scn from v$database;
CURRENT_SCN#
------------------
          17081990
```

3. List the names and group identifiers of the redo log files.

   The names of the online redo log files and standby redo log files of the Oracle Data Guard replica might be required in a later step.

   ```
   RMAN> SELECT type, group#, member FROM v$logfile;
   ```

4. Refresh the standby control file of the Oracle Data Guard replica from the source database to make the control file current.

   a. Reconnect to the Oracle Data Guard replica as the RMAN target.

   b. Restart the target in `NOMOUNT` mode.

      ```
      RMAN> startup nomount force;
      ```

   c. Restore the standby control file by using the control file on the source database.

      The following example restores the control file on the Oracle Data Guard replica by using the database control file from `SOURCEMASTER`, the source database.

      ```
      RMAN> RESTORE STANDBY CONTROLFILE FROM SERVICE SOURCEMASTER;
      ```

   d. Mount the Oracle Data Guard replica.

      ```
      RMAN> ALTER DATABASE MOUNT;
      ```

5. Update the names of the data files and the temp files in the standby control file.

   If you are not using an RMAN catalog, the names of files in the standby control file are the names that were used in the source database, not the standby.

   Use the `CATALOG` command and the `SWITCH` command to update all the data file names. The `SWITCH` command will be used after restoring any newly created files from the source database in step 7.

   In the following example, `+DATA/TESTMASTER/DATAFILE/` is the location of the data files on the Oracle Data Guard replica. All data files must be stored in this location.

   ```
   RMAN> CATALOG START WITH '+DATA/TESTMASTER/DATAFILE/';
   ```

6. Determine if new files were added that need to be restored from the source database.

   Use the `CURRENT_SCN` from step 2.

   ```
   RMAN> SELECT file# FROM v$datafile WHERE creation_change# >= 17081990;
    FILE#
   ----------
            9
           10
   ```

7. If there are files returned by the previous query, restore those data files from the source database.

   Run an RMAN command block similar to the following using the list of `FILE#` values returned by the previous step. If no `FILE#` values were returned, then skip this step.

   ```
   RMAN> run{
   2> set newname for database to '+DATA';
   3> restore datafile 9,10 from service SOURCEMASTER;
   4> }
   ```

8. If not using an RMAN catalog, rename the data files in the standby control file.

   Switch to the copies cataloged in step 5.

   ```
   RMAN> SWITCH DATABASE TO COPY;
   ```

9. Update the names of the online redo logs and standby redo logs in the standby control file.

   Use one of the following methods:

   - Use the `ALTER DATABASE CLEAR` command to clear the log files in all redo log groups of the Oracle Data Guard replica. RMAN then recreates all the standby redo logs and the online redo log files.

     Note:

     Clearing log files is recommended only if the Oracle Data Guard replica does not have access to the online redo log files and standby redo log files of the source database. If the Oracle Data Guard replica has access to the redo log files of the source database and the redo log file names of the source database are OMF names, then the `ALTER DATABASE` command will delete log files on the source database.

     Also, the clearing of the log files will create new log files. Any existing log files are not used because the control file is not aware of those existing files. To conserve space, delete the existing log files from Oracle ASM prior to running the `ALTER DATABASE CLEAR` commands.

     The `GROUP#` column of the `V$LOGFILE` view queried in step 5 provides the redo log group identifiers of the log groups that must be cleared. Use separate `ALTER DATABASE CLEAR` commands to clear each redo log group.

     For example, the following command clears the redo log group with identifier 2.

     ```
     SQL> ALTER DATABASE CLEAR LOGFILE GROUP 2;
     ```

   - Use the `ALTER DATABASE RENAME FILE` command to rename the redo log files. Use a separate command to rename each log file listed in step 5.

     To rename log files, the `STANDBY_FILE_MANAGEMENT` initialization parameter must be set to `MANUAL`. Renaming log files is recommended when the number of online redo logs files and standby redo log files is the same in the source database and the Oracle Data Guard replica.

10. Use RMAN `RECOVER....FROM SERVICE` to roll forward the data files to current state.

No additional space is required for this operation. Note that this process can only bring the files totally current; it cannot bring the files to a previous point in time. Connect via RMAN to the Oracle Data Guard replica as target using the TNS entry created in step 3. The service specified should point to the primary.

```
RMAN> recover database noredo from service SOURCEMASTER;
```

11. Enable redo shipping to the Oracle Data Guard replica and start redo apply.

    This is necessary to update the control file with the blocks applied as part of step 10.

    ```
    DGMGRL> edit database TESTMASTER set property logshipping=ON;
    Property "logshipping" updated
    DGMGRL> edit database TESTMASTER set state=apply-on;
    Succeeded.
    ```

12. After redo has been applied, repeat the process you used to convert the Oracle Data Guard replica into a test master database and then create Exadata database snapshots.

    Remember to once again disable log shipping and redo apply at the standby.

9.7.5.5 Close the Test Master and Make All Test Master Data Files Read-Only

After the Test Master has been updated, you can revert it to a read-only test master.

Complete one of the tasks described in "Setting the Ownership of the Test Master Data Files".

9.7.5.6 Re-create All Snapshots

After the Test Master has been updated and made read-only again, re-create all the snapshot databases to get the latest updates.

You can create an Exadata snapshot database of a pluggable database (PDB) or of a full database as described in "Creating Snapshots."

9.7.6 Creating a Snapshot Database from Another Snapshot Database

To create a snapshot from a snapshot:

1. Create a first level snapshot. In the following example, the snapshot is called `PDB1S1`.

    ```
    create pluggable database PDB1S1
      from PDB1TM1
      create_file_dest='+SPARSE'
      snapshot copy;
    ```

2. Open and close the PDB, so you can re-open it as read-only in the next step.

    ```
    alter pluggable database PDB1S1 open;
    alter pluggable database PDB1S1 close;
    ```

3. Open the PDB in read-only mode so it can serve as a test master.

    ```
    alter pluggable database PDB1S1 open read only;
    ```

4. Create a snapshot from the snapshot created in step 1. In the following example, the second level snapshot is called `PDB1S1_A`.

    ```
    create pluggable database PDB1S1_A
      from PDB1S1
      create_file_dest='+SPARSE'
    ```

```
  snapshot copy;

alter pluggable database PDB1S1_A open;
```

Related Topics

- Hierarchical Snapshot Databases
  Hierarchical snapshots enable you to create snapshot databases from other snapshot databases.

9.7.7 Creating Sparse Test Masters from a Single Full Database Copy

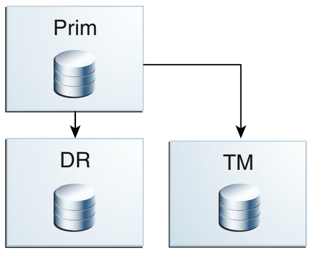
You can create multiple sparse test masters from a single copy of a full database.

In this procedure, the source for the test masters is a full copy of a Data Guard physical standby database. This standby database should not be used as a target for switchover or failover; it should only be used as the source for the test masters defined in this process.

This process takes advantage of the hierarchical snapshot functionality to allow redo to be shipped and applied, keeping the standby database current with production while also providing files to be used as source for test masters to be used by Exadata storage snapshots. The physical standby database begins as a full copy of the primary database. When you are ready to create storage snapshots, sparse data files are created pointing to the full database files to apply redo shipped from the primary. These sparse files are then used in the standby database instance to apply redo. You can also open the sparse data files in Active Data Guard mode to supply read only access of current data.

When additional snapshots are required at different points in time, you repeat the process of creating new sparse files on top of the previously created sparse files to apply redo and keep the data current. This allows you to use a single full copy of the data files to use as multiple test masters from different points in time. Also, you can create a new test master in a matter of minutes because you do not have to drop the existing snapshots.

Figure 9-14 Starting Configuration



The following tasks assume that a physical standby database has already been created to be used as the source for the test masters. The database can be a container database (CDB) or non-container database (non-CDB).

- Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master
- Task 2: Configure the Sparse Test Master and Sparse Files on the Standby Site
  In this task you convert the standby into a test master and create sparse files to receive and apply redo from the primary database.
- Task 3: Create Full Database Snapshots Using the New Sparse Test Master
- Task 4: Create a New Sparse Test Master Using a Previously Created Sparse Test Master
  Create a new set of snapshots to provide a new test master and new sparse files to contain changes from the primary.

Related Topics

- Using an Oracle Data Guard Standby Database as the Test Master
  If the test master is a complete database that needs to be refreshed regularly, Oracle recommends creating the test master database as an Oracle Data Guard physical standby dedicated to this purpose.
- Creating a PDB in a Primary Database

9.7.7.1 Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master

The existing files for the standby database are used to support snapshots. You create a series of sparse files pointing to the existing files of the standby. Redo received from the primary database is applied to these files. These sparse files allow the standby database to be used as a sparse test master and also kept current with the primary database.

1. Stop redo apply at the standby.

   To ensure that the structure of the database is at a quiesced state for creating the supporting files to build snapshots, redo apply should be turned off at the standby database.

   ```
   DGMGRL> edit database tm_standby set state='APPLY-OFF';
   ```

2. Prepare the current standby database to be used as a test master.

   The system must be configured as follows:

   a. The disk group containing the database files must have the `access_control.enabled` attribute set to `TRUE`.

      As SYSASM, log into Oracle ASM using SQL*Plus and configure the disk group.

      For example:

      ```
      SQL> alter diskgroup DATA set attribute 'ACCESS_CONTROL.ENABLED'='TRUE';
      ```

   b. The operating system (OS) user of the database owner must be added as an explicit user of the disk group containing the database files.

      For example:

      ```
      SQL> alter diskgroup DATA add user 'scott';
      ```

   c. The database files that are to be used must have explicit permissions granted to the database owner OS user.

      You must perform this step for all OS users that will be creating snapshots using these files and for all files that will be referenced by the snapshots.

      The following script can be used to build SQL statements to configure the ownership settings. Run the script while connected to the standby database using SQL*Plus. If the standby is a container database (CDB) you must be connected to the `cdb$root` container:

      ```
      set newpage 0
      set linesize 999
      set pagesize 0
      set feedback offset heading off
      set echo off
      ```

```
set space 0
set tab off
set trimspool on
spool set_owner.sql
select 'ALTER DISKGROUP DATA set ownership
owner='||''''||'scott'||''''||'
 for file '||''''||name||''''||';' from v$datafile;
exit
```

After running the previous script, log in to Oracle ASM using SQL*Plus as the SYSASM user, and run the commands in `set_owner.sql`.

```
SQL> @set_owner
```

3. Create a backup of the controlfile.

   All snapshots are created using the current state of the standby database, so they need to know all of the files that make up the standby. Create a binary backup of the control file to allow future creation of the CREATE CONTROLFILE script required for additional snapshots

   ```
   SQL> ALTER DATABASE BACKUP CONTROLFILE TO '/home/oracle/snap_tm/
   control_tm.ctl';
   ```

4. Create the `rename_files.sql` script to create the sparse data files for the snapshot.

   This script builds a series of RENAME statements to create the sparse data files to be used for the snapshot to apply redo received from the primary. Use a SQL statement similar to the following. Note that this statement uses the same directory structure as the original files, but the files will be created in the SPARSE disk group. The new file names will be created replacing '.' (periods) with '_' (underscores).

   ```
   set newpage 0
   set linesize 999
   set pagesize 0
   set feedback off
   set heading off
   set echo offset space 0
   set tab off
   set trimspool on
   spool rename_files.sql
   select 'EXECUTE dbms_dnfs.clonedb_renamefile ('||''''||name||''''||
   ','||''''||replace(replace(name,'.','_'),'DATA/','SPARSE/')||''''||
   ');' from v$datafile;
   exit
   ```

   This script produces output similar to the following:

   ```
   EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/DATAFILE/
   system.515.9304
   75939','+SPARSE/TM_STANDBY/DATAFILE/system_515_930475939');

   EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/
   429CE0836E0166ACE05382C8
   E50A1154/DATAFILE/system.567.930475945','+SPARSE/TM_STANDBY/
   ```

```
429CE0836E0166ACE053
82C8E50A1154/DATAFILE/system_567_930475945');

EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/DATAFILE/
sysaux.571.9304
75939','+SPARSE/TM_STANDBY/DATAFILE/sysaux_571_930475939');

EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/
429CE0836E0166ACE05382C8
E50A1154/DATAFILE/sysaux.516.930475945','+SPARSE/TM_STANDBY/
429CE0836E0166ACE053
82C8E50A1154/DATAFILE/sysaux_516_930475945');

EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/DATAFILE/
undotbs1.497.93
0475939','+SPARSE/TM_STANDBY/DATAFILE/undotbs1_497_930475939');

EXECUTE dbms_dnfs.clonedb_renamefile ('+DATA/TM_STANDBY/
429CE0836E0166ACE05382C8
E50A1154/DATAFILE/undotbs1.564.930475945','+SPARSE/TM_STANDBY/
429CE0836E0166ACE0
5382C8E50A1154/DATAFILE/undotbs1_564_930475945');
```

5. Using ASMCMD, create directories for all of the directories identified in the script `rename_files.sql`.

   When the `dbms_dnfs.clonedb_renamefile` function runs, it requires that all directory structures used for the files already exist in ASM.

   Use the output from the previous step to determine the structures required and then create them as needed. You can use ASMCMD to create the directories as in the following example:

```
cd ASMCMD [+] > cd sparse
ASMCMD [+sparse] > ls
ASMCMD [+sparse] > mkdir tm_standby
ASMCMD [+sparse] > cd tm_standby
ASMCMD [+sparse/tm_standby] > mkdir datafile
ASMCMD [+sparse/tm_standby] > mkdir 429DC0E1BCBD1B90E05382C8E50A8E80
ASMCMD [+sparse/tm_standby] > mkdir 429CE0836E0166ACE05382C8E50A1154
ASMCMD [+sparse/tm_standby] > cd 429DC0E1BCBD1B90E05382C8E50A8E80
ASMCMD [+sparse/tm_standby/429DC0E1BCBD1B90E05382C8E50A8E80] > mkdir
datafile
ASMCMD [+sparse/tm_standby/429DC0E1BCBD1B90E05382C8E50A8E80] > cd ../
429CE0836E0166ACE05382C8E50A1154
ASMCMD [+sparse/tm_standby/429CE0836E0166ACE05382C8E50A1154] > mkdir
datafile
```

9.7.7.2 Task 2: Configure the Sparse Test Master and Sparse Files on the Standby Site

In this task you convert the standby into a test master and create sparse files to receive and apply redo from the primary database.

Note:

During this process you will not be creating a full snapshot database, you will be using portions of the existing standby database and adding sparse data files to the standby. The standby database controlfile will be modified to use the sparse files that are added. Going forward, the same standby instance is used, but redo apply will use the sparse files to store changes, leaving the original standby data files to serve as a sparse test master files for snapshots.

The existing data files are used to support full database snapshots with data as of the point in time the process was run.

1. Shutdown all instances of the TM_STANDBY database.

   ```
   $ srvctl stop db -d tm_standby -o abort
   ```

2. Using SQL*Plus, start one of the TM_STANDBY instances in mount mode.

   ```
   SQL> startup mount
   ```

3. Change the DB_CREATE_FILE_DEST setting in the standby instance to point to the SPARSE disk group.

   This ensures all new data files that are created will reside in the SPARSE disk group. To perform this step you must disable standby file management.

   ```
   SQL> alter system set standby_file_management='MANUAL';
   SQL> alter system set db_create_file_dest='+SPARSE';
   ```

4. Run the `rename_files.sql` script created in Task 1 against the standby database.

   Running the script renames the data files in the TM_STANDBY controlfile and creates the sparse files for the snapshot.

   ```
   SQL> @rename_files
   ```

5. Re-enable standby file management.

   Completing this step ensures that all new data files added to the primary will automatically be created by the standby when it receives the redo to create the data file.

   ```
   SQL> alter system set standby_file_management='AUTO';
   ```

6. Enable redo apply on TM_STANDBY.

Completing this step applies redo to the snapshot, keeping it current and preparing for the next round of snapshot creation.

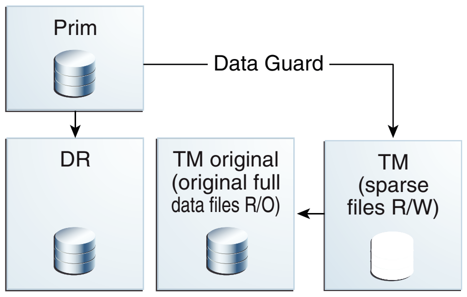
```
DGMGRL> edit database tm_standby set state='APPLY-ON';
```

7. Restart the remaining instances in mount mode.

```
$ srvctl start db -d tm_standby -o mount
```

> **Note:**
>
> If you plan to clone a local pluggable database (PDB) at the primary database, then enable Active Data Guard mode at the standby. This requires an Active Data Guard license.

Figure 9-15 Configuration with Test Master Files and Sparse Files for Redo Apply



9.7.7.3 Task 3: Create Full Database Snapshots Using the New Sparse Test Master

At this time you can create full snapshots against the original files of the standby database as described in Creating a Snapshot of a Full Database.

You must use the backup controlfile created in step 2 of Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master to build the CREATE CONTROLFILE statement. To use the file, you can create a temporary database instance to mount the controlfile and run the 'backup controlfile to trace' command.

1. Create a small PFILE file for the instance to use.

   At a minimum, the PFILE file should contain the following parameters:

```
control_files='/home/oracle/snap_tm/control_tm.ctl'  # This should be the
control file name created above
db_name=primary              # This should be the db_name used in the Data
Guard configuration
db_unique_name=temp          # This should be a unique name for a database
instance on this host
sga_target=5g                # Provide enough memory to start the instance
```

2. Set your environment to point to a unique ORACLE_SID.

```
$ export ORACLE_SID=temp
```

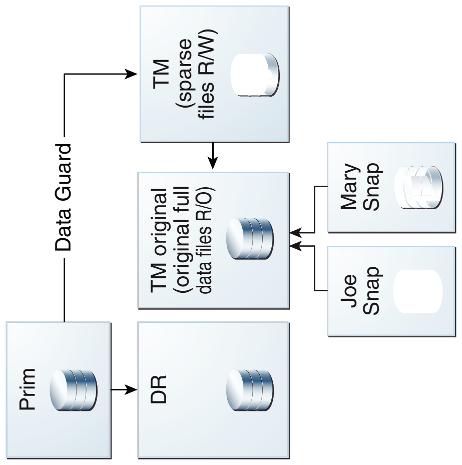
3. Using SQL*Plus, start the instance in mount mode using the PFILE created in Step 1.

```
SQL> startup mount pfile='/home/oracle/snap_tm/pfile.ora'
```

4. Build the create controlfile statement and the rename files script.

   Use steps 1 and 2 in Creating a Snapshot of a Full Database to build the CREATE CONTROLFILE statement and the rename files script. The rename files script created in step 4 of Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master can be used, but you must modify the directory structure of the sparse files to be created.

Figure 9-16 Configuration After Creating Snapshots



9.7.7.4 Task 4: Create a New Sparse Test Master Using a Previously Created Sparse Test Master

Create a new set of snapshots to provide a new test master and new sparse files to contain changes from the primary.

It is possible that at periodic intervals you will want to use the standby database to create additional snapshots without having to build a complete copy of the test master. You can repeat the process performed in the previous three tasks to do just that, taking advantage of the hierarchical snapshot functionality. The new test master is built on top of the latest existing snapshot that is applying redo. This snapshot becomes read-only and a new snapshot is built to continue the redo apply processing.

Do the following to configure the standby for the new test master time line:

1. Perform the steps from Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master with the following changes:

   The procedure is generally the same, however in this case the test master already uses sparse files.

   a. In Step 2, change all the commands that alter the DATA disk group to instead alter the SPARSE disk group.

      If the SPARSE disk group is already appropriately configured, you can skip the corresponding `ALTER DISKGROUP ... SET ATTRIBUTE` or `ALTER DISKGROUP ... ADD USER` command.

b. In Step 4, supply a different name for the snapshot.

You are creating a new snapshot, so the files need unique names from what was previously used. As a suggestion, you can append an identifier to the end of the file name to help identify it with the snapshot to be built. For example, if this was the original command:

```
EXECUTE dbms_dnfs.clonedb_renamefile ('+SPARSE/TM_STANDBY/DATAFILE/
system_515
_930475939','+SPARSE/TM_STANDBY/DATAFILE/system_515_930475939');
```

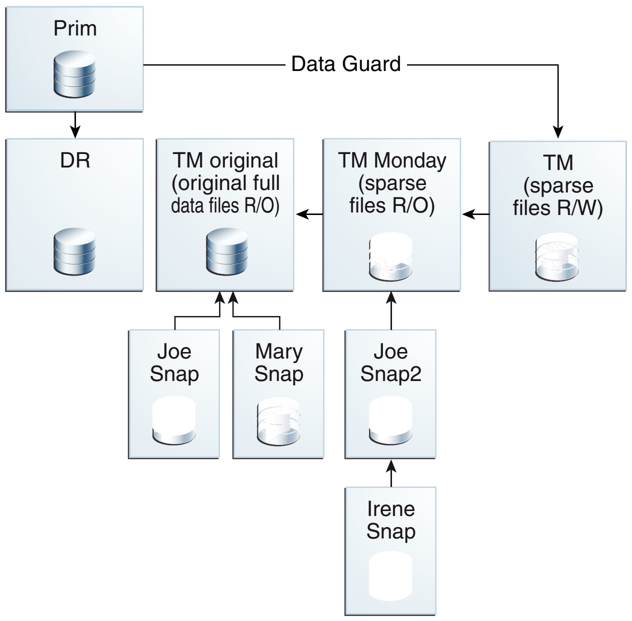
You can add an identifier to the end of the file name to create a unique file name, as shown here:

```
EXECUTE dbms_dnfs.clonedb_renamefile ('+SPARSE/TM_STANDBY/DATAFILE/
system_515
_930475939','+SPARSE/TM_STANDBY/DATAFILE/
system_515_930475939_Dec_15_16');
```

This step must be repeated for each statement for the `rename_files.sql` script.

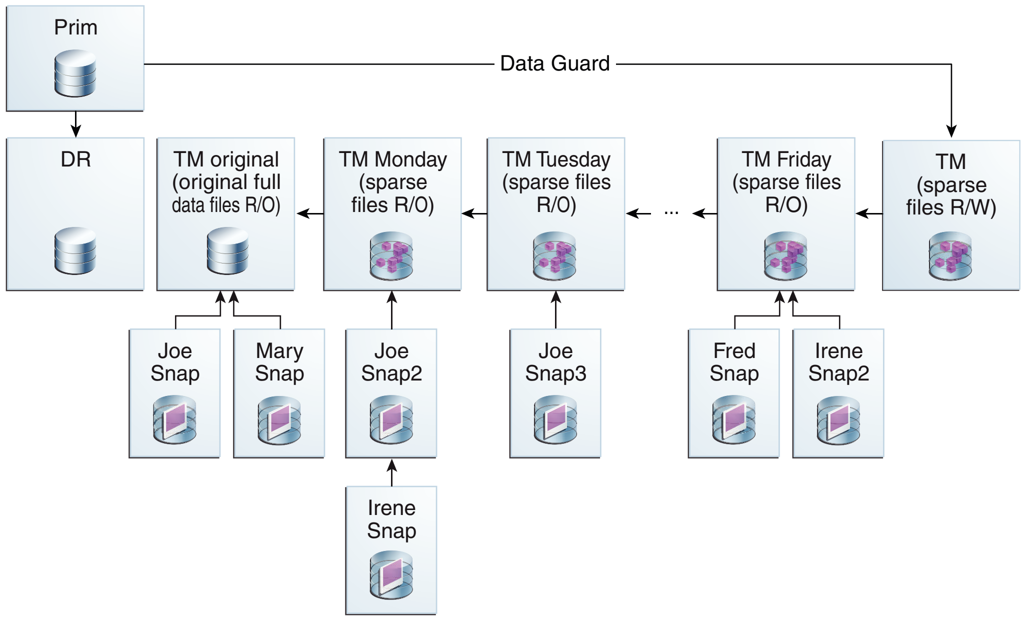
2. Perform the steps as described in Task 2: Configure the Sparse Test Master and Sparse Files on the Standby Site.

Figure 9-17 Configuration After Repeating the Process



The process to create a new sparse test master can be repeated up to 9 times, which would create an environment 10 levels deep with the original standby data files and 9 hierarchical snapshots. When repeating the process for the ninth time, do not create a new snapshot to receive the redo from the primary database.

Figure 9-18 Potential Future Configuration


Prim
Data Guard
DR
TM original (original full data files R/O)
TM Monday (sparse files R/O)
TM Tuesday (sparse files R/O)
...
TM Friday (sparse files R/O)
TM (sparse files R/W)
Joe Snap
Mary Snap
Joe Snap2
Joe Snap3
Fred Snap
Irene Snap2
Irene Snap

If you reach the maximum 10 levels, you have multiple options:

- If you have enough space to maintain multiple copies of the standby database and snapshots, then create a new tree of hierarchical snapshots based on another standby database. The original standby data files and snapshots can remain as long as required.
- If you do not have enough space to maintain multiple copies of the standby database and snapshots, then:
  1. Delete the snapshots and their associated data files, including any snapshots used as test masters.
  2. Refresh the standby.
  3. Create a new tree of hierarchical snapshots.
- Create a new standby database on a different environment and create a new tree of hierarchical snapshots.

Related Topics

- Task 1: Prepare the Standby Database to Be Used as a Sparse Test Master
- Task 2: Configure the Sparse Test Master and Sparse Files on the Standby Site
  In this task you convert the standby into a test master and create sparse files to receive and apply redo from the primary database.

9.7.8 Creating Sparse Test Masters for PDBs

This procedure creates a hierarchical snapshot tree or sparse test masters manually for a pluggable database (PDB) in an Oracle Multitenant database.

The test master must be closed while making the daily reference snapshot. The downtime is very short (less than 5 minutes). You can use a replication mechanism, such as Oracle GoldenGate, to keep the sparse test master current with the production PDB. For more information about configuring Oracle GoldenGate with PDBs, see Configuring Oracle

GoldenGate in a Multitenant Container Database in *Oracle GoldenGate Oracle Installation and Setup Guide*. The following example assumes you are using Oracle GoldenGate.

STEP 1: Create First Test Master PDB From the PROD PDB

This is a traditional PDB clone operation to instantiate the test master PDB. Once the clone completes, you configure Oracle GoldenGate to extract changes from the `PRODPDB1` PDB in production and replicate these changes to the test master `TMPDB1` PDB.

1. Run the following commands on the `PROD` container database (CDB) root:

   ```
   PRODCDB> alter pluggable database prodpdb1 close;

   PRODCDB> alter pluggable database prodpdb1 open read only;
   ```

2. Run the following commands from the test master CDB root:

   ```
   TMCDB> create database link PROD_DBLINK
   connect to system identified by password using 'PROD_CDB';

   TMCDB> create pluggable database TMPDB1
   from PRODPDB1@PROD_DBLINK;

   TMCDB> alter pluggable database TMPDB1 open;
   ```

3. Configure Oracle GoldenGate so that the changes made at the `PRODPDB1` PDB will be extracted, replicated and applied to the `TMPDB1` PDB. After configuring the extract and replicat and starting the extract process, open `PRODPDB1` PDB in read write mode.

   Note:

   The `PRODPDB1` PDB cannot be opened for changes until after Oracle GoldenGate has been configured and the extract process started.

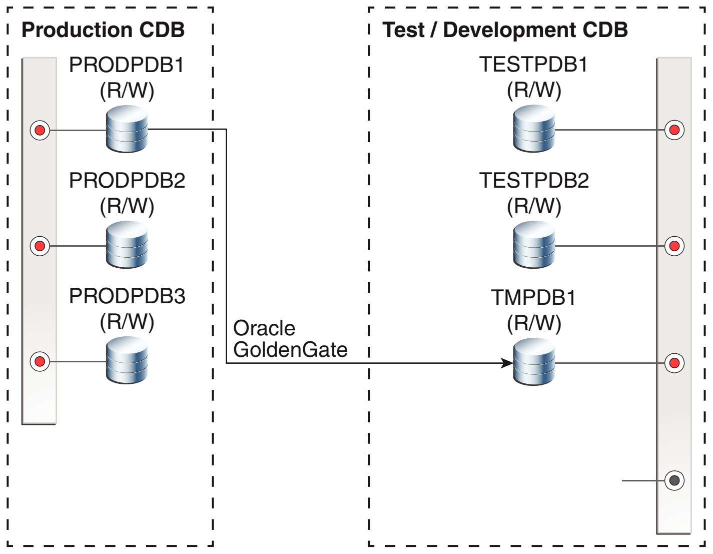
   ```
   PRODCDB> alter pluggable database PRODPDB1 close;
   PRODCDB> alter pluggable database PRODPDB1 open;
   ```

At this point, you have a full copy of `PRODPDB1` on test master as `TMPDB1` receiving all data changes made at `PRODPDB1`.

Note:

Oracle GoldenGate does not replicate data dictionary changes such as CREATE TABLESPACE or ADD DATAFILE. Only schema changes are replicated from `PRODPDB1` to `TMPDB1`.

Figure 9-19 TMPDB1 Created From the PRODPDB1 Pluggable Database



Although `TMPDB1` can be opened in read/write mode, you should leave it in read-only mode because the only changes it should receive are from `PRODPDB1` through Oracle GoldenGate.

To create a snapshot from `TMPDB1`, the test master PDB must be opened in read-only mode. To provide a test master PDB from which you can create snapshots, and a test master PDB that is kept current with its source, you need two PDBs. The next step shows how you can accomplish this.

STEP 2: Create Daily Read-Only Snapshot and Move TMPDB1 PDB to a New Sparse Test Master PDB

This step creates a read-only snapshot PDB that acts as a test master. You can then create read/write snapshot PDBs from this read-only snapshot PDB every day. The main steps are:

- Create a (daily) read-only snapshot PDB that you can make available to private read/write clients.
- Create a new sparse `TMPDB1` PDB pointing back to the read-only daily snapshot PDB. The new `TMPDB1` PDB also accepts and applies changes from `PRODPDB1`.

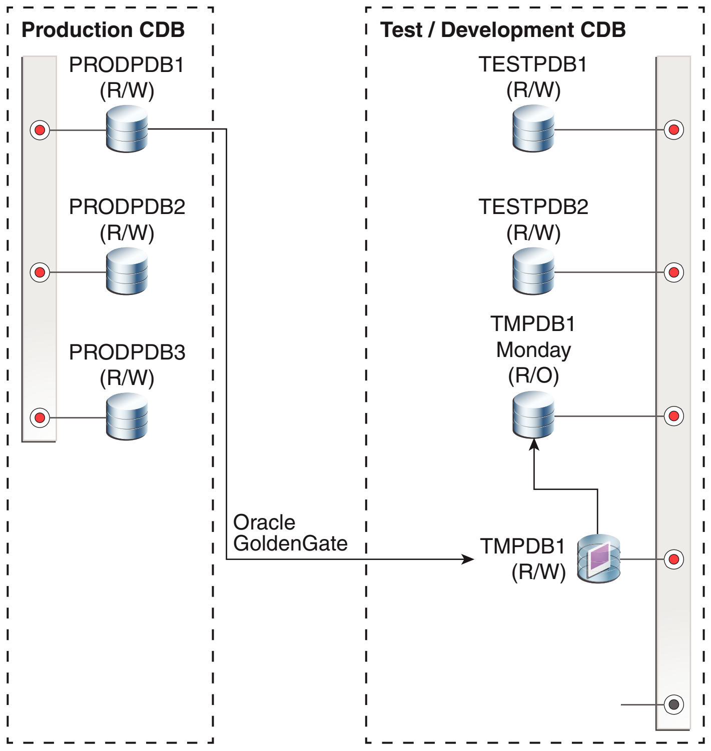
Connect to the `TMPDB1` PDB, then run the following commands:

```
TMCDB> alter session set container = CDB$ROOT;

# Stop the Oracle GoldenGate replicat process at the Test Master database.
This allows
# all changes made at PRODPDB1 to continue to be extracted and then applied
to
# TMPDB1 when the replicat process is restarted.

# Close the test master PDB.
TMCDB> alter pluggable database TMPDB1 close;

# Write the test master PDB metadata to an XML file.
TMCDB> alter pluggable database TMPDB1 unplug into
       '/home/oracle/snapshot/TMPDB1_monday.XML';
```

```
# Drop the test master PDB, but keep the data files.
TMCDB> drop pluggable database TMPDB1 keep datafiles;

# Create a TMPDB1_MONDAY PDB using the XML file you just created.
#Use the NOCOPY clause to reuse the original data files.

TMCDB> create pluggable database TMPDB1_MONDAY using
       '/home/oracle/snapshot/TMPDB1_monday.XML' nocopy;

# Open the new TMPDB1_MONDAY PDB. The PDB must be opened
# once in read/write mode to complete the creation process.

TMCDB> alter pluggable database TMPDB1_MONDAY open;
TMCDB> alter pluggable database TMPDB1_MONDAY close;
TMCDB> alter pluggable database TMPDB1_MONDAY open read only;

# Create the new TMPDB1 PDB to receive changes from PRODPDB1. This PDB
# must have the same name as the original test master PDB to ensure no
# changes are required to the Oracle GoldenGate configuration.

TMCDB> create pluggable database TMPDB1 from TMPDB1_MONDAY
        create_file_dest='+SPARSE'
        snapshot copy;

# Open the new TMPDB1 PDB. The PDB must be opened once in read/write
# mode to complete the PDB creation process.

TMCDB> alter pluggable database TMPDB1 open;
TMCDB> alter pluggable database TMPDB1 close;
TMCDB> alter pluggable database TMPDB1 open read only;


# Restart the Oracle GoldenGate replicat process to the new TMPDB1
# PDB. The Oracle GoldenGate replicat process now applies changes from
# PRODPDB1 to the TMPDB1 snapshot and all changes are written to
# sparse files.
```

The following figure shows the `TMPDB1` created from `TMPDB1_MONDAY`. The original TMPDB1 has been renamed to TMPDB1_Monday as part of the `DROP PLUGGABLE DATABASE`/`CREATE PLUGGABLE DATABASE` steps listed above. The new TMPDB1 is a sparse snapshot pluggable database that, until any changes are made to TMPDB1, looks exactly like TMPDB1_Monday. Oracle GoldenGate applies redo to the new TMPDB1 snapshot without having to make any changes to the replicat configuration

Figure 9-20 TMPDB1 Created From TMPDB1_MONDAY



STEP 3: Create Read/Write Snapshot From TMPDB1_MONDAY

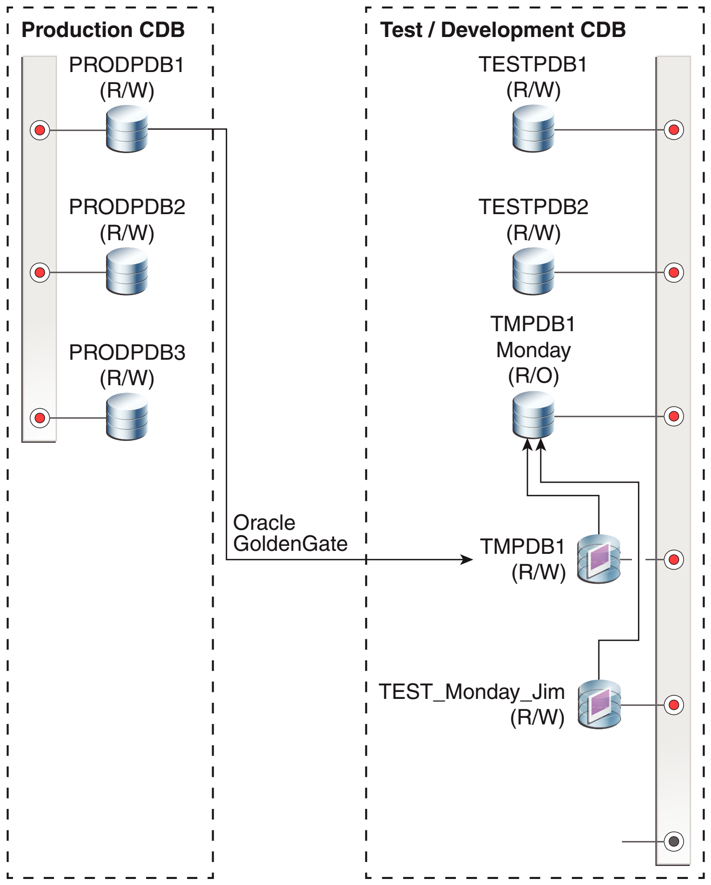
You create the snapshots from `TMPDB1_MONDAY`, not from `TMPDB1`. This allows `TMPDB1` to continue receiving and applying changes from `PRODPDB1`.

Connect to the `TMPDB1_MONDAY` PDB, then run the following commands:

```
TMCDB> alter session set container = cdb$ROOT;

TMCDB> create pluggable database TEST_MONDAY_JIM from TMPDB1_MONDAY
   create_file_dest='+SPARSE'
   snapshot copy;

TMCDB> alter pluggable database TEST_MONDAY_JIM open;
```

The following figure shows the TEST_MONDAY_JIM snapshot PDB created from TMPDB1_MONDAY. TEST_MONDAY_JIM uses TMPDB1_MONDAY as its parent so all data in TMPDB1_MONDAY_JIM is that same as the data in TMPDB1_MONDAY until changes are made to the snapshot PDB. Oracle GoldenGate continues to receive and apply redo to TMPDB1.

Figure 9-21 TEST_MONDAY_JIM Created From TMPDB1_MONDAY



When you need to create another test master and snapshot, you just need to repeat Step 2. For example, to create a test master on Tuesday, you can do the following:

Start a SQL*Plus session for the `TMPDB1` PDB.

```
TMCDB> alter session set container = CDB$ROOT;

# Stop the Oracle GoldenGate replicat process from applying changes to
# TMPDB1

# Close the test master PDB
TMCDB> alter pluggable database TMPDB1 close;

# Write the test master PDB metadata to an XML file
TMCDB> alter pluggable database TMPDB1 unplug into
'/home/oracle/snapshots/TMPDB1_tuesday.XML';

# Drop the test master PDB, but keep the data files
TMCDB> drop pluggable database TMPDB1 keep datafiles;

# Create a TMPDB1_TUESDAY PDB from the XML file
TMCDB> create pluggable database TMPDB1_TUESDAY using
'/home/oracle/snapshot/TMPDB1_tuesday.XML' nocopy;
```

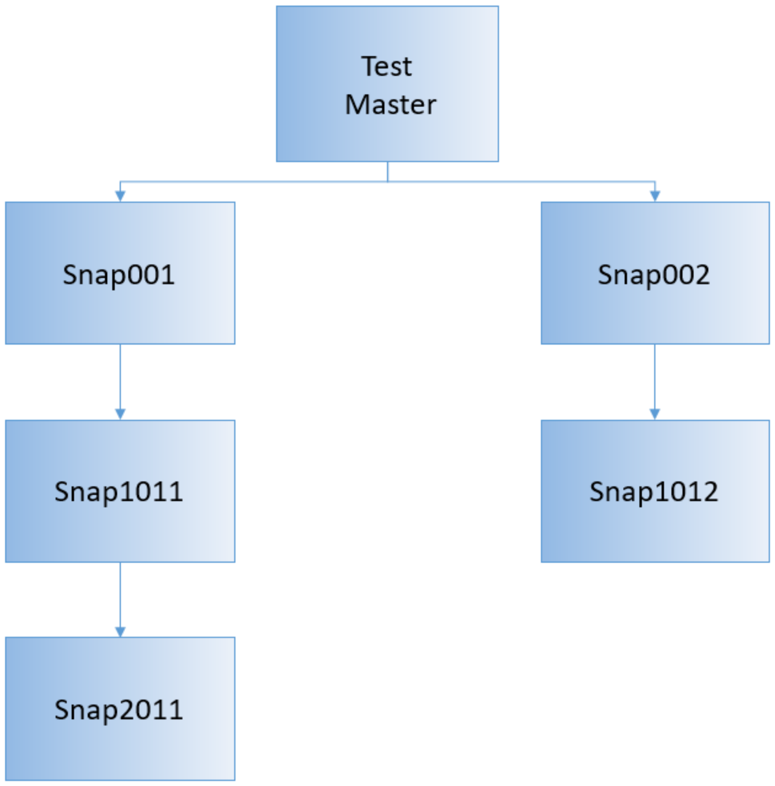
```
# Open the new TMPDB1_TUESDAY PDB
TMCDB> alter pluggable database TMPDB1_TUESDAY open;
TMCDB> alter pluggable database TMPDB1_TUESDAY close;
TMCDB> alter pluggable database TMPDB1_TUESDAY open read only;

# Create the new TMPDB1 PDB as a snapshot PDB
TMCDB> create pluggable database TMPDB1 from TMPDB1_TUESDAY
   create_file_dest='+SPARSE'
   snapshot copy;

# Open the TMPDB1 PDB
TMCDB> alter pluggable database TMPDB1 open;
TMCDB> alter pluggable database TMPDB1 close;
TMCDB> alter pluggable database TMPDB1 open read only;

# Restart the Oracle GoldenGate replicat process to apply changes to
# the new TMPDB1
```

You can now create read/write snapshot PDBs from TMPDB1_TUESDAY, similar to Step 3 above. The same as with full database sparse test masters, you can repeat this process up to 9 times in total before needing to either create a new TMPDB1 test master or drop and recreate the original TMPDB1 to begin building a new hierarchical snapshot tree.

9.7.9 Determining All Snapshots Associated with a Test Master

Use this query to discover all of the children associated with a test master.

Consider the following configuration for a test master with multiple children.



You can use a query of the associated SYSTEM data files for each database to list all the children within the same tree. The query selects just the SYSTEM data file for each database or PDB. The data file must exist for all clones and parents, and there should be only 1 data file for each. The `START WITH` clause provides a starting point of a file that is not a cloned file, which is the original test master parent.

Connect to the Oracle ASM instance and run this command as the SYSASM user.

```
SELECT clonefilename "Child", snapshotfilename "Parent"
FROM v$clonedfile
WHERE LOWER(snapshotfilename) LIKE '%system.%'
START WITH snapshotfilename NOT IN (SELECT clonefilename FROM v$clonedfile)
CONNECT BY LOWER(clonefilename) = PRIOR (snapshotfilename);
```

The results of this query for database-based snapshots would be similar to the following:

```
Child
  Parent
-----------------------------------------------------------
  ----------------------------------------------------------------
+SPARSE/SNAP001/DATAFILE/SYSTEM.256.1011532891
  +DATA/TESTMASTER/DATAFILE/system.270.1011530981
+SPARSE/SNAP002/DATAFILE/SYSTEM.265.1011532969
  +DATA/TESTMASTER/DATAFILE/system.270.1011530981
+SPARSE/SNAP1011/DATAFILE/SYSTEM.270.1011533005
  +SPARSE/SNAP001/DATAFILE/system.256.1011532891
+SPARSE/SNAP1012/DATAFILE/SYSTEM.275.1011780925
  +SPARSE/SNAP001/DATAFILE/system.256.1011532891
```

```
+SPARSE/SNAP2011/DATAFILE/SYSTEM.281.1011781103
  +SPARSE/SNAP1011/DATAFILE/system.270.1011533005
```

If you created folders in Oracle ASM that contained the database name, as shown in the above result, then the database name in the `CLONEFILENAME` string is the snapshot, and the database name in the `SNAPSHOTFILENAME` string is the master for that snapshot.

The results of this query for PDB-based snapshots would be similar to the following:

```
CLONEFILENAME
  SNAPSHOTFILENAME
------------------------------------------------------------------------------
---

------------------------------------------------------------------------------
---
+SPARSEC1/CDB001/8BDBC355D43721F5E053412E850AB5D1/DATAFILE/
SYSTEM.256.1011532891
  +DATAC1/CDB001/8BDBC355D42D21F5E053412E850AB5D1/DATAFILE/
system.270.1011530981
+SPARSEC1/CDB001/8BDBC355D43E21F5E053412E850AB5D1/DATAFILE/
SYSTEM.265.1011532969
  +DATAC1/CDB001/8BDBC355D42D21F5E053412E850AB5D1/DATAFILE/
system.270.1011530981
+SPARSEC1/CDB001/8BDBC355D44021F5E053412E850AB5D1/DATAFILE/
SYSTEM.270.1011533005
  +SPARSEC1/CDB001/8BDBC355D43721F5E053412E850AB5D1/DATAFILE/
system.256.1011532891
+SPARSEC1/CDB001/8BDBC355D44821F5E053412E850AB5D1/DATAFILE/
SYSTEM.275.1011780925
  +SPARSEC1/CDB001/8BDBC355D43721F5E053412E850AB5D1/DATAFILE/
system.256.1011532891
+SPARSEC1/CDB001/8BDBC355D44D21F5E053412E850AB5D1/DATAFILE/
SYSTEM.281.1011781103
  +SPARSEC1/CDB001/8BDBC355D44021F5E053412E850AB5D1/DATAFILE/
system.270.1011533005
```

In this case, the folder name in Oracle ASM is the GUID associated with the PDB. To determine the name of teach snapshot PDB and its master, you must do the following:

1. Log in to the CDB that has the name shown in the results, for example, `CDB001`.
2. Run a query against the `CDB_PDBS` view to translate the GUIDs into the PDB names, as shown below:

   ```
   SELECT pdb_name, guid FROM CDB_PDBS
   WHERE guid IN
   ('8BDBC355D42D21F5E053412E850AB5D1','8BDBC355D43721F5E053412E850AB5D1'
   '8BDBC355D44821F5E053412E850AB5D1','8BDBC355D43E21F5E053412E850AB5D1',
   '8BDBC355D44021F5E053412E850AB5D1','8BDBC355D44D21F5E053412E850AB5D1');

   PDB_NAME                 GUID
   ----------------------- -----------------------------------
   TESTMASTER               8BDBC355D42D21F5E053412E850AB5D1
   SNAP001                  8BDBC355D43721F5E053412E850AB5D1
   SNAP1012                 8BDBC355D44821F5E053412E850AB5D1
   ```

```
    SNAP02                  8BDBC355D43E21F5E053412E850AB5D1
    SNAP1011                8BDBC355D44021F5E053412E850AB5D1
    SNAP2011                8BDBC355D44D21F5E053412E850AB5D1
```

Then use this information to determine the parent/child relationship among the PDBs in the original query results.

9.7.10 Doing a Sparse Copy

The ASM `cp` command copies a sparse file to a new destination. However, this operation copies the sparse file with all the blocks instantiated from the parent. The “sparse copy” feature enables you to do a sparse copy of a file.

You can have multiple ASM instances running at the same time. If an operation involves a source or a destination on a different ASM instance other than the one it is executing on, it is treated as a remote ASM instance. You can do a sparse copy on a local ASM instance, or between a local and a remote ASM instance. However, sparse copy does not work between two remote ASM instances.

To do a sparse copy, you use the new `--sparse` option in the existing ASM `cp` command. The syntax looks like the following:

```
ASMCMD> cp --sparse <src_sparse_file> <tgt_file>
```

A new ASM command called `setsparseparent` enables you to set the parent of a sparse file. If you do a sparse copy of a file to a sparse destination on a local ASM instance, its parent is set as part of the sparse copy operation. However, if the destination is on a remote ASM instance, you have to set its parent explicitly using the `setsparseparent` command.

The `setsparseparent` command requires sparse child file and parent file as parameters. It sets the parent of the sparse child file to the new parent file. The syntax looks like the following:

```
ASMCMD> setsparseparent <sparse_file> <parent_file>
```

The `cp` ASM command performs the following validations before doing a sparse copy operation. The operation is allowed only if it satisfies the following criteria:

- The source file must exist and must be a sparse file.
- If you specify multiple source sparse files, all of them must be on the same ASM instance.
- Copying multiple sparse files on a remote ASM instance to a destination on a local ASM instance and vice versa is allowed provided all source files are on the same ASM instance.
- Destination file should be backed by a sparse disk group. However, it can be a non-sparse file if event “KFTST_KFPKG_CP_SPARSE” is set. This event is required to validate sparse copy operation by merging and copying the files to a non-sparse destination.
- Both source and destination cannot be on a remote ASM instance. However, either source or destination can be on a remote ASM instance.
- If the destination is on a remote ASM instance, its file type cannot be validated and you have to ensure that it is backed by a sparse disk group. You also have to set the parent explicitly using the ASM `setsparseparent` command.
- If the destination is a non-sparse file and you run the `setsparseparent` command, the command will fail because the child file should be sparse. This is a second-level validation if the destination is a non-sparse file.

The `setsparseparent` ASM command performs the following validations before it sets the parent. The operation is allowed only if it satisfies the following criteria:

- The child file must exist and must be a sparse file.
- The parent file must exist. It can be a sparse or a non-sparse file.
- Parent and child files must be present on same ASM instance.

> **Note:**
>
> You have to ensure that the files you specify in the `setsparseparent` ASM command have a valid parent-child relationship. The command cannot perform this check for files on remote ASM instances. If the files do not have a valid parent-child relationship, then data integrity and corruption issues are likely to occur.

Example 1: The following ASM command copies sparse file “TBS_1.264.908376549” to the destination “+SPARSEDG/child_1”.

```
ASMCMD> cp --sparse +SPARSEDG/MERGE/DATAFILE/TBS_1.264.908376549 +SPARSEDG/child_1
```

Example 2: The following ASM command sets parent “tbs_1.269.908374993” for the sparse file “remote_child_10”.

```
ASMCMD> setsparseparent +SPARSEDG/remote_child_10 +DATAFILE/DATAFILE/tbs_1.269.908374993
```

Example 3: The following command copies sparse child files child_1, child_2 and child_3 to the destination directory +SPARSEDG.

```
ASMCMD> cp --sparse +SPARSEDG/DATAFILE/child_1 +SPARSEDG/DATAFILE/child_2 +SPARSEDG/
DATAFILE/child_3 +SPARSEDG/
```

9.8 Managing Sparse Griddisks

You can resize, recreate, or monitor the activity of sparse griddisks.

- Resizing the Virtual Space
  When `V$ASM_DISKGROUP.FREE_MB` or `V$ASM_DISK.FREE_MB` is running low, you need to increase the virtual address space.
- Resizing the Physical Space
- Monitoring Sparse Disk Group Utilization
- Repurposing Sparse Griddisks
  You can change sparse griddisks back to normal griddisks.

9.8.1 Resizing the Virtual Space

When `V$ASM_DISKGROUP.FREE_MB` or `V$ASM_DISK.FREE_MB` is running low, you need to increase the virtual address space.

1. To increase the size of the virtual space:

a. Run the following command on the cells, specifying all the grid disks for the SPARSE disk group:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL01,SPARSE_CD_01_CELL01,....,SPARSE_CD_11_CELL01
virtualSize=newSize
```

For example, on the first cell:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL01,SPARSE_CD_01_CELL01,SPARSE_CD_02_
CELL01,SPARSE_CD_03_CELL01,SPARSE_CD_04_CELL01,SPARSE_CD_05_CELL01,SPARS
E_CD_
06_CELL01,SPARSE_CD_07_CELL01,SPARSE_CD_08_CELL01,SPARSE_CD_09_CELL01,SP
ARSE_
CD_10_CELL01,SPARSE_CD_11_CELL01 virtualSize=12000G

GridDisk SPARSE_CD_00_CELL01 successfully altered
GridDisk SPARSE_CD_01_CELL01 successfully altered
...
```

For example, on the next cell:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL02,SPARSE_CD_01_CELL02,SPARSE_CD_02_
CELL02,SPARSE_CD_03_CELL02,SPARSE_CD_04_CELL02,SPARSE_CD_05_CELL02,SPARS
E_CD_
06_CELL02,SPARSE_CD_07_CELL02,SPARSE_CD_08_CELL02,SPARSE_CD_09_CELL02,SP
ARSE_
CD_10_CELL02,SPARSE_CD_11_CELL02 virtualSize=12000G

GridDisk SPARSE_CD_00_CELL02 successfully altered
GridDisk SPARSE_CD_01_CELL02 successfully altered
...
```

Note:

You must change the size of the grid disks on all cells before making any changes in Oracle ASM.

b. On an ASM instance, resize the disk group to this new size:

```
SQL> alter diskgroup SPARSE resize all size newSize;
```

For example:

```
SQL> alter diskgroup SPARSE resize all size 12000G;
```

2. To decrease the size of the virtual space:

a. On an ASM instance, resize the disk group to this new size:

```
SQL> alter diskgroup SPARSE resize all size newSize;
```

For example:

```
SQL> alter diskgroup SPARSE resize all size 8000G;
```

b. Run the following command on the cells, specifying all the grid disks for the SPARSE disk group:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL01,SPARSE_CD_01_CELL01,....,SPARSE_CD_11_CELL01
virtualSize=newSize
```

For example, on the first cell:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL01,SPARSE_CD_01_CELL01,SPARSE_CD_02_
CELL01,SPARSE_CD_03_CELL01,SPARSE_CD_04_CELL01,SPARSE_CD_05_CELL01,SPARS
E_CD_
06_CELL01,SPARSE_CD_07_CELL01,SPARSE_CD_08_CELL01,SPARSE_CD_09_CELL01,SP
ARSE_
CD_10_CELL01,SPARSE_CD_11_CELL01 virtualSize=8000G

GridDisk SPARSE_CD_00_CELL01 successfully altered
GridDisk SPARSE_CD_01_CELL01 successfully altered
...
```

For example, on the next cell:

```
CellCLI> alter griddisk
SPARSE_CD_00_CELL02,SPARSE_CD_01_CELL02,SPARSE_CD_02_
CELL02,SPARSE_CD_03_CELL02,SPARSE_CD_04_CELL02,SPARSE_CD_05_CELL02,SPARS
E_CD_
06_CELL02,SPARSE_CD_07_CELL02,SPARSE_CD_08_CELL02,SPARSE_CD_09_CELL02,SP
ARSE_
CD_10_CELL02,SPARSE_CD_11_CELL02 virtualSize=8000G

GridDisk SPARSE_CD_00_CELL02 successfully altered
GridDisk SPARSE_CD_01_CELL02 successfully altered
...
```

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

9.8.2 Resizing the Physical Space

Commencing with Oracle Exadata System Software 22.1.0, you can use the `sizeAllocated` grid disk attribute to determine the total size of materialized space used by data in a sparse grid disk. When `sizeAllocated` approaches the physical grid disk size, you need to increase the physical grid disk size to support further data growth.

Otherwise, you can determine the amount of available physical space in the sparse grid disks by examining the difference between the `TOTAL_MAT_MB` and `ALLOCATED_MAT_MB` column values in `V$ASM_DISK_SPARSE`.

To increase the physical size of the grid disks:

1. Ensure that there is free space available on the respective cell disks.

   For example:

   ```
   # dcli -g cell_group -l root "cellcli -e list celldisk attributes
   name,freespace"
   exa01celadm01: CD_00_exa01celadm01 0
   exa01celadm01: CD_01_exa01celadm01 0
   exa01celadm01: CD_02_exa01celadm01 0
   exa01celadm01: CD_03_exa01celadm01 0
   exa01celadm01: CD_04_exa01celadm01 0
   exa01celadm01: CD_05_exa01celadm01 0
   exa01celadm01: CD_06_exa01celadm01 0
   exa01celadm01: CD_07_exa01celadm01 0
   exa01celadm01: CD_08_exa01celadm01 0
   exa01celadm01: CD_09_exa01celadm01 0
   exa01celadm01: CD_10_exa01celadm01 0
   exa01celadm01: CD_11_exa01celadm01 0
   ...
   ```

   If there is no available free space, then you have to free up disk space being used by other grid disks, either by reducing the size of other grid disks or by removing unnecessary grid disks.

2. Run the `ALTER GRIDDISK` command on the cells, specifying the grid disks to resize and the new physical size for each grid disk:

   The command syntax is:

   ```
   CellCLI> alter griddisk gridDisk1,gridDisk2,...,gridDiskN
   size=newPhysicalSize
   ```

   Run the command on each cell.

   For example, on the first cell:

   ```
   CellCLI> alter griddisk
   data01_CD_00_exa01celadm01,data01_CD_01_exa01celadm01,
   data01_CD_02_exa01celadm01,data01_CD_03_exa01celadm01,data01_CD_04_exa01cel
   adm01,
   data01_CD_05_exa01celadm01,data01_CD_06_exa01celadm01,data01_CD_07_exa01cel
   adm01,
   data01_CD_08_exa01celadm01,data01_CD_09_exa01celadm01,data01_CD_10_exa01cel
   adm01,
   data01_CD_11_exa01celadm01 size=12000G
   ```

   Then on the next cell:

   ```
   CellCLI> alter griddisk
   data01_CD_00_exa01celadm02,data01_CD_01_exa01celadm02,
   ```

```
data01_CD_02_exa01celadm02,data01_CD_03_exa01celadm02,data01_CD_04_exa01cel
adm02,
data01_CD_05_exa01celadm02,data01_CD_06_exa01celadm02,data01_CD_07_exa01cel
adm02,
data01_CD_08_exa01celadm02,data01_CD_09_exa01celadm02,data01_CD_10_exa01cel
adm02,
data01_CD_11_exa01celadm02 size=12000G
```

And so on.

After you increase the physical size of the grid disks, the sparse disk group automatically consumes the additional space as required.

To shrink the physical size of the grid disks:

1. Check the amount of materialized space used by data in the sparse grid disks. You cannot resize a grid disk to be smaller than the current amount of materialized data.

   Commencing with Oracle Exadata System Software 22.1.0, check the `sizeAllocated` grid disk attribute.

   For example:

   ```
   # dcli -g cell_group -l root "cellcli -e list griddisk attributes
   name,sizeAllocated"
   exa01celadm01: data01_CD_00_exa01celadm01 1023.9375M
   exa01celadm01: data01_CD_01_exa01celadm01 1024.4375M
   exa01celadm01: data01_CD_02_exa01celadm01 1023.4375M
   exa01celadm01: data01_CD_03_exa01celadm01 1024.9375M
   exa01celadm01: data01_CD_04_exa01celadm01 1023.9375M
   exa01celadm01: data01_CD_05_exa01celadm01 1024.4375M
   exa01celadm01: data01_CD_06_exa01celadm01 1023.4375M
   exa01celadm01: data01_CD_07_exa01celadm01 1024.9375M
   exa01celadm01: data01_CD_08_exa01celadm01 1023.9375M
   exa01celadm01: data01_CD_09_exa01celadm01 1024.4375M
   exa01celadm01: data01_CD_10_exa01celadm01 1023.4375M
   exa01celadm01: data01_CD_11_exa01celadm01 1024.9375M
   ...
   ```

   Otherwise, examine the `ALLOCATED_MAT_MB` column value in `V$ASM_DISK_SPARSE`. For example:

   ```
   SQL> SELECT allocated_mat_mb FROM v$asm_disk_sparse
         WHERE group_number = spare_disk_group_number;
   ```

   If you need to shrink the grid disks to be smaller than the amount of currently materialized data, you must drop objects from the sparse disk group.

2. Run the `ALTER GRIDDISK` command on the cells, specifying the grid disks to shrink and the new physical size for each grid disk:

   The command syntax is the same as for increasing the grid disk size:

   ```
   CellCLI> alter griddisk gridDisk1,gridDisk2,...,gridDiskN
   size=newPhysicalSize
   ```

   Run the command on each cell.

For example, on the first cell:

```
CellCLI> alter griddisk
data01_CD_00_exa01celadm01,data01_CD_01_exa01celadm01,
data01_CD_02_exa01celadm01,data01_CD_03_exa01celadm01,data01_CD_04_exa01cel
adm01,
data01_CD_05_exa01celadm01,data01_CD_06_exa01celadm01,data01_CD_07_exa01cel
adm01,
data01_CD_08_exa01celadm01,data01_CD_09_exa01celadm01,data01_CD_10_exa01cel
adm01,
data01_CD_11_exa01celadm01 size=4000G
```

Then on the next cell:

```
CellCLI> alter griddisk
data01_CD_00_exa01celadm02,data01_CD_01_exa01celadm02,
data01_CD_02_exa01celadm02,data01_CD_03_exa01celadm02,data01_CD_04_exa01cel
adm02,
data01_CD_05_exa01celadm02,data01_CD_06_exa01celadm02,data01_CD_07_exa01cel
adm02,
data01_CD_08_exa01celadm02,data01_CD_09_exa01celadm02,data01_CD_10_exa01cel
adm02,
data01_CD_11_exa01celadm02 size=4000G
```

And so on.

Related Topics

- Resizing Grid Disks
  You can resize grid disks and Oracle ASM disk groups to shrink one with excess free space and increase the size of another that is near capacity.

9.8.3 Monitoring Sparse Disk Group Utilization

The `V$ASM_DISK` and `V$ASM_DISKGROUP` views contain information about the virtual size and utilization of the sparse ASM disk group. New views `V$ASM_DISK_SPARSE` and `V$ASM_DISKGROUP_SPARSE` contain information about the actual size and utilization of the sparse ASM disk group. `V$ASM_DISK_SPARSE` also contains performance and usage metrics.

The following table describes the columns in `V$ASM_DISK_SPARSE`:

Table 9-1 V$ASM_DISK_SPARSE Columns and Descriptions

Column	Description
`GROUP_NUMBER`	The number of the disk group containing the disk.
`DISK_NUMBER`	The number assigned to the disk within this disk group.
`INCARNATION`	The incarnation number for the disk.
`ALLOCATED_MAT_MB`	The total used physical and materialized capacity on the disk.
`TOTAL_MAT_MB`	The total physical capacity on the disk.
`SPARSE_READS`	The total number of I/O read requests on non-materialized regions of the disk.

Table 9-1 (Cont.) V$ASM_DISK_SPARSE Columns and Descriptions

Column	Description
`SPARSE_BYTES_READ`	The total number of bytes read from non-materialized regions of the disk.
`SPARSE_READ_TIME`	The time taken by sparse read I/O operations.

The following table describes the columns in `V$ASM_DISKGROUP_SPARSE`:

Table 9-2 V$ASM_DISKGROUP_SPARSE Columns and Descriptions

Column	Description
`GROUP_NUMBER`	The cluster-wide number assigned to the disk group.
`ALLOCATED_MAT_MB`	The total used physical and materialized capacity of the disk group.
`TOTAL_MAT_MB`	The total physical capacity of the disk group.

The following example shows the used (allocated) space and the total space for disks in a specific disk group:

```
SQL> select
     DISK_NUMBER          dsk_num,
     ALLOCATED_MAT_MB     alloc,
     TOTAL_MAT_MB         total
from V$ASM_DISK_SPARSE
where GROUP_NUMBER = 5;

DSK_NUM    ALLOC      TOTAL
---------- ---------- ----------
         0       5536      57336
         1       5424      57336
         2       5532      57336
         3       5424      57336
         4       5424      57336
```

In the following example, sparse ASM grid disks were created with an actual size of 56 GB and a virtual size of 560 GB. When you query `V$ASM_DISK`'s `OS_MB` and `TOTAL_MB` columns, you can see the virtual size of 573440 MB (573440 MB / 1024 = 560 GB).

```
SQL> select os_mb, total_mb from v$asm_disk where group_number=4;

     OS_MB   TOTAL_MB
---------- ----------
    573440     573440
    573440     573440
    573440     573440
```

Querying `V$ASM_DISK_SPARSE` for `TOTAL_MB`, you can see the actual size of the ASM grid disk available for use. Note that each ASM grid disk stores metadata information of approximately 2 MB per 16 GB of space allocated to the sparse ASM grid disk. For 56 GB allocated per grid disk in this example, 8 MB of space is reserved for sparse disk metadata (57336 MB + 8 MB = 57344 MB / 1024 = 56 GB).

```
SQL> select total_mb from v$asm_disk_sparse where group_number=4;

 TOTAL_MB
```

```
---------
 57336
 57336
 57336
```

9.8.4 Repurposing Sparse Griddisks

You can change sparse griddisks back to normal griddisks.

If you previously created sparse griddisks, but now want to use them as normal griddisks, you can drop and recreate the disks.

1. Drop the snapshot database that is currently using the sparse griddisks.

   ```
   RMAN> startup mount force;
   RMAN> delete database;
   ```

2. Using SQL*Plus or ASMCMD, drop the Oracle ASM disk group that contains the sparse griddisks.

   ```
   SQL> DROP DISKGROUP sparse INCLUDING CONTENTS force;
   ```

3. Using CellCLI, drop the griddisks in the storage cells.

   ```
   cellcli -e drop griddisk all harddisk prefix=SPARSEC1
   ```

4. Recreate the griddisks.

   When you create the griddisks, use a similar size as the other disks and add back to the disk group of your choice. Do not specify the sparse attribute. See CREATE GRIDDISK for the command syntax.

5. Add the recreated griddisks to an Oracle ASM disk group.

   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*';
   ```

Related Topics

- Creating a Sparse Grid Disk

9.9 Monitoring Exadata Snapshots Using Database Statistics and Wait Events

The following table describes database specific statistics that are useful for monitoring Exadata Snapshots. The statistics are available in various dynamic performance views, including `V$SYSSTAT`, and may be displayed in the Global Activity Statistics or Instance Activity Statistics section of an AWR report.

Statistic	**Description**
`physical read snap IO requests base`	The number of physical I/Os on the parent or base file.
`physical read snap IO requests copy`	The number of physical I/Os on the snapshot file.

Statistic	Description
`physical read snap bytes base`	The number of bytes read from the parent or base file
`physical read snap bytes copy`	The number of bytes read from the snapshot file.
`physical read snap IO requests new allocation`	The number of new allocations on the snapshot file.
`physical read snap IO requests no data`	The number of physical read I/O requests for which no physical I/O is done on the child file level.

The following table describes specific database wait events that are useful for monitoring Exadata Snapshots. The wait events are visible in various dynamic performance views, including `V$SESSION`, `V$SYSTEM_EVENT` and `V$SESSION_EVENT`, and may be displayed in the Wait Event sections of the AWR report.

Wait Event	Description
`cell physical read no I/O`	The wait event appears when a read is done from a sparse disk group, and did not return any data.

The availability of a specific statistic or wait event is subject to the version of Oracle Database being used.

A

Upgrading Oracle Exadata System Software

Oracle Exadata System Software, operating systems, and component firmware are upgraded and patched using Oracle utilities.

Patches include all required software updates, operating system updates, and firmware updates. Use the following utilities to upgrade and patch Oracle Exadata:

- The `patchmgr` utility is used for Oracle Exadata System Software, and RDMA Network Fabric switches.
- The DB Node Update utility (`dbnodeupdate.sh` utility) is used for database servers.

In addition, Oracle recommends running the Oracle EXAchk utility before and after performing planned maintenance.

Upgrade and patching information is available from My Oracle Support note 888828.1. Additional information is available in the patch README, and associated support notes.

For detailed information about updating the Oracle Exadata System Software, operating systems, and component firmware, refer to Updating Exadata Software.

Related Topics

- Exadata Database Machine and Exadata Storage Server Supported Versions (My Oracle Support Doc ID 888828.1)

B

Exadata-Specific Background Processes

The following background processes are part of Oracle Database and Oracle ASM instances in an Exadata environment:

- diskmon Process
- XDMG Process
- XDWK Process

B.1 diskmon Process

The `diskmon` process is a fundamental component of Oracle Exadata System Software, 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 (CRS). This process is important for Oracle Exadata System Software 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 Diagnostic and Alert Log Data
  - Overview of Oracle Clusterware Platform-Specific Software Components
- *Oracle Database Concepts* for information about Oracle Database processes
- *Oracle Database Reference* for a description of the `V$BGPROCESS` view that displays information about background processes

B.2 XDMG Process

The `XDMG` (Exadata Automation Manager) process initiates automation tasks used for monitoring storage. This background process monitors all configured Oracle Exadata Storage Servers 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.

B.3 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.

C

Exadata-Specific Information in Oracle Database Dictionary Views

- Oracle Database Dictionary Views
- Automatic Workload Repository Views

C.1 Oracle Database Dictionary Views

The Oracle Database dictionary contains views that are specific to Oracle Exadata.

- Using the V$CELL and GV$CELL Views to Display Oracle Exadata Storage Server Identification
  The `V$CELL` view provides identifying information about cells.
- About the V$CELL_ Views
  The views prefixed with `V$CELL_` contain internal statistical information, which is used by Oracle Support Services.
- Using V$BACKUP_DATAFILE with Oracle Exadata Storage Server
  The `V$BACKUP_DATAFILE` view contains columns relevant to Oracle Exadata Storage Server during Oracle Recovery Manager (RMAN) incremental backups.
- Using V$ASM_DISK_SPARSE and V$ASM_DISKGROUP_SPARSE to Monitor Sparse Disks
  The `V$ASM_DISK_SPARSE` and `V$ASM_DISKGROUP_SPARSE` views contain information about sparse disks.

C.1.1 Using the V$CELL and GV$CELL Views to Display Oracle Exadata Storage Server Identification

The `V$CELL` view provides identifying information about cells.

Table C-1 V$CELL View Columns and Descriptions

Column	Description
`CELL_HASHVAL`	A numeric hash value for the cell. For example: `138889696` **Note:** The cell hash value is often contained in the `P1` argument for cell-related wait events in the `V$SESSION_WAIT` and `V$ACTIVE_SESSION_HISTORY` views. You can join to this column to identify the corresponding cell.
`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.
`CELL_TYPE`	The type of storage cell. Either `EXADATA` or `BDSQL`.

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.

C.1.2 About the V$CELL_ Views

The views prefixed with `V$CELL_` contain internal statistical information, which is used by Oracle Support Services.

This includes the following views:

- `V$CELL_CONFIG`
- `V$CELL_CONFIG_INFO`
- `V$CELL_THREAD_HISTORY`
- `V$CELL_DB`
- `V$CELL_DISK`
- `V$CELL_DISK_HISTORY`
- `V$CELL_GLOBAL`
- `V$CELL_GLOBAL_HISTORY`
- `V$CELL_IOREASON`
- `V$CELL_IOREASON_NAME`
- `V$CELL_METRIC_DESC`
- `V$CELL_REQUEST_TOTALS`
- `V$CELL_STATE`
- `V$CELL_THREAD_HISTORY`
- `V$CELL_OFL_THREAD_HISTORY`
- `V$CELL_OPEN_ALERTS`

These views contain instantaneous statistical information that may be easily misinterpreted or incomplete when viewed in isolation. Instead, use the corresponding Automatic Workload Repository (AWR) views, which are prefixed with `DBA_HIST_CELL_`. Or, use the AWR report.

Related Topics

- Automatic Workload Repository Views

C.1.3 Using V$BACKUP_DATAFILE with Oracle Exadata Storage Server

The `V$BACKUP_DATAFILE` view contains columns relevant to Oracle Exadata Storage Server during Oracle Recovery Manager (RMAN) incremental backups.

Table C-2 V$BACKUP_DATAFILE Columns and Descriptions

Column	Description
`BLOCKS`	Size of the backup data file in blocks.

Table C-2 (Cont.) V$BACKUP_DATAFILE Columns and Descriptions

Column	Description
`BLOCKS_READ`	The 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 Oracle Exadata Storage Server 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 Oracle Exadata System Software 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.

Related Topics

- *Oracle Database Reference*
- *Oracle Database Backup and Recovery User's Guide*

C.1.4 Using V$ASM_DISK_SPARSE and V$ASM_DISKGROUP_SPARSE to Monitor Sparse Disks

The `V$ASM_DISK_SPARSE` and `V$ASM_DISKGROUP_SPARSE` views contain information about sparse disks.

Table C-3 V$ASM_DISK_SPARSE Columns and Descriptions

Column	Description
GROUP_NUMBER	The number of the disk group containing the disk.
`DISK_NUMBER`	The number assigned to the disk within this disk group.
`INCARNATION`	The incarnation number for the disk.
`ALLOCATED_MAT_MB`	The total used physical and materialized capacity on the disk.
`TOTAL_MAT_MB`	The total physical capacity on the disk.
`SPARSE_READS`	The total number of I/O read requests on non-materialized regions of the disk.
`SPARSE_BYTES_READ`	The total number of bytes read from non-materialized regions of the disk.
`SPARSE_READ_TIME`	The time taken by sparse read I/O operations.

Table C-4 V$ASM_DISKGROUP_SPARSE Columns and Descriptions

Column	Description
GROUP_NUMBER	The cluster-wide number assigned to the disk group.
ALLOCATED_MAT_MB	The total used physical and materialized capacity of the disk group.
TOTAL_MAT_MB	The total physical capacity of the disk group.

C.2 Automatic Workload Repository Views

The Automatic Workload Repository (AWR) views contain information that is specific to Oracle Exadata.

- DBA_HIST_ASM_BAD_DISK
- DBA_HIST_ASM_DISKGROUP
- DBA_HIST_ASM_DISKGROUP_STAT
- DBA_HIST_ASM_DISK_STAT_SUMMARY
- DBA_HIST_CELL_CONFIG
- DBA_HIST_CELL_CONFIG_DETAIL
- DBA_HIST_CELL_DB
- DBA_HIST_CELL_DISKTYPE
- DBA_HIST_CELL_DISK_NAME
- DBA_HIST_CELL_DISK_SUMMARY
- DBA_HIST_CELL_GLOBAL
- DBA_HIST_CELL_GLOBAL_SUMMARY
- DBA_HIST_CELL_IOREASON
- DBA_HIST_CELL_IOREASON_NAME
- DBA_HIST_CELL_METRIC_DESC
- DBA_HIST_CELL_NAME
- DBA_HIST_CELL_OPEN_ALERTS

C.2.1 DBA_HIST_ASM_BAD_DISK

The DBA_HIST_ASM_BAD_DISK view displays historic information about non-online Oracle Automatic Storage Management (Oracle ASM) disks. This view contains snapshots of V$ASM_DISK.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
GROUP_NUMBER	NUMBER	NOT NULL	Number of the disk group containing the disk.
NAME	VARCHAR2(128)	NOT NULL	Name of the disk.

Column	Datatype	NULL	Description
PATH	VARCHAR2(256)		Operating system path name portion of the name returned by discovery.
STATUS	VARCHAR2(8)	NOT NULL	Global status for the disk. Only non-online disks are stored in this view.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.2 DBA_HIST_ASM_DISKGROUP

The DBA_HIST_ASM_DISKGROUP view contains information about Oracle ASM disk groups. This retrieves a subset of columns from V$ASM_DISKGROUP.

Column	Datatype	NULL	Description
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CURRENT_SNAP_ID	NUMBER	NOT NULL	AWR snapshot identifier for when the disk group was captured.
GROUP_NUMBER	NUMBER	NOT NULL	Cluster-wide number assigned to the disk group.
NAME	VARCHAR2(128)	NOT NULL	Name of the disk group.
TYPE	VARCHAR2(6)		Redundancy type for the disk group.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.3 DBA_HIST_ASM_DISKGROUP_STAT

The DBA_HIST_ASM_DISKGROUP_STAT view displays historic information about Oracle ASM disk groups.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
GROUP_NUMBER	NUMBER	NOT NULL	Cluster-wide number assigned to the disk group.
TOTAL_MB	NUMBER		Total capacity of the disk group in MB.

Column	Datatype	NULL	Description
FREE_MB	NUMBER		Unused capacity in MB of the disk group.
NUM_DISK	NUMBER		Number of disks in the disk group.
NUM_FAILGROUP	NUMBER		Number of failure groups in the disk group.
STATE	VARCHAR2(32)		State of the disk group. See also V$ASM_DISKGROUP.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.4 DBA_HIST_ASM_DISK_STAT_SUMMARY

The `DBA_HIST_ASM_DISK_STAT_SUMMARY` view displays historic summary information about Oracle ASM disk groups. It aggregates information from `V$ASM_DISK_STAT`.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
INSTANCE_NUMBER	NUMBER	NOT NULL	Database instance number.
GROUP_NUMBER	NUMBER	NOT NULL	Cluster-wide number assigned to the disk group.
READS	NUMBER		Total number of I/O read requests for all disks in the disk group.
WRITES	NUMBER		Total number of I/O write requests for all disks in the disk group.
READ_ERRS	NUMBER		Total number of failed I/O read requests for all disks in the disk group.
WRITE_ERRS	NUMBER		Total number of failed I/O write requests for all disks in the disk group.
READ_TIMEOUT	NUMBER		Total number of failed I/O read requests that are timed out for all disks in the disk group.
WRITE_TIMEOUT	NUMBER		Total number of failed I/O write requests that are timed out for all disks in the disk group.
READ_TIME	NUMBER		Total I/O time (in seconds) for read requests for all disks in the disk group.
WRITE_TIME	NUMBER		Total I/O time (in seconds) for write requests for all disks in the disk group.
BYTES_READ	NUMBER		Total number of bytes read for all disks in the disk group.

Column	Datatype	NULL	Description
BYTES_WRITTEN	NUMBER		Total number of bytes written for all disks in the disk group.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.5 DBA_HIST_CELL_CONFIG

The DBA_HIST_CELL_CONFIG view contains information about the configuration of the cells. This data is from V$CELL_CONFIG_INFO.

Column	Datatype	NULL	Description
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CURRENT_SNAP_ID	NUMBER	NOT NULL	AWR snapshot identifier for when the configuration was captured.
CELLNAME	VARCHAR2(256)	NOT NULL	Unique identifier for the cell.
CELLHASH	NUMBER	NOT NULL	Hash number to uniquely identify the cell.
CONFTYPE	VARCHAR2(15)	NOT NULL	Configuration type. The configuration type determines the type of information stored in the CONFVAL column. • AWRXML: configuration information pertinent to AWR reports. • CELL: configuration information about the cell. • CELLDISK: configuration information about cell disks. • GRIDDISK: configuration information about grid disks. • IORM_OBJ: IORM objective. • OFFLOAD: configuration information for offload servers. • PHYSICALDISK: configuration information about physical disks.
CONFVAL	CLOB		XML data associated with CONFTYPE.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.6 DBA_HIST_CELL_CONFIG_DETAIL

The `DBA_HIST_CELL_CONFIG_DETAIL` view displays historic information about the configuration of the cells. This view has the same descriptions as `DBA_HIST_CELL_CONFIG`, but includes the data for each snapshot, such as a historic view.

Column	Datatype	NULL	Description
`SNAP_ID`	NUMBER	NOT NULL	Unique snapshot identifier.
`DBID`	NUMBER	NOT NULL	Database identifier for the snapshot.
`CELLNAME`	VARCHAR2(256)	NOT NULL	Unique identifier for the cell.
`CELLHASH`	NUMBER	NOT NULL	Hash number to uniquely identify the cell.
`CONFTYPE`	VARCHAR2(15)	NOT NULL	Configuration type. The configuration type determines the type of information stored in the CONFVAL column. • AWRXML: configuration information pertinent to AWR reports. • CELL: configuration information about the cell. • CELLDISK: configuration information about cell disks. • GRIDDISK: configuration information about grid disks. • IORM_OBJ: IORM objective. • OFFLOAD: configuration information for offload servers. • PHYSICALDISK: configuration information about physical disks.
`CONFVAL`	CLOB		XML data associated with the CONFTYPE.
`CON_DBID`	NUMBER		The database identifier.
`CON_ID`	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.7 DBA_HIST_CELL_DB

The `DBA_HIST_CELL_DB` view displays historic information about the databases consuming resources on the cells. Only the top 10 databases are stored in each AWR snapshot. This is a snapshot of `V$CELL_DB`.

Column	Datatype	NULL	Description
`SNAP_ID`	NUMBER	NOT NULL	Unique snapshot identifier.
`DBID`	NUMBER	NOT NULL	Database identifier for the snapshot.
`CELL_HASH`	NUMBER	NOT NULL	Hash number to uniquely identify the cell.

Column	Datatype	NULL	Description
INCARNATION_NUM	NUMBER	NOT NULL	Incarnation number of the cell. Each cell reboot will increment the incarnation number.
SRC_DBID	NUMBER	NOT NULL	Database identifier performing the I/Os.
SRC_DBNAME	VARCHAR2(256)		DB Name performing the I/Os.
DISK_REQUESTS	NUMBER		Number of disk I/O requests performed by the database.
DISK_BYTES	NUMBER		Number of disk I/O bytes processed by the database.
FLASH_REQUESTS	NUMBER		Number of flash I/O requests performed by the database.
FLASH_BYTES	NUMBER		Number of flash I/O bytes processed by the database.
DISK_SMALL_IO_REQS	NUMBER		Number of small IO requests issued to disks by the database. Parallel cell metric: DB_IO_RQ_SM
DISK_LARGE_IO_REQS	NUMBER		Number of large IO requests issued to disks by the database. Parallel cell metric: DB_IO_RQ_LG
FLASH_SMALL_IO_REQS	NUMBER		Number of small IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_RQ_SM
FLASH_LARGE_IO_REQS	NUMBER		Number of large IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_RQ_LG
DISK_SMALL_IO_SERVICE_TIME	NUMBER		The latency (in nanoseconds) of small IO requests issued to disks by the database. Parallel cell metric: DB_IO_TM_SM
DISK_SMALL_IO_QUEUE_TIME	NUMBER		The IORM wait time (in nanoseconds) for small IO requests issued to disks by the database. Parallel cell metric: DB_IO_WT_SM
DISK_LARGE_IO_SERVICE_TIME	NUMBER		The latency (in nanoseconds) of large IO requests issued to disks by the database. Parallel cell metric: DB_IO_TM_LG
DISK_LARGE_IO_QUEUE_TIME	NUMBER		The IORM wait time (in nanoseconds) for large IO requests issued to disks by the database. Parallel cell metric: DB_IO_WT_LG
FLASH_SMALL_IO_SERVICE_TIME	NUMBER		The latency (in nanoseconds) of small IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_TM_SM
FLASH_SMALL_IO_QUEUE_TIME	NUMBER		The IORM wait time (in nanoseconds) for small IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_WT_SM

Column	Datatype	NULL	Description
FLASH_LARGE_IO_SERVICE_TIME	NUMBER		The latency (in nanoseconds) of large IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_TM_LG
FLASH_LARGE_IO_QUEUE_TIME	NUMBER		The IORM wait time (in nanoseconds) for large IO requests issued to flash by the database. Parallel cell metric: DB_FD_IO_WT_LG
IS_CURRENT_SRC_DB	NUMBER		Value is 1 for the database that captured the AWR data. Otherwise, value is NULL.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.8 DBA_HIST_CELL_DISKTYPE

The DBA_HIST_CELL_DISKTYPE view displays historic information about cells, including the types of disks and the capacity of the disks. This view is derived from DBA_HIST_CELL_CONFIG and DBA_HIST_CELL_CONFIG_DETAIL.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number to uniquely identify the cell.
CELL_NAME	VARCHAR2(4000)		User-readable cell name.
HARD_DISK_TYPE	VARCHAR2(4000)		The type of hard disk. The format of which is H/*size*, where *size* is suffixed by G, or T.
FLASH_DISK_TYPE	VARCHAR2(4000)		The type of flash, the format of which is F/*size*, where *size* is suffixed by G, or T.
NUM_CELL_DISKS	NUMBER		Number of cell disks.
NUM_GRID_DISKS	NUMBER		Number of grid disks.
NUM_HARD_DISKS	NUMBER		Number of hard disks.
NUM_FLASH_DISKS	NUMBER		Number of flash disks.
MAX_DISK_IOPS	NUMBER		The maximum number of IOPs for the hard disk type.
MAX_FLASH_IOPS	NUMBER		The maximum number of IOPs for the flash disk type.

Column	Datatype	NULL	Description
MAX_DISK_MBPS	NUMBER		The maximum I/O throughput for the hard disk type, in megabytes per second.
MAX_FLASH_MBPS	NUMBER		The maximum I/O throughput for the flash disk type, in megabytes per second.
MAX_CELL_DISK_IOPS	NUMBER		The maximum number of hard disk IOPs for the cell. This is calculated using MAX_DISK_IOPS*NUM_HARD_DISKS.
MAX_CELL_FLASH_IOPS	NUMBER		The maximum number of flash IOPs for the cell. This is calculated using MAX_FLASH_IOPS*NUM_FLASH_DISKS.
MAX_CELL_DISK_MBPS	NUMBER		The maximum I/O throughput of hard disk for the cell., in megabytes per second. This is calculated using MAX_DISK_MBPS*NUM_HARD_DISKS.
MAX_CELL_FLASH_MBPS	NUMBER		The maximum I/O throughput flash for the cell, in megabytes per second. This is calculated using MAX_FLASH_MBPS*NUM_FLASH_DISKS.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.9 DBA_HIST_CELL_DISK_NAME

The `DBA_HIST_CELL_DISK_NAME` view displays historic information about disk names on cells. This view is derived from `DBA_HIST_CELL_CONFIG` and `DBA_HIST_CELL_CONFIG_DETAIL`.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number to uniquely identify the cell.
DISK_ID	NUMBER	NOT NULL	Unique identifier of the disk.
DISK_NAME	VARCHAR2(4000)		User-readable cell name.
DISK	VARCHAR2(4000)		Disk type, either FlashDisk or HardDisk.
CON_DBID	NUMBER		The database identifier.

Column	Datatype	NULL	Description
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.10 DBA_HIST_CELL_DISK_SUMMARY

The DBA_HIST_CELL_DISK_SUMMARY view displays historic information about the performance of disks on cells. For each AWR snapshot, a summary of the per-minute metrics is visible in this view. Each metric stores the sum and the sum of squares, similar to DBA_HIST_SYSMETRIC_SUMMARY. This view is a summary of V$CELL_DISK_HISTORY.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number to uniquely identify the cell.
DISK_ID	NUMBER	NOT NULL	Unique identifier of the disk.
NUM_SAMPLES	NUMBER		Number of one-minute samples the summary is based on.
DISK_UTILIZATION_SUM	NUMBER		Sum of the per-minute disk utilization metrics. This is based on OS statistics.
READS_SUM	NUMBER		Sum of the per-minute read requests metrics per second. This is based on OS statistics.
READ_MB_SUM	NUMBER		Sum of the per-minute read metrics, in megabytes per second. This is based on OS statistics.
WRITES_SUM	NUMBER		Sum of the per-minute write requests metrics, per second. This is based on OS statistics.
WRITE_MB_SUM	NUMBER		Sum of the per-minute write metrics, in megabytes per second. This is based on OS statistics.
IO_REQUESTS_SUM	NUMBER		Sum of the per-minute IOPs. This is based on OS statistics.
IO_MB_SUM	NUMBER		Sum of the per-minute I/O metrics, in megabytes per second. This is based on OS statistics.
SERVICE_TIME_SUM	NUMBER		Sum of the per-minute service time metrics. This is based on OS statistics.
WAIT_TIME_SUM	NUMBER		Sum of the per-minute wait time metrics. This is based on OS statistics.

Column	Datatype	NULL	Description
SMALL_READS_SUM	NUMBER		Sum of the per-minute small read requests metrics per second. This is from cell server statistics.
SMALL_WRITES_SUM	NUMBER		Sum of the per-minute small write requests metrics per second. This is from cell server statistics.
LARGE_READS_SUM	NUMBER		Sum of the per-minute large read requests metrics per second. This is from cell server statistics.
LARGE_WRITES_SUM	NUMBER		Sum of the per-minute large write requests metrics per second. This is from cell server statistics.
SMALL_READ_BYTES_SUM	NUMBER		Sum of the per-minute small read bytes metrics per second. This is from cell server statistics.
SMALL_WRITE_BYTES_SUM	NUMBER		Sum of the per-minute small write bytes metrics per second. This is from cell server statistics.
LARGE_READ_BYTES_SUM	NUMBER		Sum of the per-minute large read bytes metrics per second. This is from cell server statistics.
LARGE_WRITE_BYTES_SUM	NUMBER		Sum of the per-minute large write bytes metrics per second. This is from cell server statistics.
SMALL_READ_LATENCY_SUM	NUMBER		Sum of the per-minute small read latency metrics. This data is from cell server statistics.
SMALL_WRITE_LATENCY_SUM	NUMBER		Sum of the per-minute small write latency metrics. This data is from cell server statistics.
LARGE_READ_LATENCY_SUM	NUMBER		Sum of the per-minute large read latency metrics. This data is from cell server statistics.
LARGE_WRITE_LATENCY_SUM	NUMBER		Sum of the per-minute large write latency metrics. This data is from cell server statistics.
APP_IO_REQUESTS_SUM	NUMBER		Sum of the per-minute IOPs metrics. This data is from cell server statistics.
APP_IO_BYTES_SUM	NUMBER		Sum of the per-minute I/O metrics, in megabytes per second. This data is from cell server statistics.
APP_IO_LATENCY_SUM	NUMBER		Sum of the per-minute I/O latency metrics. This data is from cell server statistics.
*_SUMX2	NUMBER		All the previous columns from DISK_UTILIZATION to APP_IO_LATENCY are repeated with the SUMX2 suffix. This is the sum of squares of the per-minute metrics.

Column	Datatype	NULL	Description
*_AVG	NUMBER		All the previous columns from DISK_UTILIZATION to APP_IO_LATENCY are repeated with the AVG suffix. This is the average of the per-minute metrics.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.11 DBA_HIST_CELL_GLOBAL

The `DBA_HIST_CELL_GLOBAL` view displays historic information about cell performance statistics. This is a snapshot of `V$CELL_GLOBAL`.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number that uniquely identifies the cell.
INCARNATION_NUM	NUMBER	NOT NULL	Incarnation number of the cell.
METRIC_ID	NUMBER	NOT NULL	Metric_id that identifies the metric.
METRIC_NAME	VARCHAR2(257)		Name of the metric.
METRIC_VALUE	NUMBER		The value of the metric.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.12 DBA_HIST_CELL_GLOBAL_SUMMARY

The `DBA_HIST_CELL_GLOBAL_SUMMARY` view displays historic information about the performance of cells. For each AWR snapshot, a summary of the per-minute metrics is visible in this view. This is a summary of `V$CELL_GLOBAL_HISTORY`.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot ID

Column	Datatype	NULL	Description
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number that uniquely identifies the cell.
INCARNATION_NUM	NUMBER	NOT NULL	Incarnation number of the cell.
NUM_SAMPLES	NUMBER		Number of 1 minute samples the summary is based on.
CPU_USAGE_SUM	NUMBER		Sum of the per-minute CPU usage percentage metrics.
SYS_USAGE_SUM	NUMBER		Sum of the per-minute system CPU usage percentage metrics.
USER_USAGE_SUM	NUMBER		Sum of the per-minute user CPU usage percentage metrics.
NETWORK_BYTES_RECD_SUM	NUMBER		Sum of the per-minute network received metrics, in bytes per second.
NETWORK_BYTES_SENT_SUM	NUMBER		Sum of the per-minute network sent metrics, in bytes per second.
*_SUMX2	NUMBER		All previous columns from CPU_USAGE to NETWORK_BYTES_SENT are repeated with the SUMX2 suffix. This is the sum of squares of the per-minute metrics.
*_AVG	NUMBER		All previous columns from CPU_USAGE to NETWORK_BYTES_SENT are repeated with the SUMX2 suffix. This is the average of the per-minute metrics.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.13 DBA_HIST_CELL_IOREASON

The DBA_HIST_CELL_IOREASON view displays historic information about the reasons for performing I/Os on the cells. This is a snapshot of V$CELL_IOREASON.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number that uniquely identifies the cell.

Column	Datatype	NULL	Description
INCARNATION_NUM	NUMBER	NOT NULL	Incarnation number of the cell.
REASON_ID	NUMBER	NOT NULL	Unique identifier for the I/O reason.
REASON_NAME	VARCHAR2(257)		User readable I/O reason.
REQUESTS	NUMBER		Number of I/O requests performed for the reason.
BYTES	NUMBER		Number of IO Bytes performed for the reason.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.
READS	NUMBER		Number of read requests for the reason. This value includes the flash reads counted in the READS_FD column.
WRITES	NUMBER		Number of write requests for the reason. This value includes the flash writes counted in the WRITES_FD column.
READS_FD	NUMBER		Number of flash read requests for the reason. This value is a subset of the value in the READS column.
WRITES_FD	NUMBER		Number of flash write requests for the reason. This value is a subset of the value in the WRITES column.

C.2.14 DBA_HIST_CELL_IOREASON_NAME

The DBA_HIST_CELL_IOREASON_NAME view displays the reasons for different I/Os. This data is from V$CELL_IOREASON_NAME.

Column	Datatype	NULL	Description
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
REASON_ID	NUMBER	NOT NULL	Unique identifier for an I/O reason.
REASON_NAME	VARCHAR2(257)		User-readable description of the I/O reason.
CON_DBID	NUMBER		The database identifier.

Column	Datatype	NULL	Description
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.15 DBA_HIST_CELL_METRIC_DESC

The DBA_HIST_CELL_METRIC_DESC view displays information about cell metrics. This data is from V$CELL_METRIC_DESC.

Column	Datatype	NULL	Description
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
METRIC_ID	NUMBER	NOT NULL	Unique identifier for the metric
METRIC_NAME	VARCHAR2(257)		User-readable metric name
METRIC_TYPE	VARCHAR2(17)	NOT NULL	Unit for the metrics, if NULL this is a count.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.16 DBA_HIST_CELL_NAME

The DBA_HIST_CELL_NAME view displays historic information about the names of cells. This view is derived from DBA_HIST_CELL_CONFIG and DBA_HIST_CELL_CONFIG_DETAIL.

Column	Datatype	NULL	Description
SNAP_ID	NUMBER	NOT NULL	Unique snapshot identifier.
DBID	NUMBER	NOT NULL	Database identifier for the snapshot.
CELL_HASH	NUMBER	NOT NULL	Hash number that uniquely identifies the cell.
CELL_NAME	VARCHAR2(4000)		User-readable cell name.
CON_DBID	NUMBER		The database identifier.
CON_ID	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

C.2.17 DBA_HIST_CELL_OPEN_ALERTS

The `DBA_HIST_CELL_OPEN_ALERTS` view displays the open alerts on cells. Only the 10 most-recent alerts for each cell are stored in AWR. This data is from `V$CELL_OPEN_ALERTS`.

Column	Datatype	NULL	Description
`SNAP_ID`	NUMBER	NOT NULL	Unique snapshot identifier.
`DBID`	NUMBER	NOT NULL	Database identifier for the snapshot.
`CELL_HASH`	NUMBER	NOT NULL	Hash number that uniquely identifies the cell.
`BEGIN_TIME`	DATE	NOT NULL	Time that the alert was opened.
`SEQ_NO`	NUMBER	NOT NULL	Sequence number of the alert.
`MESSAGE`	VARCHAR2(1024)		Message describing the alert.
`STATEFUL`	VARCHAR2(1)		Whether or not the alert has state.
`SEVERITY`	VARCHAR2(64)		The severity of the alert. Possible values are as follows: • critical • warning • info
`CON_DBID`	NUMBER		The database identifier.
`CON_ID`	NUMBER		The identifier of the container identified by CON_DBID. Possible values are as follows: • 0: This value is used for non-container database (CDB). • 1: This value is used for the root container of the CDB.

D

Oracle Exadata System Software Accessibility Recommendations

Use these tips for using screen readers and screen magnifiers with Oracle Exadata System Software utilities.

Oracle Exadata System Software includes tools such as `dcli` and `cellcli` that you can run from the command line.

- Tips on Using Screen Readers and Braille Displays
  Use the following tips when using screen readers with Oracle Exadata System Software
- Tips on Using Screen Magnifiers
  Use the following tips when using screen magnifiers with Oracle Exadata System Software
- Tips on Using Exawatcher Charts
  Exawatcher is an Exadata specific tool that collects performance data from Exadata storage cells.
- Oracle Exadata Deployment Assistant (OEDA) Web interface Accessibility
  The OEDA Web interface conforms with the Web Content Accessibility Guidelines version 2.0 at the AA level (WCAG 2.0 AA).

D.1 Tips on Using Screen Readers and Braille Displays

Use the following tips when using screen readers with Oracle Exadata System Software

Examples of screen readers include JAWS, SuperNova, and NVDA. Each of these provides text-to-speech output and supports braille displays.

- Use a character mode based terminal such as Putty or Cygwin. Do not use an X-Windows-based VNC.
- For screen reader users, we recommend installing "screen" in order to get multiple session support. The Linux based screen program allows for multiple sessions in different windows. You can access each session with keyboard based commands, for example, `Ctrl-a`. Screen allows you to detach or re-attach to a given window session. Like VNC, if you get disconnected when running Oracle ExaCHK, or patchmgr, or other program, you can re-attach to and resume that session.

  The screen package is not installed by default on Exadata. You will need to install it using `yum`. See the "How To Use Linux Screen" tutorial at `https://www.rackaid.com/blog/linux-screen-tutorial-and-how-to/` for details.
- In the settings of the terminal software, set the cursor type to "block" cursor, not blinking or flashing.
- The output of the commands can generate a significant amount of information and might spill off the terminal window, and the virtual window or braille display. For example, the following command can generate a long alert history output:

  ```
  dcli -g cell_group -l root cellcli list alerthistory
  ```

To display the output one screen-full at a time, pipe the output through the `more` command, as in the following:

```
dcli -g cell_group -l root cellcli list alerthistory | more
```

You can then use the space bar key to page through the output.

- When `exachk` or `dbnodeupdate.sh` is launched interactively, do not pipe its output to the `more` or `page` commands. As it runs, it displays informational messages on the terminal. The messages pause when `exachk` requires user input, then resume after input is received. Important messages, user input, errors, and check results are logged in various files. The results from `exachk` are written to an HTML report. All you need to do is to transfer the HTML report to a computer that runs your assistive technology and open the HTML report in a browser that you can access with your assistive technology.
- If you are running the `patchmgr` utility, and it is performing a task that takes some time to complete, the output displays a "spinner" and a countdown clock. The "spinner" cycles through the `\`, `|`, and `/` characters in-place, and the countdown clock is updated periodically. When the task is done, the output displays a "success" or "error" message, depending on the outcome. The output messages are also logged in a log file.
- A few recommended screen reader settings include the following (JAWS is used here just as an example):
  - Set the JAWS cursor to "All". Use the key combination of `Insert + s` until you hear "All".
  - You may need to turn off virtual cursor. If you are using JAWS, you can do this using the key combination of `Insert + z`.
  - Use the virtual window to capture text. If you are using JAWS, you can do this using the key combination of `Insert + Alt + w`.

D.2 Tips on Using Screen Magnifiers

Use the following tips when using screen magnifiers with Oracle Exadata System Software

Examples of screen magnifiers include ZoomText, MAGic, and SuperNova.

- Screen magnifiers can support both character-based terminals and X-Window-based VNC.
- If you are using the screen reader function of the screen magnifier (ZoomText screen reader), then you should use a character-based terminal as described above.
- If you are using a VNC, decide your preference for a window display, for example, TWM or ICE. A display setting for ICE can be done with the following:

  ```
  vncserver -geometry 1600x950 :2
  ```

  `1600x950` specifies the display size, and `:2` specifies the VNC display number.

D.3 Tips on Using Exawatcher Charts

Exawatcher is an Exadata specific tool that collects performance data from Exadata storage cells.

The data collected by Exawatcher can be graphed and presented in web pages. The web pages are based on Oracle JavaScript Extension Toolkit (JET), HTML and some use of ARIA. The Exawatcher pages can present charts which can be navigated using only a keyboard.

D.4 Oracle Exadata Deployment Assistant (OEDA) Web interface Accessibility

The OEDA Web interface conforms with the Web Content Accessibility Guidelines version 2.0 at the AA level (WCAG 2.0 AA).

The OEDA Web interface was built using Oracle JET, which provides support for:

- Keyboard and touch navigation

  Oracle JET components follow the Web Accessibility Initiative - Accessible Rich Internet Application (WAI-ARIA) guidelines. You can find the Oracle JET hotkey information at Oracle JavaScript Extension Toolkit (JET) Keyboard and Touch Reference.

- Zoom

  Oracle JET supports browser zooming up to 200%. For example, on the Firefox browser, you can choose **View**, then **Zoom**, and then **Zoom In**.

- Screen reader

  Oracle JET supports screen readers such as JAWS, Apple VoiceOver, and Google Talkbalk by generating content that complies with WAI-ARIA standards, and no special mode is needed.

- Oracle JET component roles and names

  Each component has an appropriate role, such as `button`, `link`, and so on, and each component supports an associated name (label), if applicable.

- Sufficient color contrast

  Oracle JET provides the Alta theme which is designed to provide a luminosity contrast ratio of at least 4.5:1.

OEDA Web Interface Overview

The OEDA Web interface contains two regions. The navigation region occupies the leftmost portion of the page, and the main region occupies the rest. The navigation region contains a list of sections while the main region contains various fields and other elements that are used to specify various configuration details and control the user interface.

The major sections are:

- Select Hardware
- Choose Operating System
- RAC Networks
- Users and Groups
- Define Clusters
- Diskgroups
- Create Database Homes
- Create Database

- Cluster Networks
- Alerting
- Comments

Initially, the Select Hardware section is the only active section. As each section is completed, the next section becomes active in the navigation region and corresponding details are added to the main region. Sections are generally completed in order, but the interface allows users to navigate back to previous sections and make changes. Elements relating to each section are grouped in collapsible areas and each area contains buttons to expand and collapse the area.

OEDA-Specific Keyboard Shortcuts

The following table lists keyboard shortcuts for controlling and navigating around the OEDA Web interface.

Description	Windows	Mac OS
Collapse all sections	`Alt+Ctrl+C`	`Option+Ctrl+C`
Expand all sections	`Alt+Ctrl+E`	`Option+Ctrl+E`
Move focus to the Options menu	`Alt+Ctrl+M`	`Option+Ctrl+M`
Move focus to the main region	`Alt+Ctrl+N`	`Option+Ctrl+N`

Keyboard and Screen Reader Navigation using OEDA

When you start the OEDA Web interface, the focus is initially placed on the Options menu button. Activating the Options menu button opens the menu and displays the list of menu items. When the focus is on the Options menu button, pressing the Tab key moves the focus to the navigation region. When the focus is on the navigation region, pressing the Tab key moves the focus to the main region.

To access the Options menu from anywhere, use the keyboard shortcut (`Alt+Ctrl+M` on Windows or `Option+Ctrl+M` on Mac OS). To return the focus back to the main region, use `Alt+Ctrl+N` on Windows or `Option+Ctrl+N` on Mac OS.

OEDA uses a tabbed interface to represent multiple instances in a section. For example, the Select Hardware section contains the "+" button, which creates a new tab that contains the details for another rack. When a new tab is created, the focus moves to the new tab. You can navigate across multiple tabs by using the arrow keys and you can activate the tab by using the space bar or Enter key. When the focus is on a tab, the JAWS screen reader identifies the tab label and whether it is removable.

Selecting an active section in the navigation region does not move focus to the associated area in the main region. This is a known issue, which is tracked using bug 35029653. To work around the issue using only the keyboard, do the following:

1. Use the keyboard shortcut to collapse all sections in the main region (`Alt+Ctrl+C` on Windows or `Option+Ctrl+C` on Mac OS).
2. Use the Tab key to navigate to the main region.
3. In the main region, use the Tab key (or Shift+Tab) to move to the desired section in the list of collapsed sections.
4. Activate and expand the desired section by using the space bar or Enter key.

To work around the issue using the JAWS screen reader:

1. Use the JAWS keyboard shortcut to get a list of active buttons (`Ctrl+Insert+B`).

2. Use the JAWS button list dialog to navigate to the expand/collapse button associated with the desired section.

When using JAWS, note the following:

- The JAWS button list dialog contains all active buttons, which may be a lengthy list.
- Except for the page heading, the OEDA Web interface does not contain HTML headings (`<H2>`, `<H3>`, and so on). Therefore, you cannot navigate using HTML headings within JAWS.
- The OEDA Web interface does not contain any links. Therefore, you cannot navigate using links within JAWS.

Related Topics

- WAI-ARIA: Developing a Keyboard Interface