Deployment Best Practices for Oracle Database with Dell EMC PowerMax

February 2020
H17390.3

Abstract
The Dell EMC PowerMax storage system is designed and optimized for high-performance NVMe flash storage, while providing ease of use, reliability, availability, security, and versatility. This white paper explains and demonstrates the benefits and best practices for deploying Oracle databases on PowerMax storage systems.
Contents

Executive summary..........................................................................................................................5
  Audience ...................................................................................................................................5

PowerMax Key benefits for Oracle databases .............................................................................5
  Performance .................................................................................................................................5
  Data reduction ............................................................................................................................6
  Local replications .......................................................................................................................7
  Remote replications ....................................................................................................................7
  Data protection ............................................................................................................................8

PowerMax product overview .........................................................................................................8
  PowerMax architecture ................................................................................................................9
  PowerMax compression and deduplication .................................................................................10
  PowerMax and FC-NVMe ...........................................................................................................11

PowerMax and Oracle performance tests .....................................................................................13
  Test environment ......................................................................................................................13
  OLTP performance test cases ..................................................................................................14
  DSS performance test cases .....................................................................................................22
  Compression performance tests ...............................................................................................24

PowerMax data reduction at work .................................................................................................26
  Compression and deduplication of a non-encrypted Oracle database ....................................27
  Compression and deduplication of an encrypted Oracle database ..........................................28
  Conclusion ...............................................................................................................................29
  Data reduction management using CLI commands ...................................................................29

PowerMax Service Levels .............................................................................................................30
  Service Levels overview ..........................................................................................................30
  How Service Levels work ..........................................................................................................30

Best Practices for PowerMax and Oracle databases .....................................................................33
  Storage considerations ..............................................................................................................33
  Server considerations ................................................................................................................40
  Oracle ASM best practices ......................................................................................................48
  Oracle sequential read I/O size .................................................................................................50
  4 KB redo log sector size .........................................................................................................50
  Choosing the Linux kernel I/O scheduler ................................................................................50

Appendixes ....................................................................................................................................51
  Appendix I. Blk-mq and scsi-mq ..............................................................................................51
| Appendix II. Oracle ASM online storage reclamation | 53 |
| Appendix III. Overview of PowerMax compression and deduplication | 56 |
| Appendix IV. Linux `iostat` command | 58 |
| Appendix V. Oracle AWR I/O related information | 59 |
Executive summary

The PowerMax family was specifically created to fully unlock the performance benefits that NVMe offers to applications. NVMe is a set of standards that define a PCI express (PCIe) interface used to efficiently access Non-Volatile Memory (NVM) storage media. PowerMax NVM media includes both NAND-based flash storage and dual-ported Storage Class Memory (SCM) drive technology such as Intel Optane.

In the PowerMaxOS Q3 2019 release, PowerMax added support for SCM drives together with Machine Learning (ML) algorithms for data placement when both NAND and SCM drives are used. The update also introduced end-to-end FC-NVMe (NVMe over Fiber-Channel Fabrics) to optimize data access between server and storage, and 32 Gb front-end modules that allow faster connectivity speed per port.

At the same time, the PowerMax family continues to provide all the features required by enterprise applications such as six-nines (99.9999%) availability, encryption, replications, data reduction, and massive consolidation, now delivered with I/O latencies that are measured in microseconds.

This white paper explains and demonstrates the benefits and best practices of deploying Oracle databases on PowerMax storage systems.

Audience

This white paper is intended for database and system administrators, storage administrators, and system architects who are responsible for implementing, managing, and maintaining Oracle databases with PowerMax storage systems. It is assumed that readers have some familiarity with Oracle and the PowerMax family, and are interested in achieving higher database availability, performance, and ease of storage management.

PowerMax Key benefits for Oracle databases

The following is a brief summary of the key PowerMax features that benefit Oracle database deployments.

Performance

General performance benefits

- **PowerMax cache provides extreme read and write performance**—The PowerMax storage system supports up to 16 TB of raw DRAM-based cache. While a portion of PowerMax cache is used for system metadata, the majority is used to support extreme performance for applications’ read and write operations.

- **32 Gb front-end modules**—with the PowerMaxOS Q3 2019 release, PowerMax now supports front-end connectivity of 32 Gb per port allowing faster speeds per port for applications’ read and write operations. These modules support both FC and FC-NVMe connectivity.

- **End-to-end FC-NVMe**—with the PowerMaxOS Q3 2019 release, PowerMax now expands its NVMe backend connectivity to support end-to-end NVMe over Fabrics between server and storage, allowing additional protocol optimizations as compared to FC.
PowerMax Key benefits for Oracle databases

- **Host I/O limits and Service Levels**—Some customers prefer to leverage the ability to enforce performance limits, such as for non-production workloads or multitenant designs (for example, for chargeback or for Service Providers). The PowerMax Host I/O limits feature caps the IOPS or bandwidth of specific storage groups (SGs). Similarly, Service Levels (SL) set performance goals on SGs.

  For more information, see PowerMax Service Levels.

**Optimized writes**

- **Write persistent cache**—PowerMax cache is mirrored for writes and vaults if there is a power failure. It is therefore considered persistent, and as a result, all application writes are acknowledged to the server as soon as they are registered in the cache\(^1\), providing extremely low write latencies.

- **Write-folding**—Database writes tend to update the same block (or adjacent blocks) multiple times in a short period of time. The PowerMax write-folding feature allows multiple updates in cache (DRAM), persisting periodically the latest update to NVMe flash media. Thus, the media is better preserved, and storage resource utilization is improved by avoiding unnecessary writes.

- **Write-coalescing**—When the PowerMax storage system writes the cached data to the NVMe flash media, it can often aggregate and optimize the writes to larger I/O sizes than the application writes, eliminating unnecessary I/O operations.

**Optimized reads**

- **FlashBoost**—Database read I/Os that are serviced from PowerMax cache are already extremely fast. However, if the data is not in cache (that is, a “read-miss”), the PowerMax storage system expedites the data transfer by sending it from the back end (NVMe flash media) to the front end (server) and only then placing it in cache for possible future reads.

**Data reduction** Data reduction features include:

- **Thin devices**—All PowerMax storage devices are created thin by default, which means that storage capacity is allocated only when the application writes to them.

- **Compression and deduplication**—The PowerMax Adaptive Compression Engine (ACE) provides inline storage compression and deduplication. Hardware modules support both compression and deduplication to provide high-performance, together with other software features such as Activity Based Compression (ABC) and Fine-Grain Data Packing.

  For more information, see PowerMax compression and deduplication.

- **ASM online storage reclamation**—Oracle ASM Filter Driver (AFD) enables you to declare ASM disk groups as capable of online storage reclamation. If large data sets are deleted within ASM (for example, if a legacy database is

---

\(^1\) Except for synchronous replications that require a write to also register with the remote PowerMax cache before the I/O is acknowledged to the originating host.
delete), the PowerMax storage system frees up the deleted capacity within the storage system, even while the ASM disk group remains online.

For more information, see Appendix II. Oracle ASM online storage reclamation.

**Local replications**

PowerMax SnapVX software allows you to create up to 256 local snapshots from each storage group, protecting the source data. These snapshots can be restored at any time and can be linked to up to 1,024 targets. A linked snapshot target allows direct access to the snapshot’s data. SnapVX instantly creates (or restores) copies of the database for purposes such as point-in-time protection, creation of test environments, and backup and recovery images.

SnapVX snapshots are:

- **Consistent**—All snapshots are natively “storage-consistent” (restartable database replicas). By following the best practices for Oracle backup and recovery, the snapshots can become “application-consistent” (recoverable database replicas), allowing database roll-forward recovery.

- **Protected**—All snapshots are protected. A snapshot can be restored over and over again (for example, during patch testing) until it is successful. Also, a snapshot can be linked to target devices that are then mounted by another server. Changes to the target devices don’t affect the original snapshot’s data.

- **Named**—All snapshots are given a user-friendly name when they are created. When the same name is used, a new generation of the snapshot is created for ease of management.

- **Automatically expired**—Optionally, snapshots can be given an automatic expiration date and time on which they terminate.

- **Secure**—Optionally, snapshots can be made secure. Secure snapshots cannot be deleted before their expiration date.

- **Ad-hoc or scheduled**—Snapshots can be taken at any time ad-hoc, or they can be scheduled using Unisphere.

For more information about SnapVX, see the Oracle Database Backup, Recovery, and Replications Best Practices with VMAX All Flash Storage White Paper.

**Remote replications**

PowerMax SRDF provides a variety of replication modes and topologies, which include synchronous and asynchronous modes, cascaded, Star, and Metro (active/active capabilities that work well with Oracle extended RAC) topologies.

For more information about SRDF, see the Oracle Database Backup, Recovery, and Replications Best Practices with VMAX All Flash Storage White Paper.

---

2 Starting with Oracle 12c, SnapVX snapshots can be used for database recovery without having to create them while Oracle is in hot-backup mode. This allows for much more aggressive backup and recovery strategy using SnapVX.
Data protection features include:

- **Internal T10-DIF**—T10-DIF (Data Integrity Field), or T10-PI (Protection Information), is a standard for data protection that changes the SCSI block from 512 bytes to 520 bytes and adds 8 bytes of protection information such as CRC and block address. Internally, all data within the PowerMax system is protected with T10-DIF as it moves between the front-end modules, cache, back-end, and flash storage. PowerMax T10-DIF protection includes local and remote replications, preventing data corruption.

- **External T10-DIF**—With supported configurations, the PowerMax storage system allows for T10-DIF protection to extend to the database server and back. The participating layers validate all read and write I/Os in real time and block corruptions. External T10-DIF is implemented by Oracle ASMlib, Red Hat Linux, and others. Consult the Dell EMC support matrix for a full list of supported configurations.

- **PowerProtect Storage Direct**—PowerProtect Storage Direct (previously called ProtectPoint) is an integration between the PowerMax storage system and the Data Domain backup storage appliance that allows backups of large databases to be performed in seconds. The Data Domain catalogs the backups and adds compression, deduplication, and optional remote replications. Restore time is also optimized in the integrated system.

- **D@RE**—Data at rest encryption (D@RE) provides transparent data encryption in the storage system. The PowerMax storage system uses specialized hardware modules to avoid performance penalties.

**PowerMax product overview**

The Dell EMC PowerMax family consists of two models, as shown in the following figure:

- **PowerMax 2000**—Designed to provide customers with efficiency and maximum flexibility in a 20U footprint

- **PowerMax 8000**—Designed for massive scale and performance, all within a two-floor-tile footprint
Both PowerMax storage systems have, at their foundation, the trusted Dynamic Virtual Matrix architecture and a new version of HYPERMAX OS management software, rewritten for the NVMe platform, called PowerMaxOS 5978. PowerMaxOS can run natively on both PowerMax storage systems and on legacy VMAX All Flash systems as an upgrade. PowerMax storage systems are specifically targeted to meet the storage capacity and performance requirements of the enterprise data center.

PowerMax configurations consist of modular building blocks called PowerMax bricks, as shown in the following figure. The modular brick architecture reduces complexity and allows for easier system configuration and deployment.

The initial PowerMax brick includes a single engine consisting of two directors, two system power supplies (SPS), and two 24-slot 2.5” NVMe Drive Array Enclosures (DAEs).
The PowerMax 2000 comes with an initial capacity of 11 or 13 TBu, depending on the RAID configuration. The PowerMax 8000 comes with an initial capacity of 53 TBu for open systems.

The brick concept allows PowerMax storage systems to scale up and scale out. Customers can scale up by adding Flash Capacity Packs. Each Flash Capacity Pack for the PowerMax 8000 storage system has 13 TBu of usable storage and the PowerMax 2000 storage system has 11 TBu or 13 TBu of usable storage (depending on the RAID protection type).

The PowerMax storage system scales out by aggregating up to two bricks for the PowerMax 2000 storage system, and up to eight bricks for the PowerMax 8000 storage system in a single system with fully shared connectivity, processing power, and linear scalability.

For more information about the PowerMax architecture and features, see the following:

- [Dell EMC PowerMax: Family Overview White Paper](#)
- [Dell EMC PowerMax Family Data Sheet](#)
- [Dell EMC PowerMax Family Specification Sheet](#)

### PowerMax Adaptive Compression Engine (ACE)

The PowerMax storage system uses a strategy that is targeted to provide optimal data reduction without compromising performance. The PowerMax Adaptive Compression Engine (ACE) is the combination of the following components:

- **Hardware acceleration**—Each PowerMax engine is configured with two hardware compression modules (one per director) that handle data compression and decompression. These hardware modules are also capable of generating Hash IDs that enable deduplication and are more powerful than the modules used with VMAX All Flash arrays.

- **Optimized data placement**—The application’s data is stored in different compression pools that provide a compression ratio (CR) from 1:1 (128 KB pool) up to 16:1 (8 KB pool) and is spread across the PowerMax back end for best performance. The pools are dynamically added or deleted based on need.

- **Activity Based Compression (ABC)**—Typically, the most recent data is the most active, creating an “access skew”. ABC relies on that skew to prevent constant compression and decompression of data extents that are frequently accessed. The ABC function marks the busiest 20 percent of all allocated data extents in the system and allows them to skip the compression workflow. Data extents that are highly active remain uncompressed, even if compression is enabled for their storage group. As the data extents become less active, they are automatically compressed while newly active extents become part of the “hottest” 20 percent (as long as enough free storage capacity is available).

- **Fine-Grain Data Packing**—When PowerMax compresses data, each 128 KB track is split into four 32 KB buffers. All buffers are compressed in parallel. The

---

3 TBu: usable capacity in Terabyte. This refers to the amount of physical capacity available in the storage system, taking into account the RAID efficiency of the RAID type in use.
total of the four buffers results in the final compressed size and determines in which compression pool the data is allocated. Included in this process is a zero reclaim function that prevents the allocation of buffers with all zeros and no actual data. For a small size write or read, only the necessary buffers participate, not all four buffers.

- **Extended Data Compression (EDC)**—Data that is already compressed automatically goes through additional, more powerful compression if it is untouched for over 30 days, increasing storage efficiency.

Additionally, note the following:

- Compression is enabled or disabled at a storage group level for ease of management. Generally, most databases can benefit from storage compression. Customers might decide not to enable compression if the database is fully encrypted or if a storage group contains data that is continuously overwritten (such as Oracle redo logs).

- When compression is enabled, all new writes benefit from inline compression. If the storage group already contains data when compression is enabled, it goes through background compression with low priority (as compared to application I/Os, which get higher priority).

**PowerMax deduplication**

In addition to providing more powerful hardware compression modules, the PowerMax storage system also allows data deduplication (dedupe). PowerMax deduplication is automatically enabled or disabled when compression is enabled or disabled (compression and deduplication cannot be managed separately).

PowerMax deduplication operates at 128 KB granularity. Because Oracle ASM Allocation Units (AU) have a granularity of 1 MB or larger, PowerMax deduplication works well with Oracle databases residing in ASM disk groups. Any new ASM extent is aligned at 1 MB (or higher) offsets, allowing the PowerMax storage system to easily determine if the data is unique, without misalignment concerns. As shown later in this white paper, the PowerMax storage system achieves 100 percent deduplication benefits for Oracle databases residing in ASM.

For more information about PowerMax data reduction, see this white paper:

- [Dell EMC PowerMax: Data Reduction Technical White Paper](#)

**PowerMax and FC-NVMe**

With the PowerMaxOS Q3 2019 release, PowerMax introduced end-to-end NVMe over FC fabrics, or FC-NVMe. Internally, PowerMaxOS was already accessing both NAND and SCM flash media using NVMe protocol. New to this release is that now also the database servers can use the NVMe protocol over FC fabric (SAN switches) to access the PowerMax storage system for all read and write I/O operations.

To use this protocol, 32 Gb PowerMax front-end modules are required, together with Gen 6 FC switch and HBAs (32 Gb). This infrastructure allows for either FC or FC-NVMe deployment. The only difference is the protocol that the servers use to access storage devices. When PowerMax front-end adapter ports are set for FC connectivity they are listed as FAs; when set for FC-NVMe they are listed as FNs.
Why use NVMe?
Starting with NAND SSD flash storage, but much more importantly with SCM, storage latency is no longer measured in milliseconds but in microseconds. In addition, each SCM supports hundreds of thousands of IOPS.

NVMe protocol was built from the ground-up to replace legacy SAS and SATA storage access protocols meant for spinning drives. NVMe is NUMA-optimized to take advantage of multi-core CPUs so cores can share control over I/O submission queues. It is built around fast PCIe access between flash media and CPU, and it allows for much higher queue depth, which provides higher concurrency. This is important, because such high-performance drives often service higher I/O density (concurrency of I/Os per GB storage).

Why use FC-NVMe?
While on-board server PCIe NVMe flash drives are not new, there are a few disadvantages as compared to SCM inside a storage system, connected to the server using PCIe FC-NVMe protocol. First, high-performance NVMe flash drives such as SCM cost more than NAND SSD. Placing such drives in the server requires high server utilization to justify the cost. However, servers perform at their peak capacity for only a small percentage of their daily, weekly, or monthly usage. The rest of the time their utilization is low. This creates a potential waste of money and resources.

Placing the SCM drives in the storage system allows many servers to access the high-performance media over NVMe fabrics. Therefore, regardless of which server peaks in activity, the media is shared and can always be highly utilized. This adds value to the investment in SCM and allows for better resource utilization. In addition, PowerMax Service Levels set priorities to storage groups’ consumption of SCMs.

A similar comparison can be made with HCI (Hyper-Converged Infrastructure). HCI is based on a group of servers that provide compute (CPU), memory, networking, and storage resources for the deployment of applications. When on-board server SCM flash drives are used, given the nature of HCI, where applications may end up on any node in the cluster, it is very likely that the SCM drives are now needed on all the servers equally.

If the workload is not balanced, only a few of the servers will be busy at any given time. Either the busy servers could benefit from access to more SCM drives than those available locally (there will be other servers with SCM drives possibly underutilized), or in an opposite example, the busy servers are not busy enough to fully utilize even a single SCM drive installed locally.

By using FC-NVMe, these high-performance resources can be shared by any server that needs them since they are located in the storage system. This saves the cost of placing SCM drives in all the HCI nodes, and increases the utilization of the drives, because all servers connected to the storage system can leverage them, based on their designated Service Levels.

Because FC-NVMe is relatively new, Linux OS and multipathing software support is still limited. Refer to the section FC-NVMe multipathing options for more information.
PowerMax and Oracle performance tests

This section describes the performance tests and results we achieved running Oracle workloads in our lab. The test cases are meant to demonstrate different situations so we can learn how they affect database performance.

It is important to note that the PowerMax storage system we used for the tests was a single-engine (brick) system. Therefore, the performance numbers should not be viewed as a platform ‘hero’ numbers. Instead, they provide an example of the performance levels an Oracle database workload can achieve with a relatively small configuration (a single PowerMax 8000 brick and four 28-core PowerEdge R740 servers). These tests also allowed us to leverage and demonstrate the best practices described in this paper.

Test environment

Hardware and software configuration

Table 1 describes the hardware and software components that were used for the performance tests.

In general, due to the new support offered by the operating system vendors of FC-NVMe, we’ve tested Oracle database performance (both FC, and FC-NVMe) with SLES 15, RHEL 8.0, and OL 7.7/UEK5 update 2. Overall the performance achieved was very similar among the different operating systems. As it doesn’t make sense to report the same results over and over again, the OLTP test results in this paper are based on OL 7.7/UEK5u2, and the DSS test results in this paper are based on RHEL 8.0. As some of the tests focused on high IOPS, we used FC emulation instead of FC-NVMe (refer to the section: FC or FC-NVMe protocol choice and core allocations to see why it matters for benchmark testing of a single brick).

The PowerMax 8000 storage system used for testing had a single brick (1 engine) and 1 TB raw cache, which is the smallest configuration for this system.

Oracle 19c Grid Infrastructure and databases were configured as a four-node cluster (RAC).

SLOB 2.4 benchmark was used to generate Oracle OLTP workloads. The SLOB configuration consisted of 96 users (also database schemas, or tables), with a scale of 30 GB for a total dataset size of 2.8 TB (96 x 30 GB). Together with Oracle system and undo tablespaces, the +DATA ASM disk group was close to 3.5 TB capacity used. The performance tests were run with “lite” redo generation and 30 percent update (slob.conf parameters).

For the Oracle DSS tests (large-I/O sequential reads), the dbgen utility from TPC-H tools was used to create a 1 TB Lineitem table.

Table 1. Hardware and software components

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Quantity/Size</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage system</td>
<td>PowerMax 8000 storage system</td>
<td>• 1 x brick, 1 TB raw cache</td>
<td>PowerMaxOS 5978.444.444 based on Q3 2019 release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 30 x NVMe NAND SSD</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 8 x NVMe SCM</td>
<td></td>
</tr>
</tbody>
</table>
## PowerMax and Oracle performance tests

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Quantity/Size</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database servers</td>
<td>4 x Dell R740</td>
<td>• Each Dell server: 2 x Intel Xeon E5-2690v4 2.6GHz (total 28 cores), 128 GB RAM</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>OL 7.7 with UEK5u2 (OLTP tests), RHEL 8.0 (DSS tests)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(OS)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Host bus adapter</td>
<td>Broadcom (Emulex)</td>
<td>Each server: 2 x dual port 32 Gb HBAs (total of 4 initiator ports)</td>
<td>2 x LPe32002 (per server)</td>
</tr>
<tr>
<td>(HBA)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle Database</td>
<td>Oracle Database and Grid Infrastructure 19c with ASM</td>
<td>Four-node Oracle RAC</td>
<td>Oracle Database and Grid Infrastructure 19.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Benchmark tools   | OLTP and DSS                                                         | OLTP: SLOB 2.4  
DSS: Lineitem table created using TPC-H tools (dbgen). |         |

We set ASM disk groups with external redundancy for all the disk groups except the +GRID disk group, which we set with normal redundancy. The +DATA ASM disk group contained the data files and the +REDO ASM disk group contained the redo logs. The redo logs were striped using the fine-grain ASM striping template (128 KB).

In the OLTP tests we used 16 paths per device (16 front-end ports). In the DSS tests we used 24 paths per device (24 front-end ports). See the best practices section: Storage connectivity for considerations around number of paths per device and storage ports.

### OLTP performance test cases

#### OLTP test cases overview and results summary

We used SLOB 2.4 to run the OLTP test cases. The first test case demonstrates a “worst-case scenario”. It simulates a complete “read-miss” workload, or a case in which the PowerMax cache doesn’t benefit reads because most if not all of the reads are serviced from the storage flash media. This test case shows that while PowerMax cache provides a huge advantage and serves as a differentiator from other all-flash storage systems, even without its help, PowerMax delivers high performance and low latencies.

The second test case demonstrates a more realistic workload in which customers access recent data, which is a small portion of the whole database. As a result, the data is partially cached in PowerMax to a typical 60% read-hit (60% of the reads’ data is found in PowerMax cache). As with the first test case, we focused on achieving high IOPS while maintaining relatively low latencies.

The third test case is similar to the second. However instead of trying to achieve high IOPS, we focused on low latencies. To do so, we reduced the system load but maintained a reasonable level of IOPS. The justification for this test case is that some database workloads place more importance on fast response time (low latencies) as opposed to maximum IOPS (for example, financial transactions, web counters, or credit card validation).
Table 2. OLTP performance test cases and summary of results (AWR-based data)

<table>
<thead>
<tr>
<th>Test case</th>
<th>Test details</th>
<th>Data files IOPS</th>
<th>Data file read latency (ms)</th>
<th>Log writer write latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6% read-hit, focus on high IOPS</td>
<td>324,402</td>
<td>0.58</td>
<td>0.43</td>
</tr>
<tr>
<td>2</td>
<td>60% read-hit, focus on high IOPS</td>
<td>489,944</td>
<td>0.49</td>
<td>0.72</td>
</tr>
<tr>
<td>3</td>
<td>60% read-hit, focus on low latency</td>
<td>279,972</td>
<td>0.23</td>
<td>0.31</td>
</tr>
</tbody>
</table>

**Test case 1. OLTP, 6 percent cache read-hit, high IOPS**

The goal of test case 1 was to test a complete “read-miss” workload (we achieved 6 percent read-hit), while achieving the highest possible IOPS level with good latencies. We ran SLOB across the entire 2.8 TB database. This behavior is not realistic because PowerMax cache algorithms are very efficient and normally produce a high read-hit rate. In addition, database workloads tend to access the most recent data, which is only a small portion of the whole database. The reason for this test was to demonstrate performance under “worst” conditions.

Figure 3 shows the Oracle AWR IOPS for test case 1. The report shows that on average, the database performance during the test was 248,500 read IOPS and 75,902 write IOPS, for a total of 324,402 IOPS.

**System Statistics (Global)**

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
<th>Average</th>
<th>Std Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read I/O requests</td>
<td>448,082,675</td>
<td>248,464.87</td>
<td>216.99</td>
<td>62,116.22</td>
<td>792.25</td>
<td>61,168.22</td>
<td>63,020.54</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>3,670,791,839,744</td>
<td>2,035,478,507.93</td>
<td>1,775,188.62</td>
<td>508,869,726.96</td>
<td>6,491,659.42</td>
<td>501,100,719.26</td>
<td>516,278,364.84</td>
</tr>
<tr>
<td>physical read total IO requests</td>
<td>448,145,629</td>
<td>248,489.78</td>
<td>216.72</td>
<td>62,124.96</td>
<td>792.57</td>
<td>61,176.50</td>
<td>63,029.50</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>3,696,308,049,408</td>
<td>2,049,627,820.57</td>
<td>1,787,528.22</td>
<td>512,405,956.14</td>
<td>6,736,613.39</td>
<td>604,356,332.64</td>
<td>520,020,758.13</td>
</tr>
<tr>
<td>physical reads</td>
<td>24,343</td>
<td>13.50</td>
<td>0.01</td>
<td>3.37</td>
<td>0.25</td>
<td>3.09</td>
<td>3.62</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>448,094,707</td>
<td>248,471.55</td>
<td>216.70</td>
<td>62,117.89</td>
<td>792.44</td>
<td>61,169.52</td>
<td>63,022.26</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>12,043</td>
<td>6.68</td>
<td>0.01</td>
<td>1.67</td>
<td>0.25</td>
<td>1.31</td>
<td>1.99</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>8</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>physical reads direct temporary tablespace</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>physical write I/O requests</td>
<td>132,423,356</td>
<td>73,429.65</td>
<td>54.04</td>
<td>18,357.41</td>
<td>237.44</td>
<td>18,063.19</td>
<td>18,611.67</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>1,095,950,278,856</td>
<td>607,711,841.24</td>
<td>529,999.67</td>
<td>151,927,960.31</td>
<td>1,963,937.58</td>
<td>149,496,505.89</td>
<td>154,036,033.95</td>
</tr>
<tr>
<td>physical write total IO requests</td>
<td>136,881,740</td>
<td>75,501.36</td>
<td>68.20</td>
<td>18,975.46</td>
<td>247.89</td>
<td>18,689.44</td>
<td>19,239.13</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,140,744,526,848</td>
<td>632,550,556.05</td>
<td>551,662.99</td>
<td>158,113,638.76</td>
<td>2,040,630.07</td>
<td>166,617,597.12</td>
<td>169,336,183.89</td>
</tr>
<tr>
<td>physical write total multi block requests</td>
<td>314</td>
<td>0.17</td>
<td>0.00</td>
<td>0.04</td>
<td>0.02</td>
<td>0.03</td>
<td>0.06</td>
</tr>
<tr>
<td>physical writes</td>
<td>133,782,993</td>
<td>74,183.57</td>
<td>54.70</td>
<td>18,545.89</td>
<td>239.74</td>
<td>18,249.89</td>
<td>18,803.23</td>
</tr>
</tbody>
</table>

**Figure 3. Test case 1. AWR IOPS results**

Figure 4 shows the Oracle AWR response times for test case 1. The report shows that on average, the data files read response time was 579.8 microseconds (0.58 ms), and the redo log writer response time was 422.7 microseconds (0.43 ms).
PowerMax and Oracle performance tests

Top Timed Events

<table>
<thead>
<tr>
<th>#</th>
<th>Class</th>
<th>Event</th>
<th>Waits</th>
<th>%Timeouts</th>
<th>Total(s)</th>
<th>Avg Wait</th>
<th>%DB time</th>
<th>Avg</th>
<th>Min</th>
<th>Max</th>
<th>Std Dev</th>
<th>Cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User I/O</td>
<td>db file sequential read</td>
<td>446,424,508</td>
<td>0.00</td>
<td>258,824.07</td>
<td>575.77ms</td>
<td>94.46</td>
<td>579.88us</td>
<td>570.31us</td>
<td>591.03us</td>
<td>8.98us</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>System I/O</td>
<td>gc file parallel write</td>
<td>2,295,545</td>
<td>0.00</td>
<td>690.75</td>
<td>627.24ms</td>
<td>0.36</td>
<td>427,38us</td>
<td>417.56us</td>
<td>444.66us</td>
<td>12.50us</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Cluster</td>
<td>gc or grant 2-way</td>
<td>9,243,472</td>
<td>0.00</td>
<td>556.23</td>
<td>103.45ms</td>
<td>0.35</td>
<td>103.55us</td>
<td>96.03us</td>
<td>106.62us</td>
<td>5.04us</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>System I/O</td>
<td>db file parallel write</td>
<td>2,216,219</td>
<td>0.00</td>
<td>280.52</td>
<td>126.62ms</td>
<td>0.10</td>
<td>127.50us</td>
<td>105.51us</td>
<td>145.88us</td>
<td>18.02us</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Cluster</td>
<td>gc buffer busy release</td>
<td>345</td>
<td>0.00</td>
<td>241.92</td>
<td>699.19us</td>
<td>0.09</td>
<td>579.11ms</td>
<td>567.38ms</td>
<td>579.94ms</td>
<td>97.91ms</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>User I/O</td>
<td>ASM IO for non-blocking poll</td>
<td>2,779,340</td>
<td>0.00</td>
<td>57.10</td>
<td>20.54us</td>
<td>0.02</td>
<td>20.86us</td>
<td>15.96us</td>
<td>22.75us</td>
<td>3.27us</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>Other</td>
<td>ASM file metadata operation</td>
<td>21,916</td>
<td>0.00</td>
<td>13.04</td>
<td>594.91us</td>
<td>0.00</td>
<td>594.34us</td>
<td>571.80us</td>
<td>610.41us</td>
<td>16.33us</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>Other</td>
<td>KSV master wait</td>
<td>11,428</td>
<td>47.27</td>
<td>11.05</td>
<td>97ms</td>
<td>0.00</td>
<td>97ms</td>
<td>91.33us</td>
<td>1.00ms</td>
<td>38.15us</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>System I/O</td>
<td>control file sequential read</td>
<td>37,721</td>
<td>0.00</td>
<td>0.95</td>
<td>263.86us</td>
<td>0.00</td>
<td>263.02us</td>
<td>250.07us</td>
<td>276.72us</td>
<td>13.12us</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 4. Test case 1. AWR response time results

Figure 5 shows Unisphere performance metrics for Oracle data files (data_sg storage group). The flat lines indicate a very stable workload. The report shows 322,387 IOPS, 0.52 ms read response time, and 0.35 ms write response time.

These metrics are very similar to the metrics Oracle AWR reported. This is important because some storage systems report internal metrics that look artificially good, but don’t match the application performance. Unisphere metrics should be similar to Oracle AWR metrics; if not, investigate possible bottlenecks between the two (such as server queueing issues, or lack of sufficient connectivity).

Figure 5. Test case 1. Unisphere metrics: Oracle data files performance

Figure 6 shows Unisphere performance metrics for Oracle redo log files (redo_sg storage group). This test case demonstrates 1,485 IOPS with a 0.22 ms write response time for a 100% write workload. Archive logs were disabled during the test, although archive logs would not be expected to show a performance impact because archiving is performed in the background.
In summary, although test case 1 depicted an unrealistic behavior where the active dataset of the workload is so large that it doesn't benefit at all from cache algorithms (a read-miss workload), we still achieved over 320K IOPS with a data files read response time of 0.58 ms, which is very good.

**Test case 2. OLTP, 60 percent cache read-hit, high IOPS**

Test case 2 features a more realistic workload in which the active dataset benefits from the PowerMax cache (at 60% cache read-hit) with high IOPS and minimal latency. To simulate an active dataset smaller than the full size of the database, we used the SLOB 'hot-spot' feature that allows each database user to access a portion of the full data.

Figure 7 shows the Oracle AWR IOPS for test case 2. On average, the database performance during the test was 373,310 read IOPS and 116,635 write IOPS, for a total of 489,945 IOPS.
PowerMax and Oracle performance tests

System Statistics (Global)

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read IO requests</td>
<td>673,127,417</td>
<td>373,253.23</td>
<td>208.62</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>6,238,654,485,700</td>
<td>3,458,200,096.89</td>
<td>1,932,697.14</td>
</tr>
<tr>
<td>physical read total IO requests</td>
<td>673,229,026</td>
<td>373,306.57</td>
<td>208.65</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>6,275,332,096,912</td>
<td>3,479,708,463.35</td>
<td>1,944,915.47</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>5,812,857</td>
<td>3,223.26</td>
<td>1.80</td>
</tr>
<tr>
<td>physical reads</td>
<td>761,298,130</td>
<td>422,144.20</td>
<td>235.95</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>682,654,565</td>
<td>378,536.08</td>
<td>211.58</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>14,459,423</td>
<td>8,017.81</td>
<td>4.48</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>78,643,590</td>
<td>43,608.22</td>
<td>24.37</td>
</tr>
<tr>
<td>physical reads direct temporary tablespace</td>
<td>105</td>
<td>0.05</td>
<td>0.00</td>
</tr>
<tr>
<td>physical write IO requests</td>
<td>203,458,425</td>
<td>112,618.93</td>
<td>63.00</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>1,684,028,514,304</td>
<td>933,804,043.95</td>
<td>521,914.45</td>
</tr>
<tr>
<td>physical write total IO requests</td>
<td>210,340,555</td>
<td>116,659.77</td>
<td>65.19</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,754,065,664,512</td>
<td>972,640,068.48</td>
<td>543,638.08</td>
</tr>
<tr>
<td>physical write total multi block requests</td>
<td>252</td>
<td>0.14</td>
<td>0.00</td>
</tr>
<tr>
<td>physical writes</td>
<td>205,569,887</td>
<td>113,988.75</td>
<td>63.71</td>
</tr>
</tbody>
</table>

Figure 7. Test case 2. AWR IOPS results

Figure 8 shows the Oracle AWR response times for test case 2. On average, the data files read response time was 490.2 microseconds (0.49 ms), and the redo log writer response time was 724.4 microseconds (0.72 ms).

Top Timed Events

<table>
<thead>
<tr>
<th>By</th>
<th>Class</th>
<th>Event</th>
<th>Waits</th>
<th>%Timeouts</th>
<th>Total(s)</th>
<th>Avg Wait</th>
<th>%DB time</th>
<th>Avg</th>
<th>Min</th>
<th>Max</th>
<th>Std Dev</th>
<th>Cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>User I/O</td>
<td>db file sequential read</td>
<td>564,001,750</td>
<td>0.00</td>
<td>325,458.94</td>
<td>400.22us</td>
<td>94.14</td>
<td>490.28us</td>
<td>483.69us</td>
<td>497.49us</td>
<td>6.29us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>User I/O</td>
<td>DB CPU</td>
<td>37,170.34</td>
<td>0.00</td>
<td>37,170.34</td>
<td>37,170.34</td>
<td>10.75</td>
<td>724.12us</td>
<td>709.89us</td>
<td>741.21us</td>
<td>14.55us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>System I/O</td>
<td>log file parallel write</td>
<td>3,523,513</td>
<td>0.00</td>
<td>2,552.46</td>
<td>272.41us</td>
<td>0.74</td>
<td>760.74ms</td>
<td>703.72ms</td>
<td>836.49ms</td>
<td>57.64ms</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>Cluster</td>
<td>gc buffer busy release</td>
<td>1,738</td>
<td>0.00</td>
<td>1,737.56</td>
<td>792.61ms</td>
<td>0.40</td>
<td>98.09us</td>
<td>96.22us</td>
<td>100.09us</td>
<td>1.65us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>User I/O</td>
<td>db file scattered read</td>
<td>1,036,180</td>
<td>0.00</td>
<td>639.74</td>
<td>617.40us</td>
<td>0.18</td>
<td>91.16us</td>
<td>86.70us</td>
<td>91.53us</td>
<td>1.97us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>System I/O</td>
<td>db file parallel write</td>
<td>7,124,953</td>
<td>0.00</td>
<td>633.91</td>
<td>88.97us</td>
<td>0.18</td>
<td>99.16us</td>
<td>94.07us</td>
<td>101.09us</td>
<td>1.97us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>User I/O</td>
<td>read by other session</td>
<td>1,216,222</td>
<td>0.00</td>
<td>604.38</td>
<td>496.12us</td>
<td>0.17</td>
<td>496.16us</td>
<td>406.26us</td>
<td>504.56us</td>
<td>7.57us</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>User I/O</td>
<td>direct path read</td>
<td>576,371</td>
<td>0.00</td>
<td>313.30</td>
<td>541.69us</td>
<td>0.09</td>
<td>541.66us</td>
<td>541.69us</td>
<td>541.69us</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>*</td>
<td>Cluster</td>
<td>gc or multi block</td>
<td>799,603</td>
<td>0.00</td>
<td>204.72</td>
<td>256.03us</td>
<td>0.06</td>
<td>203.34us</td>
<td>176.92us</td>
<td>256.03us</td>
<td>36.00us</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 8. Test case 2 AWR response time results

Figure 9 shows Unisphere performance metrics for Oracle data files (data_sg storage group). The flat lines indicate a very stable workload. This test produced 481,349 IOPS with a 0.46 ms read response time and a 0.68 ms write response time.
In summary, with a more realistic workload behavior, our lab environment was able to generate close to half a million IOPS at less than half a millisecond read response time. This higher transaction rate caused the redo log write latency to rise slightly to 0.7 ms, which is still quite good.

**Test case 3. OLTP with 60 percent cache read-hit and low response times**

The goal of test case 3 was to focus on generating low database read response times. To do so, we reduced the load on the database because some applications place more importance on low latencies than on high IOPS.
Figure 11 shows the Oracle AWR IOPS for test case 3. The test results show that on average, the database generated 212,548 read IOPS and 67,426 write IOPS, for a total of 279,973 IOPS.

### System Statistics (Global)

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
<th>Average</th>
<th>Std Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read IO requests</td>
<td>383,256,402</td>
<td>212,511.17</td>
<td>201.54</td>
<td>53,127.79</td>
<td>326.52</td>
<td>52,672.97</td>
<td>53,435.83</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>3,140,162,699,264</td>
<td>1,741,183,292.71</td>
<td>1,651,267.57</td>
<td>435,295,823.18</td>
<td>2,678,782.29</td>
<td>431,564,476.73</td>
<td>437,827,083.79</td>
</tr>
<tr>
<td>physical read total IO requests</td>
<td>383,322,556</td>
<td>212,541.57</td>
<td>201.57</td>
<td>53,136.89</td>
<td>326.07</td>
<td>52,682.43</td>
<td>53,443.84</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>3,165,858,295,808</td>
<td>1,755,320,301.38</td>
<td>1,664,874.54</td>
<td>438,830,075.34</td>
<td>2,392,150.69</td>
<td>435,348,495.54</td>
<td>440,597,511.94</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>26,255</td>
<td>14.56</td>
<td>0.01</td>
<td>3.64</td>
<td>0.46</td>
<td>2.94</td>
<td>3.92</td>
</tr>
<tr>
<td>physical reads</td>
<td>383,320,642</td>
<td>212,546.79</td>
<td>201.57</td>
<td>53,136.70</td>
<td>327.00</td>
<td>52,681.21</td>
<td>53,445.69</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>383,317,806</td>
<td>212,545.22</td>
<td>201.57</td>
<td>53,136.30</td>
<td>327.00</td>
<td>52,680.82</td>
<td>53,445.29</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>64,079</td>
<td>35.53</td>
<td>0.03</td>
<td>8.88</td>
<td>1.06</td>
<td>7.83</td>
<td>10.00</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>2,836</td>
<td>1.57</td>
<td>0.00</td>
<td>0.39</td>
<td>0.00</td>
<td>0.39</td>
<td>0.40</td>
</tr>
<tr>
<td>physical write IO requests</td>
<td>117,759,793</td>
<td>65,296.42</td>
<td>61.92</td>
<td>16,324.11</td>
<td>134.94</td>
<td>16,128.14</td>
<td>16,431.67</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>974,915,870,720</td>
<td>540,579,386.00</td>
<td>512,663.55</td>
<td>135,144,846.50</td>
<td>1,134,738.15</td>
<td>133,497,454.09</td>
<td>136,049,344.12</td>
</tr>
<tr>
<td>physical write total IO requests</td>
<td>121,600,093</td>
<td>67,425.82</td>
<td>63.94</td>
<td>16,856.46</td>
<td>135.27</td>
<td>16,659.96</td>
<td>16,962.70</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,015,920,395,264</td>
<td>563,315,912.68</td>
<td>534,225.95</td>
<td>140,828,978.17</td>
<td>1,177,888.55</td>
<td>139,120,126.89</td>
<td>141,769,609.70</td>
</tr>
<tr>
<td>physical write total multi block requests</td>
<td>227</td>
<td>0.13</td>
<td>0.00</td>
<td>0.03</td>
<td>0.01</td>
<td>0.02</td>
<td>0.05</td>
</tr>
<tr>
<td>physical writes</td>
<td>119,008,285</td>
<td>65,988.69</td>
<td>62.58</td>
<td>16,497.17</td>
<td>138.52</td>
<td>16,296.08</td>
<td>16,607.59</td>
</tr>
</tbody>
</table>

Figure 12. Test case 3. AWR response time results

Figure 12 shows the Oracle AWR response times for test case 3. On average, the data files read response time was 232.0 microseconds (0.23 ms), and the redo log writer response time was 308.6 microseconds (0.31 ms).

### Top Timed Events

<table>
<thead>
<tr>
<th>Id</th>
<th>Class</th>
<th>Event</th>
<th>Wait Time</th>
<th>Summary Avg Wait Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Wait</td>
<td>Summary</td>
<td>Avg</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total(s)</td>
<td>Avg Time</td>
<td>Avg</td>
</tr>
<tr>
<td>1</td>
<td>User I/O</td>
<td>db file sequential read</td>
<td>66,509.55</td>
<td>232.0002</td>
</tr>
<tr>
<td>2</td>
<td>System I/O</td>
<td>log file parallel write</td>
<td>626.28</td>
<td>305.62</td>
</tr>
<tr>
<td>3</td>
<td>Cluster</td>
<td>gc cr grant 2-way</td>
<td>460.51</td>
<td>119.88</td>
</tr>
<tr>
<td>4</td>
<td>System I/O</td>
<td>db file parallel write</td>
<td>202.68</td>
<td>140.48</td>
</tr>
<tr>
<td>5</td>
<td>Cluster</td>
<td>gc buffer busy release</td>
<td>91.66</td>
<td>565.81</td>
</tr>
<tr>
<td>6</td>
<td>User I/O</td>
<td>read by other session</td>
<td>63.53</td>
<td>204.75</td>
</tr>
<tr>
<td>7</td>
<td>User I/O</td>
<td>ASM IO for non-blocking poll</td>
<td>37.93</td>
<td>109.30</td>
</tr>
<tr>
<td>8</td>
<td>Configuration</td>
<td>enq: HW - contention</td>
<td>37.89</td>
<td>19.67</td>
</tr>
<tr>
<td>9</td>
<td>Concurrency</td>
<td>buffer busy waits</td>
<td>2.44</td>
<td>11.38</td>
</tr>
</tbody>
</table>

Figure 13 shows Unisphere performance metrics for Oracle data files (data_sg storage group). The flat lines indicate a very stable workload. Testing generated 279,116 IOPS, a 0.14 ms read response time, and a 0.15 ms write response time.
PowerMax and Oracle performance tests

21

Deployment Best Practices for Oracle Database with Dell EMC PowerMax
White Paper

Figure 13. Test case 3. Unisphere metrics: Oracle data files performance

Figure 14 shows Unisphere performance metrics for Oracle redo log files (redo_sg storage group). The testing generated 1,317 IOPS, and a 0.13 ms write response time.

Figure 14. Test case 3. Unisphere metrics: Oracle redo logs performance

In summary, to achieve very low latencies we need to keep the system utilization moderate, but not too high (as IOPS increase, latencies also increase). Compared to test case 2 (489,934 IOPS), test case 3 generated 279,073 IOPS, which was about 60% of the IOPS of test case 2. We can achieve higher IOPS and maintain low latencies by scaling up the system (adding servers and bricks).
DSS tests overview and results summary

Decision Support Systems (DSS) test cases show PowerMax capabilities for servicing sequential reads, similar to those used by data warehouses and analytics queries. Unlike OLTP tests, the focus of DSS tests is not on IOPS or latencies, but on bandwidth (GB/s). The higher the bandwidth, the faster the report execution completes.

We used the dbgen toolkit from TPC-H tools to generate nearly 1 TB of data for the Lineitem table, with a primary partition by date and a secondary hash partition. To force a full-table scan, we used a hint in the SQL query and confirmed by reviewing the execution plan. We ran the query in a loop without delay to ensure that each test lasted 30 minutes in a steady state run. This scenario simulates a database with many users performing similar analytics queries, as they look at the same business data from different angles.

In our tests we used a database multi-block read I/O size of 128 KB. See the best practices section: Oracle sequential read I/O size section to learn more on setting the multi-block read I/O size for sequential reads.

We ran two DSS test cases. The first test case is more typical of large data warehouses in which the dataset scanned is much larger than the PowerMax cache. It will be a low read-hit workload.

The second test case shows the benefit of a read-hit workload. We performed this test using the Oracle partition pruning feature which optimizes the SQL query to look for the data in a subset of the dataset instead of the whole dataset. When partition pruning was used, the dataset of the query was much smaller and fit nicely in PowerMax cache to provide a higher bandwidth. While it isn’t always possible to optimize Oracle queries, it is certainly beneficial as the results show.

The following table summarizes the test cases and AWR results.

### Table 3. DSS performance test cases and summary of results

<table>
<thead>
<tr>
<th>Test case</th>
<th>Test details</th>
<th>Cache read-hit %</th>
<th>MBRC</th>
<th>Data file read GB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Full-table scan, no partition pruning</td>
<td>31</td>
<td>16 (128 KB)</td>
<td>11.8</td>
</tr>
<tr>
<td>2</td>
<td>Full-table scan, With partition pruning</td>
<td>100</td>
<td>16 (128 KB)</td>
<td>29.7</td>
</tr>
</tbody>
</table>

**Test case 1. DSS, full-table scan without partition pruning**

The goal of test case 1 was to perform a full-table scan of the large Lineitem table without any query optimizations. While it represents a ‘worst case scenario’ it is hard to predict how analytics and data warehouse queries are built, and if the Oracle Optimizer will be able to take advantage of partition pruning, especially when often users create ‘ad-hoc’ queries using Business Intelligence (BI) tools that are not optimized for query execution.

Figure 15 shows the Oracle AWR bandwidth for test case 1. The report shows that on average, the data files read bandwidth was 12,703,573,103 bytes/sec, or 11.8 GB/sec.
System Statistics (Global)

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read I/O requests</td>
<td>240,894,031</td>
<td>97,171.84</td>
<td>1,771,279.64</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>31,492,251,025</td>
<td>12,703,318,428.61</td>
<td>231,560,069,304.47</td>
</tr>
<tr>
<td>physical read total I/O requests</td>
<td>240,932,570</td>
<td>97,167.18</td>
<td>1,771,563.01</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>31,492,882,382</td>
<td>12,703,571,102.39</td>
<td>231,565,311,638.59</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>238,578,557</td>
<td>96,237.82</td>
<td>1,754,254.10</td>
</tr>
<tr>
<td>physical reads</td>
<td>3,844,268,924</td>
<td>1,550,698.05</td>
<td>28,266,683.26</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>20,671</td>
<td>8.34</td>
<td>151.99</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>12,152</td>
<td>4.90</td>
<td>89.35</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>3,844,248,253</td>
<td>1,550,689.71</td>
<td>28,266,531.27</td>
</tr>
</tbody>
</table>

Figure 15. Test case 1. AWR bandwidth results

Figure 16 shows Unisphere performance metrics for Oracle data files (tpch_sg storage group). The flat lines indicate a very stable workload. The report shows 12,135 MB/sec (11.85 GB/sec) which is very similar to the bandwidth Oracle AWR reported. It also shows that the average read I/O size was 128 KB, as the tests used MBRC of 16. Finally, the report shows a read response time of 1.7 ms, which is based on the 128KB average read I/O size.

Figure 16. Test case 1. Unisphere metrics: Oracle data files performance

In summary, although test case 1 is a low read-hit workload (31%) due to a large dataset, it still provided Oracle with a bandwidth of 11.8 GB/sec, even considering our small configuration featuring a single PowerMax brick.

Test case 2. DSS, full-table scan with partition pruning

In test case 2, we performed the same full-table scan of the large Lineitem table, but this time we used a WHERE clause in the SQL syntax to allow Oracle to reduce the number of partitions it needs to scan to fetch the required data (an optimization Oracle refers to as partition pruning). By limiting the amount of data to fetch, the query benefits from PowerMax cache and finishes faster.

Figure 17 shows the Oracle AWR bandwidth for test case 2. As shown in this report, the data files read bandwidth was 31,114,539,806 bytes/sec, or 29.7 GB/sec.
PowerMax and Oracle performance tests

System Statistics (Global)

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
<th>per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read I/O requests</td>
<td>437,447,745</td>
<td>238,089.05</td>
<td>1,792,818.63</td>
<td>59,522.26</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>57,167,123,570,688</td>
<td>31,114,268,114.52</td>
<td>234,291,490,043.80</td>
<td>7,778,567,028.63</td>
</tr>
<tr>
<td>physical read total I/O requests</td>
<td>437,477,058</td>
<td>238,105.01</td>
<td>1,792,938.76</td>
<td>59,526.25</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>57,167,622,756,864</td>
<td>31,114,529,805.86</td>
<td>234,293,535,888.79</td>
<td>7,778,634,051.47</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>432,910,419</td>
<td>235,619.53</td>
<td>1,774,223.03</td>
<td>58,904.88</td>
</tr>
<tr>
<td>physical reads</td>
<td>6,978,408,639</td>
<td>3,798,128.43</td>
<td>28,600,035.41</td>
<td>949,532.11</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>6.555</td>
<td>3.576</td>
<td>26.861</td>
<td>0.890</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>3.871</td>
<td>2.119</td>
<td>15.861</td>
<td>1.050</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>6,978,402,084</td>
<td>3,798,124.86</td>
<td>28,600,008.54</td>
<td>949,531.22</td>
</tr>
</tbody>
</table>

Figure 17. Test case 2. AWR bandwidth results

Figure 18 shows Unisphere performance metrics for Oracle data files (tpch_sg storage group). The flat lines indicate a very stable workload. The report shows a data files read bandwidth of 29,832 MB/sec (29.13 GB/sec) which is very similar to the bandwidth Oracle AWR reported. It also shows that the average read I/O size was 128 KB, as the tests used MBRC of 16. Finally, the report shows a read response time of 0.3 ms, even with the 128 KB I/O size. This result is due to the data already being cached in PowerMax.

Figure 18. Test case 2. Unisphere metrics: Oracle data files performance

In summary, in test case 2, where the query benefits from partition pruning, the outcome is a reduction in the amount of data requested by the query, and an opportunity for the PowerMax cache to increase the query performance.

Compression performance tests

This section shows the capability of PowerMax to maintain excellent performance for Oracle workloads, with or without storage group compression.

Recall from the PowerMax compression and deduplication section that the Adaptive Compression Engine (ACE) does not compress the most active data extents immediately, even if they belong to a storage group marked for compression. It keeps the most active 20 percent of the allocated storage capacity uncompressed (while storage space permits). Typically, the most recent data is being accessed most frequently. Over time, new data is written and accessed frequently. What was previously considered “hot” becomes less active and is automatically compressed.
While this method applies to real world database access patterns, benchmark tools tend to ignore it and run randomly across the whole database. The SLOB “hot-spot” feature allows access to a portion of each user table more frequently, simulating real world behavior.

To make the PowerMax compressions tests as realistic as possible, we loaded SLOB with semi-random data, resulting in a 3.0:1 compression ratio. We used a 5 GB buffer cache and SLOB hot-spot. This configuration resulted in a workload with 80 percent storage read I/Os and 60 percent cache read-hit. Therefore, 80 percent of the I/O requests sent to storage were reads, creating an OLTP type workload while also ensuring there were many requests for data that might be compressed. The 40 percent read-miss meant that of all the reads, at least 40 percent of the data was not found in the PowerMax cache and had to be brought from the flash media (compressed or not).

The SLOB workload was run using the two Dell servers in the cluster.

The following figure shows the compression-disabled test results taken from Oracle AWR. Under ‘Top Timed Events’, AWR reported a 0.28ms data file read latency (db file sequential read metric). Under ‘System Statistics (Global)’, the total data file IOPS were 253,477 (184,270 reads + 69,207 writes).

**Top Timed Events**

<table>
<thead>
<tr>
<th>#</th>
<th>Class</th>
<th>Event</th>
<th>Waits</th>
<th>%Timeouts</th>
<th>Total(s)</th>
<th>Avg Wait</th>
<th>%DD time</th>
<th>Avg</th>
<th>Min</th>
<th>Max</th>
<th>Std Dev</th>
<th>Cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>User I/O</td>
<td>db file sequential read</td>
<td>330,519,461</td>
<td>0.00</td>
<td>53,965.64</td>
<td>284.39us</td>
<td>91.33</td>
<td>284.39us</td>
<td>283.83us</td>
<td>284.56us</td>
<td>830.63ns</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>DB CPU</td>
<td></td>
<td>17,645.57</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>System I/O</td>
<td>log file parallel write</td>
<td>2,879,516</td>
<td>0.00</td>
<td>1,244.34</td>
<td>432.14us</td>
<td>1.21</td>
<td>432.14us</td>
<td>431.83us</td>
<td>432.65us</td>
<td>724.31ns</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>System I/O</td>
<td>db file parallel write</td>
<td>2,794,576</td>
<td>0.00</td>
<td>148.63</td>
<td>53.25us</td>
<td>0.14</td>
<td>53.25us</td>
<td>53.00us</td>
<td>53.55us</td>
<td>352.55ns</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>Other</td>
<td>RMA: IPC9 completion sync</td>
<td>3,727</td>
<td>0.00</td>
<td>72.60</td>
<td>19.43ms</td>
<td>0.07</td>
<td>19.43ms</td>
<td>19.42ms</td>
<td>19.44ms</td>
<td>10.75us</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>User I/O</td>
<td>read by other session</td>
<td>117,535</td>
<td>0.00</td>
<td>36.27</td>
<td>367.54us</td>
<td>0.04</td>
<td>367.54us</td>
<td>366.03us</td>
<td>368.09us</td>
<td>2.18us</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>Other</td>
<td>LGWR: worker group</td>
<td>104,003</td>
<td>0.00</td>
<td>9.99</td>
<td>154.63us</td>
<td>0.03</td>
<td>154.63us</td>
<td>154.02us</td>
<td>155.26us</td>
<td>895.50ns</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>Other</td>
<td>LGWR: worker group ordering</td>
<td>57,615</td>
<td>0.00</td>
<td>9.99</td>
<td>173.36us</td>
<td>0.01</td>
<td>173.36us</td>
<td>173.88us</td>
<td>173.85us</td>
<td>885.49ns</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>Application</td>
<td>concurrency: TX - row lock contention</td>
<td>709</td>
<td>0.00</td>
<td>4.99</td>
<td>7.04ms</td>
<td>0.00</td>
<td>7.04ms</td>
<td>6.65ms</td>
<td>7.22ms</td>
<td>294.03us</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>System I/O</td>
<td>control file sequential read</td>
<td>20,670</td>
<td>0.00</td>
<td>3.99</td>
<td>152.83us</td>
<td>0.00</td>
<td>152.83us</td>
<td>152.40us</td>
<td>153.26us</td>
<td>538.52ns</td>
<td>2</td>
</tr>
</tbody>
</table>

**System Statistics (Global)**

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
<th>Average</th>
<th>Std Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read I/O requests</td>
<td>331,383,665</td>
<td>184,246.89</td>
<td>111.74</td>
<td>92,120.34</td>
<td>1,491.23</td>
<td>91,065.38</td>
<td>93,174.30</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>2,714,833,543,168</td>
<td>1,509,376,668.05</td>
<td>915,426.45</td>
<td>754,688,334.03</td>
<td>12,204,639.86</td>
<td>746,058,774.68</td>
<td>763,317,893.37</td>
</tr>
<tr>
<td>physical read total I/O requests</td>
<td>331,437,622</td>
<td>184,270.68</td>
<td>111.78</td>
<td>92,135.34</td>
<td>1,491.02</td>
<td>91,081.03</td>
<td>93,180.55</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>2,748,847,705,672</td>
<td>1,523,267,657.14</td>
<td>926,895.94</td>
<td>764,143,228.57</td>
<td>12,090,945.48</td>
<td>755,554,239.03</td>
<td>772,663,418.11</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>32,882</td>
<td>18.28</td>
<td>0.01</td>
<td>9.14</td>
<td>0.18</td>
<td>9.02</td>
<td>9.27</td>
</tr>
<tr>
<td>physical reads</td>
<td>331,400,582</td>
<td>184,250.08</td>
<td>111.75</td>
<td>92,125.04</td>
<td>1,489.75</td>
<td>91,071.63</td>
<td>93,178.45</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>331,400,580</td>
<td>184,250.08</td>
<td>111.75</td>
<td>92,125.04</td>
<td>1,489.75</td>
<td>91,071.63</td>
<td>93,178.45</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>15,914</td>
<td>9.40</td>
<td>0.01</td>
<td>4.70</td>
<td>1.48</td>
<td>4.65</td>
<td>5.75</td>
</tr>
<tr>
<td>physical write I/O requests</td>
<td>110,030,526</td>
<td>66,122.49</td>
<td>40.10</td>
<td>33,061.25</td>
<td>329.50</td>
<td>32,626.26</td>
<td>33,294.24</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>984,146,862,592</td>
<td>552,741,974.33</td>
<td>335,234.18</td>
<td>278,370,987.16</td>
<td>3,105,245.41</td>
<td>274,175,247.08</td>
<td>278,566,727.25</td>
</tr>
<tr>
<td>physical write total I/O requests</td>
<td>124,479,110</td>
<td>65,267.14</td>
<td>41.97</td>
<td>34,603.57</td>
<td>353.26</td>
<td>34,353.76</td>
<td>34,653.33</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,056,549,963,560</td>
<td>587,409,231.75</td>
<td>356,259.51</td>
<td>293,704,815.87</td>
<td>3,370,199.62</td>
<td>291,321,524.87</td>
<td>296,087,706.88</td>
</tr>
</tbody>
</table>

Figure 19. Storage group Compression disabled, AWR statistics

The following figure shows compression-enabled test results taken from Oracle AWR. Under ‘Top Timed Events’, AWR reports a 0.31ms data file read latency (the db file
PowerMax data reduction at work

sequential read metric). Under ‘System Statistics (Global)’, the total data file IOPS were 250,743 (181,296 reads + 69,447 writes).

The two AWR reports for the PowerMax storage system with enabled and disabled compression show an approximate 1 percent difference in Oracle data file total IOPS and a 0.03 ms data file read response time difference. Users do not perceive these differences, which demonstrates the strength of the PowerMax architecture that supports data reduction while maintaining high performance.

Top Timed Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Wait</th>
<th>%Time waits</th>
<th>Total(s)</th>
<th>%DB time</th>
<th>Avg Wait</th>
<th>Min</th>
<th>Max</th>
<th>Std Dev</th>
<th>Conn</th>
</tr>
</thead>
<tbody>
<tr>
<td>User I/O db file sequential read</td>
<td>325,190,738</td>
<td>0.00</td>
<td>100,801.50</td>
<td>310.26us</td>
<td>52.11</td>
<td>310.36us</td>
<td>369.42us</td>
<td>311.18us</td>
<td>1.25us</td>
</tr>
<tr>
<td>DE CPU</td>
<td></td>
<td></td>
<td>17,321.71</td>
<td></td>
<td>15.81</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System I/O log file parallel write</td>
<td>2,591,175</td>
<td>0.06</td>
<td>1,120.66</td>
<td>435.56us</td>
<td>1.03</td>
<td>435.56us</td>
<td>434.94us</td>
<td>436.23us</td>
<td>911.07ms</td>
</tr>
<tr>
<td>System I/O db file parallel write</td>
<td>2,212,040</td>
<td>0.06</td>
<td>83.52</td>
<td>42.20us</td>
<td>0.09</td>
<td>42.75us</td>
<td>39.09us</td>
<td>47.42us</td>
<td>6.68us</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td></td>
<td>72.92</td>
<td>18.41ms</td>
<td>0.07</td>
<td>19.41ms</td>
<td>19.40ms</td>
<td>19.42ms</td>
<td>14.87us</td>
</tr>
<tr>
<td>User I/O read by other session</td>
<td>123,383</td>
<td>0.06</td>
<td>40.96</td>
<td>231.59us</td>
<td>0.04</td>
<td>332.06us</td>
<td>331.70us</td>
<td>332.31us</td>
<td>431.08ms</td>
</tr>
<tr>
<td>Other LGWR by any worker group</td>
<td>127,307</td>
<td>0.06</td>
<td>25.44</td>
<td>196.62us</td>
<td>0.02</td>
<td>199.63us</td>
<td>199.79us</td>
<td>199.66us</td>
<td>44.01ms</td>
</tr>
<tr>
<td>Other LGWR worker group ordering</td>
<td>46,589</td>
<td>0.06</td>
<td>8.23</td>
<td>176.59us</td>
<td>0.01</td>
<td>176.58us</td>
<td>175.98us</td>
<td>177.17us</td>
<td>54.21ms</td>
</tr>
<tr>
<td>Application sql TX - row lock contention</td>
<td>75</td>
<td>0.00</td>
<td>7.23</td>
<td>9.50ms</td>
<td>0.01</td>
<td>9.50ms</td>
<td>7.75ms</td>
<td>11.30ms</td>
<td>2.55ms</td>
</tr>
<tr>
<td>System I/O control file sequential read</td>
<td>20,764</td>
<td>0.00</td>
<td>4.08</td>
<td>196.43us</td>
<td>0.00</td>
<td>196.44us</td>
<td>193.49us</td>
<td>199.39us</td>
<td>4.17us</td>
</tr>
</tbody>
</table>

System Statistics (Global)

<table>
<thead>
<tr>
<th>per Second</th>
<th>per Trans</th>
<th>Average</th>
<th>Std Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read IO requests</td>
<td>325,976,273</td>
<td>101,268.97</td>
<td>122.32</td>
<td>90,834.49</td>
<td>3,102.52</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>2,670,496,569,684</td>
<td>1,485,012,115.26</td>
<td>1,003,728.73</td>
<td>742,596,057.63</td>
<td>26,053,874.29</td>
</tr>
<tr>
<td>physical read total IO requests</td>
<td>326,926,029</td>
<td>101,268.97</td>
<td>122.32</td>
<td>90,849.32</td>
<td>3,102.45</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>2,700,603,355,883</td>
<td>1,485,012,115.26</td>
<td>1,003,728.73</td>
<td>750,799,297.26</td>
<td>26,053,874.29</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>26,377</td>
<td>15.76</td>
<td>0.01</td>
<td>7.69</td>
<td>0.07</td>
</tr>
<tr>
<td>physical reads</td>
<td>325,968,719</td>
<td>101,268.97</td>
<td>122.32</td>
<td>90,837.55</td>
<td>3,101.01</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>325,968,719</td>
<td>101,268.97</td>
<td>122.32</td>
<td>90,837.55</td>
<td>3,101.02</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>12,483</td>
<td>6.44</td>
<td>0.00</td>
<td>3.47</td>
<td>1.46</td>
</tr>
<tr>
<td>physical write IO requests</td>
<td>115,899,945</td>
<td>65,673.84</td>
<td>45.07</td>
<td>33,335.57</td>
<td>302.28</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>1,001,677,669,833</td>
<td>558,879,583.42</td>
<td>376,263.09</td>
<td>278,339,775.71</td>
<td>2,953,370.10</td>
</tr>
<tr>
<td>physical write total IO requests</td>
<td>124,667,555</td>
<td>65,447.56</td>
<td>46.94</td>
<td>34,723.78</td>
<td>319.16</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,057,276,619,584</td>
<td>587,392,349.54</td>
<td>367,387.04</td>
<td>263,986,174.77</td>
<td>3,058,270.11</td>
</tr>
</tbody>
</table>

Figure 20. Storage group Compression enabled, Top Timed Event

PowerMax data reduction at work

The following examples show the use and benefits of PowerMax compression and deduplication with Oracle databases. The first example shows the advantages of compression and deduplication of an Oracle database that is not encrypted. The second example shows what happens when the database is fully encrypted. Encryption makes the data appear completely random and interferes with compression benefits.

In both examples, we reported only the Oracle data files’ consumed capacity as reported by the database or storage. The capacity of the redo logs is not reported in the graphs, as it is relatively small.
The examples are based on a SLOB database which was modified so that the data is semi-random. As the example shows, the PowerMax storage system compressed the semi-random database at a 3.1:1 compression ratio, which is approximately the compression ratio that we expect for Oracle databases.

In this example we look at the Oracle data files’ consumed capacity reported from both the Oracle database (left-side bars in blue color) and the PowerMax storage system (right-side bars in green). When the Oracle database was created, the capacity of the data files was approximately 1.35 TB, reported by the Oracle database. Because the data_sg storage group (SG) had compression enabled, the storage it actually consumed was only 450 GB, reported by the PowerMax. This means a Data Reduction Ratio (DRR) of 3.1:1. Note that because in an Oracle database every data block has a unique header (regardless of its content), there are no deduplication benefits within a single database.

![Diagram showing PowerMax compression and deduplication of an Oracle database](image)

**Figure 21. Example 1: PowerMax compression and deduplication of an Oracle database**

We then created two SnapVX snapshots and linked (presented) them to another server. The snapshot creation and linking took only seconds. As a result, we had three copies of the original database, or approximately 4 TB (3 x 1.35 TB). When we inspected the storage, no capacity was added, resulting in a 9.3:1 DRR, because PowerMax snapshots only consume storage capacity when data is modified. We then deleted the snapshots.

Next, we created a database copy by using the RMAN DUPLICATE command. RMAN created a binary copy of the source database in a target database server and ASM disk group. Because RMAN used the network to make a full copy of the source database, the process took a few hours. After RMAN completed the database cloning operation, at a database level, we saw a 2.64 TB combined capacity of source and cloned databases. However, the storage capacity associated with the source and target storage groups was only 450 GB for a 6.0:1 DRR.
The reason for this result is because ASM Allocation Units (AU) are 4 MB with Oracle 12.2 and 1 MB with earlier releases. The PowerMax storage system, with 128 KB deduplication granularity, identified the cloned database extents as identical to the source and fully deduplicated them.

Finally, we created a second database copy using RMAN DUPLICATE command. Now, we had the source database and two copies for a total of 4 TB capacity at the database level. Again, the PowerMax storage system fully deduplicated the data and the storage capacity that was associated with the three databases remained 450 GB for a 9:1 DRR.

In this example, all the Oracle data files’ consumed capacity is reported from the storage system. The first bar from the left shows that the storage consumed capacity of the data files was approximately 1.35 TB, as in this example, the PowerMax storage group compression was not enabled initially.

We enabled PowerMax compression on data_sg and waited for the background compression to complete. At the end of the process, data_sg consumed only 465 GB, which translates to a 3.0:1 DRR. This result is not very different from the previous example where the database was created in an SG with compression already enabled.

Next, Oracle Transparent Database Encryption (TDE) was used to encrypt all the tablespaces. DBAs can choose to encrypt only certain table columns or a few tablespaces; however, we wanted to see the effect of encrypting the entire database. The result was that the data_sg storage consumption grew to 1.35 TB, or its original size. We can clearly see that database encryption negates the benefits of storage compression.

We did not create storage snapshots because we have already seen that they do not add capacity. We used the RMAN DUPLICATE command to clone the database. Initially, the target storage group did not have compression enabled. As a result, after RMAN finished, the total storage consumption doubled.
Finally, we enabled compression on the target storage group. PowerMax deduplication again provided 100 percent deduplication benefits and the storage consumption of both the source and target storage groups reverted to 1.35 TB.

**Conclusion**

We can see that PowerMax compression is extremely efficient for Oracle databases and, with the semi-random SLOB database, we achieved approximately 3:1 data reduction through compression.

When SnapVX is used to create database copies (which is the recommended method), the operation takes seconds and provides the most capacity efficiency benefits.

When the DBA uses RMAN DUPLICATE command to clone a database, the operation takes a long time because the full database is copied over the network. However, due to an ASM AU granularity of 1 MB or 4 MB, the PowerMax storage system can fully deduplicate the data because it is an identical binary copy of the source database.

When using Unisphere to create new storage groups, PowerMax compression is enabled by default. You can disable PowerMax compression by unchecking the compression checkbox in the SG creation wizard. Unisphere also includes views and metrics that show the compression ratio of compressed storage groups, potential compressibility of uncompressed storage groups, and more. The following section shows how to perform these operations or display data reduction-related information by using the Solutions Enabler Command Line Interface (CLI).

A storage group (data_sg in the examples) must be associated with the PowerMax storage system’s Storage Resource Pool (SRP) to enable compression. To enable compression and associate the SRP, type the following command:

```
# symsg -sg data_sg set -srp SRP_1 -compression
```

Similarly, to disable compression on a storage group where compression is enabled, type the following command:

```
# symsg -sg data_sg set -srp SRP_1 -nocompression
```

To display the compression ratio of a storage group, type the following command.

```
# symcfg list -tdev -sg data_sg -gb [-detail]
```

**Note:** The `-detail` option includes the data allocations in each compression pool and enables you to see exclusive allocations. When data is deduplicated, it does not consume exclusive allocations.

To display the estimated compression ratio of storage groups, including SGs with compression disabled, type the following command:

```
# symcfg list -sg_compression -by_compressibility -all
```

To display overall system efficiency, type the following command:
PowerMax Service Levels

Service Levels overview

With high-capacity and powerful NVMe flash storage such as the PowerMax storage system, there are often many databases and applications that are consolidated into one storage system. The PowerMax storage system uses Service Levels (SL) to determine the performance objectives and priorities of applications by managing the I/O latencies of the storage groups (SGs) in accordance with their SL.

By default, the PowerMax storage system assigns an Optimized SL to new SGs. This SL receives the best performance the system can give it, but has the same priority as all other SGs that are also set with the Optimized SL. In this case, it is possible that a sudden high load from one SG (such as an auxiliary application) might affect the performance of another SG (such as a key mission-critical application) because they all share the same system priorities and performance goals. Using specific SLs can prevent this situation.

Use cases for SLs include “caging” the performance of a “noisy neighbor”, prioritizing Production versus Test/Dev systems performance, and satisfying the needs of Service Providers or organizations using “chargeback” in which their clients pay for a service level.

Service Levels and their target response times

The following table lists the service level prioritizations and their associated performance target goals. When using SLs, critical systems can be assigned a high SL such as Diamond, Platinum, or Gold to ensure that their performance goals have a higher priority over applications with a lower SL such as Silver, Bronze, or Optimized.

In addition, some SLs have a lower response time limit, meaning that assigning them to SGs will force read and write latencies no lower than that limit. Also, notice that Diamond and Platinum SLs have lower target response times when SCM drives are used.

<table>
<thead>
<tr>
<th>Service Level</th>
<th>Target response time without SCM (ms)</th>
<th>Target response time with SCM (ms)</th>
<th>Lowest response time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diamond</td>
<td>0.6</td>
<td>0.4</td>
<td>N/A</td>
</tr>
<tr>
<td>Platinum</td>
<td>0.8</td>
<td>0.6</td>
<td>N/A</td>
</tr>
<tr>
<td>Gold</td>
<td>1.0</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Silver</td>
<td>3.6</td>
<td>N/A</td>
<td>~3.6</td>
</tr>
<tr>
<td>Bronze</td>
<td>7.2</td>
<td>N/A</td>
<td>~7.2</td>
</tr>
<tr>
<td>Optimized</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Unlike any of the other SLs, Optimized does not have a specific performance target. If SGs with a non-Optimized SL struggle to maintain their performance goals, they can add latency to SGs with Optimized SL as they try to preserve their own goals.

For more information about the PowerMax Adaptive Compression Engine and deduplication, see Data Reduction with Dell EMC PowerMax.

# symcfg list -efficiency -detail
Similarly, if any non-Optimized SGs struggle to maintain their performance goals, they can add latency to SGs that are set with a lower priority SL. For example, Diamond SGs can affect Platinum SGs, which can affect Gold SGs, and so on.

**Service Levels and SCM drives**

The PowerMaxOS Q3 2019 release introduced PowerMax support for storage class memory (SCM), in addition to the existing NAND SSD flash drives. The support for different drive technologies introduces tiered storage called *automated data placement* (ADP). The functionality of ADP is included as part of the service levels (SLs) and uses machine learning to place data on the appropriate drive type. To improve the response times managed by SLs, ADP places the most active data on the faster drive technology. ADP movements occur as either a promotion of data or a demotion of data. Promotions are the movement of data to the SCM drives, and demotions are the movement of data out of the SCM drives. PowerMaxOS uses the SL prioritization together with activity metrics to determine promotions and demotions, where the basic strategy is described in Table 5.

**Table 5. SCM drives promotion and demotion strategy based on Service Levels**

<table>
<thead>
<tr>
<th>Service Level</th>
<th>Priority</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Promotion</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diamond</td>
<td>Highest promotion priority</td>
<td>During optimal utilization, PowerMax OS attempts to put all data with Diamond SL on SCM drives</td>
</tr>
<tr>
<td>Platinum, Gold, Optimized</td>
<td>All data has equal priority</td>
<td></td>
</tr>
<tr>
<td>Silver, Bronze</td>
<td>Excluded from promotion</td>
<td></td>
</tr>
<tr>
<td><strong>Demotion</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Silver, Bronze</td>
<td>Highest demotion priority</td>
<td></td>
</tr>
<tr>
<td>Platinum, Gold, Optimized</td>
<td>Equal demotion priority</td>
<td>Demotions occur when there is a need to create available space in SCM for higher priority data or same priority data with higher activity</td>
</tr>
<tr>
<td>Diamond</td>
<td></td>
<td>Data is demoted when there is other Diamond SL data with higher activity</td>
</tr>
</tbody>
</table>

For more information about PowerMax Service Levels see: [Dell EMC PowerMax: Service Levels for PowerMaxOS Technical White Paper](#).
**Service Levels example with a single database workload**

The following figure shows the typical effect of service levels on a single Oracle database workload. In this test, a single OLTP workload ran without interruption or change. Only the data_sg SL changed every 30 minutes.

![Service Level changes on a single Oracle workload](image)

**Figure 23.** Service Level changes on a single Oracle workload

We see that a Bronze SL forced an average of 5 ms latency, and a performance level of 37,000 IOPS. After the SL changed to Silver, the latency dropped to 2 ms and IOPS increased to 79,000. The Gold SL reduced latency to 0.4 ms and IOPS increased to 192,000. The Platinum and Diamond SLs were not significantly different as they both performed at 0.3 ms latency and 204,000 IOPS.

When a SL changes, the effect is immediate because it takes place at the PowerMaxOS software layer. We also see that SL latencies affect both reads and writes I/O response times.

**Service Levels example with two database workloads**

The following figure shows the effect of Service Levels on two Oracle databases. In this test, two OLTP workloads ran without interruption or change. As seen on the left of the figure, the SL of the workload (represented by the top yellow line) is set to Diamond SL simulating a key mission critical application. The SL of the other workload (represented by the lower blue line) started at Bronze SL and was increased every 30 min until it reached Diamond SL.

![Service Level changes on two Oracle workloads](image)

**Figure 24.** Service Level changes on two Oracle workloads

We can see that as the "caged" application improved its SL, it slowly took more resources from the application with the Diamond SL, until they shared the same SL and system.
resources. This result demonstrates the value of setting a lower priority SL to lower priority applications.

**Best Practices for PowerMax and Oracle databases**

### Storage considerations

<table>
<thead>
<tr>
<th>Storage connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>In a typical SAN configuration, HBA ports (initiators) and storage front-end ports (targets) are connected to a switch. The switch software creates zones, pairing initiators and targets. Each pairing creates a physical path between the server and the storage through which I/Os can pass. For configuring server and storage connectivity, using SAN switches, the following best practices apply:</td>
</tr>
<tr>
<td>- For redundancy and high availability, use at least two switches so that if one is unavailable due to a fault or maintenance, the server does not lose storage access.</td>
</tr>
<tr>
<td>- Spread connectivity across storage engines, directors, I/O modules, and ports to achieve the best performance and availability (instead of allocating all ports on one storage director first, before moving to the next).</td>
</tr>
<tr>
<td>- It is recommended to avoid crossing switches when connecting server initiators to storage port targets. That is, avoid Inter-Switch Links (ISLs) because they are a shared resource, often with limited connectivity and unpredictable utilization.</td>
</tr>
<tr>
<td>- When considering server to storage connectivity, remember that at times, even in a clustered environment, a single node may perform data loads or RMAN backups. Plan connectivity appropriately.</td>
</tr>
</tbody>
</table>

The following points provide guidelines for number of paths per device and server to storage connectivity.

- Remember that each PowerMax brick has two directors which hold the front-end I/O modules, and each FC or FC-NVMe front-end I/O module has 4 ports. A single brick PowerMax 8000 supports up to 24 front-end ports per brick (3 x I/O modules per director), while a PowerMax 8000 with 2-8 bricks, as well as PowerMax 2000 with 1-2 bricks each support up to 32 front-end ports (4 x I/O modules per director). See Figure 16, 17, and 18 on pages 30-31 in the Dell EMC PowerMax Family Overview white paper to learn more about PowerMax director layout and I/O modules.

- For most OLTP workloads (mixed with some database reports and batch), 4 or 8 front-end ports (4 or 8 paths per device) can provide very good throughput (IOPS) and moderate bandwidth (GB/s). Figure 25 shows a connectivity example with 8 paths per device to a single brick PowerMax 8000.
For high-performance OLTP databases (mixed with reports and batch), 16 front-end ports (16 paths per device) can provide maximum throughput. Figure 26 shows a connectivity example with 16 paths per device to a single-brick PowerMax 8000.

For most DSS-focused workloads (data warehouses and Analytics applications) where the query dataset is large and doesn’t fit in PowerMax cache, 16 front-end ports can provide maximum bandwidth (GB/s). In cases where the dataset does fit in PowerMax cache (100% read-hit) additional front-end ports can provide higher bandwidth.

- In our high-performance DSS tests with 100% read-hit, with 16 ports we achieved ~23 GB/s and with 24 ports we achieved ~30 GB/s. Note that without partition pruning (~30% read-hit), the database bandwidth only reached ~12 GB/s. This shows that 16 ports are more than sufficient in most high-performance environments.
FC or FC-NVMe protocol choice and core allocations

When using FC protocol, the PowerMax 32Gb front-end modules are configured and referred to as FAs (FC front-end adapters). When using FC-NVMe protocol, the PowerMax 32Gb front-end modules use the same hardware, but are configured and referred to as FNs (FC-NVMe front-end adapters).

Because the PowerMax embedded management requires access to the storage, even when the system uses FNs exclusively for server connectivity, at least one port per director on the first engine (brick) must be configured as FA. When a port is configured, CPU cores from that director are assigned to it.

As a result, when comparing storage CPU core assignment between a single-engine system configured exclusively for FC server connectivity, to a system configured exclusively for FC-NVMe server connectivity, all other things being equal, there are slightly fewer cores to support FC-NVMe ports.

This difference is not important in most cases and does not detract from FC-NVMe advantages such as improved latency and optimized I/O access. In addition, the more engines (bricks) that are configured, the less of a difference this will make as it only affects the first engine. However, in cases where maximum IOPS are expected from the system (such as in acceptance tests and benchmarks), it could be seen as an advantage to using FC instead of FC-NVMe.

Masking Views

PowerMax uses masking views to determine which devices are visible to servers. A masking view contains a Storage Group (SG), Port Group (PG), and Initiator Group (IG). When you create a masking view, the devices in the SG are made visible to the server(s) initiators in the IG, with access to storage by the ports in the PG.

When changes are made to any of the masking view components, they automatically update the masking view. For example, adding devices to the SG automatically makes the new devices visible to the server through the initiators and ports in the masking view.

Storage group

A storage group (SG) contains a group of devices that are managed together. Additionally, an SG can contain other SGs, in which case the top level SG is called a parent SG, and the lower level SGs are called child SGs. In a parent/child SG configuration, devices are managed either by using any of the child SGs directly or by using the parent SG so that the operation affects all the child SGs. For example, use the parent SG for the masking view and the child SGs for database backup/recovery snapshots and more granular performance monitoring.

- For databases that do not require granular performance monitoring or snapshots capable of database backup and recovery, adding all the database devices into a single SG for masking alone is sufficient.
- For mission critical Oracle databases, we recommend separating the following database components into different ASM disk groups and matching SGs:
  - **data_sg**—Used for the database data, such as data files, control files, undo tablespace, system tablespace, and so on. By separating data from logs (separate data_sg and redo_sg), storage replications can be used for database
backup and recovery operations as well as more granular performance monitoring.

- **redo_sg**—Used for the database redo logs.

- **fra_sg**—Used for the database archive logs and flashback logs (if used). Note that flashback logs might consume much more capacity than the archive logs. Also, unlike archive logs, flashback logs must be consistent with the data files if protected with storage replications. For these reasons, consider separating archive logs and flashback logs to distinct ASM disk groups and SGs.

- **grid_sg**—Used for Grid Infrastructure (GI), which is a required component when using Oracle ASM or RAC (cluster). Even in single instance (non-clustered) deployments, we recommend that you create this ASM disk group and SG so that database data is not mixed with GI management components.

**Note:** For more information about ASM disk groups and matching SGs that can leverage fast storage replicas that are valid for database backup and recovery, see Oracle database backup, recovery, and replications best practices with VMAX All Flash storage.

### Initiator Group

An initiator group (IG) contains a group of server initiators’ (HBA ports) World Wide Names (WWNs) to which storage devices are mapped. Additionally, an IG can contain other IGs, in which case the top level IG is called a parent IG, and the lower level IGs are called child IGs.

A parent/child IG deployment is useful when the database is clustered. Each child IG contains the initiators from a single server and the parent IG aggregates all of them. When the masking view is created, the parent IG is used. When a cluster node is added or removed from the cluster, the masking view does not change. Only the parent IG is updated by adding or removing the child IG that matches the node that is being added or removed.

### Port Group

A port group (PG) contains a group of targets (storage front-end ports). When placed in a masking view, these are the storage ports through which the devices in the SG are accessed.

Because the physical connectivity is determined by the SAN zone sets, for simplicity of management, we recommend that you include all the storage ports that the database will be using in the PG. The specific path relationships between the PG ports and IG initiators are determined by the zone sets.

### Masking view

For environments that are not mission-critical, it is sufficient to create a simple masking view for the entire database with all the devices in a single SG, and therefore use a single masking view.

The following guidelines apply to high-performance, mission critical databases in which data and log SGs are separated to allow backup and recovery by using storage snapshots and more granular performance monitoring.
In this case, data_sg and redo_sg are joined under a parent dataredo_sg SG, and FRA is in its own SG. The following table shows that there are two masking views for the database and one for the cluster or Grid Infrastructure.

Table 6. Sample masking view design

<table>
<thead>
<tr>
<th>Masking view</th>
<th>Storage Group</th>
<th>Child SGs</th>
<th>Initiator Group</th>
<th>Child IGs</th>
<th>Port Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>App1_DataRedo</td>
<td>App1_DataRedo</td>
<td>App1_Data, App1_Redo</td>
<td>App1_servers</td>
<td>Server1, Server2, ...</td>
<td>PG1</td>
</tr>
<tr>
<td>App1_FRA</td>
<td>App1_FRA</td>
<td>(none)</td>
<td>(same as above)</td>
<td>(same as above)</td>
<td>(same as above)</td>
</tr>
<tr>
<td>Grid</td>
<td>Grid</td>
<td>(none)</td>
<td>(same as above)</td>
<td>(same as above)</td>
<td>(same as above)</td>
</tr>
</tbody>
</table>

If the database is clustered, the IG is a parent IG that contains the cluster nodes. If the database is not clustered, the IG can contain the single server initiators (no child IGs). Similarly, if the database is clustered, “Grid” ASM disk group devices can be in their own SG and masking view. If the database is not clustered, the Grid masking view is optional.

There are several advantages to this design:

- Performance can be monitored for the whole database (App1_DataRedo), or separately for data (App1_Data) and redo logs (App1_Redo).
- If storage-consistent snapshots are created as part of a restart solution, the parent SG App1_DataRedo is used. If snapshots are created as part of a recovery solution, during recovery the App1_Data alone SG can be restored, without overwriting the production redo logs (App1_Redo) with the latest database transactions.
- If SRDF is used to replicate the database, the parent App1_DataRedo SG is used to set the storage-consistent replications of the whole database.

The following example shows Command Line Interface (CLI) execution, from device creation all the way to masking view. Masking views can be created in Unisphere using Wizards. CLI is recommended only when such commands are scripted and saved.

**Masking view creation example using Command Lines Interface (CLI)**

- Devices creation:

```bash
set -x
export SYMCLI_SID=<SID>       # Storage ID
export SYMCLI_NOPROMPT=1

# Create ASM Disk Groups devices
symdev create -v -tdev -cap 40 -captype gb -N 3  # +GRID
symdev create -v -tdev -cap 200 -captype gb -N 16 # +DATA
symdev create -v -tdev -cap 50 -captype gb -N 8  # +REDO
symdev create -v -tdev -cap 150 -captype gb -N 4  # +FRA
```
**Note:** When creating new devices for FC-NVMe connectivity, add '-mobility' flag to the syntax above. If changing existing devices from FC to FC-NVMe protocol, use a syntax like the following to change the devices’ type to mobility:

```bash
symdev -devs <START:END> set -device_id mobility
```

- Storage groups creation (device IDs are based on the output from device creation step above):

```bash
# SGs
symsg create grid_sg # Stand-alone SG for Grid infrastructure
symsg create fra_sg # Stand-alone SG for archive logs
symsg create data_sg # Child SG for data and control file devices
symsg create redo_sg # Child SG for redo log devices
symsg create dataredo_sg # Parent SG for database (data+redo) devices

# Add appropriate devices to each SG
symsg -sg grid_sg addall -devs 12E:130 # modify device IDs
symsg -sg data_sg addall -devs 131:133,13C:148 # as necessary
symsg -sg redo_sg addall -devs 149:150
symsg -sg fra_sg addall -devs 151:154

# Add the child SGs to the parent
symsg -sg data redo_sg add sg data_sg,redo_sg
```

- Port group creation:

```bash
# PG
symaccess -type port -name 188_pg create # 188 is the storage SID
symaccess -type port -name 188_pg add -dirport 1D:4,1D:5,1D:6,1D:7
symaccess -type port -name 188_pg add -dirport 2D:4,2D:5,2D:6,2D:7
symaccess -type port -name 188_pg add -dirport 1D:8,1D:9,1D:10,1D:11
symaccess -type port -name 188_pg add -dirport 2D:8,2D:9,2D:10,2D:11
```

- Initiator groups creation (4-node RAC example with 4 servers: dsib0144, dsib0146, dsib0057, dsib0058).

**Note:** the devices WWN can be found by using Unisphere, or by running on each database server the command: cat /sys/class/fc_host/host*/port_name.

```bash
# IG
symaccess -type initiator -name dsib0144_ig create
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa910b2
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa910b3
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa90f86
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa90f87
symaccess -type initiator -name dsib0146_ig create
symaccess -type initiator -name dsib0146_ig add -wwn 10000090faa910aa
```
Best Practices for PowerMax and Oracle databases

Number and size of devices

PowerMax uses thin devices exclusively, which means that storage capacity is only consumed when applications write to the devices. This approach saves flash capacity because storage is only consumed with actual demand.

PowerMax devices can be sized from a few megabytes to multiple terabytes. Therefore, you might be tempted to create only a few very large devices, but consider the following:

- When Oracle ASM is used, devices (members) of an ASM disk group should be of similar capacity. If devices are sized large initially, each capacity increment to the ASM disk group will also need to be large.

- Oracle ASM best practice is to add multiple devices together to increase the ASM disk group capacity, rather than adding one device at a time. This method spreads ASM extents during rebalance to avoid hot spots. Use a device size that allows increments in which multiple devices are added to the ASM disk group together, each with the same size as the original devices.

- Another benefit of using multiple devices is that the server creates I/O queue per device path and multiple I/O queues provide for more parallelism of I/Os and prevents queueing problems. In addition, the storage system benefits from more
parallelism during local or remote replications when data transfer occurs across multiple devices.

- Until Oracle release 12.1, ASM device size was limited to 2 TB. With Oracle release 12.2 and above, ASM allows for much larger device sizes.

While there is no one size that fits all databases, for the size and number of devices, we recommend the following:

- 8 to 16 data devices and 4 to 8 redo log devices are often sufficient for high performance databases. For Oracle releases up to 12.1, this allows up to 32 TB (16 data devices x 2TB = 32 TB). For databases larger than 32 TB, more devices might be needed to satisfy the capacity requirements of the database. If using Oracle release 12.2 and above, the 2 TB is no longer a limit.

- When storage replications such as SRDF/Sync, SRDF/Metro, and PowerProtect (which uses SnapVX) are deployed, we recommend doubling the numbers above to increase replications concurrency. That means 16 to 32 data devices, and 8 to 16 redo log devices. It can provide higher replication bandwidth.

**Note:** Keep in mind that Although PowerMax devices are thin, and don’t consume storage capacity unless written to, the larger the device, the more metadata it consumes (which affects the available storage cache). Therefore, devices should be sized reasonably, based on their expected capacity needs.

### Server considerations

**Partition alignment**

Although not required on Linux, Oracle recommends creating one partition on each ASM device. By default, Oracle Linux (OL) and Red Hat Enterprise Linux (RHEL) release 7 and later create partitions with a default offset of 1 MB. However, earlier releases of OL or RHEL defaulted to a 63-block partition offset, or 63 x 512 bytes = 31.5KB.

Because PowerMax uses a 128 KB track size, a 0 offset (if no partitions are created) or a 1 MB offset are perfectly aligned.

If you are creating a partition for ASM devices on an older Linux release that defaults to a 31.5 KB offset, we highly recommend that you align the partition offset to 1 MB.

To create and align partitions, use the `parted` or `fdisk` commands. Parted is often easier to use and script.

The following steps show how to use the `fdisk` command:

1. Create a single primary partition on the device.
2. Use `x` to enter fdisk **expert** mode.
3. Use `b` to change the partition **beginning** offset.
4. Enter 2,048 for a 1 MB offset (2,048 x 512 bytes blocks).
5. Optionally, use `p` to **print** the partition table layout
6. Use `w` to **write** the partition table.
The following example shows how to use the Linux `parted` command with PowerPath devices (a similar script can be written for Device Mapper for example):

```bash
for i in {a..h}; do
  parted -s /dev/emcpower$i mklabel msdos
  parted -s /dev/emcpower$i mkpart primary 2048s 100%
  chown oracle.oracle /dev/emcpower$i
done
fdisk -lu # lists server devices and their partition offset
```

When RAC is used, other nodes are not aware of the new partitions. Rebooting, or reading and writing the partition table on all other nodes can resolve this condition. For example:

```bash
for i in {a..h}; do
  fdisk /dev/emcpower$i << EOF
  w
EOF
  chown oracle.oracle /dev/emcpower$i
done
fdisk -lu # lists server devices and their partitions offset
```

**Multipathing software**

Multipathing software is critical for any database deployment for a few reasons:

- **Load-balancing and performance**—Each storage device may need to handle thousands of IOPS. With all-flash storage, customers tend to use lower quantity but bigger capacity devices and as a result, the IOPS demand per device increases. Multipathing software allows processing read and write I/Os to the device across multiple HBA ports (initiators) and storage ports (targets). It spreads the load across the ports (‘load-balance’) to avoid single path or port I/O limits.

- **Path-failover**—The multipath software creates a pseudo device name or alias, which the application uses. Meanwhile, I/Os to the pseudo device are serviced by all the different paths available between initiators and targets. If a path stops working, the multipath software automatically directs I/Os to the remaining paths. If a path comes back, the software will resume servicing I/Os to that path automatically. Meanwhile, the application continues to use the pseudo device, regardless of which paths are active.

- **Multipath pseudo devices work well with Oracle ASM**—Oracle ASM requires a single presentation of each storage device and will not list a device as an ASM candidate if it finds multiple paths to that device. By using the multipath pseudo name (or user-friendly alias), ASM recognizes a single presentation, and the multipath software takes care of spreading the I/Os across the different paths of that pseudo device.
**FC-NVMe multipathing options**

As FC-NVMe is relatively new, there are Linux OS releases and multipathing software considerations. While some older Linux releases may claim support for FC-NVMe, this section explains why it may not be an optimal choice.

To achieve FC-NVMe high performance and low latencies, the multipath software needs to address the changes and optimizations made in the protocol. Dell EMC PowerPath 7 is the first PowerPath release that provides FC-NVMe multipath support. PowerPath 7 provides full multipathing functionality for FC-NVMe, including path failover, load-balancing, and high performance.

When considering Linux native multipathing software for FC-NVMe, there are two options:

- Linux Device Mapper (DM)
- A new FC-NVMe multipath

While DM can work with NVMe and FC-NVMe devices, and even with older Linux releases, it is not optimized for that purpose, and in all our tests performance was degraded. As a result, we don’t recommend using DM for FC-NVMe.

Therefore, for native multipathing, we recommend that you use the new FC-NVMe multipath software. Consider the following:

- In early Linux releases supporting FC-NVMe native multipath, only path failover was enabled without load-balancing. The path I/O policy was set as ‘numa’ and allowed only a single active path. All other paths were in a standby state ready for failover. This configuration does not promote high-performance and is not recommended.
- Later Linux releases added a ‘round-robin’ path I/O policy to the FC-NVMe native multipath. This policy allows for both load-balancing and failover and does support high-performance.

**Important:** For the reasons cited above, we recommend using either PowerPath (with FC-NVMe support) or Linux native FC-NVMe multipathing (not to be confused with DM) implementing the ‘round-robin’ path I/O policy.

At the time this paper was written, the minimum multipathing options and OS releases available were summarized in the following table.

**Table 7. FC-NVMe Linux OS release and multipathing options**

<table>
<thead>
<tr>
<th>Operating System</th>
<th>PowerPath</th>
<th>Native MP with ‘round-robin’ I/O policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuSE Enterprise Linux</td>
<td>Available⁴</td>
<td>Available⁵</td>
</tr>
</tbody>
</table>

---

⁴ Minimum requirements of PowerPath for Linux 7.0 and SLES 15. Our lab tests showed a minimum requirement of kernel 4.12.14-25.25.

⁵ Based on SuSE support update [link] NVMeoF multipath ‘round-robin’ policy was introduced with SLES 12-SP4 and SLES 15-SP4. Our lab tests showed a minimum requirement of kernel 4.12.14-150.
### Operating System

<table>
<thead>
<tr>
<th>Operating System</th>
<th>PowerPath</th>
<th>Native MP with 'round-robin' I/O policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat</td>
<td>Available⁶</td>
<td>Available⁷</td>
</tr>
<tr>
<td>Oracle Linux</td>
<td>Available⁸</td>
<td>Available⁹</td>
</tr>
</tbody>
</table>

**Note**: refer to Dell EMC eLab Navigator note "Dell EMC PowerMaxOS 5978.444.444 & 5978.479.479 – 32G FC-NVMe Support Matrix" for details on Dell EMC supported configurations.

---

### FC multipathing options

Three multipath choices for Oracle on Linux using FC protocol are available:

- Dell EMC PowerPath software (for bare metal, or PowerPath/VE for VMware)
- Linux Device Mapper (DM) native multipathing software
- VMware native multipathing

**Note**: While iSCSI protocol is also a very valid option to use for Oracle databases on PowerMax, it is not covered in this paper.

All three options have been available on FC for many years and provide path failover, load-balancing, and high-performance.

### Device permissions

When the server reboots, all devices receive root user permissions by default. However, Oracle ASM devices (‘disks’) require an Oracle user permission. The permission setting of the Oracle ASM devices must be part of the boot sequence, which is when Grid Infrastructure and ASM start.

Oracle ASMlib sets the device permissions automatically. When ASMlib is not used, using udev rules is the simplest way to set device permissions during the boot sequence.

Udev rules are added to a text file in the `/etc/udev/rules.d/` directory. The rules are applied in order, based on the index number preceding the file names in this directory.

The content of the text file with the rules involves identifying the Oracle devices correctly. For example, if all devices with partition 1 can have Oracle permissions, set a general rule that applies to all such devices. If not all devices can use the same permissions, then the devices must be specified individually, often based on either their UUID / WWN.

When devices are identified individually to receive Oracle user permissions, udev rules can also be used to create a user-friendly alias for each device for easy identification.

---

⁶ Minimum requirements of PowerPath for Linux 7.1 and RHEL 8.0

⁷ Minimum requirement of RHEL 8.1.

⁸ Minimum requirement of PowerPath for Linux 7.1 and Oracle Linux 7.7/UEK5u2.

⁹ Minimum requirements of Oracle Linux 7.7/UEK5u2. Our lab tests showed a minimum requirement of kernel 4.14.35-1902.5.2.
When using Linux Device Mapper, the `/etc/multipath.conf` file can also be used to create device aliases.

As there are many ways and options to identify devices and set their permissions, the following examples are not a complete list.

Keep in mind that while a single partition for each device is not required for Oracle ASM on Linux, Oracle recommends it, and it provides an easy way to assign Oracle user permissions to all the devices with partition 1.

**PowerPath example (both FC and FC-NVMe)**

When using PowerPath, we often use the `/dev/emcpower` notation without user friendly aliases, because PowerPath devices are persistent across reboot and do not change. Therefore, the udev rule can be generic to all the pseudo devices’ partition 1 as shown in the following example:

```
# vi /etc/udev/rules.d/85-oracle.rules
ACTION="add|change", KERNEL="emcpower*1", OWNER="oracle", GROUP="oracle", MODE="0660"
```

The PowerPath pseudo device naming convention (and therefore the device permission) is the same, regardless of whether you are using FC or FC-NVMe.

**Linux Device Mapper example (FC only)**

When you use device mapper, there are multiple ways for you to set pseudo device names or aliases. For example, you can set aliases in the `/etc/multipath.conf` file, or directly in the udev rules file. Based on how you set the aliases, you can apply the udev rule to set device permissions.

The following example assumes that user-friendly aliases are set in the `/etc/multipath.conf` file (see Linux Device Mapper example section for instructions). In this case, the udev rule can be generic to all the aliases with partition 1 (for example, if devices aliases are created such as: “ora_prod_data1p1”, or “ora_prod_redo3p1”, etc):

```
# vi /etc/udev/rules.d/12-dm-permissions.rules
ENV{DM_NAME}="ora_prod_*p1", OWNER="oracle", GROUP="oracle", MODE="660"
```

**Linux native FC-NVMe multipathing example**

When you use FC-NVMe with its native multipathing, devices with partition 1 appear with a multipath pseudo name such as `/dev/nvmeXXp1`. If you do not need aliases, then you can apply a generic udev rule as shown in the following example:

```
# vi /etc/udev/rules.d/12-dm-permissions.rules
KERNEL="nvme*p1", ENV{DEVTYPE}="partition", OWNER="oracle", GROUP="oracle", MODE="660"
```

**VMware example (FC only)**

Since VMware multipath presents devices to the VM as `/dev/sd` block devices (the multipathing is operating at the ESXi level), and block device assignment may change when devices are added or removed, you can specify device permissions based on the
UUID for each device separately (see VMware native multipathing example section for instructions).

**Consistent device names across RAC nodes**

When Oracle RAC is used, the same storage devices are shared across the cluster nodes. ASM places its own labels on the devices in the ASM disk group. Therefore, matching server device names across RAC nodes is not necessary to ASM. However, it often makes storage management operations easier for the user. This section provides instructions regarding how to match server device names.

**PowerPath example**

To match PowerPath pseudo device names between cluster nodes, follow these steps:

1. Use the `emcpadm exportMappings` command on the first server (cluster node) to create an XML file with the PowerPath configuration:

   ```bash
   # emcpadm exportMappings -f /tmp/emcp_mappings.xml
   ```

2. Copy the file to the other nodes.

3. On the other nodes, import the mapping:

   ```bash
   # emcpadm importMappings -f /tmp/emcp_mappings.xml
   ```

**Note:** The PowerPath database is kept in the `/etc/emcp_devicesDB.idx` and `/etc/emcp_devicesDB.dat` files. These files can be copied from one of the servers to the others, followed by a reboot. We recommend the `emcpadm export/import` method to match PowerPath device names across servers, where the file copy is a shortcut that overwrites existing PowerPath mapping on the other servers.

**Linux Device Mapper example**

Device Mapper (DM) can use UUIDs to identify devices persistently across RAC nodes. While using the UUID is sufficient, you may prefer more convenient aliases (for example, `/dev/mapper/ora_data1`, `/dev/mapper/ora_data2`, and so on).

The following example shows how to set the `/etc/multipath.conf` DM configuration file with aliases. To find the device UUID's, use the `scsi_id -g /dev/sdXX` Linux command. If the command is not already installed, add it by installing the `sg3_utils` Linux package.

```bash
# /usr/lib/udev/scsi_id -g /dev/sdb
36000097000019870067533030314633

# vi /etc/multipath.conf
...multipaths {
    multipath {
        wwid 36000097000019870067533030314633
        alias ora_data1
    }
}
To match aliases across RAC nodes, copy the `/etc/multipath.conf` configuration file to the other nodes and reboot them. If you prefer to avoid a reboot, follow these steps:

1. Set up all multipath devices on one server and then stop multipathing on the other servers:

   ```
   # service multipathd stop
   # service multipath -F
   ```

2. Copy the multipath configuration file from the first server to all other servers. If user friendly names must be consistent, copy the `/etc/multipath/bindings` file from the first server to all others. If aliases must be consistent (aliases are set up in `multipath.conf`), copy the `/etc/multipath.conf` file from the first server to all others.

3. Restart multipath on the other servers:

   ```
   # service multipathd start
   ```

**VMware native multipathing example**

When running Oracle in VMware virtual machines (VMs), although PowerPath/VE or VMware native multipathing is in effect, the VM devices appear as `/dev/sdXX`. You can use Udev rules to create device aliases that match across RAC nodes, and to assign Oracle user permission to the devices.

To allow identifying devices’ UUID on a VM, enable `disk.EnableUUID` for the VM. Follow these steps:

1. In vSphere, power-off the VM.
2. Right click on the VM and choose **Edit Settings**.
3. Choose the **VM options** tab.
4. Expand the **Advanced** option and click on **Edit Configuration**.
5. Choose **Add Configuration Params** to add the parameter as shown in the figure below, and add a parameter: `disk.EnableUUID`. Set the parameter to TRUE.
Best Practices for PowerMax and Oracle databases

Configuration Parameters

![Configuration Parameters](image)

Add New Configuration Param

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>disk.EnableUUID</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

Figure 27. Enable UUID device identification

6. Restart the VM

7. Identify the devices’ UUID using the `scsi_id` command as shown below.

Note: In RHEL 7 and above, the `scsi_id` command is located in `/lib/udev/scsi_id`. In previous releases, it was located in `/sbin/scsi_id`.

```
# /lib/udev/scsi_id -g -u -d /dev/sdb
36000c29127c3ae0670242b058e863393
```

8. Build a udev rule to set Oracle user permissions to all the devices with partition 1, based on their UUID. In the example shown below, notice that an alias is given to the device so that the ASM disk string can be set using it.

```
# /lib/udev/scsi_id -g -u -d /dev/sdb
36000c29127c3ae0670242b058e863393
```

```
# cd /etc/udev/rules.d/
# vi 99-oracle-asmdvices.rules
KERNEL="sd*1", SUBSYSTEM="block", PROGRAM="/usr/lib/udev/scsi_id -g -u -d /dev/$parent", RESULT="36000c29127c3ae0670242b058e863393", SYMLINK="ora-data1", OWNER="oracle", GROUP="oracle", MODE="0660"
```

Block multi-queue (MQ)

In recent Linux kernels, the block I/O device driver in the server operating system was rewritten to address NVMe devices and the need for ultra-low latencies and higher IOPS. The new design is referred to as block multi-queue, or MQ for short. When SCSI block devices are presented to the server (/dev/sd) MQ is not enabled by default, because of the chance that these devices are supported by legacy spinning hard disk drives (HDDs). If the storage is, in fact, all-flash (such as with PowerMax), you can enable MQ manually for SCSI devices as described in Appendix I. Blk-mq and scsi-mq.
When devices are presented to the server as NVMe devices (/dev/nvme)—such as when using PowerMax FC-NVMe—they will automatically be enabled for MQ. No additional changes are needed to enable the functionality.

Likely because of the high adoption of flash media, in RHEL 8.0, MQ has been made a default even for SCSI (/dev/sd) devices.

In previous testing, we had found that using MQ provided a significant performance advantage. In recent testing, however, when using newer CPUs, 32Gb SAN, and 4-node Oracle 19c RAC, the servers were no longer a ‘bottleneck’ and MQ did not seem to provide a noticeable advantage.

Dell EMC recommends that in an FC environment in which the server is heavily utilized, you should enable MQ and test to see if it provides significant benefits for your Oracle FC deployment. Remember that FC-NVMe deployments will have MQ enabled automatically.

### Oracle ASM best practices

#### ASM disk groups

For mission-critical Oracle databases, we recommend separating the following database components to different ASM disk groups and matching SGs:

- **+DATA**—ASM disk group used for the database data, such as data files, control files, undo tablespace, system tablespace, and so on. By separating data from logs (separate +DATA and +REDO ASM disk groups), storage replications can be used for database backup and recovery operations as well as more granular performance monitoring.

- **+REDO**—ASM disk group used for the database redo logs.

- **+FRA**—ASM disk group used for the database archive logs and flashback logs (if used). Note that flashback logs might consume much larger capacity than the archive logs. Also, unlike archive logs, flashback logs must be consistent with the data files if protected with storage replications. For these reasons, consider separating archive logs and flashback logs to distinct ASM disk groups and SGs.

- **+GRID**—ASM disk group used for Grid Infrastructure (GI), which is a required component when using Oracle ASM or RAC (cluster). Even in single instance (non-clustered) deployments, we recommend that you create this ASM disk group and SG so that database data is not mixed with GI management components.

All ASM disk groups should use external redundancy (no ASM mirroring) except for +GRID, which can use normal redundancy (two mirrors). +GRID should not contain any user data and therefore remains small. When set to normal redundancy, Oracle creates three quorum files rather than a single one (such as the case when external redundancy is used). Having three quorum files helps to avoid delays while nodes try to register with the quorum during high database activity.

**Note:** For in-depth information about considerations for ASM disk groups and matching SGs that can leverage fast storage replicas that are valid for database backup and recovery, see Oracle database backup, recovery, and replications best practices with VMAX All Flash storage.

#### ASM striping

By default, ASM uses an Allocation Unit (AU) size of 1 MB (4 MB starting with release 12.2) and stripes the data across the whole disk group by using the AU as its stripe-depth.
This default ASM striping method is called “Coarse Striping” and is optimal for OLTP-type applications. The DBA might decide to increase the size of the AU from its default, though there is no clear benefit for doing that.

ASM has an alternative striping method called “Fine-Grain Striping”. With Fine-Grain Striping, ASM selects eight devices in the disk group (if available), allocates an AU on each, and further stripes (divides) each AU into 128 KB chunks. It then allocates data in a round-robin fashion across the eight devices by filling up the 128 KB chunks. When all eight AUs are full, it repeats the process by selecting another set of eight devices.

Fine-Grain striping is the PowerMax-recommended striping method for Oracle objects with primarily sequential writes. Because sequential writes tend to be large I/Os, by breaking them in to 128 KB stripes, latencies improve and PowerMax can handle them more efficiently (as its track size is also 128 KB).

For this reason, we recommend that you use ASM Fine-Grain striping for the redo logs. This method is especially useful for In-Memory databases (where transactions are fast and redo write load can be heavy) or for batch data loads (as again, the redo logs write load is heavy).

In Data Warehouses where Oracle Temp files can be I/O intensive, Temp files can also benefit from Fine-Grain Striping.

The type of striping for each Oracle ASM file type is kept in an ASM template. Each ASM disk group has its own set of templates. Modifying a template (for example, changing the redo logs template to Fine-Grain) only applies to the ASM disk group in which the template was changed.

In addition, existing ASM allocations are not affected by template changes, only new extents. Therefore, if you change the redo logs template in the +REDO ASM disk group, you must create new redo log files afterwards, make sure the database is using them, then remove the old ones. Creating and removing redo logs does not take long and can be performed while the database is running.

To inspect the ASM templates, run the following query from the ASM instance:

```sql
SQL> select DG.name Disk_Group, TMP.name Template, TMP.stripe from v$asm_diskgroup DG, v$asm_template TMP where DG.group_number=TMP.group_number order by DG.name;
```

To change the database redo logs template in the +REDO ASM disk group, run the following command:

```sql
SQL> ALTER DISKGROUP REDO ALTER TEMPLATE onlinelog ATTRIBUTES (FINE);
```

To change the temp files template in the +TEMP ASM disk group, run the following command:

```sql
SQL> ALTER DISKGROUP TEMP ALTER TEMPLATE tempfile ATTRIBUTES (FINE);
```
Typically, an OLTP workload reads or updates a single record at a time, and therefore, Oracle reads or modifies single database blocks (typically 8 KB in size). This is also the I/O size of read and write operations to the data files that comprise an OLTP workload. However, when a query requires a set of data to be fetched, such as for a report, a list, merge of data from multiple sources, or predicate evaluation, Oracle will perform a multi-block read in a single operation.

When Oracle performs a multi-block read it issues large I/Os of sizes up to 1 MB. The size of the large reads can be controlled by a database parameter: `db_file_multiblock_read_count` (MBRC). This database parameter determines the Oracle maximum size of a multi-block read I/O operation. The maximum I/O size is calculated as a multiplication of the MBRC and the Oracle block size. With an 8 KB database block size, when MBRC is set to 16, the result is a maximum of 128 KB database read I/O size (16 x 8KB = 128KB). When MBRC is set to 128, the result is a maximum of 1 MB database read I/O size (128 x 8 KB = 1,024 KB).

Our tests showed very small differences in bandwidth between the two options. However, the I/O size has a big effect on IOPS and latencies. A 128KB I/O size is eight times smaller than a 1MB I/O size, and will therefore generate more IOPS to achieve the same bandwidth. However, because each I/O is smaller, the response time is much lower than a 1MB I/O.

Our recommendation is to use the 128KB I/O size, as most environments run a mix of OLTP and DSS workloads (single-block, and multi-block operations). In a mixed environment, allowing lower latency for the multi-block operations means that single-block I/Os are not waiting long to be serviced. Even if the system is a dedicated data warehouse (mainly multi-block reads), since the bandwidth is very similar, the lower latency is still an advantage, as long as total IOPS are not an issue to the overall system utilization.

Oracle 11gR2 introduced the ability to change the redo log block size from its default 512 bytes to 4 KB. One reason was because certain drives used 4 KB as their native block size (for example, SSD drives). Another reason was to reduce the metadata overhead that was associated with high density Hard Disk Drives (HDD) by increasing the block size from its legacy 512 bytes to 4 KB.

When using PowerMax storage, there are two key reasons why the redo log block size should not be changed from the default 512 bytes per sector:

- The database never writes directly to the flash drives. Instead, all writes to the PowerMax storage system go to the PowerMax cache, where they can be aggregated to provide optimized writes to the flash media at a later time. Therefore, such a change does not benefit the drives directly.
- There is an increase in redo wastage that is often significant. When the Oracle database commits frequently, it requires an immediate write of the log buffer. With 4 KB blocks and frequent commits, the redo log buffer may be mostly empty, creating unnecessary write overhead and redo wastage.

A part of the Linux block I/O driver is the I/O scheduler (also known as I/O elevator). The I/O scheduler can take I/Os in the submission queue and reorder them, for example, to prioritize reads over writes, coalesce smaller I/Os to larger ones, and so on.
According to Red Hat, the preferred I/O scheduler for best database latency is Deadline. Starting with RHEL 7, Deadline became the default I/O scheduler. However, in earlier releases it was CFQ.

Based on our tests with VMAX All Flash and later PowerMax storage systems, Deadline provided good performance. CFQ performance was not as good. Therefore, we recommend that you use Deadline when choosing a Linux I/O scheduler for Oracle databases on PowerMax storage systems.

An alternative to using the Linux I/O scheduler is to use the new enhancements to the Linux block I/O driver with blk-mq (MQ). For more information, see Appendix I. Blk-mq and scsi-mq. Note that when using FC-NVMe, MQ is enabled by default.

### Appendixes

#### Appendix I. Blk-mq and scsi-mq

**What is blk-mq**

The Linux block device I/O layer was designed to optimize Hard Disk Drives (HDD) performance. It was based on a single queue (SQ) for I/O submission for each block device; a single locking mechanism shared by all CPU cores whenever I/Os were submitted, removed, or reordered in the submission queues; and inefficient hardware interrupt handling. For more information, see [Linux Block IO: Introducing Multi-queue SSD Access on Multi-core Systems](#) and [Improving Block-level Efficiency with scsi-mq](#).

With the increased use of Non-Volatile Memory (NVM) as primary storage (flash storage devices) the I/O bottleneck shifted from the storage media to the server I/O layer. This shift opened the door to a new design: block multi-queue (MQ), also known as blk-mq.

---

**Note:** The implementation of blk-mq with SCSI-type block devices (/dev/sd) is called scsi-mq.

blk-mq introduces a two-layer design in which each block device has multiple software I/O submission queues (one per CPU core) that eventually fan into a single queue for the device driver. Queue handling is based on FIFO ordering that the core submitting the I/Os handles. It no longer requires the interrupts or shared lock mechanism.

The current design omits I/O reordering (scheduler) because NVM media performance is not affected by the I/O pattern being random or sequential. However, I/O scheduling can be introduced by using kernel I/O scheduler such as `mq-deadline` (RHEL 8).

We tested blk-mq with VMAX All Flash and PowerMax storage systems. When the server was the bottleneck (older CPUs, high server utilization), MQ provided excellent performance and server efficiency benefits. With newer servers, CPU, and SAN (32Gb), we didn’t notice such benefits.

In addition, when using FC-NVMe protocol, MQ is already enabled by default for /dev/nvme devices. Likely due to do the high adoption of flash storage, in RHEL 8 and above, MQ is enabled by default even when using FC protocol (for the /dev/sd devices). For FC environments where the server may be a bottleneck, we recommend trying blk-mq if it isn’t already enabled by default.
Note the following about the use of blk-mq:

- PowerPath supports blk-mq on all Linux 4.12.x kernels (SLES 12 SP3-SP5, SLES 15, SLES 15 SP1, Oracle Linux 7 UEK5, RHEL 8, RHEL 8.1).
- In our tests, Linux native multipathing worked well with blk-mq, but only when the Linux kernel was version 4.x. For example, with OL/UEK 7.4 and above or RHEL 8.0 and above.
- It is easy to enable/disable blk-mq. Therefore, we recommend that VMAX All Flash and PowerMax customers using FC protocol on Linux with kernel 4.x test to see if blk-mq provides performance and server-efficiency benefits to their high-performance databases.

**Enabling or disabling blk-mq**

Linux kernels enable blk-mq by default for devices presented with NVMe protocol (devices appearing as `/dev/nvme`). For devices presented via FC protocol (devices appearing as `/dev/sd`), blk-mq may not be enabled.

To determine if FC devices have MQ enabled run the following commands, where the first command determines if MQ is enabled in general, and the second command is only relevant if using device mapper multipath.

```bash
# cat /sys/module/scsi_mod/parameters/use_blk_mq
N
# cat /sys/module/dm_mod/parameters/use_blk_mq
N
```

To enable blk-mq for FC devices, update the `/boot/grub2/grub.cfg`. This can be done by editing the `GRUB_CMDLINE_LINUX` parameters as shown below. As before, the `dm_mod.use_blk_mq` parameter is only relevant if using device mapper multipath.

```bash
# vi /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto resume=UUID=7b5d3708-53b9-482e-80f6-01d4086f30b2 rhgb quiet scsi_mod.use_blk_mq=1 dm_mod.use_blk_mq=y"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
```

The `scsi_mod.use_blk_mq=1` parameter enables blk-mq for SCSI-type block devices at the kernel level (where 0 disables it).

The `dm_mod.use_blk_mq=y` parameter enables blk-mq for device-mapper (DM) Linux native multipathing (where n disables it).

Recreate the `grub.cfg` file and remember to reboot the server for the changes to take effect.

```bash
# grub2-mkconfig -o /boot/grub2/grub.cfg
```
Deleted data and storage reclamation

When database clients delete data, ASM is aware of the additional free space; however, the storage system is not. The storage extents that are based on the deleted data remain allocated. You can reclaim the storage extents by:

- Removing and deleting the storage device.
- Executing a `symsg free` or `symdev free` command (after the devices are made not-ready to the server), which erases the content of the whole device, but not the actual devices.
- Using the Linux `blkdiscard` command on the whole device, or on a partition, to efficiently free up capacity by using SCSI unmap commands. This again will erase the content of the device or partition, but does not require that the device be removed from the server first.
- Running the `symsg reclaim` or `symdev reclaim` commands. These commands free (reclaim) any storage extents (128 KB) that are zeroed. However, the deleted ASM extents do not contain zeros.

Oracle ASM Filter Driver online storage reclamation provides a way to reclaim the storage space of deleted data while the ASM disk group remains online and active.

ASM Filter Driver

ASM Filter Driver (AFD) is a kernel module that resides in the I/O path of the Oracle ASM disks. It is available as of Grid Infrastructure release 12.

AFD has many advantages over ASM. These advantages include the ability to:

- **Protect ASM devices from writes that are not originated by Oracle processes**—For example, there is no harm to the ASM disks when you run either of the following commands:
  - `dd if=/dev/zero of=<my_ASM_device>`
  - `blkdiscard <my_ASM_device>`

  The protection is disabled only when the AFD labels are removed from the devices.

- **Label ASM devices for ease of management**—When creating an ASM disk group, AFD can provide automatic labels that are based on the disk group name and a disk number. Users can also provide their own labels.

- **Reclaim ASM disk group storage online**—The ASM disk group remains online, while deleted space is reclaimed inside the ASM disk group and the storage system.

When to use AFD online storage reclamation

ASM is efficient in reusing deleted space. For example, if data files are deleted and new files of similar capacity are created, there is no need to reclaim the deleted storage space. ASM simply reuses it.

If a large data set is deleted in ASM, such as a legacy database or copies of the database that consumed large capacity, it is an opportunity to reclaim this storage in the array and make it available to other applications.
During ASM storage reclamation, ASM first performs a manual rebalance, which defrags (compacts) the ASM disk group by moving ASM extents to gaps created by the deleted data. When the ASM disk group is compacted, its High Water Mark (HWM) is updated based on its new allocated capacity. Next, ASM sends SCSI unmapped commands to the storage system to reclaim the space above the new HWM. The reclamation is efficient and fast.

How to use AFD online storage reclamation

To use AFD storage reclamation, the ASM disk group must have a 12.1 (or later) compatibility setting.

To enable AFD storage reclamation, give the ASM disk group the THIN attribute by using the following command:

```
ALTER DISKGROUP <NAME> SET ATTRIBUTE 'THIN_PROVISIONED'='TRUE';
```

After this you can reclaim the ASM disk group storage any number of times, using the following command:

```
ALTER DISKGROUP <NAME> REBALANCE WAIT;
```

The WAIT option enables the user to receive the prompt only when the operation completes.

AFD online storage reclamation example

The following example illustrates the value of AFD online storage reclamation.

We start with an empty ASM disk group to illustrate the capacity reduction. (In a real world deployment, the ASM disk group will not be empty.)

1. Give the ASM disk group the THIN attribute:

```
ALTER DISKGROUP TEST SET ATTRIBUTE 'THIN_PROVISIONED'='TRUE';
```

2. Add a 300 GB tablespace:

```
CREATE BIGFILE TABLESPACE TP1 DATAFILE '+TEST' size 300G ONLINE;
```

While a new data file is created, Oracle initializes the space and it is allocated in the storage system.

3. Delete the tablespace:

```
DROP TABLESPACE TP1 INCLUDING CONTENTS AND DATAFILES;
```

At the storage level, no space is released yet, as shown in the following figure:
Deployment Best Practices for Oracle Database with Dell EMC PowerMax

White Paper

Appendixes

SYMMETRIX THIN DEVICES

<table>
<thead>
<tr>
<th>Sym</th>
<th>Pool Name</th>
<th>Flgs</th>
<th>Total</th>
<th>Allocated</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GBs (%)</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>----</td>
<td>------</td>
<td>----------</td>
<td>-----</td>
</tr>
<tr>
<td>00151</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>75.0</td>
<td>50</td>
</tr>
<tr>
<td>00152</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>75.1</td>
<td>50</td>
</tr>
<tr>
<td>00153</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>75.1</td>
<td>50</td>
</tr>
<tr>
<td>00154</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>75.0</td>
<td>50</td>
</tr>
</tbody>
</table>

Total

<table>
<thead>
<tr>
<th>GBs</th>
<th>600.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocated</td>
<td>300.2</td>
</tr>
<tr>
<td>Ratio</td>
<td>50</td>
</tr>
</tbody>
</table>

Figure 28. PowerMax storage group before AFD reclamation

4. Rebalance the ASM disk group:

```
ALTER DISKGROUP TEST REBALANCE WAIT;
```

The following figure shows that other than the small ASM metadata (because we started with an empty ASM disk group), no space is consumed.

SYMMETRIX THIN DEVICES

<table>
<thead>
<tr>
<th>Sym</th>
<th>Pool Name</th>
<th>Flgs</th>
<th>Total</th>
<th>Allocated</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GBs (%)</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>----</td>
<td>------</td>
<td>----------</td>
<td>-----</td>
</tr>
<tr>
<td>00151</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>0.0</td>
<td>0</td>
</tr>
<tr>
<td>00152</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>0.1</td>
<td>0</td>
</tr>
<tr>
<td>00153</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>0.1</td>
<td>0</td>
</tr>
<tr>
<td>00154</td>
<td>-</td>
<td>F..B</td>
<td>150.0</td>
<td>0.1</td>
<td>0</td>
</tr>
</tbody>
</table>

Total

<table>
<thead>
<tr>
<th>GBs</th>
<th>600.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocated</td>
<td>120.0</td>
</tr>
<tr>
<td>Ratio</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 29. PowerMax storage group after AFD reclamation
The following section provides details about PowerMax compression and deduplication of server I/Os.

When new writes arrive from the database servers, they are registered at the PowerMax cache and immediately acknowledged to the server, achieving low write latencies, as shown in the following figure.

Figure 30.   Deduplication step 1: Server writes the register in the PowerMax cache

Because PowerMax cache is persistent, it does not have to write the data to the NVMe flash media immediately. Oracle can continue to write to the same or adjacent database blocks multiple times.

When PowerMax does write the data to the NVMe flash storage, if compression is enabled for the storage group, then the 128 KB cache slot with the new data is sent to the hardware compression module where the data is compressed and Hash IDs are generated.

The cache slot is tested for uniqueness and if indeed it is unique, the compressed version of the data is stored in the appropriate compression pool, and the thin device pointers are updated to point to the data’s new location, as shown in the following figure.
If the compressed data is not unique—that is, if a previous identical copy of that same data is already stored compressed in PowerMax—the data is not stored again. Instead, just the thin devices’ pointers are updated to point to the existing compressed version of the data, as shown in the following figure.

This example shows the power of deduplication: multiple copies of identical data are stored only once in the PowerMax storage system.
If read I/O latencies are higher than expected, it is best practice to use the Linux `iostat` command to investigate. Run the following command:

```
iostat -xtzm <interval> <iterations> [optional: <list of specific devices separated by space>]
```

Capture the output to a file. After a few iterations, stop the command and inspect the file. Ignore the first interval and focus on interval 2 and higher.

Because the file can be large, either include the specific devices to monitor, or find the pseudo name of one of the Oracle data files and focus on that single device\(^{10}\), as shown in the following:

```
Device:     rrqm/s  wrqm/s  r/s  w/s  rsec/s  wsec/s avgrq-sz avgq-sz  await  svctm  %util
  dm-395    0.00    0.00  2145.33 937.33  16.76   7.51    16.13   2.99    0.97   0.28  87.77
```

Note how many I/Os are queued to the device (`avgq-sz`), the amount of time I/Os take to be serviced, including queue time (`await`), and the amount of time I/Os take to be serviced when they leave the queue (`svctm`). If the `await` time is long but `svctm` time is short, the server is likely experiencing a queuing issue and requires more LUNs (more I/O queues), and perhaps more paths to the storage. If the `svctm` time is high, it might indicate a SAN or storage performance bottleneck.

The following table summarizes some of the `iostat` metrics and provides advice about how to use them.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>The device name as listed in the <code>/dev</code> directory</td>
<td>When multipathing is used, each device has a pseudo name (such as dm-xxx, or emcpowerxxx) and each path has a device name (such as /dev/sdxxx). Use the pseudo name to inspect the aggregated metrics across all paths.</td>
</tr>
<tr>
<td>r/s, w/s</td>
<td>The number of read or write requests that are issued to the device per second.</td>
<td>r/s + w/s provides the server IOPS requests for the device. The ratio between these metrics provides read to write ratio.</td>
</tr>
<tr>
<td>rMB/s, wMB/s</td>
<td>The number of MB read or written from or to the device per second (512 bytes per sector)</td>
<td>Review the bandwidth performance of the device. You can determine the average read I/O size by dividing the rMB/s by r/s. Similarly, you can determine the average write size by dividing wMB/s read by w/s.</td>
</tr>
<tr>
<td>avgrq-sz</td>
<td>The average size (in sectors) of the requests that are issued to the device</td>
<td>The queue size is not very important in most performance issues. Focus should be on the parameter: ‘avgq-sz’ below.</td>
</tr>
</tbody>
</table>

\(^{10}\) With Oracle ASM striping, I/Os should be distributed evenly across all data devices.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>avgqu-sz</td>
<td>The average queue length of the requests that were issued to the device</td>
<td>The number of requests queued to the device. If the queues are large they will cause increase in latency. If the devices are in a SG with a low service level consider improving the service level. Otherwise, if the storage system is not over utilized, consider adding more devices or paths to allow for better I/O distribution at the server level.</td>
</tr>
<tr>
<td>await</td>
<td>The average time (in milliseconds) for I/O requests issued to the device to be served. This includes the time spent by the requests in queue and the time spent servicing them</td>
<td>If the <code>await</code> time, which includes the queuing time, is much larger than the <code>svctm</code> time, it might indicate a server queuing issue. See <code>avgqu-sz</code> metric above.</td>
</tr>
<tr>
<td>svctm</td>
<td>The average service time (in milliseconds) for I/O requests that were issued to the device</td>
<td>For active devices, the <code>await</code> time should be within the expected service level time (for example &lt;=1 ms for flash storage, ~6 ms for 15k rpm drives, and so on).</td>
</tr>
</tbody>
</table>

**Appendix V. Oracle AWR I/O related information**

Collect AWR reports over peak workload periods to identify any potential bottlenecks. AWR averages all metrics over the duration of the report; therefore a 24-hour report is generally not useful. A useful AWR report is produced for a short period of time, for example, 15 minutes, 30 minutes, or one hour, when the workload is stable and heavy.

When using Oracle RAC, AWR reports can be produced for each instance separately or for the cluster as a whole. The Instance AWR metrics only represent the workload on that specific database server. The RAC AWR metrics represent the workload from the entire cluster. In the following examples, we show both types. Note that with each Oracle release some changes were made to the report.

**AWR Load Profile**

The Load Profile area in an Instance AWR report includes “Physical reads (blocks)”, “Physical writes (blocks)”, and “Logical reads” metrics, as shown in the following figure. The units for these metrics are database blocks. Block units cannot be directly translated to I/O metrics. However, using these numbers can provide an indication of the database I/O profile, such as reads versus writes ratio and how many of the reads are satisfied from the buffer cache (logical reads) versus actual read I/Os (physical reads).

Typically, expect a high percent of an OLTP read workload to be satisfied by database cache (logical reads). In benchmarks, we often limit the database cache size to generate more I/Os and the numbers are much closer to each other.

Note that in an AWR report for an Oracle 12c database, the load profile also provides actual I/O metrics (as shown by the second group of highlighted metrics).
Figure 33.  Load Profile section in a single Instance AWR Report

The cluster AWR report provides similar information, as shown in the following figure.

<table>
<thead>
<tr>
<th>System Statistics - Per Second</th>
<th>Per Second</th>
<th>Per Transaction</th>
<th>Per Exec</th>
<th>Per Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB Time(s):</td>
<td>80.9</td>
<td>0.2</td>
<td>0.04</td>
<td>87.89</td>
</tr>
<tr>
<td>DB CPU(s):</td>
<td>6.7</td>
<td>0.0</td>
<td>0.00</td>
<td>7.24</td>
</tr>
<tr>
<td>Background CPU(s):</td>
<td>1.3</td>
<td>0.0</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Redo size (bytes):</td>
<td>10,801,337.4</td>
<td>20,760.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logical read (blocks):</td>
<td>145,906.9</td>
<td>280.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block changes:</td>
<td>68,121.0</td>
<td>130.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical read (blocks):</td>
<td>132,661.0</td>
<td>255.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical write (blocks):</td>
<td>36,090.4</td>
<td>69.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read IO requests:</td>
<td>132,660.8</td>
<td>255.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write IO requests:</td>
<td>35,774.8</td>
<td>68.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read IO (MB):</td>
<td>1,036.4</td>
<td>2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write IO (MB):</td>
<td>262.0</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IM scan rows:</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session Logical Read IM:</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Cache blocks received:</td>
<td>6.7</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Cache blocks served:</td>
<td>5.5</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User calls:</td>
<td>0.9</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parses (SQL):</td>
<td>1.2</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hard parses (SQL):</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Work Area (MB):</td>
<td>0.1</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logons:</td>
<td>0.2</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Executes (SQL):</td>
<td>2,081.3</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rollbacks:</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transactions:</td>
<td>520.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 34.  Cluster AWR report: Load Profile

AWR Top Foreground Events

Ideally, the database should wait most of the time for CPU and I/O, which is an indication that the system is working at its physical limit. Ensure that the db file sequential read field of the AWR report (which actually means random-read) has an average wait time appropriate to the storage type and application needs. For example, 596usec, or 0.6ms I/O latency, as shown in the following figure.
### Top Timed Events

<table>
<thead>
<tr>
<th>Wait</th>
<th>Event</th>
<th>Wait Time</th>
<th>Summary Avg Wait Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>User I/O</td>
<td>db file sequential read</td>
<td>893,729,747</td>
<td>0.00</td>
</tr>
<tr>
<td>System I/O</td>
<td>log file parallel write</td>
<td>3,016,244</td>
<td>0.00</td>
</tr>
<tr>
<td>Configuration</td>
<td>free buffer waits</td>
<td>194,424</td>
<td>0.00</td>
</tr>
<tr>
<td>System I/O</td>
<td>db file parallel write</td>
<td>9,081,166</td>
<td>0.00</td>
</tr>
<tr>
<td>User I/O</td>
<td>read by other session</td>
<td>1,135,914</td>
<td>0.00</td>
</tr>
<tr>
<td>Other</td>
<td>LGWR any worker group</td>
<td>436,390</td>
<td>0.00</td>
</tr>
<tr>
<td>Cluster</td>
<td>gc gc grant 2-way</td>
<td>1,360,847</td>
<td>0.00</td>
</tr>
<tr>
<td>Other</td>
<td>RMA: IPC0 completion sync</td>
<td>7,542</td>
<td>0.00</td>
</tr>
<tr>
<td>Other</td>
<td>LGWR worker group ordering</td>
<td>98,744</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Figure 35. Cluster AWR report: Top Timed Events**

The log file parallel write metric indicates how fast the Oracle Log Writer is able to write the redo logs to storage. Oracle can generate redo log writes in different I/O sizes, based on the load (up to 1 MB in size). The larger the I/O, the longer it takes to complete. However, we see in this example that the log file parallel write metric shows 1.24 ms write latency, which is an excellent number for large I/Os.

**AWR data file read and write I/O metrics**

To find IOPS and MB/sec I/O-related metrics in the AWR report, look for the physical read total IO requests, physical write total IO requests, physical read total bytes, and physical write total bytes metrics. These metrics provide read IOPS, write IOPS, read bandwidth, and write bandwidth.

The following figure shows that the cluster executed 385,808 read I/Os per second, 108,197 write I/Os per second, 2.96GB/sec read bandwidth (3,179,107,539 / 1024 / 1024 to 1024 to change from bytes/sec to GB/sec), and 0.84GB/sec write bandwidth (900,994,499 / 1024 / 1024 / 1024). Of course, the bandwidth is of more interest during DSS workloads.
Appendixes

System Statistics (Global)

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>Average</th>
<th>Std Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>physical read IO requests</td>
<td>695,479,821</td>
<td>385,769.18</td>
<td>253.86</td>
<td>96,442.30</td>
<td>64,484.59</td>
<td>1.16</td>
</tr>
<tr>
<td>physical read bytes</td>
<td>5,697,533,953,616</td>
<td>3,160,311,685.17</td>
<td>2,079,679.61</td>
<td>790,077,921.29</td>
<td>528,269,759.32</td>
<td>9.529.13</td>
</tr>
<tr>
<td>physical read total IO requests</td>
<td>695,550,651</td>
<td>385,808.47</td>
<td>253.89</td>
<td>96,452.12</td>
<td>64,483.72</td>
<td>12.72</td>
</tr>
<tr>
<td>physical read total bytes</td>
<td>5,731,419,739,648</td>
<td>3,179,107,539.01</td>
<td>2,092,048.40</td>
<td>794,776,884.75</td>
<td>527,375,543.67</td>
<td>6.402,809.29</td>
</tr>
<tr>
<td>physical read total multi block requests</td>
<td>32,948</td>
<td>18.28</td>
<td>0.01</td>
<td>4.57</td>
<td>2.64</td>
<td>0.88</td>
</tr>
<tr>
<td>physical reads</td>
<td>695,499,748</td>
<td>385,780.24</td>
<td>253.87</td>
<td>96,445.06</td>
<td>64,485.05</td>
<td>1.16</td>
</tr>
<tr>
<td>physical reads cache</td>
<td>695,480,034</td>
<td>385,769.30</td>
<td>253.86</td>
<td>96,442.32</td>
<td>64,484.63</td>
<td>1.16</td>
</tr>
<tr>
<td>physical reads cache prefetch</td>
<td>1,632</td>
<td>0.91</td>
<td>0.00</td>
<td>0.23</td>
<td>0.17</td>
<td>0.00</td>
</tr>
<tr>
<td>physical reads direct</td>
<td>19,713</td>
<td>10.93</td>
<td>0.01</td>
<td>10.93</td>
<td>10.93</td>
<td>10.93</td>
</tr>
<tr>
<td>physical reads direct (lob)</td>
<td>17</td>
<td>0.01</td>
<td>0.00</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>physical reads direct temporary tablespace</td>
<td>19,696</td>
<td>10.93</td>
<td>0.01</td>
<td>10.93</td>
<td>10.93</td>
<td>10.93</td>
</tr>
<tr>
<td>physical write IO requests</td>
<td>189,229,458</td>
<td>104,961.94</td>
<td>69.07</td>
<td>26,240.49</td>
<td>17,525.92</td>
<td>0.36</td>
</tr>
<tr>
<td>physical write bytes</td>
<td>1,564,020,015,104</td>
<td>867,531,829.36</td>
<td>570,889.19</td>
<td>216,882,957.34</td>
<td>144,848,823.42</td>
<td>4.021.59</td>
</tr>
<tr>
<td>physical write total IO requests</td>
<td>195,062,463</td>
<td>108,197.40</td>
<td>71.20</td>
<td>27,049.35</td>
<td>18,062.80</td>
<td>0.87</td>
</tr>
<tr>
<td>physical write total bytes</td>
<td>1,624,347,823,104</td>
<td>900,994,499.05</td>
<td>592,909.68</td>
<td>225,248,624.76</td>
<td>150,433,900.75</td>
<td>12.754.92</td>
</tr>
<tr>
<td>physical write total multi block requests</td>
<td>1,102</td>
<td>0.61</td>
<td>0.00</td>
<td>0.15</td>
<td>0.18</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Figure 36. Cluster AWR report: System Statistics

AWR and redo log switches

Redo logs are key to Oracle database resiliency and performance. The Oracle write size to the logs ranges from 512 bytes up to 1 MB. Oracle switches to the next log file based on multiple conditions, such as how full the log buffer and the log file are, and time.

Configure the redo logs size so that Oracle switches log files only a small number of times per hour. Ensure that there are enough log files so that they never wait for the archiving processes to complete at log switch time.

The log switch count is in the Instance AWR report, as shown in the following figure. The number under Total is how many switches occurred during the AWR report duration. The per Hour value is a derived number, an estimate based on the activity during the AWR report.

Instance Activity Stats - Thread Activity

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>log switches (derived)</td>
<td>2</td>
<td>3.99</td>
</tr>
</tbody>
</table>

Figure 37. Instance AWR report: log_switch