ORACLE ASM ON DELL EMC SCALEIO

Best practices for deploying and managing ASM

January 2018

Abstract

This white paper covers best practices and methodologies for managing the Oracle ASM logical volume manager on Dell EMC ScaleIO.

H16866

WHITE PAPER

Copyright

Contents

Chapter 1 Executive Summary

This chapter presents the following topics:

Overview

This technical document explores optimal methods for Oracle DBAs to consume ScaleIO software-defined elastic storage through Oracle's ASM volume manager and file system.

The document highlights best practices at all layers the solution from ScaleIO pool design and ASM diskgroup configuration to database parameter settings.

Oracle ASM is a flexible and powerful technology that delivers optimal performance for Oracle databases including Oracle RAC. ScaleIO compliments this technology with an elastic and scalable software-defined storage solution that can meet user's most demanding database needs.

Oracle Database is a sophisticated system designed for high performance transaction processing and/or advanced data mining and analytics. An optimal storage sub-system is critical to the performance of the database. With database footprints growing exponentially, advanced storage technologies from Dell EMC provide leading-edge performance and flexibility.

The document includes many specific recommendations that are summarized at the end of each chapter.

Document purpose

Oracle's Automatic Storage Management (ASM) product is an optional component of the Oracle database software suite that provides storage management functions to the database as well as other applications through an interface similar to an Oracle database.

Oracle ASM replaces functionality traditionally provided by a Logical Volume Manager (LVM) or File System.

ASM has become the standard volume management system for Oracle databases using Oracle Real Application Clusters (RAC), and is increasingly becoming the standard volume management system for all Oracle databases using block-based storage.

Scope

This document provides best practices, tips, scripts and methodologies for deploying and managing Oracle ASM on Dell EMC ScaleIO software defined storage.

Note: This document is not an ASM user's guide nor a database administration guide, although both areas are covered in some detail.

This document assumes a high level of technical knowledge and familiarity with both Oracle ASM and ScaleIO.

Notes

Versions

This document was prepared primarily using Oracle Enterprise Linux 7, Oracle ASM 12.2.0.1, Oracle 12.2.0.1 database and VMware ESX 6, although the architecture outlined here would be sufficient for other software versions of this stack.

Features

The features described in this document may or may not be implemented on earlier releases of the described software stack.

Limitations

Limitations described may have been fixed on subsequent releases.

Terminology

Table 1 provides definitions for some of the terms used in this white paper.

Table 1. Terminology

Term	Definition
ASM	Automatic Storage Management—ASM is Oracle's preferred method of storage for Oracle files on block storage devices.
CRS	Cluster Ready Services—A layer of software provided by Oracle to facilitate an Oracle RAC cluster.
MESH Mirror	ScaleIO's technique to protect against node failure by mirroring data blocks on separate nodes of the protection domain. (It implies many to many storage server host network interconnectivity.)
Pool	A logical grouping of physical storage devices from SDS nodes in a single protection domain from which are created volumes that are presented to SDC hosts.
Protection domain	A group of ScaleIO Data Clients (SDCs) and ScaleIO Data Servers (SDSs) grouped together to optimize network traffic and provide a level of isolation for critical applications.
RAC	Real Application Clusters—Oracle's horizontal scale out solution where a single set of database files can be processed by Oracle instances on multiple nodes simultaneously.
RAID	Redundant array of independent disks—A method for storing information where the data is stored on multiple disk drives to increase performance and storage capacity and to provide redundancy and fault tolerance.
SDC	ScaleIO Data Client—A node inside or outside of a ScaleIO storage cluster that can consume storage from the ScaleIO system.
SDS	ScaleIO Data Server—A node in a ScaleIO cluster that stores data on local disk, whether SSD, or HDD, and presents logical volumes to clients.

Term	Definition
Storage pool	Storage pools are logical collections of disks in ScaleIO. All disks within a given pool should be of the same performance, caching mechanism, and capacity. Multiple storage pools can have the same characteristics, but a single disk can only belong to a single storage pool.
VMDK	VMware virtual disk—VMDKs appear as block devices to guest OSes running under VMware, but are stored in VMFS datastores.
VMFS	VMware File System—VMware's proprietary clustered file system that is used to create datastores on block devices presented from an array. VMware storages VMDKs as well as other files required by the guest OSes running under VMware.
Volume	An SCSI block device presented to the host via the SDC client software, by the ScaleIO cluster.

We value your feedback

Dell EMC and the authors of this document welcome your feedback on the solution and the solution documentation. Contact EMC.Solution.Feedback@emc.com with your comments.

Authors: Graham Thornton, Fiona O'Neill

Chapter 2 Preparing ScaleIO for Oracle ASM

This chapter presents the following topics:

Overview

In the following chapter we explore the process to prepare storage on ScaleIO for use by Oracle ASM.

Converged or hyperconverged

When planning to deploy an Oracle database that uses ScaleIO, the solution architect or DBA must decide if the ScaleIO cluster will be converged or hyperconverged.

With a converged architecture, as shown in Figure 1, each node in the cluster will either be a ScaleIO Data Client (SDC) consuming storage from ScaleIO and running the database software or a ScaleIO Data Server (SDS) storing ASM data and providing it to the SDC/database nodes.

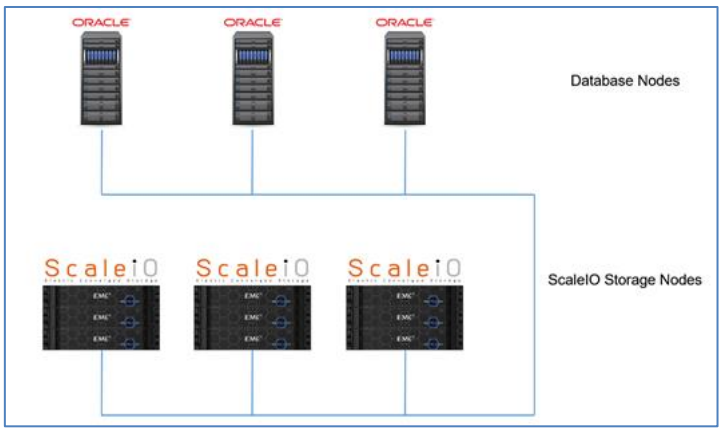

Figure 1. Oracle and ScaleIO—Converged infrastructure

In a hyperconverged architecture, as shown in Figure 2, some or all nodes will be both SDCs and SDSs, both running the database software and acting as storage nodes for ScaleIO.

The hyperconverged architecture may allow for more efficient utilization of the compute, network and storage resources, but may require additional Oracle software licenses.

Figure 2. Oracle and ScaleIO—Hyperconverged infrastructure

Keeping SDCs and SDSs separate may lead to some nodes being less than fully utilized, but may be beneficial from a license management perspective, while providing for more I/O headroom.

Disk type selection

When designing Oracle ASM storage, it is important to match the capabilities of the underlying storage infrastructure to the storage requirements of the database.

The types of disks that are suitable will depend on the quantity of data to be stored and the expected performance needs of the databases located on the ASM storage.

ScaleIO is a software-defined storage solution allowing the user to create a cluster of nodes configured with single or multiple types of storage, along with appropriate caching that act together to deliver storage to applications, meeting any application performance requirements for throughput. I/O operations, and/or latency.

ScaleIO nodes can use spinning disks including high-capacity SATA disks including10k 15K RPM spinning disks, as well as flash drives and Non-Volatile Memory Express (NVMe).

As a general rule:

- For low performance and higher storage capacity requirements, use spinning disks.
- For moderate performance and higher storage capacity requirements, use spinning disks along with SSD enable cache.
- For high performance and lower storage capacity requirements, use flash or NVMe.

Note: Dell EMC recommends leveraging your Dell EMC Systems Engineer to properly size any ScaleIO solution to your workload.

Storage pool design

All physical storage devices in ScaleIO are organized into storage pools. The volumes presented to hosts are created from one of the storage pools within ScaleIO. ScaleIO allows pools of physical devices to be created and shared across numerous nodes within a ScaleIO cluster.

Pools can be created using drives with a mix of capacity and performance characteristics, although this approach will lead to non-deterministic performance to the database. In general, the performance characteristics will match the performance of the lowest performance devices within a pool.

Creating pools – best practices

Best practice for ASM is to create pools of devices with identical performance and capacity characteristics.

Multiple storage pools may be configured, although in general, it is recommended to use fewer storage pools as this reduces complexity and ease of management.

A single storage pool can contain up to 330 physical devices. If the ScaleIO cluster has more than 330 physical devices then multiple pools will be necessary.

Storage pool volumes

Volumes created in storage pools are automatically protected against node failure and data loss through the MESH mirroring technique.

ScaleIO takes care of distributing the blocks across the cluster, as well as protecting the user's data by creating mirrors of every block written to disk on redundant nodes using MESH mirroring.

Snapshots of volumes can be created in the same storage pool of the volume itself, and will consume capacity consistent with the rate of change applied to the source volume or the snapshot. Note that ScaleIO utilizes a redirect-on-write mechanism to maintain snapshot copies of data.

Pools can be created with the "Use RAM Read Cache" option which if enabled allows the RAM on the SDS node to be used as a read cache. This can be particularly beneficial to storage pools based on spinning disks.

Cache option

If the "Use RAM Read Cache" option is enabled, the write handling of the pool may also be cached, or set to pass-through.

The caching option may have performance benefits for storage pools with slower spinning disks but is usually not advantageous for pools of solid state or NVMe devices and may even degrade performance due to the extra processing involved.

If the ScaleIO cluster includes NVMe devices, these may be configured for the read cache of a pool by selecting "Read Flash Cache".

Checksum option

A storage pool may have additional checksums generated, this is enabled by selecting the "Use Checksum" option. While this results in some I/O processing overhead, it provides additional piece of mind for sensitive, mission-critical data.

Volumes and consistency groups

Volumes

When suitable storage pools have been established, storage may be allocated in the form of volumes that will be presented to the database hosts.

ASM will consume these volumes as ASM disks, or they may be used to create VMFS datastores under VMware from which will be created VMDK virtual disks that ASM will consume as ASM disks. Additionally, creating filesystems and deploying non-RAC databases on these filesystems is an option.

The volume creation wizard provided by the ScaleIO GUI interface allows the user to specify the volume name, size, storage pool in which to create the volume, whether the new volume should use a RAM Read Cache (if available at the pool level), and whether the new volume should be thick or thin provisioned.

Note: There is a small performance overhead associated with thinly provisioned volumes especially on spinning disks. With solid state storage, this overhead is more than offset by the capacity management advantage of thin provisioning.

Consistency groups

Consistency groups allow multiple volumes to be grouped together into a single logical entity when snapshotting.

Existing volumes may be added to a consistency group, or the consistency group may be created with all new volumes.

All ASM disks in a single ASM diskgroup or VMFS datastore should belong to the same consistency group. This allows ScaleIO to make storage crash-consistent snapshots of ASM diskgroups which can then be re-provisioned to secondary hosts.

Volume sizing

With Oracle ASM 11g, the largest supported ASM disk size is 2 TB. Do not create ScaleIO volumes (or VMDKs) larger than this size for ASM 11g use.

For Oracle ASM 12c the largest ASM disk size is 32 PB.

The smallest ASM disk that ASM can handle is 4 MB.

Note: Within an ASM diskgroup, all ASM disks should be of equal size. ASM is not able to balance extents across non-uniform ASM disk sizes within a single diskgroup.

For ASM 12c larger volume sizes may be used, but if using virtualized hosts, be aware that VMware has a maximum supported volume size of 64 TB for an RDM, and 62 TB for a VMDK disk.

Chapter 3 Presenting Volumes to the Host

This chapter presents the following topics:

Volume mapping

The Linux database hosts must have the ScaleIO SDC client software installed, and the new volumes must be mapped to the database hosts in the ScaleIO interface, before Linux can access the new volumes.

Devices presented to Linux will appear as /dev/sciniX in the Linux device table, where X is a letter assigned to the device.

Partition offset

Neither ScaleIO nor ASM require volumes to be partitioned, but many system administrators and database administrations prefer to create partitions as this clearly informs the system administrator that the device is in use. Either approach is valid.

The optional ASMLib package requires a partition offset on a device before it can be stamped for ASM, but ASMLib is not recommended for use with ScaleIO (see ASMLib note later in this section).

For Red Hat Linux 7 and later the offset of partition is automatically set to 1 MB for devices larger than 4 GB. For older Linux systems, the default offset is 31.5 KB. This older offset can lead to degraded I/O performance. To avoid this alignment problem, Dell EMC recommends setting the partition offset to at least 1 MB.

The following example shows a partition being created on Linux with a 1MB (2048 sector) offset.

```
[root@localhost ~]# parted /dev/scinia mklabel gpt ; parted /dev/scinia mkpart
primary 2048s 100%

Information: Don't forget to update /etc/fstab, if necessary
```

UDEV rules

UDEV functionality was introduced into Linux with kernel 2.5, and allows devices to be given consistent names and permissions across clusters.

The UDEV rules use the SCSI ID of a device to assign an OS name and permission. Since the SCSI ID of a device does not change, and is consistent across cluster nodes, this allows ASM to see a consistent device name across clusters and reboots.

In bare metal environments, the ScaleIO drv_cfg program can be used to inspect the SCSI ID of new ScaleIO devices presented to the Linux host:

```
[root@sdc01 ~]# /opt/emc/scaleio/sdc/bin/drv_cfg --query_block_device_id --
block_device /dev/scinia
23719f5a70163008-fa12defe0000000b
```

The SCSI identifier of the new disk is 23719f5a70163008-fa12defe0000000b.

The DBA can verify which ScaleIO device this is by taking the last 16 bytes of the SCSI ID and using the following command:

```
[root@sio01-mgmt ~]# scli --query_all_volumes | grep fa12defe0000000b
 Volume ID: fa12defe0000000b Name: DBASM9 Size: 1000.0 GB (1024000 MB) Mapped to 4
SDC Thin-provisioned
```

The new device is volume DBASM9 in ScaleIO. It is 1 TB in size and is thin provisioned.

We can now use the SCSI ID with UDEV rules to set the permission of the new device as well as an alias.

Then the DBA needs to create (or edit) a file in the */etc/udev/rules.d* directory. In this case the file is named *99-oracleasm.rules*.

```
[root@sio01-mgmt ~]# vi /etc/udev/rules.d/99-oracleasm.rules
```

Add the following – this should all be one line:

```
KERNEL=="scini*", SUBSYSTEM=="block", PROGRAM="/opt/emc/scaleio/sdc/bin/drv_cfg --
query_block_device_id --block_device /dev/%k", RESULT=="23719f5a70163008-
fa12defe0000000b", SYMLINK+="oracleasm/dbcrs01", OWNER="grid", GROUP="asmadmin",
MODE="0660"
```

In the above example Linux creates an alias called */dev/oracleasm/dbcrs01* when it finds the *SCSI ID 23719f5a70163008-fa12defe0000000b*. The alias will be owned by the grid:asmadmin and have permissions of 660.

Once the UDEV rules file is created, restart UDEV. The example below is from Red Hat Enterprise Linux 7.

```
[root@sio01-mgmt ~]# /sbin/udevadm control --reload-rules
[root@sio01-mgmt ~]# /sbin/udevadm trigger
```

Note that older versions of Linux use different syntax. Check the documentation for your version of Linux to ensure you are using the correct syntax.

Check that the alias exists:

```
[root@sio01-mgmt ~]# ls -al /dev/oracleasm/*
lrwxrwxrwx 1 root root 10 Aug  3 17:31 /dev/oracleasm/dbcrs01 -> ../scinia
```

The DBA can now use *asmca* with a discovery string of */dev/oracleasm/** to locate the devices and create ASM diskgroups from them.

ASMLib

ASMLib is an optional library available on Linux platforms to simplify storage management and to reduce the load on the operating system.

ASMLib will scan for ASMLib disks upon system startup if the ORACLEASM_SCANBOOT directive is set to TRUE in the ASMLib configuration file found at */etc/sysconfig/oracleasm*.

However the ASMLib “scandisks” step is performed in the Linux boot sequence before ScaleIO has presented volumes, resulting in ASMLib finding no disks.

Therefore it is necessary if the DBA wishes to use ASMLib to create an additional startup script in the /etc/rc3.d directory that issues a second ASMLib “scandisks” command once ScaleIO has presented the volumes. For this reason, Dell EMC recommends against using ASMLib with ScaleIO.

ASM Filter Driver (12cR2)

Oracle introduced ASM Filter Driver in Oracle Database 12c (12.1.0.2). The ASM Filter Driver replaces the functionality of ASMLib, and adds new functionality including the ability to protect ASM devices from I/Os that do not originate from the Oracle stack.

Oracle has stated that future versions of ASM Filter Driver will include support for TRIM on thinly provisioned disks, allowing deleted blocks to be released back to the pool for reuse.

Note: ASM Filter Driver does not require partition headers.

New devices can be stamped as ASM disks as follows:

```
[root@oel6solo ~]# asmcmd afd_label DATA1 /dev/sdb
```

In the example above we have stamped the device */dev/sdb* as the ASM disk DATA1.

With ASM Filter Driver, the DBA will see devices named *AFD:diskname*.

```
[oracle@oel6solo asm]$ asmcmd lsdsk --statistics
Reads  Write  Read_Errs  Write_Errs  Read_time  Write_Time  Bytes_Read
Bytes_Written  Voting_File  Path
 1374    488           0           0       .624        .455    18339328
2746880             N  AFD:DATA1
 2213     80           0           0       .776        .098    30583808
1172992             N  AFD:DATA2
 1780    333           0           0       .746        .305    34169344
4833280             N  AFD:DATA3
 1576    111           0           0       .682        .107    29904896
904704             N  AFD:DATA4
```

The ASM instance parameter ASM_DISKSTRING should be set to “*AFD*:*” when using ASM Filter Driver.

```
[oracle@oel6solo asm]$ srvctl config asm
ASM home: <CRS home>
Password file: +DATA/orapwasm
ASM listener: LISTENER
Spfile: +DATA/ASM/ASMPARAMETERFILE/registry.253.937659731
ASM diskgroup discovery string: AFD:*
```

Chapter 4 Preparing ASM for Oracle Database

This chapter presents the following topics:

Overview

The following section explores how the ASM instance should be configured to work with ScaleIO. We also investigate the performance impacts of modifying diskgroup and disk settings.

Shared or database-specific ASM diskgroups

Oracle recommends that an ASM diskgroup is shared between multiple databases to keep the number of diskgroups down to a manageable level. In the case where multiple databases share common I/O characteristics, are of similar version and hardware levels and of largely equal importance to an organization then consolidation of ASM diskgroups across multiple databases may be beneficial, but that approach will limit the effectiveness of individual crash-consistent database snapshots.

However where one system, such as an ERP, is clearly of higher importance than other databases, uses a significantly different release of Oracle or is likely to require a significantly more complex upgrade strategy than other systems, then such consolidation may introduce logistical headaches or performance issues later on.

Finally, if ASM diskgroups are shared among multiple databases any upgrade of the ASM infrastructure will require a lockstep upgrade of all databases.

Mixing of production and non-production data in the same ASM diskgroups is not advised.

When designing the ASM diskgroup layout, the DBA should be aware of Oracle's ASM limits, as shown in Table 2.

Table 2. ASM limits

Maximum	ASM 11g	ASM 12c
Diskgroups per ASM instance	63	511
ASM disks per ASM diskgroup	10,000	10,000
ASM disk size	2 TB	32 PB
Storage size	20 PB	320 Exabytes
Files per ASM diskgroup	1 million	1 million

How many ASM diskgroups?

Oracle recommends just two diskgroups by default:

- A DATA diskgroup to hold all data, index, undo and temp files
- An FRA diskgroup that holds archivelogs and backups

By default, redo logs and control files are mirrored to both the DATA and FRA diskgroups.

For DBAs planning to use ScaleIO snapshots, a two-diskgroup solution should be considered the minimum, but control files and redo logs should NOT be mirrored to both diskgroups.

When using ScaleIO snapshots, it can be beneficial to snapshot the DATA and FRA diskgroups independently. This allows for the FRA diskgroup to be restored at a time slightly ahead of DATA, allowing Oracle to perform additional database recovery through the application of archivelogs.

When using a mix of storage pools with both high capacity spinning disks and high performance solid state devices additional diskgroups should be considered, which correspond to the different performance storage pools.

It is important to ensure that control files, redo log files, temp and undo reside exclusively in a pool with high performance.

Since Oracle Database 11gR2 it is recommended to store the OCR and Voting Files in a standalone ASM diskgroup, as shown in Table 3.

Table 3. ASM diskgroups

ASM diskgroup	Purpose	Database tablespaces	Storage pool characteristics	RAM/Flash cache
GRID	Cluster Voting Disk, OCR and ASMSPFile	N/A	Performance	No
DATA	Database data files, temp files, redo logs and undo	SYSTEM, SYSAUX, DATA, INDEX, USERS, TEMP, UNDO	Performance	No
FRA	Archive logs and backups	N/A	Capacity	No

ASM disk sizes and counts

Oracle recommends a minimum of four ASM disks per ASM diskgroup. Each ASM disk should be the same size and from the same storage pool to allow ASM to properly stripe data in a manner that compliments with the ScaleIO striping.

Multiple ASM disks allows for multiple I/O queues in the OS, and so can assist with performance. However too many disks may prove cumbersome to manage.

When choosing how many disks to use in a new ASM diskgroup, one option is to determine the expected data size after the first 12 months, and then factor in the expected rate of growth.

Because all disks in a diskgroup should be of uniform size, a smaller initial disk size will provide for more granular growth. A larger disk size will result in a fewer number of disks initially and will allow for larger capacity growth without demanding an excessive number of new disks later on. However, if the DBA selects thick provisioning, larger disks may waste capacity.

Table 4 summarizes the disk size/growth calculation.

Table 4. Disk size and growth calculation

Expected rate of growth (%YOY)	Number of disks for Year 1
25% or higher	4
20% to 25%	5
16% to 20%	6
12% to 16%	8
Less than 12%	10

Assuming an expected 1 TB of data by the end of the first year, and an expected rate of growth of 25 percent, we might select to create our ASM diskgroup with four 250 GB volumes.

- **At the end of the year one**—One additional volume provides sufficient growth to handle the next 12 months. Total capacity is now 1.25 TB.
- **At the end of year two**—Two additional volumes for a total of seven are added. Total capacity is now 1.75 TB.
- **At the end of year three**—One additional volume is added bringing the total capacity to 2 TB.
- **At the end of year four**—Two final volumes are added for a total capacity of 2.5 TB.
- **At the end of 60 months**—There are 10 volumes, as shown in Figure 3.

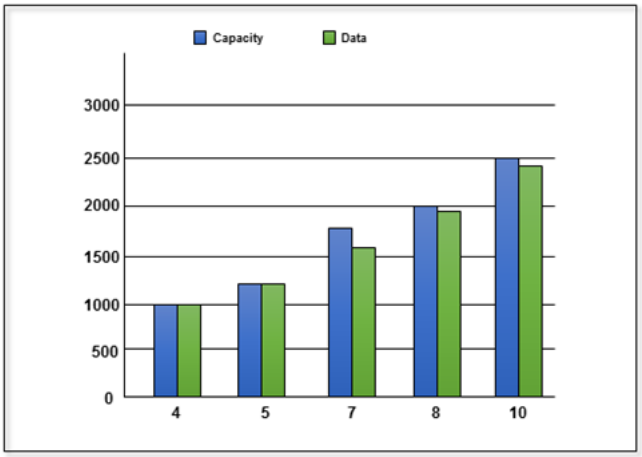

Figure 3. Capacity sizing over 60 months

Remember that the largest disk Oracle ASM is able to use is 2 TB for ASM 11gR2, and 32 PB for ASM 12c.

Note: Dell EMC recommends a minimum of four disks per ASM diskgroup.

Resizing existing ASM disks

ScaleIO allows for existing volumes to be resized larger to accommodate data growth.

By resizing existing volumes, we can grow an ASM diskgroup without having to add new ASM disks. This alternative approach has the following benefits:

- New volumes do not have to be added to any Consistency Group or mapping.
- No ASM data block relocation/compaction takes place.
- In VMware, which has a 256 LUN limit per ESX host, the additional capacity does not subtract from that limit.

Note: If ASMLib is in use, then the diskgroup must be taken offline for ASM disk resizing.

WARNING: As with any maintenance operation, selecting a time when workloads are low and having a recent backup to fall back on in the event of a disaster is strongly advised.

To resize the ASM disks and diskgroup in this manner, first use the ScaleIO GUI or command line interface to resize the size of each volume. Resize all volumes of the ASM diskgroup to the same size.

- **If VMware is in use**—Rescan the HBAs of each ESX server in the VMware cluster where the database nodes that use the ASM diskgroup reside.

  Rescan the SCSI bus of the guest OS of each database node. This can be accomplished with the following Linux command:

  ```
  for hst in $(ls /sys/class/scsi_host) ; do echo "- - -" >
  /sys/class/scsi_host/$hst/scan ; done
  ```

- **If ASMLib is in use**—Execute the /etc/init.d/oracleasm scandisks command.
- If the ASM disks have partition headers—Repartition each disk in the ASM diskgroup. Note that the Linux parted tool will warn that the contents of the disk will be lost. With ASM this is not the case and the warning can be safely ignored.
- If the ASM diskgroup was taken offline—Bring it back online.

Resize all volumes

1. Connect to the ASM instance and execute the command:

   ```
   SQL> alter diskgroup MYDISKGROUP resize all;
   Diskgroup altered.
   ```

2. Use the *asmcmd lsd*g command to verify the additional capacity is now visible.

ScaleIO volumes, ASM diskgroup, and database datafile relationship

Figure 4 demonstrates the relationship between the layers of interacting storage that store data for the Oracle database.

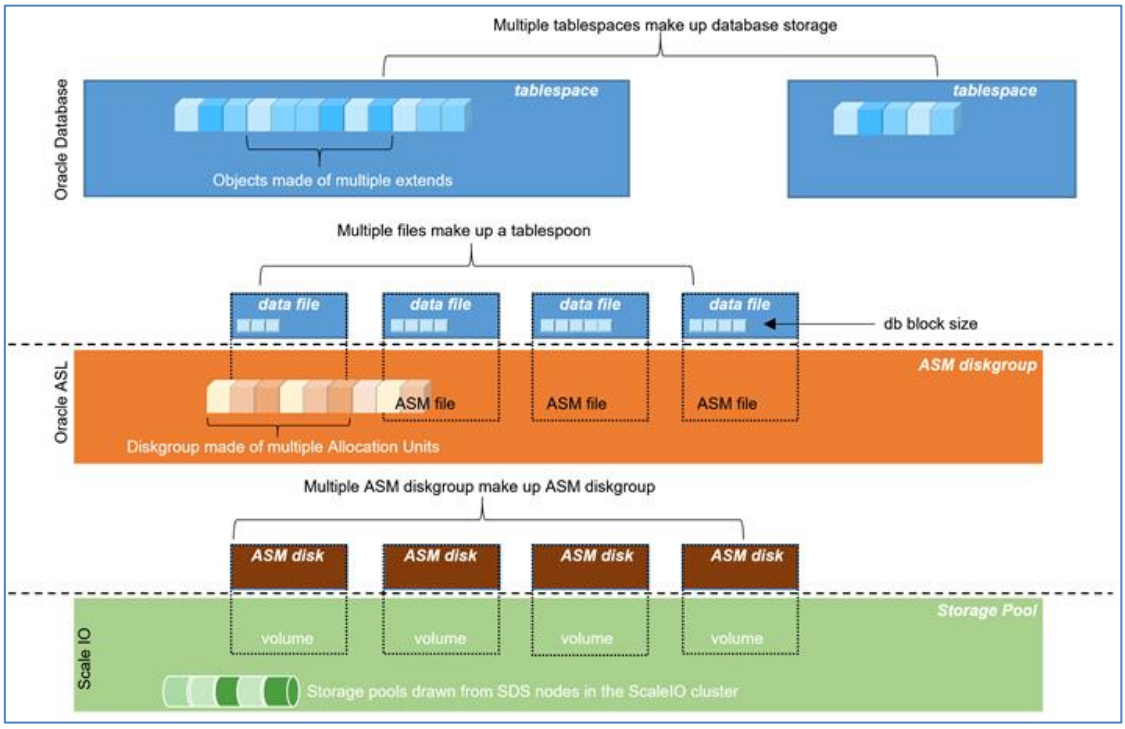

Figure 4. ScaleIO volume, ASM disk, Oracle data file and tablespace relationship

At the highest level in blue is the Oracle tablespace which contains objects made up of extents. These tablespaces comprise multiple data files stored in ASM.

The ASM files are stored in an ASM diskgroup made up of ASM Allocation Units. Each ASM diskgroup comprises multiple ASM disks.

The ASM disks map to storage volumes which are stored in storage pools in ScaleIO.

ASM redundancy

ASM includes the ability to mirror diskgroups to protect data from physical failures.

Available protection levels are NORMAL (one mirror), HIGH (two mirrors) or EXTERNAL where the data protection is handled by ScaleIO.

ScaleIO automatically protects the ASM data using a MESH mirror technique, whereby every block of data is held on two different nodes of the ScaleIO system. In the event of failure, ScaleIO immediately recreates a redundant copy of the data on a surviving node.

Therefore protecting data with ASM is unnecessary and ASM diskgroup protection be set to EXTERNAL.

Note: When storing cluster voting disks in ASM, the DBA may consider using a protection level of NORMAL with two failure groups.

This will create mirrored copies of the voting disks as shown in Table 5:

Table 5. Voting disks and failure groups

ASM redundancy	Copies of voting disk	Number of failure groups required
External	1	N/A
Normal	3	2
High	5	3

See Oracle documentation *Managing Oracle Cluster Registry and Oracle Local Registry* sub-section *Storing Voting Disks on Oracle ASM*.

Recommendation

Dell EMC recommends using external redundancy in all ASM diskgroups except for ASM diskgroups where cluster voting disks are stored, for which normal redundancy with two failure groups should be considered.

Allocation Unit (AU) size

The Allocation Unit or AU size is the smallest amount of storage that may be allocated to or removed from an ASM diskgroup.

Before ASM 11g

The AU size was set at the ASM instance level via the hidden parameter *_asm_ausize*.

ASM 11g

The AU when a diskgroup is created, but once created it cannot be altered, so it is important to set this correctly before data is loaded into ASM.

ASM 11g and 12c

The diskgroup AU size still defaults to the value specified in the hidden *_asm_ausize* parameter. The ASM AU size may be specified as 1 MB, 2 MB, 4 MB, 8 MB, 16 MB, 32 MB or 64 MB.

ASM 12c

The AU size governs the maximum size of the disks that may be added to a diskgroup, as shown in Table 6:

Table 6. AU and ASM disk size

AU size	Maximum ASM disk size
1 MB	4 PB
2 MB	8 PB
4 MB	16 PB

AU size	Maximum ASM disk size
8 MB and above	32 PB

The following example shows a new diskgroup called BIG_DATA being created on a Linux system with ASMLib with a 16 MB allocation unit:

```
SQL> create diskgroup BIG_DATA external redundancy disk '/dev/oracleasm/bigdata01'
  2  attribute 'au_size' = '16M';
Diskgroup created.
```

The allocation unit does not directly govern the size of I/O to the storage subsystem that is controlled by the extent size or the maximum ASM I/O size parameter which will be covered later.

However, allocation unit size does control the size of the metadata held in ASM, and for larger databases there is a performance advantage of using a larger AU size.

A 1 GB data file using the default Allocation Unit would require 1024 1 MB extents. Setting the AU to 4 MB would reduce the extent count to 256.

Furthermore, due to ASM managing data in ASM extents which are derived from the allocation unit, a larger AU size increases the likelihood that data will be stored sequentially on spinning disk.

In our example above, with 1 GB of data, the 1024 1 MB extents may be stored randomly across the available disks as ASM seeks to balance the capacity of each disk.

The 256 4 MB extents will also be stored randomly, but during any scanning of the data there will be far fewer extents of data to be randomly retrieved. Data is more likely to occur sequentially and seek times are significantly reduced where spinning disks are used.

Testing has shown that large data warehouses can benefit from up to 16 percent performance improvement using a larger AU size of 8 MB or 16 MB.

Recommendations

Dell EMC recommends an AU size of between 1 MB and 4 MB for mostly OLTP workloads, and an 8 MB or larger AU size for OLAP workloads.

For very large databases, a 16 MB or larger AU size should be considered for diskgroups where the majority of the data will be stored.

ASM extents

ASM organizes data within diskgroups into extents.

An ASM extent is distinct from an Oracle table extent. An ASM extent consists of one or more allocation units.

It is important to remember that when using ASM mirroring, ASM mirrors by ASM extent, not disk or diskgroup. Although Dell EMC discourages the use of ASM mirroring, for those

DBAs who wish to leverage the feature it must be remembered that this extent based mirroring when used in conjunction with MESH mirroring means that ASM has no visibility to the physical disk layout, and therefore ASM mirrored extents may be allocated back to the same physical device undermining the protection of the mirror.

In ASM 11g, the extent size increases with the size of the ASM file as shown in Table 7.

Table 7. ASM file extents

Extents in ASM File	Extent Size (11.1)	Extent Size (11.2)
0 to 19,999	1 * AU Size	1 * AU Size
20,000 to 39,999	8 * AU Size	4 * AU Size
40,000 and higher	64 * AU Size	16 * AU Size

Note: The DBA has no direct control over the extent size in ASM.

ASM stripesize, coarse, and fine grained striping

The ASM stripe size is defined by the hidden parameter *_asm_stripesize* and defaults to 128KB. This value is used by files that have the fine grained striping option set in the ASM file template.

In ASM 10g, control files, redo log files and flashback log files all feature fine grained striping. In ASM 11g and 12c, only control files have fine grained striping.

Files that do not use fine striping are considered coarse. Coarse striping is the size of the AU. We can review which files use which setting in the V$ASM_TEMPLATE view:

```
SQL> select name, stripe from V$ASM_TEMPLATE;

NAME                           STRIPE
------------------------------ ------
PARAMETERFILE                  COARSE
ASMPARAMETERFILE               COARSE
ASMPARAMETERBAKFILE            COARSE
DUMPSET                        COARSE
CONTROLFILE                    FINE
FLASHFILE                      COARSE
ARCHIVELOG                     COARSE
ONLINELOG                      COARSE
DATAFILE                       COARSE
TEMPFILE                       COARSE
BACKUPSET                      COARSE
AUTOBACKUP                     COARSE
XTRANSPORT                     COARSE
CHANGETRACKING                 COARSE
FLASHBACK                      COARSE
DATAGUARDCONFIG                COARSE
OCRFILE                        COARSE
```

```
OCRBACKUP                      COARSE
ASM_STALE                      COARSE

19 rows selected.
```

Dell EMC has observed a 7 percent to 12 percent performance improvement from setting redo logs to fine grained striping of 128 KB.

To change the striping from coarse to fine or vice-versa, the DBA can modify the ASM diskgroup template as follows:

```
SQL> ALTER DISKGROUP MYDATA ALTER TEMPLATE onlinelog ATTRIBUTES
(FINE);
Diskgroup altered.
```

Recommendation

Dell EMC recommends setting the template for redo log files and temp files to fine, and leaving the ASM stripesize at 128 KB.

ASM maximum I/O size

The hidden parameter *_asm_maxio* governs the size of the largest I/O that ASM will make to the storage subsystem.

The parameter defaults to 1 MB on ASM 10g, 11g and 12c.

Although the parameter may be increased it does not actually allow I/O to exceed to 1 MB as this value appears to be a hard limit. Setting max I/O to a value less than 1 MB will likely increase latency for scanning operations.

ASM diskgroup—adding and removing disks

As databases grow and require more storage, ASM diskgroups can be expanded by adding new disks.

ASM will automatically rebalance allocations across all available disks to maintain a uniform distribution of data throughout the diskgroup.

When adding new disks to a diskgroup, it is advisable to add all new disks in the same command. This allows ASM to rebalance once for all new disks, instead of repeatedly as each new disk is added.

The DBA may assign a rebalance power limit to the operation to limit the impact on a production system of the rebalance operation. The higher the rebalance power limit the more resources will be used to rebalance the disk allocation, and the faster the rebalance operation will complete.

In the following example we add two new disks - */dev/oracleasm/data03* and */dev/oracleasm/data04* to the existing DATA disk group.

```
SQL> alter diskgroup DATA add disk '/dev/oracleasm/data03','/dev/oracleasm/data04'
rebalance power 1;
```

The DBA can monitor the progress of the rebalance operation by observing the V$ASM_OPERATION view:

```
SQL> @operation11
DISKGROUP_NAME  OPERATION            STATE     PWR  ACTUAL
PCT_DONE EST_MIN ERROR_CODE
--------------- -------------------- -------- ---- ------- -------
- ------- ----------
DATA            REBAL                RUN         1       1
2.8%       2
```

Removing disks

If disks need to be removed from an ASM disk group, then again all disks should be removed in a single command to minimize the rebalance operations:

```
SQL> alter diskgroup DATA drop disk DATA_0002,DATA_0003 rebalance power 6;
```

In the above command, the disk names specified must match those reported in the V$ASM_DISK view.

ASM intelligent data placement and physical disk placement

In the past, Oracle DBAs seeking maximum performance sought to place critical files, usually the redo logs, on the outer rim of physical disks to achieve maximum performance.

The introduction of Zone Bit Recorded (ZBR) disks meant that there was a greater volume of data on the outer edges of the physical spinning disk, and with the higher relative speed of the head passing over the disk, the transfer rate was up to 50 percent higher on the outer edge.

Since version 11.2.0.3, ASM offers Intelligent Data Placement or IDP that seeks to place files on the fastest outer parts of disks.

However for ScaleIO, such complexities are not relevant and will yield little to no performance improvement. The underlying geometry of any spinning disks is hidden from the SDC host.

Recommendation

DELL EMC recommends relying on the built in optimization technology of ScaleIO to handle data placement within physical disks.

ASM compaction and rebalance (ASM 11.1.0.7 and later)

When disks are added or removed from a diskgroup, ASM rebalances the blocks on the ASM disks so that the data is evenly spread across all ASM disks.

After blocks have been relocated, ASM enters a compaction phase where blocks are consolidated to the outer part of each disk, any evacuated block space is eliminated and the high water mark of each disk is reset.

This compaction phase is unnecessary on ScaleIO as the physical geometry of the disks is hidden from the application, and the LBA is no indication of the physical placement of a block on the storage media.

ASM Compaction may be disabled by setting the hidden parameter _DISABLE_REBALANCE_COMPACT to true in the ASM instances.

Alternatively, the hidden ASM diskgroup attribute _REBALANCE_COMPACT may be set to FALSE on a diskgroup by diskgroup basis.

Although this step may be disabled by the parameters shown above, Dell EMC does not recommend setting hidden parameters in ASM without the approval of Oracle Support.

Although the compaction phase is redundant on ScaleIO, leaving it enabled does not negatively impact performance once the rebalance operation has completed.

Summary

The following summarizes Dell EMC best practices for ASM diskgroup design when deployed on ScaleIO:

- **Dedicated or shared diskgroups**—Do not mix production and non-production workloads into the same diskgroups. Use dedicated ASM diskgroups where ScaleIO functions such as snaps will be used for a single database.
- **How many diskgroups?**—Basic configuration is DATA and FRA. RAC adds GRID. Additional disk groups may be evaluated based on application needs.
- **ASM disk sizes and counts**—Dell EMC recommends a minimum of four ASM disks per ASM diskgroup.
- **Redundancy**—Dell EMC recommends using EXTERNAL redundancy in all ASM diskgroups. See exception for ASM diskgroups where voting disks are stored.
- **ASM AU size**—Dell EMC recommends an AU size for OLTP workloads of between 1 MB and 4 MB, and an AU size for OLAP workloads of 16 MB or greater.
- **Coarse or fine grained striping**—Dell EMC recommends fine grained striping for control files, redo and flashback logs. Coarse grained striping should be used for all other files.
- **Migrations**—During migrations add all new disks and remove all old disks with a rebalance power setting of zero. Then rebalance the entire diskgroup in a single step.

Chapter 5 Using VMDKs and ASM with Virtualized Databases

This chapter presents the following topics:

Overview

Many customers now choose to virtualize their Oracle Database systems to gain the many advantages of running under VMware.

ScaleIO can deliver storage to databases virtualized with VMware which may be configured as VMFS datastores or as Raw Device Maps (RDMs).

VMFS provides flexibility to guest operating systems, and includes the ability to suspend a running VM, snapshot a VM, create clones and templates and share VMFS file systems across multiple guest operating systems across multiple ESX hosts.

Virtual disks, called VMDKs, may be created within a VMFS file system, and presented to the guest OS in a similar fashion to LUNs. These virtual disks can then be used as ASM disks by Oracle.

With VMFS 6, each VMDK may be up to 62 TB.

VMDKs or RDMs

VMware also supports Raw Device Mapping (RDMs) and NFS as alternatives to VMDKs.

Raw Device Maps or RDMs allow volumes from ScaleIO to be presented directly to the guest operating system and to largely bypass the hypervisor.

With RDMs, a mapping file is still created in VMFS, but all reads and writes go directly to the volume on ScaleIO.

RDMs come in two versions: Virtual Mode RDMs (vRDMs) and Physical Mode RDMs (pRDMs) which are also sometimes referred to as pass-thru RDMs.

- vRDMs are commonly used when creating a virtualized Microsoft Clustered Service across more than one ESX host, and have no benefit over pRDMs or VMDKs for Oracle databases on Linux using ScaleIO. They will not be discussed further in this document.
- pRDMs pass all SCSI commands to the volume except for REPORT LUN, exposing array based capabilities to the guest OS, but do not allow VM operations such as suspend, snapshot or clone.

  pRDMs may be used to create hybrid clusters with some nodes virtual and others physical.

  The largest pRDM supported in vSphere 6 is 64 TB.

Contrary to some published information, both VMDKs and pRDMs allow the use of VM vMotion, VM HA and VM DRS, although certain restrictions apply.

During testing, Dell EMC has found that VMFS incurs a performance penalty of between 5 percent to 13 percent compared to using RDMs. Note that some non-Dell EMC published reports have found little or no performance difference between VMFS and RDMs.

Note: ESX hosts are limited to 256 LUNs for VMFS volumes or RDMs. Since ASM typically uses multiple ASM disks or volumes to achieve performance and workload segregation, using RDMs may cause your ESX host to exceed the 256 LUN limit

Traditional VMFS/VMDK datastores

Traditional VMFS datastores hold individual files and VMDKs used by VMware and guest OS systems.

ASM disks are represented in VMFS as VMDKs, which are stored within datastores.

A datastore may reside on a single VMFS volume, or it may span up to 32 volumes by adding extents. If a data store is exhausted, it may be extended by adding extents on the same or a different VMFS volume.

If extending a datastore by adding extents on a new VMFS volume, ensure the new VMFS volume is created on a ScaleIO volume sharing the same size and performance characteristics as the primary datastore extent.

Datastores and VMDK sizes can be adjusted, but all VMDKs within a single Oracle ASM disk group should be the same size and reside on the same ScaleIO storage pool.

A VMFS volume may only be assigned to one datastore.

Datastores may be resized to use remaining free space on a VMFS volume. Volumes may also be resized, allowing VMFS volumes to be expanded and therefore datastores to be increased.

Adding extents to a datastore to allow it to span multiple VMFS volumes allows ESX to leverage multiple I/O queues to service I/O requests to a single data store but offers limited performance benefit as extents are concatenated.

High performance environments

For high performance environments, dedicate specific datastores to specific ASM diskgroups (e.g. +DATA, +FRA, +GRID etc.) and dedicate those ASM diskgroups to specific databases.

For very high performance environments, map VMDK's to separate datastores to increase isolation and performance parallelism, but be aware this will increase management overhead.

For high performance environments, do not consolidate multiple VMs onto the same datastores. This is especially important where workloads differ, such as OLAP and OLTP workloads.

Reserve approximately 20 percent of available capacity on each datastore for VMFS overhead if you are planning to leverage advanced VMware features such as VM snapshots or vMotion.

VMFS limits

When selecting VMFS 6, be aware of the following limits:

- Datastores may be created up to 64 TB.
- Datastore minimum size is 1.3 GB although 2 GB is suggested as a minimum.
- A datastore comprises between 1 to 32 extents, each extent may reside on a different VMFS volume.
- A single ESX 6 host can access up to 256 VMFS volumes or RDM Volumes and 256 datastores.

VMFS volumes

VMFS formats ScaleIO volumes as VMFS volumes.

A VMFS volume consists of the ScaleIO volume number and the disk serial number as seen by the ESX host, which are written back to the new VMFS volume as a header signature.

A VMFS volume does not necessarily have to consume the entire ScaleIO volume, but sharing a ScaleIO volume between multiple VMFS volumes or even another file system through partitioning is poor design and will likely lead to highly unpredictable performance, especially if that VMFS volume is used to support a datastore that houses virtual ASM disks.

VMFS volumes should be created in vSphere to ensure that the start sectors on are properly aligned.

Oracle RAC—Selecting VMDKs or RDMs

When using Oracle RAC additional considerations must be taken into account.

For a VMDK or pRDM to be shared between more than one guest OS, the multi-writer flag must be set. (See VMware KB 1034165). Doing so allows multiple guest OS systems to write to the same VMDK or RDM. However this modification also means that the VM can no longer be suspended, migrated with Storage vMotion, supported by VM HA or DRS, and snapshots are no longer possible.

Shared VMDKs must use a SCSI controller with the physical sharing flag enabled. Shared VMDKs must be set to Independent-Persistent.

VMDKs for use with Oracle RAC should be selected as Eager Zero Thick, as this is required when setting the multi-writer flag. Eager Zero Thick will negate the benefit of thin provisioning.

VMDKs and pRDMs feature summary

Table 8 summarizes the various features and limitations of VMDKs and pRDMs, including where the multi-writer flag (MWF) is enabled for Oracle RAC.

Table 8. VMDK and pRDM features and limitations

VMware/ volume feature	VMDK	VMDK w/MWF	pRDM	pRDM w/MWF	Notes
Maximum ASM disk size	62 TB		64 TB		
Suspend VM	Yes	No	Yes	No	Note 1
VM Snapshot – see independent persistent	Yes	No	Yes	No	Note 2
VM vMotion	Yes	Yes	Yes	Yes	
VM High Availability (HA)	Yes	Yes	Yes	Yes	
VM Distributed Resource Scheduler (DRS)	Yes	Yes	Yes	Yes	
VMware Data Recovery (VDR)	Yes	No	Yes	No	
Storage vMotion	Yes	No	Limited	No	Note 3
Virtual to physical migration	Yes	No	Yes	Yes	Note 4
Guest OS SCSI target-based software	Full	No	Yes	Yes	
SAN features (snapshot, clone, replicate)	Limited	Limited	Full	Full	Note 5
Hybrid RAC (VM and physical)	N/A	No	N/A	Yes	
Maximum RAC nodes	N/A	32	N/A	100+	

Note 1: Suspending VMs that are part of a cluster will result in node-eviction from the cluster.

Note 2: Making a snapshot of a VM with RDMs will result in the RDMs being converted to VMDKs.

Note 3: When migrating an RDM with Storage vMotion, the mapping file will be moved to the target datastore, but the volume will remain as before.

Note 4: Despite VMware documentation that states vRDMs cannot be presented to physical hosts, this is incorrect. Also note in testing, VMDKs have been successfully exported as iSCSI targets to physical hosts.

Note 5: You can utilize SAN based snapshots and clones of all volumes supporting VMFS datastores from outside of the guest OS without the use of tools such as AppSync, but such snaps and clones will be crash consistent, not application consistent.

VMFS and pRDMs offer advantages in a virtualized environment and careful thought should be given when selecting which technology to use.

Dell EMC ScaleIO SAN features for traditional VMDKs

ScaleIO capabilities may be used for volumes that support traditional VMFS datastores provided that all VMFS volumes of the datastore are grouped into a single consistency group. In this scenario the entire datastore and all VMDKs in it will be snapped as a single unit.

Typically, a single VMFS datastore will occupy a single VMFS volume on a single ScaleIO volume. But if the datastore uses multiple extents to span volumes, multiple ScaleIO volumes may support a single datastore.

If multiple VMFS datastores are used to support multiple VMDKs for multiple ASM disks within an ASM diskgroup, then all volumes for all datastores for all ASM diskgroups for an Oracle database should be included in a consistency group.

ScaleIO supports Write Order Fidelity (sometimes called Write Ordering), meaning that any snapshot or clone used in this fashion will provide a crash consistent copy of the VMFS datastores.

pRDMs may utilize the full capabilities of ScaleIO array including initiating such features from inside the guest OS.

Summary

The following summarizes Dell EMC best practices for VMFS datastores and virtualized ASM disks when deployed on ScaleIO:

- If using traditional VMFS datastores, create volumes that use the entire ScaleIO volume.
- Regardless of the VMFS volume/datastore configuration, a minimum of four ASM disks per diskgroup is still recommended to ensure that the guest OS has sufficient I/O queues to provide adequate performance.
- Dedicate one or more datastores for ASM use. Do not mix non ASM files into datastores used for ASM virtual disks.
- Add Scale IO volumes/VMFS volumes to extend datastores. Additional VMFS volumes will provide additional I/O queues to ESX.
- If multiple ScaleIO volumes are used for the ASM diskgroups, create a consistency group if array based snapshots, clones or replication will be used.
- Spread database disks (VMDK or pRDM) evenly across all available SCSI controllers in the VM.
- Use paravirtual (PVSCSI) adapters where available.
- If using RAC, the multi-writer flag must be set in the VMX file. (See VMware KB 1034165), and the VMDK must be selected as Eager Zero Thick.
- If using UDEV rules in Linux on VMware, ensure that the *disk.EnableUUID=true* directive is added to the VMX file of each VM that will access ASM.

Chapter 6 Preparing the Oracle Database for ASM

This chapter presents the following topics:

Overview

The following chapter explores how the Oracle database should be configured to work with ASM on a Dell EMC ScaleIO array.

Database/tablespace block size

The basic unit of storage in any Oracle database is the database block. Before Oracle 9i, the block size was fixed for the entire database at creation time, but with 9i Oracle introduced the ability to use different block sizes for each tablespace.

Small block sizes such as 2K or 4K mean that relatively large amounts of each block are wasted for metadata, such as the block header and tail check.

Large block sizes result in wasted storage and memory, since entire blocks have to be brought into the SGA block cache for a single row requested by a user.

In addition, on RAC systems, larger block sizes increase the likelihood of inter-node block contention where users selecting different rows of data on different RAC nodes, find they both need the same physical block.

Selecting a block size that is too small may also result in ORA-01450 maximum key length exceeded errors, where root index blocks of composite indexes can no longer fit into a single tablespace block.

Block size also affects the total amount of data that an Oracle database can store.

As of Oracle 10g, each data file of the database can store 4 billion blocks, and each database can have up to 65,536 files.

With these limits, the maximum database size is shown for each block size in Table 9.

Table 9. Maximum datafile and database size

Block size	Maximum datafile size	Maximum database size
2 K	8 GB	512 TB
4 K	16 GB	1 PB
8 K	32 GB	2 PB
16 K	64 GB	4 PB
32 K	128 GB	8 PB

With Oracle 10g a new feature was introduced called BIGFILES.

BIGFILE tablespaces allow for one very large datafile per tablespace instead of multiple smaller files. This can simplify management, facilitate larger databases and improve performance since checkpoint operations no longer have to update so many data file headers.

Note that BIGFILE datafiles can be backed up by RMAN using multiple channels simultaneously. The amount of data backed up by each channel is controlled by the RMAN SECTIONSIZE parameter.

This is an example of a BIGFILE tablespace:

```
SQL> create bigfile tablespace my_large_data
  2  datafile '+MYDATA' size 1024G
  3  blocksize 8K;
Tablespace created.
```

With this option, maximum datafile and database sizes increase as shown in Table 10.

Table 10. Maximum datafile and database size

Block size	Maximum datafile size	Maximum database size
2 K	8,192 GB	536 PB
4 K	16,384 GB	1,073 PB
8 K	32,768 GB	2,147 PB
16 K	65,536 GB	4,294 PB
32 K	131,072 GB	8,589 PB

Recommendations

Dell EMC recommends an 8 K block size for OLTP applications including Oracle E-Business Suite and SAP. OLAP and Data Warehouse applications should consider a 16 K or 32 K block size.

Dell EMC recommends the use of BIGFILEs where large amounts of data will be stored.

Mixed block sizes are also recommended where applicable, provided there is enough SGA to create buffer caches for each block size in the database.

Autoextend data files

Autoextend data files have been available in Oracle since Oracle 8. Auto-extend allows a DBA to set a file size to an initial allocation, but allows Oracle to grow the file automatically as objects increase in size.

Autoextend works well with the thin provisioned storage from ScaleIO, and ensures that capacity is not wasted.

However, the operation to grow a data file is relatively intensive, and DBAs should take care to ensure that production data files are not continuously extending. The increment by which a data file grows can be set using the NEXT clause as follows:

```
SQL> alter tablespace USERS add datafile '+DATA' size 1024M
autoextend on next 8M;

Tablespace altered.
```

Note: Dell EMC recommends that the NEXT clause should be an even multiple of the ASM Allocation Unit size.

DBAs should also specify a MAXSIZE to limit the overall growth of the data file. This is especially important where users have some flexibility over the queries or transactions they may execute against the production database:

```
SQL> alter tablespace USERS add datafile '+DATA' size 1024M autoextend on next 8M
maxsize 2048M;
Tablespace altered.
```

Without the MAXSIZE clause, a rogue update or insert statement might generate gigabytes of invalid data before an error is generated.

The MAXSIZE clause is especially important for the TEMP tablespace, where OLAP users may launch complex queries that generate excessive amounts of temporary data.

Autoextend should NOT be used for the UNDO tablespaces of a production database. Setting the UNDO data files to auto-extend mode presents the opportunity for a transaction to fail if the UNDO tablespace is unable to extend due to storage exhaustion.

In this scenario, the UNDO data for the database is now corrupt.

During recovery, Oracle will apply redo data to data files before opening the database to users. It then applies pending UNDO. If that UNDO data is corrupt, the recovered database will crash.

Typically in this scenario, the only option is to restore from the last known good backup.

Oracle Database parameter settings

The following section lists suggested parameter settings to maximize I/O throughput with ASM on Dell EMC ScaleIO storage.

Note: These suggestions are only guidelines. Exact settings will depend on the application workload.

These settings should be set in the INIT.ORA or SPFILE.

archive_lag_target

Default setting: 0, Unit of Measure: Seconds.

The *archive_lag_target* parameter determines the number of seconds allowed to elapse before a redo log switch is forced.

Forcing redo log switches helps to limit the amount of data lost in the event of a failure that destroys on the online redo logs.

Dell EMC recommends setting the *archive_lag_target* to 900 seconds which provides a redo log switch at least every fifteen minutes.

db_block_checksum

Default setting: FALSE (10g), TYPICAL (11g, 12c)

The *db_block_checksum* parameter determines if the DB Writer process will calculate a checksum and store it in the cache header of data blocks when they are written to disk.

If the parameter is not set to OFF, the checksums are verified when the blocks are read.

When the parameter is set to FULL, the database also checks and re-computes the checksums during UPDATE and DELETE operations.

Table 11. db_block_checksum settings

Setting	Performance overhead	Process effect
OFF	None	None
TYPICAL	2%	Checksums generated and checked when blocks are read.
FULL	5%	Checksums generated and checked when blocks are read. Checked and re-generated during UPDATE and DELETE operations.
TRUE	2%	Same as TYPICAL
FALSE	None	Same as OFF

db_block_checking

Default setting: FALSE

The *db_block_checking* parameter controls if the Oracle database will perform block checking for all user data blocks. Block checking is always enabled for SYSTEM blocks, and this parameter will enable it for non-SYSTEM blocks as well.

When enabled, Oracle will verify the integrity of every block of data to protect against memory and storage corruption.

Enabling this option can degrade performance by as much as 15 percent.

Recommendation

With the advance data corruption protection mechanisms built into ScaleIO, Dell EMC recommends setting the *db_block_checking* parameter to FALSE.

db_block_size

Default setting: 8192, Unit of measure: Bytes

The *db_block_size* parameter setting is covered in detail in the previous section.

db_file_multiblock_read_count

Default setting: (10g and 11g: 16, 12c platform dependent) Unit of measure: Database Blocks

The *db_file_multiblock_read_count* (MBRC) parameter is used by Oracle for multi-block I/O operations including full table scans and index fast full scans.

During these multi-block operations, Oracle will request multiple blocks at a time from the storage subsystem.

Since Oracle 10g, the database will determine the optimum value automatically and adjust large block operations accordingly.

db_writer_processes

Default setting: CPU/8, Unit of measure: Number of processes.

The tight integration of ScaleIO with Linux and the power of modern servers, storage devices and networking mean that it is possible that the infrastructure is able to sustain higher write loads that the default *db_writer_processes* parameter allows.

The number of DB Writers can therefore become a bottleneck in write intensive databases. Setting the *db_writer_processes* to CPU/4, or even CPU/2 is acceptable in these scenarios.

disk_asynch_io

Default setting: TRUE

The *disk_asynch_io* parameter enables asynchronous I/O to the storage subsystem. Without this option enabled, all I/O will be synchronous.

Since Oracle ASM by-passes the traditional file system, the *disk_asynch_io* parameter is the only option for controlling synchronous or asynchronous I/O to database files in ASM.

Async I/O offers significant performance benefits to most Oracle databases.

Recommendation

Dell EMC recommends setting *disk_asynch_io* to TRUE where it is supported by the Operating System. Check Oracle documentation to ensure async I/O is supported on your implementation.

fast_start_mttr_target

Default setting: 0, measured in seconds.

The *fast_start_mttr_target* parameter allows the DBA to specify the number of seconds the database takes to perform crash recovery of a single instance.

In effect what this does is to control how aggressively the database DB writers flush dirty blocks from the DB Block Cache to the ASM disks.

When left at the default setting of zero, the DB Writers may not flush dirty blocks from the cache for an extended period of time. Database ACID compliance is guaranteed by the redo logs.

filesystemio_options

Default setting: NONE

The *filesystemio_option*s parameter is used to govern how Oracle interacts with file-based storage.

By manipulating this parameter, the DBA may affect the I/O to the file system as shown in Table 12.

Table 12. Input/output settings

	Buffered I/O	Direct I/O
Synchronous I/O	NONE	DIRECTIO
Asynchronous I/O	ASYNC	SETALL

The parameter still honors the *disk_asynch_io* parameter. If this parameter is set to FALSE, then setting *filesystemio_options* to async will have no effect.

Since ASM bypasses the file system layer and provides Oracle direct access to the ASM files, the *filesystemio_options* parameter has no effect when all files are placed on ASM.

Recommendation

Dell EMC recommends setting *filesystemio_options* to SETALL since some database files, notably RMAN backup sets, may still be generated on file systems not created from ASM volumes.

Database parameter summary

Table 13 lists the parameters and recommended settings.

Table 13. Parameters and recommended settings

Parameter	Recommended setting
archive_lag_target	900
db_block_checksum	FALSE
db_block_checking	FALSE
db_block_size	8192 for OLTP, 16384 for OLAP
db_file_multiblock_read_count	Do not change
db_writer_processes	CPU/4
disk_asynch_io	TRUE
fast_start_mttr_target	120
filesystemio_options	SETALL

Flashback logs

Flashback logs were introduced in Oracle Database 11g.

These optional files allow the database to be rolled backwards in a similar fashion to a redo log allowing a database to be rolled forward.

Although flashback is possible using data in the UNDO tablespace, such operations are limited to the available UNDO extents which are subject to the UNDO_RETENTION target and may be prematurely expired if active transactions are generating undo blocks.

In addition, flashback logs are required if a database is to be rolled forward through a reset logs operation using redo logs from a different database incarnation.

Flashback logs add up to a 20 percent performance penalty due to the additional I/O they generate.

Dell EMC recommends disabling flashback logs unless there is a compelling business case for their use. Most replication and disaster recovery use cases can be handled by array based technologies such as Recover Point.

Chapter 7 Configuration

This chapter presents the following topics:

Database redo log configuration

The database redo log files are amongst the most mission critical of all Oracle files.

The redo log files hold a journal of all changes made to the database. As logs are filled up they are typically archived to the Flash Recovery Area (FRA) to ensure full recoverability.

DBAs therefore frequently seek to place redo log files on the fastest storage available such as the Extreme Performance tier.

Oracle suggests in Metalink document ID 1376916.1 that only Oracle Engineered systems feature flash drives optimized for redo log writes. This statement is untrue and Dell EMC does not discourage the use of the Extreme Performance tier for Oracle redo log files.

Where databases do suffer from redo related performance problems, such as the event Log File Sync frequently appearing in the top five timed events of an AWR report, Dell EMC recommends that database performance analysis be undertaken to determine a precise cause.

In most cases, Log File Sync and Log Buffer Wait are not consistently top events in an AWR report. Where they are, the reason might be I/O related, or might be related other factors such as OS resource limitations or network latency to a Data Guard replicated database.

Excessive application commits or Oracle bugs may also cause excessive Log File Sync events.

Determine that I/O is the cause of the Log File Sync events in the database before undertaking the provisioning of new storage and the relocation of redo log files.

Redo log block size

Prior to Oracle Database11gR2, most platforms used a hard coded 512 byte block size for redo log files, although some platforms used 1 KB or even 2 KB. The block size was a function of the lowest common denominator for supported direct I/O operations on the platform in question.

Since Oracle Database 11.2.0, Oracle now offers user definable redo block sizes if the database is running on an operating system that can understand 4 KB sector sizes such as Windows 8, Windows Server 2012, Red Hat Enterprise Linux kernel 2.6.32 or later or Solaris 11.1.

Flash drives are typically rated using 4 KB random I/O, but ScaleIO presents all devices to Linux as 512-byte, therefore Oracle will fail with an ORA-1377 error if the DBA tries to create a redo log file with a block size other than 512 bytes:

```
SQL> alter database add logfile thread 1 group 5 '+FRA' size 500M blocksize 4096;
alter database add logfile thread 1 group 5 '+FRA' size 50M blocksize 4096
*
ERROR at line 1:
ORA-01378: The logical block size (4096) of file +FRA is not compatible with
the disk sector size (media sector size is 512 and host sector size is 512)
```

While the DBA can override this by setting the *_disk_sector_size_override* parameter, there is little performance benefit to doing so on ScaleIO.

Recommendation

Dell EMC recommends a 512-byte block size for redo log files on ScaleIO.

Redo log file size

Redo log files should normally be able to sustain a minimum of fifteen minutes of transactions before requiring a switch to the next group.

For workloads that experience excessive bursts, it is acceptable for redo log switches to occur more frequently at peak, but excessive redo log switches should be minimized as much as possible.

Log file switches are relatively expensive in database resource terms, since the entire database must be checkpointed including updating the data file headers of every file of the database in addition to the control files.

Dell EMC recommends redo log group members sized between 512 MB to 2 GB for demanding OLTP systems.

If the DBA is concerned that a larger redo log group will result in insufficient switching during non-peak period, the *archive_lag_target* parameter should be set to force periodic switching. This setting will help ensure a smooth recovery in the event of a database failure.

Recommendation

Dell EMC recommends setting the *archive_lag_target* parameter to 900 to force a log switch every fifteen minutes.

Redo log groups and members

An insufficient number of redo log groups may cause *ORA-16014 Log X sequence# Y not archived errors in addition to Checkpoint Not Complete* warnings in the database alert log and will affect overall performance.

Recommendation

Dell EMC recommends a minimum of five redo log groups per thread for any database. Enterprise grade storage arrays largely negate the benefits of multiple redo log group members.

Redo log performance and ASYNCH IO

Redo log data is written using a synchronous process. LGWR must confirm that redo blocks are written to disk before it can acknowledge a user commit.

Exact implementation details differ depending on the platform, but async I/O will still benefit the LGWR process by allowing additional writes to be queued before pending writes complete.

Recommendation

Dell EMC recommends setting *disk_asynch_io* to TRUE if I/O performance is determined to be a contributing factor to Log File Sync being listed as a top timed event.

Controlfile configuration

With the Oracle standard recommendation of two ASM diskgroups – DATA and FRA, many DBAs place their database control files with one file in each of the two diskgroups.

Whereas this approach is suitable for many environments, it can cause performance problems if the FRA diskgroup is located in a storage pool that uses high capacity spinning disks.

Control files should be placed on the high performance storage pools only.

With ScaleIO MESH mirroring it is acceptable to place all control files in the same ASM diskgroup.

Chapter 8 Conclusion

This chapter presents the following topics:

Conclusions

In this document we have explored every relevant aspect of deploying an Oracle database using Oracle ASM with Dell EMC ScaleIO elastic software-defined storage.

By following these guidelines DBAs should be assured of optimal performance and stability combined with the ability to use advanced data services such as ScaleIO snapshots.

When deploying Oracle ASM on ScaleIO, the DBA team should be included in the design process as early as possible to ensure that the final design meets the needs of the DBAs both from a technical and a license management perspective.

Dell EMC has a long history of working closely with Oracle customers and is committed to their ongoing success.

Note: Additional information on Dell EMC and our portfolio of Oracle solutions can be found at the dbasociety page at Dell EMC: http://www.emc.com/dbasociety.

Appendix A UDEV Rules

This appendix presents the following topic:

Shell script for UDEV rules

The following is a korn shell script for generating the UDEV rules to present ScaleIO devices to ASM.

The output of this file should be added to the UDEV rules directory located at */etc/udev/rules.d*

```
#!/bin/sh

# generate UDEV rules for ScaleIO devices suitable for ASM

sd_list=$(ls -1 /dev/scini*? )

let i=0

for sd in ${sd_list}
do
  let i=i+1
  myscid=`/opt/emc/scaleio/sdc/bin/drv_cfg --query_block_device_id --block_device
${sd}`
  printf "KERNEL==\"scini*\", SUBSYSTEM==\"block\",
PROGRAM=\"/opt/emc/scaleio/sdc/bin/drv_cfg --query_block_device_id --block_device
/dev/%%k\", RESULT==\"%s\", SYMLINK+=\"oracleasm/disk%02d\", OWNER=\"grid\",
GROUP=\"asmadmin\", MODE=\"0660\"\n" ${myscid} $i
done
```

The output of this script should look something similar to this:

```
KERNEL=="scini*", SUBSYSTEM=="block", PROGRAM="/opt/emc/scaleio/sdc/bin/drv_cfg --
query_block_device_id --block_device /dev/%k", RESULT=="23719f5a70163008-
fa12df0500000012", SYMLINK+="oracleasm/disk37", OWNER="grid", GROUP="asmadmin",
MODE="0660"
KERNEL=="scini*", SUBSYSTEM=="block", PROGRAM="/opt/emc/scaleio/sdc/bin/drv_cfg --
query_block_device_id --block_device /dev/%k", RESULT=="23719f5a70163008-
fa12df0400000011", SYMLINK+="oracleasm/disk38", OWNER="grid", GROUP="asmadmin",
MODE="0660"
KERNEL=="scini*", SUBSYSTEM=="block", PROGRAM="/opt/emc/scaleio/sdc/bin/drv_cfg --
query_block_device_id --block_device /dev/%k", RESULT=="23719f5a70163008-
fa12df0300000010", SYMLINK+="oracleasm/disk39", OWNER="grid", GROUP="asmadmin",
MODE="0660"
```

Appendix B SQL Scripts

This appendix presents the following topics:

Overview

The following SQL scripts are used with ASM to monitor activity and display disk and diskgroup organization. Each script is shown with its' compatibility, sample output and then the text of the script.

These are designed to be run against the ASM instance as a SYSASM or SYSDBA privileged account.

ATTRIBUTE.SQL

Table 14 includes compatibility information for the attribute.sql script.

Table 14. Script details—attribute.sql

Type of information	Details
Name	attribute.sql
Compatibility	11g, 12c
Purpose	Shows attributes of the ASM instance.

Sample output:

```
NAME                                       VALUE       GN  ATINC RONLY SYS
------------------------------------------ ----------- --- ------ ----- ----
access_control.enabled                     FALSE         1      1 N     Y
access_control.umask                       066           1      1 N     Y
au_size                                    1048576       1      1 Y     Y
cell.smart_scan_capable                    FALSE         1      1 N     N
compatible.asm                             11.2.0.0.0    1      1 N     Y
compatible.rdbms                           10.1.0.0.0    1      1 N     Y
disk_repair_time                           3.6h          1      1 N     Y
sector_size                                512           1      1 Y     Y
template.ARCHIVELOG.mirror_region          0             1      1 N     Y

<output removed to aid clarity>

template.XTRANSPORT.primary_region         0             1      1 N     Y
template.XTRANSPORT.redundancy             17            1      1 N     Y
template.XTRANSPORT.stripe                 0             1      1 N     Y
template_version                           186646528     1      1 N     Y
```

Table 15 contains the key to the output.

Table 15. Key to ATTRIBUTE.SCRIPT columns

Column	Meaning
NAME	Attribute name
VALUE	Attribute value
GN	Group Number

Column	Meaning
ATINC	Attribute Incarnation
RONLY	Read Only
SYS	System Created

Script text:

```
set linesize 132
set pagesize 999

col diskgroup_name a15

col name  for a45
col value for a10
col atidx for 99999
col atinc for 99999
col gn    for 99
col ronly for a5
col sys   for a4

select
  vad.name "DISKGROUP_NAME",
  vaa.name,
  vaa.value,
--  vaa.group_number "GN",
--  vaa.attribute_index "ATIDX",
--  vaa.attribute_incarnation "ATINC",
  vaa.read_only "RONLY",
  vaa.system_created "SYS"
from
  v$asm_attribute vaa,
  v$asm_diskgroup vad
where 1=1
and vaa.group_number = vad.group_number
order by vaa.name
/
```

CLIENTS.SQL

Table 16 includes compatibility information for the clients.sql script.

Table 16. Script details—clients.sql

Type of information	Details
Name	clients.sql
Compatibility	10g, 11g, 12c
Purpose	Shows clients connected to the ASM instance.

Sample output:

```
DB_NAME  INSTANCE DISKGROUP_NAME   STATUS     SOFTWARE_VERSION   COMPATIBLE_VERSION
-------  -------- --------------- ----------- ------------------- --------------
+ASM     +ASM1    CRS             CONNECTED    11.2.0.1.0          11.2.0.1.0
+ASM     +ASM1    DATA            CONNECTED    11.2.0.1.0          11.2.0.1.0
PHLORA   PHLORA1  DATA            CONNECTED    11.2.0.1.0          11.2.0.0.0
PHLORA   PHLORA1  FRA             CONNECTED    11.2.0.1.0          11.2.0.0.0
PHLORA   PHLORA1  REDO            CONNECTED    11.2.0.1.0          11.2.0.0.0
```

Table 17 contains the key to the output.

Table 17. Key to clients.sql columns

Column	Meaning
DB_NAME	Database name of client
INSTANCE	Instance name of client
DISKGROUP_NAME	Diskgroup name
STATUS	Status of the client
SOFTWARE_VERSION	Version of the client
COMPATIBLE_VERSION	Version of the API

Script text:

```
set linesize 132
set pagesize 999

col instance            for a8
col software_version    for a20
col compatible_version  for a20
col diskgroup_name      for a15

select
  vac.db_name,
  vac.instance_name "INSTANCE",
  vad.name "DISKGROUP_NAME",
  vac.status,
  vac.software_version,
  vac.compatible_version
from
  v$asm_client vac,
  v$asm_diskgroup vad
where 1=1
and vac.group_number = vad.group_number
order by 1,2,3
```

DISK11

Table 18 includes compatibility information for the disk11.sql script.

Table 18. Script details—disk11

Type of information	Details
Name	disk11.sql
Compatibility	11g, 12c
Purpose	Shows each disk and the diskgroup it is assigned to.

Sample output:

```
DISK_NAME DISKGROUP_NAME  DSK MNT_STA HDR_STA STATE  OS_SZ TOTAL FREE PATH
--------- -------------- --- ------- ------- ------ ----- ----- ---- -------------
CRS_0000  CRS              0 CACHED  MEMBER  NORMAL    1G    1G   1G /dev/oracleasm/disks/CRSVOL1
CRS_0001  CRS              1 CACHED  MEMBER  NORMAL    1G    1G   1G /dev/oracleasm/disks/CRSVOL2
CRS_0002  CRS              2 CACHED  MEMBER  NORMAL    1G    1G   1G /dev/oracleasm/disks/CRSVOL3
DISK1     DATA             0 CACHED  MEMBER  NORMAL    2G    2G   0G /dev/oracleasm/disks/DATA1
DISK2     DATA             1 CACHED  MEMBER  NORMAL    2G    2G   0G /dev/oracleasm/disks/DATA2
DISK3     DATA             2 CACHED  MEMBER  NORMAL    2G    2G   0G /dev/oracleasm/disks/DATA3
DISK4     DATA             3 CACHED  MEMBER  NORMAL    2G    2G   0G /dev/oracleasm/disks/DATA4
DISK5     DATA             4 CACHED  MEMBER  NORMAL    2G    2G   0G /dev/oracleasm/disks/DATA5
DISK1     FRA              0 CACHED  MEMBER  NORMAL    2G    2G   2G /dev/oracleasm/disks/FRA1
DISK2     FRA              1 CACHED  MEMBER  NORMAL    2G    2G   2G /dev/oracleasm/disks/FRA2
DISK3     FRA              2 CACHED  MEMBER  NORMAL    2G    2G   2G /dev/oracleasm/disks/FRA3
DISK1     REDO             0 CACHED  MEMBER  NORMAL    1G    1G   1G /dev/oracleasm/disks/REDO1
DISK2     REDO             1 CACHED  MEMBER  NORMAL    1G    1G   1G /dev/oracleasm/disks/REDO2
```

Table 19 contains the key to the output:

Table 19. Key to disk11.sql columns

Column	Meaning
DISK_NAME	Name of the ASM disk
DISKGROUP_NAME	Diskgroup name
DSK	Disk number
MNT_STA	Mount status
HDR_STA	Header status
STATE	Device status
OS_SZ	Device size as reported to OS
TOTAL	Total space on the disk (after mirroring)
FREE	Free space on the disk (after mirroring)
PATH	Device path

Script text:

```
set linesize 132
set pagesize 999

col diskgroup_name for a15

col dsk   for 999
col gn    for 99
col hdr_sta for a11
col mnt_sta for a7

col free  for a6
col os_sz for a6
col total for a6

col path  for a30
col raid  for a8
col state for a10

col disk_name for a12

select
  vad.name "DISK_NAME",
  vag.name "DISKGROUP_NAME",
--  vad.group_number "GN",
--  vad.voting_file
  vad.disk_number "DSK",
  vad.mount_status "MNT_STA",
  vad.header_status "HDR_STA",
  vad.state,
--  vad.redundancy "RAID",
  decode(floor(vad.os_mb/1048576),0,
    to_char(vad.os_mb/1024,'9999')||'G',
    to_char(vad.os_mb/1048576,'99.9')||'T'
  ) "OS_SZ",
  decode(floor(vad.total_mb/1048576),0,
    to_char(vad.total_mb/1024,'9999')||'G',
    to_char(vad.total_mb/1048576,'99.9')||'T'
  ) "TOTAL",
  decode(floor(vad.free_mb/1048576),0,
    to_char(vad.free_mb/1024,'9999')||'G',
    to_char(vad.free_mb/1048576,'99.9')||'T'
  ) "FREE",
  vad.path
from
  v$asm_disk vad,
  v$asm_diskgroup vag
where 1=1
and vad.group_number = vag.group_number(+)
```

```
order by vag.name, vad.name
/
```

DISKGROUP

Table 20 includes compatibility information for the diskgroup.sql script.

Table 20. Script details—diskgroup

Type of information	Details
Name	diskgroup.sql
Compatibility	10g, 11g, 12c
Purpose	Shows each diskgroup and its characteristics.

Sample output:

```
GN  DISKGROUP_NAME SEC_SZ BLK_SZ AU   STATE      PROT   TOTAL  FREE
--- -------------- ------ ------ ---- ---------- ------ ------ --
  1 MYDATA         512    4096   1M   CONNECTED  EXTERN  20G   18G
```

Table 21 contains the key to the output:

Table 21. Key to diskgroup.sql columns

Column	Meaning
GN	Diskgroup number
DISKGROUP_NAME	Diskgroup name
SEC_SZ	Sector size in bytes
BLK_SZ	Block size in bytes
AU	Allocation unit size
STATE	State of the diskgroup
PROT	RAID Protection Level – Normal, High or External
TOTAL	Total size of the diskgroup
FREE	Free space on the diskgroup

Script text:

```
col gn              for 99
col diskgroup_name  for a15
col sec_sz          for 99999
col blk_sz          for 99999
col au              for a4
col state           for a10
col type            for a8

col total           for a6
```

```
col free            for a6

select
  vad.group_number "GN",
  vad.name "DISKGROUP_NAME",
  vad.sector_size "SEC_SZ",
  vad.block_size "BLK_SZ",
  decode(floor(vad.allocation_unit_size/1048576),0,
    to_char(vad.allocation_unit_size/1024,'99')||'K',
    to_char(vad.allocation_unit_size/1048576,'99')||'M'
  ) "AU",
  vad.state,
  vad.type "PROT",
  decode(floor(vad.total_mb/1048576),0,
    to_char(vad.total_mb/1024,'9999')||'G',
    to_char(vad.total_mb/1048576,'99.9')||'T'
  ) "TOTAL",
  decode(floor(vad.free_mb/1048576),0,
    to_char(vad.free_mb/1024,'9999')||'G',
    to_char(vad.free_mb/1048576,'99.9')||'T'
  ) "FREE"
from
  v$asm_diskgroup vad
order by vad.group_number
/
```

FILE

Table 22 includes compatibility information for the file.sql script.

Table 22. Script details—file.sql

Type of information	Details
Name	file.sql
Compatibility	10g, 11g, 12c
Purpose	Shows each file known to ASM.

Sample output:

```
FN  DISKGROUP_NAME BLK_SZ STR_SZ SIZE   FILE_TYPE        FILE_NAME
--- -------------- ------ ------ ------ ---------------- -------------------------
253 CRS               512     1M     2K ASMPARAMETERFILE PHL-ORA-cluster/ASMPARAMETERFILE/REGISTRY.
255 CRS              4096     1M   260M OCRFILE          PHL-ORA-cluster/OCRFILE/REGISTRY.255.77896
256 DATA             8192     1M   680M DATAFILE         PHLORA/DATAFILE/SYSTEM.256.779189873
256 FRA             16384   128K    18M CONTROLFILE      PHLORA/CONTROLFILE/Current.256.779190159
256 REDO            16384   128K    18M CONTROLFILE      PHLORA/CONTROLFILE/Current.256.779190159
257 DATA             8192     1M   720M DATAFILE         PHLORA/DATAFILE/SYSAUX.257.779189875
257 FRA               512     1M    50M ONLINELOG        PHLORA/ONLINELOG/group_1.257.779190177
257 REDO              512     1M    50M ONLINELOG        PHLORA/ONLINELOG/group_1.257.779190177
```

```
258 DATA             8192      1M    45M DATAFILE         PHLORA/DATAFILE/UNDOTBS1.258.779189877
258 FRA               512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_2.258.779190179
258 REDO              512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_2.258.779190177
259 DATA             8192      1M     1G DATAFILE         PHLORA/DATAFILE/USERS.259.779189877
259 FRA               512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_3.259.779190507
259 REDO              512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_3.259.779190505
260 DATA             8192      1M    20M TEMPFILE         PHLORA/TEMPFILE/TEMP.260.779190213
260 FRA               512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_4.260.779190507
260 REDO              512      1M    50M ONLINELOG        PHLORA/ONLINELOG/group_4.260.779190507
261 DATA             8192      1M   100M DATAFILE         PHLORA/DATAFILE/EXAMPLE.261.779190219
262 DATA             8192      1M   400M DATAFILE         PHLORA/DATAFILE/UNDOTBS2.262.779190451
263 DATA              512      1M     5K PARAMETERFILE    PHLORA/PARAMETERFILE/spfile.263.779190507
264 DATA             8192      1M    20M TEMPFILE         PHLORA/TEMPFILE/TEMP.264.780252785
```

Table 23 contains the key to the output.

Table 23. Key to file.sql columns

Column	Meaning
GN	Diskgroup number
DISKGROUP_NAME	Diskgroup name
SEC_SZ	Sector size in bytes
BLK_SZ	Block size in bytes
AU	Allocation unit size
STATE	State of the diskgroup
PROT	RAID Protection Level – Normal, High or External
TOTAL	Total size of the diskgroup
FREE	Free space on the diskgroup

Script text:

```
set linesize 132
set pagesize 999

col gn              for 99
col fn              for 99999
col blk_sz          for 99999
col str_sz          for a6
col file_type       for a16

col size            for a8
col tot             for a6
col free            for a6

col file_name       for a60
col diskgroup_name for a15
```

```
select
  vaf.file_number "FN",
  vad.name "DISKGROUP_NAME",
--  vaf.group_number "GN",
  vaf.block_size "BLK_SZ",
  decode(vaf.striped,'COARSE',stripe_size.extent,stripe_size.stripsz)
"STR_SZ",
  decode(floor(vaf.bytes/1099511627776),0,
    decode(floor(vaf.bytes/1073741824),0,
      decode(floor(vaf.bytes/1048576),0,
        to_char(vaf.bytes/1024,'9999')||'K',
        to_char(vaf.bytes/1048576,'9999')||'M'
      ),
      to_char(vaf.bytes/1073741824,'9999')||'G'
    ),
    to_char(vaf.bytes/1099511627776,'99.9')||'T'
  ) "SIZE",
  vaf.type "FILE_TYPE",
  vaa3.name||'/'||vaa2.name||'/'||vaa1.name "FILE_NAME"
from
  v$asm_diskgroup vad,
  v$asm_file vaf,
  v$asm_alias vaa1,
  v$asm_alias vaa2,
  v$asm_alias vaa3,
  (
  select
    decode(floor(y1.ksppstvl/1048576),0,
      to_char(y1.ksppstvl/1024,'9999')||'K',
      to_char(y1.ksppstvl/1048576,'9999')||'M'
    ) "STRIPSZ",
    decode(floor(y2.ksppstvl/1048576),0,
        to_char(y2.ksppstvl/1024,'9999')||'K',
        to_char(y2.ksppstvl/1048576,'9999')||'M'
    ) "EXTENT",
    y1.ksppstvl,
    y2.ksppstvl
  from
    x$ksppcv y1,
    x$ksppi x1,
    x$ksppcv y2,
    x$ksppi x2
  where 1=1
  and x1.indx = y1.indx
  and x1.ksppinm ='_asm_stripesize'
  and x2.indx = y2.indx
  and x2.ksppinm ='_asm_ausize'
  ) stripe_size
where 1=1
and vaf.group_number = vad.group_number
and vaf.group_number = vaa1.group_number
and vaf.file_number = vaa1.file_number
and vaf.incarnation = vaa1.file_incarnation
```

```
and vaa1.parent_index = vaa2.reference_index
and vaa2.parent_index = vaa3.reference_index
order by
  vaf.file_number,
  vaf.group_number
/
```

OPERATION11

Table 24 includes compatibility information for the operation11.sql script.

Table 24. Script details—operation11.sql

Type of information	Details
Name	operation11.sql
Compatibility	11g, 12c
Purpose	Shows currently running operations in ASM.

Sample output:

```
SQL> @operation11

DISKGROUP_NAME  OPERATION          STATE      PWR  ACTUAL PCT_DONE EST_MIN ERROR_CODE
--------------- ------------------- -------- ---- ------- -------- ------- ------
DATA             REBAL              RUN        6      6     5.1%       1
```

Table 25 contains the key to the output.

Table 25. Key to operation11.sql columns

Column	Meaning
DISKGROUP_NAME	Diskgroup name
OPERATION	Sector size in bytes
STATE	Current state of the operation
PWR	Requested power level
PCT_DONE	Estimated percentage complete
EST_MIN	Estimated minutes to completion
ERROR_CODE	Any error codes reported while performing the operation

Script text:

```
set linesize 132
set pagesize 999

col diskgroup_name for a15

col gn    for 99

col operation   for a20
col state       for a8
col pwr         for 999
col actual      for 999999
col sofar       for 999999

col est_work    for 999999
col est_min     for 999999
col pct_done    for a8

col error_code  for a10

select
  vag.name "DISKGROUP_NAME",
--  vao.group_number "GN",
  vao.operation,
  vao.state,
  vao.power "PWR",
  vao.actual,
  lpad(to_char(100*(vao.sofar/vao.est_work),'999.9')||'%',8) "PCT_DONE",
--  vao.sofar,
--  vao.est_work,
  vao.est_minutes "EST_MIN",
  vao.error_code
from
  v$asm_operation vao,
  v$asm_diskgroup vag
where 1=1
and vag.group_number = vao.group_number
/
```