EXAM OBJECTIVE: DESCRIBE AUTOMATIC STORAGE MANAGEMENT

When you receive a new server a very important function involves creating the storage system layout on it. The tasks associated with storage management can be quite complicated and the Oracle Database 10g has a new and exciting feature the Automatic Storage Management (ASM) that lets DBAs execute many of the above tasks completely within the Oracle framework.

Using ASM you can transform a bunch of disks to a highly scalable filesystem/volume manager using the Oracle 10g software at no additional cost. To give you an example of this capability, let us say we have 10 disks that need to be used for the Oracle database. With ASM, you don't have to create anything on the Operating System side; the feature will group a set of physical disks to a logical entity known as a **diskgroup**. It is important to note that this diskgroup is not a general-purpose filesystem for storing user files and it is not buffered.

A database can have a combination of ASM files, OMF files and manually managed files (RAW), all at the same time. You can move existing files to ASM.

ASM ARCHITECTURE

The primary component of ASM is the diskgroups. These diskgroups for the database indicate the default location for files created in the database.

The disk group consists of grouping of disks that are managed as a unit. These disks are referred to as ASM disks. I/Os to the disks in a disk group are balanced across all the disks in the group. Disks in a disk group should have similar size and performance characteristics to obtain optimal I/O tuning.

Files are created on the ASM disks and are known as ASM files. File names are generated automatically by ASM. User-friendly alias names can be created for these ASM files; however you need to create hierarchical directory structure for the alias names. ASM files are not visible to the OS or its utilities, but are visible to RMAN and other Oracle supplied tools. File activity is always balanced by uniformly distributing the file extents across all disks in a disk group.

An allocation unit (AU) is the smallest contiguous disk space that ASM allocates. The AU size is not user configurable and is typically sized at one megabyte. ASM does not allow physical blocks to be split across allocation units.

You can affect how ASM places files on disks by creating failure groups. Failure groups define disks that share components, such that if one fails then another disk sharing the components might also fail. Also failure groups are used to determine

which ASM disks to use for sharing redundant data. The functionality of ASM can be summarized as:

- It manages groups of disks, called the disk group disk groups.
- It protects the data within the disk group using mechanisms such as mirroring and striping.
- It provides near optimal I/O balancing without any manual tuning.
- It enables the user to manage database object such as tablespaces without needing to specify and track file names.
- It supports large files.

For ASM to operate you need to configure an ASM instances. The ASM instance contains the metadata needed to make ASM files available to database instances. Both ASM instances and database instances have access to a common set of disks called diskgroups. Database instances communicate with the ASM instance only to obtain information about the layout of these files.

An ASM instance contains two types of background processes.

- The **RBAL** background process is responsible for coordinating rebalance activity for disk groups.
- The **ARB*n***, where *n* = 0, 1, 2 and so on.. Performs data extent movements.

The database instance using ASM has two new background processes called ASMB and RBAL.

- RBAL performs global opens to the disks in the disk groups in the database instance.
- ASMB runs in database instances and connects to foreground processes in ASM instances.

Database instances only communicate with ASM instances on the same node.

A diskgroup can contain files for many different Oracle databases. Thus multiple database instances serving different databases can access the same disk group even on a single system without RAC. Alternatively one Oracle database may store its files in multiple disk groups managed by the same ASM instance.

The failure of a database instance does not affect ASM instances.

Database backups of ASM files must be made with RMAN, since each file in a disk group is physically spread across all disks in a disk group, a backup of a single disk is not useful.

RETREIVING DATA DICTIONARY INFORMATION ABOUT ASM

View	Description
V$ASM_DISKGROUP	In an ASM instance, describes a disk group (number, name, size related info, state, and redundancy type). In a DB instance, contains one row for every ASM disk group mounted by the local ASM instance.
V$ASM_CLIENT	In an ASM instance, identifies databases using disk groups managed by the ASM instance. In a DB instance, contains one row for the ASM instance if the database has any open ASM files.
V$ASM_DISK	In an ASM instance, contains one row for every disk discovered by the ASM instance, including disks that are not part of any disk group. In a DB instance, contains rows only for disks in the disk groups in use by that DB instance.
V$ASM_FILE	In an ASM instance, contains one row for every ASM file in every disk group mounted by the ASM instance. In a DB instance, contains no rows.
V$ASM_TEMPLATE	In an ASM instance, contains one row for every template present in every disk group mounted by the ASM instance. In a DB instance, contains no rows
V$ASM_ALIAS	In an ASM instance, contains one row for every alias present in every disk group mounted by the ASM instance. In a DB instance, contains no rows.
V$ASM_OPERATION	In an ASM instance, contains one row for every active ASM long running operation executing in the ASM instance. In a DB instance, contains no rows.

EXAM OBJECTIVE: SET UP INITIALIZATION PARAMETER FILES FOR ASM AND DATABASE INSTANCES.

An ASM instance is started like a database instance. Initialization parameters used for an ASM instance include:

ASM INSTANCE INITIALIZATION PARAMETERS

- **INSTANCE_TYPE** – should be set to **ASM** for ASM instances.
  Example: INSTANCE_TYPE=ASM

- **DB_UNIQUE_NAME** – specifies the service provider name for which this ASM instance manages disk groups. The default value of **+ASM** should be valid in most cases.
  Example: DB_UNIQUE_NAME=+ASM

- **ASM_POWER_LIMIT** – controls the speed for a rebalance operation. Possible values range from 1 to 11 where11 is the fastest. The default value is 1. Example: ASM_POWER_LIMIT=1

- **ASM_DISKSTRING** – is an operating system-dependent value used by ASM to limit the set of disks considered for discovery. When a new disk is added to a disk group, each ASM instance that has the disk group mounted must be able to discover the new disk using its ASM_DISKSTRING. If not specified, it is assumed to be NULL and ASM disk discovery finds all disks to which the ASM instance has read and write access.
  Example: ASM_DISKSTRING = '/dev/rdsk/*s2' , '/dev/rdsk/c1*'

- **ASM_DISKGROUPS** – is a list of name of disk groups to be mounted by an ASM instance at startup, or when the ALTER DISKGROUP ALL MOUNT command is used.

The internal packages used by ASM instances are executed from the LARGE_POOL. Therefore set the value of the LARGE_POOL_SIZE to a value greater than 8MB.

DATABASE INSTANCE PARAMETER CHANGES

The following parameters need to be changed.

INSTANCE_TYPE – specifies that this instance is an RDBMS instance. It defaults to RDBMS.

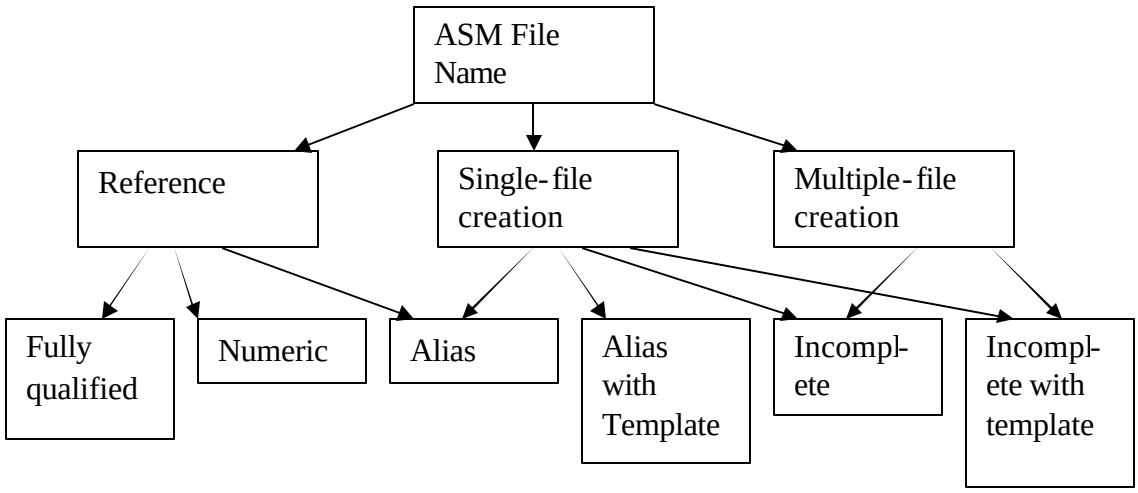
LOG_ARCHIVE_FORMAT is ignored if LOG_ARCHIVE_DEST is set to an incomplete ASM filename such as +dgroupA. If LOG_ARCHIVE_DEST is set to an ASM directory, for example +dgroupA/myarchlogdir/, then LOG_ARCHIVE_FORMAT is used, and the files are non-OMF. Unique file names are created for the archive logs automatically by the Oracle database.

EXAM OBJECTIVE: EXECUTE SQL COMMANDS WITH ASM FILE NAMES

ASM Filenames

ASM filenames can take several forms:

- Fully Qualified
- Numeric
- Alias
- Alias with Template
- Incomplete
- Incomplete with template



1. Fully qualified :

   - Used for referencing existing ASM files.
   - Specify a disk group name, a database name, a file type, a type specific tag (like a tablespace name), a file number and incarnation number (to ensure uniqueness).

   A fully qualified name has the form:
   +<group>/<dbname>/<filetype>/<tag>.<file>.<incarnation>

   <*group*> - is the disk group name
   <*dbname*> - is the database name to which the file belongs
   <*file type*> is the Oracle file type, one of CONTROLFILE, DATAFILE and so on.
   <*tag*> - is type specific information about the file, like the tablespace name for a datafile.
   <*file*>.<*incarnation*> - is the file/incarnation number pair used to ensure uniqueness.

2. Numeric ASM filenames :

   - Are used for referencing existing ASM files.

- They specify a disk group name, a file number, and an incarnation number. They are derived from a fully qualified name.

An example of a numeric ASM file name is : +dgroupA.257.8675309

3. Alias ASM File name :
   - Alias names specify a disk group name, and a user-friendly name string.
   - Alias ASM names are distinguished from fully qualified or numeric names because they do not end in a dotted pair of numbers.
   - Alias names are implemented using a hierarchical directory structure, with the slash (/) character separating name components.

   Examples of Alias ASM file names are:

   +dgroupA/myfiles/control_file1
   +dgroupA/ What a long name /for a file

   An alias can be created for each file during creation or using the ALTER DISKGROUP ADD ALIAS command. An example is shown below:

```
SQL> ALTER DISKGROUP dgroup1
       ADD ALIAS '+dgroup1/mydir/second.dbf'
       FOR '+dgroupA/sample/datafile/mytable.342.3';
```

4. Alias ASM File Names with Templates :
   - Are used only for ASM file creation operations.
   - They specify a disk group name, an alias name, and a file creation template name.
   - If an alias ASM filename with template is specified, and the alias portion refers to an existing file, then the template specification is ignored.

   An example is:
   +dgroupA/config1(parameterfile)

5. Incomplete ASM file names :

   - Used only for file creation operations.
   - They consist of a disk group name only.

   An example of an incomplete ASM file name is : +dgroupA

6. Incomplete ASM filenames with templates :

Templates are used to provide the attribute information about ASM files created in ASM disk groups. These templates simplify file creation by mapping complex file attribute specifications into a single name. For example, a template

named ONLINELOG provides the file redundancy and striping attributes for all redo log files written to ASM disks.

- Used only for file creation operations.
- They consist of a disk group name followed by a template name.

An example of an incomplete ASM file name with templates is :
+dgroupA(datafile)

ASM filenames are accepted in SQL commands wherever file names are legal. For most commands there is an alternate method for identifying the file so that the file name need not be typed. An example could be a file number. However you are discouraged from using ASM file names as much as possible. In the example datafiles stored in an ASM disk group should be given to the CREATE CONTROLFILE command using the file reference context form. The use of the RESETLOGS option requires the use of the file creation context form for the specification of log files.

```
CREATE CONTROLFILE DATABASE sample
LOGFILE GROUP 1 ('+dgroupA','+dgroupB') SIZE 100M,
        GROUP 2 ('+dgroupA','+dgroupB') SIZE 100M
RESETLOGS
DATAFILE '+dgroupA.261.12345678' SIZE 100M
DATAFILE '+dgroupA.262.87654321' SIZE 100M
MAXLOGFILES 5
MAXLOGHISTORY 100
MAXDATAFILES 10
MAXINSTANCES 2
ARCHIVELOG;
```

EXAM OBJECTIVE: STARTUP AND SHUTDOWN ASM INSTANCES

STARTING AN ASM INSTANCE

ASM instances are started similarly to Oracle Database instances with some minor differences. These are:

- The initialization parameter file or spfile must contain:
  INSTANCE_TYPE=ASM
- For ASM instances, STARTUP tries to mount the disk groups specified by the ASM_DISKGROUPS initialization parameter.
- The SQL*Plus STARTUP command parameters are interpreted by ASM as given below:
-

STARTUP Parameter	Description
FORCE	Issues a SHUTDOWN ABORT to the ASM instance before restarting it
MOUNT	Mounts the disk groups specified in the ASM_DISKGROUPS initialization parameter
NOMOUNT	Starts up the ASM instance without mounting any disk groups
OPEN	Invalid for an ASM instance

- The following is a sample SQL*Plus session where an ASM instance is started:

```
% sqlplus /nolog
SQL> CONNECT / AS sysdba
Connected to an idle instance.
SQL> STARTUP
ASM instance started
Total System Global Area 147936196 bytes
Fixed Size 324548 bytes
Variable Size 96468992 bytes
Database Buffers 50331648 bytes
Redo Buffers 811008 bytes
ASM diskgroups mounted
```

SHUTTING DOWN AN ASM INSTANCE

Upon receiving the shutdown command, the ASM instance forwards the shutdown command with the same shutdown mode (NORMAL, IMMEDIATE, TRANSACTIONAL) to all database instances dependent on the ASM instance.
In a NORMAL shutdown, ASM waits for the connected ASM instances to exit before shutting down the instance. In an IMMEDIATE shutdown, ASM waits for any-in-progress SQL to complete before shutting down the ASM instance, but does not wait for database instances to disconnect. TRANSACTIONAL is the same as IMMEDIATE. Except for the case of SHUTDOWN ABORT issued to ASM, the ASM instance waits for all dependent databases to complete their shutdown before ASM shuts down. In the case of a SHUTDOWN ABORT it will immediately terminate any open connections, and all dependent databases will immediately abort as a consequence.
Using SQL*Plus, Automatic Storage Management shutdown is initiated by issuing the `SHUTDOWN` command. For example:

```
% sqlplus /nolog
SQL> CONNECT / AS sysdba
Connected.
SQL> SHUTDOWN NORMAL
```

EXAM OBJECTIVE: ADMINISTER ASM DISK GROUPS

In order to administer ASM disk groups you require connecting with SYSDBA privileges, and issuing them from an ASM instance.

CREATING DISK GROUPS

Assume the ASM disk discovery identified the following disks in the /devices directory.
/devices/diskA1
/devices/diskA2
/devices/diskA3
/devices/diskA4
/devices/diskB1
/devices/diskB2
/devices/diskB3
/devices/diskB4
/devices/diskA1
Assume disks A1, A2, A3, A4 are on separate SCSI controller from the disks B1, B2, B3, B4.

The example below shows how to set up a disk group called DGROUPA with two failure groups CONTROLLER1 and CONTROLLER2.

The disk group is defined with NORMAL REDUNDANCY. With this redundancy level, and using the default template, ASM can tolerate the loss of a single failure group without data loss. Disk groups that use NORMAL REDUNDANCY must contain at least two failure groups.

The keywords NORMAL REDUNDANCY result in two-way mirroring of all files on the disk group. You can specify EXTERNAL REDUNDANCY if your disks each map to a RAID array managed externally or if you do not care to mirror the files at all. You can be extra cautious and specify HIGH REDUNDANCY to get three-way mirroring.

As shown in the example, you can provide an optional disk name. If not supplied, ASM creates a default name of the form <group>_n where <group> is the disk group name and *n* is the disk number. Optionally you can also provide the size for the disk.

The FORCE option indicates that the specified disk should be added to the specified disk group even though the disk is already formatted as a member of an ASM disk group.

```
CREATE DISKGROUP dgroupA NORMAL REDUNDANCY
FAILGROUP controller1 DISK
'/devices/diskA1' NAME diskA1 SIZE 120G FORCE,
'/devices/diskA2',
'/devices/diskA3'
FAILGROUP controller2 DISK
'/devices/diskB1',
'/devices/diskB2',
'/devices/diskB3'
```

DELETING DISK GROUPS

You can delete a disk group along with all its files. To avoid accidental deletions, the INCLUDING CONTENTS option must be specified in the disk group still contains any files besides the internal ASM metadata.
The disk group must be mounted. After ensuring that none of the disk groups are open the group and all its drives are removed from the disk group. Then the header of each disk is overwritten to eliminate the ASM formatting information.

```
DROP DISKGROUP dgroupA INCLUDING CONTENTS;
```

ADDING DISKS TO DISK GROUPS

You can use the ALTER DISKGGROUP ADD DISK command to add disks to a disk group. When a disk is added to a disk group, the ASM instance ensures the disk is addressable and usable. The disk is formatted and then rebalanced. During rebalancing extents from every file is moved onto the new disk. During rebalancing database operations are not blocked but there is a high impact on I/O performance. In the example shown below, four new disks are added to disk group DGROUPA.

```
ALTER DISKGROUP DGROUPA ADD DISK
'/dev/rdsk/c0t4d0s2' NAME diskA5,
'/dev/rdsk/c0t5d0s2' NAME diskA6,
'/dev/rdsk/c0t6d0s2' NAME diskA7,
'/dev/rdsk/c0t7d0s2' NAME diskA8;
```

ALTERING A DISK GROUP

- Removing a disk from its disk group.

```
ALTER DISKGROUP DGROUPA DROP DISK diskA5;
```

- Adding and dropping a disk in a single command.

```
ALTER DISKGROUP dgroupA
DROP DISK diskA6
```

```
ADD FAILGROUP fred
      DISK '/dev/rdsk/c0t8d0s2' NAME diskA9;
```

- Cancel the drop of a disk. The UNDROP works only on pending drops of a disk.

```
ALTER DISKGROUP dgroupA UNDROP DISKS;
```

- You can rebalance a disk, to improve load balancing of disks. This command is not usually necessary.

```
ALTER DISKGROUP dgroupB REBALANCE POWER 5;
```

- Dismounting a disk group can be used to make a disk group unavailable to the database instances running on the same node as the ASM instance. MOUNT can be used to make it once again available.

```
ALTER DISKGROUP dgroupA DISMOUNT;
```

- Adding a new template to a disk group. In this example the RELIABLE template is created in disk group DGROUPA, which is two way mirrored.

```
ALTER DISKGROUP dgroupA
      ADD TEMPLATE reliable ATTRIBUTES (MIRROR);
```

- Removing a previously defined template.

```
ALTER DISKGROUP dgroupA
      DROP TEMPLATE reliable;
```

- Removing a file from a disk group.

```
ALTER DISKGROUP dgroupA
      DROP FILE '+dgroupA.268.8675309';
```

- The statement below creates a directory called MYDIR. Following this is the creation of an alias for the file +dgroupA.274.38745.

```
ALTER DISKGROUP dgroupA
ADD DIRECTORY '+dgroupA/mydir';

ALTER DISKGROUP dgroupA
ADD ALIAS '+dgroupA/mydir/datafile.dbf'
FOR '+dgroupA.274.38745';
```

- To delete the alias you would issue a command like:

```
ALTER DISKGROUP dgroupA
DROP ALIAS '+dgroupA/mydir/datafile.dbf';
```

- To verify the internal consistency of disk group metadata, you can use the CHECK option. In the example we are performing a check across all disks. If an error is found, a summary error message is displayed, and the details of the error are written to the alert log.

ALTER DISKGROUP dgroupA CHECK ALL;

- Deleting a directory along with its contents:

ALTER DISKGROUP dgroupA
DROP DIRECTORY '+dgroupA/mydir' FORCE;

- Renaming a directory

ALTER DISKGROUP dgroupA
RENAME DIRECTORY '+dgroupA/mydir'
TO '+dgroupA/yourdir';

To prevent database instances from connecting to an ASM instance you can issue the ALTER SYSTEM ENABLE RESTRICTED SESSION command to an ASM instance. To enable database instances to connect you can issue ALER SYSTEM DISABLE RESTRICTED SESSION.

EXAM OBJECTIVE: USE RMAN TO MIGRATE YOUR DATABASE TO ASM

This migration is performed using Recovery Manager (RMAN) even if you are not using RMAN for your primary backup and recovery strategy.

Preparing to migrate a database to ASM:

- Determine the DBID for the database. The easiest way to find out your DBID is to connect the RMAN client to the database to be migrated. RMAN displays the DBID whenever it starts up.

```
% rman TARGET /
Recovery Manager: Release 10.1.0.2.0 - Production

Copyright (c) 1995, 2003, Oracle.  All rights reserved.

connected to target database: RDBMS (DBID=774627068)

RMAN> exit
```

- Obtain the names of all the controlfiles, datafiles, and online redo logs for your database.

MIGRATION PROCESS

Given below are the basic steps to perform the migration. To get a detailed procedure please use the Oracle supplied documentation.

1. Disable change tracking by issuing:

SQL> ALTER DATABASE DISABLE BLOCK CHANGE TRACKING;

2. Shudown the database consistently.

SQL> SHUTDOWN IMMEDIATE;

3. Modify the initialization parameter file of the target database as follows:
   - Set DB_CREATE_FILE_DEST and DB_CREATE_ONLINE_LOG_DEST_n (where n=1-4) to refer to the desired ASM disk groups.
   - If the database uses a server parameter file (SPFILE), remove the CONTROL_FILES parameter that specifies the locations of the control file. The control file will be moved to the DB_CREATE_* destination and the server parameter file will be automatically updated. If you are using a client-side parameter file (PFILE) then set the CONTROL_FILES parameter to the ASM alias for the control file name. For example CONTROL_FILE=+disk_group/cf1.

4. Startup the database in nomount mode

```
RMAN> STARTUP NOMOUNT;
```

5. Restore the control file into new locations from location specified in the old SPFILE or PFILE.

```
RMAN> RESTORE CONTROLFILE
FROM 'filename_of_old_control_file';
```

6. Mount the database.

```
RMAN> ALTER DATABASE MOUNT;
```

7. Copy the database into the ASM disk group using the command :

```
RMAN> BACKUP AS COPY DATABASE FORMAT '+disk_group';
```

8. Switch all datafiles into new ASM disk group. This will result in setting all datafiles to ASM.

RMAN> SWITCH DATABASE TO COPY;

9. Open the database.
   RMAN> ALTER DATABASE OPEN;

If you were using change tracking for incremental backups you can re-enable it now:
SQL> ALTER DATABASE ENABLE BLOCK CHANGE TRACKING;

Cleanup after ASM migration:

You can delete the old database files to free disk space.
RMAN> DELETE COPY OF DATABASE;
RMAN> HOST 'rm old_online_redo_logs';
RMAN> HOST 'rm old_control_files';