ASM FULL DOCUMENTATIONS

Adding an ASM Disk Group

ASM disk groups are the foundation of ASM. It is where your data will be stored. ASM disk groups can be allocated to raw disk, cooked disk. To add a ASM disk group you first need to set the ASM_DISKSTRING parameter. You can then add disk groups to your ASM instance.

Setting the ASM_DISKSTRING Parameter

The parameter file for the ASM instance should contain a parameter called ASM_DISKSTRING (see Creating the ASM Instance). This parameter contains the paths that Oracle will use to try to find the various candidate disks available for ASM's use. The process of ASM finding disks in the ASM_DISKSTRING path is known as discovery.

You may not need to set ASM_DISKSTRING. ASM_DISKSTRING has a number of different default values depending on the platform you are using. Here is a table that contains the list of platform specific default values (these will be set if ASM_DISKSTRING is set to a NULL value only):

Platform Name	Default ASM_DISKSTRING value
AIX	/dev/rhdisk*
HP-UX	/dev/rdsk/*
Linux	/dev/raw/*
Mac OS X	/dev/rdisk*s*s1
Solaris	/dev/rdsk/*
Tru64UNIX	/dev/rdisk/*

You can have multiple locations in the ASM_DISKSTRING parameter (we will provide an example of this in just a moment). If you insert a ? placeholder at the beginning of the string, Oracle will expand that out to represent the location of ORACLE_HOME.in the parameter values. The ASM_DISKSTRING can be dynamically altered, which is nice if your friendly system administrator adds a some storage to your system that you want Oracle to be able to use. If you happen to change ASM_DISKSTRING dynamically and the new disk path is not present it will revert to the old disk path. Removing an existing disk path, when that disk path is in use, will result in a failure of the command.

Another thing to consider when determining how to configure the ASM_DISKSTRING parameter is performance. Leaving this parameter set to NULL, and thus taking the Oracle default, will often be sufficient. However, if you set ASM_DISKSTRING, using a more restrictive set of parameters, you may find that discovery of disks will be faster. For example, using the default Linux setting of /dev/raw/* will result in ASM scanning the entire /dev/raw file system structure (it does not search sub folers). If you have a large number of devices in this structure, this may take some time. If however your disk devices in this structure are all prefixed with the work raw (raw1, raw2, raw3, etc) then setting the ASM_DISKSTRING to '/dev/raw/raw*' could reduce the time it take ASM to perform discovery and improve performance of the startup of the ASM instance.

Something that is common to all ASM_DISKSTRING parameters is the use of the asterisk. The asterisk is required when defining the ASM_DISKSTRING parameter. Here are some examples of setting the ASM_DISKSTRING parameter.

In this example, we will point ASM to look for disks in devices when we create disk groups:

```
Alter system set ASM_DISKSTRING='/devices/*';
```

In this example, we are pointing ASM_DISKSTRING to ORACLE_HOME/disks:

```
Alter system set ASM_DISKSTRING='?/disks/*';
```

In this example we are pointing ASM_DISKSTRING to two different locations:

```
Alter system set ASM_DISKSTRING='?/disks/d1/*,?/disks/d21/*';
```

We could also use some adjunctive regular expressionish type extensions, and perform the allocation this way:

```
Alter system set ASM_DISKSTRING='?/disks/d[12]/*';
```

ASM Disk Group Attributes

In Oracle 10g and earlier you had to use templates to assign attributes to ASM disk groups. Oracle 11g introduced the new ASM attribute clause. The attribute clause allows you to assign attributes directly to ASM disk groups without having to use templates. Oracle 11g adds new attributes that you can assign via a template or using the attribute clause. Both the CREATE DISKGROUP and the ALTER DISKGROUP commands allow you to define or modify these settings as required. The following table shows a list of the attributes you can set with the new attribute clause.

Attribute	Description
au size	This attribute defines the allocation unit size. This parameter defaults to 1MB. The au attribute can only be set when the disk group is created. The value of AU can be any power of 2 (1,2,4, 8, etc..) from 1M to 64M. For example, 4M would be 4194304.
compatible.rdbms	This defines the compatibility setting of the disk group with respect to the database. The default value is 10.1. This value cannot be rolled back to a previous, lower version setting.
compatible.asm	This defines the format of the data on the ASM disks with respect to a given database version. Example '11.0'. The default value is 10.1.

disk_repair_time	Enables the use of fast disk resync. Valid values are from 0 to 136 years. Support for this attribute is only available if both COMPATIBLE attributes (COMPATIBLE.RDBMS and COMPATIBLE.ASM) are set to 11.1 or higher.

You can query the V$ASM_ATTRIBUTE view to see the individual attributes assigned to a given disk group. Here is an example:

```
SELECT group_number, name, value
FROM v$asm_attribute
ORDER BY group_number, name;
GROUP_NUMBER NAME                 VALUE
------------ -------------------- --------------------
           1 au_size              1048576
           1 compatible.asm       11.1.0.0.0
           1 compatible.rdbms     11.1.0
           1 disk_repair_time     300M
```

ASM Disk Discovery

When the ASM instance is started, it will use the paths listed in the ASM_DISKSTRING parameter and discover the disks that are available. These disks can then be added to ASM disk groups which we will discuss in the next section. Once discover is complete and the ASM instance is open, you can review the disks discovered by looking at the V$ASM_DISK view as seen in this example:

```
column path format a20
set lines 132
set pages 50

select path, group_number group_#, disk_number disk_#, mount_status,
       header_status, state, total_mb, free_mb
from v$asm_disk
order by group_number;

PATH           GROUP_# DISK_# MOUNT_S HEADER_STATU STATE      TOTAL_MB
FREE_MB
-------------- ------- ------ ------- ------------ -------- ---------- ------
----
/dev/raw/raw4        0      1 CLOSED  FOREIGN      NORMAL           39
0
/dev/raw/raw5        0      0 CLOSED  FOREIGN      NORMAL           39
0
/dev/raw/raw3        0      2 CLOSED  FOREIGN      NORMAL           39
0
/dev/raw/raw6        0      2 CLOSED  CANIDATE     NORMAL         2048
2048
ORCL:ASM01_004       1      3 CACHED  MEMBER       NORMAL        34212
30436
ORCL:ASM01_005       1      4 CACHED  MEMBER       NORMAL        34212
30408
ORCL:ASM01_006       1      5 CACHED  MEMBER       NORMAL        34212
30420
```

```
ORCL:ASM01_007       1      6 CACHED  MEMBER       NORMAL        34212
30297
ORCL:ASM01_008       1      7 CACHED  MEMBER       NORMAL        34212
30507
ORCL:ASM01_009       1      8 CACHED  MEMBER       NORMAL        34212
30404
ORCL:ASM01_010       1      9 CACHED  MEMBER       NORMAL        34212
30509
ORCL:ASM01_011       1     10 CACHED  MEMBER       NORMAL        34212
30449
ORCL:ASM01_012       1     11 CACHED  MEMBER       NORMAL        34212
30340
ORCL:ASM01_013       1     12 CACHED  MEMBER       NORMAL        34212
30357
```

In this view there are 3 disks that are not assigned to any group (those with GROUP_# set to 0). These are unassigned disks that ASM has discovered but that have not been assigned to a disk group. Note the mount status of CLOSED on those three disks which also indicates that the disk is not being accessed by ASM. The HEADER_STATUS of FOREIGN indicates that these disks contain data already, and are owned by some process other than ASM (in this case, these are voting disks for a RAC cluster). If the HEADER_STATUS says CANIDATE, as with /dev/raw/rdisk6, then we could add this disk to an ASM disk group.

Notice that most of the disks have a MOUNT_STATUS of CACHED and a HEADER_STATUS of MEMBER. This means that the disk is currently part of an ASM disk group.

There are some cases where the V$ASM_DISK view will not report any disks. For example, on a Windows XP system there may be no raw disks to discover, so the V$ASM_DISK view will simply be blank. This is not a problem, because you can use an existing file system as a location for an ASM disk. This is discussed in the next section on how to add disk groups to ASM.

Some things to be aware of with regards to ASM disk discovery:

1. ASM can discover no more than 10,000 disks. If you have more than that, ASM will only discover the first 10,000 disks. This can occur when your ASM disk string is not sufficiently restrictive and the directory that you are searching in has a number of raw devices, but many of them are not going to be assigned to ASM.

2. ASM will not discover disks that contains an operating system partition table.

3. ASM may discover disks that already contain Oracle data on them (as in our example above with the voting disks).

Adding an ASM Disk Group

Once ASM has discovered the available disks, it is easy to create an ASM disk group. You use the create disk group command to create an ASM disk group. When you issue the command you will assign the disk group it's name, and you will add one or more discovered (unallocated) disks to that disk group. Here is an example of the use of the create disk group command:

```
CREATE DISKGROUP dgroup1 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK
'/devices/diska1'
failgroup diskcontrol2 DISK
'/devices/diskb1';
```

Let's look at some of the details of this command in a bit more detail next.

Redundancy

In the previous example, we created an ASM disk group using the NORMAL REDUNDANCY parameter. When configuring ASM disk groups you can use one of three different redundancy-setting options:

- Normal: Typically employs two way mirroring by default, and thus requires two failgroups be allocated.

- High: Typically employs three way mirroring by default, and thus requires three failgroups be allocated.

- External: Does not employ any mirroring or striping. This setting is typically used when the disk group is being assigned to an external disk that is attached to some device that already employs some disk redundancy.

The redundancy setting defines attributes for that disk group including what kind of striping occurs and if the data will be mirrored or not. You can take the default template setting, or you can assign another ASM template to a given disk group using the . For example, if you assigned the

The following list gives you some guidance to the type of configuration that will result depending on the redundancy setting you use:

Template name	**Striping**	**Mirroring when using a normal redundancy disk group**	**Mirroring when using a high redundancy disk group**	**Mirroring when using an external redundancy disk group**
Controlfile	Fine	3-Way Mirroring	3-Way Mirroring	No Mirroring
Datafile	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Onlinelog	Fine	2-Way Mirroring	3-Way Mirroring	No Mirroring
Archivelog	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Tempfile	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Backupset	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Parameterfile	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Dataguardconfig	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Flashback	Fine	2-Way Mirroring	3-Way Mirroring	No Mirroring

Changetracking	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Dumpset	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Xtransport	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring
Autobackup	Coarse	2-Way Mirroring	3-Way Mirroring	No Mirroring

So, in the earlier example of the creation of a disk group, we would have ended up with 2 way mirroring and coarse striping. In this case, with the two way mirroring, everything that would be written to the /devices/diska1 device would also be mirrored to the /devices/diskb1 device.

In the previous example we used the failgroup parameter of the create diskgroup command to designate a logical disk group structure within the diskgroup itself. Thus, in the previous example there were two failgroup groups defined, each of which had one disk assigned to it. If you are using a disk group that is doing 2-Way mirroring, then you will need at least two failgroup entries in each create diskgroup command. If you do not define a failgroup, Oracle will assign each disk to it's own failgroup.

We can opt for three-way mirroring by using the high redundancy parameter. In this case you will also need an additional failgroup entry for each mirror as seen in this example:

```
CREATE DISKGROUP dgroup1 HIGH REDUNDANCY
failgroup diskcontrol1 DISK
'/devices/diska1' NAME diska1
failgroup diskcontrol2 DISK
'/devices/diskb1' NAME diskb1
failgroup diskcontrol3 DISK
'/devices/diskc1' NAME diskc1;
```

You might have noticed the name clause in the create diskgroup command example earlier. You can also name the disks being assigned to the ASM disk group using the name clause of the create diskgroup command. Failure to use the name clause will result in each disk receiving it's own system default assigned name.

The previous examples assumed that you were creating a diskgroup on a raw partition. Of course, ASM can support file systems that have already been allocated via the operating system. To create ASM disk groups on cooked file systems we need to create a file on the file system so that we can point ASM at that file. When using non-raw file systems, we need to set the parameter ASM_DISKSTRING to the location we want to create the ASM disks devices:

```
C:\>mkdir c:\oracle\asm_disk
Sqlplus sys as sysdba
SQL>Alter system set ASM_DISKSTRING='c:\oracle\asm_disk\_file*';
```

We will also need to set a hidden parameter called _ASM_ALLOW_ONLY_RAW_DISKS to allow ASM to see non-raw disks. This is a hidden parameter so use it with caution:

```
SQL>alter system set "_asm_allow_only_raw_disks"=false scope=spfile;
```

Next, we need crate the files that ASM will see as disks. To do this, we can use a cool little pearl program like this one (on UNIX you could use the dd utility for example):

```
my $s='0' x 2**20;

open(DF1,">C:/oracle/asm_disk/_file_disk1") || die "Cannot create file -
$!\n";
open(DF2,">C:/oracle/asm_disk/_file_disk2") || die "Cannot create file -
$!\n";
open(DF3,">C:/oracle/asm_disk/_file_disk3") || die "Cannot create file -
$!\n";
open(DF4,">C:/oracle/asm_disk/_file_disk4") || die "Cannot create file -
$!\n";

for (my $i=1; $i<100; $i++) {
  print DF1 $s;
  print DF2 $s;
  print DF3 $s;
  print DF4 $s;
}

exit
```

And execute the script. Oracle comes with a useful Pearl interpreter:

```
C:\oracle\product\10.2.0\DB01\perl\5.8.3\bin\MSWin32-x86-multi-thread>perl
CreateFiles.pl
```

Once this is complete, check the c:\oracle\asm_disk directory and you will see files:

```
C:\oracle\asm_disk>dir
 Volume in drive C has no label.
 Volume Serial Number is 08DE-E1AB

 Directory of C:\oracle\asm_disk

01/27/2007  10:27 PM    <DIR>          .
01/27/2007  10:27 PM    <DIR>          ..
01/27/2007  10:27 PM       103,809,024 _file_disk1
01/27/2007  10:27 PM       103,809,024 _file_disk2
01/27/2007  10:27 PM       103,809,024 _file_disk3
01/27/2007  10:27 PM       103,809,024 _file_disk4
               4 File(s)    415,236,096 bytes
               2 Dir(s)  68,573,462,528 bytes free
```

Now, ASM should be seeing the new disks. We can tell by querying the V$ASM_DISK view as seen here:

```
SELECT group_number, disk_number, mount_status,
header_status, state, path
FROM   v$asm_disk;
```

And the results:

```
GROUP_NUMBER DISK_NUMBER MOUNT_S HEADER_STATU STATE    PATH
------------ ----------- ------- ------------ -------- ----------------------
--------
           0           0 CLOSED  CANDIDATE    NORMAL
C:\ORACLE\ASM_DISK\_FILE_DISK1
           0           3 CLOSED  CANDIDATE    NORMAL
C:\ORACLE\ASM_DISK\_FILE_DISK4
           0           2 CLOSED  CANDIDATE    NORMAL
C:\ORACLE\ASM_DISK\_FILE_DISK3
           0           1 CLOSED  CANDIDATE    NORMAL
C:\ORACLE\ASM_DISK\_FILE_DISK2
```

Note that our four disks now show up as available disks in ASM. We can now create disk groups using these disks:

```
CREATE DISKGROUP cooked_dgroup1 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK
'c:\oracle\asm_disk\_file_disk1' NAME file_diska1
failgroup diskcontrol2 DISK
'c:\oracle\asm_disk\_file_disk2' NAME file_diskb1;
```

We can see the resulting disk group in the v$ASM_DISKGROUPS view:

```
SQL> select group_number, name from v$ASM_DISKGROUP;

GROUP_NUMBER NAME
------------ ------------------------------
           1 COOKED_DGROUP1
```

When you create an ASM disk group, Oracle will add that disk group to the ASM_DISKGROUPSparameter if you are using an SPFILE. If you are not using an SPFILE then you will need to manually add the disk group to the ASM_DISKGROUPSparameter. The ASM_DISKGROUPSparameter tells Oracle which disk groups it should mount when the ASM instance is started. You can see the ASM_DISKGROUPSparameter setting by using the show parameter command from SQL*Plus as seen here:

```
SQL> show parameter ASM_DISKGROUPs

NAME                                 TYPE        VALUE
------------------------------------ ----------- --------------------------
ASM_DISKGROUPs                       string      COOKED_DGROUP1, SP_DGROUP2
```

If you do not add the disk group to the ASM_DISKGROUPS parameter, then you will need to manually mount the disk group.

We now have an ASM disk group that is ready to have Oracle files created on it.

Striping

Note the striping column in the table we presented earlier in the section on redundancy. There are two values there, FINE and COARSE. This refers to the strip size that ASM applies to the disks that the disk groups are assigned to. If FINE striping is selected, the ASM will use a 128KB stripe size. If coarse is selected then Oracle uses a 1MB stripe size.

Templates

When you create an ASM disk group, Oracle will assign a default template to that disk group (see the earlier list with the default templates listed). A template is simply a named collection of attributes. For example, if you create a diskgroup using the default template and then create datafiles in that disk group, the datafile template will define the redundancy and striping for that data.

There may be cases where you want to define your own template for a disk group. In this case you can you will need to first create the disk group, and then alter the disk group using the add template parameter of the alter diskgroup commands as seen in this example:

```
CREATE DISKGROUP sp_dgroup2 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK
'c:\oracle\asm_disk\_file_disk3' NAME file_diska1
failgroup diskcontrol2 DISK
'c:\oracle\asm_disk\_file_disk4' NAME file_diskb1;

ALTER DISKGROUP sp_dgroup2
ADD TEMPLATE new_template
ATTRIBUTES (mirror);
```

After the mirror template has been added, you can create files in that disk group using the new template.

One note on user defined templates. When you add a template to a disk group, the template can not be retroactively applied to files already in that disk group. As a result you will need to use RMAN to backup and then restore files that already exist in the disk group in order for them to take on the attributes of the new template.

You can see the templates associated with a given disk group by querying the V$ASM_TEMPLATE view as seen in this example:

```
SQL> select * from v$asm_template
  2  where group_number=2;

GROUP_NUMBER ENTRY_NUMBER REDUND STRIPE S NAME
------------ ------------ ------ ------ - --------------------
           2            0 MIRROR COARSE Y PARAMETERFILE
           2            1 MIRROR COARSE Y DUMPSET
           2            2 HIGH   FINE   Y CONTROLFILE
           2            3 MIRROR COARSE Y ARCHIVELOG
           2            4 MIRROR FINE   Y ONLINELOG
           2            5 MIRROR COARSE Y DATAFILE
           2            6 MIRROR COARSE Y TEMPFILE
           2            7 MIRROR COARSE Y BACKUPSET
           2            8 MIRROR COARSE Y AUTOBACKUP
           2            9 MIRROR COARSE Y XTRANSPORT
           2           10 MIRROR COARSE Y CHANGETRACKING
           2           11 MIRROR FINE   Y FLASHBACK
           2           12 MIRROR COARSE Y DATAGUARDCONFIG
           2           13 MIRROR COARSE N NEW_TEMPLATE
```

In this output note that the new template (new_template) has been created and is ready for use. You can drop a template with the alter diskgroup command using the drop template parameter as seen in this example:

```
ALTER DISKGROUP sp_dgroup2
DROP TEMPLATE new_template;
```

And you can alter a user defined template with the alter template parameter of the alter diskgroup command. Notice in this example we are actually changing one of the attributes of the default templates. You cannot drop the default templates, but you can modify them as seen here:

```
ALTER DISKGROUP sp_dgroup2
ALTER TEMPLATE datafile
ATTRIBUTES (coarse);
```

Dropping an ASM Disk Group

To remove an ASM disk group you use the DROP DISKGROUP command. By default, if any files exist in the disk group, ASM will not allow you to drop that disk group unless you use the including contents clause. The DROP DISKGROUP statement is synchronous in nature, so once the prompt returns, the diskgroup is deleted. When the DROP DISKGROUP command is executed, ASM will unmount the disk from the ASM instance and write over all the ASM related information on that disk. The ASM_DISKGROUPS parameter will also be changed if you are using a SPFILE. Here is an example of removing an ASM disk group with the DROP DISKGROUP command:

```
DROP DISKGROUP sp_dgroup2;
```

If the ASM Diskgroup has files in it, use this version:

```
DROP DISKGROUP sp_dgroup2 including contents;
```

Managing an ASM Disk Group

The ALTER DISKGROUP command is used to modify ASM disk groups. With the ALTER DISKGROUP command you can:

- Add disks to an ASM disk group

- Remove disks from an ASM disk group

- Undrop disks from an ASM disk group
- Resize disks in a disk group
- Manually rebalance a disk group
- Mount and unmount disk groups
- Check the consistency of a disk group
- Create ASM disk group directories
- Manage ASM disk group directories

Adding disks to an ASM Disk group

As databases grow you need to add disk space. The ALTER DISKGROUP command allows you to add disks to a given disk group to increase the amount of space available to a given disk group. Adding a disk to an existing disk group is can be done with the ALTER DISKGROUP command as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
add disk 'c:\oracle\asm_disk\_file_disk3'
name new_disk;
```

When you add a disk to a disk group, Oracle will start to rebalance the load on that disk group. Also, notice that in the example above that we did not assign the disk to a specific failgroup. As a result, each disk will be assigned to it's own failgroup when it's created. For example, when we added the disk above to the cooked_dgroup1 disk group, a new failgroup called cooked_dgroup1_0002 was created as seen in this output:

```
SQL> select disk_number, group_number, failgroup from v$asm_disk;

DISK_NUMBER GROUP_NUMBER failgroup
----------- ------------ ------------------------------
          1            0
          0            1 DISKCONTROL1
          1            1 DISKCONTROL2
          2            1 COOKED_DGROUP1_0002
```

We can add a disk to an existing failgroup by using the failgroup parameter as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
add failgroup DISKCONTROL1
disk 'c:\oracle\asm_disk\_file_disk4'
name new_disk;
```

Dropping Disks From an ASM Disk group

The ALTER DISKGROUP command allows you to remove disks from an ASM disk group using the drop disk parameter. ASM will first rebalance the data on the disks to be dropped, assuming enough space is available. If insufficient space is available to move the data from the disk to be dropped to another disk, then an error will be raised. You can use the force parameter to force ASM to drop the disk, but this can result in data loss. Here is an example of dropping a disk from a disk group:

```
ALTER DISKGROUP cooked_dgroup1
drop disk 'c:\oracle\asm_disk\_file_disk4'
```

Another option that the ALTER DISKGROUP command gives you is the option to drop all disks from a failgroup that are assigned to a disk group. Use the in failgroup clause to indicate the name of the failgroup as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
drop disks in failgroup diskcontrol1;
```

When you drop a disk from a disk group, the operation is an asynchronous operation. Therefore when the SQL prompt returns this does not indicate that the operation has completed. To determine if the operation has completed you will need to review the V$ASM_DISK view. When the disk drop is complete the column HEADER_STATUS will take on the value of FORMER as seen in this example:

```
SQL> select disk_number, header_status from v$asm_disk;

DISK_NUMBER HEADER_STATU
----------- ------------
          0 FORMER
          1 FORMER
          1 MEMBER
          2 MEMBER
```

If the drop is not complete (the v$asm_disk column STATE will read dropping), you can check the V$ASM_OPERATION view and it will give you an idea of how long the operation is expected to take before it is complete. Here is an example query that will provide you with this information:

```
select group_number, operation, state, power, est_minutes
from v$asm_operation;
```

Undropping disks from an ASM Disk group

If you accidentally drop a disk from a disk group and you realize your mistake only after the drop operation has completed, you can recover. If you have accidentally dropped a disk, simply use the ALTER DISKGROUP command using the undrop disks parameter as seen here:

```
ALTER DISKGROUP sp_dgroup2 undrop disks;
```

This will cancel the pending drop of disks from that disk group. You can not use this command to restore disks dropped if you dropped the entire disk group with the DROP DISKGROUP command.

Resizing disks assigned to an ASM Disk group

Sometimes when you need more space, all a disk administrator needs to do is add that additional space to the disk devices that are being presented for ASM to use. If this is the case, you will want to indicate to ASM that it needs to update it's metadata to represent the correct size of the disks it's using so you get the benefit of the additional space. This is accomplished using the ALTER DISKGROUP command using the resize all command as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1 resize all;
```

This command will query the operating system for the current size of all of the disk devices attached to the disk group and will automatically resize all disks in that disk group accordingly. You can indicate that a specific disk needs to be resized by including the disk name (from the NAME column in V$ASM_DISK) as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1 resize disk FILE_DISKB1;
```

You can also resize an entire failgroup at one time:

```
ALTER DISKGROUP cooked_dgroup1 resize disks in failgroup DISKCONTROL2;
```

Manually rebalance disks assigned to an ASM Disk group

Manually rebalancing disks within ASM is typically not required since ASM will perform this operation automatically. However, in cases where you might wish to have some more granular control over the disk rebalance process, you can use the ALTER DISKGROUP command along with the rebalance parameter to manually rebalance ASM disks.

When we discuss rebalancing disks in ASM we often discuss the power that is assigned to that rebalance operation. The setting of power with regards to a rebalance operation really defines the urgency of that operation with respect to other operations occurring on the system (e.g. other databases or applications). When a rebalance operation occurs with a low power (e.g. 1, the typical default) then that operation is not given a high priority on the system As a result the rebalance operation can take some time. When a higher power setting is used (e.g. 11, the maximum) the ASM is given higher priority. This can have an impact on other operations on the system. If you use a power of 0, this will have the effect of suspending the rebalance operation.

You can set the default power limit for the ASM instance by changing the asm_power_limit parameter.

Here is an example of starting a manual rebalance of a disk group:

```
ALTER DISKGROUP cooked_dgroup1 rebalance power 5 wait;
```

In this example, notice that we used the wait parameter. This makes this rebalance operation synchronous for our session. Thus when the SQL prompt returns we know the rebalance operation has completed. The default is nowait which will cause the operation to be synchronous in nature. You can check the status of the rebalance operation using the V$ASM_OPERATION view during asynchronous rebalance operations. If you use the wait parameter and you wish to convert the operation to an asynchronous operation, you can simply hit control-c on most platforms and an error will be returned along with the SQL prompt. The rebalance operation will continue, however.

If you do not use the power parameter during a manual rebalance operation, or if an implicit rebalance operation is occurring (because you are dropping a disk for example) you can effect the power of that rebalance operation by dynamically changing the ASM_POWER_LIMIT parameter to a higher value with the alter system command.

Finally, you can also use the rebalance parameter along with the power parameter when adding, dropping or resizing disks within a disk group as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1 resize all rebalance power 5;
```

Manually Mount and Unmount an ASM Disk group

If an ASM disk group is not assigned to the ASM_DISKGROUPS parameter, of if the disk group is unmounted for some other reason, you will need to mount the ASM disk group. You can use the ALTER DISKGROUP command with the mount clause to mount the disk group.

Additionally if you need to dismount an ASM disk group, you can use the ALTER DISKGROUP command to unmount that disk group. Here are some examples:

```
ALTER DISKGROUP sp_dgroup2 dismount;
ALTER DISKGROUP sp_dgroup2 mount;
```

Note that when you dismount a disk group, that disk group will be automatically removed from the ASM_DISKGROUPS parameter if you are using a SPFILE. This means that when ASM is restarted, that diskgroup will not be remounted. If you are using a regular text parameter file, you will need to remove the disk group manually (assuming it's in the parameter to begin with) or ASM will try to remount the disk group when the system is restarted.

Check the consistency of a disk group

On occasion you might wonder if there is some problem with an ASM disk group, and you will want to check the consistency of the ASM disk group metadata. This need might arise because of an error that occurs when the ASM instance is started, or as the result of an Oracle Database error that might be a result of some ASM corruption. To perform this check simply use the ALTER DISKGROUP command with the check all parameter as seen in this example:

```
ALTER DISKGROUP sp_dgroup2 check all;
```

That you executed a ALTER DISKGROUP check all command and the results are written to the alert log of the database. ASM will attempt to correct any errors that are detected. Here is an example of the ASM instance alert log entry after an ALTER DISKGROUP check all run that was successful:

```
Sun Jan 28 08:08:09 2007
SQL> ALTER DISKGROUP cooked_dgroup1 check all

Sun Jan 28 08:08:09 2007
NOTE: starting check of diskgroup COOKED_DGROUP1
SUCCESS: check of diskgroup COOKED_DGROUP1 found no errors
```

ASM Disk Group Directories

Create ASM Disk Group Directories

When you create an ASM disk group, it includes a system generated directory structure for the ASM files that will be stored in that disk group. The system generated directory structure takes on the following format:

```
+disk_group/database_name/object_type/ASM_file_name
```

Where disk_group is the root of the directory hierarchy. The database name will be the name of the database that the data is associated with. The object_type is the type of object being stored (e.g. datafile) and the ASM_file_name is the system generated file name assigned to that ASM file.

ASM allows you to create your own directories within these predefined structures. This allows you to define alias names to the ASM files that you will create. This can make working with ASM files easier.

To create a directory structure you use the ALTER DISKGROUP command with the add directory parameter as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
ADD DIRECTORY '+cooked_dgroup1/stuff';
```

Manage ASM Disk Group Directories

The ALTER DISKGROUP command is also used to manage ASM disk group directories. If you wish to drop an ASM disk group directory simply use the drop directory clause of the ALTER DISKGROUP command as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
DROP DIRECTORY '+cooked_dgroup1/stuff';
```

You can also rename directories with the ALTER DISKGROUP command using the rename directory clause as seen in this example:

```
ALTER DISKGROUP cooked_dgroup1
RENAME DIRECTORY '+cooked_dgroup1/stuff'
to '+cooked_dgroup1/badstuff';
```

You can see the ASM disk group directories in the V$ASM_ALIAS view as seen in this example:

```
select a.name "Alias Name", b.name "Disk Group"
from v$asm_alias a, v$ASM_DISKGROUPSb
where a.group_number=b.group_number

Alias Name                                         Disk Group
-------------------------------------------------- ---------------
badstuff                                           COOKED_DGROUP1
```

ASM Fast Disk Resync

Oracle 11g introduced ASM fast disk resync, which allows you to resynchronize ASM disks within an ASM disk group with a surviving disk group after a failure group becomes unavailable. ASM fast disk resync can resynchronize the missing disks as soon as the interruption is corrected. This resynchronization is dependent on the attached disk media not becoming corrupted. The time to perform the resync is dependent on a number of factors, but it often can be much quicker than rebuilding the entire disk group. ASM disks can be used during the resync operation though you will find that there will be performance degradation during the time of the disk repair (and the impact can be significant). If you are going to operate your database during a resync operation you will also want to consider the risks, since you will loose redundancy during the overall resync operation.

When a disk goes offline, ASM will drop that disk after some time. You need to avoid dropping the disk so that fast disk resynchronization will work. Oracle 11g provides the ability to configure the time between the disk failure and when ASM will drop the disk. You use the disk_repair_time attribute to keep a given disk from being dropped and to give you time to resync the disk. When the disk_repair_time attribute is set and a disk goes offline, the resync process will begin to keep track of changes to disk extents that belong to the offline disk. When the disk is brought back online, the resync process will then synchronize the disk and return it back to the disk group.

Disk_repair_time is defined in units of minutes or hours (using m/M or h/H to indicate the unit) including use of a decimal indicator for partial hours (for example, 3.5H). The default setting for disk_repair_time is 3.6 hours, which is not very long for a hardware failure, so you may wish to increase that time. The accumulated value for disk_repair_time is reset each time a disk is brought back online. Additionally you cannot set or reset the disk_repair_time attribute of a disk that has already gone offline.

If a disk goes offline and the disk repair time elapses, then the disk will be dropped. You can also drop a disk before the repair time expires by issuing the ALTER DISKGROUP ... disk offline statement using the drop after clause.

Here are some examples of setting a disk group's disk repair time attribute:

```
ALTER DISKGROUP DG1 SET ATTRIBUTE 'DISK_REPAIR_TIME'='12H';
ALTER DISKGROUP DG2 SET ATTRIBUTE 'DISK_REPAIR_TIME'='24.5H';
ALTER DISKGROUP DG3 SET ATTRIBUTE 'DISK_REPAIR_TIME'='200M';
```

If you want to set the disk_repair_time attribute then you will need to set the compatible ASM attributes or you will get the following error:

```
ORA-15032: not all alterations performed
ORA-15242: could not set attribute DISK_REPAIR_TIME
ORA-15283: ASM operation requires compatible.rdbms of 11.1.0.0.0 or higher
```

ASM Preferred Mirror Read

Your ASM configuration may involve remote mirroring to disks that are a fair distance away. When some of your disk mirrors are far away then those disks may not be the best set of disks for a given instance to read from. For example, you might have a Real Application Cluster database with local and remote mirrored disks. In this case, you want to have the RAC instances primarily read from the local disks to ensure the best performance. ASM preferred mirror read is designed to indicate to Oracle which disk failgroup is the preferred read disk group.

ASM preferred mirror read is only available if you are using RAC. Also preferred mirror read is generally used only with clustered ASM instances, but this is not a requirement. To take advantage of ASM preferred mirror read, you should configure each disk failure group with specific geographically located set of disks. Use the Oracle 11g parameter, asm_preferred_read_failure_groups, to configure a databse instance with a list of preferred disk failure group names to use when that instance accesses ASM disks. The format of the values of the asm_preferred_read_failure_groups parameter is diskgroupname.failuregroupname where diskgroupname is the name of the disk group that the failure group belongs to and failuregroupname is the preferred failure groups name. Include multiple diskgroup/failgroup names by separating each preferred read group defined with a comma as seen in this example:

```
asm_preferred_read_failure_groups=dgroup1.fdisk2, dgroup2.fdisk2
```

In the event ASM cannot read from the preferred disk failure group, then the non-preferred failure groups will be read. To determine if a given disk file group is a preferred disk group you can use the PREFERRED_READ column of the V$ASM_DISK view.

ASMCMD Command Line Tool

The ASMCMD tool is a command line utility that allows you to manage ASM instances and the disk structures and files within those instances. ASMCMD is only available in Oracle Database 10g Release 2. With ASMCMD you can:

- List contents of ASM disk groups
- Perform searches (like directory listings)
- Add or remove directories
- Display space availability and utilization
- Backup and restore ASM Diskgroup Metadata

ASMCMD allows you to traverse the ASM disks like a directory structure. From the root of the ASM instance, you can move down the various disk structures to find the disks you are interested in. Many of the commands in ASMCMD are UNIX like (i.e. ls, cd) and therefore the ASMCMD is fairly easy to learn how to use.

Starting ASMCMD

To start ASMCMD, set your ORACLE_SID to +ASM and then type ASMCMD from the command line as seen here:

```
C:\>Set ORACLE_SID=+ASM
C:\>ASMCMD
```

Or from UNIX:

```
/opt/oracle>export ORACLE_SID=+ASM
/opt/oracle>ASMCMD
```

Note: You will need to make sure that perl.exe is in the path before you run ASMCMD. If you have installed more than one ORACLE_HOME, it may take some setup to get the pathing set correctly. Make sure the following is set to the correct ORACLE_HOME:

- ORACLE_HOME
- PATH
- PERL5LIB
- PERLBIN

When ASMCMD starts, you will see the ASMCMD prompt as seen here:

```
ASMCMD>
```

You can start ASMCMD with the –p option and it will display the directory level that you are at as seen in this example:

```
C:\oracle\product\10.2.0\DB01\BIN>ASMCMD -p
ASMCMD [+] >
```

ASMCMD Commands

ASMCMD has a basic set of 16 commands. You can see these commands from the ASMCMD prompt if you type in help as seen in this output:

```
ASMCMD> help
        ASMCMD [-p] [command]

        The environment variables ORACLE_HOME and ORACLE_SID determine the
        instance to which the program connects, and ASMCMD establishes a
        bequeath connection to it, in the same manner as a SQLPLUS / AS
        SYSDBA.  The user must be a member of the SYSDBA group.

        Specifying the -p option allows the current directory to be displayed
        in the command prompt, like so:

        ASMCMD [+DATAFILE/ORCL/CONTROLFILE] >

        [command] specifies one of the following commands, along with its
        parameters.

        Type "help [command]" to get help on a specific ASMCMD command.

        commands:
        --------
        cd
        du
        find
        help
        ls
        lsct
        lsdg
        lsdsk
        md_backup
        md_restore
        mkalias
        mkdir
        pwd
        remap
        rm
        rmalias
```

cd – Change directory

The cd command will move you in and out of ASM directory structures. This command works just like it's UNIX equivelent, and the special designators for previous directory ".." are available for use and use "+" to indicate the root (instead of a forward or backwards slash (/ or \)). Here are some examples:

```
ASMCMD [+cooked_dgroup1] > cd ..
ASMCMD [+] > cd cooked_dgroup1
ASMCMD [+cooked_dgroup1] > cd +cooked_dgroup2
ASMCMD [+cooked_dgroup2] > cd rob10gr2
```

du – Disk Used

The du command indicates how much disk has been used on the disk group. Here is an example:

```
ASMCMD [+cooked_dgroup1/rob10gr2] > du
Used_MB      Mirror_used_MB
     44                  88
```

find – Find a directory or file

The find command can help you find a directory or file within the ASM system as seen here:

```
ASMCMD [+cooked_dgroup1] > find + rob10gr2
+COOKED_DGROUP1/ROB10GR2/
+COOKED_DGROUP2/ROB10GR2/
```

help

The help command provides you with help for all the ASMCMD commands. You can enter help, followed by the command name, for more detailed information on the command you are interested in.

ls

The ls command is the list file command. This will list the directories and/or files contained within the ASM disk structure. Here are some examples of using the ls command:

```
ASMCMD [+cooked_dgroup1/rob10gr2/datafile] > ls -l
Type      Redund  Striped  Time             Sys  Name
DATAFILE  MIRROR  COARSE   JAN 29 20:00:00  Y    ALIAS_TBS.260.613168611
DATAFILE  MIRROR  COARSE   JAN 29 20:00:00  Y    PART_ASM_TBS.256.613066047
DATAFILE  MIRROR  COARSE   JAN 29 20:00:00  Y    PART_ASM_TBS.257.613083267
```

The ls command has new functionality in Oracle 11g adding two new parameters, -c and -g. When used, the -c parameter will use the V$ASM_DISKGROUP view for the source of its output. Additionally the -g parameter is used, then GV$ASM_DISKGROUP will be used as the source of the output of the command. Thus, the ls command becomes "RAC" aware for a clustered ASM instance.

lsct

The lsct command lists all clients that are using ASM. You can specify a disk group to restrict the output or you can just allow the command to display all the clients. The output of the command displays the databases connected to the ASM instance, the version of the database connected. Also included is the compatible version of the database (set via the COMPATIBLE parameter setting) and the instance name of the instance connected to the ASM instance. Here is an example of the output from the lsct command:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > lsct
DB_Name   Status        Software_Version  Compatible_version  Instance_Name
rob10gr2  CONNECTED           10.2.0.1.0          10.2.0.1.0  rob10gr2
```

In Oracle 11g, the lsct command includes the -g parameter, which will cause it to select from the GV$ASM_CLIENT view rather than the V$ASM_CLIENT view.

lsdg

The lsdg command provides information on the disk groups that have been created in an ASM instance. This command returns a great deal of information about the disk group. Of particular interest is the free_MB column, which indicates how much space is available on the disk group. Here is an example of the output of the lsdg command:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > lsdg
State    Type    Rebal  Unbal  Sector  Block       AU  Total_MB  Free_MB
Req_mir_free_MB  Usable_file_MB  Offline_disks  Name
MOUNTED  NORMAL  N      N         512   4096  1048576       198       22
0              11             0  COOKED_DGROUP1/
MOUNTED  NORMAL  N      N         512   4096  1048576       198       58
0              29             0  COOKED_DGROUP2/
```

The lsdg command has new functionality in Oracle 11g adding two new parameters, -c and -g. The -c and -g parameters provide a view into the configuration of a clustered ASM instance. The -c parameter uses the V$ASM_DISKGROUP view for the source of its output. If the -g parameter is used, then GV$ASM_DISKGROUP will be used as the source of the output of the command.

lsdsk (Oracle 11g)

The lsdsk command lists the disks that are visible to ASM. Here is an example:

```
[oracle@localhost ~]$ asmcmd lsdsk
Path
/oracle/asmdata/asm_dgroup1_01.asm
/oracle/asmdata/asm_dgroup1_02.asm
```

md_restore and **md_backup (Oracle 11g)**

Commands that allow you to back up and restore the metadata associated with your ASM disk groups. See below for more information.

mkalias

Use the mkalias command to create ASM aliases for a given ASM file name. This command is equivalent to the alter diskgroup add alias command from the SQL prompt when connected to an ASM instance. Here is an example of using the mkalias to create an alias for an ASM file:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > mkalias
+cooked_dgroup1/rob10gr2/controlfile/Current.258.613087119
+cooked_dgroup1/control01.ctl
```

One thing to note is that the alias must reside in the same disk group as the ASM file. So the following command would not be legal:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > mkalias
+cooked_dgroup1/rob10gr2/controlfile/Current.258.613087119
+cooked_dgroup2/control01.ctl
```

mkdir

Mkdir allows you to create ASM directories. You can not create a directory in the root directory (+), and you can create multiple directories at one time by simply listing them one at a time on the mkdir command line separated by spaces. This command is roughly equivalent to the alter diskgroup add directory command. Here are a couple of examples of using the mkdir command:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > mkdir old

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > mkdir older oldest

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.258.613087119
Current.259.613088323
old/
older/
```

```
oldest/
```

pwd

Use the pwd command to determine where in the ASM directory tree you are. Here is an example:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > pwd
+cooked_dgroup1/rob10gr2/controlfile
```

remap (Oracle 11g)

If you find that you have physical blocks corrupted on your ASM disks the new remap command can be used to repair those blocks. Pass the command the required parameters including the disk group name, the disk name, and a range of blocks to start the remapping/repaid process. This command does not repair the data in the blocks, but rather just reconstructs the blocks in an attempt to make them usable again. As a result, media recovery will still be required after remapping of the block has been completed. In the following example we are remapping blocks 1000 to 3000 on disk disk_00A assigned to disk group dgroupA:

```
[oracle@boots ~]$ asmcmd remap dgroupA disk_00A 1000-3000
```

rm

The rm command is used to remove ASM directories or files. The rm command will only remove a directory if that directory is empty. This is equivalent to the alter diskgroup drop directory command. Here is an example of removing a directory with the rm command:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.258.613087119
Current.259.613088323
old/
older/
oldest/
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > rm old
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.258.613087119
Current.259.613088323
older/
oldest/
```

The rm command is also used to remove ASM files and associated aliases. This is equivalent to the alter diskgroup drop file command. Here is an example of the use of the rm command to remove an ASM control file:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.258.613087119
Current.259.613088323
older/
oldest/
```

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > rm Current.258.613087119
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.259.613088323
older/
oldest/
```

You can use the "*" wildcard character with the rm command too. If you use the wildcard character, Oracle will warn you to make sure that you truly want to remove the file. Here is an example:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
Current.259.613088323
older/
oldest/

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > rm current*
You may delete multiple files and/or directories.
Are you sure? (y/n) y

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
older/
oldest/

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] >
```

The rm command also comes with an –r parameter that allows you to recurse child directories. Use the –r parameter and some file, directory or alias specification. As with the use of the wildcard character, ASMCMD will query you to make sure you really want to complete the operation as seen in this example:

```
ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls
older/
oldest/

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > rm -r *
You may delete multiple files and/or directories.
Are you sure? (y/n) y

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] > ls

ASMCMD [+cooked_dgroup1/rob10gr2/controlfile] >
```

rmalias

The rmalias command is used to remove ASM aliases as seen in this example:

```
ASMCMD [+cooked_dgroup1/rob10gr2/datafile] > ls
ALIAS_TBS.260.613168611
PART_ASM_TBS.256.613066047
PART_ASM_TBS.257.613083267
```

```
ASMCMD [+cooked_dgroup1/rob10gr2/datafile] > mkalias
+cooked_dgroup1/rob10gr2/datafile/part_asm_tbs.256.613066047
+cooked_dgroup1/ro
b10gr2/datafile/alias_tbs_01.dbf

ASMCMD [+cooked_dgroup1/rob10gr2/datafile] > ls
+cooked_dgroup1/rob10gr2/datafile
ALIAS_TBS.260.613168611
PART_ASM_TBS.256.613066047
PART_ASM_TBS.257.613083267
alias_tbs_01.dbf

ASMCMD [+cooked_dgroup1/rob10gr2/datafile] > rmalias
+cooked_dgroup1/rob10gr2/datafile/alias_tbs_01.dbf
```

ASM Diskgroup Metadata Backup and Recover

As of Oracle 11g, asmcmd provides commands that allow you to back up and restore the metadata associated with your ASM disk groups. You can use the md_backup command to create a backup of all the ASM metadata, or a subset of ASM metadata, depending on the parameters that you use. Here is an example of using the md_backup command to back up the entire ASM metadata repository:

```
[oracle@localhost ~]$ asmcmd
ASMCMD> md_backup
```

This command will create the metadata backup file in the current working directory by default. On a Linux system this file was called ambr_backup_intermediate_file. The backup file is pretty much text-based, so you can read through it. More advanced use of the md_backup command allows you to control where the backup file will be created, specify the disk groups that you want to back up, and to override various options. The md_backup command will not overwrite a previous backup file. So you will need to move the file somewhere after the backup so the next backup will be successful.

You can restore the ASM metadata with the md_restore command. You can restore from the backup without any changes, or the md_backup command gives you numerous options to override settings too. In this example we are performing a metadata restore with the md_restore command:

```
[oracle@localhost ~]$ asmcmd
ASMCMD> md_restore –b ambr_backup_intermediate_file
```

Both commands can be run as a parameter from the command line as in this example:

```
[oracle@localhost ~]$ asmcmd md_backup
```

So, why would you use md_restore and md_backup? As with Oracle databases there is a possibility that the ASM metadata stored in the ASM disk groups could become corrupt (perhaps because of some in-memory OS corruption) or perhaps someone accidently used the dd command and overwrote the contents of the disk header. While this is probably a rare problem, it is in cases like these that your ASM metadata backups could come in handy. If you are using ASM, a metadata backup should become one of your new daily backup operations.

The md_restore command can also reinstate dropped disks, in some cases. Another use of that is developing an automatic change control process whenever you add, change, or drop disks.