Unit 5: Oracle Recovery Manager (RMAN)

- Database corruption
- automatic storage management
- RMAN configuration
- Database Archival

Database Corruption:

A data block is corrupted when it is not in a recognized Oracle Database format, or its contents are not internally consistent. Data block corruption can damage internal Oracle control information or application and user data, leading to crippling loss of critical data and services. Block corruptions may affect only a single block or a large portion of the database (making it essentially unusable). While relatively rare, corruptions are inevitable, but Oracle provides a complete set of technologies to prevent and mitigate block corruptions.

A database block is corrupted when its content has changed from what Oracle Database expects to find. If not prevented or repaired, block corruption can bring down the database and possibly result in the loss of key business data.

While you cannot prevent all block corruptions, there is a comprehensive set of data protection solutions that you can implement to address most of them. Oracle offers sophisticated solutions—such as:

- Oracle Data Guard
- Data Recovery Advisor
- Oracle Flashback
- Oracle Recovery Manager (RMAN)
- Automatic Diagnostic Recovery (ADR)
- Oracle Secure Backup
- The MAA Advisor component of Oracle Enterprise Manager Grid Control
- Exadata Storage

These features offer database optimized ways to protect your data and ensure high availability of your data and hence, the application.

How Corruption Manifests Itself

When Oracle issues a write operation, it moves through the following I/O sequence:

- to the file system
- to the volume manager
- to the device driver
- to the Host-Bus Adapter
- to the storage controller
- to the disk drive where data is written

Hardware failures or bugs in any layer can result in corrupt data being written to disk, or good data not written to disk (termed a “lost write”) yet reported as written to Oracle.

Physical and Logical Corruptions

Data corruption can manifest itself as a physical or a logical corruption:

Physical Corruption of a block manifests as an invalid checksum or header, or when the block contains all zeroes. When that happens, the database will not recognize the block as a valid Oracle block, regardless of its content. A physical corruption is also called a media corruption.

Logical Corruption happens when a data block has a valid checksum, etc., but the block contents are logically inconsistent. Logical block corruption can also occur when the structure below the beginning of the block (below the block header) is corrupt. In this case, the block checksum is correct but the block structures may be corrupt. Logical corruption can also result from a lost write in which a version of the block that was sent to disk somehow was not actually written. The result is that the version of that block on disk is older than the version in the buffer cache. Lost writes are usually caused by bugs in the operating system or hardware.

Intrablock and Interblock Corruptions

The data blocks to which we refer are Oracle data blocks, which are comprised of multiple operating system blocks that make up the database. The data blocks are stored on disk, but are also temporarily stored in the buffer cache in memory. Thus, corruptions do not always appear on disk and can be related to memory and transient in nature.

- for **intrablock corruption**, the corruption occurs in the block itself and can be either a physical or a logical corruption.
- for **interblock corruption**, the corruption occurs between blocks and can only be a logical corruption.

Automatic Storage Management (ASM):

Automatic Storage Management (ASM) is an integrated, high-performance database file system and disk manager. ASM is based on the principle that the database should manage storage instead of requiring an administrator to do it. ASM eliminates the need for you to directly manage potentially thousands of Oracle database files.

ASM groups the disks in your storage system into one or more disk groups each of which comprises of several physical disks that are controlled as a single unit. The physical disks are knows as ASM disks, while the files that reside on the disks are known as ASM files. The locations and names for the files are controlled by ASM.

ASM provides the following benefits:

- **Striping**—ASM spreads data evenly across all disks in a disk group to optimize performance and utilization. This even distribution of database files eliminates the need for regular monitoring and I/O performance tuning.
- **Mirroring**— Mirroring means keeping redundant copies, or mirrored copies, of each extent of the file, to help avoid data loss caused by disk failures. ASM can increase availability by optionally mirroring any file. ASM mirrors at the file level, unlike operating system mirroring, which mirrors at the disk level. The mirrored copy of each file extent is always kept on a different disk from the original copy. If a disk fails, ASM

can continue to access affected files by accessing mirrored copies on the surviving disks in the disk group. ASM supports 2-way mirroring, where each file extent gets one mirrored copy, and 3-way mirroring, where each file extent gets two mirrored copies.

- **Online storage reconfiguration and dynamic rebalancing**—ASM permits you to add or remove disks from your disk storage system while the database is operating. When you add a disk, ASM automatically redistributes the data so that it is evenly spread across all disks in the disk group, including the new disk. This redistribution is known as **rebalancing**. It is done in the background and with minimal impact to database performance. When you request to remove a disk, ASM first rebalances by evenly relocating all file extents from the disk being removed to the other disks in the disk group.
- **Managed file creation and deletion**—ASM further reduces administration tasks by enabling files stored in ASM disk groups to be Oracle-managed files. ASM automatically assigns filenames when files are created, and automatically deletes files when they are no longer needed.

In summary ASM provides following functionalities:

- Manages group of disks, called diskgroups.
- Manages disk redundancy within a disk group.
- Provides near-optimal I/O balancing without any manual tuning.
- Enables management of database objects without specifying mount points and filenames.
- Supports large files.

Recovery Manager (RMAN):

Recovery Manager is a client application that performs backup and recovery operation using the database server sessions. It is an Oracle utility that can back up, restore, and recover database files. It is a feature of the Oracle database server and does not require separate installation.

RMAN stores metadata about its operations in the control file of the target database and, optionally, in a recovery catalog schema in an Oracle database.

You can invoke RMAN as a command-line executable from the operating system prompt or use some RMAN features through the Enterprise Manager GUI.

RMAN was introduced in Oracle release 8.0 and is not compatible with Oracle databases prior to release 8.0.

Why use RMAN?

- It’s FREE (with your Oracle license).
- Backups can be checked for corruption before it become a problem.
- Minimize downtime after failure.
- Recover single blocks of data.
- Backup directly to tape.
- Creating duplicate instances, including standbys for failover.
- Backups can be taken online without the additional overhead of traditional online

backups.

- Automatically includes new datafiles and tablespaces without manual intervention.
- Only backs up used data blocks.
- Allows for compressed backups.
- Works with third-party media management.
- Allows for centralized management and reporting of backups.

RMAN processes most commands in two phases:

Compilation phase

During compilation phase, RMAN determines which objects the command will access. Then, RMAN constructs a sequence of remote procedure call (RPCs) that instruct the server sessions on the target database to perform the desired operation.

Execution phase

During the execution phase, RMAN sends the RPC calls to the target database, monitors their progress, and collects the results. If more than one channel is allocated, then RMAN can execute certain commands in parallel so that all of the channels' target database sessions are concurrently executing an RPC call.

Starting RMAN and Connecting to a Database

The RMAN client is started by issuing the rman command at the command prompt of your operating system. RMAN then displays a prompt for your commands as shown in the following example:

```
% rman
RMAN>
```

RMAN connections to a database are specified and authenticated in the same way as SQL*Plus connections to a database. The only difference is that RMAN connections to a target or auxiliary database require the SYSDBA privilege. The AS SYSDBA keywords are implied and cannot be explicitly specified. See Oracle Database Administrator's Guide to learn about database connection options for SQL*Plus.

You can connect to a database with command-line options or by using the CONNECT TARGET command. The following example starts RMAN and then connects to a target database through Oracle Net, AS SYSDBA is not specified because it is implied. RMAN prompts for a password.

```
% rman
RMAN> CONNECT TARGET SYS@prod
 target database Password: password
connected to target database: PROD (DBID=39525561)
```

The following variation starts RMAN and then connects to a target database by using operating system authentication:

```
% rman
RMAN> CONNECT TARGET /
connected to target database: PROD (DBID=39525561)
```

To quit the RMAN client, enter EXIT at the RMAN prompt:

```
RMAN> EXIT
```

RMAN Environment

The RMAN environment consists of the utilities and databases that play a role in backing up your data. At a minimum, the environment for RMAN must include the following components:

A target database

An Oracle database to which RMAN is connected with the TARGET keyword. A target database is a database on which RMAN is performing backup and recovery operations. RMAN always maintains metadata about its operations on a database in the control file of the database. The RMAN metadata is known as the RMAN repository.

The RMAN client

An Oracle Database executable that interprets commands, directs server sessions to execute those commands, and records its activity in the target database control file. The RMAN executable is automatically installed with the database and is typically located in the same directory as the other database executables. For example, the RMAN client on Linux is located in $ORACLE_HOME/bin.

Some environments use the following optional components:

A fast recovery area

A disk location in which the database can store and manage files related to backup and recovery. You set the fast recovery area location and size with the DB_RECOVERY_FILE_DIST and
DB_RECOVERY_FILE_DEST_SIZE initialization parameters.

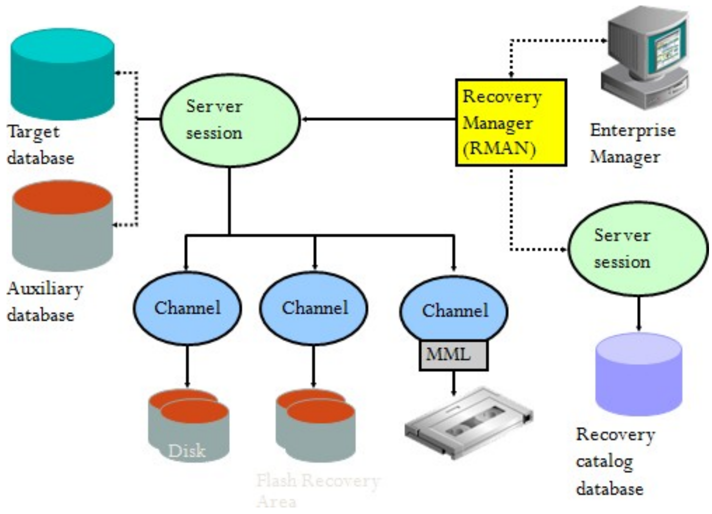
A media manager

An application required for RMAN to interact with sequential media devices such as tape libraries. A media manager controls these devices
during backup and recovery, managing the loading, labeling, and unloading of media. Media management devices are sometimes called SBT (system backup to tape) devices.

A recovery catalog

A separate database schema used to record RMAN activity against one or more target databases. A recovery catalog preserves RMAN repository metadata if the control file is lost, making it much easier to restore and recover following the loss of the control file. The database may overwrite older records in the control file, but RMAN maintains records forever in the catalog unless the records are deleted by the user.

RECOVERY MANAGER COMPONENTS:



Target Database

The target database is the database that RMAN is backing up, restoring, or recovering.

You can use a single recovery catalog in conjunction with multiple target databases. For example, assume that your data center contains 10 databases of varying sizes. You can use a single recovery catalog located in a different data center to manage the metadata from all of these databases.

RMAN Repository:

The RMAN repository is the collection of metadata about the target databases that RMAN uses for backup, recovery, and maintenance. RMAN always stores this information in records in the control file. The version of this information in the control file is the authoritative records of RMAN's backups of your database. This is one reason why protecting your control file is an important part of your backup strategy. RMAN can conduct all necessary backup and recovery operations using just the control file to store the RMAN repository information, and maintain all records necessary to meet your configured retention policy.

Among other things, RMAN stores information about:

- Backup sets and pieces
- Image copies (including archived redo logs)
- Proxy copies
- The target database schema
- Persistent configuration settings

You can access this metadata by issuing LIST, REPORT, and SHOW commands in the RMAN interface, or by using SELECT statements on the catalog views (only if you use a recovery catalog)

Media Management Interface:

To store backups on tape, RMAN requires a media manager. A media manager is a software program that loads, labels, and unloads sequential media such as tape drives used to back up and recover data.

Backup Types:

RMAN supports a number of different backup methods, depending on your availability needs, the desired size of your recovery window, and the amount of downtime you can endure while the database or a part of the database is involved in a recovery operation.

Consistent and Inconsistent Backup:

A physical backup can be classified by being a consistent or an inconsistent backup. In a consistent backup, all datafiles have the same SCN; in other words, all changes in the redo logs have been applied to the datafiles. Because an open database with no uncommitted transactions may have some dirty blocks in the buffer cache, it is rare that an open database backup can be considered consistent. As a result, consistent backups are taken when the database is shut down normally or in a MOUNT state.

In contrast, an inconsistent backup is performed while the database is open and users are accessing the database. Because the SCNs of the datafiles typically do not match when an inconsistent backup is taking place, a recovery operation performed using an inconsistent backup must rely on both archived and online redo log files to bring the database into a consistent state before it is opened. As a result, a database must be in ARCHIVELOG mode to use an inconsistent backup method.

Full and Incremental Backup:

Full backups include all blocks of every datafile within a tablespace or a database; it is essentially a bit-for-bit copy of one or more datafiles in the database. Either RMAN or an operating system command can be used to perform a full backup, although backups performed outside of RMAN must be cataloged with RMAN before they can be used in an RMAN recovery operation.

An alternative strategy to relying on full backups with archived redo logs is to use incremental backups along with archived redo logs for recovery. The initial incremental backup is known as a level 0 incremental backup. Each incremental backup after the initial incremental backup (also known as a level 1 incremental backup) contains only changed blocks and as a result takes less time and space.

Incremental level 1 backups can either be

- Cumulative Incremental or
- Differential Incremental

RMAN Recovery Catalog

The RMAN architecture revolves primarily around the recovery catalog. The recovery catalog is an Oracle database and associated objects in a schema that are created to manage RMAN backups. Although the recovery catalog is not required to use RMAN, some of the backup/recovery functions are not available without it being set up correctly. Generally, to set up a catalog, you identify the database in which the catalog will exist. This database should be separate from other Oracle databases. Once you have identified or created the database that will contain the recovery catalog, you create a user in that database and grant that user the RECOVERY_CATALOG_OWNER role. Then you will create the recovery catalog, create any backup scripts and begin to use the recovery catalog.

So, what are the benefits of using the recovery catalog? RMAN does not require the recovery catalog in most cases. Without the recovery catalog in place however, these RMAN features are not available:

- Tablespace point-in-time recovery
- Stored scripts
- Recovery when the control file is lost or damaged

The bottom line is that Oracle strongly recommends that you use the recovery catalog.

Once you have set up the recovery catalog, you use the RMAN executable in concert with RMAN manual commands or stored scripts to back up, recover, and report on backups. The next few sections of this topic look in more detail at the setup, backup, recovery, and reporting functions of RMAN. Finally, the topic returns (in much more detail) to the recovery catalog and addresses maintenance issues and reporting from the views Oracle supplies for the DBAs. First, let's look at how to set up the catalog and use it to register, back up, and recover databases.

RMAN Channels

Before you can execute a backup or recovery using RMAN, you must allocate a channel between the backup processes and the operating system. The following operations in RMAN must have at least one channel allocated to operate correctly:

- BACKUP
- COPY
- RESTORE
- RECOVER

To allocate a channel, you use the RMAN command ALLOCATE CHANNEL. When you allocate the channel, you also specify the type of device that will be used for the operation that will occur through that channel. Multiple channels can be allocated, allowing for multiple backup sets or file copies to run in parallel by the execution of just a single RMAN command. Each time you issue an ALLOCATE CHANNEL command, a separate connection between the backup processes and the operating system is created.

RMAN Vs Traditional Backup Methods:

Skip unused blocks:

Blocks that have never been written to, such as blocks above the high water mark (HWM) in a table, are not backed up by RMAN when the backup is an RMAN backupset. Traditional backup methods have no way to know which blocks have been used.

True incremental backups:

For any RMAN incremental backup, unchanged blocks since the last backup will not be written to the backup file. This saves a significant amount of disk space, I/O time, and CPU time.

Block-level recovery:

To potentially avoid downtime during a recovery operation, RMAN supports *block-level recovery for recovery operations that only need to restore or repair a* small number of blocks identified as being corrupt during the backup operation.

Multiple I/O channels:

During a backup or recovery operation, RMAN can utilize many I/O channels, via separate operating system processes, to perform concurrent I/O. Traditional backup methods, such as a Unix **cp command or an Oracle export, are** typically single-threaded operations.

Cataloging:

A record of all RMAN backups is recorded in the target database control file, and optionally in a recovery catalog stored in a different database.

Scripting capabilities:

RMAN scripts can be saved in a recovery catalog for retrieval during a backup session.

Encrypted backups:

RMAN uses backup encryption integrated into Oracle Database 11***g*** to store encrypted backups. Storing encrypted backups on tape requires the Advanced Security Option.

Backup Settings:

Default Device Type: Determines whether you will be backing up to disk or tape

- Options are mutually exclusive.
- Default may be overridden with the backup device type. parameter Syntax:
- CONFIGURE DEFAULT DEVICE TYPE TO DISK;
- CONFIGURE DEFAULT DEVICE TYPE TO SBT;

Format:

Multiple channels may be allocated. Various format options available

- %U: System generated unique filename (default).
- %F: DBID, day, month, year, sequence.
- %s: backup set.
- %p: backup piece. Syntax:

- CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT '/u01/backup/PPRD/rman/PPRD_%U';

Backup Method:

- Used to further define how backups will be carried out by default device type
- If compressed backup sets are chosen, backup files will be smaller, but it will take longer to restore from them.
- Syntax:
- CONFIGURE DEVICE TYPE DISK BACKUP TYPE TO BACKUPSET;
- CONFIGURE DEVICE TYPE DISK BACKUP TYPE TO COMPRESSED BACKUPSET;
- CONFIGURE DEVICE TYPE DISK BACKUP TYPE COPY;

RMAN - Retention Policies

When making backups, it's a good idea to define how long you want to keep a given backup set. A retention policy is a policy that revolves around how long backup sets are kept. In Oracle, RMAN allows you to define a retention policy regarding backup sets. There are two types of retention policies available in RMAN:

- Recovery window
- Redundancy

These policies are mutually exclusive of each other, and only one policy can be defined at a time. Policies are established with the CONFIGURE RETENTION POLICY command.

Note: Media manager retention policies can cause havoc if they are not properly synchronized with RMAN retention policies! For example, when using optimization, you could have your media manager software remove a backup, and RMAN would not know about it. This might cause the database to be unrecoverable.

Recovery Window Retention Policies:

The recovery window retention policy is used to ensure that a backup is available that will facilitate a recovery within the defined time window. For example, if the recovery window is set to seven days, then no backup will be reported as obsolete unless it is older than 7 days, the defined recovery window retention time.

To set a recovery window retention policy, use the CONFIGURE RETENTION POLICY command, as shown in the following example:

```
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 7 DAYS;
```

In this case, we have configured a retention policy of seven days. When the DELETE OBSOLETE command is executed, only backups older than seven days will be reported as obsolete.

Note: If the control_file_record_keep_time database parameter value is less than the time set for the retention policy, you will need to use a recovery catalog.

Redundancy Retention Policies

The redundancy retention policy defines a fixed number of backups that need to be retained by RMAN. If there are a number of backups later than the number defined by the retention policy, then those backups can be deleted using the DELETE OBSOLETE command, which we will discuss in the next section.

To set a redundancy retention policy, use the configure retention policy command, as shown here:

```
CONFIGURE RETENTION POLICY TO REDUNDANCY 7;
```

Use care when choosing your retention policies. While the redundancy policy will work fine if you are sure you will do just a certain number of backups per day, it is possible that you could easily loose backups that you don't intend to if you should back up your database multiple times on the same day (say, for example, because you just upgraded the database). Generally a recovery window retention policy is best for production databases to assure you can recover to the time you want. Redundancy policies might be more appropriate for test and development environments, in which backup space might be at a premium.

Remove the Retention Policy

If you define a policy and you wish to remove it, you can issue the CONFIGURE RETENTION POLICY command. To remove the existing retention policy, issue the following command

```
CONFIGURE RETENTION POLICY TO NONE;
```

Optionally, you can return the retention policy to the default policy, which is one day, by issuing the command below:

```
CONFIGURE RETENTION POLICY TO DEFAULT;
```

DELETE OBSOLETE

The RMAN DELETE OBSOLETE command is used in conjunction with the defined retention policy to remove obsolete backups. The example below will remove all backups that are obsolete based on the currently existing policy:

```
DELETE OBSOLETE;
```

The delete obsolete command also can take parameters to define a recovery window or backup redundancy that is different from the existing policy. If your default recovery window is seven days and you want to remove backups that are five days old, then you could issue this command:

```
DELETE OBSOLETE RECOVERY WINDOW OF 5 DAYS;
```

Also, the delete obsolete command provides a method of removing orphaned backups that were created from a different incarnation of the database. An example of this command would look like this:

```
DELETE OBSOLETE ORPHAN;
```