

7

Using RMAN to Perform Recovery

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Usman Qamar (usman.qamar@live.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to use RMAN to:

- Perform complete recovery when a critical or noncritical data file is lost
- Recover using incrementally updated backups
- Switch to image copies for fast recovery
- Restore a database onto a new host
- Recover using a backup control file

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Objectives

For more details, see the *Oracle Database Backup and Recovery User's Guide*.

Using RMAN RESTORE and RECOVER Commands

- RESTORE command: Restores database files from backup
- RECOVER command: Recovers restored files by applying changes recorded in incremental backups and redo log files

```
RMAN> SQL 'ALTER TABLESPACE inv_tbs OFFLINE IMMEDIATE';
RMAN> RESTORE TABLESPACE inv_tbs;
RMAN> RECOVER TABLESPACE inv_tbs;
RMAN> SQL 'ALTER TABLESPACE inv_tbs ONLINE';
```

- The Enterprise Manager Recovery Wizard creates and runs an RMAN script to perform the recovery.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Using RMAN RESTORE and RECOVER Commands

Reconstructing the contents of an entire database or a part of it from a backup typically involves two phases: retrieving a copy of the data file from a backup, and reapplying changes to the file since the backup from the archived and online redo logs, to bring the database to the desired SCN (usually the most recent one).

- RESTORE {DATABASE | TABLESPACE name [,name]... | DATAFILE name [,name] }...

The RESTORE command retrieves the data file onto disk from a backup location on tape, disk, or other media, and makes it available to the database server. RMAN restores from backup any archived redo logs required during the recovery operation. If backups are stored on a media manager, channels must be configured or allocated for use in accessing backups stored there.

- RECOVER {DATABASE | TABLESPACE name [,name]... | DATAFILE name [,name] }...

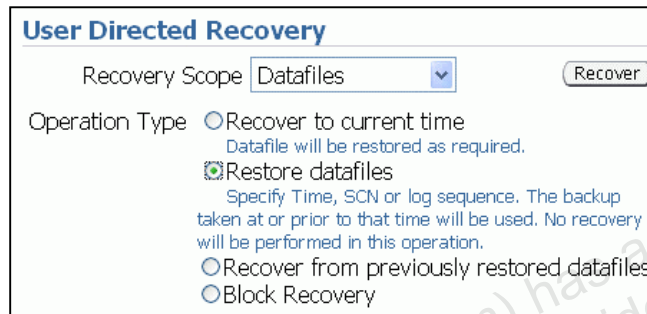
The RECOVER command takes the restored copy of the data file and applies to it the changes recorded in the incremental backups and database's redo logs.

You can also perform complete or point-in-time recovery by using the Recovery Wizard available through Enterprise Manager. On the Availability page, click Perform Recovery in the Backup/Recovery section.

Note: An automated method of detecting the need for recovery, and carrying out that recovery makes use of the Data Recovery Advisor, which is covered in the lesson titled “Diagnosing the Database.”

Performing Complete Recovery: Loss of a Noncritical Data File in ARCHIVELOG Mode

If a data file is lost or corrupted, and that file does not belong to the SYSTEM or UNDO tablespace, then restore and recover the missing data file.



User Directed Recovery

Recovery Scope: Datafiles [v] [Recover]

Operation Type:

- ☐ Recover to current time
Datafile will be restored as required.
- ☒ Restore datafiles
Specify Time, SCN or log sequence. The backup taken at or prior to that time will be used. No recovery will be performed in this operation.
- ☐ Recover from previously restored datafiles
- ☐ Block Recovery

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Complete Recovery: Loss of a Noncritical Data File in ARCHIVELOG Mode

With the database in ARCHIVELOG mode, the loss of any data file not belonging to the SYSTEM or UNDO tablespaces affects only those objects that are in the missing file.

To restore and recover the missing data file using Enterprise Manager, perform the following steps:

1. Click Perform Recovery on the Availability properties page.
2. Select “Datafiles” as Recovery Scope and “Restore datafiles” as Operation Type.
3. Add all data files that need recovery.
4. Specify from what backup the files are to be restored.
5. Determine whether you want to restore the files to the default location or (if a disk or controller is missing) to a new location.
6. Submit the RMAN job to restore and recover the missing files.

Because the database is in ARCHIVELOG mode, recovery up to the time of the last commit is possible and users are not required to reenter any data.

Performing Complete Recovery: Loss of a System-Critical Data File in ARCHIVELOG Mode

If a data file is lost or corrupted, and that file belongs to the SYSTEM, UNDO (or SYSAUX) tablespace, then perform the following steps:

1. The instance may or may not shut down automatically. If it does not, use SHUTDOWN ABORT to shut the instance down.
2. Mount the database.
3. Restore and recover the missing data file.
4. Open the database.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Complete Recovery: Loss of a System-Critical Data File in ARCHIVELOG Mode

Data files belonging to the SYSTEM tablespace or containing UNDO data are considered system critical. If Enterprise Manager is used for recovery, the SYSAUX tablespace is critical as well. A loss of one of these files requires the database to be restored from the MOUNT state (unlike other data files that may be restored with the database open).

Perform the following steps for complete recovery:

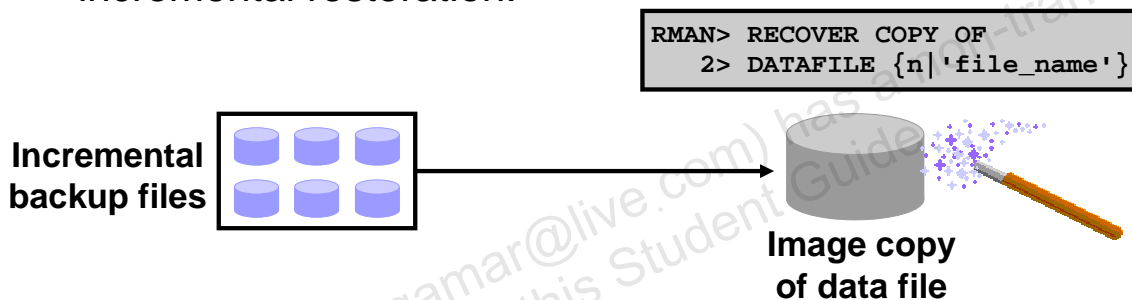
1. If the instance is not already shut down, shut it down.
2. Mount the database.
3. Click Perform Recovery on the Maintenance properties page.
4. Select "Datafiles" as the recovery type, and then select "Restore to current time."
5. Add all data files that need recovery.
6. Determine whether you want to restore the files to the default location or (if a disk or controller is missing) to a new location.
7. Submit the RMAN job to restore and recover the missing files.
8. Open the database. Users are not required to reenter data because the recovery is up to the time of the last commit.

Note: This kind of recovery situation is detected by the Data Recovery Advisor, which is covered in the lesson titled "Diagnosing the Database."

Recovering Image Copies

RMAN can recover image copies by using incremental backups:

- Image copies are updated with all changes up to the incremental backup SCN.
- Incremental backup reduces the time required for media recovery.
- There is no need to perform an image copy after the incremental restoration.



ORACLE

Copyright © 2009, Oracle. All rights reserved.

Recovering Image Copies

You can use RMAN to apply incremental backups to data file image copies. With this recovery method, you use RMAN to recover a copy of a data file—that is, you roll forward (recover) the image copy to the specified point in time by applying the incremental backups to the image copy. The image copy is updated with all changes up through the SCN at which the incremental backup was taken. RMAN uses the resulting updated data file in media recovery just as it would use a full image copy taken at that SCN, without the overhead of performing a full image copy of the database every day. The following are the benefits of applying incremental backups to data file image copies:

- You reduce the time required for media recovery (using archive logs) because you need to apply archive logs only since the last incremental backup.
- You do not need to perform a full image copy after the incremental restoration.

If the recovery process fails during the application of the incremental backup file, you simply restart the recovery process. RMAN automatically determines the required incremental backup files to apply, from before the image data file copy until the time at which you want to stop the recovery process. If there is more than one version of an image copy recorded in the RMAN catalog, RMAN automatically uses the latest version of the image copy. RMAN reports an error if it cannot merge an incremental backup file with an image copy.

Recovering Image Copies: Example

If you run these commands daily:

```
RMAN> recover copy of database with tag 'daily_inc';  
RMAN> backup incremental level 1 for recover of copy  
2> with tag 'daily_inc' database;
```

This is the result:

	RECOVER	BACKUP
Day 1	Nothing	Create image copies
Day 2	Nothing	Create incremental level 1
Day 3 and onward	Recover copies based on incremental	Create incremental level 1

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Recovery Image Copies: Example

If you run the commands shown in the slide daily, you get continuously updated image copies of all the database data files at any time.

The chart shows what happens for each run. Note that this algorithm requires some priming; the strategy does not come to fruition until after day 3.

Day 1: The RECOVER command does nothing. There exist no image copies to recover yet. The BACKUP command creates the image copies.

Day 2: The RECOVER command, again, does nothing. This is because there is no incremental backup yet. The BACKUP command creates the incremental backup, now that baseline image copies have been created on day 1.

Day 3: The RECOVER command applies the changes from the incremental backup to the image copies. The BACKUP command takes another incremental backup, which will be used to recover the image copies on day 4. The cycle continues like this.

It is important to use tags when implementing this kind of backup strategy. They serve to link these particular incremental backups to the image copies that are made. Without the tag, the most recent, and possibly the incorrect, incremental backup would be used to recover the image copies.

Performing a Fast Switch to Image Copies

Perform fast recovery by performing the following steps:

1. Take data files offline.
2. Use the SWITCH TO . . . COPY command to switch to image copies.
3. Recover data files.
4. Bring data files online.

Now the data files are recovered and usable in their new location.

Optionally, do the following to put the files back into their original location:

5. Create an image copy of the data file in the original location.
6. Take data files offline.
7. SWITCH TO . . . COPY
8. Recover data files.
9. Bring data files online.

```
SQL> SWITCH DATAFILE 'filename' TO COPY;
```

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing a Fast Switch to Image Copies

You can use image copies of data files for fast recovery by performing the following steps:

1. Take the data file offline.
2. Use the SWITCH TO . . . COPY command to point to the image copy of the files.
3. Recover the data files.
4. Bring the data files online.

At this point, the database is usable, and the data files are recovered. But, if you want to put the data files back into their original location, proceed with the following steps:

5. Create an image copy of the data files in the original location using the BACKUP AS COPY command.
6. Take the data files offline.
7. Switch to the copy you made in step 5 using the SWITCH TO COPY command.
8. Recover the data files.
9. Bring the data files online.

You can recover data files, tablespaces, tempfiles, or the entire database with this command. The files being switched to must be image copies.

Using SET NEWNAME for Switching Files

- Use the SET NEWNAME command in a RUN block to restore to a nondefault location.

```
RUN
{
  ALLOCATE CHANNEL dev1 DEVICE TYPE DISK;
  ALLOCATE CHANNEL dev2 DEVICE TYPE sbt;
  SQL "ALTER TABLESPACE users OFFLINE IMMEDIATE";
  SET NEWNAME FOR DATAFILE '/disk1/oradata/prod/users01.dbf'
    TO '/disk2/users01.dbf';
  RESTORE TABLESPACE users;
  SWITCH DATAFILE ALL;
  RECOVER TABLESPACE users;
  SQL "ALTER TABLESPACE users ONLINE";
}
```

- Instead of individual names, specify a default name format for all files in a database or in a named tablespace.
- The default name is used for DUPLICATE, RESTORE, and SWITCH commands in the RUN block.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Using SET NEWNAME for Switching Files

The SET NEWNAME command can be used only inside a RUN block. It prepares a name mapping for subsequent operations. In the example in the slide, the SET NEWNAME command defines the location where a restore operation of that data file will be written. When the RESTORE command executes, the users01.dbf data file is restored to /disk2/users01.dbf. It is written there, but the control file is still not pointing to that location. The SWITCH command causes the control file to be updated with the new location.

A more efficient way is to use the SET NEWNAME clause to specify the default name format for all data files in a named tablespace and all data files in the database (rather than setting file names individually, as in database versions prior to Oracle Database 11gR2 (11.2)).

The order of precedence for the SET NEWNAME command is as follows:

1. SET NEWNAME FOR DATAFILE and SET NEWNAME FOR TEMPFILE
2. SET NEWNAME FOR TABLESPACE
3. SET NEWNAME FOR DATABASE

Substitution Variables for SET NEWNAME

Syntax Element	Description
%b	Specifies the file name without the directory path <i>*NEW*</i>
%f	Specifies the absolute file number of the data file for which the new name is generated
%I	Specifies the DBID
%N	Specifies the tablespace name
%U	Specifies a system-generated file name of the format: data-D-%d_id-%I_TS-%N_FNO-%f

RUN

```
{ SET NEWNAME FOR DATAFILE 1 TO '/oradata1/system01.dbf';
SET NEWNAME FOR DATAFILE 2 TO '/oradata2/sysaux01.dbf';
SET NEWNAME FOR DATAFILE 3 TO '/oradata3/undotbs01.dbf';
SET NEWNAME FOR DATAFILE 4 TO '/oradata4/users01.dbf';
SET NEWNAME FOR TABLESPACE example TO '/oradata5/%b';
DUPLICATE TARGET DATABASE TO dupldb; }
```

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Substitution Variables for SET NEWNAME

To avoid possible name collisions when restoring to another location, use the substitution variables of the SET NEWNAME command. Specify at least one of the following substitution variables: %b, %f, and %U. %I and %N are optional variables.

The example shows the SET NEWNAME FOR TABLESPACE command to set default names with a substitution variable, together with explicit SET NEWNAME clauses.

Performing Restore and Recovery of a Database in NOARCHIVELOG Mode

- If the database is in NOARCHIVELOG mode, and any data file is lost, perform the following tasks:
 - Shut down the instance if it is not already down.
 - Restore the entire database, including all data and control files, from the backup.
 - Open the database.
- Users must reenter all changes made since the last backup.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Restore and Recovery of a Database in NOARCHIVELOG Mode

The loss of any data file from a database in NOARCHIVELOG mode requires complete restoration of the database, including control files and all data files. If you have incremental backups, then you need to perform the restore and recover operations. If the lost data file belongs to a read-only tablespace, you need to restore only that file.

With the database in NOARCHIVELOG mode, recovery is possible only up to the time of the last backup. So users must reenter all changes made since that backup.

For this type of recovery, use the RESTORE and RECOVER commands, or perform the following tasks in Enterprise Manager:

1. Shut down the instance if it is not already down.
2. Click Perform Recovery on the Maintenance properties page.
3. Select Whole Database as the type of recovery.

Using Restore Points

A restore point provides a name to a point in time:

- Now:

```
SQL> CREATE RESTORE POINT before_mods;
```

- Some time in the past:

```
SQL> CREATE RESTORE POINT end_q1 AS OF SCN 100;
```

Timeline

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Using Restore Points

You can give a name to a particular point in time, or an SCN number. This is useful for future reference, when performing point-in-time recovery or flashback operations.

- The first example in the slide creates a restore point that represents the present point in time. If you were about to apply an update of an application or data in the database, and you wanted to refer back to this state of the database, you could use the `BEFORE_MODS` restore point.
- The second example in the slide creates a restore point representing a past SCN, 100. This restore point can be used in the same ways as the previous one.

Normally, restore points are maintained in the database for at least as long as specified by the `CONTROL_FILE_RECORD_KEEP_TIME` initialization parameter. However, you can use the `PRESERVE` option when creating a restore point, which causes the restore point to be saved until you explicitly delete it.

You can see restore points in the `V$RESTORE_POINT` view with name, SCN, timestamp and other information.

Performing Point-in-Time Recovery

Perform server-managed point-in-time recovery by doing the following:

1. Determine the target point of the restore: SCN, time, restore point, or log sequence number.
2. Set the NLS environment variables appropriately.
3. Mount the database.
4. Prepare and run a RUN block, using the SET UNTIL, RESTORE, and RECOVER commands.
5. Open the database in READONLY mode, and verify that the recovery point is what you wanted.
6. Open the database using RESETLOGS.



ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Point-in-Time Recovery

You can perform server-managed point-in-time recovery using the following steps. The database must be in ARCHIVELOG mode.

1. Determine the restore target. This can be in terms of a date and time, an SCN, restore point, or log sequence number. For example, if you know that some bad transactions were submitted at 3:00 PM yesterday, then you can choose 2:59 PM yesterday as the target restore point time.
2. Set the National Language Support (NLS) OS environment variables, so that the time constants you provide to RMAN are formatted correctly. These are some example settings:


```
$ export NLS_LANG = american_america.us7ascii
$ export NLS_DATE_FORMAT = "yyyy-mm-dd:hh24:mi:ss"
```
3. Mount the database. If it is open, you have to shut it down first, as in this example:


```
RMAN> shutdown immediate
RMAN> startup mount
```

Performing Point-in-Time Recovery (continued)

4. Create a RUN block and run it. The RECOVER and RESTORE commands should be in the same RUN block so that the UNTIL setting applies to both. For example, if you choose to recover to a particular SCN, the RESTORE command needs to know that value so it restores files from backups that are sufficiently old—that is, backups that are from before that SCN. Here is an example of a RUN block:

```
RUN
{
  SET UNTIL TIME '2007-08-14:21:59:00';
  RESTORE DATABASE;
  RECOVER DATABASE;
}
```

5. As soon as you open the database for read/write, you have committed to the restore you just performed. So, first, open the database READ ONLY, and view some data, to check whether the recovery did what you expected.

```
RMAN> SQL 'ALTER DATABASE OPEN READ ONLY';
```

6. If satisfied with the results of the recovery, open the database with the RESETLOGS option, as shown:

```
RMAN> ALTER DATABASE OPEN RESETLOGS;
```

Performing Recovery with a Backup Control File

- Restore and mount a backup control file when all copies of the current control file are lost or damaged.
- Execute the `RECOVER` command after restoring the backup control file.
- Open the database with the `RESETLOGS` option after performing complete or point-in-time recovery.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Recovery with a Backup Control File

If you have lost all copies of the current control file, you must restore and mount a backup control file before performing recovery. Your recovery operation may be to recover lost data files or it may be to simply recover the control file. If you are using a recovery catalog, the process is identical to recovery with a current control file because RMAN can use the recovery catalog to obtain RMAN metadata.

Recovery from Loss of Server Parameter File

The `FROM MEMORY` clause allows the creation of current systemwide parameter settings.

```
SQL> CREATE PFILE [= 'pfile_name' ]  
      FROM { { SPFILE [= 'spfile_name'] } | MEMORY } ;
```

```
SQL> CREATE SPFILE [= 'spfile_name' ]  
      FROM { { PFILE [= 'pfile_name' ] } | MEMORY } ;
```

ORACLE

Copyright © 2009, Oracle. All rights reserved.

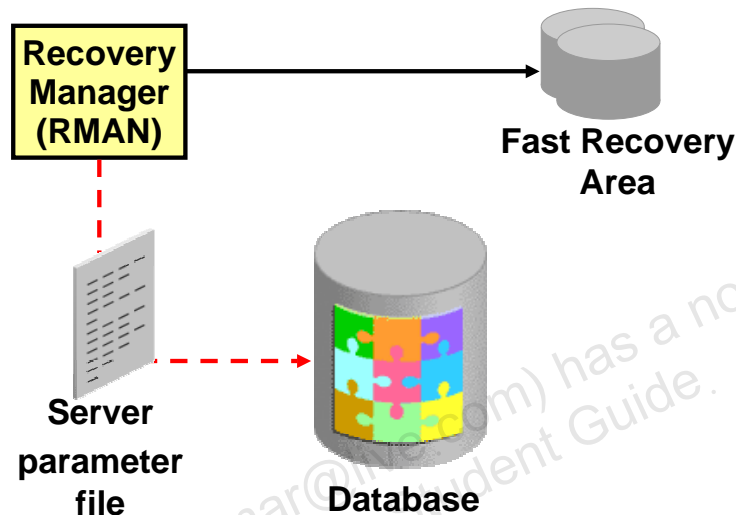
Recovery from Loss of Server Parameter File

The easiest way to recover a server parameter file is to use the `FROM MEMORY` clause, which creates a text initialization parameter file (PFILE) or server parameter file (SPFILE) using the current systemwide parameter settings. In a RAC environment, the created file contains the parameter settings from each instance.

During instance startup, all parameter settings are logged to the `alert.log` file. As of Oracle Database 11g, the `alert.log` parameter dump text is written in valid parameter syntax. This facilitates cutting and pasting of parameters into a separate file, and then using as a PFILE for a subsequent instance. The name of the PFILE or SPFILE is written to the `alert.log` at instance startup time. In cases when an unknown client-side PFILE is used, the alert log indicates this as well. To support this additional functionality, the `COMPATIBLE` initialization parameter must be set to 11.0.0.0 or higher.

Restoring the Server Parameter File from the Control File Autobackup

```
RMAN> STARTUP FORCE NOMOUNT;  
RMAN> RESTORE SPFILE FROM AUTOBACKUP;  
RMAN> STARTUP FORCE;
```



ORACLE

Copyright © 2009, Oracle. All rights reserved.

Restoring the Server Parameter File from the Control File Autobackup

If you have lost the server parameter file and you cannot use the `FROM MEMORY` clause, then you can restore it from the autobackup. The procedure is similar to restoring the control file from autobackup. If the autobackup is not in the fast recovery area, set the `DBID` for your database. Issue the `RESTORE SPFILE FROM AUTOBACKUP` command.

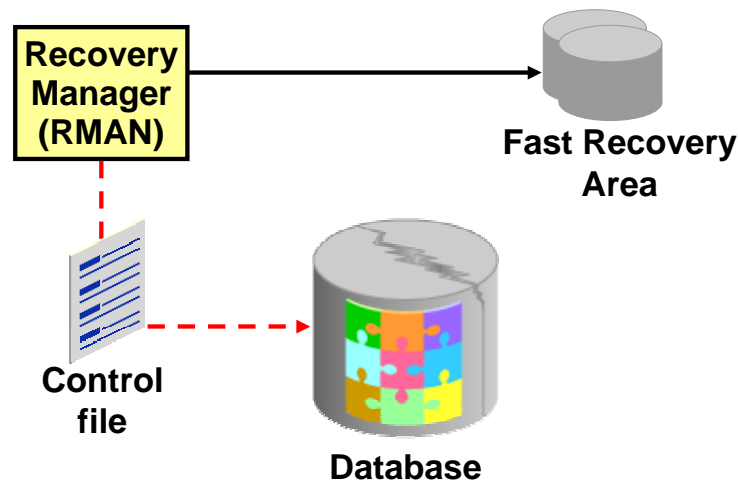
If you are restoring the SPFILE to a nondefault location, specify the command as follows:

```
RESTORE SPFILE TO <file_name> FROM AUTOBACKUP
```

If you are restoring the server parameter file from the Fast Recovery Area, specify the command as follows:

```
RMAN> run {  
2> restore spfile from autobackup  
3> recovery area = '<fast recovery area destination>'  
4> db_name = '<db_name>';  
5> }
```

Restoring the Control File from Autobackup



```
RMAN> STARTUP NOMOUNT;  
RMAN> RESTORE CONTROLFILE FROM AUTOBACKUP;  
RMAN> ALTER DATABASE MOUNT;  
RMAN> RECOVER DATABASE;  
RMAN> ALTER DATABASE OPEN RESETLOGS;
```

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Restoring the Control File from Autobackup

If you are not using a recovery catalog, you should have autobackup of the control file configured, so that you are able to quickly restore the control file if needed. The commands used for restoring your control file are the same, whether or not you are using a Fast Recovery Area. However, if you are using a Fast Recovery Area, RMAN implicitly cross-checks backups and image copies listed in the control file, and catalogs any files in the Fast Recovery Area not recorded in the restored control file, improving the usefulness of the restored control file in the restoration of the rest of your database.

Use the commands shown in the slide to recover from lost control files. First, start the instance in NOMOUNT mode. It cannot be mounted because there is no control file. Restore the control file from backup. Now that there is a control file, you can mount the database. You must now recover the database, because you now have a backup control file that contains information about an older version of the database. After recovering the database, you can open it. You must specify RESETLOGS because the new control file represents a different instantiation of the database.

Note: Tape backups are not automatically cross-checked after the restoration of a control file. If you are using tape backups, then after restoring the control file and mounting the database, you must cross-check the backups on tape.

Restoring the Control File from Autobackup (continued)

To restore the control file from an autobackup, the database must be in a NOMOUNT state. If the autobackup is not in the fast recovery area, you must set the database identifier (DBID) before issuing the `RESTORE CONTROLFILE FROM AUTOBACKUP` command, as shown in the following example:

```

RMAN> SHUTDOWN ABORT;
RMAN> STARTUP NOMOUNT;
RMAN> SET DBID 1090770270;
RMAN> RESTORE CONTROLFILE FROM AUTOBACKUP;

```

RMAN searches for a control file autobackup. If one is found, RMAN restores the control file from that backup to all the control file locations listed in the `CONTROL_FILES` initialization parameter.

If you have a recovery catalog, you do not have to set the DBID or use the control file autobackup to restore the control file. You can use the `RESTORE CONTROLFILE` command with no arguments:

```

RMAN> RESTORE CONTROLFILE;

```

The instance must be in the NOMOUNT state when you perform this operation, and RMAN must be connected to the recovery catalog. The restored control file is written to all locations listed in the `CONTROL_FILES` initialization parameter.

Use the `RESTORE CONTROLFILE... TO <destination>` command to restore the control file to a nondefault location.

If you have also lost the SPFILE for the database and need to restore it from the autobackup, the procedure is similar to restoring the control file from autobackup. You must first set the DBID for your database, and then use the `RESTORE SPFILE FROM AUTOBACKUP` command.

After you have started the instance with the restored server parameter file, RMAN can restore the control file from the autobackup. After you restore and mount the control file, you have the backup information necessary to restore and recover the database.

After restoring the control files of your database from backup, you must perform complete media recovery and then open your database with the `RESETLOGS` option.

Using Incremental Backups to Recover a Database in NOARCHIVELOG Mode

Use incremental backups to perform limited recovery of a database in NOARCHIVELOG mode.

```
STARTUP FORCE NOMOUNT;
RESTORE CONTROLFILE;
ALTER DATABASE MOUNT;
RESTORE DATABASE;
RECOVER DATABASE NOREDO;
ALTER DATABASE OPEN RESETLOGS;
```

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Using Incremental Backups to Recover a Database in NOARCHIVELOG Mode

You can perform limited recovery of a NOARCHIVELOG mode database by using incremental backups. The incremental backups must be consistent backups.

If you have taken incremental backups, RMAN will use your level 0 and level 1 backups to restore and recover the database.

You must specify the NOREDO option on the RECOVER DATABASE command if the online redo log files are lost or if the redo cannot be applied to the incremental backups. If you do not specify the NOREDO option, RMAN searches for the online redo log files after applying the incremental backups. If the online redo log files are not available, RMAN issues an error message.

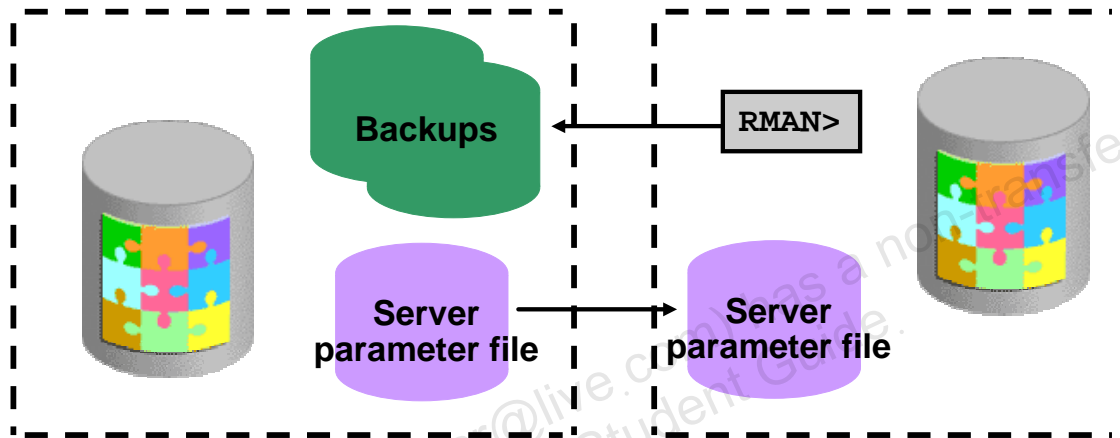
If the current online redo log files contain all changes since the last incremental backup, you can issue the RECOVER DATABASE command without the NOREDO option and the changes will be applied.

Note: You need to restore the control file only if it is not current.

Restoring and Recovering the Database on a New Host

Use the procedure to:

- Perform test restores
- Move a production database to a new host



ORACLE

Copyright © 2009, Oracle. All rights reserved.

Restoring and Recovering the Database on a New Host

Use the procedure described on the following pages to perform test restores. You can also use it to move a production database to a new host.

The database identifier (DBID) for the restored test database is the same as the DBID of the original database. If you are using a recovery catalog and connect to the test database and the recovery catalog database, the recovery catalog is updated with information about the test database. This can impact RMAN's ability to restore and recover the source database.

You should create a duplicate database using the RMAN DUPLICATE command if your goal is to create a new copy of your target database for ongoing use on a new host. The duplicate database is assigned a new DBID that allows it to be registered in the same recovery catalog as the original target database. Refer to the lesson titled "Using RMAN to Duplicate a Database" for detailed information about the DUPLICATE command.

Preparing to Restore the Database to a New Host

To prepare to restore a database, perform the following steps:

- Record the database identifier (DBID) of your source database.
- Copy the source database initialization parameter file to the new host.
- Ensure that source backups, including the control file autobackup, are accessible on the restore host.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Preparing to Restore the Database to a New Host

Perform the steps listed in the slide to prepare for the restore of the database to a new host.

Note: If you are performing a test restore, do not connect to the recovery catalog when restoring the data files. If you connect to the recovery catalog, RMAN records information about the restored data files in the recovery catalog and considers the restored database as the current target database. If your control file is not large enough to contain all of the RMAN repository data on the backups you need to restore and you must use a recovery catalog, then export the catalog and import it into a different schema or database. Use the copied recovery catalog for the test restore.

Restoring the Database to a New Host

Perform the following steps on the restore host to restore the database:

1. Configure the ORACLE_SID environment variable.
2. Start RMAN and connect to the target instance in NOCATALOG mode.
3. Set the database identifier (DBID).
4. Start the instance in NOMOUNT mode.
5. Restore the server parameter file from the backup sets.
6. Shut down the instance.
7. Edit the restored initialization parameter file.
8. Start the instance in NOMOUNT mode.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Restoring the Database to a New Host

Perform the steps listed on this page and the next on the restore host to restore the database.

1. Configure the ORACLE_SID environment variable as shown in the following example:

```
$ setenv ORACLE_SID orcl
```
2. Start RMAN and connect to the target instance. Do not connect to the recovery catalog as shown in the following example:

```
$ rman TARGET /
```
3. Set the database identifier (DBID). You can find the DBID of your source database by querying the DBID column in V\$DATABASE.

```
RMAN> SET DBID 1090770270;
```
4. Start the instance in NOMOUNT mode:

```
RMAN> STARTUP NOMOUNT
```

You will receive an error similar to the following because the server parameter file has not been restored. RMAN uses a “dummy” parameter file to start the instance.

```
startup failed: ORA-01078: failure in processing system
parameters
```

Restoring the Database to a New Host (continued)

5. Restore the server parameter file from the backup sets and shut down the instance as shown in the example:

```
RESTORE SPFILE TO PFILE '?/oradata/test/initiorcl.ora' FROM  
AUTOBACKUP;
```

6. Shut down the instance:

```
SHUTDOWN IMMEDIATE;
```

7. Edit the restored initialization parameter file to change any location-specific parameters, such as those ending in `_DEST`, to reflect the new directory structure.

8. Start the instance in NOMOUNT mode using your edited text initialization parameter file.

```
RMAN> STARTUP NOMOUNT  
      > PFILE='?/oradata/test/initiorcl.ora';
```


Restoring the Database to a New Host

9. Create a RUN block to:
 - Restore the control file
 - Mount the database
10. Create the RMAN recovery script to restore and recover the database.
11. Execute the RMAN script.
12. Open the database with the RESETLOGS option.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Restoring the Database to a New Host (continued)

9. Create a RUN block to restore the control file from an autobackup and mount the database as shown in the example:

```
RUN
{
  RESTORE CONTROLFILE FROM AUTOBACKUP;
  ALTER DATABASE MOUNT;
}
```
10. Query V\$DATAFILE on your new host to determine the database file names as recorded in the control file. Create the RMAN recovery script to restore and recover the database, including the following steps as appropriate:
 - a. Use the SET NEWNAME command to specify the path on your new host for each of the data files that is restored to a different destination than on the original host.
 - b. Use the SQL ALTER DATABASE RENAME FILE command to specify the path for the online redo log files.
 - c. Include the SET UNTIL command to limit recovery to the end of the archived redo log files.
 - d. Include the SWITCH command so that the control file recognizes the new path names as the correct names for the data files.

Restoring the Database to a New Host (continued)

An example of a recovery script follows:

```
RUN
{
SET NEWNAME FOR DATAFILE 1 TO '?/oradata/test/system01.dbf';
SET NEWNAME FOR DATAFILE 2 TO '?/oradata/test/undotbs01.dbf';
SET NEWNAME FOR DATAFILE 3 TO '?/oradata/test/sysaux.dbf';
SET NEWNAME FOR DATAFILE 4 TO '?/oradata/test/users01.dbf';
SET NEWNAME FOR DATAFILE 5 TO '?/oradata/test/example01.dbf';
SQL "ALTER DATABASE RENAME FILE
''/u01/app/oracle/oradata/orcl/redo01.log''
TO ''/?/oradata/test/redo01.log'' ";
SQL "ALTER DATABASE RENAME FILE
''/u01/app/oracle/oradata/orcl/redo02.log''
TO ''/?/oradata/test/redo02.log'' ";
SQL "ALTER DATABASE RENAME FILE
''/u01/app/oracle/oradata/orcl/redo03.log''
TO ''/?/oradata/test/redo03.log'' ";
SET UNTIL SCN 4545727;
RESTORE DATABASE;
SWITCH DATAFILE ALL;
RECOVER DATABASE;
}
```

11. Execute the recovery script.

12. Open the database with the RESETLOGS option:

```
RMAN> ALTER DATABASE OPEN RESETLOGS;
```

After you have completed your test, you can shut down the test database instance and delete the test database with all its files.

Performing Disaster Recovery

- Disaster implies the loss of the entire target database, the recovery catalog database, all current control files, all online redo log files, and all parameter files.
- Disaster recovery includes the restoration and recovery of the target database.
- Minimum required set of backups:
 - Backups of data files
 - Corresponding archived redo logs files
 - At least one control file autobackup

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Disaster Recovery

Disaster recovery includes the restoration and recovery of the target database after the loss of the entire target database, all current control files, all online redo log files, all parameter files, and the recovery catalog database (if applicable).

To perform disaster recovery, the following backups are required as a minimum:

- Backups of data files
- Corresponding archived redo logs generated after the time of the backup
- At least one autobackup of the control file

Note: Refer to the *Oracle Data Guard Concepts and Administration* manual for information about how Oracle Data Guard can provide complete disaster protection.

Performing Disaster Recovery

Basic procedure:

- Restore an autobackup of the server parameter file.
- Start the target database instance.
- Restore the control file from autobackup.
- Mount the database.
- Restore the data files.
- Recover the data files.
- Open the database with the `RESETLOGS` option.

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Performing Disaster Recovery (continued)

The basic procedure for performing disaster recovery is outlined in the slide. After you have mounted the database, follow the steps for performing recovery with a backup control file.

Quiz

When you have lost no data files and you recover the backup control file, why is the `RECOVER` command required?

1. To roll forward changes to the control file by resynchronizing from the data files
2. To roll forward changes to the control file by applying redo from the redo logs
3. To roll forward changes to the control file by using the RMAN catalog

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Answer: 2

Quiz

With the `RESTORE` command, you restore database files from backup, but you do not apply redo from redo logs.

1. True
2. False

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Answer: 1

Summary

In this lesson, you should have learned how to use RMAN to do the following:

- Perform complete recovery when a critical or noncritical data file is lost
- Recover using incrementally updated backups
- Switch to image copies for fast recovery
- Restore a database onto a new host
- Recover using a backup control file

ORACLE

Copyright © 2009, Oracle. All rights reserved.

Practice 7 Overview: Using RMAN to Perform Recovery

This practice covers the following topics:

- Recovering image copies
- Performing fast recovery

ORACLE

Copyright © 2009, Oracle. All rights reserved.