**BACKUP AND RECOVERY**

**VALIDATE DATABASE**

This example validates the database and tries to read all specified data files (sample

output included).

RMAN> VALIDATE DATABASE;

Starting validate at 20-OCT-13

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: SID=90 device type=DISK

could not read file header for datafile 7 error reason 4

RMAN-00571: ===========================================================

RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============

RMAN-00571: ===========================================================

RMAN-03002: failure of backup command at 10/20/2013 13:05:43

RMAN-06056: could not access datafile 7

This example shows that Data File 7 is corrupted.

You can then run the REPORT SCHEMA command to obtain the tablespace name and file

name for data file 7 as follows (sample output included):

RMAN> REPORT SCHEMA;

Report of database schema for database with db\_unique\_name RDBMS

List of Permanent Datafiles

===========================

File Size(MB) Tablespace RB segs Datafile Name

---- -------- -------------------- ------- ------------------------

1 450 SYSTEM \*\*\* +DATAFILE/tbs\_01.f

2 86 SYSAUX \*\*\* +DATAFILE/tbs\_ax1.f

3 15 UD1 \*\*\* +DATAFILE/tbs\_undo1.f

4 2 SYSTEM \*\*\* +DATAFILE/tbs\_02.f

5 2 TBS\_1 \*\*\* +DATAFILE/tbs\_11.f

6 2 TBS\_1 \*\*\* +DATAFILE/tbs\_12.f

7 2 TBS\_2 \*\*\* +DATAFILE/tbs\_21.f

List of Temporary Files

=======================

File Size(MB) Tablespace Maxsize(MB) Tempfile Name

---- -------- -------------------- ----------- --------------------

1 40 TEMP 32767 +DATAFILE/tbs\_tmp1.f

Identifying Data Files with SQL

**To determine whether data files require media recovery:**

**1.** Start SQL\*Plus and connect to the target database instance with administrator privileges.

**2.** Determine the status of the database by executing the following SQL query:

SELECT STATUS FROM V$INSTANCE;

If the status is OPEN, then the database is open. Nevertheless, some data files may

require media recovery.

**3.** Query V$DATAFILE\_HEADER to determine the status of your data files. Run the following:

SQL statements to check the data file headers:

SELECT FILE#, STATUS, ERROR, RECOVER, TABLESPACE\_NAME, NAME

FROM V$DATAFILE\_HEADER

WHERE RECOVER = 'YES'

OR (RECOVER IS NULL AND ERROR IS NOT NULL);

If ERROR is not NULL, then the data file header cannot be read and validated. Check for a

temporary hardware or operating system problem causing the error. If there is no such

problem, then you must restore the file or switch to a copy.

If the ERROR column is NULL and the RECOVER column is YES, then the file requires media

recovery (and may also require a restore from backup).

**4.** Optionally, query V$RECOVER\_FILE to list data files requiring recovery by data file number

with their status and error information. For example, execute the following query:

SELECT FILE#, ERROR, ONLINE\_STATUS, CHANGE#, TIME

FROM V$RECOVER\_FILE;

.

To find data file and tablespace names, you can also perform joins using the data

file number and the V$DATAFILE and V$TABLESPACE views, as shown in the

following example.

SELECT r.FILE# AS df#, d.NAME AS df\_name, t.NAME AS tbsp\_name,

d.STATUS, r.ERROR, r.CHANGE#, r.TIME

FROM V$RECOVER\_FILE r, V$DATAFILE d, V$TABLESPACE t

WHERE t.TS# = d.TS#

AND d.FILE# = r.FILE#;

------------------------------------------------------------------------------------------------------------------------------------

Previewing Backups Used in Restore Operations

You can apply RESTORE ... PREVIEW to any RESTORE operation to create a detailed list of every

backup to be used in the requested RESTORE operation, and the necessary target SCN for

recovery after the RESTORE operation is complete.

When planning your restore and recovery operation, use RESTORE ... PREVIEW or

RESTORE ... VALIDATE HEADER to ensure that all required backups are available or to identify

situations in which you may want to direct RMAN to use or avoid specific backups.

**To preview backups to be used in a restore operation:**

**1.** Run a RESTORE command with the PREVIEW option.

For example, run one of the following commands:

RESTORE DATABASE PREVIEW;

RESTORE ARCHIVELOG FROM TIME 'SYSDATE-7' PREVIEW;

If the report produced by RESTORE ... PREVIEW provides too much information, then

specify the SUMMARY option as shown in the following example:

RESTORE DATABASE PREVIEW SUMMARY

---------------------------------------------------------------------------------------------------------------------------------

Validating Backups Before Restoring Them

You can run RMAN commands to test the

availability of usable backups for any RESTORE operation.

RESTORE ... VALIDATE to test whether RMAN can restore a specific object from a backup.

RMAN chooses which backups to use.

• VALIDATE BACKUPSET to test the validity of a backup set that you specify.

-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Restoring Archived Redo Logs to a New Location

**To restore archived redo logs to a new location:**

**1.** Start RMAN and connect to a target database, as described in "Making Database

Connections with RMAN".

**2.** Ensure that the database is mounted or open.

**3.** Perform the following operations within a RUN command:

**a.** Specify the new location for the restored archived redo logs using SET

ARCHIVELOG DESTINATION.

**b.** Either explicitly restore the archived redo logs or execute commands that

automatically restore the logs.

The following sample RUN command explicitly restores all backup archived logs to

a new location:

RUN

{

SET ARCHIVELOG DESTINATION TO '/oracle/temp\_restore';

Chapter 17

Preparing for Complete Database Recovery

17-10

RESTORE ARCHIVELOG ALL;

# restore and recover data files as needed

.

.

.

}

The following example sets the archived log destination and then uses RECOVER

DATABASE to restore archived logs from this destination automatically:

RUN

{

SET ARCHIVELOG DESTINATION TO '/oracle/temp\_restore';

RESTORE DATABASE;

RECOVER DATABASE; # restores and recovers logs automatically

---------------------------------------------------------------------------------------------------------

Performing Complete Recovery of the Whole Database

This scenario assumes that database trgt has lost most or all of its data files. It also

assumes that the database uses a fast recovery area.

**To restore and recover the whole database:**

Performing Complete Database Recovery

**1.** Complete the preparation steps required for your scenario, as described in

"Preparing for Complete Database Recovery".

**2.** Start RMAN and connect to a target database and to a recovery catalog (if used),

as described in "Making Database Connections with RMAN".

RMAN displays the database status when it connects: not started, not mounted,

not open (when the database is mounted but not open), or none (when the

database is open).

**3.** If the database is not mounted, then mount but do not open the database.

For example, enter the following command:

STARTUP MOUNT;

**4.** Use the SHOW command to see which channels are preconfigured.

For example, enter the following command (sample output is included):

SHOW ALL;

RMAN configuration parameters for database with db\_unique\_name PROD1 are:

...

CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default

CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE DEVICE TYPE SBT\_TAPE PARALLELISM 1 BACKUP TYPE TO BACKUPSET; #

default

CONFIGURE CHANNEL DEVICE TYPE 'SBT\_TAPE' PARMS "SBT\_

LIBRARY=/usr/local/oracle/backup/lib/libobk.so";

If the necessary devices and channels are configured, then no action is necessary.

Otherwise, you can use the CONFIGURE command to configure automatic channels,

or include ALLOCATE CHANNEL commands within a RUN block.

**5.** Restore and recover the database. Do one of the following:

RMAN> **RESTORE DATABASE;**

Starting restore at 20-JUN-13

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: SID=35 device type=DISK

allocated channel: ORA\_SBT\_TAPE\_1

channel ORA\_SBT\_TAPE\_1: SID=34 device type=SBT\_TAPE

channel ORA\_SBT\_TAPE\_1: Oracle Secure Backup

channel ORA\_DISK\_1: starting datafile backup set restore

channel ORA\_DISK\_1: specifying datafile(s) to restore from backup set

channel ORA\_DISK\_1: restoring datafile 00001 to /disk1/oracle/dbs/

tbs\_01.f

channel ORA\_DISK\_1: restoring datafile 00002 to /disk1/oracle/dbs/

tbs\_ax1.f

..

.

Finished restore at 20-JUN-13

RMAN> **RECOVER DATABASE;**

Starting recover at 20-JUN-13

using channel ORA\_DISK\_1

allocated channel: ORA\_SBT\_TAPE\_1

channel ORA\_SBT\_TAPE\_1: SID=34 device type=SBT\_TAPE

channel ORA\_SBT\_TAPE\_1: Oracle Secure Backup

starting media recovery

channel ORA\_DISK\_1: starting archived log restore to default destination

channel ORA\_DISK\_1: restoring archived log

archived log thread=1 sequence=5

channel ORA\_DISK\_1: restoring archived log

archived log thread=1 sequence=6

...

channel ORA\_DISK\_1: reading from backup piece

/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_annnn\_

TAG20130620T113128\_29jhr197\_.bkp

channel ORA\_DISK\_1: piece

handle=/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_annnn\_

TAG20130620T113128\_29jhr197\_.bkp tag=TAG20130620T113128

channel ORA\_DISK\_1: restored backup piece 1

channel ORA\_DISK\_1: restore complete, elapsed time: 00:00:02

archived log file name=/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_

20/o1\_mf\_1\_5\_29jhv47k\_.arc thread=1 sequence=5

channel default: deleting archived log(s)

...

media recovery complete, elapsed time: 00:00:15

Finished recover at 20-JUN-13

If you manually allocate channels, then you must issue the RESTORE and RECOVER

commands together within a RUN block as shown in the following example:

RUN

{

ALLOCATE CHANNEL c1 DEVICE TYPE sbt;

RESTORE DATABASE;

RECOVER DATABASE;

}

The following example restores the database, specifying new names for three of the

data files, and then recovers the database:

RUN

{

SET NEWNAME FOR DATAFILE 2 TO '/disk2/df2.dbf';

SET NEWNAME FOR DATAFILE 3 TO '/disk2/df3.dbf';

SET NEWNAME FOR DATAFILE 4 TO '/disk2/df4.dbf';

Chapter 17

Performing Complete Database Recovery

17-15

RESTORE DATABASE;

SWITCH DATAFILE ALL;

RECOVER DATABASE;

}

--------------------------------------------------------------------------------------------------------------------------------

Performing Complete Recovery of a Tablespace

Use the RESTORE and RECOVER commands with the TABLESPACE option to perform

complete recovery of a tablespace

In the basic scenario, the database is open, and some but not all of the data files are

damaged. You want to restore and recover the damaged tablespace while leaving the

database open so that the rest of the database remains available

**To restore and recover a tablespace:**

**1.** Complete the preparation steps that are required for your recovery scenario as

described in "Preparing for Complete Database Recovery".

**2.** Start RMAN and connect to a target database and to a recovery catalog (if used),

as described in "Making Database Connections with RMAN".

**3.** If the database is open, then take the tablespace requiring recovery offline.

For example, enter the following command to take USERS offline:

ALTER TABLESPACE users OFFLINE IMMEDIATE;

**4.** Use the SHOW command to see which channels are preconfigured.

For example, enter the following command (sample output is included):

SHOW ALL;

RMAN configuration parameters for database with db\_unique\_name PROD1 are:

...

CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default

CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE DEVICE TYPE SBT\_TAPE PARALLELISM 1 BACKUP TYPE TO BACKUPSET; #

default

CONFIGURE CHANNEL DEVICE TYPE 'SBT\_TAPE' PARMS "SBT\_

LIBRARY=/usr/local/oracle/backup/lib/libobk.so";

If the necessary devices and channels are configured, then no action is necessary.

Otherwise, you can use the CONFIGURE command to configure automatic channels,

or include ALLOCATE CHANNEL commands within a RUN block.

**5.** Restore and recover the tablespace. Do one of the following:

Chapter 17

Performing Complete Database Recovery

17-16

• If you are restoring data files to their original locations, then run the RESTORE

TABLESPACE and RECOVER TABLESPACE commands at the RMAN prompt.

For example, enter the following command if automatic channels are configured

(sample output included):

RMAN> **RESTORE TABLESPACE users;**

Starting restore at 20-JUN-13

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: SID=37 device type=DISK

allocated channel: ORA\_SBT\_TAPE\_1

channel ORA\_SBT\_TAPE\_1: SID=38 device type=SBT\_TAPE

channel ORA\_SBT\_TAPE\_1: Oracle Secure Backup

channel ORA\_DISK\_1: starting datafile backup set restore

channel ORA\_DISK\_1: specifying datafile(s) to restore from backup set

channel ORA\_DISK\_1: restoring datafile 00012 to /disk1/oracle/dbs/users01.f

channel ORA\_DISK\_1: restoring datafile 00013 to /disk1/oracle/dbs/users02.f

channel ORA\_DISK\_1: restoring datafile 00021 to /disk1/oracle/dbs/users03.f

channel ORA\_DISK\_1: reading from backup piece

/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_nnndf\_

TAG20130620T105435\_29jflwor\_.bkp

channel ORA\_DISK\_1: piece

handle=/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_nnndf\_

TAG20130620T105435\_29jflwor\_.bkp tag=TAG20130620T105435

channel ORA\_DISK\_1: restored backup piece 1

channel ORA\_DISK\_1: restore complete, elapsed time: 00:00:01

Finished restore at 20-JUN-13

RMAN> **RECOVER TABLESPACE users;**

Starting recover at 20-JUN-13

using channel ORA\_DISK\_1

using channel ORA\_SBT\_TAPE\_1

starting media recovery

archived log for thread 1 with sequence 27 is on disk as file

/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_20/o1\_mf\_1\_27\_29jjmtc9\_.arc

archived log for thread 1 with sequence 28 is on disk as file

/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_20/o1\_mf\_1\_28\_29jjnc5x\_.arc

...

channel ORA\_DISK\_1: starting archived log restore to default destination

channel ORA\_DISK\_1: restoring archived log

archived log thread=1 sequence=5

channel ORA\_DISK\_1: restoring archived log

archived log thread=1 sequence=6

channel ORA\_DISK\_1: restoring archived log

archived log thread=1 sequence=7

...

channel ORA\_DISK\_1: reading from backup piece

/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_annnn\_

TAG20130620T113128\_29jhr197\_.bkp

channel ORA\_DISK\_1: piece

handle=/disk1/oracle/work/orcva/TKRM/backupset/2013\_06\_20/o1\_mf\_annnn\_

TAG20130620T113128\_29jhr197\_.bkp tag=TAG20130620T113128

Chapter 17

Performing Complete Database Recovery

17-17

channel ORA\_DISK\_1: restored backup piece 1

channel ORA\_DISK\_1: restore complete, elapsed time: 00:00:02

archived log file name=/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_

20/o1\_mf\_1\_5\_29jkdvjq\_.arc thread=1 sequence=5

channel default: deleting archived log(s)

archived log file name=/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_

20/o1\_mf\_1\_5\_29jkdvjq\_.arc RECID=91 STAMP=593611179

archived log file name=/disk1/oracle/work/orcva/TKRM/archivelog/2013\_06\_

20/o1\_mf\_1\_6\_29jkdvbz\_.arc thread=1 sequence=6

channel default: deleting archived log(s)

...

media recovery complete, elapsed time: 00:00:01

Finished recover at 20-JUN-13

The following example restores the data files in tablespace users to a new

location, and then performs recovery. Assume that the old data files were

stored in the /disk1 path and the new ones will be stored in the /disk2 path.

RUN

{

# specify the new location for each datafile

SET NEWNAME FOR DATAFILE '/disk1/oracle/dbs/users01.f' TO

'/disk2/users01.f';

SET NEWNAME FOR DATAFILE '/disk1/oracle/dbs/users02.f' TO

'/disk2/users02.f';

SET NEWNAME FOR DATAFILE '/disk1/oracle/dbs/users03.f' TO

'/disk2/users03.f';

RESTORE TABLESPACE users;

SWITCH DATAFILE ALL; # update control file with new file names

RECOVER TABLESPACE users;

}

**6.** Examine the output to see if recovery was successful. If so, bring the recovered

tablespace back online.

For example, enter the following command:

ALTER TABLESPACE users ONLINE;

-----------------------------------------------------------------------------------------------------------

Recovering Tables and Table Partitions

The RMAN RECOVER command enables you to recover tables and table partitions from RMAN

backups.

Steps Performed By RMAN to Recover Tables and Table Partitions

RMAN performs a series of steps while automating the process of recovering tables or table

partitions from an RMAN backup.

The steps include the following:

**1.** Determines which backup contains the tables or table partitions that need to be

recovered, based on the point in time specified for the recovery.

**2.** Determines if there is sufficient space on the target host to create the auxiliary instance

that will be used during the table or partition recovery process.

If the required space is not available, then RMAN displays an error and exits the recovery

operation.

**3.** Creates an auxiliary database on the target host and recovers the specified tables or

table partitions, until the specified point in time, into this auxiliary database.

You can specify the location on the target host to which the recovered data files are

stored in the auxiliary database.

**4.** Creates a Data Pump export dump file that contains the recovered tables or table

partitions.

You can specify the name and the location of the Data Pump export dump file used to

store the metadata of the recovered tables or table partitions.

**5.** (Optional) Imports the Data Pump export dump file into the target instance.

You can choose not to import the export dump file that contains the recovered tables or

table partitions into the target database. If you do not import the export dump file as part

of the recovery process, you must manually import it later using the Data Pump Import

utility.

Prerequisites for Recovering Tables and Table Partitions from

RMAN Backups

Recovering Tables and Table Partitions in PDBs

**1.** Perform the planning tasks described in "Preparing to Recover Tables and Table

Partitions".

**2.** Start RMAN and connect to the root as a user with the SYSDBA or SYSBACKUP

privilege, as described in "Making RMAN Connections to a CDB".

**3.** Recover the tables or table partitions to the specified point in time by using the

RECOVER TABLE ... OF PLUGGABLE DATABASE command.

You must use the AUXILIARY DESTINATION clause and one of the following

clauses: UNITL TIME, UNTIL SCN, or UNTIL SEQUENCE.

Depending on your requirements, you may also need to use the one or more of

the following clauses: DUMP FILE, DATAPUMP DESTINATION, NOTABLEIMPORT, REMAP

TABLE, or REMAP TABLESPACE.

The following command recovers the table PDB\_EMP in the PDB HR\_PDB to the state

that it was in 4 days before the current date. HR is the name of the schema that

contains the table. The recovered table is renamed to EMP\_RECVR.

RECOVER TABLE HR.PDB\_EMP OF PLUGGABLE DATABASE HR\_PDB

UNTIL TIME 'SYSDATE-4'

AUXILIARY DESTINATION '/tmp/backups'

REMAP TABLE 'HR'.'PDB\_EMP':'EMP\_RECVR';

Example: Recovering Tables to a Specified Point in Time

This example recovers multiple tables to a specified point in time that is represented

using SYSDATE.

Assume that you want to recover two tables EMP and DEPT to the state they were in two

days ago, before some logical corruption occurred. However, you do not want RMAN

to import these tables into the target database. RMAN must only create the export

dump file, called emp\_dept\_exp\_dump.dat, in the location /tmp/recover/dumpfiles.

Using NOTABLEIMPORT indicates that these tables must not be imported into the target

database. You can import these tables, when required, by using the Data Pump import

utility. The auxiliary destination used during the recovery process is /tmp/oracle/

recover.

To recover tables EMP and DEPT without importing them into the target database:

**1.** Perform the planning tasks described in "Preparing to Recover Tables and Table

Partitions".

In this example, you need to recover tables to a point in time specified by an expression

that uses SYSDATE. However, the recovered tables must not be imported in to the target

database.

**2.** Start an RMAN session and connect as TARGET to the target database as described in

"Making Database Connections with RMAN".

**3.** Recover the tables EMP and DEPT using the following clauses in the RECOVER command:

DATAPUMP DESTINATION, DUMP FILE, REMAP TABLE, and NOTABLEIMPORT.

The following RECOVER command recovers the EMP and DEPT tables.

RECOVER TABLE SCOTT.EMP, SCOTT.DEPT

UNTIL TIME 'SYSDATE-1'

AUXILIARY DESTINATION '/tmp/oracle/recover'

DATAPUMP DESTINATION '/tmp/recover/dumpfiles'

DUMP FILE 'emp\_dept\_exp\_dump.dat'

NOTABLEIMPORT;

Example: Recovering Table Partitions to a Specified Log Sequence

Number

**1.** Perform the planning tasks described in "Preparing to Recover Tables and Table

Partitions".

In this example, you need to recover two table partitions to a specified log sequence

number and then import these partitions into the target database.

**2.** Start an RMAN session and connect as TARGET to the target database as described in

**3.** Recover partitions using the following RECOVER command with the REMAP TABLE and

REMAP TABLESPACE clauses.

RECOVER TABLE SH.SALES:SALES\_1998, SH.SALES:SALES\_1999

UNTIL SEQUENCE 354

AUXILIARY DESTINATION '/tmp/oracle/recover'

REMAP TABLE 'SH'.'SALES':'SALES\_1998':'HISTORIC\_SALES\_1998',

Chapter 22

'SH'.'SALES':'SALES\_1999':'HISTORIC\_SALES\_1999'

REMAP TABLESPACE 'SALES\_TS':'SALES\_PRE\_2000\_TS';

In this case, the specified table partitions are imported as separate tables, called

historic\_sales\_1998 and historic\_sales\_1999, into the sales\_pre\_2000\_ts

tablespace of the target database. The REMAP TABLE clause specifies the names

used for the imported tables. The auxiliary destination used during the recovery

process is /tmp/oracle/recover.

If you omit the REMAP TABLE clause, RMAN uses default names for the imported

tables. The name is a combination of the original table name and the partition

name.

Example: Recovering a Table into a New Schema

This example recovers multiple tables into a new schema that is different from the

source schema.

In this example, the HR.DEPARTMENTS and SH.CHANNELS tables need to be recovered to

the state that they were in one day ago, before a logical corruption occurred. The

recovered tables must be renamed as NEW\_DEPARTMENTS and NEW\_CHANNELS and

imported into the EXAMPLE schema. The schema EXAMPLE exists at the time this

example is run.

The REMAP TABLE clause is used to indicate how the source schema is mapped to a

new target schema. The auxiliary destination used during the recovery process

is /tmp/auxdest.

**1.** Perform the planning tasks required to recover tables from RMAN backups. In this

example, you need to recover tables to a point in time specified by an expression

that uses SYSDATE.

**2.** Start an RMAN session and connect to the target database as described in

**3.** Recover the HR.DEPARTMENTS and SH.CHANNELS tables, rename them to

NEW\_DEPARTMENTS and NEW\_CHANNELS respectively, and then import them into the

EXAMPLE schema.

The following RECOVER command performs the required table recovery:

RECOVER TABLE HR.DEPARTMENTS, SH.CHANNELS

UNTIL TIME 'SYSDATE – 1'

AUXILIARY DESTINATION '/tmp/auxdest'

REMAP TABLE hr.departments:example.new\_departments,

sh.channels:example.new\_channels;

-----------------------------------------------------------------------------------------------------

Performing Complete Recovery of the Root

You might consider recovering only the root if a data corruption or user error occurs

that affects only the root.

However, Oracle strongly recommends that you recover all PDBs after recovering the

root to prevent metadata inconsistencies among the root and the PDBs. In this case, it

might be preferable to perform a complete recovery of the whole CDB.

**To recover the root:**

**1.** Complete the preparation steps that are required for your recovery scenario, as

**2.** Start RMAN and connect to the root as a common user with the SYSDBA or

SYSBACKUP privilege, as described in "Connecting as Target to the Root".

**3.** Place the CDB in mounted mode.

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

**4.** (Optional) Use the CONFIGURE command to configure the default device type and

automatic channels.

**5.** Restore and recover the root with the following commands:

Chapter 17

Performing Complete Recovery of CDBs

17-22

RESTORE DATABASE ROOT;

RECOVER DATABASE ROOT;

**6.** Examine the output to see if media recovery was successful. If so, proceed to the next

step.

**7.** (Strongly recommended) Recover all PDBs, including the CDB seed.

**a.** Issue the RESTORE PLUGGABLE DATABASE and RECOVER PLUGGABLE DATABASE

commands.

The following example recovers the PDBs sales and hr:

RESTORE PLUGGABLE DATABASE 'PDB$SEED', sales, hr;

RECOVER PLUGGABLE DATABASE 'PDB$SEED', sales, hr;

**b.** Examine the output to see if media recovery was successful. If so, proceed to the

next step.

**8.** Open the CDB and all PDBs.

ALTER DATABASE OPEN;

ALTER PLUGGABLE DATABASE ALL OPEN;

-------------------------------------------------------------------------------------------

Performing Complete Recovery of PDBs with RMAN

You can perform complete recovery of one or more PDBs without affecting operations of

other open PDBs

**To recover one or more PDBs while connected to the root:**

**1.** Complete the preparation steps that are required for your recovery scenario, as

described in "Preparing for Complete Database Recovery".

**2.** Start RMAN. Connect to the root as a common user with the SYSDBA or SYSBACKUP

privilege and to a recovery catalog (if used), as described in "Connecting as Target

to the Root".

**3.** Close the PDBs that you want to recover.

ALTER PLUGGABLE DATABASE sales, hr CLOSE;

If any data files are missing, an error occurs and you cannot close a PDB. You

must then connect to the PDB to which the missing data file belongs, take the

missing data file offline, and then close the PDB.

The following command takes the data file 12 offline:

ALTER PLUGGABLE DATABASE DATAFILE 12 OFFLINE;

**Note:**

If the data files that store the SYSTEM tablespace of a PDB are missing,

then follow the recovery steps that are described in "Performing

Complete Recovery of Tablespaces or Data Files in a PDB with RMAN".

**4.** (Optional) Use the CONFIGURE command to configure the default device type and

automatic channels.

**5.** Issue the RESTORE PLUGGABLE DATABASE and RECOVER PLUGGABLE DATABASE

commands.

The following example recovers the CDB seed, PDB$SEED, and the PDBs sales

and hr:

RESTORE PLUGGABLE DATABASE 'pdb$seed', sales, hr;

RECOVER PLUGGABLE DATABASE 'pdb$seed', sales, hr;

**6.** If any data files were taken offline in Step 2, make these data files online.

Connect to the PDB to which the missing data file belongs and then make the data

file online. The following command makes the data file 12 online:

ALTER DATABASE DATAFILE 12 ONLINE;

**7.** Examine the output to see if media recovery was successful. If so, open the PDBs.

ALTER PLUGGABLE DATABASE sales, hr OPEN;

**To connect to and recover one PDB:**

**1.** Complete the preparation steps that are required for your recovery scenario, as

**2.** Start RMAN. Connect to the PDB as a local user with the SYSDBA system privilege

**3.** Close the PDB.

ALTER PLUGGABLE DATABASE CLOSE;

If any data files are missing, an error occurs and you cannot close the PDB. You must

take the missing data file offline and then close the PDB.

The following command takes the data file 12 offline:

ALTER DATABASE DATAFILE 12 OFFLINE;

**Note:**

If the data files that store the SYSTEM tablespace of a PDB are missing, then

follow the recovery steps described in "Performing Complete Recovery of

Tablespaces or Data Files in a PDB with RMAN".

**4.** (Optional) Use the CONFIGURE command to configure the default device type and

automatic channels.

**5.** Issue the RESTORE DATABASE and RECOVER DATABASE commands.

RESTORE DATABASE;

RECOVER DATABASE;

**6.** If any data files were taken offline in Step 2, make these data files online.

The following command makes the data file 12 online:

ALTER DATABASE DATAFILE 12 ONLINE;

**7.** Open the PDB.

ALTER PLUGGABLE DATABASE OPEN;

---------------------------------------------------------------------------------------------

Performing Complete Recovery of Tablespaces or Data Files in

a PDB with RMAN

You must connect directly to a PDB to recover one or more of its

tablespaces. In contrast, because data file numbers and paths are unique across the

CDB, you can connect either to the root or to a PDB when recovering PDB data files.

If you connect to the root, you can recover data files from multiple PDBs with a single

command. If you connect to a PDB, you can recover only data files in that PDB.

**To restore and recover a non-SYSTEM tablespace in a PDB:**

**1.** Complete the preparation steps that are required for your recovery scenario

**2.** Start RMAN. Connect to a target database and to a recovery catalog (if used

**3.** If the database is open, then take the tablespace requiring recovery offline.

For example, enter the following command to take the USERS tablespace offline:

ALTER TABLESPACE users OFFLINE IMMEDIATE;

**4.** Use the SHOW command to see which channels are preconfigured.

If the necessary devices and channels are configured, then no action is necessary.

Otherwise, you can use the CONFIGURE command to configure automatic channels,

or include ALLOCATE CHANNEL commands within a RUN block.

**5.** Restore and recover the tablespace. Do one of the following:

• If you are restoring data files to their original locations, then run the RESTORE

TABLESPACE and RECOVER TABLESPACE commands at the RMAN prompt.

For example, enter the following commands if automatic channels are configured:

RMAN> RESTORE TABLESPACE users;

RMAN> RECOVER TABLESPACE users;

• If you are restoring some data files to new locations, then execute RESTORE

TABLESPACE and RECOVER TABLESPACE in a RUN command. Use the SET NEWNAME

command to rename data files.

**6.** Examine the output to see if recovery was successful. If so, bring the recovered

tablespace back online.

For example, enter the following command:

ALTER TABLESPACE users ONLINE;

**To restore and recover the SYSTEM tablespace in a PDB:**

**1.** Complete the preparation steps that are required for your recovery scenario

**2.** Start RMAN. Connect to the root as a common user with the SYSDBA or SYSBACKUP

privilege and to a recovery catalog (if used),

**3.** Shut down the CDB and restart it in mount mode.

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

**4.** Restore and recover the data files that store the SYSTEM tablespace of the affected PDB.

RESTORE DATAFILE 2,3;

RECOVER DATAFILE 2,3;

**5.** Open all the PDBs in the CDB.

ALTER PLUGGABLE DATABASE ALL OPEN READ WRITE;

**To recover non-SYSTEM data files in a PDB:**

**1.** Complete the preparation steps that are required for your recovery scenario, as

**2.** Do one of the following:

• Start RMAN. Connect to the root as a common user with the SYSDBA or SYSBACKUP

privilege and to a recovery catalog (if used),

• Start RMAN. Connect to the PDB as a local user with the SYSDBA privilege and to a

recovery catalog (if used), as described in "Connecting as Target to a PDB".

**3.** Issue the RESTORE DATAFILE and RECOVER DATAFILE commands.

RESTORE DATAFILE 10, 13;

RECOVER DATAFILE 10, 13;

Performing Flashback and Database Point-in-

Time Recovery

Performing a Flashback Table Operation

**1.** Ensure that the prerequisites that are described in "Prerequisites for Flashback Table"

are met.

**2.** Connect SQL\*Plus to the target database and identify the current SCN.

You cannot roll back a FLASHBACK TABLE statement, but you can issue another FLASHBACK

TABLE statement and specify a time just before the current time. Therefore, it is advisable

to record the current SCN. You can obtain it by querying V$DATABASE as follows:

SELECT CURRENT\_SCN FROM V$DATABASE;

**3.** Identify the time, SCN, or restore point to which you want to return the table.

If you have created restore points, then you can list available restore points by executing

the following query:

SELECT NAME, SCN, TIME FROM V$RESTORE\_POINT;

**4.** Ensure that enough undo data exists to rewind the table to the specified target.

If the UNDO\_RETENTION initialization parameter is set, and the undo retention

guarantee is on, then you can use the following query to determine how long undo

data is being retained:

SELECT NAME, VALUE/60 MINUTES\_RETAINED

FROM V$PARAMETER

WHERE NAME = 'undo\_retention';

**5.** Ensure that row movement is enabled for all objects that you are rewinding with

Flashback Table.

You can enable row movement for a table with the following SQL statement:

ALTER TABLE hr.temp\_employees ENABLE ROW MOVEMENT;

**6.** Determine whether the table that you intend to flash back has dependencies on

other tables. If dependencies exist, then decide whether to flash back these tables

as well.

You can issue the following SQL query to determine the dependencies, where

*schema\_name* is the schema for the table to be flashed back and *table\_name* is

the name of the table:

SELECT other.owner, other.table\_name

FROM sys.all\_constraints this, sys.all\_constraints other

WHERE this.owner = *schema\_name*

AND this.table\_name = *table\_name*

AND this.r\_owner = other.owner

AND this.r\_constraint\_name = other.constraint\_name

AND this.constraint\_type='R';

**7.** Execute a FLASHBACK TABLE statement for the objects to flash back.

The following SQL statement returns the hr.temp\_employees table to the restore

point named temp\_employees\_update:

FLASHBACK TABLE hr.temp\_employees

TO RESTORE POINT temp\_employees\_update;

The following SQL statement rewinds the hr.temp\_employees table to its state

when the database was at the time specified by the SCN:

FLASHBACK TABLE hr.temp\_employees

TO SCN 123456;

As shown in the following example, you can also specify the target point in time

with TO\_TIMESTAMP:

FLASHBACK TABLE hr.temp\_employees

TO TIMESTAMP TO\_TIMESTAMP('2013-10-17 09:30:00', 'YYYY-MM-DD HH:MI:SS');

**Note:**

The mapping of time stamps to SCNs is not always exact. When you use

time stamps with the FLASHBACK TABLE statement, the time to which the

table is flashed back can vary by up to approximately 3 seconds of the

time specified for TO\_TIMESTAMP. If an exact point in time is required, then

use an SCN rather than a time.

Keeping Triggers Enabled During Flashback Table

By default, the database disables triggers on the affected table before performing a

FLASHBACK TABLE operation. After the operation, the database returns the triggers to the state

they were in before the operation (enabled or disabled).

To keep triggers enabled during the flashback of the table, add an ENABLE TRIGGERS clause to

the FLASHBACK TABLE statement.

For example, assume that at 17:00 an HR

For example, assume that at 17:00 an HR administrator discovers that an employee is

missing from the hr.temp\_employees table. This employee was included in the table at 14:00,

the last time the report was run. Therefore, someone accidentally deleted the record for this

employee between 14:00 and 17:00. The HR administrator uses Flashback Table to return

the table to its state at 14:00, respecting any triggers set on the hr.temp\_employees table, by

using the SQL statement in the following example:

FLASHBACK TABLE hr.temp\_employees

TO TIMESTAMP TO\_TIMESTAMP('2013-03-03 14:00:00' , 'YYYY-MM-DD HH:MI:SS')

ENABLE TRIGGERS;

Flashback Drop

Flashback Drop reverses the effects of a DROP TABLE operation. Flashback Drop is faster than

other recovery mechanisms that can be used in this situation, such as point-in-time recovery,

and does not lead to downtime or loss of recent transactions

When you drop a table, the database does not immediately remove the space

associated with the table. Instead, the table is renamed and, along with any associated

objects, placed in the recycle bin

**To retrieve a dropped table:**

**1.** Ensure that the prerequisites described in "Prerequisites of Flashback Drop" are met.

**2.** Connect SQL\*Plus to the target database and obtain the name of the dropped table in

the recycle bin.

You can use the SQL\*Plus command SHOW RECYCLEBIN as follows:

SHOW RECYCLEBIN;

ORIGINAL NAME RECYCLEBIN NAME TYPE DROP TIME

---------------- --------------------------------- ------------ -------------

EMPLOYEE\_DEMO BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0 TABLE 2013-04-11:17:08:54

The ORIGINAL NAME column shows the original name of the object, whereas the

RECYCLEBIN NAME column shows the name of the object as it exists in the bin.

Alternatively, you can query USER\_RECYCLEBIN or DBA\_RECYCLEBIN to obtain the table

name. The following example queries the RECYCLEBIN view to determine the original

names of dropped objects:

SELECT object\_name AS recycle\_name, original\_name, type

FROM recyclebin;

RECYCLE\_NAME ORIGINAL\_NAME TYPE

-------------------------------- --------------------- ----------

BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0 EMPLOYEE\_DEMO TABLE

BIN$JKS983293M1dsab4gsz/I249==$0 I\_EMP\_DEMO INDEX

If you plan to manually restore original names for dependent objects, then ensure that

you make note of each dependent object's system-generated recycle bin name before

you restore the table.

Chapter 18

Rewinding a DROP TABLE Operation with Flashback Drop

18-13

**Note:**

Object views such as DBA\_TABLES do not display the recycle bin objects.

**3.** Optionally, query the table in the recycle bin.

You must use the recycle bin name of the object in your query rather than the

object's original name. The following example queries the table with the recycle bin

name of BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0:

SELECT \*

FROM "BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0";

Quotation marks are required because of the special characters in the recycle bin

name.

**Note:**

If you have the necessary privileges, then you can also use Flashback

Query on tables in the recycle bin, but only by using the recycle bin

name rather than the original table name. You cannot use Data

Manipulation Language (DML) or DDL statements on objects in the

recycle bin.

**4.** Retrieve the dropped table.

Use the FLASHBACK TABLE ... TO BEFORE DROP statement. The following example

restores the BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0 table, changes its name back to

hr.employee\_demo, and purges its entry from the recycle bin:

FLASHBACK TABLE "BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0" TO BEFORE DROP;

The table name is enclosed in quotation marks because of the possibility of

special characters appearing in the recycle bin object names.

Alternatively, you can use the original name of the table:

FLASHBACK TABLE HR.EMPLOYEE\_DEMO TO BEFORE DROP;

You can also assign a new name to the restored table by specifying the RENAME TO

clause. For example:

FLASHBACK TABLE "BIN$gk3lsj/3akk5hg3j2lkl5j3d==$0" TO BEFORE DROP

RENAME TO hr.emp\_demo;

**5.** Optionally, verify that all dependent objects retained their system-generated

recycle bin names.

The following query determines the names of the indexes of the retrieved

hr.employee\_demo table:

SELECT INDEX\_NAME

FROM USER\_INDEXES

WHERE TABLE\_NAME = 'EMPLOYEE\_DEMO';

INDEX\_NAME

------------------------------

BIN$JKS983293M1dsab4gsz/I249==$0

**6.** Optionally, rename the retrieved indexes to their original names.

The following statement renames the index to its original name of i\_emp\_demo:

ALTER INDEX "BIN$JKS983293M1dsab4gsz/I249==$0" RENAME TO I\_EMP\_DEMO;

**7.** If the retrieved table had referential constraints before it was placed in the recycle bin,

then re-create them.

This step must be performed manually because the recycle bin does not preserve

referential constraints on a table.

------------------------------------------------------------------------------

Rewinding a Database with Flashback Database

configured for flashback logging. To return the database to a guaranteed restore point, you

must have previously defined a guaranteed restore point.

prerequisites:

• No current data files are lost or damaged. You can only use FLASHBACK DATABASE to

rewind changes to a data file made by an Oracle database, not to repair media failures.

• You are not trying to recover from accidental deletion of data files, undo a shrink data file

operation, or undo a change to the database name.

• You are not trying to use FLASHBACK DATABASE to return to a point in time before the

restore or re-creation of a control file. If the database control file is restored from backup

or re-created, then all accumulated flashback log information is discarded.

• You are not trying to use FLASHBACK DATABASE to undo a compatibility change

**To perform a Flashback Database operation:**

**1.** Ensure that the prerequisites described in "Prerequisites of Flashback Database" are

met.

**2.** Connect SQL\*Plus to the target database and determine the desired SCN, restore point,

or point in time for the FLASHBACK DATABASE command.

Obtain the earliest SCN in the flashback database window as follows:

SELECT OLDEST\_FLASHBACK\_SCN, OLDEST\_FLASHBACK\_TIME

FROM V$FLASHBACK\_DATABASE\_LOG;

The most recent SCN that can be reached with Flashback Database is the current SCN

of the database. The following query returns the current SCN:

SELECT CURRENT\_SCN FROM V$DATABASE;

You can query available guaranteed restore points as follows (sample output

included):

SELECT NAME, SCN, TIME, DATABASE\_INCARNATION#,

GUARANTEE\_FLASHBACK\_DATABASE

FROM V$RESTORE\_POINT

WHERE GUARANTEE\_FLASHBACK\_DATABASE='YES';

NAME SCN TIME DATABASE\_INCARNATION# GUA

--------------- ---------- --------------------- --------------------- ---

BEFORE\_CHANGES 5753126 04-MAR-12 12.39.45 AM 2 YES

**Note:**

If the flashback window does not extend far enough back into the past to

reach the desired target time, and if you do not have a guaranteed

restore point at the desired time

**3.** Shut down the database consistently, ensure that it is not opened by any instance,

and then mount it:

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

**4.** Repeat the query in Step 2 of this procedure.

Some flashback logging data is generated when the database is shut down. If

flashback logs were deleted due to space pressure in the fast recovery area, then

your target SCN may not be reachable.

**Note:**

If you run FLASHBACK DATABASE when your target SCN is outside the

flashback window, then FLASHBACK DATABASE fails with an ORA-38729

error. In this case your database does not change.

**5.** Start RMAN and connect to the target database,

**6.** Run the SHOW command to see which channels are preconfigured.

During the flashback operation, RMAN may need to restore archived redo logs

from backup. Enter the following command to see whether channels are

configured (sample output is included):

SHOW ALL;

RMAN configuration parameters for database with db\_unique\_name PROD1 are:

...

CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default

CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE DEVICE TYPE SBT\_TAPE PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE CHANNEL DEVICE TYPE 'SBT\_TAPE' PARMS "SBT\_LIBRARY=/usr/local/oracle/

backup/lib/libobk.so";

If the necessary devices and channels are configured, then no action is necessary.

Otherwise, use the CONFIGURE command to configure automatic channels, or include

ALLOCATE CHANNEL commands within a RUN block.

**7.** Run the RMAN FLASHBACK DATABASE command.

You can specify the target time by using a form of the command shown in the following

examples:

FLASHBACK DATABASE TO SCN 46963;

FLASHBACK DATABASE

TO RESTORE POINT BEFORE\_CHANGES;

FLASHBACK DATABASE TO TIME

"TO\_DATE('09/20/12','MM/DD/YY')";

When the FLASHBACK DATABASE command completes, the database is left mounted and

recovered to the specified target time.

**8.** Open the database read-only in SQL\*Plus and run some queries to verify the database

contents.

Open the database read-only as follows:

ALTER DATABASE OPEN READ ONLY;

If you are satisfied with the state of the database, then end the procedure with Step 9. If

you are *not* satisfied with the state of the database, skip to Step 10.

**9.** If you are satisfied with the results, then perform either of the following mutually exclusive

actions:

• Make the database available for updates by opening the database with the

RESETLOGS option. If the database is currently open read-only, then execute the

following commands in SQL\*Plus:

SHUTDOWN IMMEDIATE

STARTUP MOUNT

ALTER DATABASE OPEN RESETLOGS;

.

**10.** If you find that you used the wrong restore point, time, or SCN for the flashback,

then mount the database and perform one of the following mutually exclusive

options:

• If your chosen target time was not far enough in the past, then use another

FLASHBACK DATABASE command to rewind the database further back in time:

FLASHBACK DATABASE TO SCN 42963; #earlier than current SCN

• If you chose a target SCN that is too far in the past, then use RECOVER

DATABASE UNTIL to wind the database forward in time to the desired SCN:

RECOVER DATABASE UNTIL SCN 56963; #later than current SCN

• If you want to completely undo the effect of the FLASHBACK DATABASE

command, then you can perform complete recovery of the database by using

the RECOVER DATABASE command without an UNTIL clause or SET UNTIL

command:

RECOVER DATABASE;

The RECOVER DATABASE command reapplies all changes to the database,

returning it to the most recent SCN.

------------------------------------------------------------------------------------------

Performing a Flashback Database Operation for a Whole CDB

You can perform a flashback database operation for a whole multitenant container

database (CDB) using the FLASHBACK DATABASE command.

**To perform a flashback database operation for a whole CDB:**

**1.** Ensure that the prerequisites described in "Prerequisites of Flashback Database" are

met.

**2.** Connect SQL\*Plus to the target CDB and determine the desired SCN, restore point, or

point in time for the FLASHBACK DATABASE command.

Obtain the earliest SCN in the flashback database window as follows:

SELECT OLDEST\_FLASHBACK\_SCN, OLDEST\_FLASHBACK\_TIME

FROM V$FLASHBACK\_DATABASE\_LOG;

The most recent SCN that can be reached with Flashback Database is the current SCN

of the database. The following query returns the current SCN:

SELECT CURRENT\_SCN

FROM V$DATABASE;

You can query available guaranteed restore points as follows (sample output included):

SELECT NAME, SCN, TIME, DATABASE\_INCARNATION#,

GUARANTEE\_FLASHBACK\_DATABASE

FROM V$RESTORE\_POINT

WHERE GUARANTEE\_FLASHBACK\_DATABASE='YES';

NAME SCN TIME DATABASE\_INCARNATION# GUA

--------------- ---------- --------------------- --------------------- ---

BEFORE\_CHANGES 5753126 04-MAR-12 12.39.45 AM 2 YES

**Note:**

If the flashback window does not extend far enough back into the past to reach

the desired target time, and if you do not have a guaranteed restore point at the

desired time, then you can achieve similar results by using database point-intime

recovery, as described in "Performing Point-in-Time Recovery of a Whole

CDB".

**3.** Shut down the database consistently, ensure that it is not opened by any instance, and

then mount it:

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

**4.** Repeat the query in Step 2 of this procedure.

Some flashback logging data is generated when the database is shut down. If flashback

logs were deleted due to space pressure in the fast recovery area, then your target SCN

may not be reachable.

**Note:**

If you run FLASHBACK DATABASE when your target SCN is outside the

flashback window, then FLASHBACK DATABASE fails with an ORA-38729

error. In this case, your database does not change.

**5.** Connect to the root as a common user with the SYSDBA or SYSBACKUP privilege,

**6.** To see which channels are preconfigured, run the SHOW command .

During the flashback operation, RMAN may need to restore archived redo logs

from backup. To see whether channels are configured, enter the following

command (sample output is included):

SHOW ALL;

If the necessary devices and channels are configured, then no action is necessary.

Otherwise, use the CONFIGURE command to configure automatic channels, or

include ALLOCATE CHANNEL commands within a RUN block.

**7.** Run the FLASHBACK DATABASE command to perform a flashback operation for the

whole CDB to a specified point in time.

You can specify the target time by using an SCN, a time expression, or a CDB

restore point.

The following examples perform a flashback database operation for the whole

CDB:

FLASHBACK DATABASE TO SCN 345588;

FLASHBACK DATABASE TO RESTORE POINT cdb\_before\_upgrade;

**8.** Open the CDB read-only in SQL\*Plus and run some queries to verify the database

contents.

Open the CDB read-only as follows:

ALTER DATABASE OPEN READ ONLY;

If you are satisfied with the state of the database, then end the procedure with

Step 9. If you are *not* satisfied with the state of the database, skip to Step 10.

**9.** If you are satisfied with the results, then perform either of the following mutually

exclusive actions:

• Make the database available for updates by opening the database with the

RESETLOGS option. If the database is open read-only, then execute the

following commands in SQL\*Plus:

SHUTDOWN IMMEDIATE

STARTUP MOUNT

ALTER DATABASE OPEN RESETLOGS;

**Note:**

After you perform this OPEN RESETLOGS operation, all changes to the

database after the target SCN for FLASHBACK DATABASE are abandoned.

**10.** If you find that you used the wrong restore point, time, or SCN for the flashback, then

mount the database and perform one of the following mutually exclusive options:

• If your chosen target time was not far enough in the past, then use another

FLASHBACK DATABASE command to rewind the database further back in time:

FLASHBACK DATABASE TO SCN 42963; #earlier than current SCN

• If you chose a target SCN that is too far in the past, then use RECOVER DATABASE

UNTIL to wind the database forward in time to the desired SCN:

RECOVER DATABASE UNTIL SCN 56963; #later than current SCN

• If you want to completely undo the effect of the FLASHBACK DATABASE command, then

you can perform complete recovery of the database by using the RECOVER DATABASE

command without an UNTIL clause or SET UNTIL command:

RECOVER DATABASE;

The RECOVER DATABASE command reapplies all changes to the database, returning it

to the most recent SCN.

**11.** Since the PDBs are not automatically opened when the CDB is opened, open the PDBs.

The following command, when connected to the root, opens all the PDBs:

ALTER PLUGGABLE DATABASE ALL OPEN;

If you want to open only some PDBs, then you can open each PDB separately. The

following command, when connected to the root, opens the PDB my\_pdb.

ALTER PLUGGABLE DATABASE my\_pdb OPEN;

--------------------------------------------------------------------------------------------

Performing a Flashback Database Operation for PDBs

**To perform a Flashback Database operation for a PDB:**

**1.** Connect to the root as a common user with the SYSDBA or SYSBACKUP privilege.

**2.** Ensure that the CDB is open.

The following command, when connected to the root, displays the mode in which

the CDB is open.

SELECT open\_mode from V$DATABASE;

**3.** Determine the desired SCN, restore point, or point in time for the Flashback

Database command.

Query the V$RESTORE\_POINT view to obtain the list of PDB restore points.

V$FLASHBACK\_DATABASE\_LOG displays the oldest SCN to which a flashback

operation can be performed.

**4.** Ensure that the PDB for which the Flashback Database operation must be

performed is closed. Other PDBs can be open and operational.

When connected to the root, the following ALTER PLUGGABLE DATABASE command

closes the PDB my\_pdb.

ALTER PLUGGABLE DATABASE my\_pdb CLOSE;

**5.** Perform a Flashback Database operation for the specified PDB to the desired

point in time.

The following are some examples of flashback database operations for PDBs.

• For a PDB that uses local undo:

FLASHBACK PLUGGABLE DATABASE my\_pdb TO SCN 24368;

FLASHBACK PLUGGABLE DATABASE my\_pdb TO RESTORE POINT guar\_rp;

FLASHBACK PLUGGABLE DATABASE my\_pdb TO CLEAN RESTORE POINT clean\_rp;

• For a PDB that uses shared undo, you can optionally include the AUXILIARY

DESTINATION clause to specify a location for the auxiliary instance that stores data

files restored as part of the Flashback Database operation. If you omit this clause,

then the auxiliary instance is created in the fast recovery area.

FLASHBACK PLUGGABLE DATABASE my\_pdb TO SCN 24368 AUXILIARY

DESTINATION '+data';

FLASHBACK PLUGGABLE DATABASE my\_pdb TO RESTORE POINT

before\_appl\_changes AUXILIARY DESTINATION '/temp/aux\_dest';

FLASHBACK PLUGGABLE DATABASE my\_pdb TO TIME

"TO\_DATE('03/20/15','MM/DD/YY')";

**6.** Open the PDB with RESETLOGS.

The following command opens the PDB named my\_pdb with RESETLOGS:

ALTER PLUGGABLE DATABASE my\_pdb OPEN RESETLOGS;

-------------------------------------------------------------------------------

Monitoring Flashback Database

The progress of Flashback Database during the restore phase can be monitored by

querying the V$SESSION\_LONGOPS view.

SQL> SELECT sofar, totalwork, units FROM v$session\_longops WHERE opname =

'Flashback Database';

SOFAR TOTALWORK UNITS

----- ---------- --------------------------------

17 60 Megabytes

----------------------------------------------------------------------------------------

Performing Database Point-in-Time Recovery

Performing Point-in-Time Recovery of a Whole CDB

Use the RESTORE and RECOVER commands to perform point-in-time recovery for a whole

CDB.

**To perform point-in-time recovery of a whole CDB:**

**1.** Ensure that the prerequisites described in "Prerequisites of Database Point-in-

Time Recovery" are met.

**2.** Determine the time, SCN, restore point, or log sequence that ends recovery.

You can use the Flashback Query features to help you identify when the logical

corruption occurred. If you have a flashback data archive enabled for a table, then

you can query data that existed far in the past.

You can also use the alert log to try to determine the time of the event from which

you must recover.

Alternatively, you can use a SQL query to determine the log sequence number that

contains the target SCN and then recover through this log. For example, run the

following query to list the logs in the current database incarnation (sample output

included):

SELECT RECID, STAMP, THREAD#, SEQUENCE#, FIRST\_CHANGE#

FIRST\_TIME, NEXT\_CHANGE#

FROM V$ARCHIVED\_LOG

WHERE RESETLOGS\_CHANGE# =

( SELECT RESETLOGS\_CHANGE#

FROM V$DATABASE\_INCARNATION

WHERE STATUS = 'CURRENT');

RECID STAMP THREAD# SEQUENCE# FIRST\_CHAN FIRST\_TIM NEXT\_CHANG

---------- ---------- ---------- ---------- ---------- --------- ----------

1 344890611 1 1 20037 24-SEP-13 20043

2 344890615 1 2 20043 24-SEP-13 20045

3 344890618 1 3 20045 24-SEP-13 20046

For example, if you discover that a user accidentally dropped a tablespace at 9:02

a.m., then you can recover to 9 a.m., just before the drop occurred. You lose all

changes to the database made after this time.

**3.** If you are using a target time expression instead of a target SCN, then ensure that

the time format environment variables are appropriate before invoking RMAN.

The following are sample Globalization Support settings:

NLS\_LANG = american\_america.us7ascii

NLS\_DATE\_FORMAT="Mon DD YYYY HH24:MI:SS"

**4.** Connect RMAN to the root as a common user with the SYSBACKUP or SYSDBA privilege

catalog.

**5.** Bring the CDB to a mounted state.

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

**6.** Perform the following operations within a RUN block:

**a.** For DBPITR, use SET UNTIL to specify the target time, SCN, or log sequence number,

or use SET TO to specify a restore point. If specifying a time, then use the date format

specified in the NLS\_LANG and NLS\_DATE\_FORMAT environment variables.

**b.** Restore and recover the CDB.

The following example performs DBPITR on the target CDB until SCN 1000:

RUN

{

SET UNTIL SCN 1000;

RESTORE DATABASE;

RECOVER DATABASE;

}

As shown in the following examples, you can also use time expressions, restore points,

or log sequence numbers to specify the SET UNTIL time:

SET UNTIL TIME 'Nov 15 2013 09:00:00';

SET UNTIL SEQUENCE 9923;

SET TO RESTORE POINT before\_update;

If the operation completes without errors, then DBPITR has succeeded.

**7.** Perform either of the following mutually exclusive actions:

• Open your CDB for read/write, abandoning all changes after the target SCN. In this

case, you must shut down the CDB, mount it, and then execute the following

command:

ALTER DATABASE OPEN RESETLOGS;

The OPEN RESETLOGS operation fails if a data file is offline unless the data file went

offline normally or is read-only. You can bring files in read-only or offline normal

tablespaces online after the RESETLOGS because they do not need any redo.

**8.** Open all the PDBs.

PDBs are not opened when the CDB is opened. The following command, when

connected to the root, opens all the PDBs.

ALTER PLUGGABLE DATABASE ALL OPEN;

You can open each PDB separately.

**PITR for a Container Database**

Prerequisites of Database Point-in-Time Recovery

Certain prerequisites must be met to perform database point-in-time recovery (DBPITR).

This includes the following:

Your database must be running in ARCHIVELOG mode.

You must have backups of all data files from before the target SCN for DBPITR and archived logs for the period between the SCN of the backups and the target SCN.

If the backups were created using transparent encryption, and if a password-protected software keystore was used, then the keystore password must be provided before the restore operation is performed. Use the SET command with the DECRYPTION WALLET OPEN IDENTIFIED BY option to specify the password that must be used to open the password-based keystore.

If a user with the SYSBACKUP privilege is performing the recovery, and a password-protected software keystore is used, grant the SYSKM privilege to this user.

If the backups were created using password-mode encryption, then you must provide the password used to decrypt backups before you run the RESTORE and RECOVER commands. Use the SET DECRYPTION IDENTIFIED BY command to specify the password used to decrypt the backups.

------------------------------------------------------------------------

Steps for PITR for a CDB

1. Bring the CDB to a mounted state.

SHUTDOWN IMMEDIATE;

STARTUP MOUNT;

or

srvctl stop database -d {CDB}

srvctl start database -d {CDB} -o mount

2. Connect to the CDB sid

3. Connect to RMAN

<RMAN target /

RMAN> run

{​​​​​​​​

SET UNTIL TIME "to\_date('03-NOV-2022 14:30:00','DD-MON-YYYY HH24:MI:SS')";

RESTORE DATABASE;

RECOVER DATABASE;

}​​​​​​​​

4. ALTER DATABASE OPEN;

If it fails to open it may need to be open using resetlogs option

ALTER DATABASE OPEN RESETLOGS;

5. Open all the PDBs.

PDBS should open when then CDB opens. Verify that they are open in the READ/WRITE mode and if not then open the PDBS with the following command

ALTER PLUGGABLE DATABASE ALL OPEN;

Performing Point-in-Time Recovery of PDBs

When you recover one or more PDBs to a specified point-in-time, the remaining PDBs

in the CDB are not affected and they can be open and operational.

**To perform DBPITR on a PDB:**

**1.** Ensure that the prerequisites described in "Prerequisites of Database Point-in-

Time Recovery" are met.

**2.** Determine the time, SCN, restore point, or log sequence that ends recovery.

Use Flashback Query or the alert log to determine when the logicla corruption

occurred. Or, use a SQL query to determine the log sequence number that

contains the target SCN and then recover through this log.

**3.** If you are using a target time expression instead of a target SCN, then ensure that

the time format environment variables are appropriate before invoking RMAN.

The following are sample Globalization Support settings:

NLS\_LANG = american\_america.us7ascii

NLS\_DATE\_FORMAT="Mon DD YYYY HH24:MI:SS"

**4.** Connect RMAN to the root as a common user with the SYSDBA or SYSBACKUP

privilege

**5.** Close the PDB that is being recovered. The other PDBs and the CDB can remain

open.

ALTER PLUGGABLE DATABASE pdb1 CLOSE;

**6.** Perform the following operations within a RUN block:

**a.** For DBPITR, use SET UNTIL to specify the target time, SCN, or log sequence

number, or use SET TO to specify a restore point. If specifying a time, then use

the date format specified in the NLS\_LANG and NLS\_DATE\_FORMAT environment

variables.

**b.** Restore and recover the CDB.

The following example performs DBPITR on the PDB my\_pdb until SCN 1000:

RUN

{

SET UNTIL SCN 1000;

RESTORE PLUGGABLE DATABASE my\_pdb;

RECOVER PLUGGABLE DATABASE my\_pdb;

}

**7.** Open the PDB abandoning all changes after the target SCN.

The following example opens the PDB named my\_pdb.

ALTER PLUGGABLE DATABASE my\_pdb OPEN RESETLOGS;

**Example 18-4 Recovering a PDB to a Specified Point-in-time**

This example recovers a PDB named PDB5 up to the SCN 1066 and then opens it for read/

write access. Connect to the root as a common user with the SYSDBA or SYSBACKUP privilege

and enter the following commands:

ALTER PLUGGABLE DATABASE pdb5 CLOSE;

run

{

SET UNTIL SCN 1066;

RESTORE PLUGGABLE DATABASE pdb5;

RECOVER PLUGGABLE DATABASE pdb5;

}

ALTER PLUGGABLE DATABASE pdb5 OPEN RESETLOGS;

This example assumes that a fast recovery area is being used. If you do not use a fast

recovery area, then you must specify the temporary location of the auxiliary set files by using

the AUXILIARY DESTINATION clause.

Opening the PDB with RESETLOGS creates a new PDB incarnation. You can query the

V$PDB\_INCARNATION view for the incarnation number.

**PITR Pluggable Database**

Point-In-Time Recovery is also possible in a multitenant environment. As in Non-CDB, a recovery catalog can be used or not. In this blog we will see how to recover

a dropped tablespace in a PDB. We will also see the importance of using a recovery catalog or not.

A PITR of a PDB does not affect remaining PBDs. That means that while doing a PITR in PDB, people can use the other PDBs. In this blog we are using an oracle 19c database

with local undo mode enabled

SQL>

1 SELECT property\_name, property\_value

2 FROM database\_properties

3\* WHERE property\_name = 'LOCAL\_UNDO\_ENABLED'

PROPERTY\_NAME PROPE

-------------------- -----

LOCAL\_UNDO\_ENABLED TRUE

SQL>

SELECT con\_id, tablespace\_name FROM cdb\_tablespaces WHERE tablespace\_name LIKE 'UNDO%';

CON\_ID TABLESPACE\_NAME

---------- ------------------------------

3 UNDOTBS1

4 UNDOTBS1

1 UNDOTBS1

SQL>

We suppose that

-We have a tablespace named MYTABPDB2

-We have a valid backup of the whole database

-A recovery catalog is not used

1- Connect to the root container

[oracle@oraadserver ~]$ rman target /

Recovery Manager: Release 19.0.0.0.0 - Production on Fri Sep 20 13:07:07 2019

Version 19.3.0.0.0

Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.

connected to target database: ORCL (DBID=1546409981)

RMAN>

2- Close the PDB

RMAN> ALTER PLUGGABLE DATABASE PDB2 close;

using target database control file instead of recovery catalog

Statement processed

RMAN>

3- Do the PITR

RMAN> run

{

SET TO RESTORE POINT myrestpoint;

RESTORE PLUGGABLE DATABASE pdb2;

RECOVER PLUGGABLE DATABASE pdb2;

}

executing command: SET until clause

RMAN>

4- Open the PDB on resetlogs mode

5. RMAN> alter pluggable DATABASE pdb2 open resetlogs;

Statement processed

[oracle@oraadserver trace]$ rman catalog rman/rman@rmancat

Recovery Manager: Release 19.0.0.0.0 - Production on Fri Sep 20 15:28:29 2019

Version 19.3.0.0.0

Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.

connected to recovery catalog database

RMAN> connect target /

connected to target database: ORCL (DBID=1546409981)

After closing PDB2 we run following bloc

RMAN> run

{

SET UNTIL TIME "to\_date('20-SEP-2019 15:27:00','DD-MON-YYYY HH24:MI:SS')";

RESTORE PLUGGABLE DATABASE pdb2;

RECOVER PLUGGABLE DATABASE pdb2;

}

Finished recover at 20-SEP-19

RMAN>

We then open PDB2 with resetlogs mode and then verify with sqlplus

RMAN> run

{​​​​​​​​

SET UNTIL TIME "to\_date('20-SEP-2019 15:27:00','DD-MON-YYYY HH24:MI:SS')";

RESTORE PLUGGABLE DATABASE pdb2;

RECOVER PLUGGABLE DATABASE pdb2;

}​​​​​​​​

ALTER PLUGGABLE DATABASE pdb2 OPEN RESETLOGS;

--------------------------------------------------------------------------------------------------------------------------------------------------

**PITR TABLE**

**RESTORING A TABLE LOCATED ON A PLUGGABLE DATABASE**

**Restoring a table to the same location:**

RECOVER TABLE {Schema.Table} OF PLUGGABLE DATABASE {Pluggable Database}

UNTIL TIME "TO\_DATE('07-SEP-2021 05:40', 'DD-MON-YYYY HH24:MI')"

AUXILIARY DESTINATION 'Location you want to place the AUX Database' ;

Example

RECOVER TABLE AUDIT \_RAW.AR \_INTRANSIT\_STOR OF PLUGGABLE DATABASE JFDWC

UNTIL TIME "TO\_DATE('07-SEP-2021 05:40', 'DD-MON-YYYY HH24:MI')"

AUXILIARY DESTINATION '+DATA' ;

Restoring a table to another scheama

RECOVER TABLE {Schema.Table} OF PLUGGABLE DATABASE {Pluggable Database}

UNTIL TIME "TO\_DATE('07-SEP-2021 05:40', 'DD-MON-YYYY HH24:MI')"

AUXILIARY DESTINATION 'Location you want to place the AUX Database'

REMAP TABLE 'Orig Table Schema'.'Orig Table Name':'New Schema'.'New Table Name';

Example:

RECOVER TABLE AUDIT\_RAW.AR \_INTRANSIT\_STOR OF PLUGGABLE DATABASE JFDWC

UNTIL TIME "TO\_DATE('07-SEP-2021 05:40', 'DD-MON-YYYY HH24:MI')"

AUXILIARY DESTINATION '+DATA'

REMAP TABLE 'AUDIT \_RAW'.'AR\_EBS\_INTRANSIT\_STOR:'DB3838'.'AR \_INTRANSIT\_STOR\_FRAW\_PREV';

Before starting the restore the following steps are required:

1. Make sure you have space to create the the Auxilary Database. Should be the same size of the pluggable database

2. Make sure you have space in the tablespace that the table is located. If this is not done you will receive the

following error: RMAN-05122: There is insufficient disk space 46638 MB to perform table recovery of 241387 MB.

Steps:

1. Sign into the Database SID

2. Put the terminal into screen mode

3. Start up rman

4. Start up the command to restore the table:

RECOVER TABLE AUDIT \_RAW.AR\_EBS\_INTRANSIT\_STOR OF PLUGGABLE DATABASE JFDWC

UNTIL TIME "TO\_DATE('07-SEP-2021 05:40', 'DD-MON-YYYY HH24:MI')"

AUXILIARY DESTINATION '+DATA'

REMAP TABLE 'AUDIT \_RAW'.'AR\_EBS\_INTRANSIT\_STOR:'DB3838'.'AR\_EBS\_INTRANSIT\_STOR\_FRAW\_PREV';

Once the restore starts you will receive the following info which you will need to set up the wallet

which will be needed:

Creating automatic instance, with SID='qvcc'

db\_create\_file\_dest = "+DATA"

\_clone\_one\_pdb\_recovery = TRUE

db\_domain = "idm.dla.mil"

\_system\_trig\_enabled = FALSE

db\_name = "AZPR008C"

db\_unique\_name = "qvcc\_pitr\_JFDWC\_AZDBPR008C"

diagnostic\_dest = "/u01/app/oracle"

enable\_pluggable\_database= TRUE

5. Once you receive the sid name which would be qvcc on this restore you will need to create the wallet so the wallet will open

during the restore. If not done you will receive the following error:

RMAN-03002: failure of recover command at 09/10/2021 03:19:20

RMAN-03015: error occurred in stored script Memory Script

ORA-19870: error while restoring backup piece /oraback/AZPR008CDB/B12BC35A458DB22CE0535EDA4E837600/backupset/2021\_09\_08/o1\_mf\_nnnnf\_\_jmj4mhrz\_.bkp

ORA-19913: unable to decrypt backup

ORA-28365: wallet is not open

6. Create the Wallet for the new SID:

Open up new session

cd /u01/app/oracle/admin/wallet

mkdir -p <aux sid> for ex: mkdir -p qvcc

copy the wallet information into the newly created wallet:

cd /u01/app/oracle/admin/wallet/azpr008c (If the pdb that needs the table to be recovered make sure to copy the cdb sid's wallet info)

cp cwall\* /u01/app/oracle/admin/wallet/qvcc

cp ewall\* /u01/app/oracle/admin/wallet/qvcc

Once all the steps are taken the table wil first restore the table and then recover before it completes. Once recovery is compelete and all is completed

verify that the table is in the schema restored to and data is in the table

This process takes hours to complete and will use up a lot of resources so if possible it is best to start at night

and also comment out the rman backup on the backup server so it doesnt drain out the resources and cause the server to lock

because of lack of resources. A backup can be run once the restore is completed.

**-------------------------------------------------------------------------------------------------------**

Recovering a Dropped PDB

Use the RECOVER PLUGGABLE DATABASE command to recover a PDB that was dropped.

To recover a dropped PDB:

**1.** Ensure that the prerequisites described in "Prerequisites of Database Point-in-Time

Recovery" are met.

If the PDB was configured with isolated mode setting, then open the PDB keystore.

**2.** Determine the time, SCN, or log sequence to which the PDB must be recovered.

Use Flashback Query or the alert log to determine when the logical corruption occurred.

Or, use a SQL query to determine the log sequence number that contains the target SCN

and then recover through this log.

**3.** If you are using a target time expression instead of a target SCN, then ensure that the

time format environment variables are appropriate before invoking RMAN.

**4.** Connect RMAN to the root as a common user with the SYSDBA or SYSBACKUP privilege, as

described in "Making RMAN Connections to CDBs and PDBs". If applicable, connect to a

recovery catalog.

**5.** Perform the following operations within a RUN block:

**a.** Use SET UNTIL to specify the target time, SCN, or log sequence number, or

use SET TO to specify a restore point. If specifying a time, then use the date

format specified in the NLS\_LANG and NLS\_DATE\_FORMAT environment variables.

**b.** If automatic channels are not configured, then manually allocate disk and tape

channels as needed.

**c.** Recover the dropped PDB.

The following example recovers the PDB mypdb that was previously dropped.

run

{

SET UNTIL SCN 146898;

RECOVER PLUGGABLE DATABASE mypdb;

}

If a fast recovery area is not used, include the AUXILIARY DESTINATION clause

in the RECOVER command.

**6.** Open the PDB abandoning all changes after the target SCN.

The following example opens the PDB named mypdb:

ALTER PLUGGABLE DATABASE mypdb OPEN RESETLOGS;

------------------------------------------------------------------------------------

Performing Block Media Recovery

Use block media recovery to recover one or more corrupt data blocks within a data file.

Block media recovery provides the following advantages over data file media recovery:

Block media recovery is most useful for physical corruption problems that involve a small,

well-known number of blocks. Block-level data loss usually results from intermittent, random

I/O errors that do not cause widespread data loss, and memory corruptions that are written to

19-1

disk. Block media recovery is not intended for cases where the extent of data loss or

corruption is unknown and the entire data file requires recovery. In such cases, data

file media recovery is the best solution.

Typically, block corruption is reported in the following locations:

• Results of the LIST FAILURE, VALIDATE, or BACKUP ... VALIDATE command

• The V$DATABASE\_BLOCK\_CORRUPTION view

For example, you may discover the following messages in a user trace file:

ORA-01578: ORACLE data block corrupted (file # 7, block # 3)

ORA-01110: data file 7: '/oracle/oradata/trgt/tools01.dbf'

ORA-01578: ORACLE data block corrupted (file # 2, block # 235)

ORA-01110: data file 2: '/oracle/oradata/trgt/undotbs01.dbf

About Identifying Corrupt Blocks

The V$DATABASE\_BLOCK\_CORRUPTION view displays blocks marked corrupt by database

components such as RMAN, ANALYZE, and SQL queries.

The following types of corruption result in the addition of rows to this view:

• Physical corruption (sometimes called media corruption)

The database does not recognize the block: the checksum is invalid, the block contains

all zeros, or the block header is corrupt.

Physical corruption checking is enabled by default. You can turn off checksum checking

by specifying the NOCHECKSUM option of the BACKUP command, but other physical

consistency checks, such as checks of the block headers and footers, cannot be

disabled.

----------------------------------------------------------------------------

Recovering Individual Blocks

**To recover specific data blocks using the RECOVER...BLOCK command:**

**1.** Obtain the data file numbers and block numbers of the corrupted blocks.

The easiest way to locate trace files and the alert log is to connect SQL\*Plus to the

target database and execute the following query:

SELECT NAME, VALUE

FROM V$DIAG\_INFO;

**2.** Start RMAN and connect to the target database, which must be mounted or open.

**3.** Run the SHOW ALL command to confirm that the appropriate channels are

preconfigured.

**4.** Run the RECOVER ... BLOCK command at the RMAN prompt, specifying the file

and block numbers for the corrupted blocks.

The following example recovers two blocks.

RECOVER

DATAFILE 8 BLOCK 13

DATAFILE 2 BLOCK 19;

The following

example indicates that only backups with the tag mondayam are used when

searching for blocks. You could use the FROM BACKUPSET option to restrict the type

of backup that RMAN searches, or the EXCLUDE FLASHBACK LOG option to restrict

RMAN from searching the flashback logs.

RECOVER

DATAFILE 8 BLOCK 13

DATAFILE 2 BLOCK 199

FROM TAG mondayam;

**To generate automated repair options and repair the failure using the Data**

**Recovery Advisor:**

**1.** Start RMAN and connect to the target database, as described in "Making

Database Connections with RMAN".

**2.** List the failures recorded by the Data Recovery Advisor using the following

command:

Chapter 19

Recovering Individual Blocks

19-6

LIST FAILURE;

Database Role: PRIMARY

List of Database Failures

=========================

Failure ID Priority Status Time Detected Summary

---------- -------- --------- ------------- -------

5720 HIGH OPEN 24-APR-14 Datafile 14:

'/home1/oracle/dbs/tbs\_32.f' contains one or more corrupt blocks

**3.** Generate repair options for the failure listed Step 2.

The following command generates repair options and creates a repair script to perform

the automated repair tasks.

ADVISE FAILURE;

Database Role: PRIMARY

List of Database Failures

=========================

Failure ID Priority Status Time Detected Summary

---------- -------- --------- ------------- -------

5720 HIGH OPEN 24-APR-14 Datafile 14:

'/home1/oracle/dbs/tbs\_32.f' contains one or more corrupt blocks

analyzing automatic repair options; this may take some time

using channel ORA\_DISK\_1

analyzing automatic repair options complete

Mandatory Manual Actions

========================

no manual actions available

Optional Manual Actions

=======================

no manual actions available

Automated Repair Options

========================

Option Repair Description

------ ------------------

1 Perform block media recovery of block 20 in file 14

Strategy: The repair includes complete media recovery with no data loss

Repair script: /home1/oracle/log/diag/rdbms/db12/hm/reco\_287949467.hm

**4.** Perform the automated repairs recommended by Data Recovery Advisor.

RMAN uses the repair script generated by the ADVISE FAILURE command to perform the

required repairs.

REPAIR FAILURE;

Strategy: The repair includes complete media recovery with no data loss

Repair script: /home1/oracle/log/diag/rdbms/db12/hm/reco\_287949467.hm

contents of repair script:

# block media recovery recover datafile 14 block 20;

Do you really want to execute the above repair (enter YES or NO)?

yes

executing repair script

Starting recover at 24-APR-14

using channel ORA\_DISK\_1

channel ORA\_DISK\_1: restoring block(s)channel

ORA\_DISK\_1: specifying block(s) to restore from backup setrestoring blocks

of datafile 00014

channel ORA\_DISK\_1: reading from backup piece /backups/DB121/backupset/

2014\_04\_24/o1\_mf\_nnndf\_TAG20140424T213309\_9omsd7vb\_.bkp

channel ORA\_DISK\_1: piece handle=/backups/DB121/backupset/2014\_04\_24/

o1\_mf\_nnndf\_TAG20140424T213309\_9omsd7vb\_.bkp tag=TAG20140424T213309

channel ORA\_DISK\_1: restored block(s) from backup piece 1

channel ORA\_DISK\_1: block restore complete, elapsed time: 00:00:01

starting media recovery

media recovery complete, elapsed time: 00:00:03

Finished recover at 24-APR-14repair failure complete

When the LIST FAILURE command displays more than one failures, you can

perform repair actions only for a particular failure. Use the option number

displayed in the Automated Repair Options section of the ADVISE FAILURE

command output to perform specific repair actions.

The following command performs only the repair actions listed under Option 2 of

the Automated Repair Options section.

REPAIR FAILURE USING ADVISE OPTION 2;

-----------------------------------------------------------------------------------

Recovering All Blocks in

V$DATABASE\_BLOCK\_CORRUPTION

RMAN can automatically recover all blocks listed in the

V$DATABASE\_BLOCK\_CORRUPTION view.

**To recover all blocks logged in V$DATABASE\_BLOCK\_CORRUPTION:**

**1.** Start SQL\*Plus and connect to the target database.

**2.** Query V$DATABASE\_BLOCK\_CORRUPTION to determine whether corrupt blocks exist.

For example, execute the following statement:

SQL> SELECT \* FROM V$DATABASE\_BLOCK\_CORRUPTION;

**3.** Start RMAN and connect to the target database, as descried in "Making Database

Connections with RMAN".

**4.** Recover all blocks marked corrupt in V$DATABASE\_BLOCK\_CORRUPTION.

The following command repairs all physically corrupted blocks recorded in the

view:

RMAN> RECOVER CORRUPTION LIST;

After the blocks are recovered, the database removes them from

V$DATABASE\_BLOCK\_CORRUPTION.

Recovering a NOARCHIVELOG Database with Incremental

Backups

You can perform limited recovery of changes to a database running in NOARCHIVELOG mode by

applying incremental backups. The incremental backups must be consistent, like all backups

of a database run in NOARCHIVELOG mode, so you cannot make backups of the database

when it is open.

**To recover a NOARCHIVELOG database with incremental backups:**

**1.** After connecting to the target database and the recovery catalog, place the database in a

mounted state:

STARTUP FORCE MOUNT

**2.** Restore and recover the database.

For example, you can perform incomplete recovery with the following commands:

RESTORE DATABASE

FROM TAG "consistent\_whole\_backup";

RECOVER DATABASE NOREDO;

20-1

**3.** Open the database with the RESETLOGS option.

For example, enter the following command:

ALTER DATABASE OPEN RESETLOGS;

----------------------------------------------------------------------------

Restoring the Server Parameter File

RMAN can restore a lost server parameter file to its default location or to a location of

your choice. Unlike the loss of the control file, the loss of the server parameter file

does not cause the instance to immediately stop

**To restore the server parameter file from autobackup:**

**1.** Start RMAN and do one of the following:

• If the database instance is started at the time of the loss of the server

parameter file, then connect to the target database.

• If the database instance is not started when the server parameter file is lost,

and if you are not using a recovery catalog, then run the SET DBID command

to set the DBID of the target database.

**2.** Shut down the database instance and restart it without mounting the database.

When the server parameter file is not available, RMAN starts the instance with a

dummy parameter file. For example, enter the following command:

STARTUP FORCE NOMOUNT;

**3.** Execute a RUN command to restore the server parameter file.

Depending on the situation, you may need to execute multiple commands in the

RUN command. Note the following considerations:

• If the autobackups were not produced with the default format (%F), then use

the SET CONTROLFILE AUTOBACKUP FOR DEVICE TYPE command to specify the

format in effect when the autobackup was performed.

• If the most recent autobackup was not created today, then use SET UNTIL to

specify the date from which to start the search.

• If RMAN is not connected to a recovery catalog, then use SET DBID to set the

DBID for the target database.

• To restore the server parameter file to a nondefault location, specify the TO clause or

TO PFILE clause on the RESTORE SPFILE command.

• If you know that RMAN never produces more than *n* autobackups each day, then you

can set the RESTORE SPFILE FROM AUTOBACKUP ... MAXSEQ parameter to *n* to reduce

the search time. MAXSEQ is set to 255 by default, and RESTORE counts backward from

MAXSEQ to find the last backup of the day. To terminate the restore operation if you do

not find the autobackup in the current day (or specified day), set MAXDAYS 1 on the

RESTORE command.

The following example illustrates a RUN command that restores a server parameter file

from an autobackup on tape:

RUN

{

ALLOCATE CHANNEL c1 DEVICE TYPE sbt PARMS ...;

SET UNTIL TIME 'SYSDATE-7';

SET CONTROLFILE AUTOBACKUP FORMAT

FOR DEVICE TYPE sbt TO '/disk1/control\_files/autobackup\_%F';

SET DBID 123456789;

RESTORE SPFILE

TO '/tmp/spfileTEMP.ora'

FROM AUTOBACKUP MAXDAYS 10;

}

**4.** Restart the database instance with the restored file.

If you are restarting RMAN with a server parameter file in a nondefault location, then

create an initialization parameter file with the line SPFILE=*new\_location*, where

*new\_location* is the path name of the restored server parameter file. Then, restart the

instance with the client-side initialization parameter file.

For example, create a file /tmp/init.ora which contains the single line:

SPFILE=/tmp/spfileTEMP.ora

You can use the following RMAN command to restart the instance with the restored

server parameter file:

STARTUP FORCE PFILE=/tmp/init.ora;

Restoring the Server Parameter File from a Control File Autobackup

**To restore the server parameter file from the control file autobackup:**

**1.** Set the DBID for your database.

**2.** Use the RESTORE SPFILE FROM AUTOBACKUP command.

If the autobackup is in a nondefault format, then first use the SET CONTROLFILE

AUTOBACKUP FORMAT command to specify the format.

**Example 20-1 Restoring the Server Parameter File from a Control File**

**Autobackup**

This example sets the DBID and restores the server parameter file from a control file

autobackup in a nondefault location. RMAN uses the autobackup format and DBID to

search for control file autobackups. If a control file autobackup is found, then RMAN

restores the server parameter file from that backup to its default location.

SET DBID 320066378;

RUN

{

SET CONTROLFILE AUTOBACKUP FORMAT

FOR DEVICE TYPE DISK TO '*autobackup\_format*';

RESTORE SPFILE FROM AUTOBACKUP;

}

----------------------------------------------------------------------------------------------------------------------

Performing Recovery with a Backup Control File

When all current control files are lost, you must restore a backup control file.

Whenever RMAN cannot find online redo logs and

you did not specify an UNTIL time, RMAN reports errors similar to the following:

RMAN-00571: ===========================================================

RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============

RMAN-00571: ===========================================================

RMAN-03002: failure of recover command at 08/29/2013 14:23:09

RMAN-06054: media recovery requesting unknown log: thread 1 scn 86945

You can also restore the control file to any location that you choose other than the

CONTROL\_FILES locations, by using the form RESTORE CONTROLFILE TO ' *filename*':

RESTORE CONTROLFILE TO '/tmp/my\_controlfile';

RMAN Recovery With and Without a Recovery Catalog

The process of recovering a control file depends on whether a recovery catalog is

used.

When RMAN is connected to a recovery catalog, the recovery procedure with a

backup control file is identical to recovery with a current control file.

If you are not using a recovery catalog, then you must restore your control file from an

autobackup. To restore the control file from autobackup, the database must be in a

NOMOUNT state. You must first set the DBID for your database, and then use the

RESTORE CONTROLFILE FROM AUTOBACKUP command.

example, RMAN uses the autobackup format and DBID to determine where to

hunt for the control file autobackup. If one is found, RMAN restores the control file to

all control file locations listed in the CONTROL\_FILES initialization parameter.

SET DBID 320066378;

RUN

{

SET CONTROLFILE AUTOBACKUP FORMAT

FOR DEVICE TYPE DISK TO '*autobackup\_format*';

Chapter 20

Performing Recovery with a Backup Control File

20-6

RESTORE CONTROLFILE FROM AUTOBACKUP;

}

Performing Recovery with a Backup Control File and No Recovery

Catalog

Recovering a database with a backup control file and when no recovery catalog is used

requires you to restore the control file from an autobackup.

This section assumes that you have RMAN backups of the control file, but do not use a

recovery catalog

**To recover the database with a control file autobackup in NOCATALOG mode:**

**1.** Start RMAN and connect to a target database

**2.** Start the target database instance without mounting the database. For example:

STARTUP NOMOUNT;

**3.** Set the database identifier for the target database with the SET DBID command.

RMAN displays the DBID whenever you connect to a target database. You can

also obtain it by inspecting saved RMAN log files, querying the catalog, or looking

at the file names of control file autobackup. For example, run:

SET DBID 676549873;

**4.** Write an RMAN command file to restore the autobackup control file and perform

recovery.

**5.** If recovery was successful, then open the database and reset the online logs:

ALTER DATABASE OPEN RESETLOGS;

**Multiplexing Controlfile ( can also use to restore a controlfile if one is lost but another does exist)**

There was only one controlfile: +DG01/database/controlfile/current.260.660142191

1) shutdown database on all nodes;

ensure no instance is mounting the database

2) startup nomount one instance

3) rman target /

4) RMAN> backup current controlfile;

5) set dbid=1234567 (i.e whatever your database dbid is, this is an example)

RMAN> restore controlfile to '+DGFR' from '+DG01/database/controlfile/current.260.660142191';

NOTE: This will restore a controlfile to the respective directory without creating a symlink.

Check the newly created controlfilename in the directory and edit and set control\_file parameter accordingly in step 6.

6) alter system set control\_files='+DG01/database/controlfile/current.260.660142191',

'+DGFR/database/controlfile/current.374.660394089' scope=spfile sid='\*'; check your ASM disk targets... this is only a example

7) shutdown (immediate)

8) startup or use srvctl start database -d <database name> commands

restore controlfile to '+FRA' from '+DATA/AZDBP07CDB/CONTROLFILE/current.994.1050600109

';

alter system set control\_files='+DATA/AZDBP07CDB/CONTROLFILE/current.994.1050600109',

'+FRA/AZDBP07CDB/CONTROLFILE/current.current.922.1069777643' scope=spfile ;

alter system set db\_recovery\_file\_dest\_size=2000G scope=BOTH;

UPDATED:

There was only one controlfile: +DG01/database/controlfile/current.260.660142191

1) Make backup of the current controlfile

rman target /

RMAN> backup current controlfile;

2) Get the DBID for the instance that the controlfile is multiplexing

rman target /

3) shutdown database on all nodes;

ensure no instance is mounting the database

4) startup nomount;

5) rman connect target

6) set dbid=12345 (enter id received earlier steps)

7) restore controlfile to '+FRA' from '+DATA/AZDBP07CDB/CONTROLFILE/current.994.1050600109

';

8) Verify that the newly created controlfile exist by checking the asm directory to also get the newly controlfile name

9) sqlplus / as sysdba

10) alter system set control\_files='+DATA/AZDBP07CDB/CONTROLFILE/current.994.1050600109',

'+FRA/AZDBP07CDB/CONTROLFILE/current.current.922.1069777643' scope=spfile ;

11) shutdown immediate

12) startup or use srvctl start database -d <database name> commands

13) Bounce HAS to verify will restart

14 Sign into SQL to verify the parameter is correct and has both location

**Restore a Loss Controlfile Using Autobackup**

**Step 1:- Check the RMAN configuration and controlfile autobackup feature is ON.**

[oracle@ram trace]$ export ORACLE\_SID=dbwr

[oracle@ram trace]$ rman target /

Recovery Manager: Release 19.0.0.0.0 - Production on Thu May 16 20:48:09 2019

Version 19.2.0.0.0

Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.

connected to target database: DBWR (DBID=1337475478)

RMAN> show all;

using target database control file instead of recovery catalog

RMAN configuration parameters for database with db\_unique\_name DBWR are:

CONFIGURE RETENTION POLICY TO REDUNDANCY 1; # default

CONFIGURE BACKUP OPTIMIZATION OFF; # default

CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default

CONFIGURE CONTROLFILE AUTOBACKUP ON; # default

CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO '%F'; # default

CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE MAXSETSIZE TO UNLIMITED; # default

CONFIGURE ENCRYPTION FOR DATABASE OFF; # default

CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default

CONFIGURE COMPRESSION ALGORITHM 'BASIC' AS OF RELEASE 'DEFAULT' OPTIMIZE FOR LOAD TRUE ; # default

CONFIGURE RMAN OUTPUT TO KEEP FOR 7 DAYS; # default

CONFIGURE ARCHIVELOG DELETION POLICY TO NONE; # default

CONFIGURE SNAPSHOT CONTROLFILE NAME TO '/u01/app/oracle/product/19.0.0/dbhome\_1/dbs/snapcf\_dbwr.f'; # default

**Step 2:- Simulate a failure when the database is running**

[oracle@ram trace]$ export ORACLE\_SID=dbwr

[oracle@ram trace]$ sqlplus / as sysdba

SQL\*Plus: Release 19.0.0.0.0 - Production on Thu May 16 20:49:00 2019

Version 19.2.0.0.0

Copyright (c) 1982, 2018, Oracle. All rights reserved.

Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.2.0.0.0

SQL> select open\_mode,name from v$database;

OPEN\_MODE NAME

-------------------- ---------

READ WRITE DBWR

SQL> select name from v$controlfile;

NAME

--------------------------------------------------------------

/u01/app/oracle/oradata/DBWR/controlfile/o1\_mf\_g91q1y86\_.ctl

/u01/app/oracle/fast\_recovery\_area/DBWR/controlfile/o1\_mf\_g91q1yow\_.ctl

SQL> !

[oracle@ram trace]$ cd

[oracle@ram ~]$ cd /u01/app/oracle/oradata/DBWR/controlfile/

[oracle@ram controlfile]$ rm o1\_mf\_g91q1y86\_.ctl

[oracle@ram controlfile]$ cd /u01/app/oracle/fast\_recovery\_area/DBWR/controlfile/

[oracle@ram controlfile]$ rm o1\_mf\_g91q1yow\_.ctl

[oracle@ram controlfile]$ sqlplus / as sysdba

SQL\*Plus: Release 19.0.0.0.0 - Production on Thu May 16 20:51:16 2019

Version 19.2.0.0.0

Copyright (c) 1982, 2018, Oracle. All rights reserved.

Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.2.0.0.0

SQL> alter tablespace dbwrtbs add datafile '/u01/app/oracle/oradata/DBWR/dbwrtbs03.dbf' size 100m;

alter tablespace dbwrtbs add datafile '/u01/app/oracle/oradata/DBWR/dbwrtbs03.dbf' size 100m

\*

ERROR at line 1:

ORA-00210: cannot open the specified control file

ORA-00202: control file:

'/u01/app/oracle/oradata/DBWR/controlfile/o1\_mf\_g91q1y86\_.ctl'

ORA-27041: unable to open file

Linux-x86\_64 Error: 2: No such file or directory

Additional information: 3

SQL> select status from v$instance;

STATUS

------------

OPEN

SQL> shut immediate

ORA-00210: cannot open the specified control file

ORA-00202: control file: '/u01/app/oracle/oradata/DBWR/controlfile/o1\_mf\_g91q1y86\_.ctl'

ORA-27041: unable to open file

Linux-x86\_64 Error: 2: No such file or directory

Additional information: 3

SQL> shut abort

ORACLE instance shut down.

**Step 3:- Keep the database in NOMOUNT stage and restore the controlfile**

SQL> startup nomount;

ORACLE instance started.

Total System Global Area 1778381832 bytes

Fixed Size 8897544 bytes

Variable Size 536870912 bytes

Database Buffers 1224736768 bytes

Redo Buffers 7876608 bytes

**Step 4:- Since we are not using a RMAN catalog we need to set the DBID**

[oracle@ram controlfile]$ rman target /

Recovery Manager: Release 19.0.0.0.0 - Production on Thu May 16 20:57:02 2019

Version 19.2.0.0.0

Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.

connected to target database: DBWR (not mounted)

RMAN> set dbid=1337475478;

executing command: SET DBID

RMAN> restore controlfile from autobackup;

Starting restore at 16-MAY-19

using target database control file instead of recovery catalog

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: SID=38 device type=DISK

channel ORA\_DISK\_1: looking for AUTOBACKUP on day: 20190516

channel ORA\_DISK\_1: AUTOBACKUP found: c-1337475478-20190516-06

channel ORA\_DISK\_1: restoring control file from AUTOBACKUP c-1337475478-20190516-06

channel ORA\_DISK\_1: control file restore from AUTOBACKUP complete

output file name=/u01/app/oracle/oradata/DBWR/controlfile/o1\_mf\_g91q1y86\_.ctl

output file name=/u01/app/oracle/fast\_recovery\_area/DBWR/controlfile/o1\_mf\_g91q1yow\_.ctl

Finished restore at 16-MAY-19

**Step 5:- Mount and recover the database**

RMAN> alter database mount;

released channel: ORA\_DISK\_1

Statement processed

**Step 6:- Restore a control file from a backup so we need to recover the database**

RMAN> recover database;

Starting recover at 16-MAY-19

using target database control file instead of recovery catalog

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: SID=39 device type=DISK

starting media recovery

archived log for thread 1 with sequence 1 is already on disk as file /u01/app/oracle/oradata/DBWR/onlinelog/o1\_mf\_1\_gftzchjx\_.log

archived log file name=/u01/app/oracle/oradata/DBWR/onlinelog/o1\_mf\_1\_gftzchjx\_.log thread=1 sequence=1

media recovery complete, elapsed time: 00:00:00

Finished recover at 16-MAY-19

**Step 7:- Open the database using resetlogs option**

Use **RESETLOGS** after incomplete recovery (when the entire redo stream wasn’t applied). RESETLOGS will initialize the logs, reset your log sequence number, and start a new “incarnation” of the database.

RMAN> alter database open resetlogs;   
Statement processed

Restore control file without backup

**Step 1 – Start the instance**

[oracle@trichydoyen ~]$ sqlplus / as sysdba

Connected to an idle instance.

SQL> startup

**ORA-00205: error in identifying control file, check alert log for more info**

SQL> shut abort

SQL> startup mount

SQL> show parameter control\_file

SQL> **recover database using BACKUP CONTROLFILE;**

**ORA-00283: recovery session canceled due to errors**

**ORA-01110: data file 1: '/u01/manualdb/files/data/system01.dbf'**

**ORA-01157: cannot identify/lock data file 1 - see DBWR trace file**

**ORA-01110: data file 1: '/u01/manualdb/files/data/system01.dbf'**

SQL> alter database backup controlfile to trace as '/u01/app/oracle/diag/rdbms/Biju/Biju/trace/ctl.sql';

Database altered.

SQL> shutdown immediate;

ORA-01109: database not open

**Step 2 – restore the controlfile and mount the database**

SQL> startup nomount;

SQL> CREATE CONTROLFILE REUSE DATABASE "MANUALDB" NORESETLOGS  ARCHIVELOG

**Step 3 – Restore the database**

SQL>  recover database using backup controlfile;

**Step 4 – Open the database.**

SQL> alter database open resetlogs;

Database altered.

SQL> select name,open\_mode from v$database;

**BACKUP CONTROLFILE RECOVERY**

If you are using a backup control file with an incomplete recovery, then specify the USING BACKUP CONTROLFILE option in the RECOVER command.

SQL> RECOVER DATABASE UNTIL CANCEL USING BACKUP CONTROLFILE;  
SQL> RECOVER DATABASE USING BACKUP CONTROLFILE UNTIL CANCEL;