Practice

Using RMAN in Oracle Multitenant Databases

Practice Target

In this practice, you will use RMAN to perform backup and recovery operations on a CDB database (multitenant environment).

Practice Overview

In this practice, you will perform the following tasks:

- Use RMAN to create backup files of a CDB Database
- Use RMAN to take backups for a PDB
- Enable a local PDB administrator to backup and restore its PDB
- Implement recovery scenarios in CDBs and PDBs
- Recover the root container
- Recover the seed container
- Recover from losing a non-system datafile in a PDB
- Point-in-time recovery (PITR) in a PDB

Assumptions

- This practice assumes that you have `srv1` up and running from the **CDB** snapshot.

A. Using RMAN to Create Backup files of a CDB Database

In this section of the practice, you will use RMAN to produce and manage backup files on the CDB container.

1. Invoke SQL*Plus and login to ORADB as sysdba

```
sqlplus / as sysdba
```

2. Enable ARCHIVELOG mode in the database.

```
ARCHIVE LOG LIST

SHUTDOWN IMMEDIATE
STARTUP MOUNT

ALTER SYSTEM SET LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST' SCOPE=SPFILE;

ALTER DATABASE ARCHIVELOG;

ALTER DATABASE OPEN;
ALTER PLUGGABLE DATABASE ALL OPEN;

ARCHIVE LOG LIST

exit
```

3. Invoke RMAN and connect as target to the local database

```
rman target "'/ AS SYSBACKUP'"
```

4. Take full backup of the entire CDB.

The result of the command below is to take backup of all the database datafiles (the root and all its PDBs), control files, SPFILE file, and the archived redo log files.

```
BACKUP DATABASE TAG 'CDB_FULL' PLUS ARCHIVELOG TAG 'ARC_FULL';
```

5. List the backupsets.

You should see backupsets for the following:

- Datafiles that belong to the root container (CDB$ROOT)
- Datafiles that belong to PDB1
- Datafile that belong to the seed container (PDB$SEED)
- Control file and SPFILE as part of the AUTOBACKUP file
- Archived redo log files

```
LIST BACKUPSET;
```

6. Issue the following commands and observe the difference between them.

   The first command lists all the backupsets. The second command lists the backupset of the root container only.

```
LIST BACKUPSET OF DATABASE;
LIST BACKUPSET OF DATABASE ROOT;
```

   - Can the taken backup be used to restore a PDB?

7. Issue the following command to know the answer.

```
LIST BACKUPSET OF PLUGGABLE DATABASE pdb1;
```

B. Using RMAN to Take Backup for the PDBs

In this section of the practice, you will use RMAN to produce and manage backup files on the PDBs when we are connected to the root container.

8. Take a whole PDB backup of PDB1.

```
BACKUP PLUGGABLE DATABASE pdb1 TAG 'PDB1_FULL';
```

9. Retrieve the backupset of the pluggable database PDB1.

Observe that backupsets tagged as "CDB_FULL" and "PDB1_FULL" can both be used to restore the PDB.

```
LIST BACKUPSET OF PLUGGABLE DATABASE pdb1 SUMMARY;
```

10. Take a backup of users tablespace in PDB1

```
BACKUP TABLESPACE pdb1:users TAG 'PDB1_USERS';
```

11. List the backupsets that can be used to restore the users tablespace in pdb1.

All the backupsets taken so far can be used to restore users tablespace in pdb1.

```
LIST BACKUP OF TABLESPACE pdb1:users SUMMARY;
```

12. Exit from RMAN.

```
exit
```

C. Enabling a Local PDB Administrator to Backup and Restore its PDB

In this section of the practice, you will perform a procedure to enable a local admin user of PDB1 to take backup and restore its PDB.

13. In SQL*Plus, login to the root as sysdba

```
sqlplus / as sysdba
```

14. Create a local user named as pdb1admin for PDB1.

```
ALTER SESSION SET CONTAINER=PDB1;
CREATE USER pdb1admin IDENTIFIED BY ABcd##1234 CONTAINER=current;
```

15. Grant SYSBACKUP to the local admin user of PDB1.

This grant will be applicable only in the current container (PDB1), not the CDB.

```
GRANT SYSBACKUP TO pdb1admin CONTAINER=CURRENT;

EXIT
```

16. Run RMAN and login as target to PDB1 using the local admin user.

Observe that the DBID of PDB1 is different from the DBID value for the root container.

```
rman target "'pdb1admin/ABcd##1234@PDB1 AS SYSBACKUP'"
```

17. Take a whole PDB backup of PDB1.

```
# issue either of the commands below. To the user pdb1admin point of view, they both have
# the same effect:
BACKUP DATABASE TAG 'PDB1_FULL_LOCAL';
BACKUP PLUGGABLE DATABASE pdb1 TAG 'PDB1_FULL';
```

18. List the backupset files.

Notice that this command does not only display the backupset taken by pdb1admin user, it also displays the backupset of PDB1 taken by SYS user.

```
LIST BACKUPSET;
```

19. Issue the following command. Type **NO** when it prompts for deletion.

Notice that among the files that will be deleted by the command the backups that were taken by SYS.

```
DELETE BACKUPSET;
```

20. Display backupsets of the archivelog files and try deleting them.

The user cannot delete backupsets of archive log files because the logged on user has SYSBACKUP privilege on PDB1 only, not the CDB. PDB administrators cannot delete or backup the archived redo log files.

```
LIST BACKUP OF ARCHIVELOG ALL SUMMARY;
DELETE BACKUPSET OF ARCHIVELOG ALL;
```

21. Exit from RMAN.

```
exit
```

D. Implementing Recovery Scenarios in CDB and PDBs

In this section of the practice, you will perform various recovery scenarios in a CDB database and its PDBs. The proposed solution for each scenario is not necessarily the best solution. Different solutions are proposed to gain experience on using them.

Scenario 1: Recovering the root container

In this scenario, the CDB SYSTEM data file is lost. The proposed recovery solution will restore only the lost datafile.

Scenario Simulation

In the following steps you will obtain the full name of the CDB SYSTEM datafile and delete it. CDB system datafile belongs to the root container.

22. Invoke RMAN with connecting to the local database as target

```
rman target /
```

23. Obtain the full name from the system datafile and take a note of it.

```
SELECT NAME FROM V$DATAFILE WHERE FILE# =1;
```

24. Delete the datafile.

```
host "rm -f <datafile name>";
```

25. Validate the database so that the lost datafile is discovered.

```
VALIDATE DATABASE;
```

Recovering from the loss

In the following steps, you will take actions to recover from losing the CDB system data file.

26. Issue the following commands:

You could use the "RESTORE DATABASE ROOT" command, but this command restores all the root container datafiles. Restoring the lost datafile alone is quicker.

Observe that the PDBs need to be started, by default, after starting up the CDB.

```
STARTUP FORCE MOUNT
RESTORE DATAFILE 1;
RECOVER DATAFILE 1;
ALTER DATABASE OPEN;
ALTER PLUGGABLE DATABASE ALL OPEN;
```

Scenario 2: Recovering the seed container

In this scenario, a data file that belongs to the seed container is lost. The proposed recovery solution will restore the entire PDB.

Scenario Simulation

In the following steps you will obtain the full name of a data file in the seed container and delete it.

27. Obtain the full names of the seed datafiles and take a note of one of them.

Observe that the seed container CON_ID always equals to 2.

```
SELECT NAME FROM V$DATAFILE WHERE CON_ID=2;
```

28. Delete at least one of the returned datafiles.

```
host "rm -f <datafile name>";
```

29. Invoke rman with connecting to the local database as target then validate the database.

The lost datafile number is reported.

```
VALIDATE DATABASE;
```

Recovering from the loss

In the following steps, you will take actions to recover from losing a seed datafile.

30. Issue the following shaded commands:

Observe the solution does not need to shut down the CDB. All application sessions remain in their normal operations.

It is more efficient to restore the lost datafile only, as we did in the previous section. We are using a different method just to gain experience on using it.

```
ALTER PLUGGABLE DATABASE "PDB$SEED" CLOSE;
RESTORE PLUGGABLE DATABASE "PDB$SEED";
RECOVER PLUGGABLE DATABASE "PDB$SEED";
ALTER PLUGGABLE DATABASE "PDB$SEED" OPEN READ ONLY;
```

Scenario 3: Recovery from losing a non-system datafile in a PDB

In this scenario, a data file that belongs to a user tablespace in a PDB is lost. The proposed recovery solution will restore the lost datafile tablespace.

Scenario Simulation

In the following steps you will obtain the full name of a data file in PDB1 and delete it.

31. Obtain the full names of the PDB1 datafiles. Take a note of a non-system datafile (it will be the SOETBS tablespace datafile).

```
SELECT NAME FROM V$DATAFILE WHERE CON_ID=(SELECT CON_ID FROM V$PDBS WHERE NAME='PDB1');
```

32. Delete the datafile that belongs to SOETBS tablespace. You can identify it from its name.

```
host "rm -f <datafile name>";
```

33. Validate PDB1 datafiles and observe the reported lost datafile number.

```
VALIDATE PLUGGABLE DATABASE pdb1;
```

Recovery Procedure

In the following steps, you will recover from losing a non-system PDB datafile by restoring its tablespace.

34. Issue the following queries to retrieve the tablespace name of the lost datafile.

Replace <*n*> with the lost data file number obtained from the previous step.

In a CDB database, tablespaces are not uniquely identified by their names only. To identify a tablespace, you need to specify its CON_ID as well.

```
SELECT CON_ID, NAME FROM V$TABLESPACE WHERE TS# = (SELECT TS# FROM V$DATAFILE WHERE
FILE#=<n>);

# from the CON_ID obtained above, you can obtain the PDB name:
# In our case, as PDB1 is the first PDB created in the database, its CON_ID must be 3
SELECT NAME FROM V$PDBS WHERE CON_ID=<con id>;
```

35. Exit from RMAN.

```
exit
```

36. Issue the following recovery commands.

Observe the solution does not need to shut down the CDB. Only the tablespace becomes not available.

```
rman target sys/ABcd##1234@pdb1
ALTER TABLESPACE soetbs OFFLINE IMMEDIATE;
RESTORE TABLESPACE soetbs;
RECOVER TABLESPACE soetbs;
ALTER TABLESPACE soetbs ONLINE;
```

37. Validate PDB1 datafiles. It should return no error.

```
VALIDATE PLUGGABLE DATABASE pdb1;
```

38. Exit from RMAN

```
exit
```

Scenario 4: Performing Point-in-time recovery (PITR) in a PDB

In this scenario, you will rewind PDB1 to specific point-in-time in the past to recover from losing data.

Scenario Simulation

In the following steps you will create a table in PDB1 then drop it.

39. Invoke SQL*Plus and login as SYSDBA to the local instance.

```
sqlplus / as sysdba
```

40. Switch the redo logfile

```
ALTER SYSTEM SWITCH LOGFILE;
```

41. Create a testing table owned by SYSTEM in users tablespace in PDB1.

```
ALTER SESSION SET CONTAINER = PDB1;
CREATE TABLE SYSTEM.MYDATA TABLESPACE users AS SELECT TABLE_NAME FROM DBA_TABLES;
```

42. Take a note of the current date and time

```
SELECT TO_CHAR(SYSDATE,'YYYY-MM-DD:HH24:MI:SS') FROM DUAL;
```

43. Switch the redo logfile

```
ALTER SESSION SET CONTAINER = "CDB$ROOT";
ALTER SYSTEM SWITCH LOGFILE;
```

44. Drop the testing table in PDB1 and exit from SQL*Plus.

```
ALTER SESSION SET CONTAINER = PDB1;
DROP TABLE SYSTEM.MYDATA;
exit
```

Recovery Procedure

In the following steps, you will perform PITR on PDB1 to recover the dropped table.

45. Invoke RMAN and login to the CDB as target.

```
rman target /
```

46. Issue the following commands to rewind PDB1 to the noted time. Replace the <rtime> with the noted recovery time.

```
ALTER PLUGGABLE DATABASE pdb1 CLOSE;
RESTORE PLUGGABLE DATABASE pdb1 UNTIL TIME "TO_DATE('<rtime>','yyyy:mm:dd:hh24:mi:ss')";
RECOVER PLUGGABLE DATABASE pdb1 UNTIL TIME "TO_DATE('<rtime>','yyyy:mm:dd:hh24:mi:ss')"
AUXILIARY DESTINATION '/media/sf_staging/';
ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS;
exit
```

47. Verify the data has been successfully recovered.

```
sqlplus system/ABcd##1234@pdb1
SELECT COUNT(*) FROM SYSTEM.MYDATA;
```

Clean Up

48. Drop the testing table.

```
DROP TABLE SYSTEM.MYDATA PURGE;
```

49. Delete all the backupsets and archive redo log files.

```
rman target /
DELETE NOPROMPT BACKUPSET ;
DELETE NOPROMPT ARCHIVELOG ALL;
```

50. In Oracle VirtualBox, create a new snapshot (named as "**oradb CDB database**") for `srv1` then delete the old snapshot.

Summary

In this practice, you performed the following tasks:

- Use RMAN to create backup files of a CDB Database
- Use RMAN to take backups for a PDB
- Enable a local PDB administrator to backup and restore its PDB
- Implement recovery scenarios in CDBs and PDBs
- Recover the root container
- Recover the seed container
- Recover from losing a non-system datafile in a PDB
- Point-in-time recovery (PITR) in a PDB