Practice 14: Point-in-time Recovery and Flashback

Practice Overview

In this practice you will perform the following:

- Point-in-time Recovery (PITR) in a Pluggable Database
- PITR on a Tablespace in a PDB
- Flashback at the CDB level
- Flashback at the PDB level

Practice Assumptions

- `CDB1` (in `srv1`) database is up and running.

Point-in-time Recovery

A. Performing PITR in a Pluggable Database

In this section of the practice, you will perform point-in-time recovery in PDB2. Firstly, you will perform the PITR in the pluggable database level and secondly in the tablespace level. You will insert some rows in a table, delete them, and recover the PDB to the point before the delete statement was executed.

1. Make sure the local undo mode is enabled.

```
sqlplus / as sysdba

col PROPERTY_NAME format a25
col PROPERTY_VALUE format a10

SELECT PROPERTY_NAME, PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE
PROPERTY_NAME = 'LOCAL_UNDO_ENABLED';
```

2. Verify that the table TB1 which is owned by TUSER is there. If it is not, create it.

```
conn tuser/oracle@pdb2

DESC TUSER.TB1
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ID                                                 NUMBER
NOTES                                              VARCHAR2(20)
```

3. Insert some testing rows into the table TUSER.TB1.

```
BEGIN
 FOR I IN 1..10 LOOP
  INSERT INTO TUSER.TB1 (ID,NOTES) VALUES (I, TO_CHAR(SYSDATE,'DD-MM-YY
HH24:MI'));
 END LOOP;
 COMMIT;
END;
/

SELECT COUNT(*) FROM TUSER.TB1;
```

4. Obtain the current database SCN and take a note of it.

If in real life scenario the recovery point is known by a specific time rather than by a specific SCN number, you can use the following statement to convert the timestamp into SCN number:

```
SELECT TIMESTAMP_TO_SCN(<TIMESTAMP>) DUAL;
```

```
conn / as sysdba
SELECT CURRENT_SCN FROM V$DATABASE;
```

5. Delete the rows in TUSER.TB1 and commit the deletion.

```
conn tuser/oracle@pdb2

DELETE TUSER.TB1;
COMMIT;
```

6. Perform the PDB PITR on PDB2.

a. Run rman and connect as target to the root

```
rman target /
```

b. Close PDB2

```
ALTER PLUGGABLE DATABASE pdb2 CLOSE IMMEDIATE;
```

c. Perform the PDB PITR on PDB2.

Notice that this recovery is at the PDB level. The other containers are not affected.

```
RUN {
 SET UNTIL SCN= <scn>;
 RESTORE PLUGGABLE DATABASE pdb2;
 RECOVER PLUGGABLE DATABASE pdb2;
}

ALTER PLUGGABLE DATABASE pdb2 OPEN RESETLOGS;
```

7. Verify that the data has been restored in the table.

```
sqlplus tuser/oracle@pdb2

SELECT COUNT(*) FROM TUSER.TB1;
```

8. Take backup of the whole CDB. This is recommended after every PITR process.

a. Run rman and login to the root.

```
rman target /
```

b. Delete the existing backupsets (optional; to save disk space in our practice).

```
DELETE BACKUPSET;
```

c. Take backup of the whole CDB.

```
BACKUP DATABASE PLUS ARCHIVELOG DELETE ALL INPUT;
```

B. Performing PITR on a Tablespace in a PDB

In this section of the practice you will perform PITR in the tablespace level. This method provides higher availability than PITR at the pluggable database level because the other tablespaces will still be active while you are recovering the lost tablespace. However, it involves creating temporary (auxiliary) instance. Therefore, it takes longer time to execute and requires much more disk space than the PITR at the PDB level.

9. Verify that the table TB1 which is owned by TUSER is saved in the `users` tablespace.

```
sqlplus tuser/oracle@pdb2

SELECT TABLESPACE_NAME FROM USER_TABLES WHERE TABLE_NAME='TB1';
```

10. As the `users` tablespace is the default tablespace, you need to switch the default tablespace of the PDB to some other tablespace before you proceed with the PITR procedure on it. Create a staging tablespace and set it as the default tablespace.

```
conn / as sysdba
ALTER SESSION SET CONTAINER=pdb2;

-- verify the default tablespace is users tablespace
SELECT PROPERTY_VALUE
FROM DATABASE_PROPERTIES
WHERE PROPERTY_NAME = 'DEFAULT_PERMANENT_TABLESPACE';

--  verify the users tablespace is online:
SELECT STATUS FROM DBA_TABLESPACES WHERE TABLESPACE_NAME='USERS';

-- create a staging tablespace and set it as the default tablespace:
CREATE TABLESPACE stagingtbs;
ALTER DATABASE DEFAULT TABLESPACE stagingtbs;
```

11. Take backup of the whole CDB. This is among the best practices after adding or dropping a datafile in the database.

```
rman target /
DELETE BACKUPSET;
BACKUP DATABASE;
```

12. Delete the existing rows in the table TUSER.TB1 then insert some testing rows into it.

```
sqlplus tuser/oracle@pdb2

DELETE TUSER.TB1;
COMMIT;

BEGIN
 FOR I IN 1..10 LOOP
  INSERT INTO TUSER.TB1 (ID,NOTES)
   VALUES (I, TO_CHAR(sysdate,'DD-MM-YY HH24:MI'));
 END LOOP;
 COMMIT;
END;
/

SELECT COUNT(*) FROM TUSER.TB1;
```

13. Obtain the current database SCN and take a note of it.

```
conn / as sysdba
SELECT CURRENT_SCN FROM V$DATABASE;
```

14. Switch the redo log files so that archived redo log files will be generated.

```
ALTER SYSTEM SWITCH LOGFILE;
```

15. Delete the rows in TUSER.TB1 and commit the deletion.

```
conn tuser/oracle@pdb2

DELETE TUSER.TB1;
COMMIT;
```

16. Make the users tablespace offline.

```
conn / as sysdba

ALTER SESSION SET CONTAINER=PDB2;
ALTER TABLESPACE USERS OFFLINE IMMEDIATE;
```

17. Perform the PITR at the tablespace `users`.

a. Create the directory in which the auxiliary files will be created

```
ls /home/oracle/backup
mkdir /home/oracle/backup
```

b. Run rman and connect as target to the CDB root

```
rman target /
```

c. Perform the PITR of the tablespace `users` in the database PDB2.

To execute this statement online, this statement will make RMAN create an auxiliary instance from which the `users` tablespace will be recovered. Make sure the used directory exists.

```
RECOVER TABLESPACE pdb2:USERS UNTIL SCN <scc> AUXILIARY
DESTINATION='/home/oracle/backup';
```

18. Make the `users` tablespace online.

```
sqlplus / as sysdba
ALTER SESSION SET CONTAINER=PDB2;
ALTER TABLESPACE USERS ONLINE;
```

19. Verify that the data has been restored in the table.

```
conn tuser/oracle@pdb2
SELECT COUNT(*) FROM TUSER.TB1;
```

20. Take backup of the whole CDB. This is recommended after every PITR process.

```
rman target /
DELETE BACKUPSET;
BACKUP DATABASE PLUS ARCHIVELOG DELETE ALL INPUT;
```

21. Make the `users` tablespace back as the default tablespace and drop the staging tablespace.

```
sqlplus / as sysdba

ALTER SESSION SET CONTAINER=pdb2;
ALTER DATABASE DEFAULT TABLESPACE users;
DROP TABLESPACE stagingtbs INCLUDING CONTENTS AND DATAFILES;
```

Using Flashback in CDB and PDBs

A. Performing Flashback for the CDB from Dropping a Common User

In this section of the practice you will enable database flashback in CDB1. Then you will use the flashback to recover from dropping a common user.

Common users are created in the CDB level. To use the flashback to recover from dropping a common user, you must flashback the entire CDB.

22. Create a common user.

```
sqlplus / as sysdba
CREATE USER C##USER1 IDENTIFIED BY oracle CONTAINER=ALL;
GRANT CREATE SESSION TO C##USER1 CONTAINER=ALL;
```

23. Make sure the Fast Recovery Area is enabled.

```
SHOW PARAMETER DB_RECOVERY_FILE_DEST
```

24. Enable the flashback in the CDB. It requires CDB restart.

Actually DB restart below is not needed. Thanks for the student “Amit” who pointed that out.

```
-- verify the flashback is off:
SELECT FLASHBACK_ON FROM V$DATABASE;

-- enable the flashback
SHUTDOWN IMMEDIATE
STARTUP MOUNT
ALTER SYSTEM SET DB_FLASHBACK_RETENTION_TARGET=1440 SCOPE=BOTH;
ALTER DATABASE FLASHBACK ON;
ALTER DATABASE OPEN;
```

25. Obtain the current SCN and take a note of it.

```
SELECT CURRENT_SCN FROM V$DATABASE;
```

26. Drop the common user C##USER1

```
-- verify the user exists:
col username format A20
SELECT USERNAME, COMMON, CON_ID FROM CDB_USERS
WHERE USERNAME='C##USER1';

DROP USER C##USER1 CASCADE;
```

27. Switch the logfile multiple times to generate flashback logs.

```
-- execute the following statement multiple times:
ALTER SYSTEM SWITCH LOGFILE;
```

28. Flashback the CDB database up to the obtained SCN. Observe that this operation takes less time than the PITR.

```
SHUTDOWN IMMEDIATE
STARTUP MOUNT
FLASHBACK DATABASE TO SCN <scn>;
```

29. Open the database in READ ONLY mode and verify that the common user is recovered.

Opening the CDB in READ ONLY mode does not automatically open the PDBs. You have to manually open them.

```
ALTER DATABASE OPEN READ ONLY;
ALTER PLUGGABLE DATABASE ALL OPEN READ ONLY;

-- you should see the common user recovered:
SELECT USERNAME, COMMON, CON_ID
FROM CDB_USERS
WHERE USERNAME='C##USER1';
```

30. Open the CDB with RESETLOGS option.

```
SHUTDOWN IMMEDIATE
STARTUP MOUNT
ALTER DATABASE OPEN RESETLOGS;
```

31. Take backup of the whole CDB. This is recommended after every time you open the database with RESETLOGS option.

```
rman target /
DELETE BACKUPSET;
BACKUP DATABASE PLUS ARCHIVELOG DELETE ALL INPUT;
```

B. Performing Flashback for PDBs

In this section of the practice you will flashback a PDB up to a specific restore point.

Specifically, you will create a restore point, imitate performing an unintended destructive operation in PDB2, and then flashback the PDB to recover from the destructive operation.

32. Create a guaranteed restore point in PDB2.

```
sqlplus / as sysdba

CREATE RESTORE POINT pre_change FOR PLUGGABLE DATABASE pdb2 GUARANTEE FLASHBACK
DATABASE;

col name format a20
SELECT SCN, NAME, CON_ID, PDB_RESTORE_POINT, GUARANTEE_FLASHBACK_DATABASE,
CLEAN_PDB_RESTORE_POINT
FROM V$RESTORE_POINT;
```

33. Delete all the rows in TUSER.TB1 (in PDB2). This is the destructive operation that you need to recover from.

```
CONN TUSER/oracle@pdb2
DELETE tuser.tb1;
COMMIT;
```

34. Flashback PDB2 to the restore point.

```
conn / as sysdba
ALTER PLUGGABLE DATABASE pdb2 CLOSE IMMEDIATE;
FLASHBACK PLUGGABLE DATABASE pdb2 TO RESTORE POINT pre_change;
ALTER PLUGGABLE DATABASE pdb2 OPEN RESETLOGS;
```

35. Verify that the data has been restored in the table.

```
conn tuser/oracle@pdb2
SELECT COUNT(*) FROM TUSER.TB1;
```

36. Drop the restore point.

It is important to drop the guaranteed restore points after you finish from the target of creating them. Otherwise, eventually you will run out of space in the FRA.

```
conn / as sysdba
ALTER SESSION SET CONTAINER = pdb2;
DROP RESTORE POINT pre_change;
```

37. Take backup of the whole CDB.

```
rman target /
DELETE BACKUPSET;
BACKUP DATABASE PLUS ARCHIVELOG DELETE ALL INPUT;
```

Summary

- You can perform PITR for a single PDB, keeping the CDB and the other PDBs up and running meanwhile.
- It easier to execute a PITR for a PDB when local undo is enabled than when the shared undo is being used.
- You can perform PITR on a tablespace within a PDB, keeping the PDB and the other tablespaces available for the users. However, this process involves creating an auxiliary instance. Which means longer recovery time and more disk space is needed.
- Flashback PDB is a more efficient recovery way than PITR. However, it is not practically a full replacement to the PITR. Flashback is best used for recover to a short time period in the past, whereas PITR is best used for recovery to a long time point in the past.