Practice 20

Cross-Platform Data Transport Using Backup sets

Practice Target

In this practice you will perform tablespace and database cross-platform transportation (from Windows to Linux) using backupsets.

Practice Overview

In this practice, you will implement the following cross-platform transport tasks:

- Perform cross-platform tablespace transport from `winsrv2` to `srv1` using backupsets with conversion at the source host.
- Perform cross-platform database transport from `winsrv2` to `srv1` using backupsets with conversion at the destination host.

Assumptions

This practice assumes the following are true:

- `srv1` appliance is up and running and its database `ORADB` is running in `OPEN` state.
- `winsrv2` appliance is up and running and its database `ORAWIN` is running in `OPEN` state.
- The script `create_rc_owner.sql` is downloaded from the downloadable resources section.

A. Practice Preparation Steps

Perform the following steps to prepare the environment for this practice.

1. Take snapshot of `srv1`. Give it the name "**Practice 20 Start**".
2. Take snapshot of `winsrv2`. Give it the name "**Practice 20 Start**".

B. Performing Cross-platform Tablespace Transport with Backupsets

In this section of the practice, you will transfer a tablespace (RC_TBS) from the Windows 64-bit platform running in winsrv2 to the Linux x86 64-bit platform running in srv1. You will perform the datafile conversion in the source host (winsrv2).

Preparing the Backupsets in the Source Host

Following are the steps to be done in winsrv2.

3. In the VirtualBox window of winsrv2, open a command-line prompt window and invoke SQL*Plus with connecting to the local database as sysdba.

```
sqlplus / as sysdba
```

4. In the SQL*Plus session, make sure that the source and destination platforms are supported for cross-platform transportation.

```
SELECT PLATFORM_NAME FROM V$TRANSPORTABLE_PLATFORM
WHERE UPPER(PLATFORM_NAME) LIKE '%LINUX%X86%64-BIT%'
   OR UPPER(PLATFORM_NAME) LIKE '% WINDOWS%X86%64-BIT%';
```

5. Obtain the endian of each platform.

Although in our case the endian of the source platform is the same as the endian of the destination platform, you will perform the conversion in the destination host to gain experience on it.

```
SELECT PLATFORM_NAME, ENDIAN_FORMAT FROM V$TRANSPORTABLE_PLATFORM
WHERE UPPER(PLATFORM_NAME) LIKE '%LINUX%X86%64-BIT%'
   OR UPPER(PLATFORM_NAME) LIKE '% WINDOWS%X86%64-BIT%';
```

6. Make sure the tablespace RC_TBS is self-contained

```
exec DBMS_TTS.TRANSPORT_SET_CHECK('RC_TBS', TRUE,TRUE);


# after executing the procedure above, the following query should return no row:
SELECT * FROM TRANSPORT_SET_VIOLATIONS;
```

7. Place the tablespaces to be transported in read-only mode.

```
ALTER TABLESPACE rc_tbs READ ONLY;
```

8. Exit from SQL*Plus then start RMAN with connecting to the local database as target.

```
rman target /
```

9. Create a cross-platform backup of the tablespace RC_TBS. Save the produced files in the local staging folder.

```
BACKUP TO PLATFORM 'Linux x86 64-bit'
FORMAT 'D:\temp\rc_tbs.bck'
DATAPUMP FORMAT 'D:\temp\rc_tbs.dmp'
TABLESPACE rc_tbs;
```

10. Set the tablespace back to read write mode.

```
ALTER TABLESPACE rc_tbs READ WRITE;
```

11. Move the produced files from the staging folder to the shared folder.

```
host "move D:\temp\*.* F:\backup";
```

12. On the hosting PC, move the produced files from the shared folder accessed by the Windows appliance (winrv2) to the shared folder accessed by the Linux appliance (srv1).

Note: tablespaces cannot be plugged into destination databases, if the owners of their contents are not there in the database. If this is case in a real life scenario, produce script in the source database to re-create the users that own the tablespace objects in the destination database.

Plugging in rc_tbs Tablespace into srv1

Following are the steps to be done in srv1 to plug in the rc_tbs tablespace.

13. Start Putty and connect to srv1 as oracle. Then invoke SQL*Plus with connecting to the local database as target.

```
sqlplus / as sysdba
```

14. Create RC_OWNER user by executing the code in the file create_rc_owner.sql

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

```
rman target /
```

16. Restore the backupsets that were transported from winsrv2

```
RESTORE FOREIGN TABLESPACE rc_tbs TO NEW
FROM BACKUPSET '/media/sf_extdisk/backup/RC_TBS.BCK'
DUMP FILE FROM BACKUPSET '/media/sf_extdisk/backup/RC_TBS.DMP';
```

17. Select a sample data to verify that the transportation was successful.

```
SELECT COUNT(*) FROM DBA_OBJECTS WHERE OWNER = 'RC_OWNER';
```

Clean Up

18. Shutdown both srv1 and winsrv2.

19. Restore srv1 to the snapshot "**Practice 20 Start**". Start srv1.

20. Restore winsrv2 to the snapshot "**Practice 20 Start**". Start winsrv2.

21. Delete the files from the shared folder.

C. Performing Cross-platform Database Transport with Backupsets

In this section of the practice, you will transfer the database ORAWIN from Windows 64-bit platform (winsrv2) and create it as ORADB2 in the Linux x86 64-bit platform (srv1). Converting the datafiles is performed on the source host (winsrv2).

Preparing the Files on the Source Host

Following are the steps to be done in winsrv2.

22. Open a command-prompt window, invoke SQL*Plus and connect as sysdba to the local database.

```
sqlplus / as sysdba
```

23. Re-start ORAWIN database in read-only mode.

```
SHUTDOWN IMMEDIATE
STARTUP MOUNT
ALTER DATABASE OPEN READ ONLY;
```

24. Check whether the database can be transported to a target platform.

```
SET SERVEROUTPUT ON
DECLARE
 v_ready BOOLEAN;
BEGIN
 v_ready := DBMS_TDB.CHECK_DB('Linux x86 64-bit',DBMS_TDB.SKIP_READONLY);
 IF v_ready THEN
  DBMS_OUTPUT.PUT_LINE('Transportable to the platform.');
 ELSE
  DBMS_OUTPUT.PUT_LINE('Not Transportable to the platform.');
 END IF;
END;
/
```

25. Identify external tables, directories, or BFILEs.

In real life scenario, you need to take a decision on copying those objects in the destination database. For your practice environment, we will ignore them.

```
SET SERVEROUTPUT ON
DECLARE
 v_ext BOOLEAN;
BEGIN
 v_ext := DBMS_TDB.CHECK_EXTERNAL;
END;
/
```

26. Start RMAN and connect to the local database as target

```
rman target /
```

27. Create a cross-platform backupset of the entire database, saving the produced file in a staging local folder.

```
BACKUP FOR TRANSPORT FORMAT 'D:\temp\ORAWIN%U' DATABASE;
```

28. Start the ORAWIN in read/write mode

```
SHUTDOWN
STARTUP
```

29. Move the produced backupset file to the shared folder.

```
host "move D:\temp\ORAWIN* F:\backup";
```

30. On the hosting PC, move the produced files from the shared folder accessed by the Windows appliance (winsrv2) to the shared folder accessed by the Linux appliance (srv1).

Creating the Transported Database in srv1

Following are the steps to be done in srv1 to create ORADB2.

31. Exit from any SQL*Plus or RMAN session connected to ORADB.

32. In the VirtualBox window of srv1, login as oracle and open a terminal window.

33. Invoke dbca utility and remove ORADB database.

34. In the Putty session, create the following directories for the new database.

```
mkdir -p /u01/app/oracle/oradata/ORADB2/datafile
mkdir -p /u01/app/oracle/fra/ORADB2
mkdir ~/temp
```

35. Create pfile and save the following lines in it.

```
vi ~/temp/pfileORADB2.ora
```

```
DB_NAME='ORADB2'
CONTROL_FILES='/u01/app/oracle/oradata/ORADB2/datafile/control1.ctl','/u01/app/oracle/fra
/ORADB2/control2.ctl'
DB_CREATE_FILE_DEST='/u01/app/oracle/oradata'
DB_RECOVERY_FILE_DEST='/u01/app/oracle/fra/ORADB2'
DB_RECOVERY_FILE_DEST_SIZE=42949672960
AUDIT_FILE_DEST='/u01/app/oracle/audit'
```

36. Issue the following commands:

```
export ORACLE_SID=ORADB2
rman target /
```

37. Startup the database in NOMOUNT mode

```
STARTUP NOMOUNT pfile='/home/oracle/temp/pfileORADB2.ora'
```

38. Restore the backupsets that were transferred from the source.

```
RESTORE FROM PLATFORM 'Microsoft Windows x86 64-bit'
FOREIGN DATABASE TO NEW
FROM BACKUPSET '<backup piece full name>';
```

39. Execute the following steps to configure the instance with the restored datafiles

Note: those are the same steps you used in the database cross-platform transport using image copies that you implemented in the previous practice.

a. Obtain list of the restored datafiles and take note of them.

Hint: in Putty, to highlight text vertically, keep pressing on [Alt] key then highlight with the mount pointer the text that you want.

```
host "ls -al /u01/app/oracle/oradata/ORADB2/datafile";
```

b. Create the control file using the following command.

Paste the name of the data files in their locations in the code.

```
CREATE CONTROLFILE REUSE SET DATABASE "ORADB2" RESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 292
LOGFILE
  GROUP 1 SIZE 200M BLOCKSIZE 512,
  GROUP 2 SIZE 200M BLOCKSIZE 512,
  GROUP 3 SIZE 200M BLOCKSIZE 512
DATAFILE
  '/u01/app/oracle/oradata/ORADB2/datafile/<datafile1>',
  '/u01/app/oracle/oradata/ORADB2/datafile/<datafile2>',
  '/u01/app/oracle/oradata/ORADB2/datafile/<datafile3>',
  '/u01/app/oracle/oradata/ORADB2/datafile/<datafile4>',
  '/u01/app/oracle/oradata/ORADB2/datafile/<datafile5>'
CHARACTER SET AL32UTF8;
```

c. Issue the following commands:

```
ALTER DATABASE ENABLE BLOCK CHANGE TRACKING;
ALTER DATABASE OPEN RESETLOGS UPGRADE;
ALTER TABLESPACE TEMP ADD TEMPFILE SIZE 33554432 AUTOEXTEND ON NEXT 655360 MAXSIZE
32767M;
SHUTDOWN IMMEDIATE
```

d. Exit from RMAN and invoke SQL*Plus with connecting to the local instance ORADB2

```
sqlplus / as sysdba
```

e. Issue the following commands:

```
STARTUP UPGRADE PFILE='/home/oracle/temp/pfileORADB2.ora'
@@ ?/rdbms/admin/utlirp.sql
SHUTDOWN IMMEDIATE
STARTUP PFILE='/home/oracle/temp/pfileORADB2.ora'
-- the following script recompile the PL/SQL program units, it takes a bit of time
@@ ?/rdbms/admin/utlrp.sql
```

40. Run a query to verify the data has been successfully transported and recovered.

```
SELECT COUNT(*) FROM DBA_OBJECTS WHERE OWNER='RC_OWNER';
```

Note: In real life scenario, you should consider performing further post-database-creation steps like the following:

- Create external tables
- Create directory objects
- Create SPFILE
- Set the environment variable `ORACLE_SID` to the new database
- Set the auto-start script (in Linux when Oracle Restart is not configured)
- Configure the local listener

Clean Up

41. Shutdown both `srv1` and `winsrv2`.

42. Restore `srv1` to the snapshot "**Practice 20 Start**". Start `srv1`.

43. Restore `winsrv2` to the snapshot "**Practice 20 Start**". Start `winsrv2`.

44. Delete the snapshot "Practice 20 Start" for both appliances.

45. Delete the files from the shared folder.

Summary

In this practice, you implemented the following cross-platform transport tasks:

- Perform cross-platform tablespace transport from `winsrv2` to `srv1` using backupsets with conversion at the source host.
- Perform cross-platform database transport from `winsrv2` to `srv1` using backupsets with conversion at the destination host.