Practice

Managing Data Files

Practice Target

In this practice, you will implement some tasks involved in managing data files in Oracle databases.

Specifically, you will perform the following:

- Relocate data files online
- Relocate offline data files
- Remove unavailable data files from a container

Practice Assumptions

You have the `srv1` and its CDB database up and running.

Relocating Data Files Online

In the following steps, you will relocate a data file online and perform a testing case on it.

1. Start a Putty session to `srv1` as `oracle`, invoke SQL*Plus and connect to the database as `SYS`. In the remaining steps, this session will be referred to as the **admin** session.

```
sqlplus / as sysdba
set sqlprompt "admin> "
```

2. Execute the following code to create a tablespace in `PDB1` and grant unlimited quota on it to HR.

   Note: You will learn about tablespace user quotas later in the course.

```
ALTER SESSION SET CONTAINER=PDB1;
CREATE TABLESPACE MYTBS DATAFILE '/home/oracle/mytbs1.dbf' SIZE 150M AUTOEXTEND
OFF;

ALTER USER HR QUOTA UNLIMITED ON MYTBS;
GRANT EXECUTE ON DBMS_LOCK TO HR ;
```

3. Open another Putty session and connect in it to `PDB1` as `HR`. In the remaining steps, this session will be referred to as the **client** session.

   To easily distinguish between this session and the other session, I recommend changing the font foreground color to bright green. In client session Putty window, click on the left upper corner icon > select Change **Settings** > select **Colors** > select **Default Foreground** > click on **Modify** button

```
sqlplus HR/ABcd##1234@//srv1/pdb1.localdomain
set sqlprompt "client> "
```

4. In the client session create a testing table and apply continuous updates on it as follows. The update is endless, go to next section after executing the code. Do not wait for it to finish.

```
CREATE TABLE HR.TEST( RID NUMBER, RNAME CHAR(250)) TABLESPACE MYTBS;

-- insert random data into the table
BEGIN
 FOR I IN 1..10000 LOOP
  INSERT INTO TEST (RID, RNAME) VALUES ( I, DBMS_RANDOM.STRING('U',250));
  IF MOD(I,100) =0 THEN
   COMMIT;
  END IF;
 END LOOP;
 COMMIT;
END;
/
```

```
-- keep updating the table
DECLARE
 N INTEGER :=0;
BEGIN
 WHILE (TRUE) LOOP
  UPDATE TEST SET RNAME = DBMS_RANDOM.STRING('U',250)
   WHERE RID = ROUND(DBMS_RANDOM.VALUE(1,10000));
  N := N + 1 ;
  IF N=10 THEN
   DBMS_LOCK.SLEEP(0.25);
   COMMIT;
   N := 1;
  END IF;
 END LOOP;
END;
/
```

5. In the admin session, issue the following statement to relocate the `MYTBS` datafile to a different location.

Observe the statement succeeds although the client session is still updating the table in the datafile.

Renaming the datafile in this statement has the same effect as relocating it.

```
ALTER DATABASE MOVE DATAFILE '/home/oracle/mytbs1.dbf' TO
'/home/oracle/mytbs2.dbf';
```

6. Verify that the new data file is created and the old one is not there anymore.

```
host ls -al /home/oracle/mytbs1.dbf
host ls -al /home/oracle/mytbs2.dbf
```

7. In the client session, cancel the client execution by pressing on the keys `[Ctrl]+[c]`

Relocating Data Files Offline

In the following steps, you will relocate the MYTBS data file offline.

8. In the admin session, take the data file in MYTBS offline.

The statement fails and returns the following error:

```
ORA-01145: offline immediate disallowed unless media recovery enabled
```

This error means that we need to enable the ARCHIVELOG mode in the database to make the statement succeeds. You will learn about the ARCHIVELOG mode later in the course. For now, you just execute the code in the next step to enable the ARCHIVELOG mode.

```
ALTER DATABASE DATAFILE '/home/oracle/mytbs2.dbf' OFFLINE;
```

9. Run the following code to enable the ARCHIVELOG mode in the database. It requires restarting the database instance.

```
ALTER SESSION SET CONTAINER=CDB$ROOT;
ALTER SYSTEM SET LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST'
SCOPE=SPFILE;
shutdown immediate
startup mount
ALTER DATABASE ARCHIVELOG;
ALTER DATABASE OPEN ;
ALTER PLUGGABLE DATABASE PDB1 OPEN ;
```

10. In the admin session, take the data file in MYTBS offline.

```
ALTER SESSION SET CONTAINER=PDB1;
ALTER DATABASE DATAFILE '/home/oracle/mytbs2.dbf' OFFLINE;
```

11. In the client session, reconnect to PDB1 as HR.

```
conn HR/ABcd##1234@//srv1/pdb1.localdomain
```

12. In the client session, rerun the PL/SQL block saved in the buffer. It should fail with the following error:

```
ORA-00376: file xx cannot be read at this time
ORA-01110: data file 25: '/home/oracle/mytbs2.dbf'
```

```
/
```

13. Issue the following command to copy the datafile to a new location.

The physical move is performed by the OS, not by the database.

```
host cp /home/oracle/mytbs2.dbf /home/oracle/mytbs1.dbf
```

14. In the admin session, run the either of the following statements to record the new data file location into the database dictionary.

The statement does not physically move the data file.

```
-- run ONE of the following statements:
ALTER TABLESPACE mytbs RENAME DATAFILE '/home/oracle/mytbs2.dbf' TO
'/home/oracle/mytbs1.dbf';

ALTER DATABASE RENAME FILE '/home/oracle/mytbs2.dbf' TO
'/home/oracle/mytbs1.dbf';
```

15. Take the datafile back online.

The statement fails with the following error:

`ORA-01113: file xx needs media recovery`

This error means the data file must be in consistent with the other datafiles to take it online.

```
ALTER DATABASE DATAFILE '/home/oracle/mytbs1.dbf' ONLINE;
```

16. Apply the redo logs on the `mytbs1.dbf` file.

You will learn about using RECOVER statement to apply the redo logs on data files later in the course.

```
RECOVER DATAFILE '/home/oracle/mytbs1.dbf';

ALTER DATABASE DATAFILE '/home/oracle/mytbs1.dbf' ONLINE;
```

17. In the client session, re-run the client testing block. It should continue its execution without errors.

```
/
```

18. Cancel the client execution by pressing on the keys `[Ctrl]+[c]`

Removing Unavailable Data Files from a Container

In the following steps, you will make a data file in a PDB unavailable. Then, you will go through the procedure to remove it from the database.

Important

In real life scenario, to recover from a deleted data file, you should restore it from a database backup. However, in our testing case, we assume that the database backup is not available. We also assume that it is acceptable to remove the PDB that contains the data file and lose all the data in the PDB.

19. In the admin session, delete `mytbs1.dbf` file using an operating system command.

```
host rm /home/oracle/mytbs1.dbf
```

20. In the client session, resume the code that updates the `TEST` table.

The code may continue without a failure because it does not require a read from the data file. It finds the data blocks that it wants to process in the memory.

```
/
```

21. In the admin session, submit a system checkpoint to force the database to write into the data files.

The command results in a connection interruption in each session.

```
ALTER SYSTEM CHECKPOINT ;
```

22. In the admin session, check the `OPEN_MODE` of `PDB1`.

The `MOUNTED` value in a PDB means the PDB is not available. You can verify this by trying to connect to `PDB1`. This means the entire PDB becomes unavailable as a result of losing one of its data files.

In normal cases, in such a scenario, we must restore the PDB from the backup. However, in our testing case, we assume that we do not have a backup and that we want to remove the PDB altogether.

```
-- we have two connection attempts because the first one may fail
conn / as sysdba
conn / as sysdba
SELECT OPEN_MODE, RESTRICTED FROM V$PDBS WHERE NAME='PDB1';
```

Question: Can we keep the database in its current condition and keep using the other PDBs and just ignore `PDB1`?

Answer: The other PDBs in the database keep operating as normal. However, once we need to restart the CDB, it does not open cleanly because the data files are not consistent. Therefore, we must not leave an unavailable data file registered in the database. Either we restore it or we drop it.

23. Try opening `PDB1`.

The statement fails because the data file is unavailable.

```
ALTER PLUGGABLE DATABASE PDB1 OPEN ;
```

24. Checkout the status of the unavailable data file from the `CDB_DATA_FILES`

Unfortunately, we cannot retrieve the status of the data file from the data dictionary view because the PDB is down.

```
SELECT STATUS FROM CDB_DATA_FILES WHERE FILE_NAME='/home/oracle/mytbs1.dbf';
```

25. Checkout the status of the unavailable data file from the `V$DATAFILE`

The datafile status is `ONLINE`. It was deleted from the OS but its status was not changed to `OFFLINE`.

```
SELECT STATUS FROM V$DATAFILE WHERE NAME='/home/oracle/mytbs1.dbf';
```

26. Remove `PDB1` with its all datafiles.

The only solution we end up with is to drop the entire PDB altogether.

```
DROP PLUGGABLE DATABASE PDB1 INCLUDING DATAFILES;
```

Cleanup

27. Shutdown `srv1`.

28. In Oracle VirtualBox, restore `srv1` from its **CDB** snapshot.

Summary

- A DBA can relocate a data file online from one location to another without the need to turn it offline first.
- If the database is working in `ARCHIVELOG` mode, a DBA can turn a data file offline, relocate it using a third-party tool, inform the database about its new location, and turn it online again.
- If a user permanent tablespace data file is deleted from a PDB and no backup is available, we must remove the entire PDB.