BCI

Practices for Lesson 3: Managing a Multitenant Container Database and Pluggable Databases

Practices for Lesson 3

Practices Overview

In this practice, you will perform startup and shutdown operations on CDBs, open and close operations on PDBs, and connections to PDBs to display current context.

Assumptions

`cdb1` is successfully created after Practice 2-1.

`pdb2_1` is successfully created in `cdb1` after completion of Practice 2-3.

pdb2_2 is successfully created in cdb1 after completion of Practice 2-4.

It is not necessary at this step to have successfully created `pdb1_1` and `pdb_orcl2`.

Practice 3-1: Shutdown and Startup of the CDB

Overview

In this practice you shut down `cdb1` and start up `cdb1`.

Tasks

1. Connect to the multitenant container database `cdb1` to shut it down.

a. Connect to the CDB as a user with `SYSDBA` privilege.

`From your Windows Desktop bring up sqlplus. Login in as:`

```
SQL> /@cdb1 as sysdba
Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.2 -
64bit Production
With the Partitioning, OLAP, Data Mining and Real Application
Testing options
SQL> select name, cdb, con_id from v$database;
NAME       CDB CON_ID
---------  --- ----------
CDB1       YES 0
SQL>
```

b. Shut down the CDB.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> EXIT
```

2. Connect to the multitenant container database `cdb1` and start it up.

```
From a command prompt: C:\ora12clabs>  sqlplus /@cdb1 as sysdba
Connected to an idle instance.
SQL> startup
ORACLE instance started.
Total System Global Area 1068937216 bytes
Fixed Size 2248280 bytes
Variable Size 343933352 bytes
Database Buffers 717225984 bytes
Redo Buffers 5529600 bytes
Database mounted.
Database opened.
SQL> select name, cdb, con_id from v$database;
NAME       CDB CON_ID
---------  --- ----------
CDB1       YES 0
SQL> EXIT
```

4. Explore the PDBs.

```
$ sqlplus / as sysdba
Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.2 -
```

```
64bit Production
With the Partitioning, OLAP, Data Mining and Real Application
Testing options
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- -------------------------------- ----------
2          PDB$SEED                         READ ONLY
3          PDB2_1                           MOUNTED
4          PDB2_2                           MOUNTED
5          PDB_ORCL2                        MOUNTED
6          PDB1_1                           MOUNTED
SQL>
```

5. Open all PDBs.

```
SQL> alter pluggable database all open;
Pluggable database altered.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- ------------------------------   ----------
2          PDB$SEED                         READ ONLY
3          PDB2_1                           READ WRITE
4          PDB2_2                           READ WRITE
5          PDB_ORCL2                        READ WRITE
6          PDB1_1                           READ WRITE
SQL>
```

6. Connect to any of the PDBs in your cdb1, except PDB$SEED.

```
SQL> connect sys/password@PDB2_1 AS SYSDBA
Connected.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- ------------------------------   ----------
3          PDB2_1                           READ WRITE
SQL>
```

7. Display the context of the PDB you are connected to.

```
SQL> show con_name
CON_NAME
------------------------------
PDB2_1
SQL>
```

8. Connect to another PDB left in your cdb1, except PDB$SEED.

```
SQL> connect sys/password@PDB2_2 AS SYSDBA
Connected.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- ------------------------------   ----------
4          PDB2_2                           READ WRITE
```

9. Display the context of the PDB you are connected to.

```
SQL> show con_name
CON_NAME
------------------------------
PDB2_2
SQL> EXIT
```

Practice 3-2: Closing and Opening a PDB

Overview

In this practice you close PDBs and open PDBs, and create triggers to automatically open PDBs after CDB startup.

Tasks

1. Connect to the multitenant container database `cdb1` to shut it down.

a. Connect to `cdb1` as a user with `SYSDBA` privilege.

```
$ . oraenv
ORACLE_SID = [cdb1] ? cdb1
The Oracle base remains unchanged with value /u01/app/oracle
$ sqlplus / as sysdba
Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.2 -
64bit Production
With the Partitioning, OLAP, Data Mining and Real Application
Testing options
SQL> select name, cdb, con_id from v$database;
NAME       CDB CON_ID
---------  --- ----------
CDB1       YES 0
SQL>
```

b. Shut down `cdb1`.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

c. Start up `cdb1`.

```
SQL> startup
ORACLE instance started.
Total System Global Area 1068937216 bytes
Fixed Size 2248280 bytes
Variable Size 343933352 bytes
Database Buffers 717225984 bytes
Redo Buffers 5529600 bytes
Database mounted.
Database opened.
SQL>
```

d. Notice that the PDBs are all in `MOUNTED` open mode.

```
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- -------------------------------- ----------
2          PDB$SEED                         READ ONLY
3          PDB2_1                           MOUNTED
4          PDB2_2                           MOUNTED
5          PDB_ORCL2                        MOUNTED
6          PDB1_1                           MOUNTED
SQL>
```

2. Open all PDBs manually.

```
SQL> alter pluggable database all open;
Pluggable database altered.
SQL>
```

3. Close PDB2_1.

a. Start a DML transaction in another session.

```
$ . oraenv
ORACLE_SID = [cdb1] ? cdb1
The Oracle base remains unchanged with value /u01/app/oracle
$ sqlplus sys/password@pdb2_1 as sysdba
Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.2 -
64bit Production
With the Partitioning, OLAP, Data Mining and Real Application
Testing options
SQL> create table system.mytab (c number);
Table created.
SQL> insert into system.mytab values (1);
1 row created.
SQL> commit;
Commit complete.
SQL> exit
$
```

b. In the first session, close PDB2_1 in IMMEDIATE mode.

```
SQL> alter pluggable database pdb2_1 close immediate;
Pluggable database altered.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                              OPEN_MODE
---------- ------------------------------    ----------
2          PDB$SEED                          READ ONLY
3          PDB2_1                            MOUNTED
4          PDB2_2                            READ WRITE
5          PDB_ORCL2                         READ WRITE
6          PDB1_1                            READ WRITE
SQL>
```

c. Try to connect as a user of PDB2_1.

```
SQL> connect system/password@pdb2_1
ERROR:
ORA-01033: ORACLE initialization or shutdown in progress
Process ID: 0
Session ID: 0 Serial number: 0
Warning: You are no longer connected to ORACLE.
SQL>
```

4. Open pdb2_1.

```
SQL> connect / as sysdba
Connected.
SQL> alter pluggable database PDB2_1 open;
Pluggable database altered.
SQL>
```

Reconnect to pdb2_1 and select data from SYSTEM.MYTAB table.

```
SQL> connect system/password@PDB2_1
```

```
Connected.
SQL> select * from system.mytab;
C
----------
1
SQL>
```

5. Shut down the multitenant container database `cdb1` to open and close PDBs with different clauses.

```
SQL> CONNECT / AS SYSDBA
Connected.
SQL> select name, cdb, con_id from v$database;
NAME      CDB CON_ID
--------- --- ----------
CDB1      YES 0
SQL>
```

a. Shut down `CDB1`.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

b. Start up `cdb1` in `NOMOUNT` mode.

```
SQL> startup nomount
ORACLE instance started.
Total System Global Area 1068937216 bytes
Fixed Size 2248280 bytes
Variable Size 343933352 bytes
Database Buffers 717225984 bytes
Redo Buffers 5529600 bytes
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
No rows selected.
SQL>
```

c. Mount `cdb1`.

```
SQL> alter database mount;
Database altered.
SQL>
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- -------------------------------- ----------
2          PDB$SEED                         MOUNTED
3          PDB2_1                           MOUNTED
4          PDB2_2                           MOUNTED
5          PDB_ORCL2                        MOUNTED
6          PDB1_1                           MOUNTED
SQL>
```

d. Open `cdb1`.

```
SQL> alter database open;
Database altered.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- -------------------------------- ----------
```

```
2          PDB$SEED                         READ ONLY
3          PDB2_1                           MOUNTED
4          PDB2_2                           MOUNTED
5          PDB_ORCL2                        MOUNTED
6          PDB1_1                           MOUNTED
SQL>
```

e. Open all PDBs except PDB2_2.

```
SQL> alter pluggable database all except pdb2_2 open;
Pluggable database altered.
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                             OPEN_MODE
---------- -------------------------------- ----------
2          PDB$SEED                         READ ONLY
3          PDB2_1                           READ WRITE
4          PDB2_2                           MOUNTED
5          PDB_ORCL2                        READ WRITE
6          PDB1_1                           READ WRITE
SQL>
```

Practice 3-3: Creating After Startup Trigger to Open All PDBs

Overview

In this practice, you create `AFTER STARTUP` trigger to open all PDBs of a CDB.

Tasks

1. Create a trigger in `cdb1` to open all PDBs automatically after starting up `cdb1`.

a. Create the trigger.

```
CREATE TRIGGER open_all_PDBs
AFTER STARTUP ON DATABASE
begin
execute immediate 'alter pluggable database all open';
end open_all_PDBs;
/
SQL> CREATE TRIGGER Open_All_PDBs
2 after startup on database
3 begin
4 execute immediate 'alter pluggable database ALL open';
5 end Open_All_PDBs;
6 /
Trigger created.
SQL>
```

b. Shut down `cdb1`.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

c. Start up `cdb1`.

```
SQL> startup
ORACLE instance started.
Total System Global Area 1068937216 bytes
Fixed Size 2248280 bytes
Variable Size 343933352 bytes
Database Buffers 717225984 bytes
Redo Buffers 5529600 bytes
Database mounted.
Database opened.
SQL>
```

d. Notice that the PDBs are all in `READ WRITE` open mode. Then disconnect.

```
SQL> select CON_ID, NAME, OPEN_MODE from v$pdbs;
CON_ID     NAME                           OPEN_MODE
---------- ------------------------------ ----------
2          PDB$SEED                       READ ONLY
3          PDB2_1                         READ WRITE
4          PDB2_2                         READ WRITE
5          PDB_ORCL2                      READ WRITE
6          PDB1_1                         READ WRITE
SQL>
```

Practice 3-4: Changing PDBs’ Open Mode

Overview

In this practice you will change the open mode of PDBs for specific operations.

Assumptions

If the trigger could not be created successfully, execute the following catchup script:

```
$ cd /home/oracle/ora12clabs/catchup_04_03
$ ./cr_trig.sh
$
```

Tasks

Rename the global database name for `pdb2_1` to `pdb2` in `cdb1`. For this purpose, you must open the PDB in `RESTRICTED` mode.

1. Connect to `pdb2_1`.

```
SQL> CONNECT sys/password@pdb2_1 as sysdba
Connected.
SQL>
```

2. Change the global database name for `pdb2_1` to `pdb2`.

```
SQL> alter pluggable database RENAME GLOBAL_NAME TO pdb2;
alter pluggable database RENAME global_name to pdb2
*
ERROR at line 1:
ORA-65045: pluggable database not in a restricted mode
SQL>
```

3. Close `pdb2_1`.

```
SQL> alter pluggable database close immediate;
Pluggable database altered.
SQL>
```

4. Open `pdb2_1` in restricted mode.

```
SQL> alter pluggable database open restricted;
Pluggable database altered.
SQL>
SQL> select CON_ID, NAME, OPEN_MODE, RESTRICTED from v$pdbs;
CON_ID     NAME                           OPEN_MODE  RES
---------- ------------------------------ ---------- ---
3          PDB2_1                         READ WRITE YES
SQL>
```

5. Change the global database name for `pdb2_1` to `pdb2`.

```
SQL> alter pluggable database RENAME GLOBAL_NAME TO pdb2;
Pluggable database altered.
SQL>
SQL> select CON_ID, NAME, OPEN_MODE, RESTRICTED from v$pdbs;
CON_ID     NAME                           OPEN_MODE  RES
---------- ------------------------------ ---------- ---
3          PDB2                           READ WRITE YES
SQL>
```

6. Open `PDB2`.

```
SQL> alter pluggable database close immediate;
Pluggable database altered.
SQL> alter pluggable database open;
Pluggable database altered.
SQL>
```

7. Check `PDB2` is in `READ WRITE` mode.

```
SQL> select CON_ID, NAME, OPEN_MODE, RESTRICTED from v$pdbs;
CON_ID     NAME                               OPEN_MODE  RES
---------- ------------------------------     ---------- ---
3          PDB2                               READ WRITE NO
SQL>
```

Practice 3-5: Instance Parameter Changes: Impact on PDBs (optional)

Overview

In this practice you will discover the impact of instance parameter changes on PDBs.

Tasks

1. In this example, you will use in `cdb1` the instance parameter `OPTIMIZER_USE_SQL_PLAN_BASELINES` because it is `ISPDB_MODIFIABLE` in `V$PARAMETER`.

```
SQL> CONNECT / AS SYSDBA
Connected.
SQL> select ISPDB_MODIFIABLE from v$parameter
2 where name='optimizer_use_sql_plan_baselines';
ISPDB
-----
TRUE
SQL>
```

2. Check the current value of instance parameter `OPTIMIZER_USE_SQL_PLAN_BASELINES`.

```
SQL> show parameter optimizer_use_sql_plan_baselines
NAME                                 TYPE        VALUE
------------------------------------ ----------- ---------
optimizer_use_sql_plan_baselines     boolean         TRUE
SQL> EXIT
$
```

3. Connect to `pdb2` in `cdb1` and check the current value of the same instance parameter `OPTIMIZER_USE_SQL_PLAN_BASELINES`.

1) Use netca to add the `PDB2` net service name for `pdb2` pluggable database of `cdb1` in the `tnsnames.ora` file.

```
$ netca
```

2) On the Welcome page, select the “Local Net Service Name configuration” and click Next.

3) On the Net Service Name Configuration page, accept `Add` and click Next.

4) On the Net Service Name Configuration, Service Name page, enter `pdb2` as Service Name and click Next.

5) On the Net Service Name Configuration, Select Protocols page, select TCP and click Next.

6) On the Net Service Name Configuration, TCP/IP Protocol page, enter your complete host name, for example, *`<yourservername>`*`.us.oracle.com`, or *`localhost`*, accept “Use the standard port number of 1521,” and click Next.

7) On the Net Service Name Configuration, Test page, select “No, do not test” (the pluggable database is not yet opened) and click Next.

8) On the Net Service Name Configuration, Net Service Name page, accept `pdb2` as Net Service Name and click Next.

9) On the Net Service Name Configuration, Another Net Service Name page, select No, and Next.

10) On the Net Service Name Configuration Done page, click Next.

11) When you are back on the Welcome page, click Finish.

12) Reload the listener with the new configuration:

```
$ lsnrctl reload
```

If this is not sufficient, then restart the instance.

a. Connect to pdb2 in cdb1.

```
$ sqlplus sys/password@pdb2 AS SYSDBA
Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.2 -
64bit Production
With the Partitioning, OLAP, Data Mining, Real Application
Testing
SQL> show parameter optimizer_use_sql_plan_baselines
NAME                                  TYPE        VALUE
------------------------------------  ----------- ---------
optimizer_use_sql_plan_baselines      boolean     TRUE
SQL>
```

4. Change the instance parameter value to FALSE in pdb2.

```
SQL> ALTER SYSTEM SET optimizer_use_sql_plan_baselines= FALSE
SCOPE=BOTH;
System altered.
SQL>
SQL> show parameter optimizer_use_sql_plan_baselines
NAME                                  TYPE        VALUE
------------------------------------  ----------- ---------
optimizer_use_sql_plan_baselines      boolean     FALSE
SQL>
```

5. Check the instance parameter value in other PDBs of the same CDB.

```
SQL> CONNECT sys/password@pdb2_2 AS SYSDBA
Connected.
SQL> show parameter optimizer_use_sql_plan_baselines
NAME                                  TYPE        VALUE
------------------------------------  ----------- ---------
optimizer_use_sql_plan_baselines      boolean     TRUE
SQL>
```

6. Close and open pdb2.

```
SQL> CONNECT sys/password@pdb2 AS SYSDBA
Connected.
SQL> ALTER PLUGGABLE DATABASE CLOSE IMMEDIATE;
Pluggable database altered.
SQL> ALTER PLUGGABLE DATABASE OPEN;
Pluggable database altered.
SQL> show parameter optimizer_use_sql_plan_baselines
NAME                                  TYPE        VALUE
------------------------------------  ----------- ---------
optimizer_use_sql_plan_baselines      boolean     FALSE
SQL>
```

7. Check the instance parameter value after CDB shutdown/startup both in root and PDBs.

```
SQL> connect / as sysdba
Connected.
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
SQL> startup
ORACLE instance started.
Total System Global Area 1068937216 bytes
Fixed Size 2248280 bytes
Variable Size 377487784 bytes
Database Buffers 683671552 bytes
Redo Buffers 5529600 bytes
Database mounted.
Database opened.
SQL> col VALUE format a20
SQL> select CON_ID, VALUE from V$SYSTEM_PARAMETER
2 where name ='optimizer_use_sql_plan_baselines';
CON_ID     VALUE
---------- --------------------
0          TRUE
3          FALSE
SQL> EXIT
```