**Practices for Lesson 2:**

**Creating a Multitenant**

**Container Database and**

**Pluggable Databases**

**Practices for Lesson 2**

**Practices Overview**

In this practice you will create a new CDB named cdb2 with DBCA with no PDB except the

seed.

After the CDB creation is completed, check the physical and logical structures of the new CDB.

Then, you will create several PDBs using different methods.

Create pdb2\_1 from seed in cdb2.

Clone pdb2\_2 in cdb2 from pdb2\_1.

Plug the non-CDB orclx into the CDB cdb2 as pdb\_orclx.

Merge the two CDBs cdb1 and cdb2 into cdb2, and optionally drop the database

cdb1 (optional practice).

Finally, you drop the pdb2\_3 using either DBCA or SQL Developer or SQL\*Plus after the

creation of this PDB.

**Practice 2-1: Creating a New CDB**

**Overview**

In this practice, you will create a new CDB named cdb2 with DBCA.

**Assumptions**

The created CDB cdb1 already exists.

**Tasks**

1. Create a CDB named cdb2 using DBCA. First release resources held by other instances,

shutting down the orcl, pec and cdb1 instances.

a. Shut down orclx.

$ **. oraenv**

ORACLE\_SID = [cdb1] ? **orclx**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining, Real Application

Testing

SQL> **SHUTDOWN IMMEDIATE**

Database closed.

Database dismounted.

ORACLE instance shut down.

SQL> **EXIT**

$

b. Shut down **orclx if it exists**.

$ **. oraenv**

ORACLE\_SID = [orclx] ? **orcl2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining, Real Application

Testing

SQL> **SHUTDOWN IMMEDIATE**

Database closed.

Database dismounted.

ORACLE instance shut down.

SQL> **EXIT**

$

c. Shut down cdb1.

$ **. oraenv**

ORACLE\_SID = [orclx] ? **cdb1**

The Oracle base remains unchanged with value /oracle/app

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining, Real Application

Testing

SQL> **SHUTDOWN IMMEDIATE**

Database closed.

Database dismounted.

ORACLE instance shut down.

SQL> **EXIT**

$

d. Start dbca and perform the following steps.

$ **dbca**

**Step Window/Page Description Choices or Values**

a. Step 1: Database Operation Select “Create Database”.

Click Next.

b. Step 2: Creation Mode Select “Advanced Mode”.

Click Next.

c. Step 3: Database Template Select “General Purpose or Transaction

Processing”.

Click Next.

d. Step 4: Database Identification Enter

Global Database Name: **cdb2**

SID: cdb2

Select **“Create As Container Database”**

Select **“Create An Empty Container Database”**

Click Next.

e. Step 5: Management Options Deselect “Configure Enterprise Manager (EM)

Database Express”.

Click Next.

f. Step 6: Database Credentials Select “Use same Administrative password…”

Enter: Password: password

Confirm password: password

Click Next.

g. Step 7: Network Configuration Listener Selection: Click Next

h. Step 8: Storage Locations Confirm Storage type is “File System”.

Select “Use Common Location for All Database Files”.

Click Next.

i. Step 9: Database Options Click Next.

j. Step 10: Initialization Parameters Select “Character Sets”.

Select “Use Unicode (**AL32UTF8**)”.

Click Next.

k. Step 11: Creation Option Select “Create Database”.

Click Next.

l. Step 12: Pre Requisite Checks Click Next.

m. Step 13: Summary Click Finish.

n. Step 14: Progress Page On the Database Configuration Assistant page (for

password management) click Exit.

Click Close.

**Practice 2-2: Exploring CDB and PDB Structures**

**Overview**

In this practice, you check the physical and logical structures of the new CDB cdb2 and its seed

PDB.

**Tasks**

1. Connect to the multitenant container database cdb2.

$ **. oraenv**

ORACLE\_SID = [cdb1] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL>

a. Check if the database is a multitenant container database.

SQL> **SELECT name, cdb, con\_id from v$database;**

NAME CDB CON\_ID

--------- --- ----------

CDB2 YES 0

SQL>

b. Check the instance name.

SQL> **SELECT INSTANCE\_NAME, STATUS, CON\_ID from v$instance;**

INSTANCE\_NAME STATUS CON\_ID

---------------- ------------ ----------

cdb2 OPEN 0

SQL> **EXIT**

$

2. Explore the services.

a. Check services.

$ **lsnrctl status**

LSNRCTL for Linux: Version 12.2.0.0.2 - 07-SEP-2012 01:10:16

Copyright (c) 1991, 2012, Oracle. All rights reserved.

Connecting to

(DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC1521)))

STATUS of the LISTENER

------------------------

Alias LISTENER

Version TNSLSNR for Linux: Version 12.2.0.0.2

- Production

Start Date 14-SEP-2012 03:04:56

Uptime 16 days 21 hr. 48 min. 35 sec

Trace Level off

Security ON: Local OS Authentication

SNMP OFF

Listener Parameter File

/oracle/app/product/12.2.0/dbhome\_1/network/admin/listener.ora

Listener Log File

/oracle/app/diag/tnslsnr/*yourserver*/listener/alert/log.xml

Listening Endpoints Summary...

(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=EXTPROC1521)))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=1521)

))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5501)

)(Presentation=HTTP)(Session=RAW))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5502)

)(Presentation=HTTP)(Session=RAW))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5500)

)(Presentation=HTTP)(Session=RAW))

Services Summary...

Service "**cdb2**" has 1 instance(s).

Instance "cdb2", status READY, has 1 handler(s) for this

service...

Service "cdb2XDB" has 1 instance(s).

Instance "cdb2", status READY, has 1 handler(s) for this

service...

Service "em12rep" has 1 instance(s).

Instance "em12rep", status READY, has 1 handler(s) for this

service...

Service "em12repXDB" has 1 instance(s).

Instance "em12rep", status READY, has 1 handler(s) for this

service...

The command completed successfully

$

b. List the services automatically created for each container.

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL> **col name format A20**

SQL> **SELECT name, con\_id from v$services;**

NAME CON\_ID

-------------------- ----------

cdb2XDB 1

cdb2 1

SYS$BACKGROUND 1

SYS$USERS 1

SQL>

Notice that PDB$SEED service is not listed. No one should connect to this service because

there should be no operation performed on this container. It is reserved as a template to

create other PDBs.

3. Display the pluggable databases. Use a new view V$PDBS.

SQL> **SELECT CON\_ID, NAME, OPEN\_MODE from v$pdbs;**

CON\_ID NAME OPEN\_MODE

---------- ------------------------------ ----------

2 PDB$SEED READ ONLY

SQL>

Notice that the seed PDB is in READ ONLY open mode.

4. View new family of views CDB\_*xxx*:

SQL> **connect / as sysdba**

Connected.

SQL> **col PDB\_NAME format a8**

SQL> **col CON\_ID format 999999**

SQL> **SELECT PDB\_ID, PDB\_NAME, DBID, GUID, CON\_ID**

2 **from cdb\_pdbs order by 1;**

PDB\_ID PDB\_NAME DBID GUID CON\_ID

------ -------- ---------- ---------------------------- ------

2 PDB$SEED 4012275228 203F5F3EDB7F0000000000000000 1

SQL>

5. Check all files of the CDB.

a. View the redo log files of the CDB.

SQL> **col MEMBER format A42**

SQL> **SELECT GROUP#, MEMBER, CON\_ID from v$logfile;**

GROUP# MEMBER CON\_ID

---------- ------------------------------------------ -------

3 /oracle/app/oradata/cdb2/redo03.log 0

2 /oracle/app/oradata/cdb2/redo02.log 0

1 /oracle/app/oradata/cdb2/redo01.log 0

SQL>

b. View the control files of the CDB.

SQL> **col name format A55**

SQL> **SELECT name, con\_id from v$controlfile;**

NAME CON\_ID

------------------------------------------------------- -------

/oracle/app/oradata/cdb2/control01.ctl 0

/oracle/app/fast\_recovery\_area/cdb2/control02.ctl 0

SQL>

c. View all data files of the CDB, including those of the root and all PDBs, with

CDB\_DATA\_FILES view.

SQL> **col file\_name format A65**

SQL> **SELECT FILE\_NAME, TABLESPACE\_NAME, FILE\_ID, con\_id**

2 **from cdb\_data\_files**

3 **order by con\_id ;**

FILE\_NAME TS\_NAME FILE\_ID CON\_ID

----------------------------------------- ---------- ------- ------

/oracle/app/oradata/cdb2/users01.dbf USERS 6 1

/oracle/app/oradata/cdb2/undotbs01.dbf UNDOTBS1 4 1

/oracle/app/oradata/cdb2/sysaux01.dbf SYSAUX 3 1

/oracle/app/oradata/cdb2/system01.dbf SYSTEM 1 1

/u01/……………………../cdb2/pdbseed/system01.dbf SYSTEM 5 2

/u01/………………………../cdb2/pdbseed/sysaux01.dbf SYSAUX 7 2

6 rows selected.

SQL>

d. Still connected to the root, now use DBA\_DATA\_FILES view.

SQL> **col file\_name format A42**

SQL> **col tablespace\_name format A10**

SQL> **SELECT FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from dba\_data\_files;**

FILE\_NAME TABLESPACE FILE\_ID

------------------------------------------ ---------- -------

/oracle/app/oradata/cdb2/users01.dbf USERS 6

/oracle/app/oradata/cdb2/undotbs01.dbf UNDOTBS1 4

/oracle/app/oradata/cdb2/sysaux01.dbf SYSAUX 3

/oracle/app/oradata/cdb2/system01.dbf SYSTEM 1

SQL> **EXIT**

$

Notice that only root data files are listed.

e. Start the cdb1 database.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb1**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to an idle instance.

SQL> **STARTUP**

ORACLE instance started.

Total System Global Area 400846848 bytes

Fixed Size 2271568 bytes

Variable Size 243271344 bytes

Database Buffers 146800640 bytes

Redo Buffers 8503296 bytes

Database mounted.

Database opened.

SQL> **EXIT**

$

1) Use netca to add the PDB1\_1 net service name for pdb1\_1 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 pdb1\_1 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>*, 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 pdb1\_1

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.

f. Open the pdb1\_1 pluggable database in cdb1.

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining, Real Application

Testing

SQL> **ALTER PLUGGABLE DATABASE pdb1\_1 OPEN;**

Pluggable database altered.

SQL> **EXIT**

$

g. Connect to the pdb1\_1 of cdb1, and use DBA\_DATA\_FILES view.

$ **sqlplus system/password@pdb1\_1**

SQL\*Plus: Release 12.2.0.0.2 Production on Fri Sep 7 01:28:32

2012

Copyright (c) 1982, 2012, Oracle. All rights reserved.

Last Successful login time: Wed Aug 22 2012 13:16:11 +00:00

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL> **col file\_name format A65**

SQL> **SELECT FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from dba\_data\_files;**

FILE\_NAME TABLESPACE\_NAME FILE\_ID

------------------------------------------- --------------- ------

/u01/<oracle base>/cdb1/pdb1\_1/system01.dbf SYSTEM 8

/u01//<oracle base>/cdb1/pdb1\_1/sysaux01.dbf SYSAUX 9

/u01//<oracle base>/cdb1/pdb1\_1/….sers01.dbf USERS 10

/u01//<oracle base>/cdb1/pdb1\_1/ex.ple01.dbf EXAMPLE 11

SQL>

Notice that only pdb1\_1 data files are listed.

h. Now use V$TABLESPACE and V$DATAFILE view.

SQL> **col NAME format A12**

SQL> **SELECT FILE#, ts.name, ts.ts#, ts.con\_id**

**2 from v$datafile d, v$tablespace ts**

**3 where d.ts#=ts.ts#**

**4 and d.con\_id=ts.con\_id**

**5 order by 4;**

FILE# NAME TS# CON\_ID

---------- ------------ ---------- ----------

4 UNDOTBS1 2 0

11 EXAMPLE 4 3

10 USERS 3 3

8 SYSTEM 0 3

9 SYSAUX 1 3

SQL>

i. List the temp files of the PDB.

SQL> **SELECT FILE\_NAME, TABLESPACE\_NAME from dba\_temp\_files;**

FILE\_NAME TABLESPACE\_NAME

----------------------------------------------------- --------------

/oracle/app/oradata/cdb1**/pdb1\_1/**pdb1\_1**\_**temp01.dbf TEMP

SQL> **EXIT**

$

j. List the password file and SPFILE of both cdb1 and cdb2.

$ **cd $ORACLE\_HOME/dbs**

$ **ls -l orapw\* spfile\***

-rw-r----- 1 oracle oinstall 7680 Sep 5 10:43 orapwcdb1

-rw-r----- 1 oracle oinstall 7680 Sep 7 00:48 orapwcdb2

-rw-r----- 1 oracle oinstall 7680 Sep 5 07:23 orapwem12rep

-rw-r----- 1 oracle oinstall 7680 Sep 5 10:02 orapworcl

-rw-r----- 1 oracle oinstall 7680 Sep 5 10:19 orapworcl2

-rw-r----- 1 oracle oinstall 3584 Sep 7 01:23 spfilecdb1.ora

-rw-r----- 1 oracle oinstall 3584 Sep 7 01:08 spfilecdb2.ora

-rw-r----- 1 oracle oinstall 3584 Sep 6 18:13 spfileem12rep.ora

-rw-r----- 1 oracle oinstall 3584 Sep 6 10:00 spfileorcl2.ora

-rw-r----- 1 oracle oinstall 3584 Sep 7 00:35 spfileorcl.ora

$

k. Check ADR files, directories, new DDL statement in alert.log.

$ **cd $ORACLE\_BASE/diag/rdbms/**

$ **ls**

cdb1 cdb2 em12rep orclx orcl2

$ **cd cdb2/cdb2/trace**

**$ vi alert\_cdb2.log**

**…**

Mon Feb 06 09:27:09 2012

**Fri Sep 07 00:41:54 2012**

**create pluggable database PDB$SEED as clone using**

**'/oracle/app/product/12.2.0/dbhome\_1/assistants/dbca/templat**

**es//pdbseed.xml' source\_file\_name\_convert =**

**('/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/temp01.dbf',**

**'/oracle/app/oradata/cdb2/pdbseed/pdbseed\_temp01.dbf',**

**'/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/system01.dbf'**

**,'/oracle/app/oradata/cdb2/pdbseed/system01.dbf',**

**'/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/sysaux01.dbf'**

**,'/oracle/app/oradata/cdb2/pdbseed/sysaux01.dbf') NOCOPY**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Pluggable Database PDB$SEED with pdb id - 2 is created as**

**UNUSABLE.**

**If any errors are encountered before the pdb is marked as NEW,**

**then the pdb must be dropped**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**…**

**Post plug operations are now complete.**

**Pluggable database PDB$SEED with pdb id - 2 is now marked as**

**NEW.**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Completed: create pluggable database PDB$SEED as clone using**

**'/oracle/app/product/12.2.0/dbhome\_1/assistants/dbca/templat**

**es//pdbseed.xml' source\_file\_name\_convert =**

**('/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/temp01.dbf',**

**'/oracle/app/oradata/cdb2/pdbseed/pdbseed\_temp01.dbf',**

**'/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/system01.dbf'**

**,'/oracle/app/oradata/cdb2/pdbseed/system01.dbf',**

**'/ade/b/3895122769/oracle/oradata/seeddata/pdbseed/sysaux01.dbf'**

**,'/oracle/app/oradata/cdb2/pdbseed/sysaux01.dbf') NOCOPY**

**alter pluggable database PDB$SEED open restricted**

**Pluggable database PDB$SEED dictionary check beginning**

**Pluggable Database PDB$SEED Dictionary check complete**

**…**

$

6. List all users created in the new CDB cdb2.

a. Connect to cdb2 instance.

$ **. oraenv**

ORACLE\_SID = [orcl] ? **cdb2**

The Oracle base for

ORACLE\_HOME=/oracle/app/product/12.2.0/dbhome\_1 is

/u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL>

b. Verify that the SYSTEM user is created.

SQL> **col username format A30**

SQL> **select username, common, con\_id from cdb\_users**

2 **where username ='SYSTEM';**

USERNAME COM CON\_ID

------------------------------ --- -------

SYSTEM YES 1

SYSTEM YES 2

SQL>

Notice that the user SYSTEM exists in all containers as a common user.

c. List all common users in the CDB.

SQL> **select distinct username from cdb\_users**

2 **where common ='YES' order by 1;**

USERNAME

------------------------------

ANONYMOUS

APEX\_040200

APEX\_PUBLIC\_USER

APPQOSSYS

AUDSYS

CTXSYS

DBSNMP

DIP

DVF

DVSYS

………………

ORACLE\_OCM

ORDDATA

ORDPLUGINS

ORDSYS

OUTLN

SI\_INFORMTN\_SCHEMA

SPATIAL\_CSW\_ADMIN\_USR

SPATIAL\_WFS\_ADMIN\_USR

SYS

SYSBACKUP

SYSDG

SYSKM

SYSTEM

WMSYS

XDB

XS$NULL

35 rows selected.

SQL>

d. List all local users in the CDB.

SQL> **select distinct username, CON\_ID from cdb\_users**

2 **where common ='NO';**

no rows selected

SQL>

e. List local users in root.

SQL> **select distinct username from dba\_users**

2 **where common ='NO';**

no rows selected

SQL>

Notice that there is no local user in the root container because it is impossible to create any

local user in the root.

7. View distinct accesses by different containers to the single SGA.

SQL> **select distinct status, con\_id from v\_$bh order by 2 ;**

STATUS CON\_ID

---------- ----------

cr 1

free 1

xcur 1

cr 2

xcur 2

SQL> **EXIT**

$

**Practice 2-3: Creating a PDB from Seed**

**Overview**

In this practice, you will create a new PDB pdb2\_1 in cdb2 from seed.

**Assumptions**

The creation of the CDB cdb2 is successful.

**Tasks**

Either use DBCA or SQL Developer or SQL commands.

The creation using SQL is described below.

1. Create a directory for the new data files of pdb2\_1 of cdb2.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **cd $ORACLE\_BASE/oradata/cdb2**

$ **mkdir pdb2\_1**

$

2. Run SQL\*Plus and connect to the root with a user with CREATE PLUGGABLE DATABASE

privilege.

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL> **CREATE PLUGGABLE DATABASE pdb2\_1 ADMIN USER pdb2\_1\_admin**

2 **IDENTIFIED BY password ROLES=(CONNECT)**

3 **FILE\_NAME\_CONVERT=('/oracle/app/oradata/cdb2/pdbseed'**

4 **,'/oracle/app/oradata/cdb2/pdb2\_1');**

Pluggable database created.

SQL>

3. Check the open mode of pdb2\_1.

SQL> **col con\_id format 999**

SQL> **col name format A10**

SQL> **select con\_id, NAME, OPEN\_MODE,DBID, CON\_UID from V$PDBS;**

CON\_ID NAME OPEN\_MODE DBID CON\_UID

------ ---------------------- ---------- ---------- ----------

2 PDB$SEED READ ONLY 4029890286 4029890286

3 PDB2\_1 MOUNTED 3071827262 3071827262

SQL>

4. Open pdb2\_1.

a. Open the PDB.

SQL> **alter pluggable database pdb2\_1 open;**

Pluggable database altered.

SQL> **EXIT**

b. Connect to pdb2\_1 AS SYSDBA.

1) Use netca to add the PDB2\_1 net service name for pdb2\_1 pluggable database of

cdb2 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\_1 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>*, 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\_1

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.

$ **sqlplus sys/password@pdb2\_1 AS SYSDBA**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics, Real

Application Testing

SQL>

5. The service is now available and registered with the listener.

SQL> **!lsnrctl status**

The command completed successfully

LSNRCTL for Linux: Version 12.2.0.0.2 - Production on 07-SEP-

2012 01:47:28

Copyright (c) 1991, 2012, Oracle. All rights reserved.

Connecting to

(DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC1521)))

STATUS of the LISTENER

------------------------

Alias LISTENER

Version TNSLSNR for Linux: Version 12.2.0.0.2

- Production

Start Date 14-SEP-2012 03:04:56

Uptime 16 days 22 hr. 0 min. 0 sec

Trace Level off

Security ON: Local OS Authentication

SNMP OFF

Listener Parameter File

/oracle/app/product/12.2.0/dbhome\_1/network/admin/listener.o

ra

Listener Log File

/oracle/app/diag/tnslsnr/*yourserver*/listener/alert/log.xml

Listening Endpoints Summary...

(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=EXTPROC1521)))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=1521)

))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5501)

)(Presentation=HTTP)(Session=RAW))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5502)

)(Presentation=HTTP)(Session=RAW))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=*yourserver*)(PORT=5500)

)(Presentation=HTTP)(Session=RAW))

Services Summary...

Service "cdb1" has 1 instance(s).

Instance "cdb1", status READY, has 1 handler(s) for this

service...

Service "cdb1XDB" has 1 instance(s).

Instance "cdb1", status READY, has 1 handler(s) for this

service...

Service "**cdb2**" has 1 instance(s).

Instance "cdb2", status READY, has 1 handler(s) for this

service...

Service "cdb2XDB" has 1 instance(s).

Instance "cdb2", status READY, has 1 handler(s) for this

service...

Service "orclx" has 1 instance(s).

Instance "orclx", status READY, has 1 handler(s) for this

service...

Service "em12repXDB" has 1 instance(s).

Instance "em12rep", status READY, has 1 handler(s) for this

service...

Service "**pdb1\_1**" has 1 instance(s).

Instance "cdb1", status READY, has 1 handler(s) for this

service.

Service "**pdb2\_1**" has 1 instance(s).

Instance "cdb2", status READY, has 1 handler(s) for this

service.

The command completed successfully

SQL>

6. Connect to pdb2\_1 as sys user by using EasyConnect and then as pdb2\_1\_admin user.

SQL> **CONNECT sys/password@localhost:1521/pdb2\_1 AS SYSDBA**

Connected.

SQL> **connect pdb2\_1\_admin/password@PDB2\_1**

Connected.

SQL> **show con\_name**

CON\_NAME

------------------------------

PDB2\_1

SQL>

7. List the data files created.

SQL> !**ls -l $ORACLE\_BASE/oradata/cdb2/pdb2\_1/\***

-rw-r----- 1 oracle oinstall 20979712 Sep 7 01:47

/oracle/app/oradata/cdb2/pdb2\_1/pdbseed\_temp01.dbf

-rw-r----- 1 oracle oinstall 671096832 Sep 7 01:47

/oracle/app/oradata/cdb2/pdb2\_1/sysaux01.dbf

-rw-r----- 1 oracle oinstall 262152192 Sep 7 01:47

/oracle/app/oradata/cdb2/pdb2\_1/system01.dbf

SQL>

8. Check the services, data files, and tablespaces using views.

SQL> **connect system/password@pdb2\_1**

Connected.

SQL> **col name format A30**

SQL> **select name from v$services;**

NAME

----------------------------

pdb2\_1

SQL> **col file\_name format A50**

SQL> **col tablespace\_name format A8**

SQL> **col file\_id format 99**

SQL> **col con\_id format 9**

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID, con\_id**

**2 from cdb\_data\_files**

**3 order by con\_id ;**

FILE\_NAME TABLESPACE NAME FILE\_ID CON\_ID

---------------------------------------- --------------- ------- -----

/<oracle base>//cdb2/pdb2\_1/sysaux01.dbf SYSAUX 9 3

//<oracle base>//cdb2/pdb2\_1/system01.dbf SYSTEM 8 3

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

**2 from dba\_data\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

----------------------------------------- --------------- -------

/<oracle base>/cdb2/pdb2\_1/system01.dbf SYSTEM 8

/<oracle base>/cdb2/pdb2\_1/sysaux01.dbf SYSAUX 9

SQL> **col file\_name format A60**

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from cdb\_temp\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

--------------------------------------------- -------------- ------

/<oracle base>/cdb2/pdb2\_1/pdbseed\_temp01.dbf TEMP 3

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from dba\_temp\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

--------------------------------- --------------- ------

/<oracle base>/cdb2/pdb2\_1/ pdbseed\_temp01.dbf TEMP 3

SQL>

9. To be able to view all objects of all containers in the CDB, connect to the root and use

CDB\_*xxx* views.

SQL> **connect / as sysdba**

Connected.

SQL> **show con\_id**

**CON\_ID**

**------------------------------**

**1**

SQL> **show con\_name**

CON\_NAME

------------------------------

CDB$ROOT

SQL> **select name from v$services;**

NAME

------------------------------

pdb2\_1

cdb2XDB

cdb2

SYS$BACKGROUND

SYS$USERS

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID, con\_id**

**2 from cdb\_data\_files**

**3 order by con\_id, file\_id ;**

FILE\_NAME TABLESPACE NAME FILE\_ID CON\_ID

--------------------------------- -------------- ------ -----

/<oracle base>/cdb2/system01.dbf SYSTEM1 1 1

/<oracle base>/cdb2/sysaux01.dbf SYSAUX 3 1

/<oracle base>/cdb2/undotbs01.dbf UNDOTBS1 4 1

/<oracle base>/cdb2/users01.dbf USERS 6 1

/<oracle base>/pdbs.d/system01.dbf SYSTEM 5 2

/<oracle base>/pdbseed/sysaux01.dbf SYSAUX 7 2

/<oracle base>/pdb2\_1/system01.dbf SYSTEM 8 3

/<oracle base>/pdb2\_1/sysaux01.dbf SYSAUX 9 3

8 rows selected.

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from dba\_data\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

----------------------------------------- -------------- ------

/<oracle base>/cdb2/users01.dbf USERS 6

/<oracle base>/cdb2/undotbs01.dbf UNDOTBS1 4

/oracle/app/oradata/cdb2/sysaux01.dbf SYSAUX 3

/oracle/app/oradata/cdb2/system01.dbf SYSTEM 1

**SQL> select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from cdb\_temp\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

-------- -------

/oracle/app/oradata/cdb2/temp01.dbf TEMP 1

/<oracle base>/pdbseed/pdbseed\_temp01.dbf TEMP 2

/<oracle base>/pdb2\_1/pdbseed\_temp01.dbf TEMP 3

SQL> **select FILE\_NAME, TABLESPACE\_NAME, FILE\_ID**

2 **from dba\_temp\_files;**

FILE\_NAME TABLESPACE NAME FILE\_ID

------------------------------------------- -------------- -------

/oracle/app/oradata/cdb2/temp01.dbf TEMP 1

SQL> **EXIT**

$

**Practice 2-4: Cloning PDB Within the Same CDB**

**Overview**

In this practice, you will create a new PDB with the cloning method, cloning pdb2\_2 from

pdb2\_1 within the same CDB cdb2.

**Assumptions**

The pdb2\_1 creation has completed successfully in Practice 3-3.

**Tasks**

**Either use the SQL commands OR SQL Developer.**

**Method with SQL\*Plus.**

1. Create a directory for the new data files of pdb2\_2 of cdb2.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **cd $ORACLE\_BASE/oradata/cdb2**

$ **mkdir pdb2\_2**

$

2. Run SQL\*Plus and connect to the root as a user granted with CREATE PLUGGABLE

DATABASE privilege.

a. Set pdb2\_1 in READ ONLY open mode before cloning.

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL> **alter pluggable database pdb2\_1 close;**

Pluggable database altered.

SQL> **alter pluggable database pdb2\_1 open read only;**

Pluggable database altered.

b. Change OMF DB\_CREATE\_FILE\_DEST parameter value to

'/oracle/app/oradata/cdb2/pdb2\_2'.

SQL> **alter system set db\_create\_file\_dest =**

**'/oracle/app/oradata/cdb2/pdb2\_2'**;

System altered.

c. Clone pdb2\_2 from pdb2\_1.

SQL> **CREATE PLUGGABLE DATABASE pdb2\_2 FROM pdb2\_1;**

Pluggable database created.

3. Check the open mode of pdb2\_2.

SQL> **select name, open\_mode from v$pdbs;**

NAME OPEN\_MODE

------------------------------ ----------

PDB$SEED READ ONLY

PDB2\_1 READ ONLY

PDB2\_2 MOUNTED

SQL>

4. Set PDB2\_1 in READ WRITE open mode and open PDB2\_2.

a. Open PDB2\_1 in READ WRITE mode.

SQL> **alter pluggable database PDB2\_1 close;**

Pluggable database altered.

SQL> **alter pluggable database PDB2\_1 open;**

Pluggable database altered.

SQL>

b. Open PDB2\_2 in READ WRITE mode.

SQL> **alter pluggable database PDB2\_2 open;**

Pluggable database altered.

SQL> **EXIT**

$

c. Connect to pdb2\_2 AS SYSDBA.

1) Use netca to add the PDB2\_2 net service name for pdb2\_2 pluggable database of

cdb2 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\_2 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>*, 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\_2

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.

$ **sqlplus sys/password@pdb2\_2 AS SYSDBA**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL>

d. Check the open mode of the PDBs.

SQL> **CONNECT / AS SYSDBA**

Connected.

SQL> **select name, open\_mode from v$pdbs;**

NAME OPEN\_MODE

------------------------------ ----------

PDB$SEED READ ONLY

PDB2\_1 READ WRITE

PDB2\_2 READ WRITE

SQL>

5. Connect to PDB2\_2 as the SYSTEM user.

SQL> **connect system/password@PDB2\_2**

Connected.

SQL> **show con\_name**

PDB2\_2

SQL> **EXIT**

$

6. List the data files created.

$ **cd $ORACLE\_BASE/oradata/cdb2/pdb2\_2**

$ **ls -l**

total 4

drwxr-x--- 3 oracle oinstall 4096 Feb 6 13:46 CDB2

$ **cd CDB2**

$ **ls -l**

total 4

drwxr-x--- 3 oracle oinstall 4096 Jun 29 21:26

C3A419D23D3F1DE5E043160200C04142

$ **cd C3A419D23D3F1DE5E043160200C04142**

$ **ls –l**

total 4

drwxr-x--- 2 oracle oinstall 4096 Jun 29 21:26 datafile

$ **cd datafile**

**$ ls -l**

total 809836

-rw-r----- 1 oracle oinstall 566239232 Jun 29 21:27 01\_mf\_sysaux\_7yw7d38o\_.dbf

-rw-r----- 1 oracle oinstall 262152192 Jun 29 21:27 o1\_mf\_system\_7yw78h\_.dbf

-rw-r----- 1 oracle oinstall 20979712 Jun 29 21:26 o1\_mf\_temp\_7ywg92\_.dbf

**Method with SQL Developer:**

1. If you already created pdb2\_2 with SQL\*Plus and would like to test the creation with SQL

Developer, you first have to drop pdb2\_2 to recreate it.

a. Drop the pluggable database pdb2\_2.

$ **sqlplus / AS SYSDBA**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL> **ALTER PLUGGABLE DATABASE pdb2\_2 CLOSE IMMEDIATE;**

Pluggable database altered.

SQL> **DROP PLUGGABLE DATABASE pdb2\_2 INCLUDING DATAFILES;**

Pluggable database dropped.

SQL> **EXIT**

b. Remove the directory.

$ **rm -r $ORACLE\_BASE/oradata/cdb2/pdb2\_2**

$

2. Create a directory for the new data files of pdb2\_2 of cdb2.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **cd $ORACLE\_BASE/oradata/cdb2**

$ **mkdir pdb2\_2**

$

3. Run SQL\*Plus and connect to the root to set OMF directory to the pdb2\_2 directory.

$ **sqlplus / AS SYSDBA**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL> **alter system set db\_create\_file\_dest =**

**'/oracle/app/oradata/cdb2/pdb2\_2' scope=both**;

System altered.

SQL> **EXIT**

$

4. Launch SQL Developer.

$ **cd $ORACLE\_HOME/sqldeveloper**

$ **./sqldeveloper.sh**

$

5. Create a connection as SYS in root cdb2.

6. Open a connection as SYS in cdb2.

a. Choose the View option.

b. Click Connections.

c. Click + in the left Connections pane to add a new connection.

d. Fill the different fields as follows: be sure to change the host name and port number to

your assigned host name and port number.

**Window/Page Description Choices or Values**

Connection Name cdb2\_root\_SYS

Username sys

Password password

Connection Type TNS

Role SYSDBA

Network Alias cdb2

e. Click Test.

f. If the status is Success, click Save.

g. Click Connect.

7. To manage the CDB and its PDBs:

a. Choose the View option.

b. Click DBA.

c. Click + in the left Connections pane to view an existing connection.

d. From the list of existing connections, choose the one you just created.

e. Click OK.

f. Click the sign + in front of the name of the cdb2\_root\_SYS connection to expand the folder. Then click the sign + in front of “Container Database”. The list of containers in the CDB appears.

8. Right-click the pluggable database pdb2\_1 to show possible actions.

Choose Modify State to set it in READ ONLY open mode before cloning.

a. First close.

b. Click Apply.

c. Choose Modify State again.

d. Set the State Option to READ ONLY.

e. Click Apply then OK.

9. Right-click the pluggable database pdb2\_1 and choose Clone Pluggable Database….

a. Fill the different fields as follows.

**Window/Page Description Choices or Values**

Database Name pdb2\_2

Source PDB pdb2\_1

File Name Conversions None

File Name Conversions kept to None means that it uses the OMF target destination

declared in DB\_CREATE\_FILE\_DEST parameter.

b. You can view the SQL statement before applying.

c. Click Apply then OK. The new pdb2\_2 appears in the list of PDBs in cdb2.

d. Open pdb2\_2.

1) Open pdb2\_2 in READ WRITE mode. Right click pdb2\_2 and click Modify State.

2) Click Apply, then OK.

e. Open pdb2\_1 in READ WRITE mode. Right click pdb2\_1 and click Modify State. Click

Apply , then OK.

10. Leave SQL Developer.

a. Click File.

b. Then click exit.

**Practice 2-5: Plugging a Non-CDB into a CDB**

**Overview**

In this practice, you will plug the non-CDB orclx into the CDB cdb2. You will not use

Export/Import DataPump, which can be a possible method, but the method with DBMS\_PDB

package. This package executed in the non-CDB orclx generates an XML file describing the

tablespaces and data files of non-CDB orclx. The XML file is then used when creating

pdb\_orclx in cdb2.

**Tasks**

1. Use DBMS\_PDB.DESCRIBE to “unplug” non-CDB orclx.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **orclx**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to an idle instance.

SQL> **startup mount**

ORACLE instance started.

Total System Global Area 1670221824 bytes

Fixed Size 2274000 bytes

Variable Size 973081904 bytes

Database Buffers 687865856 bytes

Redo Buffers 7000064 bytes

Database mounted.

SQL>

SQL> **alter database open read only;**

Database altered.

SQL> **exec dbms\_pdb.describe**

**('/u01/oracle/app/oradata/orcl2/xmlorclx.xml')**

PL/SQL procedure successfully completed.

SQL> **shutdown immediate**

Database closed.

Database dismounted.

ORACLE instance shut down.

SQL> **EXIT**

$

2. Create a new PDB pdb\_orclx to plug non-CDB orclx into cdb2 using the XML file

generated. You will have to remove the temp file because the creation cannot complete until

it is removed to create it.

$ **. oraenv**

ORACLE\_SID = [orcl2] ? **cdb2**

The Oracle base for

ORACLE\_HOME=/oracle/app/product/12.2.0/dbhome\_1 is

/u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL> **create pluggable database PDB\_ORCLx using**

**'/oracle/app/oradata/orclx/xmlorcl2.xml' NOCOPY;**

create pluggable database PDB\_ORCL2 using 'xmlorcl2' NOCOPY

\*

ERROR at line 1:

ORA-01119: error in creating database file

'/oracle/app/oradata/orcl2/temp01.dbf'

ORA-27038: created file already exists

Additional information: 1

SQL**>!rm /u01/oracle/app/oradata/orclx/temp01.dbf**

SQL>

SQL> **create pluggable database PDB\_ORCLx using**

**'/u01/oracle/app/oradata/orclx/xmlorclx.xml' NOCOPY;**

Pluggable database created.

SQL> **EXIT**

$

3. To complete the operation, you have to convert the plugged non-CDB to a proper PDB by

deleting unnecessary metadata from PDB SYSTEM tablespace.

For this purpose, you execute the $ORACLE\_HOME/rdbms/admin/noncdb\_to\_pdb.sql

script whilst connected to the PDB.

a. Connect to pdb2\_2 as SYSDBA.

1) Use netca to add the PDB\_ORCLx net service name for pdb\_orcl2 pluggable

database of cdb2 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 pdb\_orclx

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>*, 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

pdb\_orcl2 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.

b. Now connect to pdb\_orclx using the net service name.

$ **sqlplus sys/password@pdb\_orclx as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL>

c. Execute the script. Expect around 35 minutes to complete.

SQL> **@$ORACLE\_HOME/rdbms/admin/noncdb\_to\_pdb.sql**

SQL> SET SERVEROUTPUT ON

SQL> SET FEEDBACK 1

SQL> SET NUMWIDTH 10

SQL> SET LINESIZE 80

SQL> SET TRIMSPOOL ON

SQL> SET TAB OFF

SQL> SET PAGESIZE 100

SQL>

SQL> WHENEVER SQLERROR EXIT;

SQL>

SQL> DOC

DOC>############################################################

###########

DOC>############################################################

###########

DOC> The following statement will cause an "ORA-01722: invalid

number"

DOC> error if we're not in a PDB.

DOC>############################################################

###########

DOC>############################################################

###########

DOC>#

SQL>

SQL> VARIABLE pdbname VARCHAR2(128)

SQL> BEGIN

2 SELECT sys\_context('USERENV', 'CON\_NAME')

3 INTO :pdbname

4 FROM dual

5 WHERE sys\_context('USERENV', 'CON\_NAME') <> 'CDB$ROOT';

6 END;

7 /

PL/SQL procedure successfully completed.

…

SQL>

SQL> Rem

================================================================

=

SQL> Rem Run component validation procedure

SQL> Rem

================================================================

=

SQL>

SQL> EXECUTE dbms\_registry\_sys.validate\_components;

...Database user "SYS", database schema "APEX\_040200", user#

"98" 10:21:02

...Compiled 0 out of 2998 objects considered, 0 failed

compilation 10:21:03

...263 packages

...255 package bodies

...453 tables

...11 functions

...16 procedures

...3 sequences

...458 triggers

...1322 indexes

...207 views

...0 libraries

...6 types

...0 type bodies

...0 operators

...0 index types

...Begin key object existence check 10:21:03

...Completed key object existence check 10:21:03

...Setting DBMS Registry 10:21:03

...Setting DBMS Registry Complete 10:21:03

...Exiting validate 10:21:03

PL/SQL procedure successfully completed.

SQL> SET serveroutput off

SQL>

SQL> Rem

================================================================

=

SQL> Rem END utlrp.sql

SQL> Rem

================================================================

=

SQL>

…

SQL> alter pluggable database "&pdbname" close;

Pluggable database altered.

SQL>

SQL> -- leave the PDB in the same state it was when we started

SQL> BEGIN

2 execute immediate '&open\_sql &restricted\_state';

3 EXCEPTION

4 WHEN OTHERS THEN

5 BEGIN

6 IF (sqlcode <> -900) THEN

7 RAISE;

8 END IF;

9 END;

10 END;

11 /

PL/SQL procedure successfully completed.

SQL>

SQL> alter session set container="&pdbname";

Session altered.

SQL>

SQL> WHENEVER SQLERROR CONTINUE;

SQL>

SQL>

d. Quit the session after opening the new PDB.

SQL**> alter pluggable database pdb\_orcl2 open;**

Pluggable database altered.

**SQL> EXIT**

$

4. Connect to PDB\_ORCLx.

$ **sqlplus sys/password@localhost:1521/PDB\_ORCLx as SYSDBA**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Advanced Analytics and Real

Application Testing options

SQL>

5. Verify that the application data is in the PDB pdb\_orcl2:

SQL> **select count(empno) from scott.emp;**

COUNT(EMPNO)

------------

14

SQL> **EXIT**

$

**Practice 2-6: Merging All PDBs of CDBs into a Single CDB**

**Overview**

In this practice you merge all PDBs of cdb1 into a single CDB, cdb2.

1. Merge all PDBs of cdb1 into cdb2.

2. Drop cdb1.

**Assumptions**

The CDB cdb2 exists. The cdb2 creation has completed successfully in Practice 3-1.

**Tasks**

1. Connect to the multitenant container database cdb1 to unplug all PDBs.

a. Connect to cdb1 root as a common user with ALTER PLUGGABLE DATABASE

privilege to unplug pdb1\_1. If the pdb1\_1 is still in READ WRITE mode, close the

PDB.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb1**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL> **select name, open\_mode from v$pdbs;**

NAME OPEN\_MODE

------------------------------ ----------

PDB$SEED READ ONLY

PDB1\_1 READ WRITE

SQL> **alter pluggable database PDB1\_1 unplug into**

**'xmlfilePDB1\_1.xml';**

alter pluggable database PDB1\_1 unplug into 'xmlfilePDB1\_1'

\*

ERROR at line 1:

ORA-65025: Pluggable database PDB1\_1 is not closed on all

instances.

SQL> **alter pluggable database PDB1\_1 close immediate;**

Pluggable database altered.

SQL> **alter pluggable database PDB1\_1 unplug into**

**'xmlfilePDB1\_1.xml';**

Pluggable database altered.

SQL> **col PDB\_NAME format A20**

SQL> **select PDB\_NAME, STATUS from CDB\_PDBS**

**where PDB\_NAME='PDB1\_1';**

PDB\_NAME STATUS

-------------------- -------------

PDB1\_1 **UNPLUGGED**

SQL> **drop pluggable database PDB1\_1 KEEP DATAFILES;**

Pluggable database dropped.

SQL> **EXIT**

$

b. Before plugging pdb1\_1 into cdb2, you can optionally check whether the unplugged

pdb1\_1 is compatible with cdb2 with DBMS\_PDB.CHECK\_PLUG\_COMPATIBILITY

function. Connect to cdb2 root as a common user with CREATE PLUGGABLE

DATABASE privilege to plug pdb1\_1.

Use the following PL/SQL code:

**DECLARE**

**compat BOOLEAN := FALSE;**

**BEGIN**

**compat := DBMS\_PDB.CHECK\_PLUG\_COMPATIBILITY(**

**pdb\_descr\_file =>**

**'/u01/oracle/app/product/12.2.0/dbhome\_1/dbs/xmlfilePDB1\_1.xml',**

**pdb\_name => 'pdb1\_1');**

**if compat then**

**DBMS\_OUTPUT.PUT\_LINE('Is pluggable compatible? YES');**

**else DBMS\_OUTPUT.PUT\_LINE('Is pluggable compatible? NO');**

**end if;**

**end;**

/

$ **. oraenv**

ORACLE\_SID = [cdb1] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected.

SQL> **SET SERVEROUTPUT ON**

SQL> **DECLARE**

2 **compat BOOLEAN := FALSE;**

3 **BEGIN**

4 **compat := DBMS\_PDB.CHECK\_PLUG\_COMPATIBILITY(**

5 **pdb\_descr\_file =>**

**'/oracle/app/product/12.2.0/dbhome\_1/dbs/xmlfilePDB1\_1.xml',**

**pdb\_name => 'pdb1\_1');**

6 **if compat then**

7 **DBMS\_OUTPUT.PUT\_LINE('Is pluggable compatible? YES');**

8 **else DBMS\_OUTPUT.PUT\_LINE('Is pluggable compatible? NO');**

9 **end if;**

10 **end;**

11 **/**

Is pluggable compatible? **NO**

PL/SQL procedure successfully completed.

SQL>

c. If the value returned is YES, you can immediately proceed with step d.

If the value returned is NO, examine the PDB\_PLUG\_IN\_VIOLATIONS view to see why

it is not compatible.

SQL> **select message, action from pdb\_plug\_in\_violations**

**where name='PDB1\_1';**

MESSAGE

----------------------------------------------------------------

ACTION

----------------------------------------------------------------

Parameter sga\_target mismatch: Previous value 503316480. CDB

value 5033164800

Change the parameter in PDB or the CDB

Parameter pga\_aggregate\_target mismatch: Previous value

167772160. CDB value 1677721600

Change the parameter in PDB or the CDB

SQL>

The message refers to a parameter related to PGA. The parameter will not have any

impact if you create the PDB. You can proceed with the creation of the PDB.

d. Plug pdb1\_1 into cdb2.

SQL> **create pluggable database pdb1\_1 using 'xmlfilePDB1\_1.xml'**

**NOCOPY;**

Pluggable database created.

SQL>

Notice that you use the clause NOCOPY because the cdb2 pdb1\_1 files are located in the

right place. Otherwise, you should have described the target destination to move the files

from the source to the new destination.

e. Open pdb1\_1.

SQL> **alter pluggable database pdb1\_1 open;**

Pluggable database altered.

SQL>

f. Check that the pdb1\_1 is in the PDBs list in cdb2.

SQL> **select name, open\_mode from v$pdbs;**

NAME OPEN\_MODE

------------------------------ ----------

PDB$SEED READ ONLY

PDB2\_1 READ WRITE

PDB2\_2 READ WRITE

PDB\_ORCL2 READ WRITE

PDB1\_1 READ WRITE

SQL> **EXIT**

$

2. After all PDBs are unplugged from cdb1 (in case you had created other PDBs) and

plugged into cdb2, you can drop the multitenant container database cdb1 with DBCA or

SQL commands.

$ **. oraenv**

ORACLE\_SID = [cdb2] ? **cdb1**

The Oracle base remains unchanged with value /u01/app/oracle

$ **sqlplus / as sysdba**

Connected.

SQL> **shutdown immediate**

Database closed.

Database dismounted.

ORACLE instance shut down.

SQL> **startup mount restrict**

ORACLE instance started.

Total System Global Area 722366464 bytes

Fixed Size 2276928 bytes

Variable Size 213909952 bytes

Database Buffers 503316480 bytes

Redo Buffers 2863104 bytes

Database mounted.

SQL> **DROP DATABASE;**

Database dropped.

SQL> **EXIT**

$

Remove archived logs and backups if necessary.

**Practice 3-7: Dropping a PDB (optional)**

**Overview**

In this practice, you drop the PDB pdb2\_3 of cdb2 that you quickly create from the seed.

**Tasks**

Either use DBCA or SQL Developer or SQL commands.

The creation using SQL is described below.

1. Create a directory for the new data files of pdb2\_3 of cdb2.

$ **. oraenv**

ORACLE\_SID = [cdb1] ? **cdb2**

The Oracle base remains unchanged with value /u01/app/oracle

$ **cd $ORACLE\_BASE/oradata/cdb2**

$ **mkdir pdb2\_3**

$

2. Run SQL\*Plus and connect to the root with a user with CREATE PLUGGABLE DATABASE

privilege.

$ **sqlplus / as sysdba**

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.0.2 -

64bit Production

With the Partitioning, OLAP, Data Mining and Real Application

Testing options

SQL> **CREATE PLUGGABLE DATABASE pdb2\_3 ADMIN USER pdb2\_3\_admin**

**IDENTIFIED BY password ROLES=(CONNECT)**

**FILE\_NAME\_CONVERT=('/oracle/app/oradata/cdb2/pdbseed'**

**,'/u01/oracle/app/oradata/cdb2/pdb2\_3');**

Pluggable database created.

SQL>

3. Check the open mode of pdb2\_3.

SQL> **col con\_id format 999**

SQL> **col name format A10**

SQL> **select NAME, OPEN\_MODE from V$PDBS;**

NAME OPEN\_MODE

------------------------------ ----------

PDB$SEED READ ONLY

PDB2\_1 READ WRITE

PDB2\_2 READ WRITE

PDB\_ORCL2 READ WRITE

PDB1\_1 READ WRITE

PDB2\_3 MOUNTED

SQL>

4. Open pdb2\_3.

SQL> **alter pluggable database pdb2\_3 open;**

Pluggable database altered.

SQL>

5. Still connected to cdb2 as a common user with ALTER PLUGGABLE DATABASE privilege,

you close pdb2\_3 if the PDB is not already in MOUNTED mode.

SQL> **alter pluggable database PDB2\_3 close immediate;**

Pluggable database altered.

SQL>

6. You drop the PDB pdb2\_3 and the data files.

SQL> **drop pluggable database PDB2\_3 INCLUDING DATAFILES;**

Pluggable database dropped.

SQL> **EXIT**

$