1 多租户体系结构

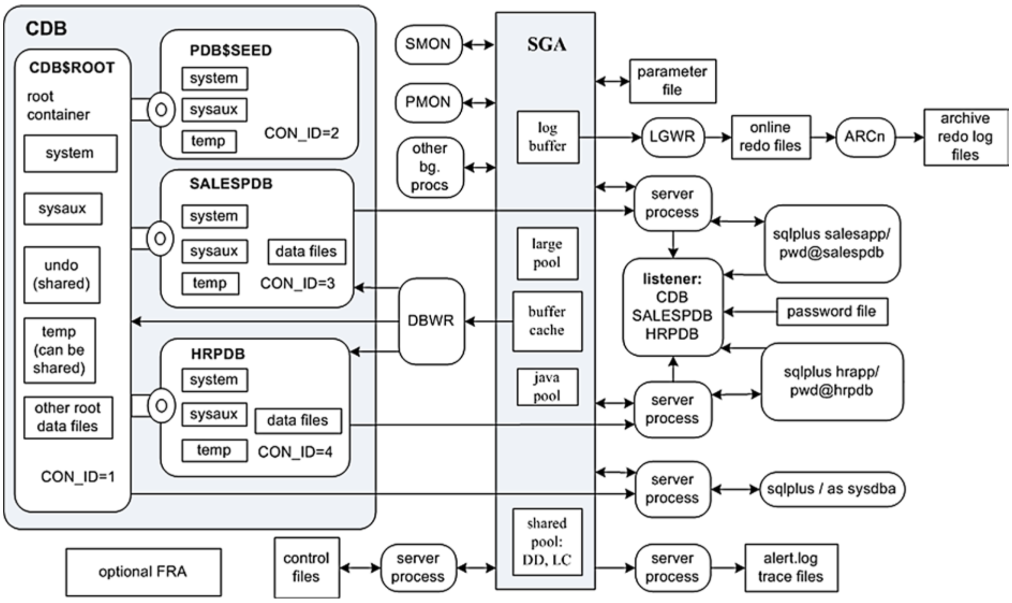
多租户是Oracle 12c的新特性，该特性能够让您在多租户容器数据库（CDB）中创建和维护多个可插拔数据库（PDB）。

多租户容器数据库（CDB）被定义为可以容纳一个或多个插件数据库的数据库，容器被定义为存在于CDB中数据文件和元数据的集合，可插拔数据库是一种特殊类型的容器，它可以通过克隆其他数据库很容易的进行创建，如果需要，可插拔数据库也可以从一个CDB迁移到另一个CDB中。

每个CDB是由一个根容器（CDB$ROOT）、一个种子容器（PDB$SEED）和0个、一个或多个可插拔数据库组成，其中，根容器包含一组主数据文件和元数据，种子容器是用于创建其它可插拔数据库，可插拔数据库包括它自己的数据文件和应用程序对象（用户、表，索引等）。12c之前的数据库都是非容器数据库（non-CDB）。

> 多租户特性需要额外收费。

下图展示了可插拔数据库的体系结构：



2 创建容器数据库CDB

2.1 使用DBCA创建

参照DBCA建库。

2.2 使用Create Database语句创建

1）指定实例SID

```
[oracle@strong ~]$ export ORACLE_SID=db01
```

2）设置环境变量

```
[oracle@strong ~]$ echo $ORACLE_SID
db01
[oracle@strong ~]$ echo $ORACLE_HOME
/u01/app/oracle/product/12.2.0/dbhome_1
```

3）创建密码文件

```
[oracle@strong ~]$ orapwd file='/u01/app/oracle/product/12.2.0/dbhome_1/dbs/orapwdb01'
format=12.2

Enter password for SYS:
```

4）创建文本初始化参数文件

```
[oracle@strong ~]$ vim initdb01.ora
db_name='db01'
memory_target=400M
control_files='/u01/app/oracle/oradata/db01/control01.ctl','/u01/app/oracle/oradata/db01/control02.ctl'
enable_pluggable_database=true
```

5）连接实例db01

```
[oracle@strong ~]$ sqlplus /nolog

SQL*Plus: Release 12.2.0.1.0 Production on Tue Aug 7 13:25:18 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.

SQL> conn / as sysdba
Connected to an idle instance.
```

6）创建服务器参数文件

```
SQL> create spfile from pfile='/home/oracle/initdb01.ora';

File created.
```

7）启动实例

```
SQL> startup nomount
```

```
ORACLE instance started.

Total System Global Area  419430400 bytes
Fixed Size                  8621424 bytes
Variable Size             301990544 bytes
Database Buffers          104857600 bytes
Redo Buffers                3960832 bytes
```

8）创建目录

```
[oracle@strong ~]$ cd /u01/app/oracle/oradata/
[oracle@strong oradata]$ mkdir db01
[oracle@strong oradata]$ cd db01/
[oracle@strong db01]$ mkdir pdbseed
```

9）创建建库脚本

```
[oracle@strong ~]$ cat db01.sql
CREATE DATABASE db01
  USER SYS IDENTIFIED BY "Foxconn168!"
  USER SYSTEM IDENTIFIED BY "Foxconn168!"
  LOGFILE GROUP 1 ('/u01/app/oracle/oradata/db01/redo01.log')
             SIZE 100M BLOCKSIZE 512,
          GROUP 2 ('/u01/app/oracle/oradata/db01/redo02.log')
             SIZE 100M BLOCKSIZE 512,
          GROUP 3 ('/u01/app/oracle/oradata/db01/redo03.log')
             SIZE 100M BLOCKSIZE 512
  MAXLOGHISTORY 1
  MAXLOGFILES 16
  MAXLOGMEMBERS 3
  MAXDATAFILES 1024
  CHARACTER SET AL32UTF8
  NATIONAL CHARACTER SET AL16UTF16
  EXTENT MANAGEMENT LOCAL
  DATAFILE '/u01/app/oracle/oradata/db01/system01.dbf'
    SIZE 700M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED
  SYSAUX DATAFILE '/u01/app/oracle/oradata/db01/sysaux01.dbf'
    SIZE 550M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED
  DEFAULT TABLESPACE deftbs
     DATAFILE '/u01/app/oracle/oradata/db01/deftbs01.dbf'
     SIZE 500M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED
  DEFAULT TEMPORARY TABLESPACE tempts1
     TEMPFILE '/u01/app/oracle/oradata/db01/temp01.dbf'
     SIZE 20M REUSE AUTOEXTEND ON NEXT 640K MAXSIZE UNLIMITED
  UNDO TABLESPACE undotbs1
     DATAFILE '/u01/app/oracle/oradata/db01/undotbs01.dbf'
     SIZE 200M REUSE AUTOEXTEND ON NEXT 5120K MAXSIZE UNLIMITED
  ENABLE PLUGGABLE DATABASE
    SEED
     FILE_NAME_CONVERT = ('/u01/app/oracle/oradata/db01/',
                          '/u01/app/oracle/oradata/db01/pdbseed/')
    LOCAL UNDO ON;
```

10）执行建库脚本

```
SQL> @db01.sql

Database created.
```

> 创建容器数据库初始化参数enable_pluggable_database必须设置为TRUE，除此之外，所需的目录也必须提前创建，否则执行脚本时会出错。

11）执行脚本

```
SQL> @?/rdbms/admin/catcdb.sql
```

出现以下提示时，在参数1处输入目录，参数2处输入文件名称：

```
SQL> host perl -I &&rdbms_admin &&rdbms_admin_catcdb --logDirectory &&1 --logFilename &&2
Enter value for 1: /home/oracle
Enter value for 2: create_cdb.log
```

3 管理Root容器

3.1 连接Root容器

以SYS用户连接到Root容器可以执行与数据库相关的所有任务，可以通过OS身份验证或网络方式进行连接。

1）OS认证方式

```
[oracle@strong ~]$ sqlplus / as sysdba

SQL*Plus: Release 12.2.0.1.0 Production on Mon Aug 6 15:52:23 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.
```

```
Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SQL> show user con_id con_name user
USER is "SYS"

CON_ID
------------------------------
1

CON_NAME
------------------------------
CDB$ROOT
USER is "SYS"
```

2）网络方式连接

```
[oracle@strong ~]$ sqlplus sys/system@orcl as sysdba

SQL*Plus: Release 12.2.0.1.0 Production on Mon Aug 6 15:54:54 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SQL>
```

3.2 展示当前连接的容器信息

1）Show命令

```
SQL> show user con_id con_name user
USER is "SYS"

CON_ID
------------------------------
1

CON_NAME
------------------------------
CDB$ROOT
```

```
USER is "SYS"
```

2）SQL语句

```
SQL> SELECT sys_context('USERENV','CON_ID') con_id,sys_context('USERENV','CON_NAME')
con_name,sys_context('USERENV','SESSION_USER') curr_user FROM DUAL;

CON_ID     CON_NAME                 CURR_USER
---------- ------------------------------ ------------------------------
1          CDB$ROOT                 SYS
```

3.3 启动/关闭Root容器

1）启动Root容器

```
ORACLE instance started.

Total System Global Area  788529152 bytes
Fixed Size                  8797728 bytes
Variable Size             595591648 bytes
Database Buffers          180355072 bytes
Redo Buffers                3784704 bytes
Database mounted.
Database opened.
```

2）打开与Root容器关联的PDB

启动ROOT容器后，但不会启动与之关联的PDB，故需要手动启动。

```
SQL> SELECT con_id,NAME,open_mode FROM v$containers;

CON_ID NAME                           OPEN_MODE
------ ------------------------------ ----------
     1 CDB$ROOT                       READ WRITE
     2 PDB$SEED                       READ ONLY
     3 ORCLPDB                        MOUNTED
SQL> alter pluggable database all open;

Pluggable database altered.
```

3）关闭Root容器

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

3.4 创建公共用户

在插件环境中，有两种类型的用户，分别是本地用户和公共用户。本地用户是在可插拔数据库中创建的普通用户，对其的管理和非容器数据库的管理相同；公共用户是Oracle 12c中的一个新概念，它只属于可插拔数据库环境，公共用户是一个存在于root容器和每个可插拔数据库中的用户，这种类型的用户必须最先在root容器创建，并在所有现有的可插拔数据库以及未来创建的任何可插拔数据库中自动创建。

> SYS和SYSTEM是Oracle在可插拔环境创建的公共用户。

公共用户的创建必须以C##或c##开头，例如：

```
[oracle@strong ~]$ sqlplus / as sysdba

SQL*Plus: Release 12.2.0.1.0 Production on Mon Aug 6 16:20:56 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SQL> create user c##stu identified by stu;

User created.
```

3.5 创建公共角色

公共角色的创建必须以C##或C##开头，例如：

```
SQL> conn / as sysdba
Connected.
SQL> create role c##select container=all;

Role created.
SQL> grant create session , select any table to c##select;

Grant succeeded.

SQL> grant create session , select any table to c##select container=all;
```

```
Grant succeeded.

SQL> grant c##select to c##stu container=all;

Grant succeeded.
```

3.6 报告容器空间

```
WITH x AS
 (SELECT c1.con_id,
         cf1.tablespace_name,
         SUM(cf1.bytes) / 1024 / 1024 fsm
    FROM cdb_free_space cf1,
         v$containers   c1
   WHERE cf1.con_id = c1.con_id
   GROUP BY c1.con_id,
            cf1.tablespace_name),
y AS
 (SELECT c2.con_id,
         cd.tablespace_name,
         SUM(cd.bytes) / 1024 / 1024 apm
    FROM cdb_data_files cd,
         v$containers   c2
   WHERE cd.con_id = c2.con_id
   GROUP BY c2.con_id,
            cd.tablespace_name)
SELECT x.con_id,
       v.name con_name,
       x.tablespace_name,
       x.fsm free,
       y.apm alloc
  FROM x,
       y,
       v$containers v
 WHERE x.con_id = y.con_id
   AND x.tablespace_name = y.tablespace_name
   AND v.con_id = y.con_id
UNION
SELECT vc2.con_id,
       vc2.name,
       tf.tablespace_name,
       NULL,
       SUM(tf.bytes) / 1024 / 1024
 FROM v$containers   vc2,
      cdb_temp_files tf
 WHERE vc2.con_id = tf.con_id
 GROUP BY vc2.con_id,
          vc2.name,
          tf.tablespace_name
 ORDER BY 1,
          2;
```

3.7 切换容器

```
SQL> alter session set container=orclpdb;

Session altered.

SQL> show con_name

CON_NAME
------------------------------
ORCLPDB
SQL> alter session set container=cdb$root;

Session altered.

SQL> show con_name

CON_NAME
------------------------------
CDB$ROOT
```

4 创建可插拔数据库PDB

4.1 克隆种子数据库

创建可插拔数据库的第一种方法是克隆种子数据库，是利用Create Pluggable Database语句通过拷贝种子数据库的数据文件来创建可插拔数据库，需要以SYS用户连接到Root容器。

```
SQL> conn / as sysdba
Connected.
SQL> create pluggable database salespdb admin user salesadm identified by system
file_name_convert=
('/u01/app/oracle/oradata/orcl/pdbseed/','/u01/app/oracle/oradata/orcl/salespdb/');
```

```
Pluggable database created.
```

4.2 克隆已存在的可插拔数据库

创建可插拔数据库的第二种方法是克隆已存在的可插拔数据库，可以通过本地或远端的方式创建可插拔数据库，若从远端数据库克隆，则需要创建DBLink。

```
SQL> conn / as sysdba
Connected.
SQL> alter pluggable database salespdb close immediate;

Pluggable database altered.

SQL> alter pluggable database salespdb open read only;

Pluggable database altered.

SQL> create pluggable database salespdb2 from salespdb file_name_convert=
('/u01/app/oracle/oradata/orcl/salespdb','/home/oracle/pdbs/salespdb2');

Pluggable database created.
```

4.3 从非容器数据库克隆

创建可插拔数据库的第三种方法是将non_CDB数据库转换为可插拔数据库。

1）将non-CDB置为只读模式

```
SQL> startup mount
ORACLE instance started.

Total System Global Area  788529152 bytes
Fixed Size                  8625656 bytes
Variable Size             633340424 bytes
Database Buffers          142606336 bytes
Redo Buffers                3956736 bytes
Database mounted.
SQL> alter database open read only;

Database altered.
```

2）利用DBMS_PDB创建XML文件，该文件用于描述non-CDB的结构

```
SQL> exec dbms_pdb.describe(pdb_descr_file => '/home/oracle/ncdb.xml');

PL/SQL procedure successfully completed.
```

3）关闭non-CDB数据库

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

4）连接CDB数据库

```
[oracle@strong ~]$ export ORACLE_SID=db01
[oracle@strong ~]$ sqlplus /nolog

SQL*Plus: Release 12.2.0.1.0 Production on Mon Aug 6 21:34:57 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.

SQL> conn / as sysdba
Connected to an idle instance.
SQL> startup
ORACLE instance started.

Total System Global Area  788529152 bytes
Fixed Size                  8797728 bytes
Variable Size             624951776 bytes
Database Buffers          150994944 bytes
Redo Buffers                3784704 bytes
Database mounted.
Database opened.
```

5）检查兼容性（可选）

```
DECLARE
  hold_var BOOLEAN;
BEGIN
  hold_var := DBMS_PDB.CHECK_PLUG_COMPATIBILITY(pdb_descr_file => '/home/oracle/ncdb.xml');
  IF hold_var THEN
    dbms_output.put_line('YES');
  ELSE
```

```
    dbms_output.put_line('NO');
  END IF;
END;
```

如果返回值为YES，说明没有兼容性问题，如果为NO，则说明可插拔数据库兼容性有问题，可通过视图pdb_plug_in_violations查看详细信息。

6）利用non-CDB创建可插拔数据库

```
SQL> create pluggable database orclpdb using '/home/oracle/ncdb.xml' copy file_name_convert=
('/u01/app/oracle/oradata/orcl/','/u01/app/oracle/oradata/db01/orclpdb/');

Pluggable database created.
```

7）以SYS用户连接刚创建的PDB数据库，并执行脚本

```
[oracle@strong ~]$ sqlplus /nolog

SQL*Plus: Release 12.2.0.1.0 Production on Mon Aug 6 21:58:28 2018

Copyright (c) 1982, 2016, Oracle.  All rights reserved.

SQL> conn sys/system@192.168.56.102:1521/orclpdb as sysdba
Connected.
SQL> @?/rdbms/admin/noncdb_to_pdb.sql
```

> 该脚本必须在PDB第一次打开之前执行。

4.4 从容器数据库拔掉可插拔数据库

创建可插拔数据库的第四种方法是将PDB拔掉，然后插入另外的容器数据库中，拔掉会产生描述PDB的XML文件，该文件用于将其插入其他CDB容器。

1）以SYS用户连接容器，关闭PDB

```
SQL> conn / as sysdba
Connected.
SQL> alter pluggable database salespdb2 close immediate;

Pluggable database altered.
```

2）拔掉PDB

```
SQL> alter pluggable database salespdb2 unplug into '/home/oracle/pdbs/salespdb2/salespdb2.xml';

Pluggable database altered.
```

> XML包含PDB的元数据信息，PDB拔掉后，在插上原来CDB之前必须先删掉，才可以插入原CDB。

3）删除PDB

```
SQL> drop pluggable database salespdb2 keep datafiles;

Pluggable database dropped.
```

4.5 插上已拔掉的可插拔数据库

创建可插拔数据库的第五种方法是将从其它容器拔掉的PDB插入新的容器中，在插入之前，必须检查兼容性（endianness），可以使用DBMS_PDB包进行检查。

1）检查兼容性

```
DECLARE
  hold_var BOOLEAN;
BEGIN
  hold_var := dbms_pdb.check_plug_compatibility(pdb_descr_file =>
'/home/oracle/pdbs/salespdb2/salespdb2.xml');
  IF hold_var
  THEN
    dbms_output.put_line('YES');
  ELSE
    dbms_output.put_line('NO');
  END IF;
END;
```

2）插入PDB

```
SQL> conn / as sysdba
Connected.
SQL> create pluggable database salespdb2 using '/home/oracle/pdbs/salespdb2/salespdb2.xml'
nocopy tempfile reuse;

Pluggable database created.
```

3）打开PDB

```
SQL> alter pluggable database salespdb2 open;

Pluggable database altered.
```

4.6 使用DBCA创建可插拔数据库

创建可插拔数据库的第六种方法是使用DBCA工具，可以使用图形界面方式，也可以使用静默方式，这里使用静默方式演示：

```
[oracle@strong ~]$ dbca -silent -createPluggableDatabase -sourceDB orcl -pdbName hrpdb -
createPDBFrom DEFAULT -pdbAdminUserName pdbadmin -pdbAdminPassword system -
pdbDatafileDestinat
ion /home/oracle/pdbs/hrpdb
Creating Pluggable Database
2% complete
6% complete
11% complete
20% complete
46% complete
Completing Pluggable Database Creation
53% complete
Executing Post Configuration Actions
100% complete
Look at the log file "/u01/app/oracle/cfgtoollogs/dbca/orcl/hrpdb/orcl.log" for further details.
```

> 可通过上面的日志查看创建过程的信息。

4.7 检查可插拔数据库的状态

1）查看所有可插拔数据库的状态

```
SQL> SELECT pdb_id,pdb_name,status FROM cdb_pdbs;

PDB_ID PDB_NAME                       STATUS
------ ------------------------------ ----------
     3 ORCLPDB                        NORMAL
     2 PDB$SEED                   NORMAL
     4 SALESPDB                   NORMAL
     6 HRPDB                      NORMAL
     5 SALESPDB2                  NORMAL
```

2）查看可插拔数据库的打开/关闭状态

```
SQL> SELECT con_id,name,open_mode FROM v$pdbs;

    CON_ID NAME                           OPEN_MODE
---------- ------------------------------ ----------
         2 PDB$SEED                       READ ONLY
         3 ORCLPDB                        READ WRITE
         4 SALESPDB                       READ ONLY
         5 SALESPDB2                 READ WRITE
         6 HRPDB                          READ WRITE
```

5 管理可插拔数据库PDB

5.1 连接PDB

1）本地方式

```
SQL> conn / as sysdba
Connected.
SQL> alter session set container=orclpdb;

Session altered.

SQL> show con_name;

CON_NAME
------------------------------
ORCLPDB
```

2）网络方式

```
SQL> conn sys/system@strong.oracle.com:1521/salespdb as sysdba
Connected.
SQL> show con_name

CON_NAME
------------------------------
SALESPDB
```

5.2 管理PDB的监听

如果启动一个默认监听（不使用listener.ora文件），那么PMON后台进程会自动将所有数据库（包括PDB数据库）注册为服务；如果使用了listener.ora文件，则监听器会将该文件中出现的服务名进行静态注册。

默认情况下，PDB注册的服务名与其数据库名字相同。可以SYS用户连接root容器通过如下查询来验证哪些服务正在运行：

```
SQL> conn / as sysdba
Connected.
SQL> select name, network_name, pdb from v$services order by pdb, name;

NAME                       NETWORK_NAME             PDB
------------------------------ -------------------- ----------
SYS$BACKGROUND                                      CDB$ROOT
SYS$USERS                                CDB$ROOT
orcl                 orcl           CDB$ROOT
orclXDB              orclXDB             CDB$ROOT
hrpdb                      hrpdb               HRPDB
orclpdb              orclpdb             ORCLPDB
salespdb             salespdb       SALESPDB
salespdb2                  salespdb2     SALESPDB2

8 rows selected.
```

也可以通过lsnrctl services查看服务。

除了默认服务外， Oracle建议可以配置额外的服务，以供应用程序访问，可以使用SRVCTL命令创建，也可使用DBMS_SERVICE包创建，这里将演示如何使用DBMS_SERVICE创建服务：

```
SQL> conn sys/system@strong.oracle.com:1521/orclpdb as sysdba
Connected.
SQL> SELECT con_id, name, open_mode FROM v$pdbs;

   CON_ID NAME                    OPEN_MODE
---------- ------------------------------ ----------
      3 ORCLPDB                   READ WRITE
SQL> exec dbms_service.create_service(service_name=>'ORCLTEST',network_name=>'ORCLTEST');

PL/SQL procedure successfully completed.

SQL> exec dbms_service.start_service(service_name=>'ORCLTEST');

PL/SQL procedure successfully completed.

SQL> conn sys/system@strong.oracle.com:1521/orcltest as sysdba
Connected.
SQL> show con_name
```

```
CON_NAME
------------------------------
ORCLPDB
```

5.3 显示当前连接的PDB

1）Show命令

```
SQL> show con_id con_name user

CON_ID
------------------------------
3

CON_NAME
------------------------------
ORCLPDB
USER is "SYS"
```

2 ）SQL查询

```
SQL> SELECT sys_context('USERENV','CON_ID') con_id,sys_context('USERENV','CON_NAME')
con_name,sys_context('USERENV','SESSION_USER') curr_user FROM DUAL;

CON_ID     CON_NAME                      CURR_USER
--------- ------------------------------ --------------------
3         ORCLPDB                        SYS
```

> SYS_CONTEXT除了显示以上信息外，还可以显示其他信息，例如：
> ```
> SELECT sys_context('USERENV', 'SERVICE_NAME') AS service_name,
>        sys_context('USERENV', 'DB_UNIQUE_NAME') AS db_unique_name,
>        sys_context('USERENV', 'INSTANCE_NAME') AS instance_name,
>        sys_context('USERENV', 'SERVER_HOST') AS server_host
>   FROM dual;
> ```

5.4 启动/关闭PDB

有两种方式可以启动/关闭PDB，分别为：以SYS用户连接root容器，或者以SYS用户直接连接PDB。

1）连接root容器

```
SQL> conn / as sysdba
Connected.
SQL> alter pluggable database salespdb close immediate ;
```

```
Pluggable database altered.

SQL> alter pluggable database all close immediate;

Pluggable database altered.

SQL> alter pluggable database orclpdb open;

Pluggable database altered.

SQL> startup pluggable database hrpdb open read only;
Pluggable Database opened.

SQL> alter pluggable database all close immediate;

Pluggable database altered.

SQL> alter pluggable database all open;

Pluggable database altered.
```

2）连接PDB

```
SQL> conn sys/system@strong.oracle.com:1521/orclpdb as sysdba
Connected.
SQL> shutdown immediate
Pluggable Database closed.
SQL>
SQL> startup
Pluggable Database opened.
```

5.5 修改PDB初始化参数

Oracle允许以特权用户连接到PDB时修改一些初始化参数，可以通过以下查询查看这些参数：

```
SELECT * FROM v$parameter t WHERE t.ispdb_modifiable = 'TRUE';
```

当连接到某个PDB时，对初始化参数的修改只会影响直接连接的PDB，不会影响root容器和其它PDB，例如：

```
SQL> conn sys/system@strong.oracle.com:1521/orclpdb as sysdba
Connected.
SQL> alter system set open_cursors=200;
```

```
System altered.
```

> 该修改只对orclpdb有效，并在数据库重启后保持不变。

```
SQL> conn / as sysdba
Connected.
SQL> show parameter open_cursors;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
open_cursors                         integer     300
```

5.6 重命名PDB

有时，可能需要重命名PDB，例如，数据库创建时可能命名错误，或者可能不再使用，并希望将_OLD附加到它的名称中，下面演示如何对PDB进行重命名。

1）重命名PDB，首先以SYSDBA连接

```
SQL> conn sys/system@strong.oracle.com:1521/salespdb2 as sysdba
Connected.
```

2）关闭PDB，以restrict模式启动

```
SQL> shutdown immediate
Pluggable Database closed.
SQL> startup restrict
Pluggable Database opened.
```

3）重命名PDB

```
SQL> alter pluggable database salespdb2 rename global_name to salespdb2_old;

Pluggable database altered.

SQL> show con_name

CON_NAME
------------------------------
SALESPDB2_OLD
```

5.7 限制PDB所占用的空间

1）以SYS用户登录PDB

```
SQL> conn sys/system@strong.oracle.com:1521/orclpdb as sysdba
Connected.
```

2）进行空间限制

```
SQL> alter pluggable database orclpdb storage(maxsize 10G);

Pluggable database altered.
```

5.8 查看PDB历史信息

通过查询数据字典CDB_PDB_HISTORY可以查看PDB的历史信息：

```
SQL> SELECT db_name,
       con_id,
       pdb_name,
       operation,
       op_timestamp,
       cloned_from_pdb_name
  FROM cdb_pdb_history
 WHERE con_id > 2
 ORDER BY con_id;

DB_NAME              CON_ID PDB_NAME             OPERATION  OP_TIMESTAMP
CLONED_FROM_PDB_NAME
-------------------- ------ -------------------- ---------- ------------------- --------------------
SEEDDATA                  3 PDB$SEED             UNPLUG     2017-01-26 15:30:11
ORCL                      3 ORCLPDB              CREATE     2018-08-06 15:20:45 PDB$SEED
ORCL                      3 PDB$SEED             PLUG       2018-08-06 15:05:45 PDB$SEED
SEEDDATA                  4 PDB$SEED             UNPLUG     2017-01-26 15:30:11
ORCL                      4 SALESPDB             CREATE     2018-08-06 17:32:29 PDB$SEED
ORCL                      4 PDB$SEED             PLUG       2018-08-06 15:05:45 PDB$SEED
SEEDDATA                  6 PDB$SEED             UNPLUG     2017-01-26 15:30:11
ORCL                      6 HRPDB                CREATE     2018-08-07 10:07:07 PDB$SEED
ORCL                      6 PDB$SEED             PLUG       2018-08-06 15:05:45 PDB$SEED

9 rows selected.
```

5.9 删除PDB

1）以SYS用户连接root容器，并关闭PDB

```
SQL> conn / as sysdba
Connected.
SQL> alter pluggable database salespdb2_old close immediate;

Pluggable database altered.
```

2）删除PDB及其数据文件

```
SQL> drop pluggable database salespdb2_old including datafiles;

Pluggable database dropped.
```

> 不加 including datafiles则表示仅仅删除数据库，数据文件仍在，可以将其插入其他CDB。