Practice 5-1: Shutting Down and Starting Up the Oracle Database

Objective:

To stop all running databases and listeners, create a new CDB named `CDBLAB` with three PDBs using DBCA, and ensure that all databases are running at the end of the practice.

Steps:

1. **Identify and Stop All Running Databases and Listeners**

   First, identify all running databases and listeners:

   ```
   pgrep -lf smon
   pgrep -lf tns
   ```

   Stop all running databases:

   ```
   sqlplus / as sysdba
   SHUTDOWN IMMEDIATE;
   ```

   Stop all running listeners:

   ```
   lsnrctl stop
   ```

2. **Set Oracle Environment Variables**

   Set the Oracle environment variables using the `oraenv` script:

   ```
   . oraenv
   ORACLE_SID = [oracle] ? CDBLAB
   ```

3. **Create a New CDB with Three PDBs Using DBCA**

   Use the following command to create a new CDB named `CDBLAB` with three PDBs (`PDB1` , `PDB2` , and `PDB3`):

   ```
   dbca -silent -createDatabase \
   -templateName General_Purpose.dbc \
   -gdbName CDBLAB -sid CDBLAB \
   -createAsContainerDatabase true \
   -numberOfPDBs 3 \
   -pdbName PDB1,PDB2,PDB3 \
   -pdbAdminPassword YourPDBPassword \
   -sysPassword fenago \
   -systemPassword fenago \
   -datafileDestination '/u01/app/oracle/oradata' \
   -storageType FS \
   -characterSet AL32UTF8 \
   -nationalCharacterSet AL16UTF16 \
   -createListener LISTENER \
   -listenerPort 1521 \
   -emConfiguration DBEXPRESS \
   -emExpressPort 5501 \
   -memoryMgmtType auto_sga \
   ```

```
-totalMemory 2048 \
-redoLogFileSize 50 \
-automaticMemoryManagement false \
-enableArchive true
```

4. **Verify the CDB and PDBs Creation**

   Connect to the CDB and verify the creation of the PDBs:

```
sqlplus / as sysdba
SELECT con_id, name, open_mode FROM v$pdbs;
```

5. **Start and Stop the CDB and PDBs**

   Shutdown the CDB:

```
SHUTDOWN IMMEDIATE;
```

   Startup the CDB in NOMOUNT Mode:

```
STARTUP NOMOUNT;
```

   Mount the CDB:

```
ALTER DATABASE MOUNT;
```

   Open the CDB:

```
ALTER DATABASE OPEN;
```

   Open All PDBs:

```
ALTER PLUGGABLE DATABASE ALL OPEN;
```

   Verify the status of the PDBs:

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

6. **Set the PDB Save States**

   Save the state of the PDBs so they open automatically on CDB startup:

```
ALTER PLUGGABLE DATABASE ALL SAVE STATE;
```

7. **Check the PDB Save States**

   Verify the saved states:

```
SELECT con_id, con_name, state FROM DBA_PDB_SAVED_STATES;
```

8. **Stop the CDB**

   Shut down the CDB:

```
SHUTDOWN IMMEDIATE;
```

9. **Start the CDB and Verify PDB States**

   Start the CDB and verify the PDB states:

   ```
   STARTUP;
   SELECT con_id, name, open_mode FROM v$pdbs;
   ```

10. **Ensure All PDBs Are Running**

    Ensure all PDBs are open:

    ```
    ALTER PLUGGABLE DATABASE ALL OPEN;
    SELECT con_id, name, open_mode FROM v$pdbs;
    ```

11. **Exit SQL*Plus**

    Exit SQL*Plus:

    ```
    EXIT;
    ```

12. **Start the Listener**

    Start the Oracle listener:

    ```
    lsnrctl start
    ```

Summary

By following these steps, you will successfully identify and stop all running databases and listeners, create a new CDB named `CDBLAB` with three PDBs, and ensure that all databases are running at the end of the practice. This practice helps in understanding the complete lifecycle management of Oracle databases using DBCA and SQL*Plus commands.

Addendum: Verifying Created Files and Running Processes

Objective:

To verify all the files created for the new CDB (`CDBLAB`) and its three PDBs (`PDB1` , `PDB2` , `PDB3`), and to find the processes that are running for all of them.

Steps:

1. **Open a New Terminal**

   Open a new terminal window.

2. **Verify Files Created for CDB and PDBs**

   Navigate to the Oracle datafile destination directory:

   ```
   cd /u01/app/oracle/oradata/CDBLAB
   ls -l
   ```

   Expected Files in `CDBLAB` Directory:

   - **Control Files:** `control01.ctl` , `control02.ctl`
   - **Redo Log Files:** `redo01.log` , `redo02.log` , `redo03.log`

- **System Data Files:** `system01.dbf`, `sysaux01.dbf`
- **Temporary Tablespace:** `temp01.dbf`
- **Undo Tablespace:** `undotbs01.dbf`
- **PDB Seed Files:** `pdbseed/system01.dbf`, `pdbseed/sysaux01.dbf`

Explanation:

- **Control Files:** Essential for database startup, containing metadata about the database structure.
- **Redo Log Files:** Used for recovery purposes, storing all changes made to the database.
- **System Data Files:** Contain the core data dictionary and system information.
- **Temporary Tablespace:** Used for temporary storage of data during SQL operations.
- **Undo Tablespace:** Stores undo information for transactions.
- **PDB Seed Files:** Template files for creating new PDBs.

3. **Verify Files for Each PDB**

   Navigate to each PDB's directory and list the files:

   For `PDB1`:

   ```
   cd /u01/app/oracle/oradata/CDBLAB/PDB1
   ls -l
   ```

   For `PDB2`:

   ```
   cd /u01/app/oracle/oradata/CDBLAB/PDB2
   ls -l
   ```

   For `PDB3`:

   ```
   cd /u01/app/oracle/oradata/CDBLAB/PDB3
   ls -l
   ```

   Expected Files in Each PDB Directory:

   - **System Data File:** `system01.dbf`
   - **Sysaux Data File:** `sysaux01.dbf`
   - **Temporary Tablespace:** `temp01.dbf`
   - **Users Tablespace:** `users01.dbf`

   Explanation:

   - **System Data File:** Contains the core data dictionary and system information for the PDB.
   - **Sysaux Data File:** Auxiliary data for the PDB.
   - **Temporary Tablespace:** Temporary storage for SQL operations within the PDB.
   - **Users Tablespace:** Default tablespace for user data within the PDB.

4. **Find Running Processes for the CDB and PDBs**

   Identify all running processes for the CDB and PDBs using the `pgrep` command:

   ```
   pgrep -lf smon
   ```

   Expected Output:

- You should see entries like `ora_smon_CDBLAB`, `ora_smon_PDB1`, `ora_smon_PDB2`, and `ora_smon_PDB3`.

Explanation:

- **SMON Process:** System Monitor process for each database, responsible for instance recovery, cleaning up temporary segments, and other tasks.

Check the listener process:

```
pgrep -lf tns
```

Expected Output:

- You should see an entry like `tnslsnr LISTENER`.

Explanation:

- **Listener Process:** Manages incoming client connections to the Oracle database.

5. **Detailed Process Information**

   For more detailed information about each process, use the `ps` command:

   ```
   ps -ef | grep smon
   ```

   Explanation:

   - This command provides detailed information about the `smon` processes, including the user running them, start time, and more.

6. **Checking Database Status in SQL*Plus**

   Connect to SQL*Plus to check the status of the CDB and PDBs:

   ```
   sqlplus / as sysdba
   ```

   Check the Status of the PDBs:

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

   Expected Output:

   - You should see the following:

     ```
     CON_ID    NAME       OPEN_MODE
     -------   -------    ----------
     2         PDB$SEED   READ ONLY
     3         PDB1       READ WRITE
     4         PDB2       READ WRITE
     5         PDB3       READ WRITE
     ```

   Explanation:

   - **PDB$SEED:** Template for creating new PDBs, should be in READ ONLY mode.
   - **PDB1, PDB2, PDB3:** User-defined PDBs, should be in READ WRITE mode.

Summary

By following these detailed steps, you will verify the files created for the new CDB and its PDBs and identify the running processes associated with them. This addendum ensures a thorough understanding of the Oracle database's file structure and process management.

Summary: Important Files and Scripts for Managing `CDBLAB` and PDBs

Oracle Home Directory Structure

The Oracle Home directory structure is as follows:

```
/u01/app/oracle/product/19.3.0/
└── dbhome_1/
    ├── bin/
    ├── config/
    ├── dbs/
    ├── include/
    ├── inventory/
    ├── javavm/
    ├── jdbc/
    ├── jlib/
    ├── lib/
    ├── network/
    ├── OPatch/
    ├── oracore/
    ├── perl/
    ├── plsql/
    ├── precomp/
    ├── racg/
    ├── rdbms/
    ├── relnotes/
    ├── slax/
    ├── sqlplus/
    ├── srvm/
    ├── suptools/
    ├── uc4/
    ├── ucp/
    └── xdk/
```

Important Directories and Files

1. **bin/**:
    - **Location**: `/u01/app/oracle/product/19.3.0/dbhome_1/bin/`
    - **Description**: Contains executable files for Oracle utilities and commands.
    - **Important Files**:
        - `sqlplus` : Command-line tool to connect to the Oracle database.
        - `lsnrctl` : Listener control utility.
        - `dbca` : Database Configuration Assistant.
2. **dbs/**:
    - **Location**: `/u01/app/oracle/product/19.3.0/dbhome_1/dbs/`

- **Description**: Contains database initialization parameter files and password files.
- **Important Files**:
  - `initCDBLAB.ora` : Initialization parameter file for `CDBLAB` .
  - `spfileCDBLAB.ora` : Server parameter file for `CDBLAB` .

3. **network/admin/**:
   - **Location**: `/u01/app/oracle/product/19.3.0/dbhome_1/network/admin/`
   - **Description**: Contains network configuration files.
   - **Important Files**:
     - `listener.ora` : Configuration file for Oracle Net Listener.
     - `tnsnames.ora` : Network service names configuration file.
     - `sqlnet.ora` : Oracle Net Services configuration file.

4. **rdbms/admin/**:
   - **Location**: `/u01/app/oracle/product/19.3.0/dbhome_1/rdbms/admin/`
   - **Description**: Contains SQL scripts for database creation, management, and maintenance.
   - **Important Files**:
     - `catalog.sql` : Script to create the Data Dictionary.
     - `catproc.sql` : Script to compile and install the PL/SQL packages.

5. **OPatch/**:
   - **Location**: `/u01/app/oracle/product/19.3.0/dbhome_1/OPatch/`
   - **Description**: Contains the OPatch utility for applying patches to the Oracle software.
   - **Important Files**:
     - `opatch` : OPatch executable file.

Datafile Locations

1. **CDBLAB Datafiles**:
   - **Location**: `/u01/app/oracle/oradata/CDBLAB/`
   - **Description**: Contains datafiles, control files, and redo log files for the `CDBLAB` .
   - **Important Files**:
     - `control01.ctl` , `control02.ctl` : Control files.
     - `redo01.log` , `redo02.log` , `redo03.log` : Redo log files.
     - `system01.dbf` : System datafile.
     - `sysaux01.dbf` : SYSAUX tablespace datafile.
     - `temp01.dbf` : Temporary tablespace datafile.
     - `undotbs01.dbf` : Undo tablespace datafile.

2. **PDB Datafiles**:
   - **Locations**:
     - `/u01/app/oracle/oradata/CDBLAB/PDB1/`
     - `/u01/app/oracle/oradata/CDBLAB/PDB2/`
     - `/u01/app/oracle/oradata/CDBLAB/PDB3/`
   - **Description**: Contains datafiles for each PDB.
   - **Important Files**:
     - `system01.dbf` : System datafile for each PDB.

- `sysaux01.dbf` : SYSAUX tablespace datafile for each PDB.
- `temp01.dbf` : Temporary tablespace datafile for each PDB.
- `users01.dbf` : Users tablespace datafile for each PDB.

Verification and Management Scripts

1. **Verify Database Status**:
   - **Command**: `SELECT con_id, name, open_mode FROM v$pdbs;`
   - **Description**: Checks the status of all PDBs within the CDB.
2. **Shutdown and Startup Commands**:
   - **Shutdown**:
     ```
     SHUTDOWN IMMEDIATE;
     ```
   - **Startup**:
     ```
     STARTUP;
     ```
3. **Managing PDBs**:
   - **Open a PDB**:
     ```
     ALTER PLUGGABLE DATABASE PDB1 OPEN;
     ```
   - **Close a PDB**:
     ```
     ALTER PLUGGABLE DATABASE PDB1 CLOSE IMMEDIATE;
     ```

Summary

This summary highlights the key directories and files necessary for managing the `CDBLAB` and its PDBs. Understanding the location and purpose of these files and directories is crucial for effective database administration and maintenance.

Addendum: Shutting Down and Restarting a Single PDB

Objective:

To provide detailed steps for shutting down and restarting a single Pluggable Database (PDB) within a Container Database (CDB).

Steps:

1. **Set Oracle Environment Variables**

   Set the Oracle environment variables using the `oraenv` script:

   ```
   . oraenv
   ORACLE_SID = [oracle] ? CDBLAB
   ```

2. **Connect to SQL*Plus as SYSDBA**

   Start SQL*Plus and log in to the database as the SYS user with the SYSDBA privilege:

```
sqlplus / as sysdba
```

3. **Check the Status of the PDBs**

   Before shutting down a PDB, check the current status of all PDBs:

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

   Expected Output:

   ```
   CON_ID    NAME       OPEN_MODE
   -------   -------    ----------
   2         PDB$SEED   READ ONLY
   3         PDB1       READ WRITE
   4         PDB2       READ WRITE
   5         PDB3       READ WRITE
   ```

4. **Switch to the PDB**

   Switch to the PDB you want to shut down. For example, to switch to `PDB1`:

   ```
   ALTER SESSION SET CONTAINER = PDB1;
   ```

5. **Shutdown the PDB**

   Shut down the PDB using the `CLOSE` command:

   ```
   ALTER PLUGGABLE DATABASE CLOSE IMMEDIATE;
   ```

   Verify the PDB Status:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDB1';
   ```

   Expected Output:

   ```
   NAME    OPEN_MODE
   ------  ----------
   PDB1    MOUNTED
   ```

   Explanation:

   - The PDB `PDB1` should now be in the MOUNTED state, indicating it is closed but not completely shut down.

6. **Restart the PDB**

   To restart the PDB, open it:

   ```
   ALTER PLUGGABLE DATABASE OPEN;
   ```

   Verify the PDB Status:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDB1';
   ```

   Expected Output:

```
NAME    OPEN_MODE
------  ----------
PDB1    READ WRITE
```

Explanation:

- The PDB `PDB1` should now be in the READ WRITE state, indicating it is fully operational.

7. **Return to the Root Container**

   After managing the PDB, return to the root container (`CDB$ROOT`):

```
ALTER SESSION SET CONTAINER = CDB$ROOT;
```

8. **Exit SQL*Plus**

   Exit SQL*Plus:

```
EXIT;
```

Summary

By following these steps, you can effectively shut down and restart a single PDB within a CDB. This procedure is essential for performing maintenance tasks or managing specific PDBs without affecting the entire CDB.