Lab 6-2: Managing PDBs Using SQL Developer

Objective:

To perform various management tasks on a Pluggable Database (PDB) using SQL Developer, including changing PDB modes, setting storage limits, renaming the PDB, modifying lockout time, unplugging, dropping, and plugging back in the PDB.

Prerequisites:

- SQL Developer installed and configured.
- Connection to the CDB (CDBLAB) and PDB (PDBLAB3) established.

Steps:

1. Changing the PDB Modes

Purpose:

Changing the mode of a PDB allows you to manage its accessibility and operational state. Modes include READ WRITE, READ ONLY, and MOUNTED.

Instructions:

1. Open SQL Developer and connect to PDBLAB3.

2. In the Connections pane, right-click on the connection for PDB3_CDBLAB and select "SQL Worksheet."

3. **Check Current Mode:**

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Verification: Ensure the output shows the current open mode of PDBLAB3.

4. **Close the PDB:**

   ```
   ALTER PLUGGABLE DATABASE PDBLAB3 CLOSE IMMEDIATE;
   ```

   Verification:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Ensure the output shows PDBLAB3 in the MOUNTED state.

5. **Open the PDB in READ ONLY Mode:**

   ```
   ALTER PLUGGABLE DATABASE PDBLAB3 OPEN READ ONLY;
   ```

   Verification:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Ensure the output shows PDBLAB3 in READ ONLY mode.

6. **Open the PDB in READ WRITE Mode (if required):**

```
ALTER PLUGGABLE DATABASE PDBLAB3 CLOSE IMMEDIATE;

ALTER PLUGGABLE DATABASE PDBLAB3 OPEN READ WRITE;
```

Verification:

```
SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
```

Ensure the output shows PDBLAB3 in READ WRITE mode.

2. Setting the PDB Storage Limit

Purpose:

Setting a storage limit helps in managing disk usage by restricting the maximum size a PDB can grow.

Instructions:

1. **Identify the `con_id` for PDBLAB3:**

   ```
   SELECT con_id FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Verification: Ensure the output shows the `con_id` for PDBLAB3.

2. **Identify the Tablespaces using the `con_id`:**

   ```
   SELECT tablespace_name FROM cdb_tablespaces WHERE con_id = <con_id>;
   ```

   Replace `<con_id>` with the actual `con_id` obtained from step 1.

   Verification: Ensure the output lists the tablespaces in PDBLAB3.

3. **Find the Existing Datafiles for Each Tablespace:**

   ```
   SELECT file_name FROM cdb_data_files WHERE tablespace_name = 'USERS' AND con_id = <con_id>;
   ```

   Replace `<con_id>` with the actual `con_id` obtained from step 1.

   Verification: Ensure the output shows the existing datafile names and paths for the USERS tablespace. Note: It will probably be this or close to it:

   ```
   /u01/app/oracle/oradata/CDBLAB/PDBLAB3/users01.dbf
   ```

4. **Set Storage Limit for the USERS Tablespace:** Add +1 to the filename identified above because you will need a new file.

   ```
   ALTER TABLESPACE users
   ADD DATAFILE '/u01/app/oracle/oradata/CDBLAB/PDBLAB3/users02.dbf'
   SIZE 500M AUTOEXTEND ON NEXT 500M MAXSIZE 2G;
   ```

   Replace `<existing_path>` with the actual path identified in step 3.

   Verification: (remember to replace the connection id with your actual connection id)

```
SELECT tablespace_name, bytes, maxbytes FROM cdb_data_files WHERE
tablespace_name = 'USERS' AND con_id = <con_id>;
```

Ensure the output shows the new datafile with the correct size and MAXSIZE set to 2G. You should see Max Bytes at 2G (2147483648) and BYTES at 500M (524288000)

These steps will help you properly set the storage limit for the PDB tablespaces.

3. Changing the Global Name of the PDB

Purpose:

Changing the global name of a PDB can help in identifying and managing the PDB within a larger environment.

Instructions:

The ORA-01109 error indicates that the database is not open. To rename the global name of the PDB, you need to ensure the PDB is in the correct state. Let's correct the steps:

Summary of Steps to Rename the Global Name of a PDB

1. **Close the PDB:**

   ```
   ALTER PLUGGABLE DATABASE PDBLAB3 CLOSE IMMEDIATE;
   ```

   Verification:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Ensure the output shows PDBLAB3 in the MOUNTED state.

2. **Open the PDB in RESTRICTED Mode:**

   ```
   ALTER PLUGGABLE DATABASE PDBLAB3 OPEN RESTRICTED;
   ```

   Verification:

   ```
   SELECT name, open_mode FROM v$pdbs WHERE name = 'PDBLAB3';
   ```

   Ensure the output shows PDBLAB3 in READ WRITE mode.

3. **Switch to the PDB Context:**

   ```
   ALTER SESSION SET CONTAINER = PDBLAB3;
   ```

4. **Rename the PDB:**

   ```
   ALTER DATABASE RENAME GLOBAL_NAME TO PDB_LAB3_CDBLAB;
   ```

   Verification:

   ```
   SELECT name FROM v$pdbs WHERE name = 'PDB_LAB3_CDBLAB';
   ```

   Ensure the output shows the new name PDB_LAB3_CDBLAB.

5. **Ensure that the PDB is in READ WRITE Mode:**

Verification:

```
SELECT name, open_mode FROM v$pdbs WHERE name = 'PDB_LAB3_CDBLAB';
```

Ensure the output shows PDB_LAB3_CDBLAB in the READ WRITE mode. Switch back to the CDBRoot:

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

By following these corrected steps, you should be able to rename the global name of the PDB without encountering the ORA-01109 error.

4. Changing the PDB Lockout Time

Purpose:

Changing the PDB lockout time adjusts the duration for which a PDB is locked out after a certain number of failed login attempts.

Instructions:

1. In the SQL Worksheet, enter the following command:

```
ALTER PROFILE DEFAULT LIMIT FAILED_LOGIN_ATTEMPTS 5 PASSWORD_LOCK_TIME 1;
```

2. Run the command.

Verification:

To verify the changes, execute:

```
SELECT resource_name, limit FROM dba_profiles WHERE profile = 'DEFAULT' AND
resource_name IN ('FAILED_LOGIN_ATTEMPTS', 'PASSWORD_LOCK_TIME');
```

Ensure the output shows the updated limits.

5. Unplugging the PDB

Purpose:

Unplugging a PDB prepares it for being moved or dropped without removing its data files.

Instructions:

1. In the SQL Worksheet, enter the following command to close and unplug the PDB:

```
ALTER PLUGGABLE DATABASE PDB_LAB3_CDBLAB CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE PDB_LAB3_CDBLAB UNPLUG INTO
'/u01/app/oracle/oradata/CDBLAB/PDB_LAB3_CDBLAB.xml';
```

2. Run the commands.

Verification:

Ensure the PDB is in the unplugged state by executing:

```
SELECT name, open_mode FROM v$pdbs WHERE name = 'PDB_LAB3_CDBLAB';
```

6. Dropping the PDB

Purpose:

Dropping a PDB removes it from the CDB but retains its data files, allowing it to be plugged back in later.

Instructions:

1. In the SQL Worksheet, enter the following command:

   ```
   DROP PLUGGABLE DATABASE PDB_LAB3_CDBLAB KEEP DATAFILES;
   ```

2. Run the command.

Verification:

To verify the PDB is dropped, execute:

```
SELECT name FROM v$pdbs;
```

Ensure PDB_LAB3_CDBLAB is not listed.

7. Plugging Back in the PDB

Purpose:

Plugging back in a PDB allows it to be reattached to the CDB, making it operational again.

Instructions:

1. In the SQL Worksheet, enter the following command to plug the PDB back in:

   ```
   CREATE PLUGGABLE DATABASE PDB_LAB3_CDBLAB USING
   '/u01/app/oracle/oradata/CDBLAB/PDB_LAB3_CDBLAB.xml' COPY FILE_NAME_CONVERT =
   ('/u01/app/oracle/oradata/CDBLAB/',
   '/u01/app/oracle/oradata/CDBLAB/PDB_LAB3_CDBLAB/');
   ```

2. Run the command.

Verification:

To verify the PDB is plugged back in, execute:

```
SELECT name, open_mode FROM v$pdbs WHERE name = 'PDB_LAB3_CDBLAB';
```

Ensure the output shows PDB_LAB3_CDBLAB in MOUNT mode.

Challenge:

Please rename PDB_LAB3_CDBLAB back to PDB_LAB3 and put that PDB in READ_WRITE mode

Summary:

By following these steps, you will have successfully managed various aspects of a PDB using SQL Developer, including changing modes, setting storage limits, renaming, adjusting lockout time, unplugging, dropping, and plugging back in the PDB. Each step includes verification to ensure the changes are correctly applied.