

Database development with PL/SQL INSY 8311



ADVENTIST UNIVERSITY
OF CENTRAL AFRICA

Instructor:

- Eric Maniraguha | eric.maniraguha@auca.ac.rw | [LinkedIn Profile](#)



6h00 pm – 9h50 pm

- Monday A –G209
- Tuesday B-G207
- Wednesday C-G207
- Thursday D-G310

September 2025

Database development with PL/SQL

Reference reading

- [4 PL/SQL Control Statements](#)
- [Oracle PL/SQL Insert, Update, Delete & Select Into \[Example\]](#)
- [YouTube Tutorial: Oracle CDBs and PDBs Explained \(With Connections\)](#)
- [Oracle 12c: Applying PSU 12.1.0.1.1 with Multitenant DB & unplug/plug](#)
- [Multitenant : Overview of Container Databases \(CDB\) and Pluggable Databases \(PDB\)](#)
- [1 Introduction to the Multitenant Architecture](#)
- [Getting started with Oracle Database 12c Multitenant Architecture](#)
- [How to Install Oracle on an M1/M2 Mac \(Finally\)](#)
- [How to install Oracle Database on Mac OS Sierra 10.12 or above](#)

Course Objectives: CDB, PDB, and Oracle Enterprise Manager

Understand Container Database (CDB) and Pluggable Database (PDB) architecture and their role in Oracle multi-tenancy.

Create, configure, and manage CDBs and PDBs using SQL*Plus and Oracle tools.

Perform SQL operations within PDBs, including creating schema objects.

Administer databases using Oracle Enterprise Manager (OEM) for monitoring and management.

Monitor and optimize database performance using OEM tools.

Automate routine tasks and schedule jobs via OEM.

Manage users and security in CDB and PDB environments.

Execute backup and recovery operations at both CDB and PDB levels.

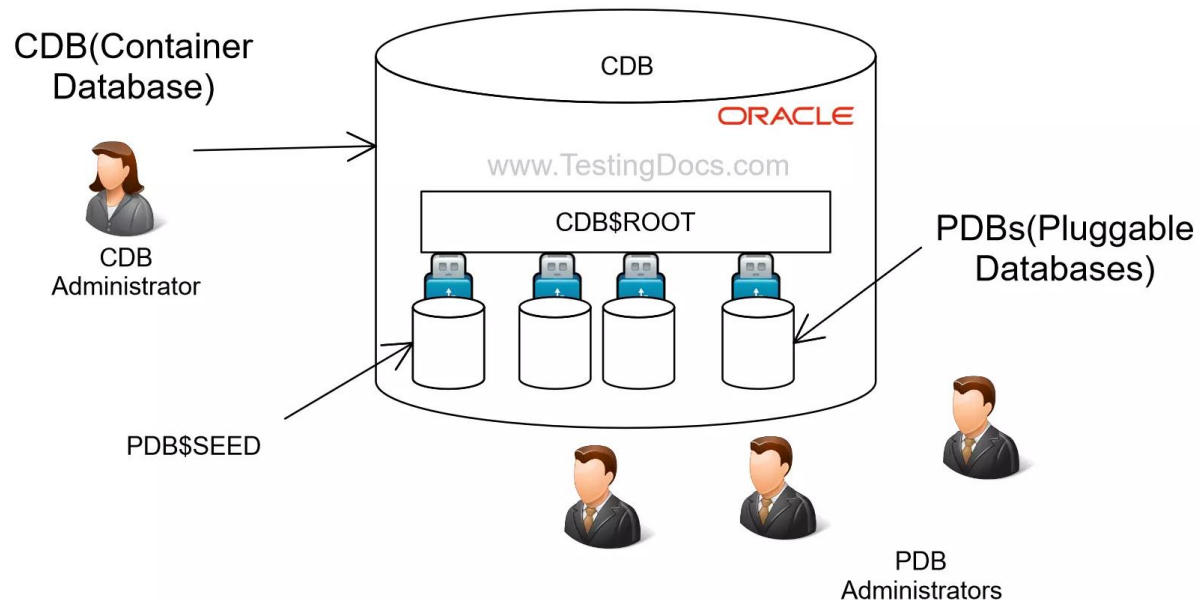
Troubleshoot common database issues using OEM diagnostic tools.

Oracle Databases CDBs and PDBs

CDB – Container Database

- A **CDB** is the main Oracle Database instance under multitenant architecture.
- It contains:
 - **Root Container (CDB\$ROOT)**: Holds Oracle-supplied metadata, system schemas, and common users.
 - **Seed PDB (PDB\$SEED)**: A template used for quickly creating new pluggable databases.
 - **User-created PDBs**: Actual application databases that store user data and application-specific schemas.

Think of the CDB as the **big container** that manages resources and metadata for all PDBs.



PDB – Pluggable Database

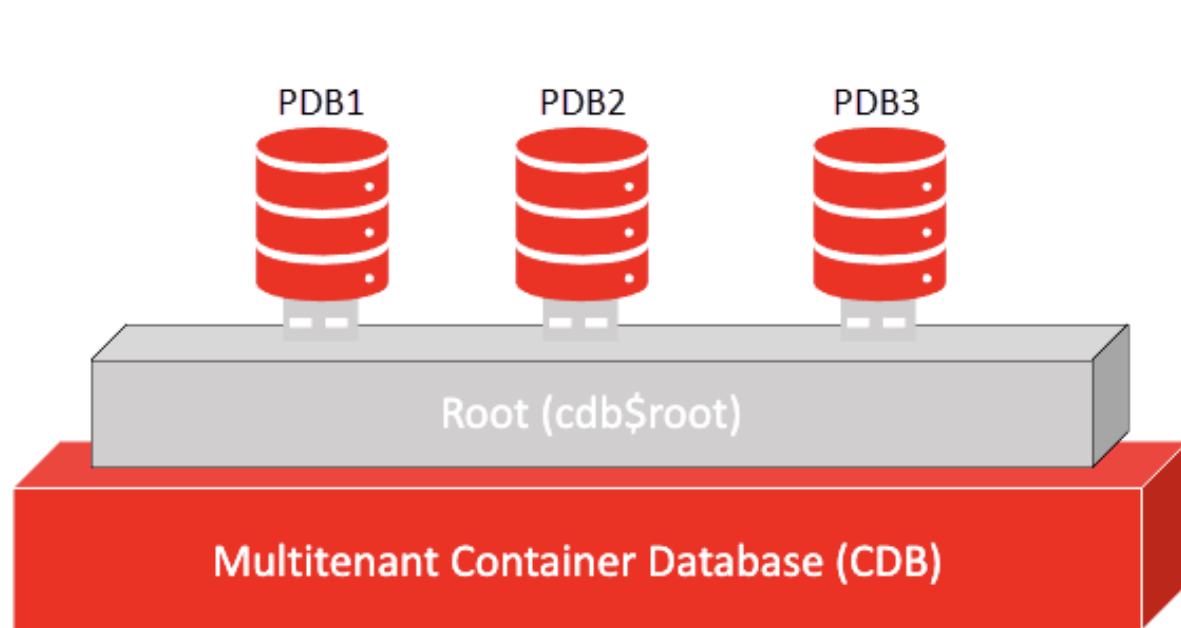
- A **PDB** is a **portable, self-contained collection of schemas and objects**.
- It contains **user data and code** (tables, indexes, PL/SQL packages, etc.).
- Multiple PDBs can run inside a single CDB.
- You can plug/unplug PDBs between CDBs — like moving a database between servers.

Think of a PDB as a **tenant's database** inside an apartment building (the CDB).

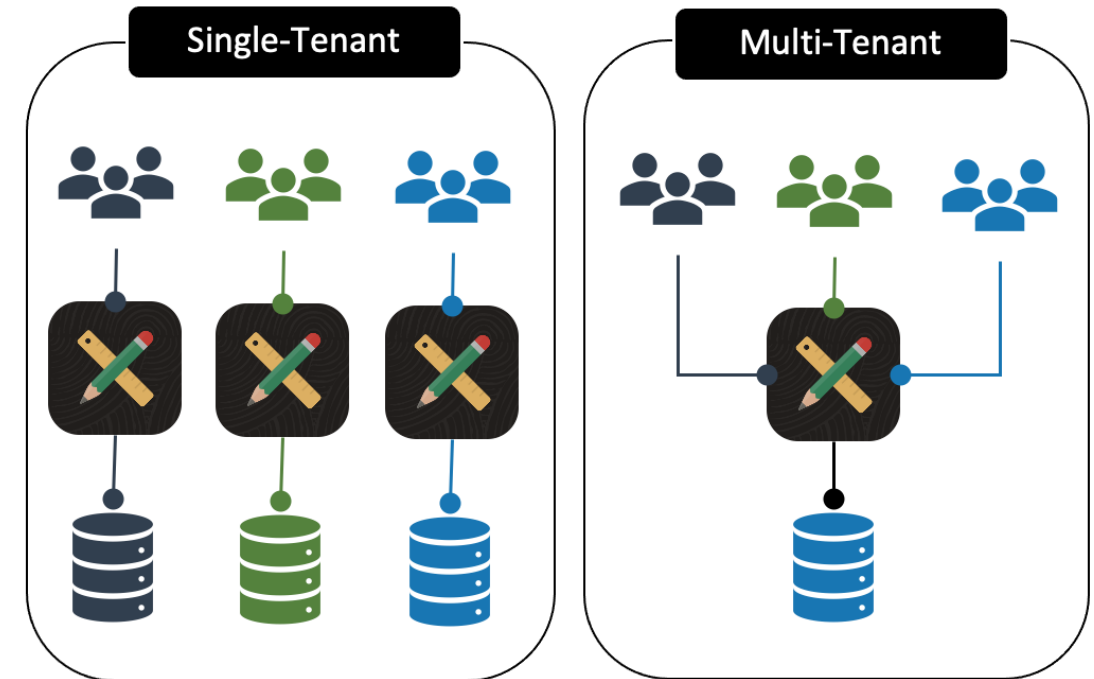
Multitenancy in Oracle refers to the architecture introduced in Oracle 12c that allows multiple, separate databases to reside in a single **Container Database (CDB)**, each represented as a **Pluggable Database (PDB)**. This architecture is aimed at improving resource utilization, scalability, and ease of management.

Multitenant Container Database Architecture

Oracle's **Multitenant Architecture** enables efficient management of multiple databases through a single **Container Database (CDB)**.



Source Image: <http://medium.com/@davidadjirackor/oracle-single-tenant-vs-multi-tenant-architecture-explained-f83f2fc6e2b1>



Source Image: <https://blog.cloudnueva.com/multi-tenant-apex-apps>

Oracle Single-Tenant (Non-CDB) vs Oracle Multi-tenant (CDB & PDB)



This is the traditional architecture where a single Oracle Database contains all components (data dictionary, user data, and system metadata) in one structure.

Features:

- Each database instance is independent.
- Requires separate system resources (memory, processes).
- Administrative overhead increases with multiple databases.
- Upgrades, patching, and backups must be done individually.
- Every database has its own set of background processes and system metadata.

Pros:

- ✓ **Simplicity** – Each database is managed separately.
- ✓ **No additional licensing required.**
- ✓ **Works well** for smaller applications with low complexity.

Cons:

- ✗ **Resource-intensive** – Each database consumes separate CPU, memory, and disk resources.
- ✗ **Difficult to manage at scale** – Administering multiple databases is time-consuming.
- ✗ **Higher maintenance costs** due to separate backup, security, and upgrade efforts.

Oracle Multitenant allows multiple **Pluggable Databases (PDBs)** to share a single **Container Database (CDB)**. This enables database consolidation while maintaining separation between individual PDBs.

Features:

- A single CDB can host multiple PDBs.
- Shared memory and background processes, improving efficiency.
- PDBs can be unplugged and moved between CDBs.
- Centralized management with common users, roles, and privileges.
- Patch, upgrade, and backup at the **CDB level**, reducing operational overhead.

Pros:

- ✓ **Resource efficiency** – PDBs share CPU, memory, and processes, optimizing resource usage.
- ✓ **Easier management** – One container database to manage multiple pluggable databases.
- ✓ **Fast provisioning** – Quickly create, clone, or unplug PDBs.
- ✓ **Better security & isolation** – Each PDB remains logically separate while benefiting from shared infrastructure.
- ✓ **Simplifies upgrades & patches** – Apply changes to the CDB, affecting all PDBs at once.

Cons:

- ✗ **Requires Oracle Multitenant license** (except for one PDB in SE2/EE).
- ✗ **More complex setup** than single-tenant.
- ✗ **CDB failure impacts all PDBs** unless configured for high availability.

Types of Patches in Oracle

1. Security Patches

- Fix vulnerabilities that hackers could exploit.
- Released quarterly by Oracle as **Critical Patch Updates (CPUs)**.

2. Bug Fix Patches

- Correct known issues in the database software.

3. Patch Set Updates (PSUs) / Release Updates (RUs)

- Bundled patches that combine bug fixes and security updates.

4. One-Off Patches

- Specific fixes for a customer issue, not part of the general release.

Imagine you have **50 databases**.

- In **Non-CDB mode**, you patch 50 times.
- In **CDB/PDB mode**, you patch **once** at the container level, and all PDBs benefit.



When to Choose Each? | Key Differences

Choose Single-Tenant (Non-CDB) if:

- You have a small number of databases.
- You prefer simpler database management.
- You don't need Oracle Multitenant features.

Choose Multitenant (CDB/PDB) if:

- You need to consolidate many databases efficiently.
- You want easier patching, cloning, and management.
- You plan to scale up database instances.
- You are using Oracle Cloud or a large enterprise system.

Feature	Single-Tenant (Non-CDB)	Multitenant (CDB/PDB)
Architecture	One database with its own resources	One CDB with multiple PDBs sharing resources
Resource Usage	Higher (each DB uses its own resources)	Lower (shared memory, processes, storage)
Management	Each database is managed separately	Centralized management for all PDBs
Scalability	Limited	High (easily add/remove PDBs)
Backup & Recovery	Per database	Can be done at CDB level for all PDBs
License Requirement	No additional license required	Requires Multitenant option (for more than one PDB)
Security Isolation	Fully separate instances	Logical separation between PDBs within a CDB
Upgrade/Patch	Done per database	Done once for all PDBs within a CDB

Multitenant, Structure of CDBs and their PDBs

1. Oracle Instance (SGA/Processes):

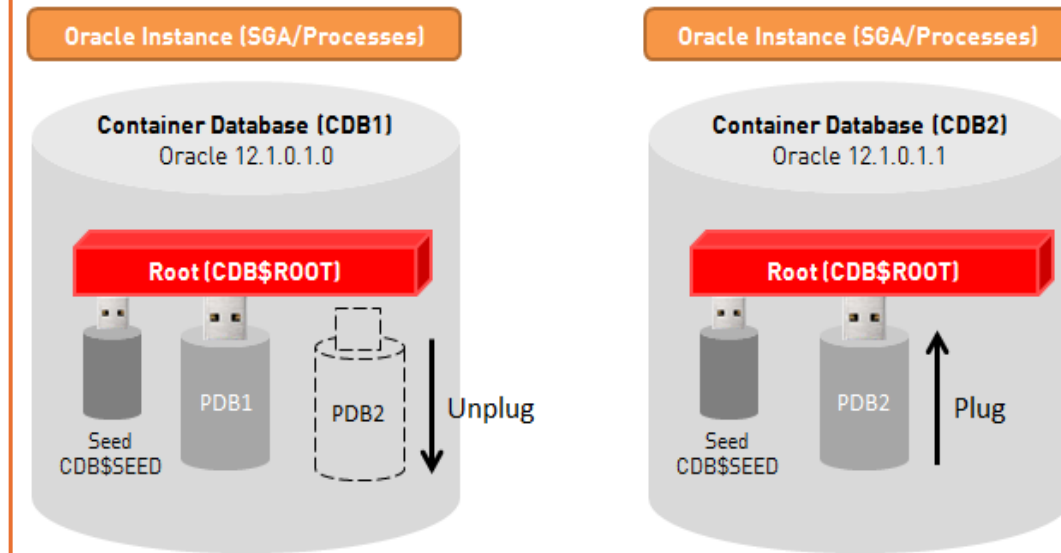
- This refers to the Oracle instance, which includes the System Global Area (SGA) and background processes that manage database operations.

2. Container Database (CDB1):

- Oracle 12.1.0.1.0:** This indicates the version of the Oracle database.
- Root (CDB\$ROOT):** The root container, which is the main container in a CDB. It contains the system metadata and common users.
- Seed (CDB\$SEED):** A template PDB used to create new PDBs.
- PDB1 and PDB2:** These are Pluggable Databases within the CDB. PDBs are separate databases that can be managed independently but share the same Oracle instance.
- Unplug:** This suggests that a PDB can be unplugged from the CDB, meaning it can be detached and moved to another CDB.

3. Container Database (CDB2):

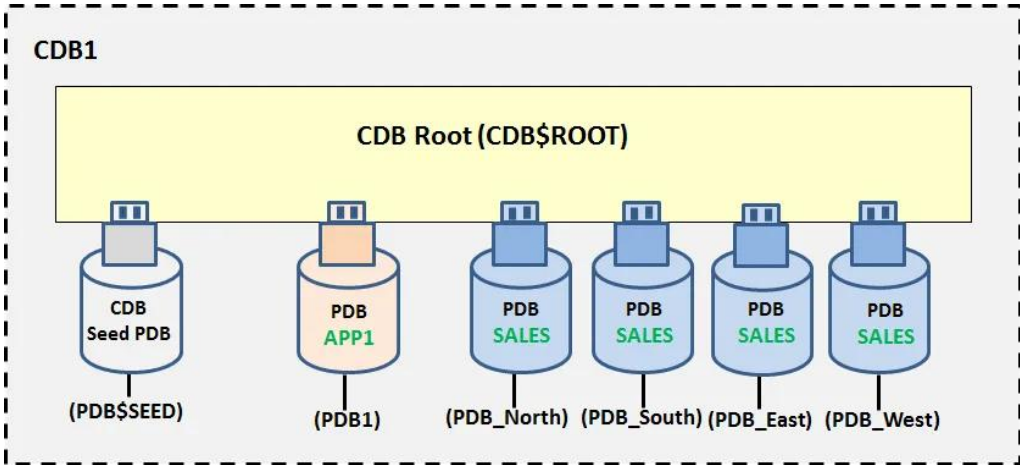
- Oracle 12.1.0.1.1:** A different version of the Oracle database.
- Root (CDB\$ROOT):** Similar to CDB1, this is the root container for CDB2.
- Seed (CDB\$SEED):** The template PDB for CDB2.
- PDB2:** A Pluggable Database within CDB2.
- Plug:** This indicates that a PDB can be plugged into the CDB, meaning it can be attached and used within this CDB.



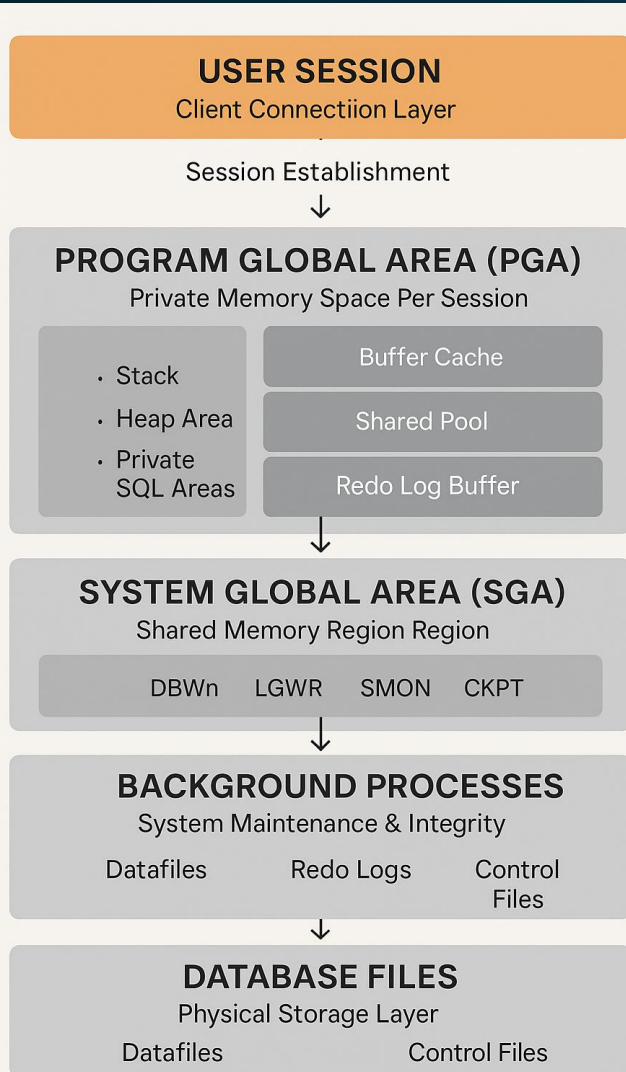
Source Image: <https://www.dbi-services.com/blog/oracle-12c-applying-psu-121011-with-multitenant-db-and-unplugplug/>

Key Differences: CDB, CDB\$ROOT, PDB and PDB\$SEED

Feature	CDB (Container Database)	CDB\$ROOT (Root of CDB)	PDB (Pluggable Database)	PDB\$SEED (PDB Seed)
Type	Top-level container for multiple PDBs	Root container for system-level data	A pluggable, fully functional database	Template for creating new PDBs
Purpose	Houses multiple PDBs and system metadata	Stores common system data and schemas	Holds user data, schemas, and apps	Used to create new PDBs
User Data	No user data	No user data	Stores user data	No user data
Modifiable	Read-write (but for managing PDBs)	Read-write (system-level management only)	Read-write	Read-only (cannot be modified)
Creation of New PDBs	Manages and contains PDBs	Contains only system schemas and data	Created from CDB\$SEED or other PDBs	Used to clone and create new PDBs
Contained Objects	Contains the entire multitenant structure	Contains system data for all PDBs	Contains user-defined schemas, tables, etc.	Contains templates for new PDBs



Oracle Instance Components – refreshment



- **Purpose:** Client connection & authentication.
- **Function:** Establishes communication between client apps and Oracle DB.
- **Scope:** Each user connection.

User Session

- **Purpose:** Private memory for each user session.
- **Components:**
 - Stack Space
 - Heap Area
 - Private SQL Areas
- **Characteristics:** Non-shared, process-specific.

Program Global Area (PGA)

- **Purpose:** Shared memory accessible by all processes.
- **Key Components:**
 - **Buffer Cache** → Stores data blocks read from datafiles.
 - **Shared Pool** → SQL statements & data dictionary cache.
 - **Redo Log Buffer** → Holds redo entries for recovery.
- **Characteristics:** Shared across the instance.

System Global Area (SGA)

- **Purpose:** System maintenance & data integrity.
- **Key Processes:**
 - **DBWn** → Writes modified buffers to datafiles.
 - **LGWR** → Writes redo log entries to redo log files.
 - **SMON** → Crash recovery & cleanup.
 - **PMON** → Cleans up failed sessions.
 - **CKPT** → Updates control files & datafile headers.

Background Processes

- **Purpose:** Physical storage for persistent data.
- **Types:**
 - **Datafiles** → Store actual database objects (tables, indexes).
 - **Redo Log Files** → Store redo entries for recovery.
 - **Control Files** → Store metadata & structure info.

Database Files

Display all PDBS

```
Command Prompt - sqlplus s X + v
Microsoft Windows [Version 10.0.22631.4037]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Eric>sqlplus sys as SYSDBA

SQL*Plus: Release 21.0.0.0.0 - Production on Sun Sep 22 14:36:16 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> show user;
USER is "SYS"
SQL>
SQL> SELECT instance_name FROM v$instance;

INSTANCE_NAME
-----
oracle21cdb

SQL> show pdbs;

  CON_ID  CON_NAME          OPEN MODE  RESTRICTED
-----
2 PDB$SEED              READ ONLY  NO
3 ORACLE21DBCPDB        READ WRITE NO

SQL> |
```

- To connect to an Oracle database using SQL*Plus as the SYS user with SYSDBA privileges.

sqlplus sys as SYSDBA
Or
sqlplus sys/admin@localhost:1521/ORACLE21CDB AS SYSDBA

- Oracle instance that is currently running, which is usually the same as the **SID** (System Identifier) of the database.

SQL> SHOW PDBS;

- This PDB helps streamline the process of managing multiple PDBs within a container database (CDB).

- This is the PDB created during the installation of Oracle 21c; for others, you might have XE.

Show path of PDBs

```
SQL> show pdbs;
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
3	ORACLE21DBCPDB	READ WRITE	NO

```
SQL> SELECT CON_ID, TABLESPACE_NAME, FILE_NAME
2 FROM CDB_DATA_FILES
3 WHERE CON_ID = 3;
```

CON_ID	TABLESPACE_NAME	FILE_NAME
3	SYSTEM	E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\SYSTEM01.DBF

3	SYSaux	E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\SYSaux01.DBF
---	--------	--

3	UNDOTBS1	E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\UNDOTBS01.DBF
---	----------	---

CON_ID	TABLESPACE_NAME	FILE_NAME
3	USERS	E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\USERS01.DBF

```
SQL> |
```

- This query retrieves the tablespace names and file names of data files for the Pluggable Database (PDB) with CON_ID = 3 from the CDB_DATA_FILES view in an Oracle multitenant environment. - **Location of XE PDB1**

```
SQL> SELECT CON_ID, TABLESPACE_NAME, FILE_NAME FROM
CDB_DATA_FILES WHERE CON_ID = 3;
```

- This path is commonly used for organizing database files, making it easier to manage multiple PDBs within a CDB.

Create PDB using PDBSEED

```

C:\> Command Prompt - sqlplus s X + v

SQL*Plus: Release 21.0.0.0.0 - Production on Sun Sep 22 17:30:49 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> CREATE PLUGGABLE DATABASE PDB2
  2 ADMIN USER pdbadmin IDENTIFIED BY admin
  3 FILE_NAME_CONVERT = ('E:\oracle21cdata\ORACLE21CDB\pdbseed\',
  4 'E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\PDB2\');

Pluggable database created.

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

PDB_NAME
-----
STATUS
-----
ORACLE21DBCPDB
NORMAL

PDB$SEED
NORMAL

PDB2
NEW
  
```

- This creates a new pluggable database by copying the structure of pdbseed and setting up a new admin user.

```

SQL> CREATE PLUGGABLE DATABASE PDB2
ADMIN USER pdbadmin IDENTIFIED BY admin
FILE_NAME_CONVERT = ('E:\oracle21cdata\ORACLE21CDB\pdbseed\',
'E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\PDB2\');
  
```

- It retrieves the name and status of all Pluggable Databases (PDBs) from the CDB_PDBS view in an Oracle multitenant environment.
- **PDB2 has status new because it has not yet opened**

```

SQL> SELECT pdb_name, status from cdb_pdbs
  
```


PDB Status

```
SQL> select name, open_mode from v$pdb;
```

```
NAME
```

```
OPEN_MODE
```

```
PDB$SEED  
READ ONLY
```

```
ORACLE21DBCPDB  
READ WRITE
```

```
PDB2  
MOUNTED
```

```
SQL> SELECT NAME, CON_ID FROM v$active_services order by 1;
```

NAME	CON_ID
SYS\$BACKGROUND	1
SYS\$USERS	1
oracle21cdb	1
oracle21cdbXDB	1
oracle21dbcpdb	3
pdb2	4

```
6 rows selected.
```

```
SQL> |
```

- In short, this query gives you the status of all PDBs in the CDB.
- The pdb created is mounted = not yet open for use.

```
SQL> select name, open_mode from v$pdb;
```

- The command lists active services and their container IDs (CDB or PDB), sorted by service name.

```
SQL> SELECT NAME, CON_ID FROM v$active_services ORDER BY 1;
```


Display Current Container – CDB\$ROOT

```

Command Prompt - sqlplus s  X + v
SQL> ALTER PLUGGABLE DATABASE PDB2 OPEN;

Pluggable database altered.

SQL> SHOW CON_NAME;

CON_NAME
-----
CDB$ROOT
SQL>
SQL>
SQL> SELECT name, open_mode from v$pdb;

NAME
-----
OPEN_MODE
-----
PDB$SEED
READ ONLY

ORACLE21DBCPDB
READ WRITE

PDB2
READ WRITE

```

- This command will return the name of the current container you are connected to, which could be the root container (CDB\$ROOT) or a specific pluggable database (PDB).

SQL> Show con_name;

or

SQL> SELECT SYS_CONTEXT('USERENV', 'CON_NAME') FROM DUAL;

- Retrieves the names and open modes (like READ WRITE, READ ONLY) of all pluggable databases (PDBs) in the Oracle container.

SQL> SELECT name, open_mode from v\$pdb;

Alter PDB2 – Open & Save

```
SQL> ALTER PLUGGABLE DATABASE PDB2 OPEN;  
ALTER PLUGGABLE DATABASE PDB2 OPEN  
*  
ERROR at line 1:  
ORA-65019: pluggable database PDB2 already open
```

```
SQL> -- Save the state so PDB2 opens automatically on CDB startup  
SQL> ALTER PLUGGABLE DATABASE PDB2 SAVE STATE;  
  
Pluggable database altered.
```

```
SQL> |
```

- After creating the PDB, it will be in a mounted state. The command opens the **PDB2** pluggable database.

```
SQL> ALTER PLUGGABLE DATABASE PDB2 OPEN;
```

- The command saves PDB pdb2's state for automatic opening in the same mode (READ ONLY or READ WRITE) on the next CDB startup.

```
SQL> ALTER PLUGGABLE DATABASE pdb2 SAVE STATE;
```

Oracle Instance

```
SQL> SELECT instance_name FROM v$instance;

INSTANCE_NAME
-----
oracle21cdb
```

- This Query retrieves the name of the current Oracle database instance you are connected to.

```
SQL> SELECT INSTANCE_NAME FROM V$INSTANCE
```

```
Microsoft Windows [Version 10.0.22631.4037]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Eric>sqlplus sys@ORACLE21CDB as sysdba

SQL*Plus: Release 21.0.0.0.0 - Production on Sun Sep 22 13:29:50 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> |
```

- The command sqlplus sys@ORACLE21CDB as sysdba; connects you to the Oracle database ORACLE21CDB using the SYS user with SYSDBA privileges. This allows you to perform administrative tasks on the container database (CDB) or any pluggable database (PDB) within it.

```
SQL> Sqlplus sys@ORACLE21CDB as sysdba;
```

Switch to the PDB & Create User & Grant Privileges

```

Command Prompt - sqlplus s
SQL> SELECT instance_name FROM v$instance;

INSTANCE_NAME
-----
oracle21cdb

SQL> ALTER SESSION SET CONTAINER = PDB2;

Session altered.

SQL> CREATE USER plsqlauca IDENTIFIED BY password;

User created.

SQL> GRANT ALL PRIVILEGES TO plsqlauca;

Grant succeeded.

SQL> |
  
```

- The command switches your current session to the pluggable database (PDB) named PDB2. This allows you to execute commands and queries specifically within that PDB context.

```
SQL> ALTER SESSION SET CONTAINER = PDB2;
```



- The command creates a new user named plsqlauca in the Oracle database, with the specified password. After executing this command, you may need to grant the necessary privileges to the user to allow them to perform actions within the database.

```
SQL> CREATE USER plsqlauca IDENTIFIED BY password;
```

- The command GRANT ALL PRIVILEGES TO plsqlauca; grants all available privileges to the user plsqlauca in the Oracle database. This allows the user to perform any action on the database, including creating, modifying, and deleting objects.

```
SQL> GRANT ALL PRIVILEGES TO plsqlauca
```

Connecting to Oracle PDB through Oracle SQL Developer for the Newly Created User

New / Select Database Connection

Connection Name	Connection Details
AUCA@2024	sys@//localhost:...
employeeManage...	sys@//localhost:...
orade21c_datab...	sys@//localhost:...
oradeclass	oradeclass@//loc...
oradepdb	sys@//localhost:...

Name

Database Type

User Info Proxy User

Authentication Type

Username

Password

Role

☐ Save Password

Connection Type

Details Advanced

Hostname

Port

☐ SID

☒ Service name

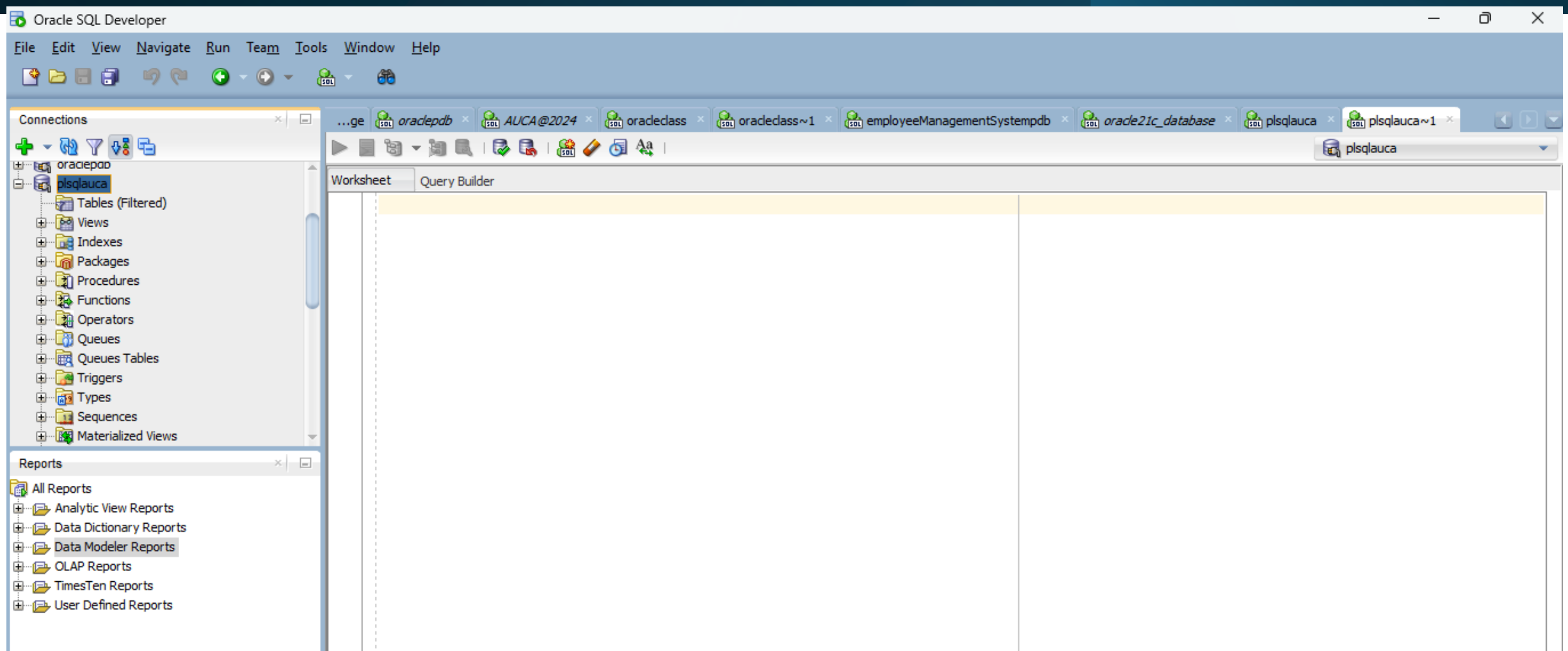
Status : Success

Help Save Clear Test Connect Cancel

Oracle PL/SQL: User Creation and Privilege Grants

Command	Description
CREATE USER XXXXXX IDENTIFIED BY password;	Creates a new user with the given username (XXXXXX) and password.
GRANT ALL PRIVILEGES TO XXXXX	Granting all privileges to a user provides them with full access to the database.
GRANT CONNECT TO XXXXXX; GRANT CREATE SESSION TO XXXXXX;	Grants basic privileges for connecting to the database and creating a session.
GRANT DBA TO XXXXXX; GRANT RESOURCE TO XXXXXX; GRANT UNLIMITED TABLESPACE TO XXXXXX;	Grants DBA privileges, resource management, and unlimited tablespace usage.
GRANT CREATE VIEW TO XXXXXX; GRANT CREATE PROCEDURE TO XXXXXX; GRANT CREATE SEQUENCE TO XXXXXX; GRANT CREATE TRIGGER TO XXXXXX;	Allows the user to create views, procedures, sequences, and triggers.
GRANT ALTER ANY TABLE TO XXXXXX; GRANT ALTER ANY PROCEDURE TO XXXXXX; GRANT ALTER ANY TRIGGER TO XXXXXX;	Grants the ability to alter tables, procedures, and triggers.
GRANT DELETE ANY TABLE TO XXXXXX;	Allows the user to delete rows from any table in the database.
GRANT DROP ANY PROCEDURE TO XXXXXX; GRANT DROP ANY TRIGGER TO XXXXXX; GRANT DROP ANY VIEW TO XXXXXX;	Enables the user to drop (remove) procedures, triggers, and views.

Open worksheet for user: plsqlauca



Steps to Permanently Delete Pluggable Databases in Oracle

- **Close the PDBs:** Ensure the pluggable databases are not in use.
- **Unmount the PDBs:** Detach the PDBs from the container database.
- **Use the DROP PLUGGABLE DATABASE Command:** Execute with the INCLUDING DATAFILES option to remove both the PDBs and their associated data files.
- **Permanent Deletion:** This process permanently deletes the databases.



Step 1: Delete the pluggable databases HR and PDB4, follow these steps

```
Command Prompt - sqlplus s X + v
C:\Users\Eric>sqlplus sys/admin@localhost:1521/ORACLE21CDB AS SYSDBA

SQL*Plus: Release 21.0.0.0.0 - Production on Wed Sep 25 15:51:22 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> col name for a20;
SQL> /
SP2-0103: Nothing in SQL buffer to run.
SQL>
SQL> select name, open_mode from v$containers
2 /

NAME                                OPEN_MODE
-----
CDB$ROOT                            READ WRITE
PDB$SEED                            READ ONLY
ORACLE21DBCPDB                      READ WRITE
HR                                  READ WRITE
PDB4                                READ WRITE

SQL> |
```

- To connect to an Oracle database using SQL*Plus as the SYS user with SYSDBA privileges.

C:\Users\Eric>sqlplus sys as SYSDBA

or

C:\Users\Eric> sqlplus sys/admin@localhost:1521/ORACLE21CDB AS SYSDBA

- The query retrieves the names and open modes of all pluggable databases in the current container database.

SQL> col name for a20; -- This is for formatting our output.

SQL> select name, open_mode from v\$containers

Step 2: Close and check the paths of Unplug DB

```
SQL> ALTER PLUGGABLE DATABASE HR CLOSE IMMEDIATE;

Pluggable database altered.

SQL> ALTER PLUGGABLE DATABASE PDB4 CLOSE IMMEDIATE;

Pluggable database altered.

SQL> |
```

- Before dropping a PDB, it needs to be closed.

```
SQL> ALTER PLUGGABLE DATABASE HR CLOSE IMMEDIATE;
SQL> ALTER PLUGGABLE DATABASE PDB4 CLOSE IMMEDIATE;
```

```
SQL> SELECT directory_name, directory_path FROM dba_directories;

DIRECTORY_NAME
-----
DIRECTORY_PATH
-----
DATA_PUMP_DIR
C:\oracle21c_config\admin\oracle21cdb\dpdump/

DBMS_OPTIM_LOGDIR
C:\oracle21c\cfgtoollogs

DBMS_OPTIM_ADMINDIR
C:\oracle21c\rdbms\admin
```

- Find or set the correct directory path for unplugging the PDBs.
- ```
SQL> SELECT directory_name, directory_path FROM dba_directories;
```

## Step 3: Use the Path in the UNPLUG Command

```
SQL> ALTER PLUGGABLE DATABASE HR UNPLUG INTO 'C:\oracle21c_config\admin\oracle21cdb\dpdump\hr.xml';
```

```
Pluggable database altered.
```

```
SQL> ALTER PLUGGABLE DATABASE PDB4 UNPLUG INTO 'C:\oracle21c_config\admin\oracle21cdb\dpdump\pdb4.xml';
```

```
Pluggable database altered.
```

```
SQL>
```

- Before dropping a PDB, it needs to be closed.

```
SQL> ALTER PLUGGABLE DATABASE HR UNPLUG INTO 'C:\oracle21c_config\admin\oracle21cdb\dpdump\hr.xml';
```

```
SQL> ALTER PLUGGABLE DATABASE PDB4 UNPLUG INTO 'C:\oracle21c_config\admin\oracle21cdb\dpdump\pdb4.xml';
```

# Step 4: Drop Pluggable Databases 'HR' and 'PDB4' and Verify Their Deletion"

```
SQL> DROP PLUGGABLE DATABASE HR INCLUDING DATAFILES;
```

Pluggable database dropped.

```
SQL> DROP PLUGGABLE DATABASE PDB4 INCLUDING DATAFILES;
```

Pluggable database dropped.

```
SQL>
SQL>
SQL>
SQL> SELECT name, open_mode FROM v$containers;
```

| NAME           | OPEN_MODE  |
|----------------|------------|
| CDB\$ROOT      | READ WRITE |
| PDB\$SEED      | READ ONLY  |
| ORACLE21DBCPDB | READ WRITE |

- Before dropping a PDB, it needs to be closed.

```
SQL> DROP PLUGGABLE DATABASE HR INCLUDING DATAFILES;
SQL> DROP PLUGGABLE DATABASE PDB4 INCLUDING DATAFILES;
```

- The query shows PDB (All pluggable database) names and their open modes.

```
SQL> SELECT name, open_mode FROM v$containers;
```

## Notes:

- **CDB\$ROOT**: The root container of the multitenant database. It cannot be deleted.
- **PDB\$SEED**: A template PDB used for creating new PDBs. It cannot be deleted.
- **ORACLE21DBCPDB**: A user-created PDB that can be deleted, but ensure it's not in use and back up important data before proceeding.

# Steps for Cloning a Pluggable Database (PDB)

In Oracle, **cloning** refers to creating an exact copy of a database or its components. This process is commonly used for tasks such as backup, testing, or creating development environments without affecting the production database. Cloning helps maintain data consistency while allowing operations on the copied database.

There are several types of cloning in Oracle:

1. **Database Cloning:** Creating a duplicate of an entire Oracle database. This is useful for testing or performing operations without affecting the original database.
2. **PDB (Pluggable Database) Cloning:** In Oracle Multitenant Architecture, you can clone a PDB within the same container database (CDB) or move it to a different CDB. It can be done as either:
  - **Cold cloning** (when the source PDB is closed)
  - **Hot cloning** (while the source PDB is open)
3. **Schema Cloning:** Copying an individual schema from one database to another. This includes all objects within the schema, such as tables, indexes, views, and triggers.
4. **Tablespace Cloning:** Duplication of a specific tablespace. This might involve creating a read-only copy of a tablespace for reporting purposes.

The cloning process can be done manually using commands like RMAN (Recovery Manager), Data Pump, or through Oracle Enterprise Manager, depending on the specific need.

# Step 1: Logging as SYSDBA & Check available PDB

```

C:\Users\Eric>sqlplus sys/admin@localhost:1521/ORACLE21CDB AS SYSDBA

SQL*Plus: Release 21.0.0.0.0 - Production on Wed Sep 25 16:47:45 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> col name for a20;
SQL> select name, open_mode from v$containers
2 /

NAME OPEN_MODE

CDB$ROOT READ WRITE
PDB$SEED READ ONLY
ORACLE21DBCPDB READ WRITE

SQL> |

```

- To connect to an Oracle database using SQL\*Plus as the SYS user with SYSDBA privileges.

SQL> sqlplus sys as SYSDBA  
or

C:\Users\Eric> sqlplus sys/admin@localhost:1521/ORACLE21CDB AS SYSDBA

- The query retrieves the names and open modes of all pluggable databases in the current container database.

SQL> col name for a20; -- This is for formatting our output.  
SQL> select name, open\_mode from v\$containers



## Step 2: Check if You Are on the Root Container and Display All PDBs

```
SQL>
SQL>
SQL> show con_name;

CON_NAME

CDB$ROOT
SQL>
SQL> show pdbs;

 CON_ID CON_NAME OPEN MODE RESTRICTED

 2 PDB$SEED READ ONLY NO
 3 ORACLE21DBCPDB READ WRITE NO

SQL>
```

- This command will return the name of the current container you are connected to, which could be the root container (CDB\$ROOT) or a specific pluggable database (PDB).

SQL> Show con\_name;

or

SQL> SELECT SYS\_CONTEXT('USERENV', 'CON\_NAME') FROM DUAL;

- Oracle instance that is currently running, which is usually the same as the **SID** (System Identifier) of the database.

SQL> SHOW PDBS;

# Step3: Identify the right path of our PDB

```
SQL>
SQL> SELECT CON_ID, TABLESPACE_NAME, FILE_NAME FROM CDB_DATA_FILES WHERE CON_ID = 3;
```

| CON_ID | TABLESPACE_NAME | FILE_NAME                                                 |
|--------|-----------------|-----------------------------------------------------------|
| 3      | SYSTEM          | E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\SYSTEM01.DBF  |
| 3      | SYSAUX          | E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\SYSAUX01.DBF  |
| 3      | UNDOTBS1        | E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\UNDOTBS01.DBF |

| CON_ID | TABLESPACE_NAME | FILE_NAME                                               |
|--------|-----------------|---------------------------------------------------------|
| 3      | USERS           | E:\ORACLE21CDATA\ORACLE21CDB\ORACLE21DBCPDB\USERS01.DBF |

- Retrieves the tablespace names and data file locations for the PDB with CON\_ID = 3. Based on the result, the **ORACLE21DBCPDB** (Pluggable Database 1) is located at the following file paths:

```
SQL> SELECT CON_ID, TABLESPACE_NAME, FILE_NAME FROM
CDB_DATA_FILES WHERE CON_ID = 3;
```

- This path is commonly used for organizing database files, making it easier to manage multiple PDBs within a CDB.

## What it does:

- CDB\_DATA\_FILES** → A **data dictionary view** in a multitenant (CDB/PDB) Oracle database. It shows **all datafiles** for **all containers (CDB + PDBs)**.
- CON\_ID** → The **container ID**:
  - 1 → **CDB\$ROOT** (root container).
  - 2 → **PDB\$SEED** (the seed template).
  - 3+ → **User-created PDBs**. In your case, CON\_ID = 3 is **ORACLE21DBCPDB**.
- TABLESPACE\_NAME** → Which **tablespace** the file belongs to (e.g., SYSTEM, SYSAUX, USERS, UNDO, TEMP).
- FILE\_NAME** → The **absolute path** on disk for the datafile storing that tablespace's data.

This is essential for **administration, monitoring, backup/recovery, and migration** of pluggable databases.

## Step 4: Clone the PDB

```
SQL>
SQL> CREATE PLUGGABLE DATABASE plsql2024
2 FROM ORACLE21DBCPDB
3 FILE_NAME_CONVERT = ('E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\',
4 'E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\plsql2024\');
```

Pluggable database created.

```
SQL>
SQL>
SQL>
SQL> show pdbs;
```

| CON_ID | CON_NAME       | OPEN MODE  | RESTRICTED |
|--------|----------------|------------|------------|
| 2      | PDB\$SEED      | READ ONLY  | NO         |
| 3      | ORACLE21DBCPDB | READ WRITE | NO         |
| 5      | PLSQL2024      | MOUNTED    |            |

```
SQL>
SQL> ALTER PLUGGABLE DATABASE plsql2024 OPEN;
```

Pluggable database altered.

```
SQL> |
```

- Command to clone existing PDB **"ORACLE21DBCPDB"** to a new one called **plsql2024**

```
SQL> CREATE PLUGGABLE DATABASE plsql2024
FROM ORACLE21DBCPDB
FILE_NAME_CONVERT =
('E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\',
'E:\oracle21cdata\ORACLE21CDB\oracle21dbcpdb\plsql2024\');
```

- Check existing PDB, plsql2024 is created with mounted status, which means that not yet opened.
- Open the new PDB (plsql2024)

```
SQL> show pdbs;
SQL> ALTER PLUGGABLE DATABASE plsql2024 OPEN;
```

# Step 5: Save the State (optional) & Create User in the New PDB

```
SQL> show con_name;

CON_NAME

CDB$ROOT
SQL>
SQL> ALTER SESSION SET CONTAINER=plsql2024;

Session altered.

SQL> CREATE USER adminpdb IDENTIFIED BY admin;

User created.

SQL> GRANT ALL PRIVILEGES TO adminpdb;

Grant succeeded.

SQL>
SQL>
SQL> SELECT username FROM dba_users WHERE username = 'ADMINPDB';

USERNAME

ADMINPDB

SQL> |
```

- Save the state: Ensure the PDB opens automatically upon the next startup of the CDB:  
SQL> ALTER PLUGGABLE DATABASE plsql2024 SAVE STATE;

- Shift from the root Container to the new DB  
SQL> ALTER SESSION SET CONTAINER=plsql2024;

- Create a new user for our new Database plsql2024  
SQL> CREATE USER adminpdb IDENTIFIED BY admin;

- Grant all privileges to the new user  
SQL> GRANT ALL PRIVILEGES TO admin;  
SQL> Display the user created

# Step 6: Connect my new user to sql developer

The screenshot shows the Oracle SQL Developer interface with the 'New / Select Database Connection' dialog box open. The dialog is configured for a new connection named 'plsql2024\_class'.

**Connections Panel (Left):**

- Oracle Connections
  - AUCA@2024
  - class2024
  - employeeManagementSystempdb
  - orade21c\_database
  - oradeclass
  - orade21c\_datab...
  - oradepdb
  - plsqlauca
- Database Schema Service Connections

**Dialog Box: New / Select Database Connection**

**Connection Name:** plsql2024\_class

**Database Type:** Oracle

**User Info:** Proxy User

**Authentication Type:** Default

**Username:** adminpdb

**Role:** default

**Password:** [Masked]

**Save Password:** [Unchecked]

**Connection Type:** Basic

**Details:** Advanced

**Hostname:** localhost

**Port:** 1521

**Service name:** plsql2024

**Status:** Success

**Buttons:** Save, Clear, Test, Connect, Cancel

**Footer:**

- SQLcl - The power of SQL Developer in a CLI
- Oracle Live SQL - Learn and share SQL, for free.

# Database User Management

## 1. What is a Database User?

- A **user** is an account created in the database that allows a person, service, or application to connect and interact with the database.
- Each user has:
  - Authentication credentials** (username/password, OS authentication, or external auth).
  - Privileges & roles** (what actions they are allowed to perform).
  - Default tablespaces** for storing data objects (in Oracle).

## 2. Key Aspects of User Management

```
CREATE USER eric IDENTIFIED BY strongPass
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE temp;
```

### a) User Creation

### b) Authentication

- Password-based** (most common).
- OS Authentication** (trusted external login).
- External Authentication** (LDAP, Kerberos, Cloud IAM).

### c) Privileges

- System Privileges** → Allow administrative actions (e.g., CREATE SESSION, CREATE TABLE).
- Object Privileges** → Allow operations on specific objects (e.g., SELECT, INSERT on a table).

```
GRANT CREATE SESSION, CREATE TABLE TO eric;
GRANT SELECT, INSERT ON employees TO eric;
```

### d) Roles

- A **role** is a collection of privileges grouped together for easier management.
- Example in Oracle:

```
CREATE ROLE analyst;
GRANT SELECT, INSERT ON sales TO analyst;
GRANT analyst TO eric;
```

### e) User Security Policies

- Enforce **password complexity and expiry**.
- Apply **least privilege principle** (give only what's needed).
- Monitor with **audit logs** (track user activities).

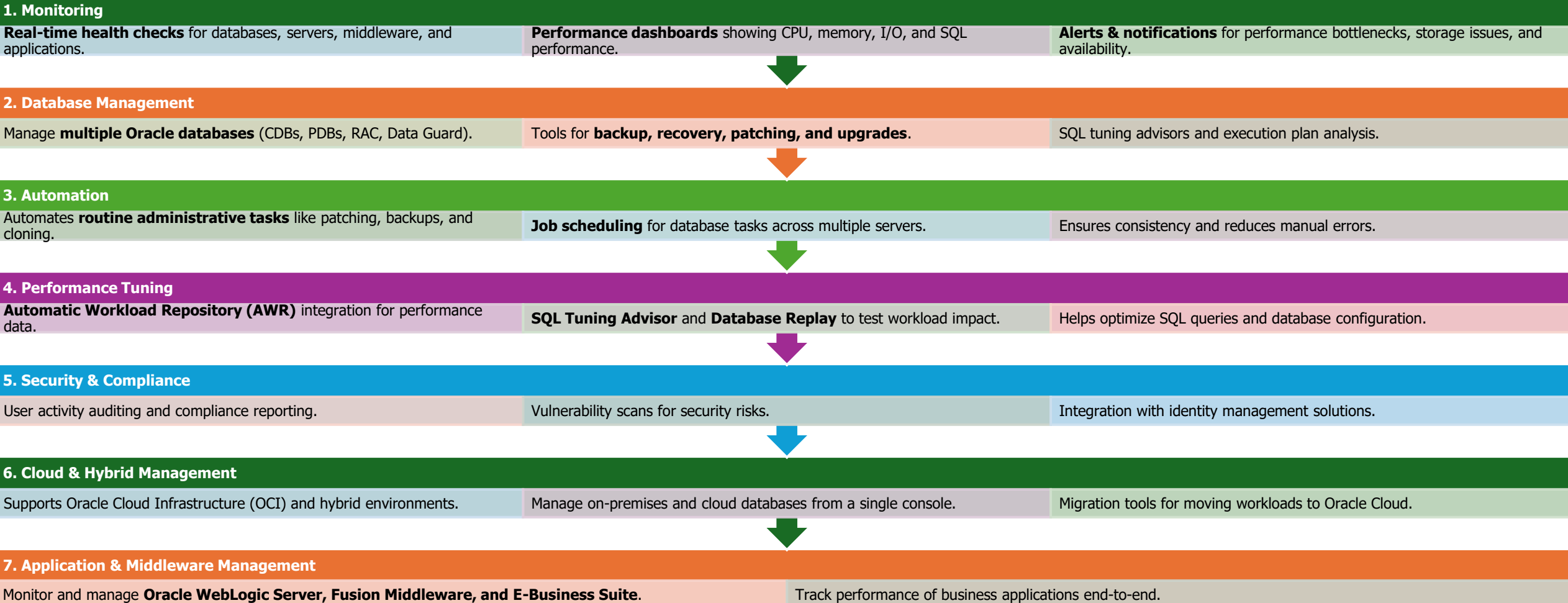
## 3. Why User Management Matters

- Ensures **security** (protects data from unauthorized access).
- Enables **accountability** (knowing who did what).
- Facilitates **resource allocation** (tablespace quotas, session limits).
- Simplifies **administration** through roles and profiles.



# Oracle Enterprise Manager (OEM)

Oracle Enterprise Manager (OEM) is Oracle’s **centralized management console** that provides administrators with tools to monitor, manage, and optimize Oracle environments — spanning databases, middleware, applications, engineered systems, and even hybrid cloud deployments.





# Oracle Enterprise Manager (OEM)



Oracle Enterprise Manager (OEM) is not just a database monitoring tool — it is a **full enterprise IT management platform** that improves **visibility, performance, automation, and security** across Oracle ecosystems.

# How to open Oracle Enterprise Manager: Connect to Sys

```
Windows PowerShell
PS C:\Users\Eric> sqlplus sys as sysdba

SQL*Plus: Release 21.0.0.0.0 - Production on Sun Sep 22 22:32:16 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> show con_name;

CON_NAME

CDB$ROOT
SQL> ALTER SESSION SET CONTAINER = CDB$ROOT;

Session altered.

SQL> SELECT SYS_CONTEXT('USERENV', 'CON_NAME') AS current_container FROM dual;

CURRENT_CONTAINER

CDB$ROOT
```

In this:

- **Connect to the System**

```
SQL> sqlplus sys as sysdba
```

- **Check if I am connected to the root container**

```
SQL> show con_name
```

- If not, I should use:

```
SQL> ALTER SESSION SET CONTAINER = CDB$ROOT;
Or
```

- Check the current container

```
SQL> SELECT SYS_CONTEXT('USERENV', 'CON_NAME') AS current_container
FROM dual;
```

# How to open Oracle Enterprise Manager: Verify configuration & set HTTP(s)

```
SQL> SELECT DBMS_XDB_CONFIG.GETHTTPPORT() AS HTTP_PORT,
2 DBMS_XDB_CONFIG.GETHTTPSPORT() AS HTTPS_PORT
3 FROM dual;
```

| HTTP_PORT | HTTPS_PORT |
|-----------|------------|
| 0         | 0          |

```
SQL>
SQL> BEGIN
2 DBMS_XDB_CONFIG.SETHTTPPORT(8080);
3 DBMS_XDB_CONFIG.SETHTTPSPORT(8443); -- Optional for HTTPS
4 END;
5 /
```

PL/SQL procedure successfully completed.

```
SQL>
SQL>
SQL> SELECT DBMS_XDB_CONFIG.GETHTTPPORT() AS HTTP_PORT,
2 DBMS_XDB_CONFIG.GETHTTPSPORT() AS HTTPS_PORT
3 FROM dual;
```

| HTTP_PORT | HTTPS_PORT |
|-----------|------------|
| 8080      | 8443       |

- **Verify Configuration:** After restarting, run the initial query again to confirm the new port settings.

```
▪ SELECT DBMS_XDB_CONFIG.GETHTTPPORT() AS HTTP_PORT,
DBMS_XDB_CONFIG.GETHTTPSPORT() AS HTTPS_PORT FROM dual;
```

- Since both HTTP\_PORT and HTTPS\_PORT are returning 0, it confirms that the HTTP(S) server is currently not enabled. Here's how to proceed:
- Set New HTTP(S) Ports: Let's set the HTTP and HTTPS ports to a specific value. Choose ports that are not in use (for example, 8080 for HTTP and 8443 for HTTPS):

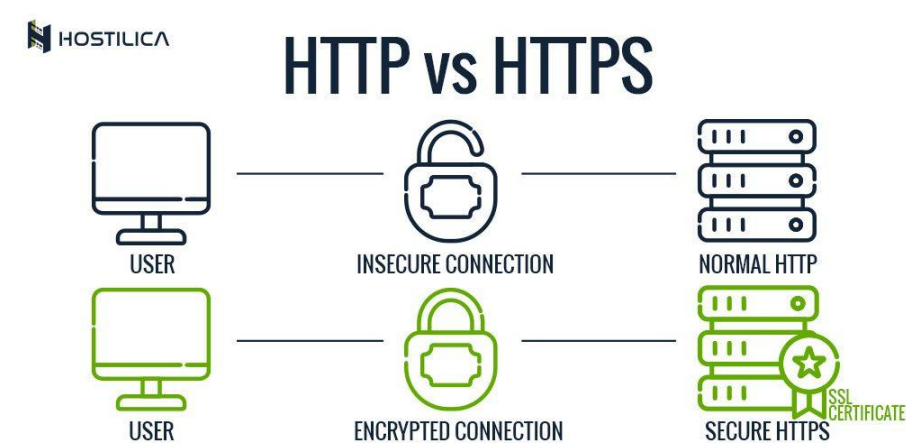
```
SQL> BEGIN
DBMS_XDB_CONFIG.SETHTTPPORT(8080);
DBMS_XDB_CONFIG.SETHTTPSPORT(8443); -- Optional for HTTPS
END;
/
```

- Check for Errors: If you encounter a port conflict again, try different port numbers.
- Verify Configuration: After executing the above, check the port settings again:

```
SQL> SELECT DBMS_XDB_CONFIG.GETHTTPPORT() AS HTTP_PORT,
DBMS_XDB_CONFIG.GETHTTPSPORT() AS HTTPS_PORT
FROM dual;
```

# Using LSNRCTL to Check Oracle Listener Status

```
Administrator: Command Prompt
C:\Windows\System32>lsnrctl status | findstr HOST
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=Eric-Admin)(PORT=1521)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=Eric-Admin)(PORT=1521)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=Eric-Admin)(PORT=5500))(Presentation=HTTP)(Session=RAW))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=Eric-Admin)(PORT=5501))(Security=(my_wallet_directory=C:\ORACLE21C_CONFIG\admin\oracle21cdb\xdb_wallet))(Presentation=HTTP)(Session=RAW))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=Eric-Admin)(PORT=8080))(Presentation=HTTP)(Session=RAW))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=Eric-Admin)(PORT=8443))(Security=(my_wallet_directory=C:\ORACLE21C_CONFIG\admin\oracle21cdb\xdb_wallet))(Presentation=HTTP)(Session=RAW))
C:\Windows\System32>
```



In this:

- **Identify the hostname that correspond to the port number 8443**

C:\Windows\System32>lsnrctl status | findstr HOST

# How to open Oracle Enterprise Manager: Restart database

```
SQL> SHUTDOWN IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP;
ORACLE instance started.

Total System Global Area 2147482528 bytes
Fixed Size 9857952 bytes
Variable Size 1392508928 bytes
Database Buffers 738197504 bytes
Redo Buffers 6918144 bytes
Database mounted.
Database opened.
SQL>
SQL> select dbms_xdb_config.gethttpsport() from dual;

DBMS_XDB_CONFIG.GETHTTPSPORT()

 8443

SQL> |
```

- Restart the Database: If the ports were successfully set, restart the database to apply changes:  
SQL> SHUTDOWN IMMEDIATE;  
STARTUP;

- This will display both port numbers in one result set.  
SQL> select dbms\_xdb\_config.gethttpsport() from dual;

# Connecting to Oracle PDB Locally Using SQL\*Plus for the Newly Created User



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Eric> sqlplus plsqlauca/password@//localhost:1521/PDB2

SQL*Plus: Release 21.0.0.0.0 - Production on Sun Sep 22 21:09:27 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Last Successful login time: Sun Sep 22 2024 19:15:47 +02:00

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> |
```



In this:

- **plsqlauca:** Username.
- **password:** Password for the plsqlauca user.
- **localhost:** Refers to your local machine.
- **1521:** The default Oracle listener port.
- **PDB2:** The service name of the pluggable database you're connecting to.


```
SQL> sqlplus plsqlauca/password@//localhost:1521/PDB2
```

# How to open Oracle Enterprise Manager: Restart database

PL/SQL Control Statements x 10 Handling PL/SQL Errors x PL/SQL For Loop - Geeksfor x PL/SQL - Loops x Sign In To Oracle Enterprise x

Not secure https://localhost:8443/em/login

YouTube Maps Gmail Messaging Wix Website Editor |...



**ORACLE ENTERPRISE MANAGER  
DATABASE EXPRESS**

Username

Password

Container Name

**Log in**

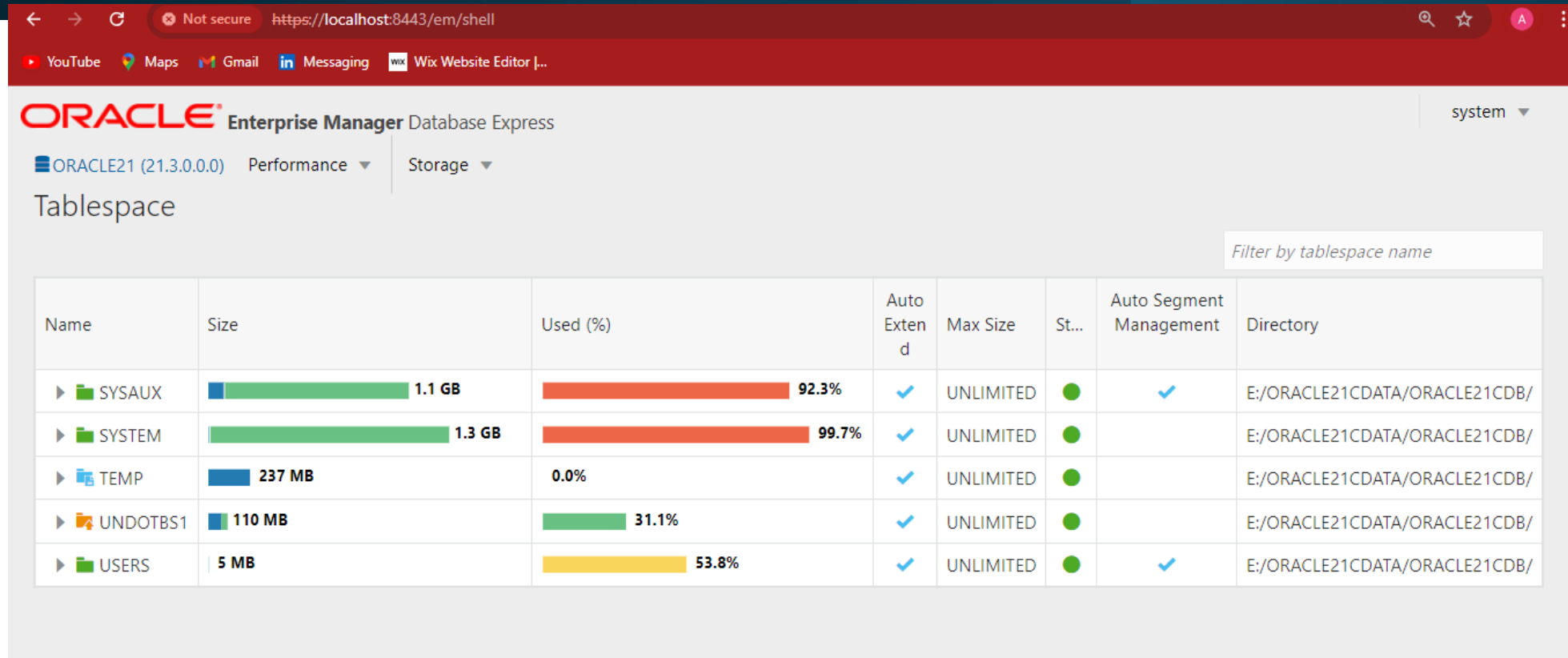
- To access Oracle Enterprise Manager, open your web browser and enter the following URL:  
`https://localhost:8443/em`

- The username should be "system" or "sys" as used during database creation.

- Use the password that was set during Oracle installation.
- Leave the container name blank.



# Oracle Enterprise Manager – Tablespaces

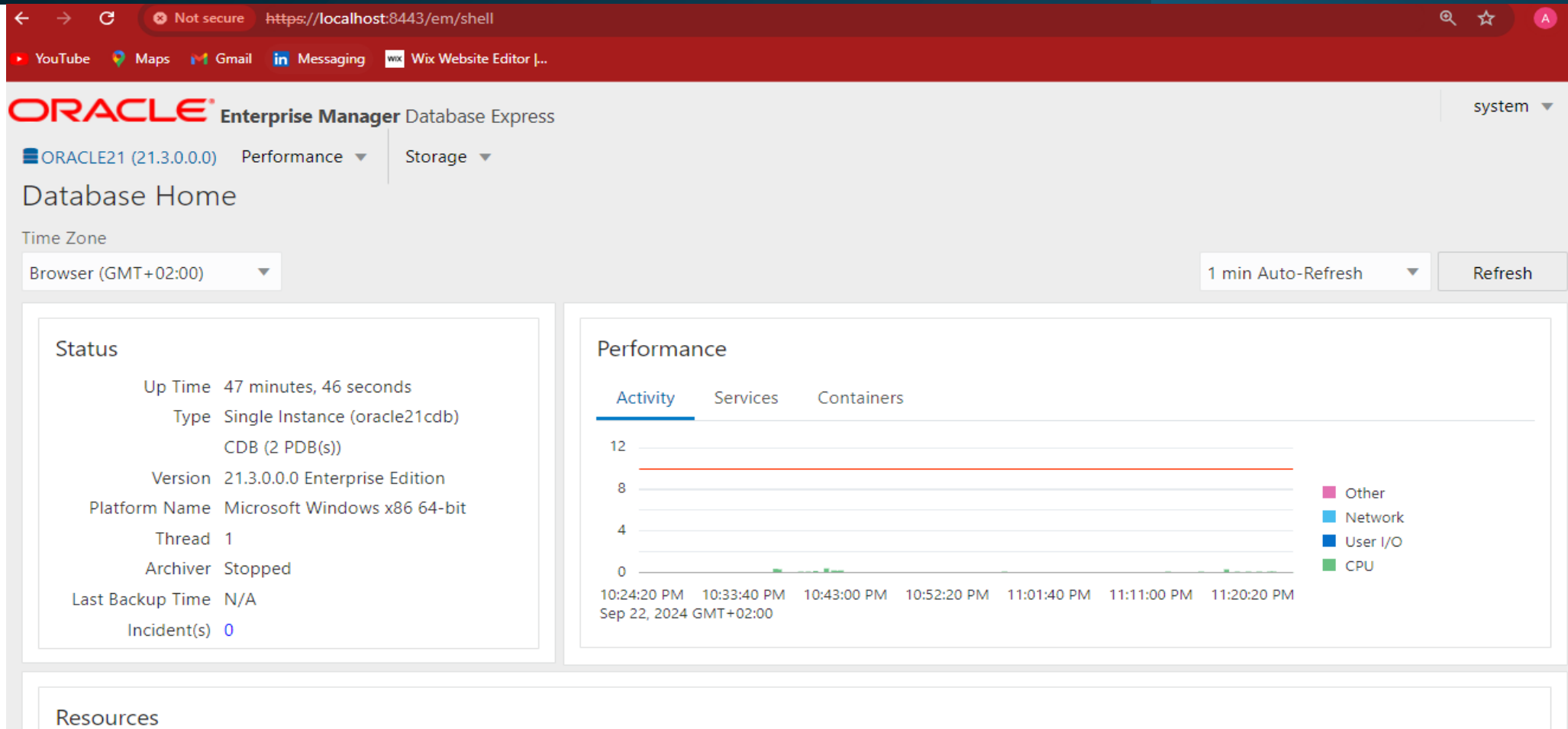


| Name       | Size   | Used (%) | Auto<br>Extend | Max Size  | St... | Auto Segment<br>Management | Directory                     |
|------------|--------|----------|----------------|-----------|-------|----------------------------|-------------------------------|
| ▶ SYSAUX   | 1.1 GB | 92.3%    | ✓              | UNLIMITED | ●     | ✓                          | E:/ORACLE21CDATA/ORACLE21CDB/ |
| ▶ SYSTEM   | 1.3 GB | 99.7%    | ✓              | UNLIMITED | ●     |                            | E:/ORACLE21CDATA/ORACLE21CDB/ |
| ▶ TEMP     | 237 MB | 0.0%     | ✓              | UNLIMITED | ●     |                            | E:/ORACLE21CDATA/ORACLE21CDB/ |
| ▶ UNDOTBS1 | 110 MB | 31.1%    | ✓              | UNLIMITED | ●     |                            | E:/ORACLE21CDATA/ORACLE21CDB/ |
| ▶ USERS    | 5 MB   | 53.8%    | ✓              | UNLIMITED | ●     | ✓                          | E:/ORACLE21CDATA/ORACLE21CDB/ |

## Example in Practice

- A **DBA (Database Administrator)** logs into OEM via a web browser.
- They see a **dashboard** showing:
  - Database health (green = healthy, red = problem)
  - Active sessions and queries
  - Storage usage trends
- If a tablespace is almost full, OEM **sends an alert** so the DBA can fix it before a failure happens.

# Oracle Enterprise Manager – Database Home



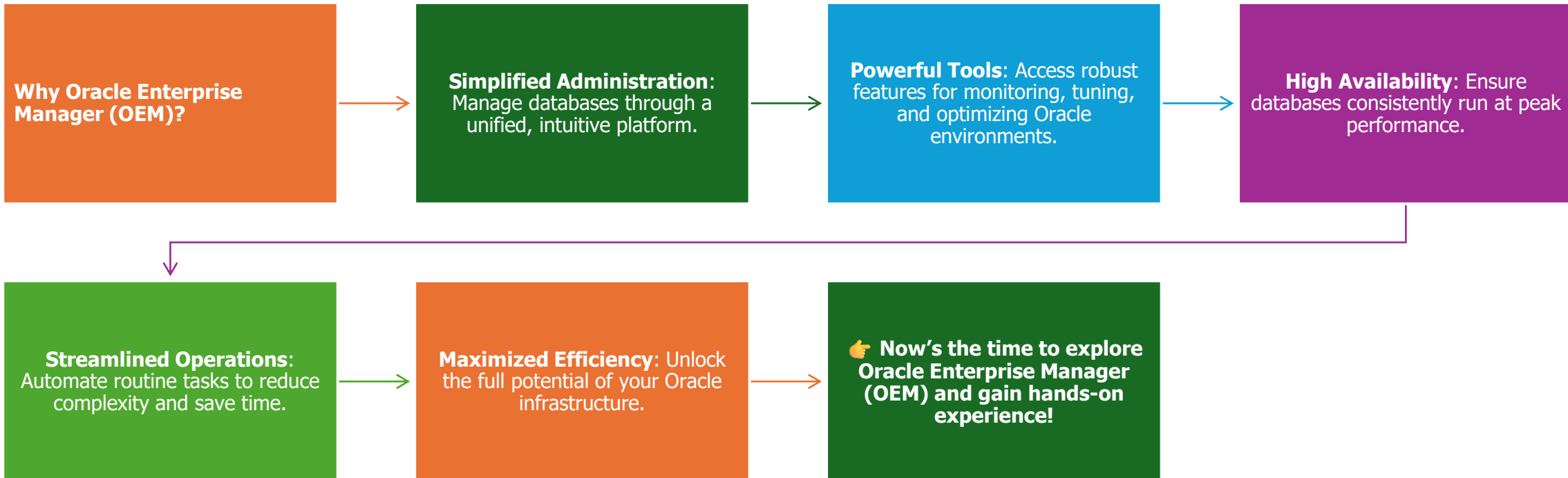
- 👉 Think of OEM as the **car dashboard**:
- It shows speed, fuel, engine status, and alerts.
  - Similarly, OEM shows the **health of your Oracle database**.

# Installing Oracle 23ai on macOS – Key Points & Caveats

- **No native macOS build**  
Oracle Database server is not natively available on macOS → must run via **containers** or a **Linux VM**
- **Container images provided**  
Oracle offers **free container images** (e.g. *Oracle Database Free / ADB Free*) runnable with **Docker / Podman / Colima**  
[Osman's DBlog](#) | [Oracle Free DB](#) | [Medium Guide](#)
- **Mac architecture matters**
- **Intel Macs** → x86\_64 images run directly in Docker/Podman
- **Apple Silicon (M1/M2/M3)** → need emulation (Colima, QEMU, Rosetta)  
[Suraj Ramesh](#) | [Stack Overflow](#) | [Ronekins Blog](#)
- **For development / learning only**  
Great for labs, demos, experiments — **not for production**.
- **Prerequisites**
  - Sufficient **CPU, RAM, disk** resources
  - Installed **Docker / Podman / Colima**
  - Familiarity with **command line, ports, env variables**
  - (*Optional*) Oracle clients (SQLcl, SQL Developer)



# It's time for you to explore Oracle Enterprise Manager (OEM) and get hands-on experience!



# Assignment II: Database Creation, Deletion & OEM

**Note:** No extensions will be granted. **Do it or get zero.**

## Task 1: Create a New Pluggable Database (PDB)

- Create a PDB named (**FirstTwoLettersOfFirstName\_pdb\_StudentID**)
  - Example: *er\_pdb\_2024101*
- Username format: **FirstName\_plsqlauca\_StudentID**
  - Example: *eric\_plsqlauca\_2024101*
- Choose a simple password.
- This account will store all your class work.

## Task 2: Create and Delete a PDB

- Create another PDB with the format:  
**FirstTwoLettersOfName\_to\_delete\_pdb\_StudentID**
  - Example: *er\_to\_delete\_pdb\_2024101*
- After creating it, delete it.
- Take **screenshots** of both creation and deletion.

## Task 3: Oracle Enterprise Manager

- Configure **Oracle Enterprise Manager (OEM)**.
- Provide a **screenshot of the dashboard** after finishing Task 1 & 2.
- Screenshot must show your username clearly.

## Report & Submission

- Prepare a **short report** containing:
  - Overview of tasks
  - Screenshots (PDB creation, deletion, OEM dashboard)
  - Notes if you encountered issues and how you solved them
- Submit your report through **GitHub (public repository link via my email)**.
- **Deadline:** Before **11:59 PM** on the same day of your class next week.

## Grading (Total: 5 points)

- Task 1: **2 points**
- Task 2: **2 points**
- Task 3: **1 point**



Stay Connected!