# Oracle Database Multitenant Architecture

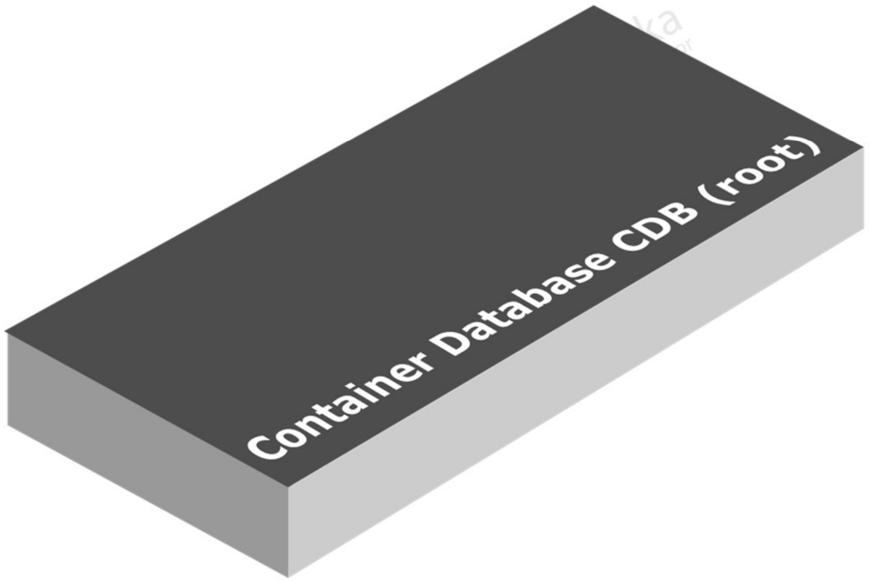
**By Ahmed Baraka**

# Objectives

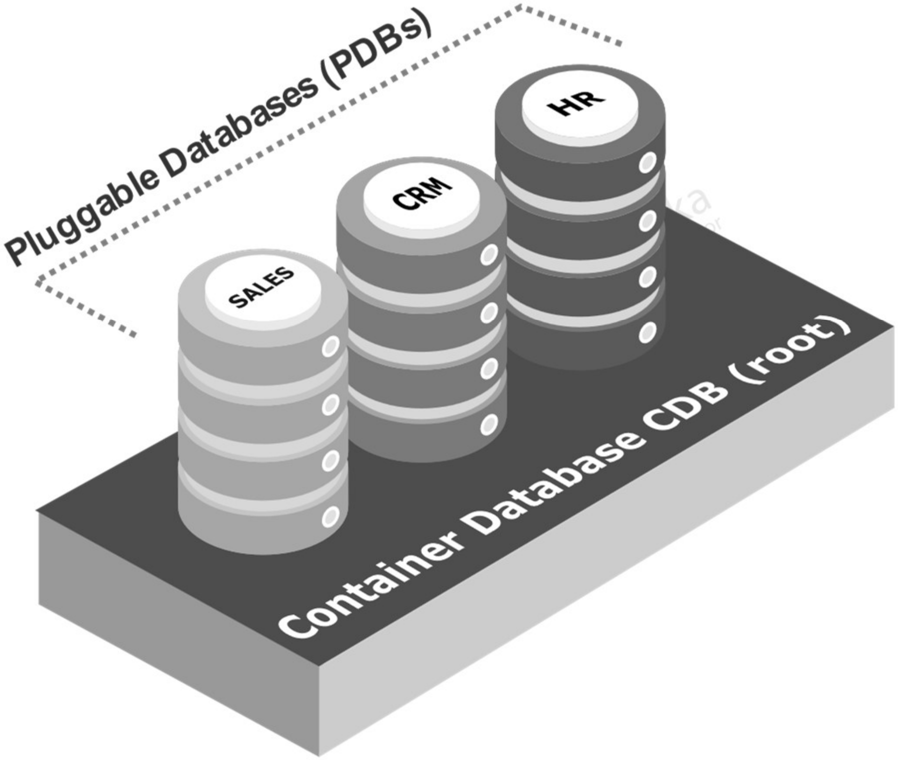
In this lecture, you will learn how to perform the following:

- Describe the difference between the CDB and PDB
- Describe the Multitenant Architecture components
- Use Data Dictionary Views and V$ Views in CDBs
- Describe the common files in CDBs
- Describe the difference between the local users and common users
- Create a multitenant database (CDB) using dbca
- Describe the difference between the shared undo and the local undo configuration

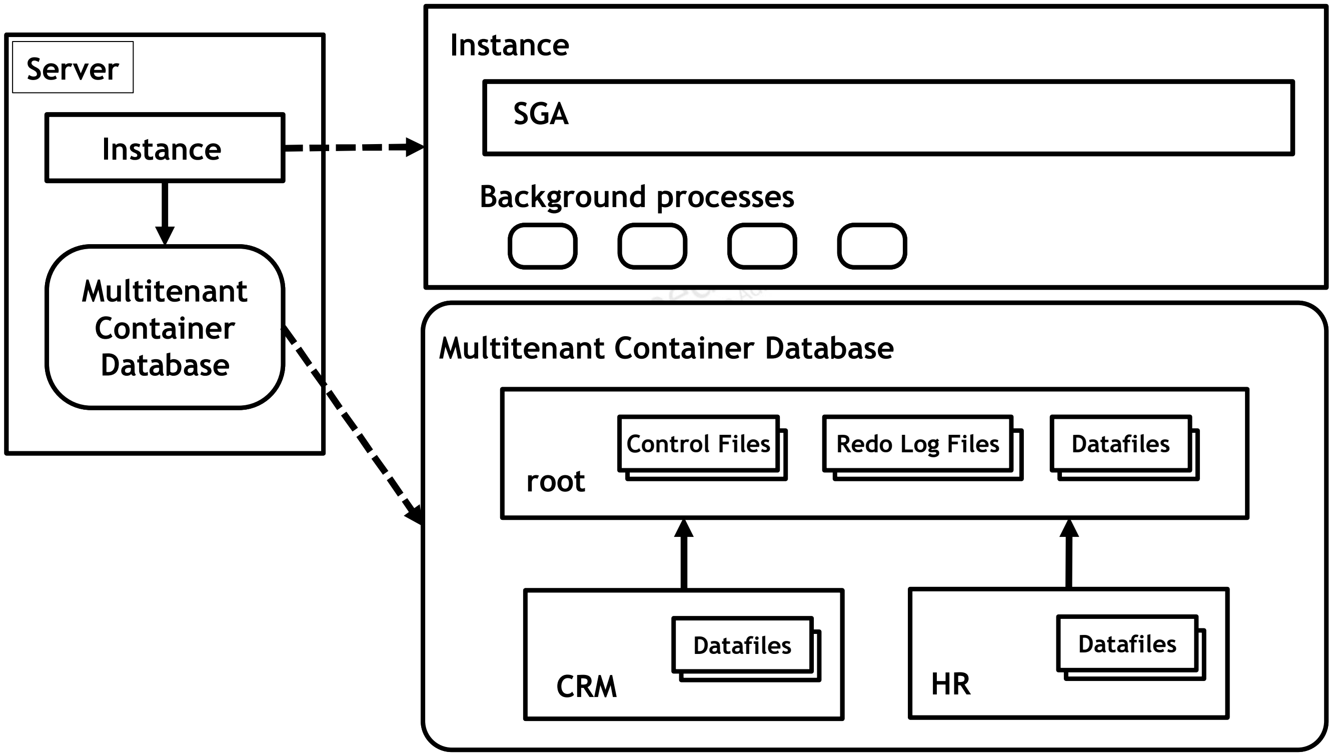
## What is the Oracle Database Multitenant Architecture?



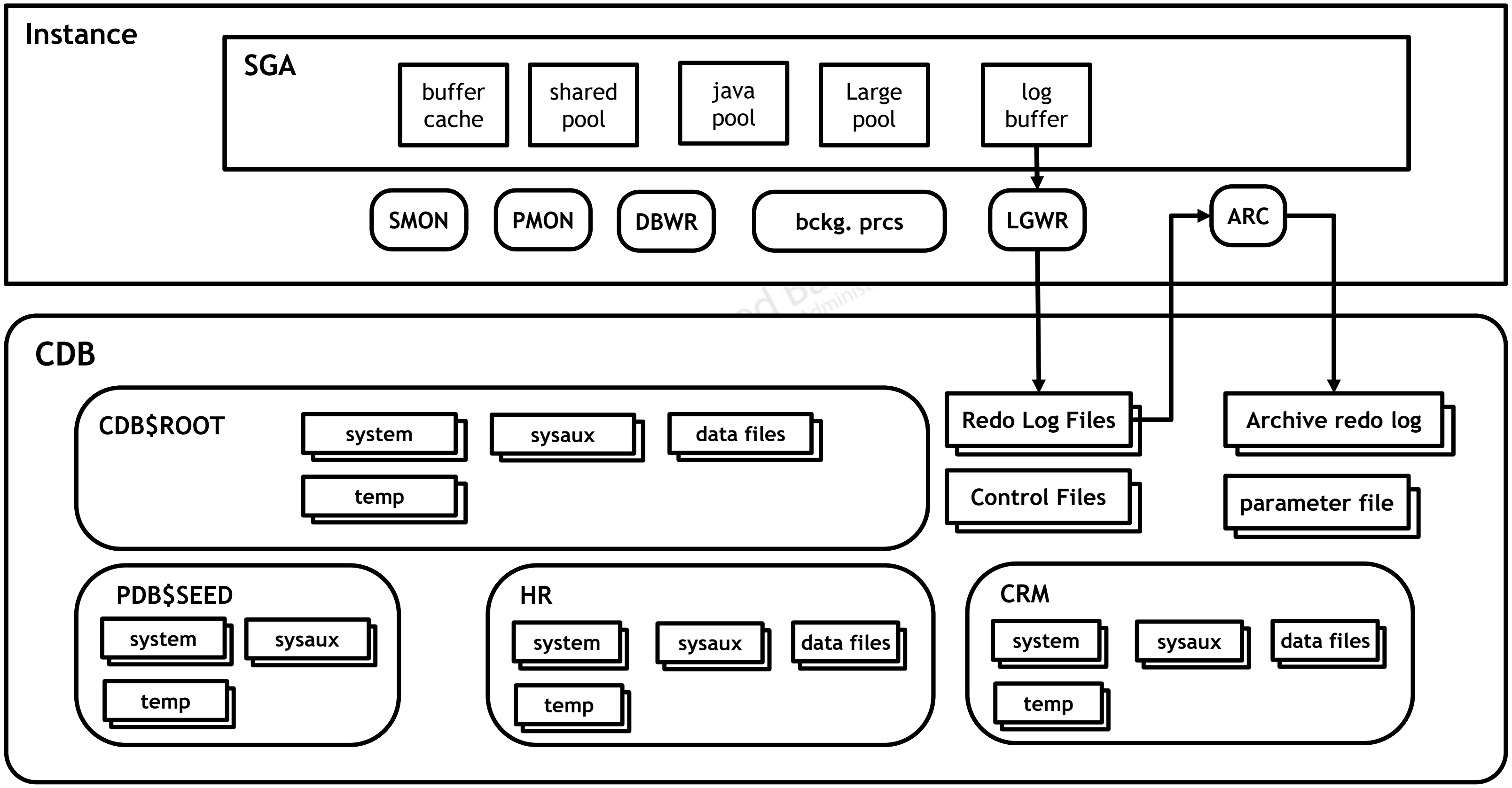
## What is the Oracle Database Multitenant Architecture?



## Multitenant Architecture



## Multitenant Architecture



# Multitenant Architecture Terms

| Term | Definition |
|---|---|
| Container Database (CDB) | A multitenant container database that houses zero or more pluggable databases. |
| non-CDB | An Oracle database that is not a multitenant container database. |
| Pluggable Database (PDB) | Portable collection of schemas, schema objects, and non-schema objects that appears to an Oracle Net client as a non-CDB. |
| Root container (CDB$ROOT) | A master set of data files and metadata (data dictionary tables, Oracle-supplied packages, system users) and containing information regarding all containers within a CDB. |
| Seed pluggable database (PDB$SEED) | A template of data files and metadata used to create new pluggable databases. |

# About Multitenant Architecture

- Multitenant is implemented by creating a CDB
- The redo log files, control files, and archive files are owned by the CDB, but not by specific PDB
- The root contains the Oracle-supplied objects and users
- The root is the parent of all the PDBs
- PDBs are created from the **`PDB$SEED`**
- Application users connect to PDBs
- Undo tablespaces can be shared or be created per PDB

# Data Dictionary Views within a PDB

| View Category within a PDB | Description |
|---|---|
| USER_*** | Objects owned by the current user |
| ALL_*** | Objects accessible by the current user |
| DBA_*** | All the objects in within the PDB |

# Data Dictionary Views in CDB level

| View Category within a PDB | Description |
|---|---|
| USER_*** | Objects owned by the current user |
| ALL_*** | Objects accessible by the current user |
| DBA_*** | All the objects in within the current container (root or pdb) |
| CDB_*** | All objects in the CDB (container identified by CON_ID)<br>Examples: CDB_PDBS, CDB_TABLESPACES, CDB_USERS |
| V$*** | SGA accessed by all the containers (container identified by CON_ID) |

# Showing the Tables Owned by Specific Schemas in Multiple PDBs

```
SELECT p.PDB_ID, p.PDB_NAME, t.OWNER, t.TABLE_NAME
FROM DBA_PDBS p, CDB_TABLES t
WHERE p.PDB_ID > 2
  AND t.OWNER ='HR'
  AND p.PDB_ID = t.CON_ID
ORDER BY p.PDB_ID;
```

# Container Identifications

- A container a name and a **CON_ID** within the CDB

| CON_ID | Corresponding Container |
|---|---|
| 0 | CDB level (no container) |
| 1 | CDB$ROOT |
| 2 | PDB$SEED |
| Anything else | User Created PDB |

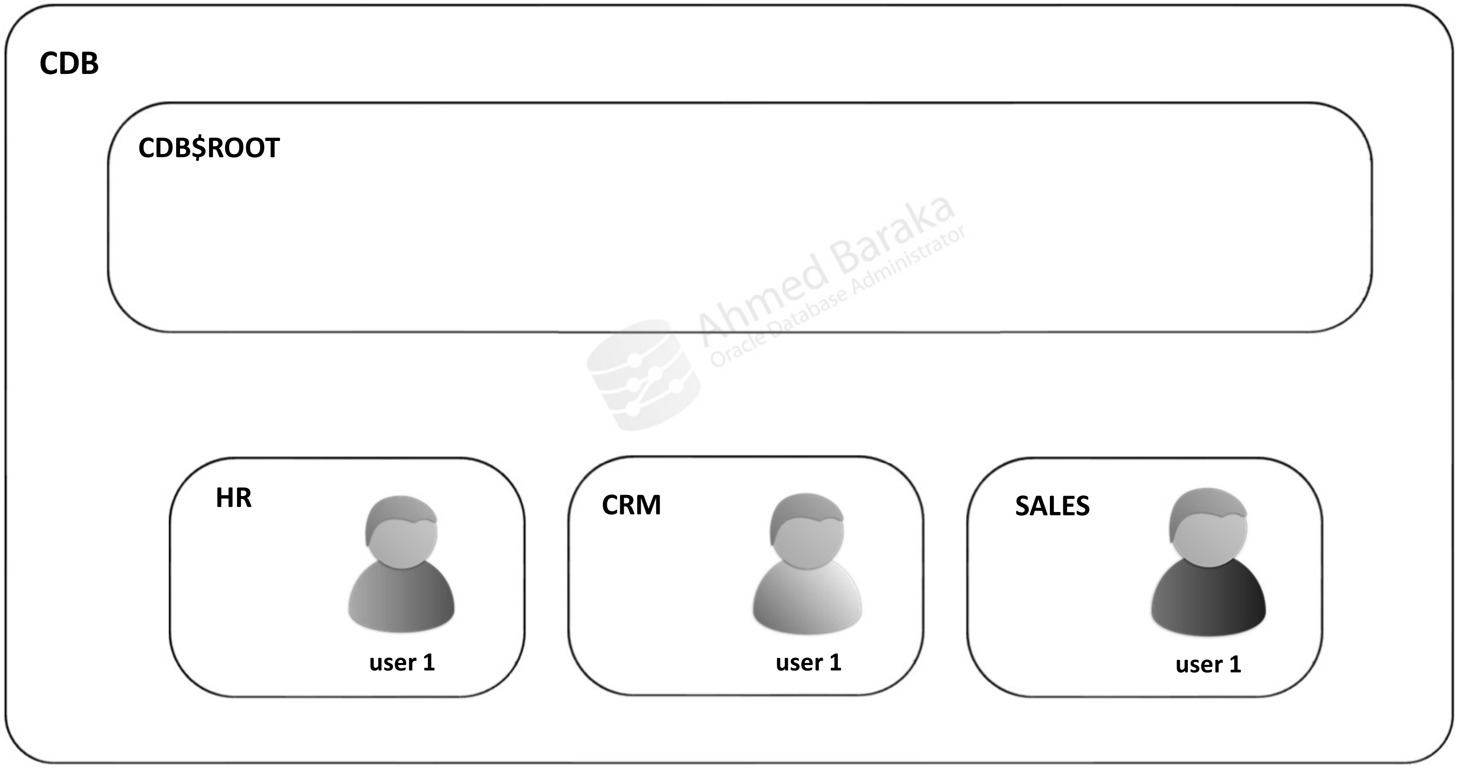
- **CON_UID** : does  not change when PDB is moved
- **GUID**: 16-byte RAW value that never changes

- They can be obtained from the **`V$CONTAINER`**

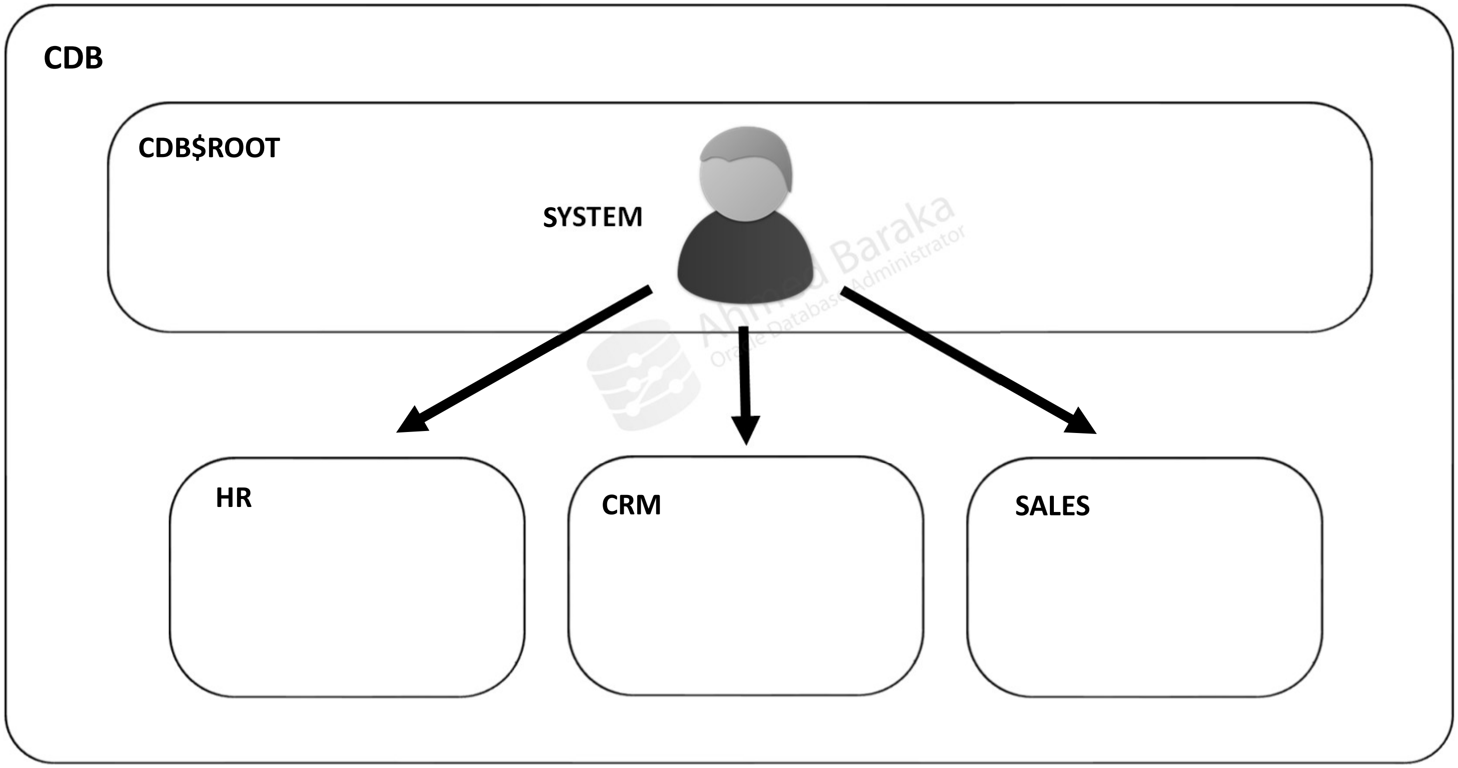
# Common Files in CDBs

- **SPFILE**
  - Some params are PDB modifiable ( ISPDB_MODIFIABLE=TRUE )
  - Saved in the CDB dictionary PDB_SPFILE$
- **Control Files**
  - Exist in the CDB level only
  - Hold records of all the PDBs
- **Temporary Tablespaces**
  - A PDB can have its own, or use the shared one at the root
  - MAX_SHARED_TEMP_SIZE sets the maximum size allowed

# Local Users



# Common Users

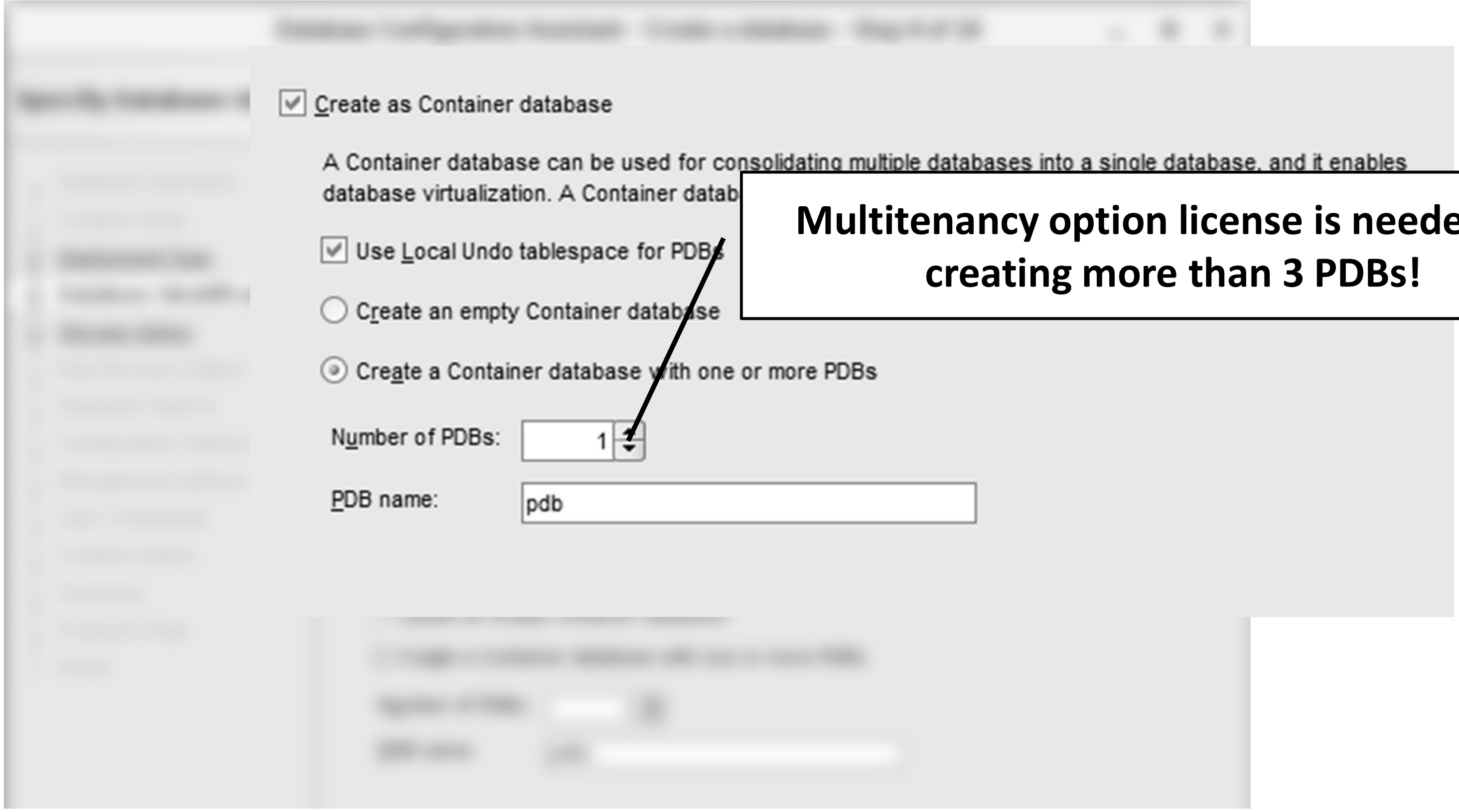


# Local and Common Users

- **A local user:**
  - Defined in the PDB and is not known outside of that PDB
  - Can connect only to the PDB where it is defined
  - If it has the privileges granted, a user can work on the application data within the PDB
  - Cannot be defined in the root

- **Common User:**
  - A common user is defined in the root
  - Only common users can be defined in the root
  - A common user is known in every existing and future PDB
  - Some administrative tasks should be done by common users

# Creating CDB Database using dbca



☑ Create as Container database

A Container database can be used for consolidating multiple databases into a single database, and it enables database virtualization. A Container datab

☑ Use Local Undo tablespace for PDBs

○ Create an empty Container database

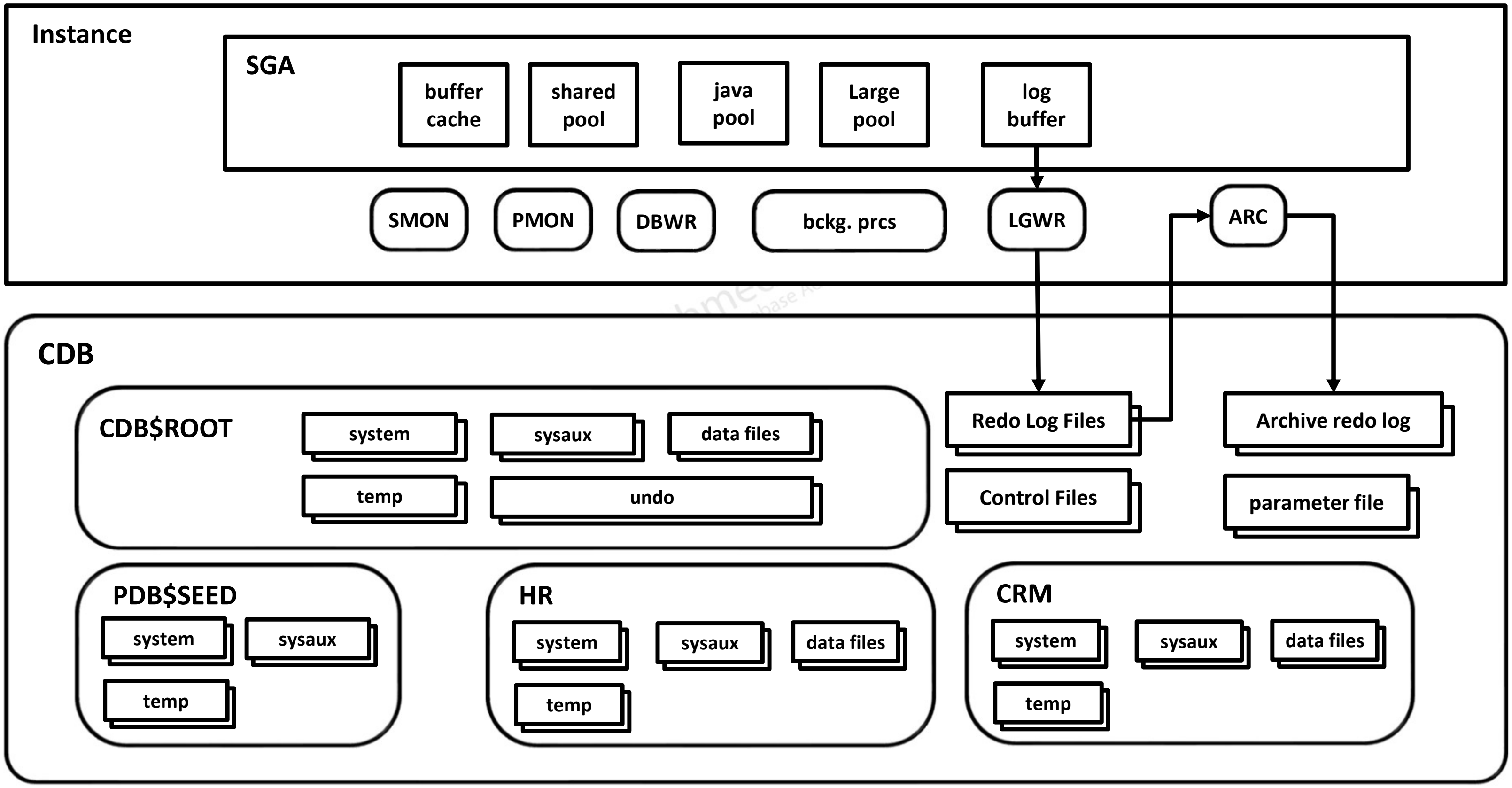
● Create a Container database with one or more PDBs

Number of PDBs: 1

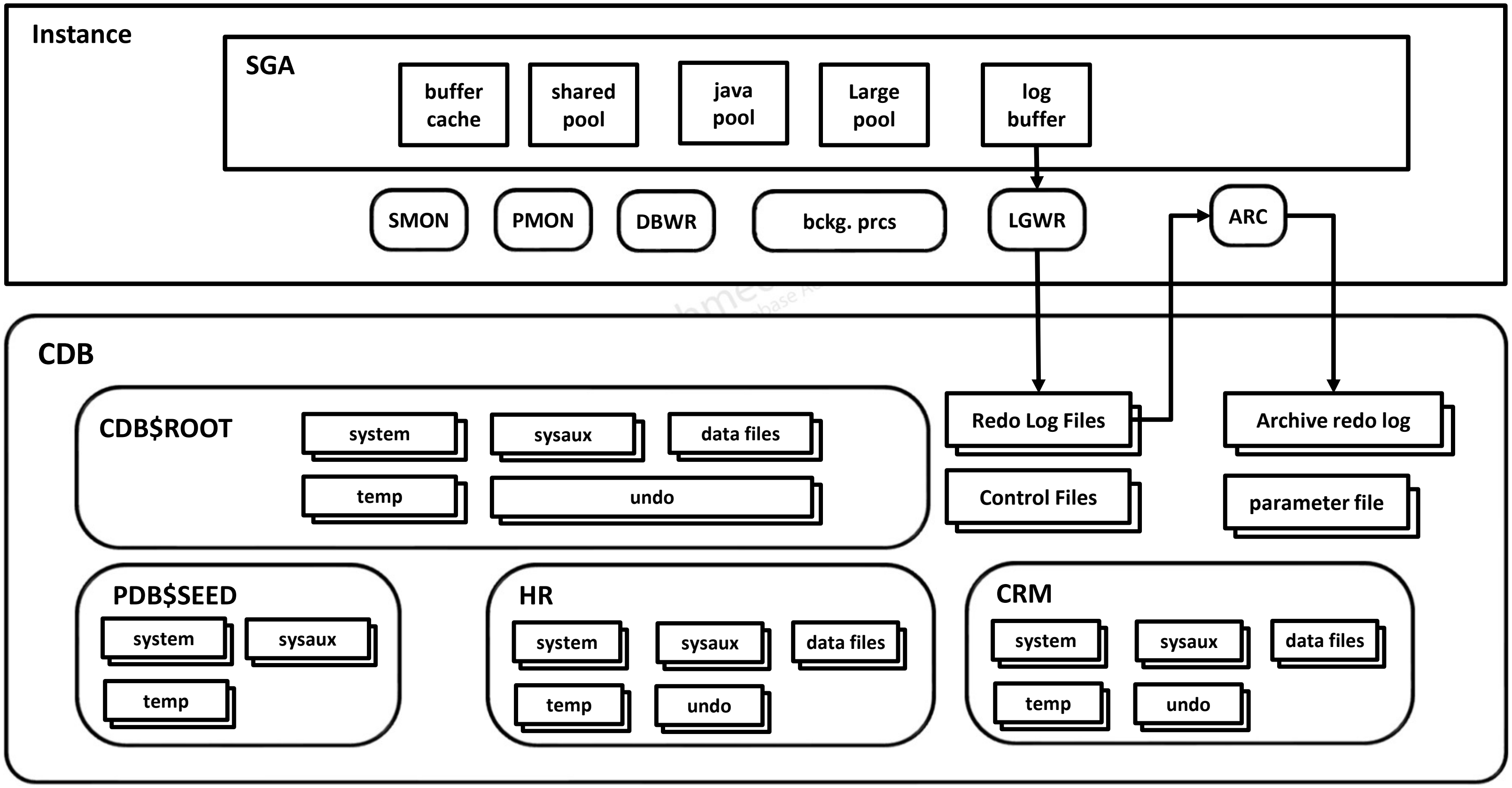
PDB name: pdb

Multitenancy option license is needed for creating more than 3 PDBs!

# Multitenant Architecture: Shared Undo



# Multitenant Architecture: Local Undo mode



# Summary

In this lecture, you should have learnt how to perform the following:

- Describe the difference between the CDB and PDB
- Describe the Multitenant Architecture components
- Use Data Dictionary Views and V$ Views in CDBs
- Describe the common files in CDBs
- Describe the difference between the local users and common users
- Create a multitenant database (CDB) using dbca
- Describe the difference between the shared undo and the local undo configuration