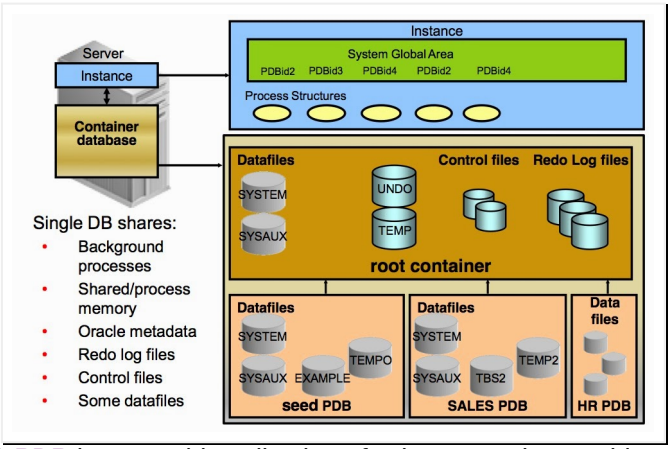
12C Database Architecture :

12 C Supports Multi tenet Architechture which is a newly added feature in Oracle 12C.The multitenant architecture enables an Oracle database to function as a multitenant container database (CDB) that includes zero, one, or many customer-created pluggable databases (PDBs). A PDB is a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a non-CDB. All Oracle databases before Oracle Database 12c were non-CDBs.



As shown in the above Diagram A **PDB** is a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a**non-CDB**. All Oracle databases before Oracle Database 12c were non-CDBs.

About Containers in a CDB

A **container** is either a PDB or the **root container** (also called *the root*). The root is a collection of schemas, schema objects, and nonschema objects to which all PDBs belong (see "Overview of Containers in a CDB").
Every CDB has the following containers:

- Exactly one root

  The root stores Oracle-supplied metadata and common users. An example of metadata is the source code for Oracle-supplied PL/SQL packages (see"Data Dictionary Architecture in a CDB"). A common user is a database user known in every container (see "Common Users in a CDB"). The root container is named CDB$ROOT.

- Exactly one **seed PDB**

  The seed PDB is a system-supplied template that the CDB can use to create new PDBs. The seed PDB is named PDB$SEED. You cannot add or modify objects in PDB$SEED.

- Zero or more user-created PDBs

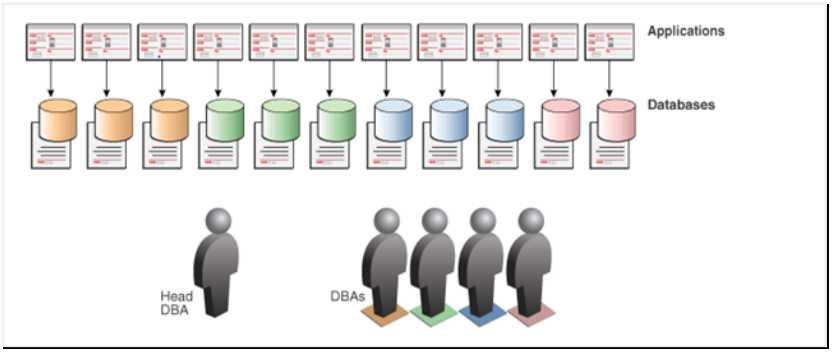
  A PDB is a user-created entity that contains the data and code required for a specific set of features. For example, a PDB can support a specific application, such as a human resources or sales application. No PDBs exist at creation of the CDB. You add PDBs based on your business requirements.

Benefits of the Multitenant Architecture

Large enterprises may use hundreds or thousands of databases. Often these databases run on different platforms on multiple physical servers. Because of improvements in hardware technology, especially the increase in the number of CPUs, servers

are able to handle heavier workloads than before. A database may use only a fraction of the server hardware capacity. This approach wastes both hardware and human resources.

For example, 100 servers may have one database each, with each database using 10% of hardware resources and 10% of an administrator's time. A team of DBAs must manage the SGA, database files, accounts, security, and so on of each database separately, while system administrators must maintain 100 different computers.



Benefits of the Multitenant Architecture for Database Consolidation

The process of consolidating data from multiple databases into one database on one computer is known as **database consolidation**. Starting in Oracle Database 12c, the Oracle Multitenant option enables you to consolidate data and code *without altering existing schemas or applications*

The **PDB/non-CDB compatibility guarantee** means that a PDB behaves the same as a non-CDB as seen from a client connecting with Oracle Net. The installation scheme for an application back end that runs against a non-CDB runs the same against a PDB and produces the same result. Also, the run-time behavior of client code that connects to the PDB containing the application back end is identical to the behavior of client code that connected to the non-CDB containing this back end. Operations that act on an entire non-CDB act in the same way on an entire CDB, for example, when using Oracle Data Guard and database backup and recovery. Thus, the users, administrators, and developers of a non-CDB have substantially the same experience after the database has been consolidated.

Using the multitenant architecture for database consolidation has the following benefits:

- Cost reduction:

  By consolidating hardware and sharing database memory and files, you reduce costs for hardware, storage, availability, and labor. For example, 100 PDBs on a single server share one database instance and one set of database files, thereby requiring less hardware and fewer personnel.

- Easier and more rapid movement of data and code:

  By design, you can quickly plug a PDB into a CDB, unplug the PDB from the CDB, and then plug this PDB into a different CDB. The implementation technique for plugging and unplugging is similar to the **transportable tablespace** technique.

- Easier management and monitoring of the physical database:

  The **CDB administrator** can attend to one physical database (one set of files and one set of database instances) rather than split attention among dozens or hundreds of non-CDBs. Backup strategies and disaster recovery are simplified.

- Separation of data and code

Although consolidated into a single physical database, PDBs mimic the behavior of non-CDBs. For example, if user error loses critical data, a PDB administrator can use Oracle Flashback or point-in-time recovery to retrieve the lost data without affecting other PDBs.

- Secure separation of administrative duties:

  A user account is common, which means that it can connect to any container on which it has privileges, or local, which means that it is restricted to a specific PDB. A CDB administrator can use a common user account to manage the CDB. A PDB administrator uses a local account to manage an individual PDB. Because a privilege is contained within the container in which it is granted, a local user on one PDB does not have privileges on other PDBs within the same CDB.

- Ease of performance tuning:

  It is easier to collect performance metrics for a single database than for multiple databases. It is easier to size one SGA than 100 SGAs.

- Support for Oracle Database Resource Manager

  In a multitenant environment, one concern is contention for system resources among the PDBs running on the same computer. Another concern is limiting resource usage for more consistent, predictable performance. To address such resource contention, usage, and monitoring issues, you can use Oracle Database Resource Manager (see "Database Resource Manager").

- Fewer database patches and upgrades :

  It is easier to apply a patch to one database than to 100 databases, and to upgrade one database than to upgrade 100 databases.