# Overview of a Multitenant Environment

<http://docs.oracle.com/database/122/ADMIN/overview-of-managing-a-multitenant-environment.htm#ADMIN14145>

## Components of a CDB

A CDB includes the following components:

* Root – The root, named CDB$ROOT, stores Oracle metadata and common users. A CDB has exactly one root.
* CDB seed – The seed, named PDB$SEED, is a template that you can use to create new PDBs. You cannot add objects to or modify objects in the CDB seed. A CDB has exactly one seed.
* PDB – A PDB appears to users and apps as if it were a standalone database. A PDB is fully backward compatible with releases before 12c.

# Creating and Configuring a CDB

<http://docs.oracle.com/database/122/ADMIN/creating-and-configuring-a-cdb.htm#ADMIN13517>

## Decide How to Configure the CDB

|  |  |
| --- | --- |
| Action | Considerations for a CDB |
| Plan the tables and indexes for the pluggable databases (PDBs) and estimate the amount of space they will require. | In a CDB, most user data is in the PDBs. The root contains no user data or minimal user data. Plan for the PDBs that will be part of the CDB. The disk storage space requirement for a CDB is the space required for the Oracle Database installation plus the sum of the space requirements for all of the PDBs that will be part of the CDB. A CDB can contain up to 4,096 PDBs, but you can limit the CDB to a smaller number of PDBs by setting the MAX\_PDBS initialization parameter. |
| Plan the layout of the underlying operating system files your CDB will comprise. | There are separate data files for the root, the CDB seed and each PDB.  There is one redo log for a single-instance CDB. |
| Plan for the number of background processes that will be required by the CDB. | There is one set of background processes shared by the root and all PDBs. |
| Set the global DB name (the name and location of the CDB) through the DB\_NAME and DB\_DOMAIN parameters. | The global name of the root is the global DB name of the CDB.  The global name of a PDB is defined by the PDB name and the DB\_DOMAIN parameter. |
| SPFILE | A CDB uses a single SPFILE. Values can be inherited by PDBs. You can set some initialization parameters for a PDB with the ALTER SYSTEM statement.  To create a CDB, the ENABLE\_PLUGGABLE\_DATABASE parameter must be set to TRUE. |
| Select the character set. | PDBs that are created from the CDB seed inherit the charater set. |
| SYSAUX tablespace | There is a separate SYSAUX tablespace for the root and each PDB. |
| Temporary tablespaces | There is a default temp tablespace for each container in the CDB, including the root and each PDB. |
| Undo | A CDB can have one active undo tablespace for the entire CDB or a separate undo for each container. You can specify the undo mode during CDB creation, and also change it later.  When you choose to have one active undo tablespace for the entire CDB, shared undo is used, and local undo is disabled.  The best practice is to use local undo. It has minor overhead but several benefits. |
| Starting/Stopping | You can start up and shut down an entire CDB. When the CDB is open, you can control the open mode of PDBs by using the ALTER PLUGGABLE DATABASE statement. ALTER DATABASE [OPEN|CLOSE] are supported against a PDB. You can also use STARTUP and SHUTDOWN for a PDB. STARTUP MOUNT is a CDB-only operation. |

## Using the CREATE DATABASE Statement for a CDB

To create a CDB, the CREATE DATABASE statement must include the ENABLE PLUGGABLE DATABASE clause. This will create a CDB with the root and seed. This statement uses the root's files to generate the names of the CDB seed's files. You must specify the locations. After completion, you can use the seed to create new PDBs.

You must specify the names and locations of the CDB seed's files in one of these ways (in order of precedence):

1. The SEED FILE\_NAME\_CONVERT clause
2. Oracle Managed Files
3. The PDB\_FILE\_NAME\_CONVERT parameter

The SEED FILE\_NAME\_CONVERT clause of the CREATE DATABASE statement specifies how to generate the names of the CDB seed's files using the names of root's files. You can use this in one of the following ways:

* One or more file name patterns and replacement file name patterns (e.g. SEED FILE\_NAME\_CONVERT = ('/oracle/dbs/', '/oracle/pdbseed/')
* NONE when no file names should be converted (the same as omitting the clause)

Add the undo\_mode\_clause to an ENABLE PLUGGABLE DATABASE clause to specify the undo mode of the CDB. To configure local undo mode, specify LOCAL UNDO ON. To configure shared undo, specify LOCAL UNDO OFF. If the clause is omitted, then shared undo mode is the default.

To create a CDB:

1. Set the ORACLE\_SID, ORACLE\_HOME and PATH environment variables
2. Use the CREATE DATABASE statement with the ENABLE PLUGGABLE DATABASE clause (other options can also be specified)
3. Run the catcdb.sql script
4. Run scripts to install additional options

# Creating and Removing PDBs with SQL\*Plus

<http://docs.oracle.com/database/122/ADMIN/creating-and-removing-pdbs-with-sql-plus.htm#ADMIN13549>

## About Creating and Removing PDBs

You can create a PDB by using the CDB seed or an application seed, cloning an existing PDB or non-CDB, or using a non-CDB.

|  |  |
| --- | --- |
| Technique | Description |
| Create a PDB by using the seed | Create a PDB in a CDB using the files of the CDB seed or application seed. This copies the files of the seed to a new location and associates the new files with the PDB. |
| Cloning an existing PDB or non-CDB | Clone a source PDB or non-CDB. This copies the files of the source DB to a new location and associates them with the new PDB. |
| Using a non-CDB | You can adopt a non-CDB into a PDB. Use the DBMS\_PDB package to create an unplugged PDB from a 12c non-CDB. Then plug it into a CDB. |

## The CREATE PLUGGABLE DATABASE Statement

To you use the CREATE PLUGGABLE DATABASE statement you must understand the clauses.

### File Location of the New PDB

You can specify the locations of files used by the new PDB.

* FILE\_NAME\_CONVERT – Use this clause when the files are not yet at their ultimate destination, and you want to copy or move them during PDB creation.
* CREATE\_FILE\_DEST – Use this clause to enable OMF.

### Storage Limits

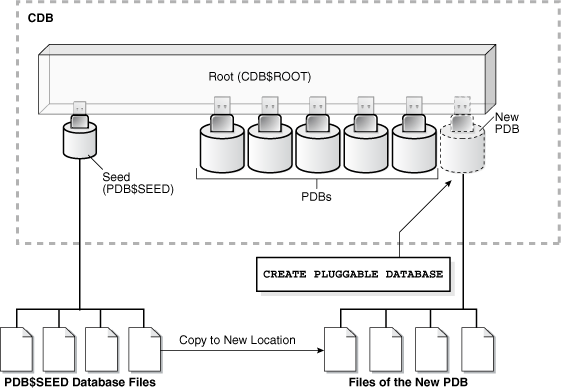
The STORAGE clause specifies the following limits:

* The amount of storage that can be used by all tablespaces that belong to the PDB
* The amount of storage that can be used by unified audit OS spillover files
* The amount of diagnostics in the ADR that can be used by the PDB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clause | Can be used when | Question | Yes | No |
| DEFAULT TABLESAPCE | Creating from seed | Do you want to specify a default TS? | Use this clause | Omit this clause |
| FILE\_NAME\_CONVERT | Creating from a seed | Do you want to spcify the target locations of the files? | Use this clause | Omit this clause |
| PARALLEL | From a seed | Do you want to use parallel execution to create PDB? | To specify degree use this. To let Oracle decide omit this clause | Set it to 0 or 1 |
| ROLES | From a seed | Do you want to grant predefined roles to the PDB\_DBA role locally? | Use this clause | Omit this clause |

## Creating a PDB Using the Seed

You can use the CREATE PLUGGABLE DATABASE statement to create a new PDB by using the files of the CDB seed. The statement copies these files to a new location and associates them with the new PDB.



When you create a new PDB from the seed, you must specify an administrator for the PDB. The statement creates the administrator as a local user in the PDB and grants the PDB\_DBA role locally to the administrator.

To create a PDB from the seed:

1. Ensure the current container is the CDB root.
2. Run the CREATE PLUGGABLE DATABASE statement and specify a local administrator for the PDB. Other clauses are optional.

After you create the PDB, it is mounted and its status is NEW. You can view the open mode of a PDB by querying the OPEN\_MODE column in the V$PDBS view. You can view the status of a PDB by querying the STATUS column of the CDB\_PDBS or DBA\_PDBS view.

A new default service is created for the PDB. The service has the same name as the PDB and can be used to access the PDB. Oracle Net Services must be configured properly for clients to access this service.

1. Open the new PDB in read/write mode. After the PDB is opened in read/write mode, its status is NORMAL.

A local user with the name of the specified local administator is created and granted the PDB\_DBA common role locally in the PDB. If this user was not granted administrator privileges during creation, use the SYS and SYSTEM common users.

# Create SPFILE

<https://docs.oracle.com/database/121/SQLRF/statements_6018.htm#SQLRF01315>

When you create a server parameter file in a multitenant container database (CDB), the current container can be the root or a PDB.

* If the current container is the root, then the values that you set for initialization parameters in the root are used as default values for all other containers.
* Starting with Oracle Database 12c Release 1 (12.1.0.2), you can issue the CREATE SPFILE statement when the current container is a PDB. In this case, the *database stores the PDB's initialization parameter values internally, rather than in a file*. Therefore, you cannot specify an spfile\_name. The values that you set for initialization parameters in the PDB are persistent and override any values set for those parameters in the root.

# DB Names

<https://docs.oracle.com/database/121/ADMIN/cdb_pdb_admin.htm#ADMIN13663>

The global name of the root is defined by the DB\_NAME and DB\_DOMAIN initialization parameters. The global database name of a PDB is defined by the PDB name and the DB\_DOMAIN initialization parameter. The global database name of each PDB must be unique within the domain.

Testing allows up to 12 characters for DB\_NAME of the CDB.

# Administering a CDB with SQL\*Plus

<http://docs.oracle.com/database/122/ADMIN/administering-a-cdb-with-sql-plus.htm#ADMIN13606>

## About the Current Container

The data dictionary in each container in a CDB is separate, and the current container is the container whose data dictionary is used for name resolution and privilege authorization. Each container has a unique ID and name in a CDB.

The following rules apply to the current container in a CDB:

* It can be the root, cdb$root, only for common users
* It can be a particular PDB for common users and local users
* It must be the CDB root or an application root when a SQL statement includes CONTAINER=ALL
* When a SQL statement includes CONTAINER=ALL in the CDB root, it affects all containers, including all PDBs, application roots and application PDBs.
* Only a common user or application common user with the SET CONTAINER privilege can run a SQL statement includes CONTAINER=ALL.

## About Admin Tasks in a CDB

Common users perform admin tasks for a CDB. A common user has a single identity and can login to the CDB root, any application root, PDB or application PDB in which it has privileges. Some tasks can only be performed by a common user.

|  |  |
| --- | --- |
| Task | Description |
| Starting up a CDB instance | The user must be a common user. When you open a CDB, the root is open but its other containers are mounted. Use ALTER PLUGGABLE DATABASE to open them. |
| Managing processes | A CDB has one set of background processes shared by the CDB root and all containers. |
| Managing memory | A CDB has a single SGA and single PGA. The memory required is the sum of the requirements for all containers. |
| Managing security | You can create and drop common users and local users in a CDB. You can also grant/revoke privileges from these users. Grant the CDB\_DBA and PDB\_DBA roles to appropriate users. |
| Monitoring errors and alerts | A CDB has one alert log for the entire CDB. The name of the container is included in records in trace files, when appropriate. |
| Managing diagnostic data | In a CDB you can use Oracle's ADR. |
| Control files | A CDB has one control file. |
| Tablespaces | You can administer normal and temp TSs for the CDB root and containers. You can specify a default TS for the root. The root has its own set of Oracle-supplied tablespaces, such as SYSTEM, and other containers have their own. |
| Online redo logs and archived redo logs | A CDB has one online redo log and one set of archived redo logs. |
| Datafiles and tempfiles | The CDB root has its own data files, and other containers have their own. In a CDB you manage files in mostly the same way, except:   * You can limit the amount of storage used by a container with the STORAGE clause. * There is a default temp TS for the CDB root and individual containers. |
| Undo | A CDB can run in local or shared mode. |
| Moving data between containers | You can move data the same way you would between non-CDBs. |
| Data Guard | DG can configure a standby version of a CDB. It operates on the entire CDB, not on individual containers. |
| Dropping a DB | When you drop a CDB, you also drop all of its containers. You can drop individual DBs with DROP PLUGGABLE DATABASE. |

## Manageability Features in a CDB

|  |  |  |
| --- | --- | --- |
| Feature | Data Location | Data Visibility |
| ASH | Most stored in memory. A small amount in the CDB root. | If your current container is root, you can see everything. If not, you can only see for your container. |
| Alerts | Thresholds for a PDB are stored in the PDB. Violations are enqueued in the CDB root. Thresholds are included when PDB unplugged. | See above. |
| AWR | Reports generated in the CDB root pertain to the entire CDB. Else, only applies to current container. Data stored in current container and included when PDB unplugged. | See above. |
| SQL Tuning Sets | STS can be stored in the CDB root or any PDB. If in the root, can load statements from any PDB. When PDB unplugged, only STS stored in it are included. | Can only see STS data in your current container (even from root). |
| SQL Tuning Advisor | Auto jobs stored in CDB root. From root you can run against statements from any PDB. When tuned, it's tuned for every container that runs it.  When run from a PDB, results stored in PDB and tuned only for that PDB. | Auto job results only seen from root. When run manually from PDB, must be current container to see results. |
| ADDM |  |  |
| Auto Optimizer Stats Collection |  |  |

## Managing Database Objects in a CDB

In a CDB different containers can contain different objects. The root and PDBs contain schemas, which contain objects. The root and PDBs can also contain nonschema objects, such as users, roles, tablespaces and directories.

In a CDB the root contains Oracle-supplied schemas and DB objects. Oracle-supplied common users, such as SYS and SYSTEM, own these schemas and common objects. They can also own local objects in both the root and PDBs.

Oracle recommends that, in the root, schemas owned by user-created common users contain only database triggers and the objects used in their definitions.

Name resolution in a CDB is similar to in a non-CDB, except that names are resolved in the context of the dictionary of the user's current container.

## Executing DDL Statements in a CDB

In a CDB some DDL statements can apply to all containers or to the current container only. To specify which are affected, use the CONTAINER clause:

CONTAINER = { CURRENT | ALL }

The following settings are possible:

* CURRENT means that the statement applies only to the current container.
* ALL means the statement applies to all containers in the CDB, including the root and all the PDBs.

The CONTAINER clause can only be used the following DDL statements:

|  |  |  |
| --- | --- | --- |
| Statement | CURRENT | ALL |
| CREATE USER | Creates local user in current PDB. | Creates a common user. |
| ALTER USER | Alters a local user. | Alters a common user. |
| CREATE ROLE | Creates a local role. | Creates a common role. |
| GRANT | Grants privilege in local container to a local user/role or common user.  SET CONTAINER can be granted to a common user in the current PDB. | Grants system/object privilege on common object to a common user/role. The privilege is granted to the user/role across the entire CDB. |
| REVOKE | Revokes privilege in local container from a local user/role or common user.  This can only revoke a privilege granted with CURRENT specified from the specified user/role in local container. Does not affect privileges granted with ALL.  SET CONTAINER can be revoked from a common user in the current PDB. | Revokes system/object privilege on a common objects from common user/role. The privilege is revoked from the user/role across the entire CDB.  Can revoke only a privilege granted with ALL from the specified common user/role. Does not affect privileges granted with CURRENT. Any privileges granted locally that depend on the privilege granted commonly are also revoked. |

All other DDL statements apply to the current container only. In addition to the usual rules for user, role and profile names, the following best practices apply in a CDB:

* It is best practice for common user, role and profile names to start with a prefix to avoid naming conflicts between common and local items. You specify this prefix with the COMMON\_USER\_PREFIX initialization parameter in the CDB root. By default the prefix is C##.
* Common user, role and profile names must consists only of ASCII characters.
* Local user, role and profile names cannot start with the prefix specified in COMMON\_USER\_PREFIX, "C##" or "c##".

# Administering PDBs with SQL\*Plus

<http://docs.oracle.com/database/122/ADMIN/administering-pdbs-with-sql-plus.htm#ADMIN13663>