# DML and DCL –Data Control Language

# Data Control Language

- ✓ **The Data Control Language (DCL) is a subset of the Structured Query Lanaguge (SQL) that is used to control data.**
- ✓ **It is used by the database administrators or owners of database objects to control the user access to various database objects.**
- ✓ **They are used to enforce database security in a multiple user database environment**.

## DCL Commands

- ✓ Grant – To give privileges
- ✓ Revoke – To remove the previously given privileges

# Privileges

A database user cannot do anything with the database unless he has the basic set of privilages. A user should have certain **System Privileges** so that they can

- Connect to the database (CREATE SESSION)
- Create objects (CREATE TABLE, CREATE INDEX)
- Modify or delete objects

Once the users start creating objects, only the user who has created that object can access it. Other users will not be able to do anything with those objects. So the object owner can give other users accesses or **Object Privileges** to

- Modify the objects (ALTER, DELETE, UPDATE, INSERT)
- Query tables views etc (SELECT)
- Execute procedures functions etc(EXECUTE)

# Object Privileges

Database Users need privileges to access, modify , execute or delete any database objects (Tables, views etc).

Following are some of the available privileges on each database objects.

- ✓ALL
- ✓ALTER
- ✓DELETE
- ✓INSERT
- ✓SELECT
- ✓UPDATE
- ✓EXECUTE

Owner of the data base objects can issue privileges to the other users or can remove the privileges using **GRANT** and **REVOKE** statements in Oracle.

# System Privileges

System privileges are given to users to perform a particular action, or to perform an action on any schema objects of a particular type. They allow users to perform certain functions that deal with managing the database and the server

Following are some of the system privileges.

- ✓CREATE USER
- ✓CREATE SESSION
- ✓CREATE ANY TABLE
- ✓DROP ANY TABLE
- ✓ALTER DATABASE
- ✓ALTER ANY TABLE

Sytem privileges also can be issued or removed using **GRANT** and **REVOKE** statements

# Grant

## Granting Object Privileges

**Syntax:-**

```
GRANT privilege_names ON object_name TO user_name [WITH
                                                GRANT OPTION]
```

## Granting System Privileges

**Syntax:-**

```
GRANT privilege_names TO user_name [WITH ADMIN OPTION]
```

# Grant - Example

## System Privileges

**GRANT CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE TO user_name1;**

**GRANT CREATE ANY INDEX TO user_name1;**

**GRANT CREATE ANY INDEX TO user_name2 WITH ADMIN OPTION;**

**GRANT CREATE SESSION TO user_name2**

## Object Privileges

**GRANT SELECT, INSERT, UPDATE, DELETE ON product_table TO user2**

**GRANT ALL ON suppliers_table TO user2;**
**note: - ALL implies ALL privileges**

**GRANT SELECT ON suppliers_table TO public;**

**GRANT EXECUTE ON Function_find_product TO user2;**

# Revoke

## Revoking Object Privileges

**Syntax:-**

```
REVOKE privilege_names ON object_name FROM user_name
```

## Revoking System Privileges

**Syntax:-**

```
REVOKE privilege_names FROM user_name
```

# Revoke - Examples

## System Privileges

**REVOKE CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE FROM user_name1;**

**REVOKE CREATE ANY INDEX FROM user_name1;**

**REVOKE ALL FROM user_name2;**
note: - ALL implies ALL privileges

**REVOKE CREATE SESSION FROM user_name2**

## Object Privileges

**REVOKE SELECT, INSERT, UPDATE, DELETE ON product_table FROM user2**

**REVOKE ALL ON suppliers_table FROM user2;**
note: - ALL implies ALL privileges

**REVOKE SELECT ON suppliers_table FROM public;**

**REVOKE EXECUTE ON Function_find_product FROM user2;**

# Roles

**A role is a set or group of privileges that can be granted to users or another role. Creating a role will help in reducing efforts to give grants to each user.**

**Syntax:-**

```
CREATE ROLE role_name  [ NOT IDENTIFIED | IDENTIFIED
{BY password | USING [schema.] package | EXTERNALLY | GLOBALLY }] ;
```

**role_name** is the name of the new role that you are creating. This is how you will refer to the grouping of privileges.

**NOT IDENTIFIED** ==> Specifies that the role is immediately enabled. No password is required to enable the role.

**IDENTIFIED** ==> Specifies that a user must be authorized by a specified method before the role is enabled. Authorization will be done using password that we are providing.

**EXTERNALLY and GLOBALLY** ==> Used along with identified clause to specify whether role can applicable to external specified users or all users

***Note: - If both NOT IDENTIFIED and IDENTIFIED are omitted in the CREATE ROLE statement, the role will be created as a NOT IDENTIFIED role***.

# Grant or Revoke Privileges to Roles

Once a role is created in Oracle, the next step is to grant privileges to that role.

**Syntax:-**

```
GRANT privileges ON object_name TO role_name;
```

The privileges that are given to a role can also be revoked.

**Syntax:-**

```
REVOKE privileges ON object_name FROM role_name;
```

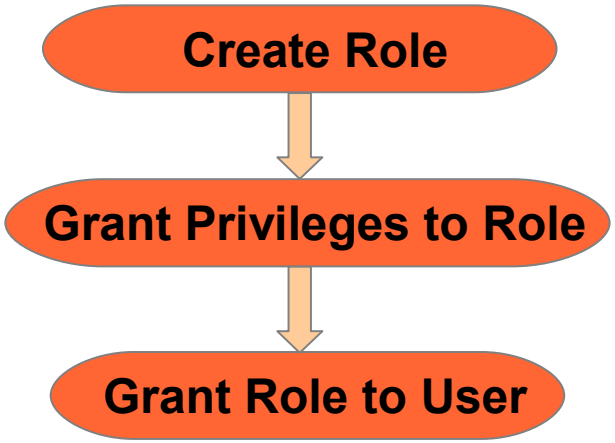
# Grant Role to User

After creating the role and assigning the privileges to the role, the final step is to grant the role to specific users.

**Syntax:**

```
GRANT role_name TO user_name;
```

**Steps :-**



# Roles - Examples

```
CREATE ROLE test_role1;

CREATE ROLE test_role2 IDENTIFIED BY test123;;

GRANT select, insert, update, delete ON products_table TO test_role1;

GRANT ALL ON products_table TO test_role2;

GRANT test_role1 TO user_name2;

REVOKE delete ON products_table FROM test_role1;

REVOKE all ON products_table FROM test_role2;
```