UNIT 3:
TRANSACTION PROCESSING AND DATABASE SECURITY

---

1. DEFINE FOLLOWING TERMS.

Transaction

- Transaction is a set of operations which are all logically related.
- Transaction is a single logical unit of work formed by a set of operations

Transaction state

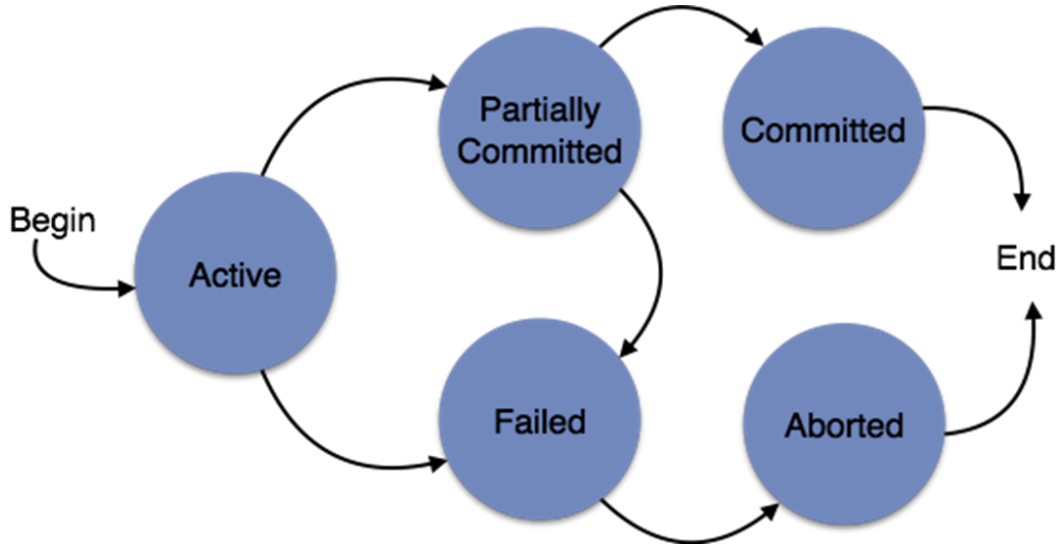
- A transaction goes through many different states throughout its life cycle. These states are called as transaction states.

2. EXPLAIN : TRANSACTION PROCESSING IN DBMS.

- Peter Chen developed ERDs in the 1970s .
- Peter Chen was a computer scientist who worked on improving database design.
- He published his proposal for entity relationship modeling in a 1976 paper titled "The Entity-Relationship Model: Toward a Unified View of Data".
- His entity relationship model was a way to visualize a database that unified other existing models into a single understanding.

3. EXPLAIN TRANSACTION STATES WITH STATE TRANSITION DIAGRAM

- A transaction is the sequence of one or more SQL statements that are combined together to form a single unit of work.
- A transaction goes through many different states throughout its life cycle. These states are called as transaction states.
- Transaction states are as follows-
  - Active state
  - Partially committed state
  - Committed state
  - Failed state
  - Aborted state
  - Terminated state



Active State:

- This is the first state in the life cycle of a transaction.
- A transaction is called in an **active state** as long as its instructions are getting executed.
- All the changes made by the transaction now are stored in the buffer in main memory.

Partially Committed State:

- After the last instruction of transaction has executed, it enters into a **partially committed state**.
- After entering this state, the transaction is considered to be partially committed.
- It is not considered fully committed because all the changes made by the transaction are still stored in the buffer in main memory.

Committed State:

- After all the changes made by the transaction have been successfully stored into the database, it enters into a committed state.
- Now, the transaction is considered to be fully committed.

Failed State:

- A transaction goes to a Failed state if it is determined that it can no longer proceed with its normal execution.

Aborted State:

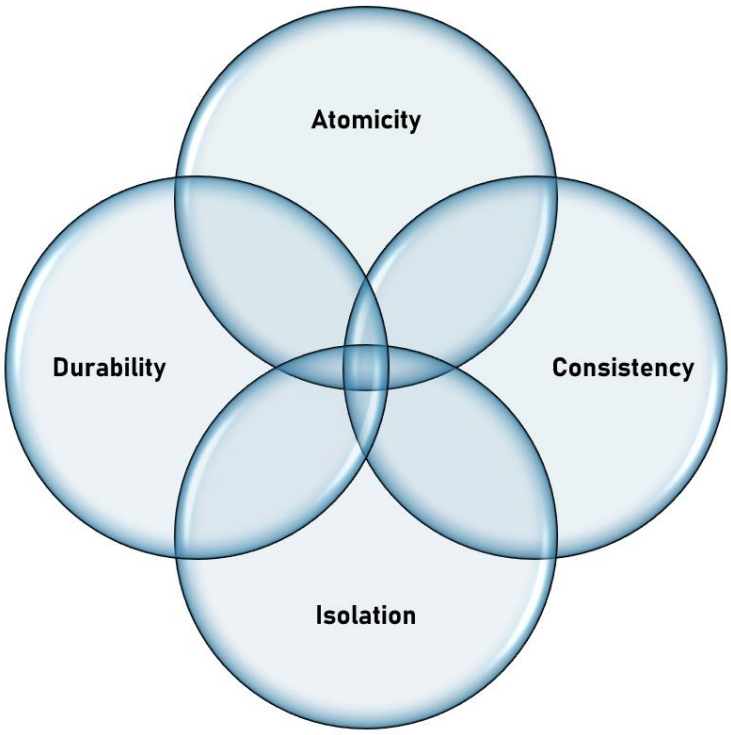
- After the transaction has failed and entered into a failed state, all the changes made by it have to be undone.
- To undo the changes made by the transaction, it becomes necessary to roll back the transaction.
- After the transaction has rolled back completely, it enters into an **aborted state**.

Terminated State:

- This is the last state in the life cycle of a transaction.
- After entering the committed state or aborted state, the transaction finally enters into a **terminated state** where its life cycle finally comes to an end.

4. WRITE A NOTE ON TRNSACTION PROPERTIES/ ACID PROPERTIES

- It is important to ensure that the database remains consistent before and after the transaction.
- To ensure the consistency of database, certain properties are followed by all the transactions occurring in the system.
- These properties are called as **ACID Properties** of a transaction.
- ACID property also called as Acidity of a transaction.



Atomicity:

- This property ensures that either the transaction occurs completely or it does not occur at all.
- In other words, it ensures that no transaction occurs partially.
- That is why, it is also referred to as “All or nothing rule“.
- It is the responsibility of Transaction Control Manager to ensure atomicity of the transactions.

Consistency:

- This property ensures that integrity constraints are maintained.
- In other words, it ensures that the database remains consistent before and after the transaction.
- It is the responsibility of DBMS and application programmer to ensure consistency of the database.
- If the transaction fails the database must be return to the state it was in prior to the execution of the failed transaction.
- But if the transaction commits the database must reflect the new values in consistence state.

Isolation:

- The isolation property of a transaction means that the data used during the execution of a transaction can not be used by a second transaction until the first transaction completes its execution.
- This property isolates the transaction from one another. In other word if the transaction T1 is being executed using data item X. The data item can not be accessed by any other transaction (T2----TN).
- Until transaction T1 ends the transaction must act as if it is only running transaction against the database.

Durability:

- This property ensures that all the changes made by a transaction after its successful execution are written successfully to the disk.
- It also ensures that these changes exist permanently and are never lost even if there occurs a failure of any kind.

- It is the responsibility of recovery manager to ensure durability in the database.

5. WRITE A NOTE ON DATA SECURITY.

- Because of the importance of data and information of organization database security is an important issue in database management.
- The data stored in DBMS is vital to the organization and is consider as a corporate assets.
- Thus database represent as an essential recourse of an organization, which should be properly secured.
- Because of the use of distributed database with client/server architecture, the database environment becomes more complex.
- Managing database security has become more difficult and time consuming. Therefore, it important for the DBA to develop overall policies, procedures and appropriate controls to protect the database.

➢ **Authorization and Authentication :**

•Authorisation:

- Authorization is the process of a grating of right or privileges to the user to have a limited access to a system or objects of the system.
- It is an administrative policy of the organization, express as a set of rules that can be used to determine which user has what type of access to which portion of database.

•Authentication:

- Authentication is a mechanism that determines whether a user is who he or she claims to be.
- In other words, an authentication checks whether a user operating upon the database is allow to doing so or not. It verifies the identity of the user.

- The simplest form of authentication is a simplest consists of a secret password which must be presented when a connection is open to database.

✓ Authorization and Authentication controls can be built into the software.

✓ Authorization rules are incorporated in DBMSs that restrict access to data and also restrict the action that people may take when they access data.

✓ For example, a user using a particular password may be authorized to read any record from database but cannot necessarily modify any of those records.

✓ For this reason authentication controls are sometimes refer to as access controls. Following two types of access control techniques are used in database security system:

- Discretionary Access Control.
- Mandatory Access Control.

The DBMS provides these access control mechanism to allow the users to access only those data for which he or she has been authorized. Most DBMS supports either discretionary security scheme or mandatory security scheme or both.

6. WRITE A NOTE ON DAC

- DAC is based on the concept of privileges and mechanism for giving such privileges to user.
- It grant the privileges to user on different object, including capability to access specific data file, records or fields in specified mode, such as, read, insert, delete or update or combination.
- A user who creates a database object such as a table or view automatically gets all applicable privileges on that object.
- The DBMS keep track of how these privileges are granted to other users and it is very flexible. However, it has certain weakness.

For Example, An unauthorized user into disclose of sensitive data.

1. Granting / Revoking Privileges
2. Audit Trail

➢ **Granting / Revoking Privileges :**

- **Users:**
  - One of the most important tasks of an administrator is controlling access to the database.
  - Oracle relies on a mechanism that allows you to register a person, called a user.
  - Each registered user has an access password, which must be provided in various situations.
  - Each user is then assigned individual privileges or roles.

- **Creating a New User:**
  - The CREATE USER command is responsible for the creating of new user.

- It creates a user or an account through which you can connect to the database, and establish the means by which Oracle will allow the user access.

- **Syntax:**

```
CREATE USER user IDENTIFIED {BY password  | EXTERNALLY
  | GLOBALLY AS 'CN = user} [DEFAULT TABLESPACE tablespace
  | TEMPORARY TABLESPACE tablespace
  | QUOTA {integer [K | M] | UNLIMITED} ON tablespace
  [QUOTA {integer [K | M]  |UNLIMITED} ON tablespace]……
  | PROFILE profile
  | PASSWORD EXPIRE
  | ACCOUNT {LOCK | UNLOCK}……
```

- **Common Syntax:**

  - CREATE USER <USER_NAME> IDENTIFIED BY <PASSWORD>;
  - To assign privileges, you must connect to the administrator using the SYSTEM name and the MANAGER password.
  - To connect to the database as a new user, you must use the CONNECT command with the following format: -

```
SQL> CONNECT <USER_NAME> / <PASSWORD>;
     SQL> CONNECT SYSTEM / MANAGER;
```

  - Now we see the CREATE USER command.
  - Both the name and password can be up to 30 characters long, without spaces or commas.
  - If you enter a user name that has already been registered, the system will issue an error message and will not allow you to register the new user.

- When the user is created, you have to assign him / her privileges. First, let's create a user called MCA and BCA, with the passwords MCA and BCA respectively.

```
SQL>    CREATE USER MCA IDENTIFIED BY MCA;
        User created
SQL>    CREATE USER BCA IDENTIFIED BY BCA;
        User created
```

- Before assigning the privileges, we will try to connect as MCA: -

```
SQL> CONNECT MCA / MCA;

        ERROR:
        ORA-01045: the user MCA needs the CREATE SESSION privilege;
        logon denied
```

There is an error in the above example because the user doesn't have any privileges assigned yet. Let's go back and connect to the system using the GRANT command.

- **Grant Command:**

  - The GRANT command has two distinct forms: - one to distribute system privileges and the other to distribute object privileges. Only the DBA can use the GRANT command to distribute system privileges.

  - **Syntax:**

```
GRANT SYSTEM PRIVILEGE / ROLE TO USER / ROLE /
PUBLIC
        WITH ADMIN OPTION;
```

ARGUMENTS	DESCRIPTION
PRIVILEGE	The name of the privilege to be assigned
USER / ROLE	The name of the user or role that is received the privilege.
WITH ADMIN OPTION	Allows a user or role that receives the privileges to assign it to other users, change it, or even delete it.

- Let's connect with SYSTEM user and assign the RESOURCE role, which allows access to the database and the creation of tables, sequences, procedures, triggers, indexes, and clusters:

```
SQL> CONNECT SYSTEM / MANAGER;

SQL>        GRANT RESOURCE TO MCA;
        Grant operation successfully.
```

From now on the user can perform any operation pertaining to the role received, such as, in the following example, the creation of a table:

```
SQL>        CONNECT MCA / MCA;
        Connected
SQL> CREATE TABLE EMP
        (EMP_NO NUMBER (4), EMP_NAME VARHCAR2 (20));
        Table created.
```

The next example shows, through SQL * PLUS, the importance of understanding to concept of the user, role, and privilege.

```
SQL>        CONNECT MCA / MCA;
        Connected
SQL> SELECT * FROM ITEM;
        Select * from item;
                *
```

ERROR on row 1:
ORA-00942: the table or view does not exist.

This occurs because the ITEM table, which was created by Scott, did not assign any access privileges to other users. Therefore, each user is restricted to his / her own activities inside the database.

- **Deleting a User:**
  - You can remove a database user with the DROP USER command.
  - This command removes both the user and all the objects contained in this user's schema.
  - You will need to specify the CASCADE clause of the command.
  - Oracle also removes all the referential integrity associated to the objects of the removed user.
  - **Syntax:**

    DROP USER <USER_NAME> [<CASCADE>];
    In the next example user MCA is removed:

```
SQL>        CONNECT SYSTEM / MANAGER
        Connected.
SQL> DROP USER MCA;
        Users eliminated.
SQL>  SELECT * FROM ALL_USERS;
```

USER NAME	USER_ID	CREATED
SYS	0	03/01/99
SYSTEM	5	03/01/99
OUTLN	11	03/01/99
DBSNMP	20	03/01/99
MTSSYS	28	03/01/99
AURORAORBUNAUTHENTICCATED	25	03/01/99

SCOTT	26	03/01/99
DEMO	27	03/01/99
ORDSYS	30	03/01/99
ORDPLUGINS	31	03/01/99
MDSYS	32	03/01/99
CTXSYS	35	03/01/99

13 rows selected.

- **Privileges:**

A privilege is an authorization given to the user to access and manipulate a database object in a certain way. For example, one user can be assigned the privilege to select tables but not change them, while another user can read, update, and even change the structure of tables and other objects. As all database manipulation is done through SQL commands, a privilege grants to a user the right to use these commands. There are two types of privileges System Privileges and Object Privileges.

1. System Privileges:

A system privilege is the right or permission to execute certain database actions in a specific type of database object. These are more than 70 types of privileges associated to the name of the action it executes. For example, a privilege called ALTER TABLE grants a user the right to modify a table. The following is a list of the some system privileges:

SYSTEM PRIVILEGE	TYPE OF ACTION
CREATE	Create an object in a user's own schema.
CREATE ANY	**Create an object in another user's schema.**
CREATE SESSION	Connect to the database.
DROP	Drop an object in user's own schema
DROP ANY	Drop an object in another

	user's schema
ALTER SYSTEM	Manipulate an instance
ALTER DATABASE	Manipulate the database
ALTER USER	Change user's role or password
ALTER/ CREATE/ DROP/ MANAGE TABLESPACE	Manipulate a tablespace

2. Object Privileges:

An object privileges is the right to perform certain actions in a specific object, such as the right to include a row in a table. These privileges do not apply to all the objects of a database links, and clusters do not have any object privileges. The following is a list of the privileges:

PRIVILEGE	OBJECT
ALTER	Tables and sequences
DELETE	Tables and views
EXECUTE	Procedures
INDEX	Tables
INSERT	Tables and views
REFERENCE	Tables
SELECT	Tables, views, and sequences
UPDATE	Tables and views

When a user creates an object, such as a table, only the user can view it. The user / owner must grant a privilege or a role to access the table. A privilege can be granted with the GRANT command.

- **Assigning Object Privileges To User:**

A variation of the GRANT command allows you to assign object privileges to a user or role. The privilege assigned can be SELECT, ALTER, DELETE, EXECUTE,

INSERT, INDEX, REFERENCE, and UPDATE. The following objects can be assigned privileges: - tables, views, sequences, snapshots, and synonyms. Any user who has authority can grant privileges on tables.

- **Syntax:**

```
GRANT
        ALL
        SELECT
        INSERT
        DELETE
        INDEX
        ALTER
        UPDATE [ (column_name) ] ON table_name / view_name
TO user / PUBLIC
WITH ADMIN OPTION;
```

ARGUMENT	DESCRIPTION
SELECT	To select data in a table or view.
INSERT	To insert rows in a table or view.
DELETE	To eliminate rows in a table or view
UPDATE	To update a table and, optionally, update only the specified columns
INDEX	To create or eliminate the indexes of a table
ALTER	To modify a table
ALL	To perform all the above privileges
Table_name	Names of existing tables (including the qualifier) in the database
View_name	Names of existing views (including the qualifier) in the database
Column_name	This is optional and determines the column of tables or views specified in the ON clause. The

	names of the columns must not be qualified.
USER	Name of the user to whom the privilege is assigned
PUBLIC	Means that all the current and new user have the privileges specified for the table or view
WITH ADMIN OPTION	Allows the user or role that receives the privilege to grant it to other users, modify it, or even delete it.

The next example shows user SCOTT granting the SELECT privilege so user MCA can access the EMP table:

```
SQL> CONNECT SCOTT / TIGER;
     Connected.

SQL>      GRANT SELECT ON EMP TO MCA;
     GRANT operation successful
```

Now we connect to MCA and access the table:

```
SQL>      SELECT ENAME FROM EMP;
     Select ename from emp
          *
     ERROR om row 1:
     ORA-00942: table or view does not exist
```

An error occurs. What is the problem, since the privilege was already assigned? When a user makes a SELECT in another user's table, he / she must provide the schema of the table or object. In other words, he / she must identify the table and an owner, if needed. In the following example, you need to provide the SCOTT.EMP schema to access the EMP table:

```
SQL> SELECT ENAME FROM SCOTT.EMP;
```

```
ENAME
----------
SMITH
ALLEN
WARD
JONES
```

If you try to access another one of Scott's tables using this schema, but without assigning an object privilege for the user, you will have the same problem again. Connected as SCOTT, we will assign to MCA the UPDATE privilege only for the DNAME column of the DEPT table:

```
SQL>  GRANT UPDATE (DNAME) ON DEPT TO MCA;
```

Then, connected as MCA, we will update two columns of the DEPT table:

```
SQL> UPDATE SCOTT.DEPT
     SET DNAME='SALES FORCE'
     WHERE DEPTNO=30;

     1 row updated.

SQL> UPDATE SCOTT.DEPT
     SET LOCATION =' MUMBAI'
     WHERE DEPTNO=30;
     Update scott.dept
            *
     ERROR on row 1:
     ORA-01031: insufficient privileges
```

Note that it was not possible to update the column without the proper privileges. In the previous example, user MCA was not assigned the UPDATE privilege for all of the columns but only for DNAME column. Now we will try to insert a new row in the SCOTT.DEPT table:

```
SQL> INSERT INTO SCOTT.DEPT VALUES
     (50, 'MARKETING', 'DELHI');
     Insert into scott.dept values (50, 'MEDIA', 'MIAMI')
                    *
     ERROR on row 1:
     ORA-01031: insufficient privileges
```

Note that MCA is prevented from inserting the table because it has not been assigned the proper privileges.

- **Viewing Users & Privileges:**

Connected as SCOTT, we will try to create a new user:

```
SQL> CREATE USER BCA1 IDENTIFIED BY BCA1;
     Create user BCA1 identified by BCA1
          *
     ERROR on row 1:
     ORA-01031: insufficient privilege
```

Remember that high-level privileges are necessary to create a user. Connected as SYSTEM / MANAGER, we will create this user and view a list of all other user. You can use one of the tables of the data dictionary to check the database users. The dictionary view that maintains this list is ALL_USERS.

```
SQL> CONNECT SYSTEM / MANAGER
     Connected.

SQL> CREATE USER BCA1 IDENTIFIED BY BCA;
     User created.

SQL> SELECT * FROM ALL_USERS;
```

- **Revoking a System Privilege:**

In addition to granting a privilege, you can also revoke it, by using the REVOKE command. To be revoked, a system privilege must have been granted with the ADMIN OPTION option.

```
SQL>        REVOKE CREATE SESSION FROM MCA;
        Revoke successful.
```

To check the functionality of the command, connect as MCA.

```
SQL> CONNECT MCA / MCA;
        ERROR:
                *
        ORA-01045: the user Arnold needs the privilege CREATE
        SESSION;
                        logon denied
        Notice: you are not connected to Oracle anymore.
```

Oracle checked MCA's privilege and noted he did not have the CREATE SESSION privilege any more, thus preventing the connection to the database.

- **Revoking An Object Privilege:**

Now we show you how to revoke an object privilege that was granted to a user. In an example we have seen before, user SCOTT granted to MCA the SELECT privilege for the EMP table. Now we will revoke this privilege using SQL command. To do that, you must connect as a user who has the authority to grant privileges. A user cannot grant or revoke privilege to him self or her self. In the following example, we have to connect as user SCOTT. Then, we use the REVOKE command, specifying the privilege, the object, and the user that will be revoked:

```
SQL>        CONNECT SCOTT / TIGER;
```

```
        Connected.
SQL>    REVOKE SELECT ON EMP FROM MCA;
        Revoke successful.
```

To verify that the revoke operation worked, we connect as MCA and try to access the SCOTT.EMP table.

```
SQL> CONNECT MCA / MCA
     Connected.
SQL>    SELECT * FROM SCOTT.EMP;
     Select * from scott.dept
            *
     ERROR on row 1:
     ORA-01031: insufficient privileges
```

Because MCA cannot view the EMP table anymore

- **Roles:**

A role is a group of privileges assigned to a name. Instead of granting eight privileges to a user, you can create a role that is assigned those eight privileges, and then assign that role to the user. The use of roles simplifies the administration of users.

Oracle has several predefined roles, such as CONNECT, RESOURCE, DBA, EXP_FULL_DATABASE, AND IMP_FULL_DATABASE. In addition to these roles, users can create other ones that meet the needs of their applications.

They are mainly three types of oracles predefined Roles available for any user:

1. Connect Role.
2. Resource Role.
3. DBA Role.

1. Connect Role:

It allows access right to the database object. i.e. if user is created with connect role then we can select, insert, update and deletes the table rows. User cannot create the object as well as destroy the object. Create operation on object is not allowed in connect role.

2. Resource Role:

The resource roles can provide the creation of object such as tables, sequence, procedures functions, indexes, triggers etc. It also allows access to the database objects. More sophisticated and regular user of database gets the resource role. The resource role allows DDL and DML command operations on the object.

3. DBA Role:

The DBA has all system privileges including unlimited space quota. The DBA user can export and import the database. The DBA has right to create and drop any users.

The following is a list of some Oracle's users, passwords, and roles:

USER NAME	PASSWORD	ROLE
SCOTT	TIGER	CONNECT & RESOURCE
SYSTEM	MANAGER	DBA
SYS	SYS1	CONNECT, RESOURCE, DBA, EXP_FULL_DATABASE & IMP_FULL_DATABASE
DEMO	DEMO	CONNECT & RESOURCE

To create a role, the user must have the CREATE ROLE privilege. When created, a role can be assigned the option ADMIN OPTION, which allows the user to grant it to other roles or users, revoke the granted role, change the access to it, and remove it.

- **Creating a Role:**

The creation of a new role follows the same rules that apply to the creation of other objects. When you create a new role, you can assign it some privileges or other roles that were previously created. The SQL command used to create a role is CREATE ROLE:

- **Syntax:**

CREATE ROLE role [NOT IDENTIFIED
| IDENTIFIED {BY password | EXTERNALLY | GLOBALY}]

ARGUMENTS	DESCRIPTION
Role	Name of the role to be created. Oracle recommends the role contain at least one single-byte character, independent of the database's set of characters, which may or may not contain multiple-byte characters.
NOT IDENTIFIED	Indicates that this role is authorized by the database and that no password is required to enable it.

IDENTIFIED	Indicates that the user must be authorized by the specified method before the role can be enabled with the SET ROLE command
BY password	The user must specify the password for Oracle to enable the role. It can contain only single-byte characters from the database's set of characters, regardless of whether this set of characters contains multiple-byte characters.
EXTERNALLY	
GLOBALLY AS external name	Indicates that the user must be authorized by an external service (such as an operating system or third party service) before the role can be enabled. Depending on the operating system, the user must specify a password before the role can be enabled.
	Indicates that the user must have authorization from Oracle Security Service to use the role before it can be enabled with the SET ROLE command, or at the login.

If both the NOT IDENTIFIED and the IDENTIFIED options are omitted, the standard role will be NOT IDENTIFIED.

In the following example, we connect as SYSTEM / MANAGER to create a role called ROLE1:

```
SQL>        CONNECT SYSTEM / MANAGER;
SQL> CREATE ROLE ROLE1;
       Role created.
```

- **Granting Privileges And Roles to a Role:**

After creating a role, you can grant it privileges using the GRANT command. The privileges can be specific to that role or from other roles. If a role is granted a role, the newly created role assumes the privileges of those roles it may receive.

  - **Syntax:**

    GRANT role / name_of_privilege TO user / role / PUBLIC WITH ADMIN OPTION

The next example grants the CONNECT and RESOURCE role to ROLE1:

```
SQL>    GRANT CONNECT, RESOURCE TO ROLE1;
        Grant operation successful
```

- **Granting A Role To a User:**

Now we assign the ROLE1 role to user MCA:

```
SQL>    GRANT ROLE1 TO MCA;
        Grant operation successful
```

Next we will connect as the new user.

```
SQL> CONNECT MCA / MCA;
        Connected.
```

- **Viewing The Role Of a User:**

Oracle has some special views that control the privileges and roles. One of these views is called USER_ROLE_PRIVS. It displays the roles assigned to the current user:

```
SQL>    SELECT * FROM USER_ROLE_PRIVS;
        USERNAME GRANTED_ROLE ADM   DFF  OS_
        ---------- ---------- -------  ------ ------
        MCA        ROLE1      NO       YES  NO
```

- **Deleting a Role:**

The command responsible for the deletion of a role is DROP ROLE. The user must have the system privilege DROP ANY ROLE to remove a role or the role must have been created with the ADMIN OPTION option.

- **Syntax:**

```
DROP ROLE <ROLE_NAME>;
```

The next example removes the ROLE1 role.

```
SQL>    CONNECT SYSTEM / MANAGER;
        Connected.
SQL>    DROP ROLE ROLE1;
        Role eliminated.
```

When we try to connect as user MCA, an error will occur, because it has no associated privilege or role:

```
SQL> CONNECT MCA / MCA;
     ERROR:
     ORA-01045: the user MCA needs the privilege CREATE SESSION;
     logon denied
     Notice: You are not connected to Oracle anymore.
```

- **Example of Creating Users and Assigning & Revoking The Privileges From Them & From The Object:**

STEP – 1: Connect with SYSTEM user.

```
SQL>  CONNECT SYSTEM / MANAGER;
```

STEP – 2: Create two users MCA and BCA respectively.

```
SQL>  CREATE USER MCA IDENTIFIED BY MCA;
SQL>  CREATE USER BCA IDENTIFIED BY BCA;
```

STEP –3: Grant CONNECT and RESOURCE roles to MCA and BCA users.

```
SQL> GRANT CONNECT, RESOURCE TO MCA;
SQL> GRANT CONNECT, RESOURCE TO BCA;
```

STEP – 4: Connect with MCA and create one table EMP with data.

```
SQL> CONNECT MCA / MCA;
SQL> CREATE TABLE EMP
     (ENO NUMBER (3), ENAME VARCHAR2 (20));
SQL> INSERT INTO EMP VALUES (&ENO, '&ENAME');
```

STEP – 5: Grant SELECT privilege on EMP table to BCA user.

```
SQL> GRANT SELECT ON EMP TO BCA;
```

STEP – 6: Connect with BCA user and display the content of EMP table.

```
SQL> CONNECT BCA / BCA;
SQL> SELECT * FROM MCA.EMP;
```

STEP – 7: Connect with MCA user and create one view named as V_EMP on EMP table, Grant SELECT and UPDATE privileges on V_EMP to BCA and Revoke SELECT privilege on EMP from BCA user.

```
SQL> CONNECT MCA / MCA;
SQL> REVOKE ALL ON EMP FROM BCA;
SQL> CREATE VIEW V_EMP AS SELECT * FROM EMP;
SQL> GRANT SELECT, UPDATE ON V_EMP TO BCA;
```

STEP – 8: Connect with BCA user and execute SELECT statement on EMP table and V_EMP view.

```
SQL> CONNECT BCA / BCA;
SQL> SELECT * FROM MCA.EMP;
     ERROR…
SQL> SELECT * FROM MCA.V_EMP;
```

STEP – 9: Connect with MCA user and assign SELECT privilege on EMP table and UPDATE privilege on ENAME column of EMP table to BCA.

```
SQL> CONNECT MCA / MCA;
SQL> GRANT SELECT, UPDATE (ENAME) ON EMP TO BCA;
```

STEP – 10: Connect with BCA user and execute SELECT, UPDATE statements on EMP table.

```
SQL> CONNECT BCA / BCA;
SQL> SELECT * FROM MCA.EMP;
SQL> UPDATE MCA.EMP SET ENAME = 'RAM'
     WHERE ENO = 100;
```

STEP – 11: Connect with SYSTEM user and give DBA rights to BCA user

```
SQL> CONNECT SYSTEM / MANAGER;
SQL> GRANT DBA TO BCA;
```

- **Example of User defined Role:**

```
SQL> CONNECT SYSTEM / MANAGER;
SQL> CREATE ROLE MYROLE;
SQL> GRANT SELECT ON EMP TO MYROLE;
SQL> GRANT ALL ON ITEM OT MYROLE;
SQL> GRANT DELETE ON STUDENT TO MYROLE;
SQL> GRANT EXECUTE ON POW TO MYROLE;
SQL> GRANT MYROLE TO MCA;
SQL> CONNECT MCA / MCA;
SQL> SELECT SYSTEM.POW (2,2) FROM DUAL;
SQL> CONNECT SYSTEM / MANAGER;
SQL> REVOKE MYROLE FROM MCA;
SQL> DROP ROLE MYROLE;
SQL> ALTER USER MCA IDETIFIED BY MASTER;
```

NOTE: - You can only change the password of the user, but you cannot change the user name with ALTER USER command.

7.WRITE A NOTE ON MAC

Mandatory Access Control (MAC) also called Security Scheme the based on system-wide policies that cannot be changed by individual users. It is used to enforced multilevel security by classifying the data and user into various security classes or levels and then implementing the appropriate security policy of the organization. Thus, in this schema each data object is labeled with a certain classification level and each user is given a certain clearance level. A given data object can then be accessed only by users with the appropriate clearance of a particular classification level. Thus, a MAC technique classifies data and users based on security classes such as (a) Top Secret (TS) (b) Secret (S) (c) Confidential (C) (d) Unclassified (U)

The DBMS determines whether a given user can read or write a given object based on certain rules that involve the security level of the object and the clearance of the user. The commonly used MAC technique for multilevel security is known as the **Bel- LaPadula** model. The Bel-LaPadula model is described in terms of **Subject** (Users, Accounts, Programs), **Objects** (Relations or Tables, Tuples, Attributes, Views, Operations) and the **level of clearance** for a particular user. This model classifies each subject and object into one of the security levels such as TS, S, C, U. The security classes in a system are organized according to a particular order, with a **most secure class or level** and a **least secure class or level**. This model enforces following two restrictions on data access based on the subject and object classification.

(a) Simple Security Property:

In this case, a subject S is not allowed read access to an object O unless classification of subject S is greater than or equal to classification of object O. In other word class (S) ≥ class(O).

(b) Star Security Property:

In this case, a subject S is not allowed to write an object O unless

classification of subject S is less than or equal to classification of an object O. In other word class (S) ≤ class (O).

A mandatory security scheme is hierarchical in nature and is rigid as compare to discretionary security scheme. So, there are some loop halls of discretionary access control mechanism which can be addressed by using mandatory access control.

8. WRITE A NOTE ON Encryption and Public Key Infrastructures

Encryption is the conversion of data into a form, called a **ciphertext**, which cannot be easily understood by unauthorized persons. It enhances security and privacy when access controls are bypassed, because in cases of data loss or theft, encrypted data cannot be easily understood by unauthorized persons.

With this background, we look at the standard definitions:

■ *Ciphertext*: Encrypted (enciphered) data.
■ *Plaintext (or cleartext)*: Intelligible data that has meaning and can be read or acted upon without the application of decryption.
■ *Encryption*: The process of transforming plaintext into ciphertext.
■ *Decryption*: The process of transforming ciphertext back into plaintext.

Encryption consists of applying an **encryption algorithm** to data using some prespecified **encryption key**. The resulting data has to be **decrypted** using a **decryption key** to recover the original data.

➢ The Data Encryption and Advanced Encryption Standards

The **Data Encryption Standard** (DES) is a system developed by the U.S. government for use by the general public. It has been widely accepted as a cryptographic standard both in the United States and abroad. DES can provide end-to-end encryption on the channel between sender *A* and receiver *B*. The DES algorithm is a careful and complex combination of two of the fundamental building blocks of encryption: substitution and permutation (transposition). After questioning the adequacy of DES, the NIST introduced the **Advanced Encryption Standard** (AES). AES introduces more possible keys, compared with DES, and thus takes a much longer time to crack.

- **Symmetric Key Algorithms**

A symmetric key is one key that is used for both encryption and decryption. By using a symmetric key, fast encryption and decryption is possible for routine use with sensitive data in the database. A message encrypted with a secret key can be decrypted only with the same secret key. Algorithms used for symmetric key encryption are called **secret-key algorithms**. Since secret-key algorithms are mostly used for encrypting the content of a message, they are also called **content-encryption algorithms**.

- **Public (Asymmetric) Key Encryption (PKI)**

In 1976, Diffie and Hellman proposed a new kind of cryptosystem, which they called **public key encryption**. Public key algorithms are based on mathematical functions rather than operations on bit patterns. They address one drawback of symmetric key encryption, namely that both sender and recipient must exchange the common key in a secure manner. In public key systems, two keys are used for encryption/decryption. The *public key* can be transmitted in a non-secure way, whereas the *private key* is not transmitted at all. The two keys used for public key encryption are referred to as the **public key** and the **private key**.

A public key encryption scheme, or *infrastructure*, has six ingredients:

1. Plaintext. This is the data or readable message that is fed into the algorithm as input.
2. Encryption algorithm. This algorithm performs various transformations on the plaintext.
3. and **4. Public and private keys.** These are a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the encryption algorithm depend on

the public or private key that is provided as input. For example, if a message is encrypted using the public key, it can only be decrypted using the private key.

5. Ciphertext. This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.

6. Decryption algorithm. This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

The essential steps are as follows:

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.

2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.

3. If a sender wishes to send a private message to a receiver, the sender encrypts the message using the receiver's public key.

4. When the receiver receives the message, he or she decrypts it using the receiver's private key. No other recipient can decrypt the message because only the receiver knows his or her private key.