6. DCL and TCL commands
Ex. 6.

In SQL, Data Control Language (DCL) commands are used to control the access and permissions for users on the database. These commands allow database administrators to manage the security of the database by granting or revoking privileges to users.

DCL commands are:

GRANT: This command is used to grant specific privileges to a user. For example, if you want to give a user permission to select from a table, you can use the following command:

```
GRANT SELECT ON table_name TO user_name;
```

REVOKE: This command is used to revoke privileges from a user. For example, if you want to remove a user's permission to select from a table, you can use the following command:

```
REVOKE SELECT ON table_name FROM user_name;
```

DENY: This command is used to deny specific privileges to a user. The difference between REVOKE and DENY is that REVOKE removes a privilege that was previously granted, while DENY explicitly denies a privilege that was never granted in the first place.

```
SQL> CONNECT username/password
```

```
SQL> GRANT SELECT ON customers TO scott;
```

Similarly, to revoke or deny a privilege, you can use the REVOKE or DENY command instead of GRANT. It is important to note that only users with appropriate privileges can execute DCL commands.

Transaction Control Language (TCL) commands in SQL are used to manage transactions in the database. These commands allow you to control the changes made to the data and ensure that they are committed or rolled back as necessary.

Here are some examples of how to use TCL commands in SQL:

COMMIT: The COMMIT command is used to permanently save changes made in the current transaction. For example, after inserting or updating data in a table, you can use COMMIT to make sure that the changes are saved:

```
--First Create required table


-- Create the table given below

SQL> CREATE TABLE customers (

       id NUMBER(5)  PRIMARY KEY,

      name VARCHAR2(50) NOT NULL,

     email VARCHAR2(100) UNIQUE);


SQL> INSERT INTO customers VALUES (5,'John', 'johnny@example.com');

SQL> COMMIT;
```

ROLLBACK: The ROLLBACK command is used to undo changes made in the current transaction. For example, if you inserted or updated data in a table but then decide that you want to undo those changes, you can use ROLLBACK to undo the changes:

```
 SQL> UPDATE customers

          SET email= 'john@example.com'

          WHERE id=5;
```

```
SQL> ROLLBACK;
```

```
SQL> Select * from customers;
```

SAVEPOINT: The SAVEPOINT command is used to create a point in the current transaction that you can return to later. For example, if you are making multiple changes to a table and want to be able to undo some of them but not all of them, you can use SAVEPOINT to mark a point where you want to be able to return to later:

```
SQL> SAVEPOINT before_update;
```

```
SQL> UPDATE customers
          SET email= 'john123@example.com'
          WHERE id=5;
```

```
SQL> COMMIT;
```

```
SQL> ROLLBACK TO SAVEPOINT before_update;
```

```
SQL> SELECT * FROM customers;
```

SET TRANSACTION is a SQL statement that allows you to set properties for a transaction, such as isolation level, read-only or read-write mode, and whether it can be rolled back or committed.

For example, you can use the SET TRANSACTION statement to set the isolation level of a transaction to "read committed" as follows:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

This will ensure that the transaction only sees committed data from other transactions and prevents unnecessary reads.