Transaction Control Language in SQL

Managing of Transactions in the database is done using TCL.

- **What is a Transaction?**

A transaction is a sequential group of queries, statements, or operations such as update, select, insert or delete to perform as a one single work unit that can be committed or rolled back.

Let us understand this using an example –

BEGIN TRANSACTION T1;
COMMIT;
SAVEPOINT S1;

```
SELECT * FROM Customers;

INSERT INTO Customers(id) VALUES (1);

UPDATE Customer SET name ='RAJU' WHERE id = 1;
```

Before we jump into the individual description let us understand what we mean by transaction using the above query example. As you can see in the above query (which is in a box), we call a transaction. We begin a transaction using the TCL command BEGIN TRANSACTION after we write all queries we use the TCL command COMMIT to record all the statements in the transaction (query inside the box). At last we save our current progress to save point S1.

Note- The box we have used is just for representational purposes.

There are mainly four type of TCL Command -

1. BEGIN TRANSACTION
2. COMMIT
3. ROLLBACK
4. SAVEPOINT

- **BEGIN TRANSACTION -**

This TCL command basically starts the transaction.

General form:

BEGIN TRANSACTION transaction_name;

- **COMMIT TRANSACTION -**

This TCL command COMMIT makes changes to the transaction.

General form:

BEGIN TRANSACTION transaction_name;

- **ROLLBACK TRANSACTION -**

This TCL commands uncommit all the changes or restore the current state to any previously saved state.

General form-1:

ROLLBACK; /* This rollback the previously committed command to it's initial state*/

General form-1:

ROLLBACK TO savedpoint; /*This is undo the current state and restore it to savepoint*/

- **SAVEPOINT TRANSACTION -**

This TCL command saves the current state into save point. That save point can later be accessed.

General form:

SAVEPOINT Save_point_name;

Now let us understand all these four commands using an example

Table - **Ninja**:

ID	Ninja_Name	Course
1	Suchit	DBMS
2	Kuldeep	OS
3	Lokesh	CP

BEGIN TRANSACTION t1;
DELETE FROM Student WHERE ID = 3;
COMMIT;

Output:
Table - **Ninja:**

ID	Ninja_Name	Course
1	Suchit	DBMS
2	Kuldeep	OS

INSERT INTO Ninja
VALUES (1, 'Lokesh', 'CP');
ROLLBACK;

ID	Ninja_Name	Course
1	Suchit	DBMS
2	Kuldeep	OS

/* Now as we can see even though we inserted new values it is not visible in changes. As we used the ROLLBACK command to undo it. */

SAVEPOINT S1;

/* this saves the current progress to the save point S1*/ DELETE FROM Ninja WHERE ID = 2;
SAVEPOINT S2; /* save point S2 created*/
Output:

ID	Ninja_Name	Course
1	Suchit	DBMS

/* after savepoint 2 above is the current state*/

ROLLBACK TO S1;

Output:

ID	Ninja_Name	Course
1	Suchit	DBMS
2	Kuldeep	OS

Locks:

1. READ LOCK: This lock allows a user to only read the data from a table.
2. WRITE LOCK: This lock allows a user to do both reading and writing into a table.

Query:-

LOCK TABLES T_name [READ | WRITE];

We can lock multiple tables together too.

Query:-

LOCK TABLES T1_name [READ | WRITE],

T2_name [READ | WRITE],. ,

Tn_name [READ | WRITE];

Ex- Now let us suppose we want to lock our table Ninja to read only we can write -

LOCK TABLES Ninja READ;

Now we can only retrieve data from the Ninja table i.e we can not make changes to it.Similarly, if we use the WRITE then we restrict users from both reading and writing.