SQL | TRANSACTIONS

What are Transactions?

Transactions group a set of tasks into a single execution unit. Each transaction begins with a specific task and ends when all the tasks in the group successfully complete. If any of the tasks fail, the transaction fails. Therefore, a transaction has only two results: **success** or **failure**.
Example of a transaction to transfer $150 from account A to account B:
1. read(A)
2. A := A – 150
3. write(A)
4. read(B)
5. B := B + 150
6. write(B)

Incomplete steps result in the failure of the transaction. A database transaction, by definition, must be atomic, consistent, isolated and durable.
These are popularly known as ACID properties. These properties can ensure the concurrent execution of multiple transactions without conflict.

How to implement Transactions using SQL?

Following commands are used to control transactions. It is important to note that these statements cannot be used while creating tables and are only used with the DML Commands such as – INSERT, UPDATE and DELETE.

1. BEGIN TRANSACTION: It indicates the start point of an explicit or local transaction.

Syntax:

```
BEGIN TRANSACTION transaction_name ;
```

2. SET TRANSACTION: Places a name on a transaction.

Syntax:

```
SET TRANSACTION [ READ WRITE | READ ONLY ];
```

3. COMMIT: If everything is in order with all statements within a single transaction, all changes are recorded together in the database is called **committed**. The COMMIT command saves all the transactions to the database since the last COMMIT or ROLLBACK command.

Syntax:

```
COMMIT;
```

Example: Sample table 1

Student				
Rol_No	Name	Address	Phone	Age
1	Ram	Delhi	9455123451	18
2	Ramesh	Gurgaon	9652431543	18
3	Sujit	Rohtak	9156253131	20
4	Suresh	Delhi	9156768971	18
3	Sujit	Rohtak	9156253131	20
2	Ramesh	Gurgaon	9652431543	18

Following is an example which would delete those records from the table which have age = 20 and then COMMIT the changes in the database.

Queries:

```
DELETE FROM Student WHERE AGE = 20;
COMMIT;
```

Output:

Thus, two rows from the table would be deleted and the SELECT statement would look like,

Rol_No	Name	Address	Phone	Age
1	Ram	Delhi	9455123451	18
2	Ramesh	Gurgaon	9652431543	18
4	Suresh	Delhi	9156768971	18
2	Ramesh	Gurgaon	9652431543	18

4. ROLLBACK: If any error occurs with any of the SQL grouped statements, all changes need to be aborted. The process of reversing changes is called **rollback**. This command can only be used to undo transactions since the last COMMIT or ROLLBACK command was issued.
Syntax:

```
ROLLBACK;
```

Example:
From the above example **Sample table1**,
Delete those records from the table which have age = 20 and then ROLLBACK the changes in the database.
Queries:

```
DELETE FROM Student WHERE AGE = 20;
ROLLBACK;
```

Output:

Student				
Rol_No	Name	Address	Phone	Age
1	Ram	Delhi	9455123451	18
2	Ramesh	Gurgaon	9652431543	18
3	Sujit	Rohtak	9156253131	20
4	Suresh	Delhi	9156768971	18
3	Sujit	Rohtak	9156253131	20
2	Ramesh	Gurgaon	9652431543	18

5. SAVEPOINT: creates points within the groups of transactions in which to ROLLBACK.
A SAVEPOINT is a point in a transaction in which you can roll the transaction back to a certain point without rolling back the entire transaction.

Syntax for Savepoint command:

```
SAVEPOINT SAVEPOINT_NAME;
```

This command is used only in the creation of SAVEPOINT among all the transactions.
In general ROLLBACK is used to undo a group of transactions.
Syntax for rolling back to Savepoint command:

```
ROLLBACK TO SAVEPOINT_NAME;
```

you can ROLLBACK to any SAVEPOINT at any time to return the appropriate data to its original state.
Example:
From the above example **Sample table1**,
Delete those records from the table which have age = 20 and then ROLLBACK the changes in the database by keeping Savepoints.
Queries:

```
SAVEPOINT SP1;
//Savepoint created.
DELETE FROM Student WHERE AGE = 20;
//deleted
```

```
SAVEPOINT SP2;
//Savepoint created.
```

Here SP1 is first SAVEPOINT created before deletion.In this example one deletion have taken place.
After deletion again SAVEPOINT SP2 is created.
Output:

Rol_No	Name	Address	Phone	Age
1	Ram	Delhi	9455123451	18
2	Ramesh	Gurgaon	9652431543	18
4	Suresh	Delhi	9156768971	18
2	Ramesh	Gurgaon	9652431543	18

Deletion have been taken place, let us assume that you have changed your mind and decided to ROLLBACK to the SAVEPOINT that you identified as SP1 which is before deletion.
deletion is undone by this statement ,

```
ROLLBACK TO SP1;
//Rollback completed.
```

Student				
Rol_No	Name	Address	Phone	Age
1	Ram	Delhi	9455123451	18
2	Ramesh	Gurgaon	9652431543	18
3	Sujit	Rohtak	9156253131	20
4	Suresh	Delhi	9156768971	18
3	Sujit	Rohtak	9156253131	20
2	Ramesh	Gurgaon	9652431543	18

6. RELEASE SAVEPOINT:- This command is used to remove a SAVEPOINT that you have created.

Syntax:

```
RELEASE SAVEPOINT SAVEPOINT_NAME
```

Once a SAVEPOINT has been released, you can no longer use the ROLLBACK command to undo transactions performed since the last SAVEPOINT.

It is used to initiate a database transaction and used to specify characteristics of the transaction that follows.

Last Updated : 25 Oct, 2022

165

Similar Reads

1. PL/SQL Transactions
2. Auto-commit commands and Compensating transactions in SQL
3. Difference between Structured Query Language (SQL) and Transact-SQL (T-SQL)
4. Configure SQL Jobs in SQL Server using T-SQL
5. SQL vs NO SQL vs NEW SQL
6. SQL | Procedures in PL/SQL
7. SQL | Difference between functions and stored procedures in PL/SQL
8. SQL SERVER – Input and Output Parameter For Dynamic SQL
9. Difference between SQL and T-SQL
10. SQL Server | Convert tables in T-SQL into XML

Previous Next

Article Contributed By :

GeeksforGeeks

Vote for difficulty