TRANSACTION in SQL

A transaction is successfully performed when all commands are successful, then all data changes made in the transaction are saved to the database.

TRANSACTION in SQL is the process of executing a group of SQL statements. These statements are executed sequentially and independently. A transaction is successfully performed when all commands are successful, then all data changes made in the transaction are saved to the database. However, if only one of them fails, the entire process will fail, meaning that the data must roll back to its original state (data is restored to the state before the transaction is executed).

Attribute of Transaction

1. Atomicity - the "all or nothing" principle, ensuring that all commands in the command group are executed successfully. Otherwise, the Transaction is canceled at the time of failure and all operations previously restored to the old state means nothing has changed in terms of data.

2. Consistency - Consistency: ensure that the database changes the exact state when a transaction is successfully executed.

3. Isolation - Independence: allows transactions to operate independently and transparently.

4. Durability - Ensures that the outcome of a transaction is determined, that no data of the Transaction after execution can be transferred back to the data state before execution.

Handling transactions

In SQL, there are the following commands used to control transactions:

1. **COMMIT:** to save the changes.
2. **ROLLBACK:** to return to the state before the change.
3. **SAVEPOINT:** create points within the Transaction groups to ROLLBACK, ie to return to that state point.
4. **SET TRANSACTION:** Name a transaction.

Transaction control commands are only used with data manipulation commands such as INSERT, UPDATE and DELETE. However, they cannot be used in the CREATE TABLE or DROP TABLE statement because these operations are automatically defined in the database.

COMMIT command in SQL

Transaction ends with either COMMIT or ROLLBACK statement.

When a complete Transaction is completed, **the COMMIT command** must be called. This is the Transaction control command used to store changes called by a Transaction to the database.

Basic syntax of COMMIT command is as follows:

```
COMMIT;
```

For example : Suppose the *NHANVIEN* table has the following records:

```
+----+----------+-----+-----------+----------+ | ID | TEN |TUOI | DIACHI | LUONG
```

The following example will delete the records from the table with *tuoi = 25* and then COMMIT the changes to the Database.

```
SQL> DELETE FROM NHANVIEN
 WHERE TUOI = 25;
SQL> COMMIT;
```

Therefore, the two rows from the table will be deleted and the SELECT statement will produce the following results.

```
+----+----------+-----+-----------+----------+ | ID | TEN |TUOI | DIACHI | LUONG
```

ROLLBACK command in SQL

The ROLLBACK command is the Transaction control command used to return the Transaction to a state before changes have not yet been saved to the Database. The ROLLBACK command can only be used to undo transactions before confirming it with the last Commit or Rollback command.

The basic syntax of ROLLBACK command is as follows:

```
ROLLBACK;
```

For example : Suppose the *NHANVIEN* table has the following records:

```
+----+----------+-----+-----------+----------+ | ID | TEN |TUOI | DIACHI | LUONG
```

Now use the ROLLBACK command with the delete command *tuoi = 25* , not yet committed as follows:

```
SQL> DELETE FROM NHANVIEN
 WHERE TUOI = 25;
SQL> ROLLBACK;
```

In the results obtained, this DELETE operation does not affect the table because of ROLLBACK changes in the database, the SELECT statement will produce the result:

```
+----+----------+-----+-----------+----------+ | ID | TEN |TUOI | DIACHI | LUONG
```

SAVEPOINT command in SQL

SAVEPOINT is a point in a Transaction that you can back Transaction back to a certain point without having to reverse the Transaction to the first state before making that change.

The basic syntax of the SAVEPOINT command is as follows:

```
SAVEPOINT TEN_SAVEPOINT;
```

This command only creates SAVEPOINT in Transaction transactions. ROLLBACK then needs to be used to undo a SAVEPOINT as follows:

```
ROLLBACK TO TEN_SAVEPOINT;
```

For example : You want to delete three different records from the NHANVIEN table and want to create SAVEPOINT before each deletion to be able to ROLLBACK back to SAVEPOINT at any time to return the appropriate data for the initial state.

Suppose the *NHANVIEN* table has the following records:

```
+----+----------+-----+-----------+----------+ | ID | TEN |TUOI | DIACHI | LUONG
```

Here is a sequence of statements:

```
SQL> SAVEPOINT SP1;
Savepoint created.
SQL> DELETE FROM NHANVIEN WHERE ID=1;
1 row deleted.
SQL> SAVEPOINT SP2;
Savepoint created.
SQL> DELETE FROM NHANVIEN WHERE ID=2;
1 row deleted.
SQL> SAVEPOINT SP3;
Savepoint created.
SQL> DELETE FROM NHANVIEN WHERE ID=3;
1 row deleted.
```

Above, there are 3 data delete operations that take place. Suppose you change your mind and decide ROLLBACK to SAVEPOINT that you identified as SP2. Because SP2 was created after the first delete operation, the last two delete operations are restored.

```
SQL> ROLLBACK TO SP2;
Rollback complete.
```

So only the first delete operation takes place after you ROLLBACK to SP2.

ID	TEN	TUOI	DIACHI	LUONG
2	Loan	25	Hanoi	1500.00
3	Nga	23	Hanam	2000.00
4	Manh	25	Hue	6500.00
5	Huy	27	Hatinh	8500.00
6	Cao	22	HCM	4500.00
7	Lam	24	Hanoi	10000.00

RELEASE SAVEPOINT command in SQL

The SAVEPOINT RELEASE command is used to remove a SAVEPOINT that you have created. When SAVEPOINT is deleted, you cannot use the ROLLBACK command to undo those SAVEPOINT transactions.

The syntax of SAVEPOINT RELEASE is as follows:

```
RELEASE SAVEPOINT TEN_SAVEPOINT;
```

The SET TRANSACTION command in SQL

The SET TRANSACTION can be used to initialize a Database Transaction. This command is used to specify properties for that Transaction. For example, you can specify a Transaction to be read only (read only) or read or write (read write).

The basic syntax of the SET TRANSACTION command in SQL is as follows:

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

In the next section, we will learn about **the representative operators - WILDCARD in SQL** , remember to watch.

Previous article: HAVING clause in SQL

Next lesson: Representative operator - WILDCARD in SQL

You finished reading the article "**TRANSACTION in SQL**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.