SQL - TRANSACTIONS

A transaction is a unit of work that is performed against a database. Transactions are units or sequences of work accomplished in a logical order, whether in a manual fashion by a user or automatically by some sort of a database program.

A transaction is the propagation of one or more changes to the database. For example, if you are creating a record or updating a record or deleting a record from the table, then you are performing transaction on the table. It is important to control transactions to ensure data integrity and to handle database errors.

Practically, you will club many SQL queries into a group and you will execute all of them together as a part of a transaction.

Properties of Transactions:

Transactions have the following four standard properties, usually referred to by the acronym ACID:

- **Atomicity:** ensures that all operations within the work unit are completed successfully; otherwise, the transaction is aborted at the point of failure, and previous operations are rolled back to their former state.
- **Consistency:** ensures that the database properly changes states upon a successfully committed transaction.
- **Isolation:** enables transactions to operate independently of and transparent to each other.
- **Durability:** ensures that the result or effect of a committed transaction persists in case of a system failure.

Transaction Control:

There are following commands used to control transactions:

- **COMMIT:** to save the changes.
- **ROLLBACK:** to rollback the changes.
- **SAVEPOINT:** creates points within groups of transactions in which to ROLLBACK
- **SET TRANSACTION:** Places a name on a transaction.

Transactional control commands are only used with the DML commands INSERT, UPDATE and DELETE only. They can not be used while creating tables or dropping them because these operations are automatically commited in the database.

The COMMIT Command:

The COMMIT command is the transactional command used to save changes invoked by a transaction to the database.

The COMMIT command saves all transactions to the database since the last COMMIT or ROLLBACK command.

The syntax for COMMIT command is as follows:

```
COMMIT;
```

Example:

Consider the CUSTOMERS table having the following records:

```
+----+----------+-----+-----------+----------+
```

```
| ID | NAME      | AGE | ADDRESS    | SALARY   |
+----+-----------+-----+------------+----------+
|  1 | Ramesh    |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan    |  25 | Delhi      |  1500.00 |
|  3 | kaushik   |  23 | Kota       |  2000.00 |
|  4 | Chaitali |  25 | Mumbai     |  6500.00 |
|  5 | Hardik    |  27 | Bhopal     |  8500.00 |
|  6 | Komal     |  22 | MP         |  4500.00 |
|  7 | Muffy     |  24 | Indore     | 10000.00 |
+----+-----------+-----+------------+----------+
```

Following is the example which would delete records from the table having age = 25 and then COMMIT the changes in the database.

```
SQL> DELETE FROM CUSTOMERS
     WHERE AGE = 25;
SQL> COMMIT;
```

As a result, two rows from the table would be deleted and SELECT statement would produce the following result:

```
+----+-----------+-----+------------+----------+
| ID | NAME      | AGE | ADDRESS    | SALARY   |
+----+-----------+-----+------------+----------+
|  1 | Ramesh    |  32 | Ahmedabad |  2000.00 |
|  3 | kaushik   |  23 | Kota       |  2000.00 |
|  5 | Hardik    |  27 | Bhopal     |  8500.00 |
|  6 | Komal     |  22 | MP         |  4500.00 |
|  7 | Muffy     |  24 | Indore     | 10000.00 |
+----+-----------+-----+------------+----------+
```

The ROLLBACK Command:

The ROLLBACK command is the transactional command used to undo transactions that have not already been saved to the database.

The ROLLBACK command can only be used to undo transactions since the last COMMIT or ROLLBACK command was issued.

The syntax for ROLLBACK command is as follows:

```
ROLLBACK;
```

Example:

Consider the CUSTOMERS table having the following records:

```
+----+-----------+-----+------------+----------+
| ID | NAME      | AGE | ADDRESS    | SALARY   |
+----+-----------+-----+------------+----------+
|  1 | Ramesh    |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan    |  25 | Delhi      |  1500.00 |
|  3 | kaushik   |  23 | Kota       |  2000.00 |
|  4 | Chaitali |  25 | Mumbai     |  6500.00 |
|  5 | Hardik    |  27 | Bhopal     |  8500.00 |
|  6 | Komal     |  22 | MP         |  4500.00 |
|  7 | Muffy     |  24 | Indore     | 10000.00 |
+----+-----------+-----+------------+----------+
```

Following is the example, which would delete records from the table having age = 25 and then ROLLBACK the changes in the database.

```
SQL> DELETE FROM CUSTOMERS
     WHERE AGE = 25;
SQL> ROLLBACK;
```

As a result, delete operation would not impact the table and SELECT statement would produce the following result:

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

The SAVEPOINT Command:

A SAVEPOINT is a point in a transaction when you can roll the transaction back to a certain point without rolling back the entire transaction.

The syntax for SAVEPOINT command is as follows:

```
SAVEPOINT SAVEPOINT_NAME;
```

This command serves only in the creation of a SAVEPOINT among transactional statements. The ROLLBACK command is used to undo a group of transactions.

The syntax for rolling back to a SAVEPOINT is as follows:

```
ROLLBACK TO SAVEPOINT_NAME;
```

Following is an example where you plan to delete the three different records from the CUSTOMERS table. You want to create a SAVEPOINT before each delete, so that you can ROLLBACK to any SAVEPOINT at any time to return the appropriate data to its original state:

Example:

Consider the CUSTOMERS table having the following records:

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Now, here is the series of operations:

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

```
1 row deleted.
```

Now that the three deletions have taken place, say you have changed your mind and decided to ROLLBACK to the SAVEPOINT that you identified as SP2. Because SP2 was created after the first deletion, the last two deletions are undone:

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

Notice that only the first deletion took place since you rolled back to SP2:

```
SQL> SELECT * FROM CUSTOMERS;
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
6 rows selected.
```

The RELEASE SAVEPOINT Command:

The RELEASE SAVEPOINT command is used to remove a SAVEPOINT that you have created.

The syntax for RELEASE SAVEPOINT is as follows:

```
RELEASE SAVEPOINT SAVEPOINT_NAME;
```

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

The SET TRANSACTION Command:

The SET TRANSACTION command can be used to initiate a database transaction. This command is used to specify characteristics for the transaction that follows.

For example, you can specify a transaction to be read only, or read write.

The syntax for SET TRANSACTION is as follows:

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