DDL (DATA DEFINITION LANGUAGE)

1. **CREATE:** create statement is used to create table in database.

Syntax

```
 CREATE TABLE  Table_name
(column1 datatype(size) constraint(s),
 column2 datatype(size) constraint(s),
 column3 datatype(size) constraint(s));
```

Ex:

```
 CREATE TABLE  Job
 (job_id number not null,
 First_Name varchar(20) not null,
 Last_Name varchar(20)not null,
 Salary  Number(10) check(salary>35000),
City varchar(30)unique,
 Education varchar(30) not null) ;
```

2. **ALTER:** Alter statement is used to modify the table with respect to columns.

i) Add the column to existing table

Syntax

```
ALTER TABLE table_name
ADD column_name datatype;
```

Ex:

```
ALTER TABLE Customers
ADD Adress varchar(40) not null;
```

ii) Delete the column in existing table

Syntax

```
ALTER TABLE  table_name
DROP COLUMN  column_name ;
```

Ex

```
ALTER TABLE job
DROP COLUMN city;
```

iii) Rename the column to existing table

Syntax

```
ALTER TABLE Table_name
RENAME COLUMN
Old_column to new_column;
```

Ex

```
ALTER TABLE job
 RENAME COLUMN
Education to Qualification;
```

3. **RENAME:** Rename statement is used to rename a table.

Syntax

```
RENAME
Old_Table_name  to new_Table_name;
```

Ex

```
RENAME
job  to Emp;
```

4. **TRUNCATE:** Truncate is used to remove all records of a table at one shot. Truncate makes table empty. Truncate will not remove table from database instead it removes records from table.

Syntax

```
Truncate Table
Table_name ;
```

Ex

```
Truncate Table
 Job;
```

5. **DROP:** Drop statement is used to remove table from database.

Syntax

```
DROP TABLE  Table_name ;
```

Ex

```
DROP TABLE   Job;
```

- IF DROP is applied on table, the table will be removed from database and gets stored in BIN. To restore the table from BIN to DATABASE we follow FLASHBACK.

Syntax

```
 FLASHBACK TABLE
Table_name
TO BEFORE DROP;
```

Ex

```
FLASHBACK TABLE
Job
TO BEFORE DROP;
```

- To remove the table PERMANENTLY from DATABASE without storing in BIN we use PURGE.

Syntax

```
 BEFORE DROP→ DROP TABLE
 Table_name
 PURGE;

AFTER DROP→PURGE TABLE
 Table_name;
```

Ex

```
BEFORE DROP: DROP TABLE
 Job
 PURGE;

AFTER DROP: PURGE TABLE
 Job
```

DML (DATA MANIPULATION LANGUAGE)

1. **INSERT:** INSERT is used to add records into the table.

   Sntax

   ```
   INSERT INTO TABLE_NAME
   Values(V1,V2,V3…)
   ```

Ex

```
INSERT INTO TABLE_NAME
 Values(1,Nithya,K,35000,Banglore,BE)
```

Or

Sntax

```
INSERT INTO TABLE_NAME
( Column1, Column2, Column3,…)
 Values(V1,V2,V3…)
```

Ex

```
INSERT INTO TABLE_NAME
(Job_id, First_name, Last_name,Salary,City,Education)
Values(12,Nithya,K,35000,Banglore,BE)
```

2. **UPDATE**: UPDATE is used to update existing data in table with newer data.

   Syntax

   ```
   UPDATE TABLE_NAME
    SET Column_ NAME =New value
   Where CONDITION(s);
   ```

   Ex

   ```
   UPDATE  Job
   SET First_name =’Mohan’
   Where   job_id=1;
   ```

3. **DELETE:** DELETE is used to remove specific records from the table.

   Syntax

   ```
   DELETE FROM  TABLENAME
   Where CONDITION(s);
   ```

   Ex:1

   ```
   DELETE FROM  Job
   Where job_id=1;
   ```

   Ex:2

   ```
   DELETE FROM  Job
   Where job_id in(1,2,3);
   ```

NOTE: ALL the actions performed using DDL is PERMANENT.
ALL the actions performed using DML is Not PERMANENT.

TCL (TRANSACTION CONTROL LANGAUGE):

Since all the DML actions are not permanent, it can be restored (undo) or it can be saved permanently using TCL

1.ROLLBACK: Rollback statement is used to restore all the DML actions.

Syntax

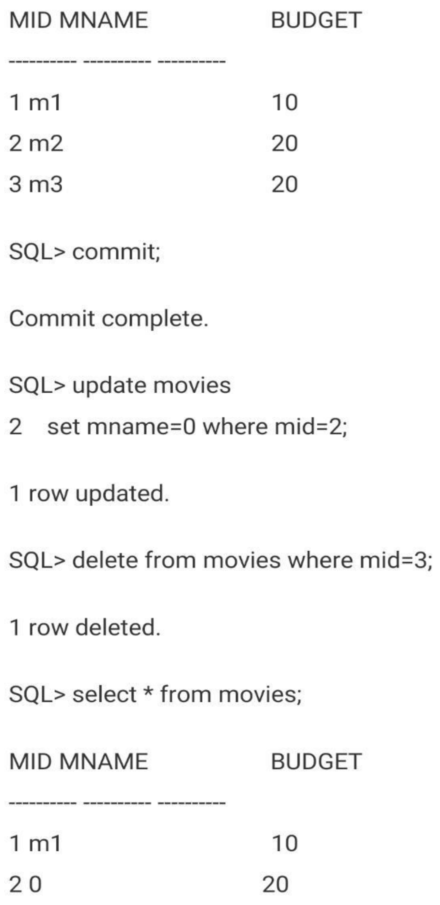
ROLLBACK;

2. COMMIT: Commit statement is used to save all the DML actions as PERMANENT.

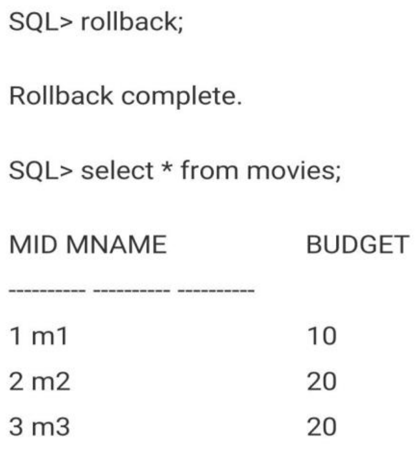
Syntax

COMMIT;

Ex



```
MID MNAME              BUDGET
---------- ---------- ----------
1 m1                   10
2 m2                   20
3 m3                   20

SQL> commit;

Commit complete.

SQL> update movies
2    set mname=0 where mid=2;

1 row updated.

SQL> delete from movies where mid=3;

1 row deleted.

SQL> select * from movies;

MID MNAME              BUDGET
---------- ---------- ----------
1 m1                   10
2 0                   20
```



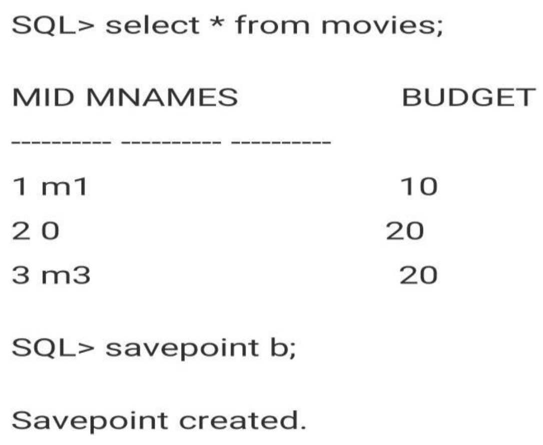
```
SQL> rollback;

Rollback complete.

SQL> select * from movies;

MID MNAME              BUDGET
---------- ---------- ----------
1 m1                   10
2 m2                   20
3 m3                   20
```

3. SAVEPOINT: Save point is a named point created for each and every DML actions.

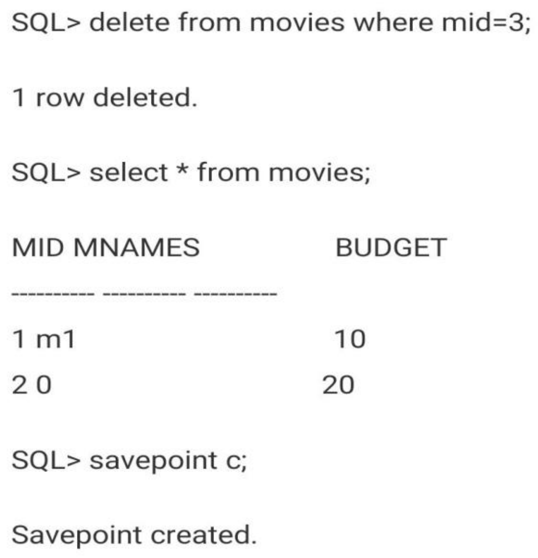
Syntax

SAVEPOINT;

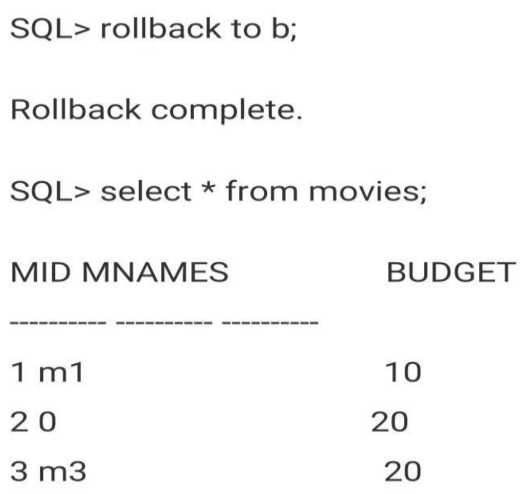
Ex



```
SQL> select * from movies;

MID MNAMES              BUDGET
---------- ---------- ----------
1 m1                     10
2 0                     20
3 m3                     20

SQL> savepoint b;

Savepoint created.
```



```
SQL> delete from movies where mid=3;

1 row deleted.

SQL> select * from movies;

MID MNAMES              BUDGET
---------- ---------- ----------
1 m1                     10
2 0                     20

SQL> savepoint c;

Savepoint created.
```



```
SQL> rollback to b;

Rollback complete.

SQL> select * from movies;

MID MNAMES              BUDGET
---------- ---------- ----------
1 m1                     10
2 0                     20
3 m3                     20
```

DCL (DATA CONTROL LANGUAGE)

DCL is mainly used to provide permission or to take back permission for accessing of a data from one user to another user.

1. **GRANT: Grant is mainly used to provide a permission for accessing of a data from one user to another user.**

   Syntax

   GRANT PERMISSION_TYPE ON TABLE_NAME;

2. **REVOKE: Revoke is mainly used to take back a permission for accessing of a data from one user to another user.**

   Syntax

   REVOKE PERMISSION_TYPE ON TABLE_NAME FROM USER_NAME