SQL Commands

- Create a database:

```
mysql> create database db1;
Query OK, 1 row affected (0.02 sec)
```

- Open a database:

```
mysql> use db;
Database changed
```

- Delete a database

```
mysql> Drop database db1
    -> ;
Query OK, 0 rows affected (0.15 sec)
```

Create table Command

- It is used to create a table in a database
- E.g

```
mysql> create table student
    -> ( ROLLNO INTEGER(4) PRIMARY KEY,
    -> NAME VARCHAR(25) NOT NULL,
    -> GENDER CHAR(1),
    -> MARKS INTEGER(3),
    -> DOB DATE,
    -> MOBILE_NO BIGINT(12),
    -> STREAM VARCHAR(15));
Query OK, 0 rows affected (0.11 sec)
```

Insert a Record

- Insert command is used to insert a record
- Insert a row in table

```
mysql> insert into student
    -> values(1,'Raj Kumar','M',93,'2000-09-17',9586774748,'Science');
Query OK, 1 row affected (0.02 sec)
```

- Insert null values in the table

```
mysql> insert into student
    -> values(3,'Raman Verma','M',76,'2000-02-22',NULL,'Science');
Query OK, 1 row affected (0.00 sec)
```

- Insert values in particular column

```
mysql> insert into student(rollno,name,gender,stream)
    -> values(7,'Samita Sachdeva','F','Commerce');
Query OK, 1 row affected (0.00 sec)
```

Display structure of table

- Describe <table name> or desc <tablename>

```
mysql> desc student;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| ROLLNO    | int(4)      | NO   | PRI | NULL    |       |
| NAME      | varchar(25) | NO   |     | NULL    |       |
| GENDER    | char(1)     | YES  |     | NULL    |       |
| MARKS     | int(3)      | YES  |     | NULL    |       |
| DOB       | date        | YES  |     | NULL    |       |
| MOBILE_NO | bigint(12)  | YES  |     | NULL    |       |
| STREAM    | varchar(15) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
7 rows in set (0.02 sec)
```

Update Command

- It is used to change the record of the table. You can change all the rows or particular row from the table. The **Update** command specifies the row with **where** clause and new data is written into respective record using **set** keyword. E.g
- Student table before change

```
mysql> select * from student;
+--------+-----------------+--------+-------+------------+------------+------------+
| ROLLNO | NAME            | GENDER | MARKS | DOB        | MOBILE_NO  | STREAM     |
+--------+-----------------+--------+-------+------------+------------+------------+
|      1 | Raj Kumar       | M      |    93 | 2000-09-17 | 9586774748 | Science    |
|      2 | Rahul singh     | M      |    90 | 2000-09-11 | 9586767987 | Commerce   |
|      3 | Raman Verma     | M      |    76 | 2000-02-22 |       NULL | Science    |
|      4 | Suman           | F      |    78 | 1999-12-03 | 9818675444 | Humanities |
|      5 | Sugandha        | F      |    82 | 1998-04-21 | 9845639990 | vocational |
|      6 | Payal Goyal     | F      |    80 | 1999-12-17 | 9897666650 | Science    |
|      7 | Samita Sachdeva | F      |  NULL | NULL       |       NULL | Commerce   |
+--------+-----------------+--------+-------+------------+------------+------------+
7 rows in set (0.00 sec)
```

Various way of Updating records

- Change a particular value

```
mysql> update student
    -> set dob='1999-10-02'
    -> where rollno=7;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- Updating multiple records

```
mysql> update student
    -> set marks=89, dob='1999-11-03'
    -> where name='Suman';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- Updating Null value

```
mysql> update student
    -> set marks=Null
    -> where name='Sugandha';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- Updating using expression

```
mysql> update student
    -> set marks=marks+5
    -> where rollno=2;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Records after Modification

```
mysql> select * from student;
+--------+-----------------+--------+-------+------------+------------+------------+
| ROLLNO | NAME            | GENDER | MARKS | DOB        | MOBILE_NO  | STREAM     |
+--------+-----------------+--------+-------+------------+------------+------------+
|      1 | Raj Kumar       | M      |    93 | 2000-09-17 | 9586774748 | Science    |
|      2 | Rahul singh     | M      |    95 | 2000-09-11 | 9586767987 | Commerce   |
|      3 | Raman Verma     | M      |    76 | 2000-02-22 |       NULL | Science    |
|      4 | Suman           | F      |    89 | 1999-11-03 | 9818675444 | Humanities |
|      5 | Sugandha        | F      |  NULL | 1998-04-21 | 9845639990 | vocational |
|      6 | Payal Goyal     | F      |    80 | 1999-12-17 | 9897666650 | Science    |
|      7 | Samita Sachdeva | F      |  NULL | 1999-10-02 |       NULL | Commerce   |
+--------+-----------------+--------+-------+------------+------------+------------+
7 rows in set (0.00 sec)
```

Delete Records

- There are two commands to delete records:
- Delete
- Truncate
- **Delete:it is used to delete all rows or particular row from the table using where clause.**
- **Truncate: to delete al the row from table and free the space for containing the table**
- **Drop Table: This command is used to physically delete the table i.e. remove structure also.**

- E.g. of delete

```
mysql> delete from student where rollno=7;
Query OK, 1 row affected (0.00 sec)
```

- E.g. of truncate

```
mysql> truncate student1;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> select * from student1;
Empty set (0.01 sec)
```

- E.g. Drop table

```
mysql> drop table student1;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from student1;
ERROR 1146 (42S02): Table 'db.student1' doesn't exist
```

Modify the Structure of Table

- Alter Table command is used to modify the structure of table by modifying the column definition of its column. It perform following operations:
- To add new column in table

```
mysql> alter table student
    -> add emailid varchar(50);
Query OK, 6 rows affected (0.14 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

  - To rename column

```
mysql> alter table student
    -> change name sname varchar(30);
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

- To change data type or modify size

```
mysql> alter table student
    -> modify stream varchar(12);
Query OK, 6 rows affected (0.02 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

- To remove a column physically

```
mysql> alter table student
    -> drop gender;
Query OK, 6 rows affected (0.15 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

Structure after Modification

Before Modification

```
mysql> desc student;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| ROLLNO    | int(4)      | NO   | PRI | NULL    |       |
| NAME      | varchar(25) | NO   |     | NULL    |       |
| GENDER    | char(1)     | YES  |     | NULL    |       |
| MARKS     | int(3)      | YES  |     | NULL    |       |
| DOB       | date        | YES  |     | NULL    |       |
| MOBILE_NO | bigint(12)  | YES  |     | NULL    |       |
| STREAM    | varchar(15) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
7 rows in set (0.02 sec)
```

After Modification

```
mysql> desc student;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| ROLLNO    | int(4)      | NO   | PRI | NULL    |       |
| sname     | varchar(30) | YES  |     | NULL    |       |
| MARKS     | int(3)      | YES  |     | NULL    |       |
| DOB       | date        | YES  |     | NULL    |       |
| MOBILE_NO | bigint(12)  | YES  |     | NULL    |       |
| stream    | varchar(12) | YES  |     | NULL    |       |
| emailid   | varchar(50) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
7 rows in set (0.01 sec)
```

SQL Commands

Grouping Records in a Query

- **Some time it is required to apply a Select query in a group of records instead of whole table.**
- **We can group records by using GROUP BY <column> clause with Select command. A group column is chosen which have non-distinct (repeating) values like City, Job etc.**
- **Generally, the following Aggregate Functions [MIN(), MAX(), SUM(), AVG(), COUNT()] etc. are applied on groups.**

Name	Purpose
SUM()	Returns the sum of given column.
MIN()	Returns the minimum value in the given column.
MAX()	Returns the maximum value in the given column.
AVG()	Returns the Average value of the given column.
COUNT()	Returns the total number of values/ records as per given column.

SQL Commands

Aggregate Functions & NULL

Consider a table Emp having following records as-

Null values are excluded while (avg)aggregate function is used

Emp		
Code	Name	Sal
E1	Mohak	NULL
E2	Anuj	4500
E3	Vijay	NULL
E4	Vishal	3500
E5	Anil	4000

SQL Queries

Result of query

```
mysql> Select Sum(Sal) from EMP;                    12000
mysql> Select Min(Sal) from EMP;                    3500
mysql> Select Max(Sal) from EMP;                    4500
mysql>   Select Count(Sal) from EMP;                3
mysql>   Select Avg(Sal) from EMP;                  4000
mysql>   Select Count(*) from EMP;                  5
```

SQL Commands

Aggregate Functions & Group

An Aggregate function may applied on a column with DISTINCT or ALL keyword. If nothing is given ALL is assumed.

Using SUM (<Column>)

This function returns the sum of values in given column orexpression.

```
mysql> Select Sum(Sal) from EMP;
mysql> Select Sum(DISTINCT Sal) from EMP;
mysql> Select Sum ( S a l ) from EMP where
                City='Jaipur ';
mysql> Select Sum ( S a l ) from EMP Group By C i t y ;
mysql> Select Job, Sum(Sal) from EMP Group By Job;
```

Using MIN (<column>)

This functions returns the Minimum value in the given column.

```
mysql> Select Min(Sal) from EMP;
mysql> Select Min(Sal) from EMP Group By C i t y ;
mysql> Select Job, Min(Sal) from EMP Group By Job;
```

SQL Commands

Aggregate Functions & Group

Using MAX (<Column>)

This function returns the Maximum value in given column.

```
mysql> Select  Max(Sal)  from  EMP;
mysql> Select  Max(Sal)  from  EMP where City='Jaipur ';
mysql> Select  Max(Sal)  from  EMP Group By C i t y ;
```

- Using AVG (<column>)
- This functions returns the Average value in the given column.

```
mysql> Select    AVG(Sal) from    EMP;
mysql> Select    AVG(Sal) from    EMP    Group    By C i t y ;
```

- Using COUNT (<*|column>)
- This functions returns the number of rows in the given column.

```
mysql> Select  Count  ( * ) from EMP;
mysql> Select  Count(Sal)   from EMP Group By C i t y ;
mysql> Select  Count(*),    Sum(Sal)  from EMP Group By Job;
```

SQL Commands

Aggregate Functions & Conditions

You may use any condition on group, if required. HAVING <condition> clause is used to apply a condition on a group.

```
mysql>  Select Job,Sum(Pay) from EMP
Group  By Job HAVING Sum(Pay)>=8000;

mysql> Select  Job,  Sum(Pay)  from  EMP
Group  By Job HAVING Avg(Pay)>=7000;

mysql> Select  Job,  Sum(Pay)  from  EMP  Group By Job HAVING
Count(*)>=5;

mysql>  Select  Job, Min(Pay),Max(Pay),  Avg(Pay) from EMP Group
By Job HAVING Sum(Pay)>=8000;

mysql> Select  Job, Sum(Pay)  from EMP Where City='Jaipur'
```

Note :- Where clause works in respect of whole table but Having works on Group only. If Where and Having both are used then Where will be executed first.

SQL Commands

Ordering Query Result – ORDER BY Clause

A query result can be orders in ascending (A-Z) or descending (Z-A)

order as per any column. Default is Ascending order.

mysql> SELECT * FROM Student ORDER BY City;

To get descending order use DESC key word.

mysql> SELECT * FROM Student ORDER BY City DESC;

mysql> SELECT Name, Fname, City FROM Student Where Name LIKE ‘R%’ ORDER BY Class;