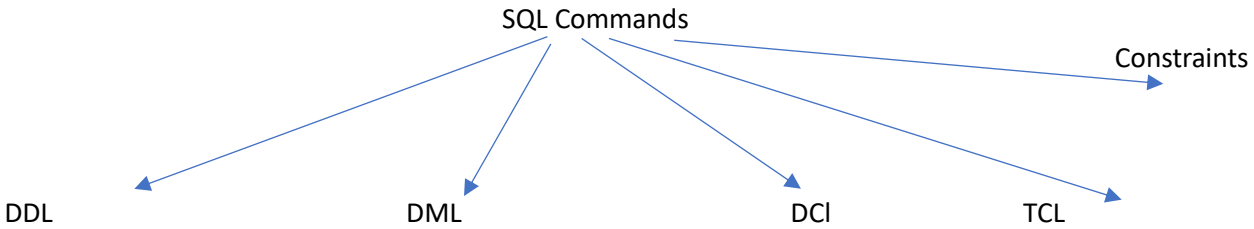
SQL:



(Data Definition Language) (Data Manipulation Language) (Data Control language) (Transaction Control Language)

Create	Select	Grant	Commit
Alter	Insert	Revoke	RollBack
Drop	Update		Save Point
Truncate	Delete		
rename			

- In order to make schema, Table, Mention attributes that's we do with the help of DDL
- But in order to insert values in that table we use DML
- In DCL we discuss about the access to users of our database
- In TCL we use for transactions like we use online payment (ACID properties work here like if transaction is complete or in between the process or got failed)
- Constraints: Like when we mention attributes there, we mention constraints (**like Keys, Check constraints, Default, Unique, Default, Null**)

Data Types:

In SQL, data types define the type of data that can be stored in a column of a table. Each database system might have its own set of data types, but here's a list of common data types along with examples:

1. INT (Integer):

Used for storing whole numbers.

```
CREATE TABLE Students (
    StudentID INT,
    Age INT
);
```

2. VARCHAR(n) (Variable-Length Character String):

Used for storing strings of variable length up to 'n' characters.

```sql
CREATE TABLE Employees (
    EmployeeID INT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50)
);
```

3. CHAR(n) (Fixed-Length Character String):

Used for storing fixed-length strings of 'n' characters.

```sql
CREATE TABLE Customers (
    CustomerID CHAR(10),
    Country CHAR(20)
);
```

4. DATE:

Used for storing dates (year, month, day).

```sql
CREATE TABLE Orders (
    OrderID INT,
    OrderDate DATE
);
```

5. TIME:

Used for storing time of day.

```sql
CREATE TABLE Appointments (
    AppointmentID INT,
    AppointmentTime TIME
);
```

6. DATETIME or TIMESTAMP:

Used for storing both date and time.

```
CREATE TABLE Events (
    EventID INT,
    EventDateTime DATETIME
);
```

7. FLOAT or DOUBLE:

Used for storing floating-point numbers (decimal numbers).

```
CREATE TABLE Products (
    ProductID INT,
    Price FLOAT
);
```

8. BOOLEAN or BIT:

Used for storing boolean values (true or false).

```
CREATE TABLE Tasks (
    TaskID INT,
    IsCompleted BOOLEAN
);
```

9. BLOB (Binary Large Object):

Used for storing binary data like images, videos, etc.

```
CREATE TABLE Images (
    ImageID INT,
    ImageData BLOB
);
```

10. ENUM:

Used for creating a set of predefined values that a column can take.

```
CREATE TABLE Colors (
    ColorID INT,
    Color ENUM('Red', 'Green', 'Blue')
);
```

DDL:

1. **Create Command:**

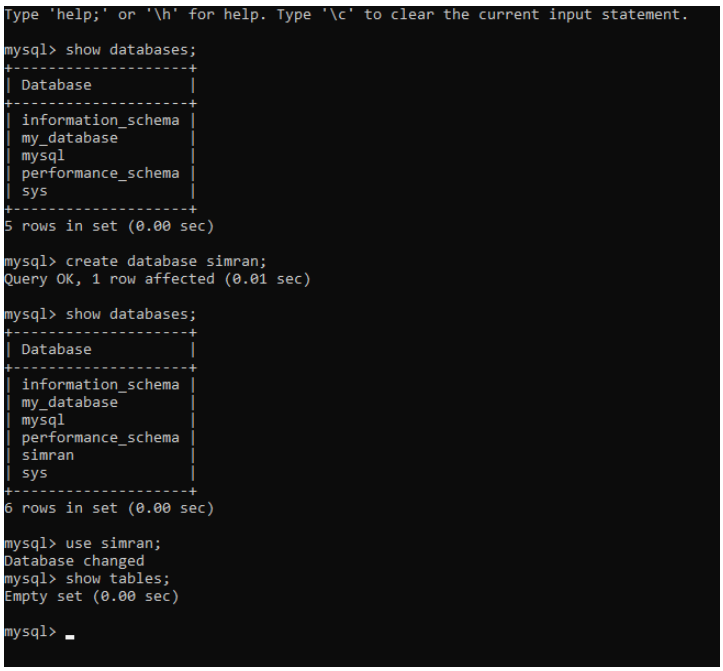
Let's create a table structure

- **Create table student (student_id int, Name varchar(30), Address varchar(50));**

- In Table Name – we can't use . or – or stu (space)table this should not be happening
- Table name should be single value
- Data types: Integer, varchar (4 byte size),
- After the last attribute don't use comma(,) and after closing the bracket put ;(semi-column)

***In workbench its graphical user interface and CMD is character ser interface**

Practice:



- Show databases;
- Create database studentdata;
- Use studentdata;
- Show tables;

```
mysql> create table student(Student_id int, Name varchar(20), Address varchar(30));
Query OK, 0 rows affected (0.02 sec)

mysql> desc student
    -> ;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Student_id | int         | YES  |     | NULL    |       |
| Name       | varchar(20) | YES  |     | NULL    |       |
| Address    | varchar(30) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

- **Example1**: Create table student (Student_id int, Name varchar(20), Address varchar(30));

  desc student;

- **Example2**: create table student1(RollNumber dec(4) primary key, StudentName varchar(20) not null, RegistrationNumber dec(5) unique, Course varchar(10), TotalMarks dec(3) default(100), MarksObtained dec(3), check(MarksObtained<=TotalMarks));

  Desc student1;

```
mysql> create table student1(RollNumber dec(4) primary key, StudentName varchar(20) not null, RegistrationNumber dec(5) unique, Course varchar(10), TotalMarks dec(3) de
fault(100), MarksObtained dec(3), check(MarksObtained<=TotalMarks));
Query OK, 0 rows affected (0.10 sec)

mysql> desc student1;
+--------------------+-------------+------+-----+---------+-------------------+
| Field              | Type        | Null | Key | Default | Extra             |
+--------------------+-------------+------+-----+---------+-------------------+
| RollNumber         | decimal(4,0) | NO   | PRI | NULL    |                   |
| StudentName        | varchar(20) | NO   |     | NULL    |                   |
| RegistrationNumber | decimal(5,0) | YES  | UNI | NULL    |                   |
| Course             | varchar(10) | YES  |     | NULL    |                   |
| TotalMarks         | decimal(3,0) | YES  |     | 100     | DEFAULT_GENERATED |
| MarksObtained      | decimal(3,0) | YES  |     | NULL    |                   |
+--------------------+-------------+------+-----+---------+-------------------+
6 rows in set (0.02 sec)
```

Alter Command:

- alter table student1 add FatherName varchar(20) after StudentName;
- desc student1;
- alter table studentdetails drop column RegistrationNumber; (Incase if you want to delete any attribute from the table)

```
mysql> alter table student1 add FatherName varchar(20) after StudentName;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc student1;
+--------------------+--------------+------+-----+---------+-------------------+
| Field              | Type         | Null | Key | Default | Extra             |
+--------------------+--------------+------+-----+---------+-------------------+
| RollNumber         | decimal(4,0) | NO   | PRI | NULL    |                   |
| StudentName        | varchar(20)  | NO   |     | NULL    |                   |
| FatherName         | varchar(20)  | YES  |     | NULL    |                   |
| RegistrationNumber | decimal(5,0) | YES  | UNI | NULL    |                   |
| Course             | varchar(10)  | YES  |     | NULL    |                   |
| TotalMarks         | decimal(3,0) | YES  |     | 100     | DEFAULT_GENERATED |
| MarksObtained      | decimal(3,0) | YES  |     | NULL    |                   |
+--------------------+--------------+------+-----+---------+-------------------+
7 rows in set (0.00 sec)

mysql>
```

Rename Command:

- Rename table student1 to StudentDetails;
- alter table student2 rename column Name to StudentName; **(to change the column name in a table)**
- show tables;

```
mysql> rename table student1 to StudentDetails;
Query OK, 0 rows affected (0.01 sec)

mysql> show tables;
+------------------+
| Tables_in_simran |
+------------------+
| student          |
| studentdetails   |
+------------------+
2 rows in set (0.01 sec)
```

Drop Command:

- drop table student;
- show tables;

```
mysql> drop table student;
Query OK, 0 rows affected (0.01 sec)

mysql> show tables;
+------------------+
| Tables_in_simran |
+------------------+
| studentdetails   |
+------------------+
1 row in set (0.00 sec)
```

- To delete database= **drop database simran**;

Truncate command:

This command will remove all rows from the studentdetails table, but the table structure will remain intact. It's important to note that TRUNCATE cannot be used on tables that are referenced by foreign key constraints unless you disable or remove the constraints temporarily.

- Truncate table studentdetails;

DML:

Now we already have tables let's insert values into cells of this table;

1. **Insert Command:**
   insert into studentdetails values (101, 'Sanjay', 'Rajender Singh', 2020, 'B.Tech', 500, 469);
   Select * from studentdetails;

```
mysql> insert into studentdetails values(101, 'Sanjay',  'Rajender Singh', 2020, 'B.Tech', 500, 469);
Query OK, 1 row affected (0.01 sec)

mysql> select * from studentdetails;
+------------+-------------+----------------+--------------------+--------+------------+---------------+
| RollNumber | StudentName | FatherName     | RegistrationNumber | Course | TotalMarks | MarksObtained |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |        500 |           469 |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
1 row in set (0.00 sec)
```

2. Now in case of RollNumber is the primary key so it will not accept duplicate entry.

```
mysql> insert into studentdetails values(101, 'Rose',  'Rajesh Singh', 2021, 'B.Tec
h', 500, 459);
ERROR 1062 (23000): Duplicate entry '101' for key 'studentdetails.PRIMARY'
mysql>
```

3. Now in registration number we mention it to be unique, so here it will show error.
   insert into studentdetails values(103, 'Sanchit', 'Rupal Kumar', 2021, 'B.Tech', 500, 359);

```
mysql> insert into studentdetails values(102, 'Rose',  'Rajesh Singh', 2021, 'B.Tec
h', 500, 459);
Query OK, 1 row affected (0.01 sec)

mysql> insert into studentdetails values(103, 'Sanchit',  'Rupal Kumar', 2021, 'B.T
ech', 500, 359);
ERROR 1062 (23000): Duplicate entry '2021' for key 'studentdetails.RegistrationNumb
er'
mysql>
```

4. Now if you don't mention any value for registrationNumber again it will show the error.

   insert into studentdetails values(103, 'Sanchit', 'Rupal Kumar', , 'B.Tech', 500, 359);

```
mysql> insert into studentdetails values(103, 'Sanchit',  'Rupal Kumar', , 'B.Tech', 500, 359);
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ', 'B.Tech',
00, 359)' at line 1
mysql>
```

5. But if we will mention NULL it will accept it.

   insert into studentdetails values(103, 'Sanchit', 'Rupal Kumar', null, 'B.Tech', 500, 359);

```
mysql> select * from studentdetails;
+------------+-------------+----------------+--------------------+--------+------------+---------------+
| RollNumber | StudentName | FatherName     | RegistrationNumber | Course | TotalMarks | MarksObtained |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |        500 |           469 |
|        102 | Rose        | Rajesh Singh   |               2021 | B.Tech |        500 |           459 |
|        103 | Sanchit     | Rupal Kumar    |               NULL | B.Tech |        500 |           359 |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
3 rows in set (0.00 sec)

mysql>
```

6. If we want to add values for two attributes instead of adding for all

   insert into studentdetails(RollNumber,StudentName) values(104, 'Abhishek');

```
mysql> insert into studentdetails(RollNumber,StudentName) values(104, 'Abhishek');
Query OK, 1 row affected (0.00 sec)

mysql> select * from studentdetails;
+------------+-------------+----------------+--------------------+--------+------------+---------------+
| RollNumber | StudentName | FatherName     | RegistrationNumber | Course | TotalMarks | MarksObtained |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |        500 |           469 |
|        102 | Rose        | Rajesh Singh   |               2021 | B.Tech |        500 |           459 |
|        103 | Sanchit     | Rupal Kumar    |               NULL | B.Tech |        500 |           359 |
|        104 | Abhishek    | NULL           |               NULL | NULL   |        100 |          NULL |
+------------+-------------+----------------+--------------------+--------+------------+---------------+
4 rows in set (0.00 sec)
```

   Here TotalMarks are not null because we already set its default value as 100

7. We used check(MarksObtained<=TotalMarks)
   insert into studentdetails values(105, 'Samye', 'Rudra Kumar', 2019, 'B.Tech', 500, 600);

```
4 rows in set (0.00 sec)

mysql> insert into studentdetails values(105, 'Samye',  'Rudra Kumar', 2019, 'B.Tec
h', 500, 600);
ERROR 3819 (HY000): Check constraint 'studentdetails_chk_1' is violated.
mysql>
```

Lets use correct command-
insert into studentdetails values(105, 'Samye', 'Rudra Kumar', 2019, 'B.Tech', 500, 499);

```
h', 500, 600);
ERROR 3819 (HY000): Check constraint 'studentdetails_chk_1' is violated.
mysql> insert into studentdetails values(105, 'Samye',  'Rudra Kumar', 2019, 'B.Tec
h', 500, 499);
Query OK, 1 row affected (0.00 sec)

mysql> select * from studentdetails;
+------------+-------------+---------------+--------------------+--------+--------
----+---------------+
| RollNumber | StudentName | FatherName    | RegistrationNumber | Course | TotalMa
rks | MarksObtained |
+------------+-------------+---------------+--------------------+--------+--------
----+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |
500 |           469 |
|        102 | Rose        | Rajesh Singh   |               2021 | B.Tech |
500 |           459 |
|        103 | Sanchit     | Rupal Kumar    |               NULL | B.Tech |
500 |           359 |
|        104 | Abhishek    | NULL           |               NULL | NULL   |
100 |          NULL |
|        105 | Samye       | Rudra Kumar    |               2019 | B.Tech |
500 |           499 |
+------------+-------------+---------------+--------------------+--------+--------
----+---------------+
5 rows in set (0.00 sec)
```

Update Command:

update studentdetails set FatherName='Prabh Singh' where RollNumber=104;
select * from studentdetails;

```
mysql> update studentdetails set FatherName='Prabh Singh' where RollNumber=104;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from studentdetails;
+------------+-------------+---------------+--------------------+--------+------------+---------------+
| RollNumber | StudentName | FatherName    | RegistrationNumber | Course | TotalMarks | MarksObtained |
+------------+-------------+---------------+--------------------+--------+------------+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |        500 |           469 |
|        102 | Rose        | Rajesh Singh   |               2021 | B.Tech |        500 |           459 |
|        103 | Sanchit     | Rupal Kumar    |               NULL | B.Tech |        500 |           359 |
|        104 | Abhishek    | Prabh Singh    |               NULL | NULL   |        100 |          NULL |
|        105 | Samye       | Rudra Kumar    |               2019 | B.Tech |        500 |           499 |
+------------+-------------+---------------+--------------------+--------+------------+---------------+
5 rows in set (0.00 sec)

mysql>
```

Delete Command:

If want to delete any record from the table

delete from studentdetails where RollNumber=103;
select * from studentdetails;

```
mysql> delete from studentdetails where RollNumber=103;
Query OK, 1 row affected (0.01 sec)

mysql> select * from studentdetails;
+------------+-------------+---------------+--------------------+--------+------------+---------------+
| RollNumber | StudentName | FatherName    | RegistrationNumber | Course | TotalMarks | MarksObtained |
+------------+-------------+---------------+--------------------+--------+------------+---------------+
|        101 | Sanjay      | Rajender Singh |               2020 | B.Tech |        500 |           469 |
|        102 | Rose        | Rajesh Singh   |               2021 | B.Tech |        500 |           459 |
|        104 | Abhishek    | Prabh Singh    |               2018 | NULL   |        100 |          NULL |
|        105 | Samye       | Rudra Kumar    |               2019 | B.Tech |        500 |           499 |
+------------+-------------+---------------+--------------------+--------+------------+---------------+
4 rows in set (0.00 sec)
```

Record is deleted now.

Compound WHERE Clause with multiple AND & OR Conditions:

- Let’s introduce a new column in the studentdetails table
- alter table studentdetails add SemesterFee int after Course;
- desc studentdetails;

```
mysql> desc studentdetails;
+---------------+--------------+------+-----+---------+-------------------+
| Field         | Type         | Null | Key | Default | Extra             |
+---------------+--------------+------+-----+---------+-------------------+
| RollNumber    | decimal(4,0) | NO   | PRI | NULL    |                   |
| Name          | varchar(20)  | NO   |     | NULL    |                   |
| FatherName    | varchar(20)  | YES  |     | NULL    |                   |
| MotherName    | varchar(20)  | YES  |     | NULL    |                   |
| Course        | varchar(10)  | YES  |     | NULL    |                   |
| SemesterFee   | int          | YES  |     | NULL    |                   |
| TotalMarks    | decimal(3,0) | YES  |     | 100     | DEFAULT_GENERATED |
| MarksObtained | decimal(3,0) | YES  |     | NULL    |                   |
+---------------+--------------+------+-----+---------+-------------------+
8 rows in set (0.01 sec)
```

- Now table will look like this:

```
mysql> select * from studentdetails;
+------------+--------+--------------+-------------+--------+-------------+------------+---------------+
| RollNumber | Name   | FatherName   | MotherName  | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+--------+--------------+-------------+--------+-------------+------------+---------------+
|          1 | Rose   | Sanjay Ghosh | Bharti      | B.Tech |      320000 |        100 |            87 |
|          2 | Rahul  | Sanjay tiwari | Laxmi      | B.Tech |      310000 |        100 |            82 |
|          3 | Nikhil | Pankaj Verma | Savitri devi | B.Tech |      320000 |        100 |            83 |
+------------+--------+--------------+-------------+--------+-------------+------------+---------------+
3 rows in set (0.00 sec)
```

Example 1: Using AND and Where to Filter Rows

```
mysql> select * from studentdetails where SemesterFee > 310000;
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
| RollNumber | Name   | FatherName   | MotherName   | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
|          1 | Rose   | Sanjay Ghosh | Bharti       | B.Tech |      320000 |        100 |            87 |
|          3 | Nikhil | Pankaj Verma | Savitri devi | B.Tech |      320000 |        100 |            83 |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
2 rows in set (0.00 sec)

mysql> select * from studentdetails where SemesterFee > 310000 AND MarksObtained = 83;
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
| RollNumber | Name   | FatherName   | MotherName   | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
|          3 | Nikhil | Pankaj Verma | Savitri devi | B.Tech |      320000 |        100 |            83 |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
1 row in set (0.00 sec)
```

```
mysql> select * from studentdetails where SemesterFee > 310000 and MarksObtained = 83;
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
| RollNumber | Name   | FatherName   | MotherName   | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
|          3 | Nikhil | Pankaj Verma | Savitri devi | B.Tech |      320000 |        100 |            83 |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
1 row in set (0.00 sec)

mysql> select * from studentdetails where SemesterFee > 310000 and MotherName = 'Savitri devi';
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
| RollNumber | Name   | FatherName   | MotherName   | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
|          3 | Nikhil | Pankaj Verma | Savitri devi | B.Tech |      320000 |        100 |            83 |
+------------+--------+--------------+--------------+--------+-------------+------------+---------------+
1 row in set (0.00 sec)
```

- select * from studentdetails where SemesterFee > 310000;
- select * from studentdetails where SemesterFee > 310000 AND MarksObtained = 83;
- select * from studentdetails where SemesterFee > 310000 and MarksObtained = 83;
- select * from studentdetails where SemesterFee > 310000 and MotherName = 'Savitri devi';

Example 1: Using WHERE,AND and OR to Filter Rows

Now the table is below mentioned:

```
mysql> select * from studentdetails;
+------------+----------+--------------+-----------------+--------+-------------+------------+---------------+
| RollNumber | Name     | FatherName   | MotherName      | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+----------+--------------+-----------------+--------+-------------+------------+---------------+
|          1 | Rose     | Sanjay Ghosh | Bharti          | B.Tech |      320000 |        100 |            87 |
|          2 | Rahul    | Sanjay tiwari| Laxmi           | B.Tech |      310000 |        100 |            82 |
|          3 | Nikhil   | Pankaj Verma | Savitri devi    | B.Tech |      320000 |        100 |            83 |
|          4 | Sanchit  | Rupal Kumar  | Vimla devi      | B.Tech |      400000 |        100 |            73 |
|          5 | Prabhaker| Rupesh Sharma| Savita devi     | B.Tech |      410000 |        100 |            72 |
|          6 | Parth    | Pawan Sharma | Sanjana devi    | Pharma |      410000 |        100 |            80 |
|          7 | Firoz    | Pravesh Kumar| Sambhawana devi | Pharma |      410000 |        100 |            81 |
+------------+----------+--------------+-----------------+--------+-------------+------------+---------------+
7 rows in set (0.00 sec)
```

- select * from studentdetails where (SemesterFee>=400000 and Course= 'Pharma') or SemesterFee>410000;

```
mysql> select * from studentdetails where (SemesterFee>=400000 and Course= 'Pharma' ) or SemesterFee>410000;
+------------+-------+--------------+----------------+--------+-------------+------------+---------------+
| RollNumber | Name  | FatherName   | MotherName     | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+-------+--------------+----------------+--------+-------------+------------+---------------+
|          6 | Parth | Pawan Sharma | Sanjana devi   | Pharma |      410000 |        100 |            80 |
|          7 | Firoz | Pravesh Kumar | Sambhawana devi | Pharma |      410000 |        100 |            81 |
+------------+-------+--------------+----------------+--------+-------------+------------+---------------+
2 rows in set (0.00 sec)
```

- select * from studentdetails where (SemesterFee>=400000 and Course= 'B.Tech') or SemesterFee>410000;

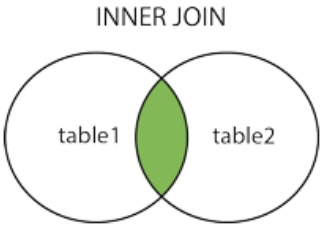
```
mysql> select * from studentdetails where (SemesterFee>=400000 and Course= 'B.Tech' ) or SemesterFee>410000;
+------------+-----------+--------------+-------------+--------+-------------+------------+---------------+
| RollNumber | Name      | FatherName   | MotherName  | Course | SemesterFee | TotalMarks | MarksObtained |
+------------+-----------+--------------+-------------+--------+-------------+------------+---------------+
|          4 | Sanchit   | Rupal Kumar  | Vimla devi  | B.Tech |      400000 |        100 |            73 |
|          5 | Prabhaker | Rupesh Sharma | Savita devi | B.Tech |      410000 |        100 |            72 |
+------------+-----------+--------------+-------------+--------+-------------+------------+---------------+
2 rows in set (0.00 sec)
```

JOINS:

SQL joins allow you to combine rows from two or more tables based on a related column between them. Here are some common SQL join types in MySQL with examples:

1. INNER JOIN:

The INNER JOIN retrieves records that have matching values in both tables.



- Create two tables having a foreign Key
- **Table1: employees**
- Create table employees (Emp_ID int primary key, Name varchar(20), Dept_ID int);
- insert into employees values(1, 'Prakash', 101);
- insert into employees(Emp_ID, Name, Dept_ID) values(2, 'Pritosh', 102),(3, 'Rimple', 103),(4, 'Ridhima', 104);

```
mysql> select * from employees;
+--------+---------+---------+
| Emp_ID | Name    | Dept_ID |
+--------+---------+---------+
|      1 | Prakash |     101 |
|      2 | Pritosh |     102 |
|      3 | Rimple  |     103 |
|      4 | Ridhima |     104 |
|      5 | Priya   |     101 |
|      6 | Ridha   |     101 |
|      7 | Seema   |     102 |
+--------+---------+---------+
7 rows in set (0.00 sec)
```

- **Table2 :departments**
- Create table departments (Dept_ID int primary key, Dept_Name varchar(20));
- insert into departments(Dept_ID, Dept_Name) values(101, 'HR'),(102, 'HR'),(103, 'Sales'),(104, 'Sales');

```
mysql> select * from departments;
+---------+-----------+
| Dept_ID | Dept_Name |
+---------+-----------+
|     101 | HR        |
|     102 | HR        |
|     103 | Sales     |
|     104 | Sales     |
+---------+-----------+
4 rows in set (0.00 sec)
```

- **INNER JOIN:**
  To retrieve a list of employees along with their department names (only employees who belong to a department):

  - select employees.Name, departments.Dept_Name from employees inner join departments on employees.Dept_ID = departments.Dept_ID;

```
mysql> select employees.Name, departments.Dept_Name from employees inner join departments on employees
+---------+------------+
| Name    | Dept_Name  |
+---------+------------+
| Prakash | HR         |
| Pritosh | Sales      |
| Rimple  | marketing  |
| Ridhima | Production |
| Priya   | HR         |
| Ridha   | HR         |
| Seema   | Sales      |
+---------+------------+
7 rows in set (0.00 sec)
```

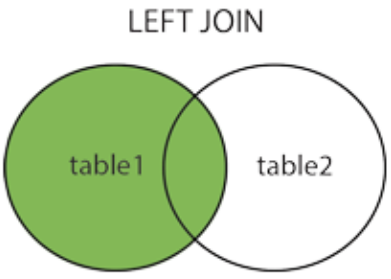
OR

- select * from employees as emp inner join departments as dept on emp.Dept_ID = dept.Dept_ID;
    - Here * is representing that in resulting table we want to see all the attributes of tables
    - AS is an alias here that helps to give a short / Temporary name
    - By using ON clause we mention that which two attributes in our employees and department table are common to perform inner join
    - In output we will see all attributes and the tuples which are common in both tables

```
mysql> select * from employees as emp inner join departments as  dept on emp.Dept_ID = dept.Dept_ID;
+--------+---------+---------+---------+-----------+
| Emp_ID | Name    | Dept_ID | Dept_ID | Dept_Name |
+--------+---------+---------+---------+-----------+
|      1 | Prakash |     101 |     101 | HR        |
|      2 | Pritosh |     102 |     102 | Sales     |
|      3 | Rimple  |     103 |     103 | marketing |
|      4 | Ridhima |     104 |     104 | Production |
|      5 | Priya   |     101 |     101 | HR        |
|      6 | Ridha   |     101 |     101 | HR        |
|      7 | Seema   |     102 |     102 | Sales     |
+--------+---------+---------+---------+-----------+
7 rows in set (0.00 sec)
```

LEFT JOIN:

In database management systems (DBMS), a LEFT JOIN, also known as a LEFT OUTER JOIN, is a type of SQL join operation that combines rows from two or more tables based on a related column, and it returns all rows from the left table (table1) and matching rows from the right table (table2). **If there is no match found in the right table, NULL values are returned for columns from the right table.**



- select employees.Name,employees.Emp_ID, departments.Dept_Name, departments.Dept_ID from employees left join departments on employees.Dept_ID = departments.Dept_ID;

```
mysql> select employees.Name,employees.Emp_ID, departments.Dept_Name, dep
artments.Dept_ID from employees left join departments on employees.Dept_I
D = departments.Dept_ID;
+---------+--------+------------+---------+
| Name    | Emp_ID | Dept_Name  | Dept_ID |
+---------+--------+------------+---------+
| Prakash |      1 | HR         |     101 |
| Pritosh |      2 | Sales      |     102 |
| Rimple  |      3 | marketing  |     103 |
| Ridhima |      4 | Production |     104 |
| Priya   |      5 | HR         |     101 |
| Ridha   |      6 | HR         |     101 |
| Seema   |      7 | Sales      |     102 |
| ritika  |      8 | NULL       |    NULL |
| manya   |      9 | NULL       |    NULL |
+---------+--------+------------+---------+
9 rows in set (0.00 sec)
```

- select employees.Name,employees.Emp_ID, departments.Dept_Name, departments.Dept_ID from employees inner join departments on employees.Dept_ID = departments.Dept_ID;

```
mysql> select employees.Name,employees.Emp_ID, departments.Dept_Name, dep
artments.Dept_ID from employees inner join departments on employees.Dept_
ID = departments.Dept_ID;
+---------+--------+------------+---------+
| Name    | Emp_ID | Dept_Name  | Dept_ID |
+---------+--------+------------+---------+
| Prakash |      1 | HR         |     101 |
| Pritosh |      2 | Sales      |     102 |
| Rimple  |      3 | marketing  |     103 |
| Ridhima |      4 | Production |     104 |
| Priya   |      5 | HR         |     101 |
| Ridha   |      6 | HR         |     101 |
| Seema   |      7 | Sales      |     102 |
+---------+--------+------------+---------+
7 rows in set (0.00 sec)
```

Or

Inner Join

select employees.Name, departments.Dept_Name from employees inner join departments on employees.Dept_ID = departments.Dept_ID;

```
mysql> select employees.Name, departments.Dept_Name from employees inner
join departments on employees.Dept_ID = departments.Dept_ID;
+---------+------------+
| Name    | Dept_Name  |
+---------+------------+
| Prakash | HR         |
| Pritosh | Sales      |
| Rimple  | marketing  |
| Ridhima | Production |
| Priya   | HR         |
| Ridha   | HR         |
| Seema   | Sales      |
+---------+------------+
7 rows in set (0.00 sec)
```

Left Join:

select employees.Name, departments.Dept_Name from employees left join departments on employees.Dept_ID = departments.Dept_ID;

```
mysql> select employees.Name, departments.Dept_Name from employees left j
oin departments on employees.Dept_ID = departments.Dept_ID;
+---------+------------+
| Name    | Dept_Name  |
+---------+------------+
| Prakash | HR         |
| Pritosh | Sales      |
| Rimple  | marketing  |
| Ridhima | Production |
| Priya   | HR         |
| Ridha   | HR         |
| Seema   | Sales      |
| ritika  | NULL       |
| manya   | NULL       |
+---------+------------+
9 rows in set (0.00 sec)
```

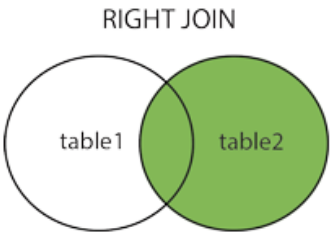
OR

- select * from employees as emp left join departments as dept on emp.Dept_ID = dept.Dept_ID;
    - Here need to note that after ON clause whatever the table we will mention (**emp.Dept_ID** = dept.Dept_ID;) on that table only the left join will get applied.
    - Like now it will show all the data of emp table and just the common that of dept table

```
mysql> select * from employees as emp left join departments as  dept on emp.Dept_ID = dept.Dept_ID;
+--------+---------+---------+---------+------------+
| Emp_ID | Name    | Dept_ID | Dept_ID | Dept_Name  |
+--------+---------+---------+---------+------------+
|      1 | Prakash |     101 |     101 | HR         |
|      2 | Pritosh |     102 |     102 | Sales      |
|      3 | Rimple  |     103 |     103 | marketing  |
|      4 | Ridhima |     104 |     104 | Production |
|      5 | Priya   |     101 |     101 | HR         |
|      6 | Ridha   |     101 |     101 | HR         |
|      7 | Seema   |     102 |     102 | Sales      |
|      8 | ritika  |    NULL |    NULL | NULL       |
|      9 | manya   |    NULL |    NULL | NULL       |
+--------+---------+---------+---------+------------+
9 rows in set (0.00 sec)
```

Right Join:

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). **The result is 0 records from the left side, if there is no match.**



Now the tables look like this:

```
mysql> select * from departments;
+---------+-----------------+
| Dept_ID | Dept_Name       |
+---------+-----------------+
|     101 | HR              |
|     102 | Sales           |
|     103 | marketing       |
|     104 | Production      |
|     105 | Accounts        |
|     106 | Design Thinking |
+---------+-----------------+
6 rows in set (0.00 sec)

mysql> select * from employees;
+--------+---------+---------+
| Emp_ID | Name    | Dept_ID |
+--------+---------+---------+
|      1 | Prakash |     101 |
|      2 | Pritosh |     102 |
|      3 | Rimple  |     103 |
|      4 | Ridhima |     104 |
|      5 | Priya   |     101 |
|      6 | Ridha   |     101 |
|      7 | Seema   |     102 |
|      8 | ritika  |    NULL |
|      9 | manya   |    NULL |
+--------+---------+---------+
9 rows in set (0.00 sec)
```

- select employees.Name, departments.Dept_Name from employees right join departments on employees.Dept_ID = departments.Dept_ID;

```
+---------+-----------------+
| Name    | Dept_Name       |
+---------+-----------------+
| Ridha   | HR              |
| Priya   | HR              |
| Prakash | HR              |
| Seema   | Sales           |
| Pritosh | Sales           |
| Rimple  | marketing       |
| Ridhima | Production      |
| NULL    | Accounts        |
| NULL    | Design Thinking |
+---------+-----------------+
9 rows in set (0.00 sec)
```

Here we can see it is showing null values for the record that doesn't matches in Table 1 (Employees).

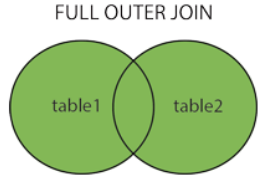
OR

```
mysql> select * from employees as emp right join departments as  dept on emp.Dept_ID = dept.Dept_ID;
+--------+---------+---------+---------+-----------------+
| Emp_ID | Name    | Dept_ID | Dept_ID | Dept_Name       |
+--------+---------+---------+---------+-----------------+
|      6 | Ridha   |     101 |     101 | HR              |
|      5 | Priya   |     101 |     101 | HR              |
|      1 | Prakash |     101 |     101 | HR              |
|      7 | Seema   |     102 |     102 | Sales           |
|      2 | Pritosh |     102 |     102 | Sales           |
|      3 | Rimple  |     103 |     103 | marketing       |
|      4 | Ridhima |     104 |     104 | Production      |
|   NULL | NULL    |    NULL |     105 | Accounts        |
|   NULL | NULL    |    NULL |     106 | Design Thinking |
+--------+---------+---------+---------+-----------------+
9 rows in set (0.00 sec)
```

- select * from employees as emp right join departments as dept on emp.Dept_ID = dept.Dept_ID;

Full Join :

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.



| Emp_ID | Name | Dept_ID |

+--------+---------+---------+

| 1 | Prakash | 101 |

| 2 | Pritosh | 102 |

| 3 | Rimple | 103 |

| 4 | Ridhima | 104 |

| 5 | Priya | 101 |

| 6 | Ridha | 101 |

| 7 | Seema | 102 |

| 8 | ritika | NULL |

| 9 | manya | NULL |

Table2:

select * from departments;

+---------+-----------------+

| Dept_ID | Dept_Name |

+---------+-----------------+

| 101 | HR |

| 102 | Sales |

| 103 | marketing |

| 104 | Production |

| 105 | Accounts |

| 106 | Design Thinking |

+---------+-----------------+

- select * from employees as emp left outer join departments as dept on emp.Dept_ID = dept.Dept_ID **union** select * from employees as emp right join departments as dept on emp.Dept_ID = dept.Dept_ID;
- Here for seeing the result we use UNION
- UNION= it simply just combines two tables
- It doesn't includes duplicate values in the resulting table.

```
mysql> select * from employees as emp left outer join departments as  dept on emp.Dept_ID = dept.Dept_I
D union select * from employees as emp right join departments as dept on emp.Dept_ID = dept.Dept_ID;
+--------+---------+---------+---------+----------------+
| Emp_ID | Name    | Dept_ID | Dept_ID | Dept_Name      |
+--------+---------+---------+---------+----------------+
|      1 | Prakash |     101 |     101 | HR             |
|      2 | Pritosh |     102 |     102 | Sales          |
|      3 | Rimple  |     103 |     103 | marketing      |
|      4 | Ridhima |     104 |     104 | Production     |
|      5 | Priya   |     101 |     101 | HR             |
|      6 | Ridha   |     101 |     101 | HR             |
|      7 | Seema   |     102 |     102 | Sales          |
|      8 | ritika  |    NULL |    NULL | NULL           |
|      9 | manya   |    NULL |    NULL | NULL           |
|   NULL | NULL    |    NULL |     105 | Accounts       |
|   NULL | NULL    |    NULL |     106 | Design Thinking |
+--------+---------+---------+---------+----------------+
```

Self Join:

A self-join is a SQL query in which a table is joined with itself. In other words, you use the same table twice within a single SQL statement, treating it as if it were two separate tables. Self-joins are often used to model hierarchical or recursive relationships within a table.

```
mysql> select * from employees;
+--------+---------+---------+------------+
| Emp_ID | Name    | Dept_ID | manager_ID |
+--------+---------+---------+------------+
|      1 | Prakash |     101 |          7 |
|      2 | Pritosh |     102 |          6 |
|      3 | Rimple  |     103 |          5 |
|      4 | Ridhima |     104 |          1 |
|      5 | Priya   |     101 |          4 |
|      6 | Ridha   |     101 |          3 |
|      7 | Seema   |     102 |          2 |
|      8 | ritika  |    NULL |       NULL |
|      9 | manya   |    NULL |       NULL |
+--------+---------+---------+------------+
```

- Now let's say this is new table and for Emp_ID there is a manager with ID = 7
- If we check whole table Seema is the manager of Prakash with Emp_ID= 1
- Now if for better clarity I want to see all details of Seema as a manager of Prakash in a single tuple then we do that by self join

- select * from employees as t1 join employees as t2 on t2.Emp_ID = t1.manager_ID;
    - here join represents inner join

```
mysql> select * from employees as t1 join employees as t2 on t2.Emp_ID = t1.manager_ID;
+--------+---------+---------+------------+--------+---------+---------+------------+
| Emp_ID | Name    | Dept_ID | manager_ID | Emp_ID | Name    | Dept_ID | manager_ID |
+--------+---------+---------+------------+--------+---------+---------+------------+
|      1 | Prakash |     101 |          7 |      7 | Seema   |     102 |          2 |
|      2 | Pritosh |     102 |          6 |      6 | Ridha   |     101 |          3 |
|      3 | Rimple  |     103 |          5 |      5 | Priya   |     101 |          4 |
|      4 | Ridhima |     104 |          1 |      1 | Prakash |     101 |          7 |
|      5 | Priya   |     101 |          4 |      4 | Ridhima |     104 |          1 |
|      6 | Ridha   |     101 |          3 |      3 | Rimple  |     103 |          5 |
|      7 | Seema   |     102 |          2 |      2 | Pritosh |     102 |          6 |
+--------+---------+---------+------------+--------+---------+---------+------------+
7 rows in set (0.00 sec)
```

- select t1.Name as employee_name, t2.Name as manager_name from employees as t1 join employees as t2 on t2.Emp_ID=t1.manager_ID;

```
mysql> select  t1.Name as employee_name, t2.Name as manager_name from employees as t1 join employees as
 t2 on t2.Emp_ID=t1.manager_ID;
+--------------+--------------+
| employee_name | manager_name |
+--------------+--------------+
| Prakash       | Seema        |
| Pritosh       | Ridha        |
| Rimple        | Priya        |
| Ridhima       | Prakash      |
| Priya         | Ridhima      |
| Ridha         | Rimple       |
| Seema         | Pritosh      |
+--------------+--------------+
7 rows in set (0.00 sec)
```

Here if we want to just see employee name and his/her manager name then we will use the upper mentioned command.

UNION:

Now we have two tables employees and employees2 having an entry common in between

If we use union to join these two tables it will not include duplicate entries

```
mysql> select * from employees;
+--------+---------+---------+------------+
| Emp_ID | Name    | Dept_ID | manager_ID |
+--------+---------+---------+------------+
|      1 | Prakash |     101 |          7 |
|      2 | Pritosh |     102 |          6 |
|      3 | Rimple  |     103 |          5 |
|      4 | Ridhima |     104 |          1 |
|      5 | Priya   |     101 |          4 |
|      6 | Ridha   |     101 |          3 |
|      7 | Seema   |     102 |          2 |
|      8 | ritika  |    NULL |       NULL |
|      9 | manya   |    NULL |       NULL |
+--------+---------+---------+------------+
9 rows in set (0.00 sec)

mysql> select * from employees2;
+--------+---------+---------+
| Emp_ID | Name    | Dept_ID |
+--------+---------+---------+
|      1 | dolly   |     101 |
|      2 | Pritosh |     102 |
+--------+---------+---------+
```

- select employees.Emp_ID, employees.Name from employees union select employees2.Emp_ID, employees2.Name from employees2;

```
mysql> select employees.Emp_ID, employees.Name from employees union select employees2.Emp_ID, employees
2.Name from employees2;
+--------+---------+
| Emp_ID | Name    |
+--------+---------+
|      1 | Prakash |
|      2 | Pritosh |
|      3 | Rimple  |
|      4 | Ridhima |
|      5 | Priya   |
|      6 | Ridha   |
|      7 | Seema   |
|      8 | ritika  |
|      9 | manya   |
|      1 | dolly   |
+--------+---------+
10 rows in set (0.00 sec)
```

- Here we can see earlier if we count the total entries in employees and employees2 table it was 11
- But when we use union to combine these tables it didn't show the duplicate entry and total count of entries is 10

UNION ALL

- select employees.Emp_ID, employees.Name from employees union all select employees2.Emp_ID, employees2.Name from employees2;

```
mysql> select employees.Emp_ID, employees.Name from employees union all select employees2.Emp_ID, emplo
yees2.Name from employees2;
+--------+---------+
| Emp_ID | Name    |
+--------+---------+
|      1 | Prakash |
|      2 | Pritosh |
|      3 | Rimple  |
|      4 | Ridhima |
|      5 | Priya   |
|      6 | Ridha   |
|      7 | Seema   |
|      8 | ritika  |
|      9 | manya   |
|      1 | dolly   |
|      2 | Pritosh |
+--------+---------+
11 rows in set (0.00 sec)
```

Cross Join:-

CROSS JOIN is also known as the Cartesian product / Cartesian join.

- A cross join, also known as a Cartesian join or a cross product, is a type of join operation in relational database management systems (RDBMS). It combines every row from one table with every row from another table, resulting in a Cartesian product of the two

tables. Unlike other join types like inner joins or outer joins, cross joins do not require a specific condition or criteria for matching rows; they simply combine all possible combinations.

- It simply multiplies each row of table 1 with each row of table 2
- Example :

```
mysql> select * from employees;
+--------+---------+---------+------------+
| Emp_ID | Name    | Dept_ID | manager_ID |
+--------+---------+---------+------------+
|      1 | Prakash |     101 |          7 |
|      2 | Pritosh |     102 |          6 |
|      3 | Rimple  |     103 |          5 |
|      4 | Ridhima |     104 |          1 |
|      5 | Priya   |     101 |          4 |
|      6 | Ridha   |     101 |          3 |
|      7 | Seema   |     102 |          2 |
|      8 | ritika  |    NULL |       NULL |
|      9 | manya   |    NULL |       NULL |
+--------+---------+---------+------------+
9 rows in set (0.00 sec)

mysql> select * from employees2;
+--------+---------+---------+
| Emp_ID | Name    | Dept_ID |
+--------+---------+---------+
|      1 | dolly   |     101 |
|      2 | Pritosh |     102 |
+--------+---------+---------+
```

- select * from employees cross join employees2;

```
mysql> select * from employees cross join employees2;
+--------+---------+---------+------------+--------+---------+---------+
| Emp_ID | Name    | Dept_ID | manager_ID | Emp_ID | Name    | Dept_ID |
+--------+---------+---------+------------+--------+---------+---------+
|      1 | Prakash |     101 |          7 |      2 | Pritosh |     102 |
|      1 | Prakash |     101 |          7 |      1 | dolly   |     101 |
|      2 | Pritosh |     102 |          6 |      2 | Pritosh |     102 |
|      2 | Pritosh |     102 |          6 |      1 | dolly   |     101 |
|      3 | Rimple  |     103 |          5 |      2 | Pritosh |     102 |
|      3 | Rimple  |     103 |          5 |      1 | dolly   |     101 |
|      4 | Ridhima |     104 |          1 |      2 | Pritosh |     102 |
|      4 | Ridhima |     104 |          1 |      1 | dolly   |     101 |
|      5 | Priya   |     101 |          4 |      2 | Pritosh |     102 |
|      5 | Priya   |     101 |          4 |      1 | dolly   |     101 |
|      6 | Ridha   |     101 |          3 |      2 | Pritosh |     102 |
|      6 | Ridha   |     101 |          3 |      1 | dolly   |     101 |
|      7 | Seema   |     102 |          2 |      2 | Pritosh |     102 |
|      7 | Seema   |     102 |          2 |      1 | dolly   |     101 |
|      8 | ritika  |    NULL |       NULL |      2 | Pritosh |     102 |
|      8 | ritika  |    NULL |       NULL |      1 | dolly   |     101 |
|      9 | manya   |    NULL |       NULL |      2 | Pritosh |     102 |
|      9 | manya   |    NULL |       NULL |      1 | dolly   |     101 |
+--------+---------+---------+------------+--------+---------+---------+
```