DBMS & SQL

Q. What is DBMS?

DBMS stands for database management system mean we can say DBMS provide the standard techniques to manage and organize user data.

Q. What is database?

Database is a collection of organized data and typically store and manage for future use as well as past use or for analysis purpose and may be another purpose called as database

Q. What are the benefits of database?

---

a) Permanent storage: if we think about database we can store data permanent on hard disk of system means we can use that in future as per our need

b) Access user data as per the requirement of user: means if we think about database we can access it as per our need as well as we can search data as per our need

c) Modify data in future as per the requirement of user

d) Provide security to user data: if we think about we provide role access to data means we can restrict users for access some specified data as well as we can grant use to access data this is dependent on situation called as data security

e) Provide global access of data to multiple users: if we think about cloud database or server database we can provide access to multiple users globally**.**

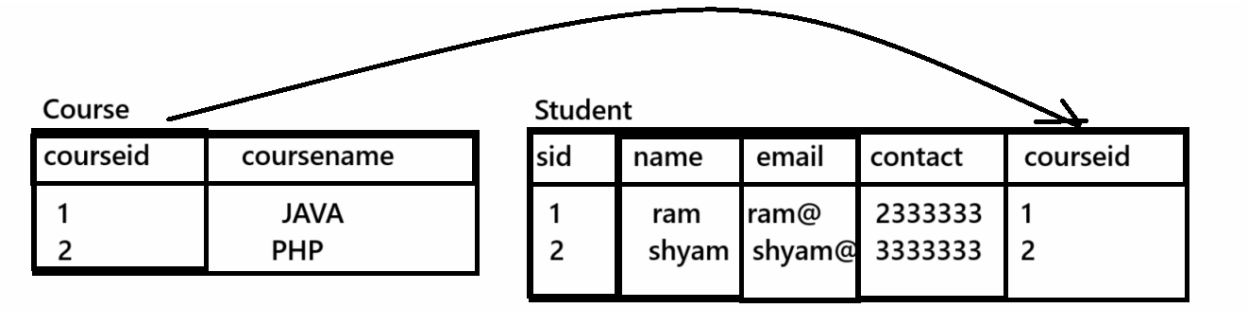
f) Provide different format for store data: means if we think about database we can store in different format like as using JSON format, using excel, rdbms etc

How many ways to manage data or Types of Databases

---

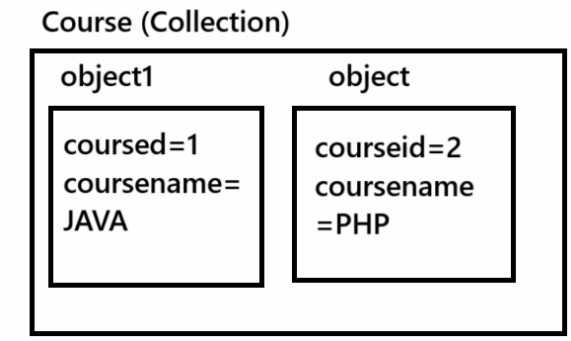
a) Relational Database Management System: if we think about this database we can store data in the form of tables/relation/entity as well as store in row/tuples/record and columns/attributes/field etc as well as can provide relationship multiple tables by keys like as primary key or foreign key etc

Example:



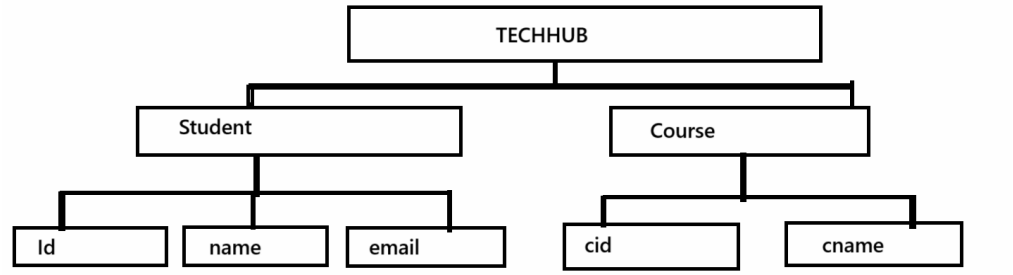
Example of Relational database management system MYSQL,Oracle,DB2,PostgreSQL etc

b) Object oriented database: If we store data in the form of objects and field called as object oriented database.

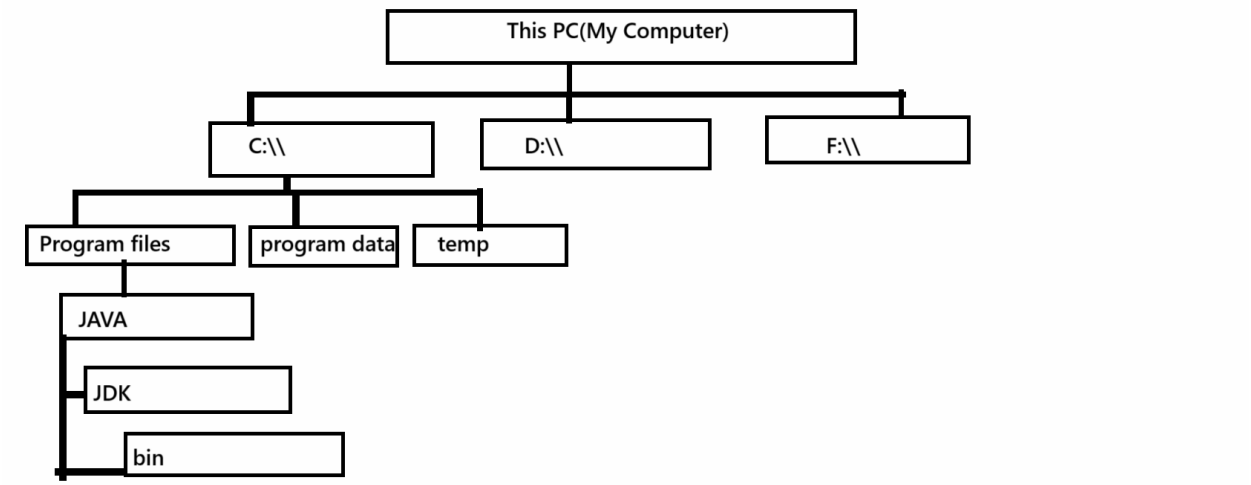


Example of object oriented database: Mango DB, Fire Base etc

c) Hierarchical Database: When we manage the data in the form of tree format called as Hierarchical database means in the form of parent and child format.

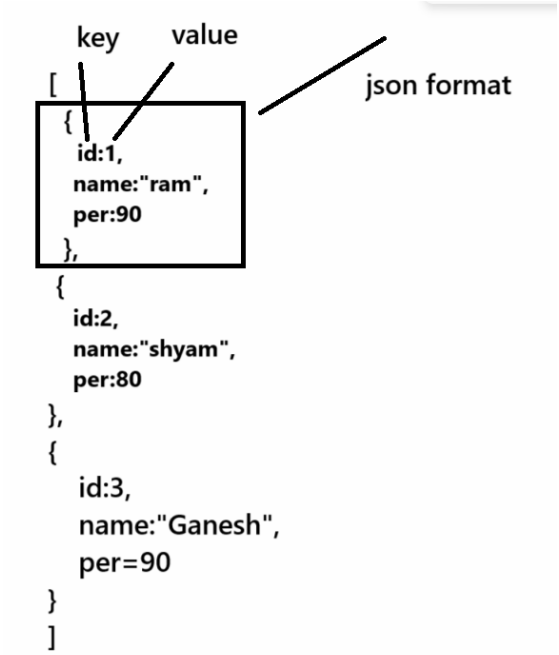


Example: Windows File Management system and folder management system is an example of Hierarchical database management system.



d) Distributed databases: Distributed database consist of two or more files or database located on multiple computes or scattered in different network component and connected with centralized database or data center called as distributed database

e) Document/JSON database: we can store data in the form of documents or json format also using key and value pair we can save in database also or we can store in other file format also called as document or json format data.



f) Cloud Database: cloud data means we have database on remote server or on cloud and we can use it centralized by multiple users and data not store on user machine but user can access it from remote locations called as cloud database

Example: Good drive, Microsoft one drive etc

etc

Q. What is SQL?

SQL stands for Structure Query Language basically it is programming language which can implement the relational database management concept means using SQL programming we can perform different operation on Relational database management system

Now we want to discuss about Relational Database management system

---

Steps to work with Relational Database Management System

1. Download the any Tool which support to Relational Database Management System

---

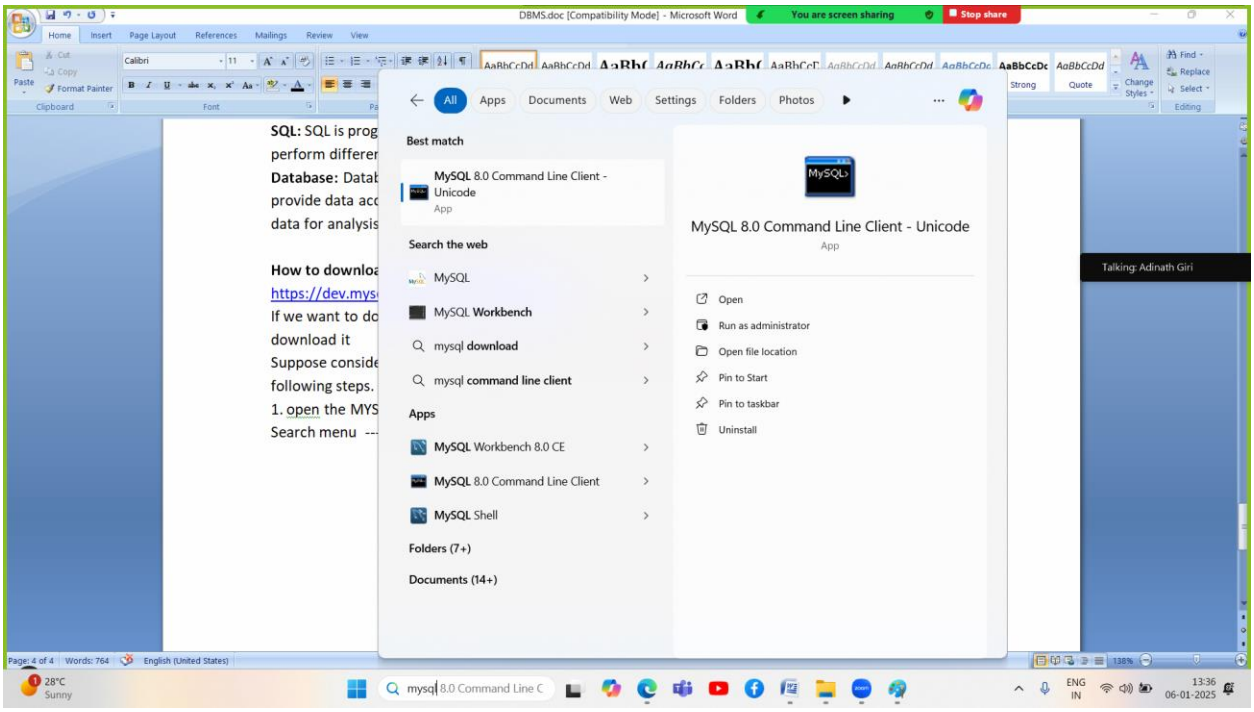
You can use Oracle or MYSQL etc for provide implementation to Relational Database Management System

So we want to use MYSQL as tool for work with Relational Database Management System.

Q. What is difference between MYSQL, SQL and Database?

MYSQL: MYSQL is vendor or Tool which provide implementation to RDBMS concept
SQL: SQL is programming language which work on RBMS concept means using SQL programming we can perform different operation on RDBMS
Database: Database is concept where we can store data and organize the data in different way and provide data access as per need of user and user can modify data in future as per his need and can use data for analysis purpose etc

How to download the MYSQL tool?
https://dev.mysql.com/downloads/installer/
If we want to download the MYSQL then visit above link and create account oracle and sign in oracle and download it
Suppose consider if we installed MYSQL then you can enter or login in MYSQL on your system using a following steps.
1. Open the MYSQL
Search menu ---- MYSQL 8.0 --- presenter

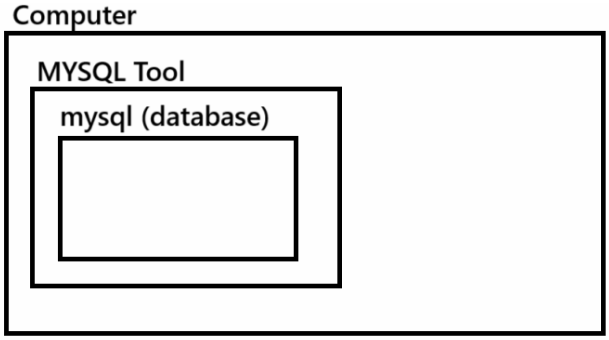


2. Enter password which we provide at the time of installation

```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.18 MySQL Community Server - GPL

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

After login we get above type of screen shot

Note: when we installed database we get one by default database from MYSQL tool name as mysql



So we need to use mysql database at initial level or enter in mysql database using a following command

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

Once we enter in MYSQL database we can create your own database or user defined database

Note: if we think about relational database management system then we can collections of tables called as database

User can create own database and for that we have command

Syntax: create database databasename;

```
mysql> use mysql;
Database changed
mysql> create database aug2024;
Query OK, 1 row affected (0.04 sec)

mysql>
```

After creating this database we have two databases present in MYSQL one is inbuilt name as mysql and one is used define names as aug2024

So if we want to create table under user defined database we required use user defined database or enter in user defined database.

Synax: user databasename;

Example: use aug2024;

```
mysql> use mysql;
Database changed
mysql> create database aug2024;
Query OK, 1 row affected (0.04 sec)

mysql> use aug2024;
Database changed
mysql>
```

We shift from MySQL database to aug2024 database.

Once we create own database we can work with database and if we want to work with database we have following types of command.

1. DDL: DDL stands for data definition language it is used for work with table structure means using this DDL command we can create table, drop table etc

Types of DDL command

a) create
b) alter
c) Drop
d) Truncate
e) desc

2. DML: DML stands Data manipulation language and it is used for work with table data means using DML we can insert record in database table, delete record from database table , update record in database table etc

Types of DML Command

a) Insert
b) Delete
c) Update
d) Select

3. DCL: Data Control language this command is used for provide grant to user or retrain grand or access of data from user by database administrator

a) grant
b revoke

4. TCL: Transaction Control Language and this command normally we use with DML statement

Types of command in TCL

a) commit
b) rollback
c) save point

Now we want to discuss about DDL command

---

1) Create command

Create command is used for create table, create procedure, create trigger, create index, create view and create database, function, cursor etc

How to create table by using create command

Syntax: create table tablename(columnname datatype(size), columnname datatype(size) constraints);

Example: we want to create employee table in aug2024 database with field id,name, and salary

```
mysql> use aug2024;
Database changed
mysql> create table employee(eid int(5),name varchar(200),sal int(5));
Query OK, 0 rows affected, 2 warnings (0.14 sec)

mysql>
```

If we want to see the table structure we have command name as desc

Desc command: Desc command can give the detail about structure means give detail columnname, its data type, size of data type and constrain on column etc

```
mysql> desc employee;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| eid   | int(5)       | YES  |     | NULL    |       |
| name  | varchar(200) | YES  |     | NULL    |       |
| sal   | int(5)       | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.10 sec)
```

Alter command: Alter command is used for modify the table structure means using alter command we can add new column in database table, remove column from database table, rename column name or modify column type or size etc

How to add new column in table using alter?

---

if we want to add new column in table using alter we have add option with alter statement

Syntax: alter table tablename add columnname datatype(size);

or

We can add more than one column at time using alter

Syntax: alter table tablename add(columnname datatype(size),columnname datatype(size))

Example: we have employee with three field given below

```
mysql> desc employee;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| eid   | int(5)       | YES  |     | NULL    |       |
| name  | varchar(200) | YES  |     | NULL    |       |
| sal   | int(5)       | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.10 sec)
```

We want to add two columns in table email and contact in existing table

```
mysql> alter table employee add column (email varchar(200),contact varchar(200));
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

After adding two columns your table structure like as

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| email   | varchar(200) | YES  |     | NULL    |       |
| contact | varchar(200) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
5 rows in set (0.03 sec)
```

if we compare after alter screen shot and before alter screen shot we added two new columns in employee table after alter

How to remove existing column from table using alter?

If we want to remove the existing column from table using alter we drop option

Syntax: alter table tablename drop columnname;

Example: we have employee table with 5 field or column id, name, sal, email and contact and we want to delete the email column

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| email   | varchar(200) | YES  |     | NULL    |       |
| contact | varchar(200) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
5 rows in set (0.03 sec)

mysql> alter table employee drop column email;
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| contact | varchar(200) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
```

Before drop table structure like as

After drop table structure like as

How to change size or data type of column using table?

If we want to change column type or column size using alter statement we have modify option

Syntax: alter table tablename modify column columnname datatype(size)

Example: we have contact column in above table with data type varchar and size 200 so we want to change data type of contact column from varchar to int and size from 200 to 10 then your alter statement like as

```
mysql> alter table employee modify column contact int(10);
Query OK, 0 rows affected, 1 warning (0.13 sec)
Records: 0  Duplicates: 0  Warnings: 1
```

Output

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| contact | varchar(200) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Note: before alter or modify

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| contact | int(10)      | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Note: after alter or modify statement

How to rename column using alter statement?

If we want to change name of column using alter we have rename option

Syntax: alter table tablename rename column old columnname to new column name;

Example: we have employee table with column id,name,sal,contact and we want to change column name from sal to salary then your query like as

```
mysql> alter table employee rename column sal to salary;
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Output

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| sal     | int(5)       | YES  |     | NULL    |       |
| contact | int(10)      | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

before rename column

```
mysql> desc employee;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| eid     | int(5)       | YES  |     | NULL    |       |
| name    | varchar(200) | YES  |     | NULL    |       |
| salary  | int(5)       | YES  |     | NULL    |       |
| contact | int(10)      | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

After rename column

Drop statement: drop command is used for remove table, view,procedure,trigger,function etc

How to delete table from database using drop

Syntax: drop table tablename;

Example: drop table employee;

```
mysql> drop table employee;
Query OK, 0 rows affected (0.02 sec)

mysql> desc employee;
ERROR 1146 (42S02): Table 'aug2024.employee' doesn't exist
mysql>
```

we get error aug2024.employee doesn't exist because we describe after deleting it so we delete table before desc using drop this is major reason we get error

truncate: truncate statement or command is used for delete the all data from table not table structure.

Syntax: truncate tablename;

```
mysql> truncate employee;
Query OK, 0 rows affected (0.07 sec)

mysql> desc employee;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| eid   | int(5)       | YES  |     | NULL    |       |
| name  | varchar(200) | YES  |     | NULL    |       |
| sal   | int(5)       | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select *from employee;
Empty set (0.00 sec)
```

Note: when we delete data using truncate we cannot recover it or rollback it

How to comment in SQL?

If we want to write comment in SQL we have following syntax

/* multi line comments

***/**

// - single line code

```
mysql> create table employee(eid int(5),name varchar(200),sal int(5));/* table creation*/

Query OK, 0 rows affected, 2 warnings (0.05 sec)

mysql>
```

DML statement

DML stands for data manipulation language using DML we can store data in table using insert command,delete data from table , update data from table as well as retrieve data from table.

How to insert data in database table?

---

for insert data in table we have insert statement

There are two types of insert statement

---

a) wild card inserts: wild card insert means we insert data in all column of database table.

Syntax: insert tablename values(value1,value2……..value…n);

Example: suppose we have table name as employee with three columns id,name and salary and we want to values to columns in row

```
mysql> insert into employee values(1,'ram',1000);
Query OK, 1 row affected (0.01 sec)

mysql> insert into employee values(2,'shyam',2000);
Query OK, 1 row affected (0.03 sec)

mysql> select *from employee;
+------+-------+------+
| eid  | name  | sal  |
+------+-------+------+
|    1 | ram   | 1000 |
|    2 | shyam | 2000 |
+------+-------+------+
2 rows in set (0.00 sec)

mysql>
```

b) partial insert: partial insert means we want to insert data in specific column of database table.

Syntax: insert tablename (columname,columnname)values(value1,value…n);

Example: we have employee table with column id, name and salary and we want to insert only data in id and name column not in salary so insert query like as

```
mysql> insert into employee (eid,name) values(3,'ganesh');
Query OK, 1 row affected (0.01 sec)

mysql> select *from employee;
+------+--------+------+
| eid  | name   | sal  |
+------+--------+------+
|    1 | ram    | 1000 |
|    2 | shyam  | 2000 |
|    3 | ganesh | NULL |
+------+--------+------+
3 rows in set (0.00 sec)
```

Note: here salary is null because with third record not insert value to salary column just we insert id and name and when we not provide any value to column then database engine provide default value to column known as null so salary is null with record 3

How to delete data from table?

If we want to delete record from a database table we have delete statement

Syntax: delete from tablename: if we think about this statement we can delete all records from database table

or

delete from tablename where condition: if think about this statement we can delete record from table according to condition

Example: we want to delete the employee record whose id is 3

```
mysql> select *from employee;
+------+--------+------+
| eid  | name   | sal  |
+------+--------+------+
|    1 | ram    | 1000 |
|    2 | shyam  | 2000 |
|    3 | ganesh | NULL |
+------+--------+------+
3 rows in set (0.03 sec)

mysql> delete from employee where eid=3;
Query OK, 1 row affected (0.02 sec)

mysql> select *from employee;
+------+-------+------+
| eid  | name  | sal  |
+------+-------+------+
|    1 | ram   | 1000 |
|    2 | shyam | 2000 |
+------+-------+------+
2 rows in set (0.00 sec)

mysql>
```

Before delete

After delete

How to update the data in table?

If we want to modify the particular data using update statement we can use update statement in SQL

Syntax: update tablename set column=value want to update ,column….. where condition;

Example: we want to change the salary of employee to 30000

```
mysql> select *from employee;
+------+-------+------+
| eid  | name  | sal  |
+------+-------+------+
|    1 | ram   | 1000 |
|    2 | shyam | 2000 |
+------+-------+------+
2 rows in set (0.00 sec)

mysql> update employee set sal=30000;
Query OK, 2 rows affected (0.04 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select *from employee;
+------+-------+-------+
| eid  | name  | sal   |
+------+-------+-------+
|    1 | ram   | 30000 |
|    2 | shyam | 30000 |
+------+-------+-------+
2 rows in set (0.00 sec)
```

Before Update

If we think about this table we update the complete sal column from employee means change the salary of every employee.

Note: if we want to change specific employee salary or specify column data then we can apply condition with update statement

After update

```
mysql> update  employee set sal=50000 where eid=2;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select *from employee;
+------+-------+-------+
| eid  | name  | sal   |
+------+-------+-------+
|    1 | ram   | 30000 |
|    2 | shyam | 50000 |
+------+-------+-------+
2 rows in set (0.00 sec)
```

How to fetch or retrieve or select data from table?

If we want to fetch data from database or display the data from table or display anything on output screen we have select statement

There are two types of select

---

1. Wild card select: wild card selection means we fetch all column data from database table called as wild card selection and wild card selection denoted by *

Syntax: select *from tablename**;**

Example:

```
mysql> select *from employee;
+------+-------+-------+
| eid  | name  | sal   |
+------+-------+-------+
|    1 | ram   | 30000 |
|    2 | shyam | 50000 |
+------+-------+-------+
2 rows in set (0.00 sec)
```

2. Partial select: partial selection means we fetch specified column data from row.

Syntax: select column1,column2…..column…n from tablename;

Example: if we think about above table we have three columns id, name and sal and we want to fetch only id and name

```
mysql> select eid,name from employee;
+------+-------+
| eid  | name  |
+------+-------+
|    1 | ram   |
|    2 | shyam |
+------+-------+
2 rows in set (0.00 sec)
```

Note: using select statement we can display anything like as variable values, string etc

```
mysql> select 100+200;
+---------+
| 100+200 |
+---------+
|     300 |
+---------+
1 row in set (0.00 sec)

mysql> select 100+200 as addition;
+----------+
| addition |
+----------+
|      300 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> select "good morning";
+--------------+
| good morning |
+--------------+
| good morning |
+--------------+
1 row in set (0.00 sec)
```

```
mysql> select "good morning" as " greeting message";
+--------------------+
| greeting message   |
+--------------------+
| good morning       |
+--------------------+
1 row in set, 1 warning (0.02 sec)
```

Note: you can arithmetic in select for display

Example: we want to increase the salary of employee by 10% and display it on output screen

```
mysql> select eid,name,sal from employee;
+------+-------+-------+
| eid  | name  | sal   |
+------+-------+-------+
|    1 | ram   | 30000 |
|    2 | shyam | 50000 |
+------+-------+-------+
2 rows in set (0.00 sec)

mysql> select eid,name,sal,sal*10/100 from employee;
+------+-------+-------+------------+
| eid  | name  | sal   | sal*10/100 |
+------+-------+-------+------------+
|    1 | ram   | 30000 |  3000.0000 |
|    2 | shyam | 50000 |  5000.0000 |
+------+-------+-------+------------+
2 rows in set (0.01 sec)

mysql> select eid,name,sal,sal*10/100,sal+(sal*10/100) from employee;
+------+-------+-------+------------+------------------+
| eid  | name  | sal   | sal*10/100 | sal+(sal*10/100) |
+------+-------+-------+------------+------------------+
|    1 | ram   | 30000 |  3000.0000 |       33000.0000 |
|    2 | shyam | 50000 |  5000.0000 |       55000.0000 |
+------+-------+-------+------------+------------------+
```

Alias concept in select Query

---

alias indicate reference or referral name to table or column in select query
if we think about above screen shot output we have eid column name , ename column name ,sal column so we want to change it on output screen like as show employee id as replacement to eid , employee name as replacement to ename just want to show on output screen only.

Syntax: select original colum name as “output column name” from tablename;

```
mysql> select eid as "Employee Id" ,name as "Employee Name",sal as "Employee Salary",sal*10/100 as "Incre
mented salary" ,sal+(sal*10/100) as "After Incrementation" from employee;
+-------------+---------------+-----------------+--------------------+----------------------+
| Employee Id | Employee Name | Employee Salary | Incremented salary | After Incrementation |
+-------------+---------------+-----------------+--------------------+----------------------+
|           1 | ram           |           30000 |          3000.0000 |           33000.0000 |
|           2 | shyam         |           50000 |          5000.0000 |           55000.0000 |
+-------------+---------------+-----------------+--------------------+----------------------+
2 rows in set (0.00 sec)
```

Where clause

Where clause is used for check the condition with DML statements
You can use where with all DML statement like as delete, update, select etc

Example: we want to fetch employee whose id is 1

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
7 rows in set (0.00 sec)

mysql> select *from employee where eid=1;
+------+------+-------+
| eid  | name | sal   |
+------+------+-------+
|    1 | ram  | 30000 |
+------+------+-------+
1 row in set (0.02 sec)
```

Example: WA SQL Query to display employee whose salary is greater than 30000

```
mysql> select *from employee where sal>30000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    2 | shyam   | 50000 |
|    6 | mangesh | 40000 |
+------+---------+-------+
2 rows in set (0.00 sec)

mysql>
```

Logical Operator in SQL?

Logical operator is used for combine more than one conditions and checks it or checks the multiple conditions at time and works as conjunction between multiple conditions and generates result as single condition.

Operator	Meaning
&& - you can use in wording format like as and	**Logical AND:** if all conditions are true then condition is true otherwise condition is false.
|| - you can use in wording format like as or	**Logical OR:** if any one condition is true then condition is true otherwise condition is false**.**
not , <>	**Logical NOT:** if condition is true then false if false then true

Example: we want to fetch employee whose salary is greater than equal 10000 and less than equal 40000

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
7 rows in set (0.00 sec)

mysql> select *from employee where sal>=10000 and sal<=40000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
+------+---------+-------+
3 rows in set (0.00 sec)
```

Note: you can use it like as

```
mysql> select *from employee where sal>=10000 && sal<=40000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
+------+---------+-------+
3 rows in set, 1 warning (0.00 sec)
```

Example: WA SQL Query to display the employee whose salary is 6000,30000 ,40000

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
7 rows in set (0.00 sec)

mysql> select *from employee where sal=6000 or sal=30000 or sal=40000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> select *from employee where sal=6000 || sal=30000 || sal=40000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
4 rows in set, 2 warnings (0.00 sec)
```

Example: WA SQL Query to display the employee list except ganesh

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
7 rows in set (0.00 sec)
mysql> select *from employee where not name='ganesh';
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
+------+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> select *from employee where  name!='ganesh';
```

```
mysql> select *from employee where  name<>'ganesh';
```

IN and between operator

Q. What is IN operator?

IN operator is used for avoid writing multiple OR condition with same column in select query as well as IN operator work as conjunction in SubQuery

Now we want to discuss about how to avoid writing multiple OR with same column in select query.

Syntax:

Select *from tablename where column name IN(value1,value2……value…n);

or

Select column1,column2….column..n from tablename where columname IN(value1…..value…n);

Example: WA SQL Query to fetch employee record whose name is ram, shyam, ganesh, sandeep, rajesh

```
mysql> select *from employee where name='ram' || name='shyam' || name='ganesh' || name='rajesh' || name='sandeep';
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
+------+---------+-------+
5 rows in set, 4 warnings (0.00 sec)
```

Note: if we think about above SQL Query we use 5 condition on same column i.e on name column using OR operator so here we use unnecessarily || operator multiple times so SQL suggest it is not good way to write condition on same column with OR operator so better way you can use IN Operator

Above SQL Query using IN Operator

```
mysql> select *from employee where name IN('ram','shyam','ganesh','rajesh','sandeep');
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
+------+---------+-------+
5 rows in set (0.03 sec)
```

Between operator

Between operators especially design fetches range of data like as >= && <=

Normally between recommend when we want to avoid >= && <= conditions on column

Syntax: select *from tablename where column name between value and value

Example: Fetch employee whose salary is greater than 6000 and less than equal 30000

```
mysql> select *from employee where sal>=6000 && sal<=30000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    7 | sangram |  6000 |
+------+---------+-------+
3 rows in set, 1 warning (0.00 sec)
```

```
mysql> select *from employee where sal between 6000 and 30000;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    7 | sangram |  6000 |
+------+---------+-------+
3 rows in set (0.00 sec)
```

Group functions or aggregate functions

Group functions are used for work with specified column and generate the single value result called as group functions.

There are five types of group function we have

---

1) count: count function is used for count the number of records from database or column and returns it.

Syntax: select count(columnname) from tablename: this function can count only non null value

or

select count(*) from tablename: this function can count non null values as well as null values.

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  NULL |
|    9 | raghav  |  NULL |
+------+---------+-------+
9 rows in set (0.00 sec)

mysql> select count(sal) from employee;
+------------+
| count(sal) |
+------------+
|          7 |
+------------+
```

Note: if we think about left hand side code we have employee table which contain total 9 record if we think about sal column it contain 2 null record and we use column name in count function
select count(sal) from employee; when we pass column in function then function only count not null values so this is major reason we have 7 count

Note: If you want to count the not null value as well as null value then we can use * in count function.

Example for counting null and not null value

```
mysql> select count(*) from employee;
+----------+
| count(*) |
+----------+
|        9 |
+----------+
1 row in set (0.01 sec)
```

2) max : max function is used for find the max value from database table

Syntax: select max(columnname) from tablename;

Example: find the maximum salary of employee;

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  NULL |
|    9 | raghav  |  NULL |
+------+---------+-------+
9 rows in set (0.00 sec)

mysql> select max(sal) from employee;
+----------+
| max(sal) |
+----------+
|    50000 |
+----------+
1 row in set (0.00 sec)
```

3) min: this function is used for calculate the minimum value from table

Syntax: select min(salary) from tablename;

Example: find the minimum salary of employee from employee table

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  NULL |
|    9 | raghav  |  NULL |
+------+---------+-------+
9 rows in set (0.00 sec)
mysql> select min(sal) from employee;
+----------+
| min(sal) |
+----------+
|     4000 |
+----------+
1 row in set (0.00 sec)
```

4) sum : this function can calculate the sum of all values of specified column from table.

Syntax: select sum(columnname) from tablename;

Example: calculate the sum of all employee salary

select sum(sal) from employee;

```
mysql> select sum(sal) from employee;
+----------+
| sum(sal) |
+----------+
|   165000 |
+----------+
1 row in set (0.03 sec)
```

5) avg : this function is used for calculate the average value of column from database table.

Syntax: select avg(columnname) from tablename;

Example: find the average salary of employee;

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  4000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  NULL |
|    9 | raghav  |  NULL |
+------+---------+-------+
9 rows in set (0.00 sec)
```

```
mysql> select sum(sal)/count(sal) from employee;
+----------------------+
| sum(sal)/count(sal) |
+----------------------+
|          23571.4286 |
+----------------------+
1 row in set (0.04 sec)
```

Internal logic of avg() function

```
mysql> select avg(sal) from employee;
+------------+
| avg(sal)   |
+------------+
| 23571.4286 |
+------------+
1 row in set (0.04 sec)
```

group by , having , like and order by clause

Q. What is group by clause?

---

group by clause is used for create group of similar values using specified column name and represent single value in output or single value as output

Normally group by help us to categories data from column using similar values.

Example: find the employee count dept wise, find the customer count citywise , find the student count branch wise, find the student count qualification wise etc

If we want to work with group by we have to use following syntax

---

Syntax: select columnname from tablename group by columnname;

Example: select sal from employee group by sal;

If we think about this select statement we can say we categories employee data by sal.

If we want to work with group by we have some important rules

---

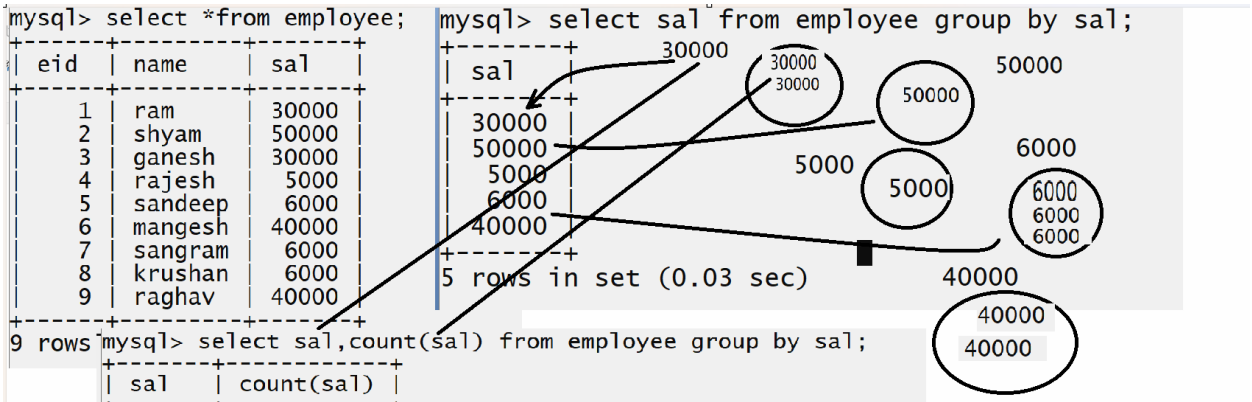
1. We can use group functions with group by clause
2. We can use column name in select query which we use with group by clause.

Example using group by clause

---

display the salary wise employee count or count the similar salaries employee.

```
mysql> select *from employee;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | shyam   | 50000 |
|    3 | ganesh  | 30000 |
|    4 | rajesh  |  5000 |
|    5 | sandeep |  6000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  6000 |
|    9 | raghav  | 40000 |
+------+---------+-------+
9 rows in set (0.00 sec)

mysql> select sal from employee group by sal;
+-------+
| sal   |
+-------+
| 30000 |
| 50000 |
|  5000 |
|  6000 |
| 40000 |
+-------+
5 rows in set (0.03 sec)

mysql> select sal,count(sal) from employee group by sal;
+-------+------------+
| sal   | count(sal) |
+-------+------------+
| 30000 |          2 |
| 50000 |          1 |
|  5000 |          1 |
|  6000 |          3 |
| 40000 |          2 |
+-------+------------+
5 rows in set (0.04 sec)
```



Example: count employee by name

```
mysql> select name,count(name) from employee  group by name;
+---------+-------------+
| name    | count(name) |
+---------+-------------+
| ram     |           2 |
| ganesh  |           2 |
| sandeep |           1 |
| mangesh |           1 |
| sangram |           1 |
| krushan |           1 |
| raghav  |           1 |
+---------+-------------+
7 rows in set (0.00 sec)
```

having clause

Having clause normally use for check the condition with group functions or aggregate function and having clause always use with group by clause.

Q. What is difference between where clause and having clause?

---

1. Where clause can check the condition without group function and having can check the condition with group functions.
2. Where clause can work without group by as well as can work with group by but having cannot use without group by

How to use having practically?

Syntax: select columnname from tablename group by columnname having condition;

Example: WA SQL Query to find the employee count with same salaries or find duplicated salary from employee table.

```
mysql> select sal,count(sal) from employee group by sal having count(sal)>1;
+-------+------------+
| sal   | count(sal) |
+-------+------------+
| 30000 |          2 |
|  6000 |          3 |
| 40000 |          2 |
+-------+------------+
3 rows in set (0.00 sec)
```

Example: WA SQL Query find the duplicated salary from employee but whose sum is more than 40000

```
mysql> select sal,count(sal) ,sum(sal) from employee group by sal having count(sal)>1 && sum(sal)>40000;
+-------+------------+----------+
| sal   | count(sal) | sum(sal) |
+-------+------------+----------+
| 30000 |          2 |    60000 |
| 40000 |          2 |    80000 |
+-------+------------+----------+
2 rows in set, 1 warning (0.04 sec)
```

order by clause

Order by clause is used for arrange record in ascending order or in descending order and by default order by clause use ascending order technique.

Syntax: select *from tablename order by columnname desc | asc

Example: WA SQL Query to display employee record by salaries using descending order

```
mysql> select *from employee order by sal desc;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    2 | ram     | 50000 |
|    6 | mangesh | 40000 |
|    9 | raghav  | 40000 |
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    5 | sandeep |  6000 |
|    7 | sangram |  6000 |
|    8 | krushan |  6000 |
|    4 | ganesh  |  5000 |
+------+---------+-------+
9 rows in set (0.00 sec)
```

```
mysql> select *from employee order by sal;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    4 | ganesh  |  5000 |
|    5 | sandeep |  6000 |
|    7 | sangram |  6000 |
|    8 | krushan |  6000 |
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
|    9 | raghav  | 40000 |
|    2 | ram     | 50000 |
+------+---------+-------+
9 rows in set (0.00 sec)
```

both are equal

Note: order by clause by default use the ascending order or asc is default format with order by clause

```
mysql> select *from employee order by sal asc;
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    4 | ganesh  |  5000 |
|    5 | sandeep |  6000 |
|    7 | sangram |  6000 |
|    8 | krushan |  6000 |
|    1 | ram     | 30000 |
|    3 | ganesh  | 30000 |
|    6 | mangesh | 40000 |
|    9 | raghav  | 40000 |
|    2 | ram     | 50000 |
+------+---------+-------+
9 rows in set (0.00 sec)
```

Now we want to discuss like operator

Like operator is used for pattern matching purpose means if we want to search specific data from database table then we can use like operator.

Example: Find the employee whose name start with r, find the employee whose company name ends with s , find the employee whose name contain at least 3 letters etc

Syntax: select col1,col2……..n from tablename where columnname like pattern

If we want to work with like operator we have two wild card characters.

% - this operator represent one or more characters

_ : this operator represent only single characters.

Example1: WA SQL Query to display details of employee whose name start with s

```
mysql> select *from employee where name like 's%';
+------+---------+------+
| eid  | name    | sal  |
+------+---------+------+
|    5 | sandeep | 6000 |
|    7 | sangram | 6000 |
+------+---------+------+
2 rows in set (0.01 sec)
```

this like operator indicate we want start with s and after s there may be any character as well as there may be single character after s or more than one characters

Example2: WA SQL Query to display the employee details whose name ends with sh.

```
mysql> select *from employee where name like '%sh';
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    3 | ganesh  | 30000 |
|    4 | ganesh  |  5000 |
|    6 | mangesh | 40000 |
+------+---------+-------+
3 rows in set (0.00 sec)
```

Note: we use % symbol before sh means pattern say before sh there may be any character or there may be single or more than one characters

Example: WA SQL Query to display the employees whose name start with m and ends with sh

```
mysql> select *from employee where name like 'm%sh';
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    6 | mangesh | 40000 |
+------+---------+-------+
1 row in set (0.00 sec)
```

Note: we use % symbol in between m and sh so the pattern say search data in name column start m and ends with sh and in between m and sh there may be single character or more than one character or any character

Example: WA SQL Query to fetch employee whose name contain only three character

```
mysql> select *from employee where name like '___';
+------+------+-------+
| eid  | name | sal   |
+------+------+-------+
|    1 | ram  | 30000 |
|    2 | ram  | 50000 |
+------+------+-------+
2 rows in set (0.00 sec)

mysql>
```

Note: if we think about this query we use ___ three underscore in select statement and here _ single underscore represent single character so three underscore represent three character so this pattern say fetch employee whose name contain three letters.

Example: WA SQL Query to find the employee list whose name contains at least three letters.

```
mysql> select *from employee where name like '___%';
+------+---------+-------+
| eid  | name    | sal   |
+------+---------+-------+
|    1 | ram     | 30000 |
|    2 | ram     | 50000 |
|    3 | ganesh  | 30000 |
|    4 | ganesh  |  5000 |
|    5 | sandeep |  6000 |
|    6 | mangesh | 40000 |
|    7 | sangram |  6000 |
|    8 | krushan |  6000 |
|    9 | raghav  | 40000 |
+------+---------+-------+
9 rows in set (0.00 sec)
```

Q. When like,where,group by ,having and order by come same SQL Query then how we can maintain its priority or priority?

---

1. Where 2. like 3. group by 4. having 5. order by

Example: WA SQL Query show the employee list who salary is same but name start with g and ends with sh

mysql> select name,sal,count(sal) from employee where name like 'g%sh' group by sal having count(sal)>1 order by sal asc;

```
+--------+-------+-------------+
| name   | sal   | count(sal)  |
+--------+-------+-------------+
| ganesh |  1000 |           2 |
| ganesh | 30000 |           2 |
+--------+-------+-------------+
2 rows in set (0.00 sec)
```

Constraints

Q. What are constraints in SQL?

Constraints are the some rules and regulation which we can apply on database table or column in database table. The Major goal of constraints is to avoid store malicious or unwanted or wrong data in database table as well as it help us to maintain the unique identity of data in column of table as well as it help us check the condition or data before inserting column of table and set the default values to column if user not provided and constraints help us to maintain integrity relations between multiple tables etc

Types of Constraints in SQL

---

1. not null: not null constraints is used for avoid to store null value in column means when user apply not null constraints on column and if we try to store null value in column then we get error means we must be store value in it except null

Note: when user not apply not null constraints on database table then by default database can store null value in column if user not provide the value to that column

Syntax: create table tablename(columnname datatype(size) not null,……)

Example: Suppose consider we have employee table with filed id ,name ,salary and age and when want to add employee record must be pass name value

```
mysql>
mysql> create table employee(id int(5),name varchar(200) not null,sal int(5),age int(5)

Query OK, 0 rows affected, 3 warnings (0.08 sec)

mysql> insert into employee (name,age) values('ganesh',18);
Query OK, 1 row affected (0.03 sec)

mysql> insert into employee (id,age) values(1,18);
ERROR 1364 (HY000): Field 'name' doesn't have a default value
mysql>
```

Note: if we think about above screen short we get error Field name does not have a default value because we set not null constrains on name and we not provide value to name column using insert query then database engine try to insert the null value in name column but user apply not null constrains on name column so database engine cannot store default value as null so we get error

Note: if we want to solve above error pass value to name column shown in next screen short

2. Unique: unique constraints is used for avoid duplicated data or value in column of database table means when we apply unique constraints on column in table then we must be store unique value in it if we try to store duplicate value then we get error at run time.
Example: When we store employee email records or contact records they must be unique , if we think RTO vehicle management record then vehicle no should be unique, license no of user should be unique etc
Syntax: create table tablename(columnname datatype(size) unique);
or
Note: you can apply more than one constraints with single column also shown in following syntax
Syntax: create table tablename(columnname datatype(size) unique not null)

Example: we want to create employee table with field id,name, email ,contact here name should not null and email and contact should be unique

mysql> create table employee(id int(5),name varchar(200) not null, email varchar(200) not null unique,contact varchar(200) not null unique);
Query OK, 0 rows affected, 1 warning (0.08 sec)

```
mysql> insert into employee values(1,'ram','ram@gmail.com','3456');
Query OK, 1 row affected (0.04 sec)

mysql> insert into employee values(2,'rama','ram@gmail.com','34456');
ERROR 1062 (23000): Duplicate entry 'ram@gmail.com' for key 'email'
```

Note: if we think about above query we get error ram@gmail.com Duplicate entry because we use unique constraints on email column and we have ram@gmail.com already present in database table and we try to store duplicate entry in email i.e ram@gmail.com and email has unique constraints so it is not possible

3. primary: primary key constraints is combination of unique key and not null means we cannot duplicate value in primary key and also cannot store null value in it and we can create only one primary key column in single table.

Syntax: create table tablename(columnname datatype(size) primary key,columnname…..)

Example: Suppose consider we want to create table name as employee with field id,name and email and contact so here id should be primary and name should be not null and email and contact should be unique.

mysql> create table employee(id int(5) primary key,name varchar(200) not null, email varchar(200) unique,contact varchar(200) unique);
Query OK, 0 rows affected, 1 warning (0.05 sec)

```
mysql> insert into employee values(1,'ram','ram@gmail.com','12345');
Query OK, 1 row affected (0.02 sec)

mysql> insert into employee values(1,'ram','ram@gmail.com','12345');
ERROR 1062 (23000): Duplicate entry '1' for key 'PRIMARY'
mysql>
```

Note: if we think about above screen shot we get error duplicate entry 1 for key primary because we have already record in employee table with id 1 i.e primary key id 1 and we try to store one more record of employee with id 1 and it is not possible because primary key cannot duplicated and we try to duplicate it so we get error.

Q. What is difference between primary and unique key constraints?

Primary key	**Unique**
Primary key is by default not null	Unique is not by default not null
Primary key cannot store null value	Unique key can store null value
Single table can have only one primary means we cannot create more than one primary key column in single table means we can say primary key column represents row i.e record	But we can create more than one column as unique in single table and unique column represents attribute i.e column or field.
Primary key column can use as reference in another table as foreign key	Unique key column cannot use as reference in another table as foreign key
Primary key column use clustered indexing technique	Unique key column non – clustered index or secondary index

4. foreign: foreign key constraints is used for inter connect two tables with each other means every foreign key is primary key in some another table and foreign key and primary key maintain parent and child relationship between table means primary key column table known as parent able and foreign key column table known as child table.

Means we can say with the help of foreign key we can provide relationship between two tables.

Course

courseid (PK)	coursename
1	JAVA
2	MERN
3	.NET

(parent table)

Student

sid	name	email	contact	courseid (FK)
1	ram	ram@	44444444	1
2	shyam	shyam@	444442222	2
3	ganesh	ganesh		3

(child parent)

How to create foreign key practically?

Syntax:

create table tablename(columname datatype(size),foreign key(columnname) references parenttable(primary key colname));

Example:

mysql> create table course(cid int(5) primary key ,name varchar(200));

Query OK, 0 rows affected, 1 warning (0.06 sec)

mysql> create table student(sid int(5) primary key ,name varchar(200),email varchar(200),contact varchar(200),cid int(5),foreign key(cid) references course(cid));

```
mysql> select *from course;
+-----+------+
| cid | name |
+-----+------+
|   1 | JAVA |
|   2 | .NET |
|   3 | MERN |
+-----+------+
3 rows in set (0.00 sec)

mysql> select *from student;
+-----+-------+-----------------+---------+------+
| sid | name  | email           | contact | cid  |
+-----+-------+-----------------+---------+------+
|   1 | ram   | ram@gmail.com   | 12345   |    1 |
|   2 | shyam | shyam@gmail.com | 123453  |    2 |
+-----+-------+-----------------+---------+------+
2 rows in set (0.00 sec)
```

Query OK, 0 rows affected, 2 warnings (0.09 sec)

If we think about above table we can say Course is parent able and Student is child table because we use the course id as foreign key in Student table

Now we want to delete the source whose id is 1

mysql> delete from course where cid=1;

ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`aug2024`.`student`, CONSTRAINT `student_ibfk_1` FOREIGN KEY (`cid`) REFERENCES `course` (`cid`))

Note: if we think above query when we try to delete the course whose id is 1 then we get error because course id 1 use by student table shown in above screen shot

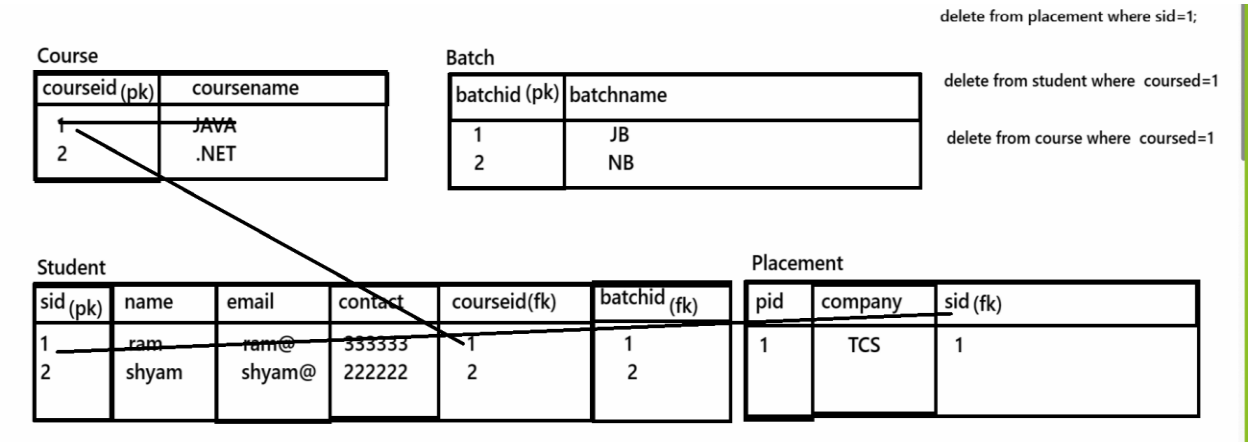
According to rule of primary key when its reference or value use by child table then we cannot delete or update the primary key directly before that we required to delete or update foreign key record or child record before parent or primary key record and after that we delete or update parent record.

Show in following screen shot

```
mysql> delete from student where cid=1;
Query OK, 1 row affected (0.04 sec)

mysql> delete from course where cid=1;
Query OK, 1 row affected (0.01 sec)

mysql> select *from student;
+-----+-------+-----------------+---------+------+
| sid | name  | email           | contact | cid  |
+-----+-------+-----------------+---------+------+
|   2 | shyam | shyam@gmail.com | 123453  |    2 |
+-----+-------+-----------------+---------+------+
1 row in set (0.00 sec)

mysql> select *from course;
+-----+------+
| cid | name |
+-----+------+
|   2 | .NET |
|   3 | MERN |
+-----+------+
2 rows in set (0.00 sec)
```

Note: above approach is not feasible in real time scenario or it may be very tedious or complicated task when we have large database

We want to discuss about Scenario



Note: if we think about above database and if we want to delete course whose id is 1 before that we need to delete student whose course id is 1 but before we need to delete placement of student whose sid is 1 means for delete course we need to execute following types of delete

Example:

delete from placement where sid=1;

delete from student where courseid=1;

delete from course where courseid=1;

here for delete course we need to write three different types of delete but consider if we have 100 tables in your table and if use courseid with 100 tables as reference Id and if we want to delete or update the course id then we need to perform deletion or updation on 100 different places by writing manual delete or update query so it is not possible in real time scenario

So if we want to resolve this problem we can use following types of constraints with foreign key

a) on delete cascade: if we think about on delete cascade this constraint delete the child automatically when we delete parent record

b) on update cascade: this constraints update the child automatically when we update the parent record.

c) on delete set null: this constraints delete only parent records and set child foreign key as null

How to use on delete cascade practically

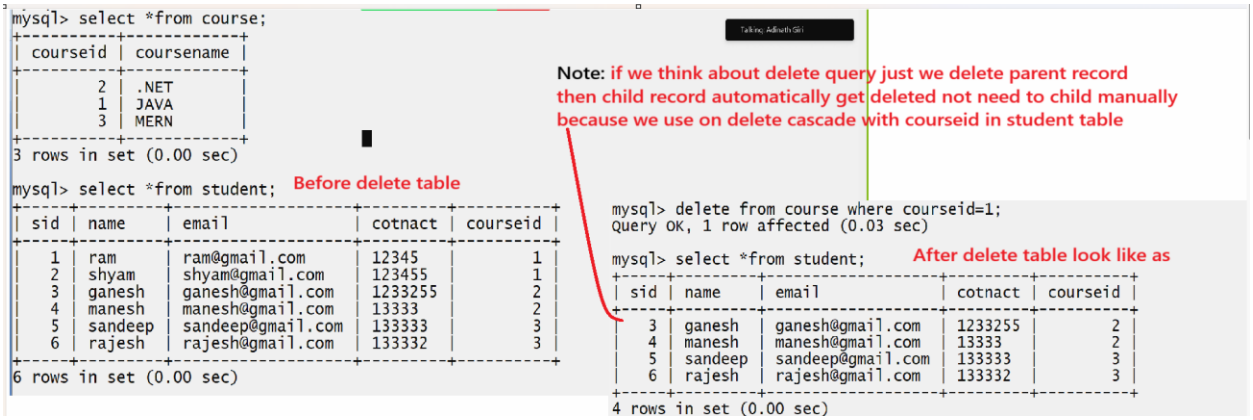
Syntax:

create table tablename(columnname datatype(size),foreign key(columnname),references parenttable(primarykey) on delete cascade);

Example:

mysql> create table course(courseid int(5) primary key ,coursename varchar(200) not null unique);
Query OK, 0 rows affected, 1 warning (0.09 sec)

mysql> create table student(sid int(5) primary key,name varchar(200),email varchar(200),cotnact varchar(200),courseid int(5),foreign key(courseid) references course(courseid) on delete cascade);
Query OK, 0 rows affected, 2 warnings (0.08 sec)



How to use on update cascade practically

Syntax:

create table tablename(columnname datatype(size),foreign key(columnname),references parenttable(primarykey) on update cascade);

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        2 | .NET       |
|        3 | MERN       |
+----------+------------+
2 rows in set (0.00 sec)

mysql> select *from student;
+-----+---------+-------------------+---------+----------+
| sid | name    | email             | cotnact | courseid |
+-----+---------+-------------------+---------+----------+
|   3 | ganesh  | ganesh@gmail.com  | 1233255 |        2 |
|   4 | manesh  | manesh@gmail.com  | 13333   |        2 |
|   5 | sandeep | sandeep@gmail.com | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com  | 133332  |        3 |
+-----+---------+-------------------+---------+----------+
4 rows in set (0.00 sec)
```

Note: if we think about course table our courseid is 2 for .NET course so we want to change courseid from 2 to 1 for .NET course but we want to courseid should automatically reflect in child table i.e in student table so we can use on update cascade.

Example:

mysql> create table course(courseid int(5) primary key ,coursename varchar(200) not null unique);
mysql> create table student(sid int(5) primary key,name varchar(200),email varchar(200),cotnact varchar(200),courseid int(5),foreign key(courseid) references course(courseid) on update cascade);
Query OK, 0 rows affected, 2 warnings (0.08 sec)

```
mysql> update course set courseid=1 where coursename='.NET';
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        3 | MERN       |
+----------+------------+
2 rows in set (0.00 sec)

mysql> select *from student;
+-----+---------+-------------------+---------+----------+
| sid | name    | email             | cotnact | courseid |
+-----+---------+-------------------+---------+----------+
|   3 | ganesh  | ganesh@gmail.com  | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com  | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com  | 133332  |        3 |
+-----+---------+-------------------+---------+----------+
4 rows in set (0.00 sec)
```

on delete set null

---

mysql> create table product(pid int(5) primary key,name varchar(200) not null unique);
Query OK, 0 rows affected, 1 warning (0.05 sec)

mysql> create table customer(cid int(5) primary key , name varchar(200),email varchar(200),contact varchar(200),pid int(5),foreign key(pid) references product(pid) on delete set null);
Query OK, 0 rows affected, 2 warnings (0.07 sec)

```
mysql> select *from product;
+-----+-------+
| pid | name  |
+-----+-------+
|   2 | Sofa  |
|   1 | Table |
+-----+-------+
2 rows in set (0.00 sec)

mysql> select *from customer;
+-----+------+-------------+---------+------+
| cid | name | email       | contact | pid  |
+-----+------+-------------+---------+------+
|   1 | a    | a@gmail.com | 12345   |    1 |
|   2 | b    | b@gmail.com | 123453  |    2 |
+-----+------+-------------+---------+------+
2 rows in set (0.00 sec)

mysql>
```

Now we want to delete product records.

```
mysql> delete from product;
Query OK, 2 rows affected (0.01 sec)

mysql> select *from customer;
+-----+------+-------------+---------+------+
| cid | name | email       | contact | pid  |
+-----+------+-------------+---------+------+
|   1 | a    | a@gmail.com | 12345   | NULL |
|   2 | b    | b@gmail.com | 123453  | NULL |
+-----+------+-------------+---------+------+
2 rows in set (0.00 sec)
```

Q. Can we use on delete cascade and on update cascade with same column?

Yes we can use on delete cascade and on update cascade with same column

mysql> create table customer(cid int(5) primary key , name varchar(200),email varchar(200),contact varchar(200),pid int(5),foreign key(pid) references product(pid) on delete cascade on update cascade);

Q. Can we use on delete cascade and on delete set null with same column?

No we cannot use the on delete cascade and on delete set null with same column because it will create conflict to manage the operations with child records and if we try to apply both with same column we get error.

Example:

mysql> create table customer(cid int(5) primary key , name varchar(200),email varchar(200),contact varchar(200),pid int(5),foreign key(pid) references product(pid) on delete cascade on delete set null);

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 'delete set null)' at line 1

Q. Can we use on delete set null and on update cascade with same column?

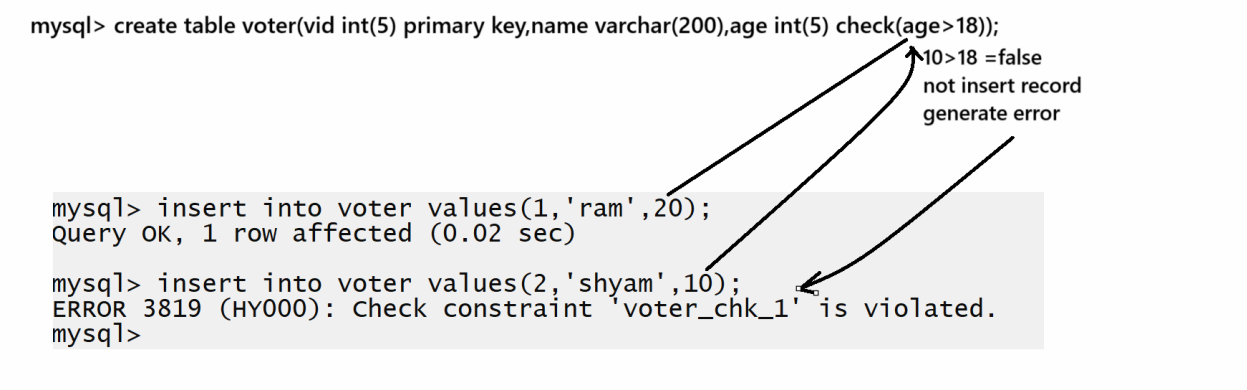
Yes we can use the both with same column name

mysql> create table customer(cid int(5) primary key , name varchar(200),email varchar(200),contact varchar(200),pid int(5),foreign key(pid) references product(pid) on delete set null on update cascade);
Query OK, 0 rows affected, 2 warnings (0.10 sec)

5. check: check constraints is used for apply the some specific condition with column in database table.
Syntax: create table tablename(columnname datatype(size) condition);

Example: Suppose consider we are working on voter database and the condition accept voter detail database if his age is greater than 18

mysql> create table voter(vid int(5) primary key,name varchar(200),age int(5) check(age>18));
Query OK, 0 rows affected, 2 warnings (0.05 sec)



6. default : default constraints help us to set the default value to table column means when user not provide value to that column then database engine insert value in column which set by user at the time of table creation and when user provide value to that column then database engine override user value on default value by user at the time of table creation.

Syntax: create table tablename(columnname datatype(size) default value);

Example: suppose consider we have employee table with field id,name and sal and we want to employee minimum should be 10000 means when we not provide salary value manually to salary column then it should be 10000.

mysql> create table employee(eid int(5) primary key,name varchar(200),salary int(5) default 10000);
Query OK, 0 rows affected, 2 warnings (0.05 sec)

```
mysql> create table employee(eid int(5) primary key,name varchar(200),salary int(5) default 10000);

mysql> insert into employee values(1,'ram',20000);
Query OK, 1 row affected (0.02 sec)
mysql> select *from employee;
+-----+------+--------+
| eid | name | salary |
+-----+------+--------+
|   1 | ram  |  20000 |
+-----+------+--------+
1 row in set (0.00 sec)
```

override user value on default because user provide value using insert statement

```
mysql> insert into employee (eid,name)values(2,'shyam');
Query OK, 1 row affected (0.01 sec)
mysql> select *from employee;
+-----+-------+--------+
| eid | name  | salary |
+-----+-------+--------+
|   1 | ram   |  20000 |
|   2 | shyam |  10000 |
+-----+-------+--------+
```

user default set by user at the time of table creation because user not provide manually value by insert statement

7. auto_increment: auto_increment normally help us to increase the value automatically by database engine normally we recommend auto increment with primary key or unique also.

Syntax:

create table tablename(columnname datatype(size) primary key auto_increment,columnname datatype(size));

or

create table tablename(columnname datatype(size) unique auto_increment,columnname datatype(size));

Example: we want to create table name as product with field id, name and price and we want to generate the product id automatically.

mysql> create table product(pid int(5) primary key auto_increment,name varchar(200),price int(5));
Query OK, 0 rows affected, 2 warnings (0.06 sec)

Note: when we want to insert primary key value with auto_increment we can use '0' or null in primary key column

```
mysql> insert into product values('0','A',100);
Query OK, 1 row affected (0.04 sec)

mysql> insert into product values('0','B',200);
Query OK, 1 row affected (0.03 sec)

mysql> insert into product values('0','C',300);
Query OK, 1 row affected (0.03 sec)

mysql> select *from product;
+-----+------+-------+
| pid | name | price |
+-----+------+-------+
|   1 | A    |   100 |
|   2 | B    |   200 |
|   3 | C    |   300 |
+-----+------+-------+
3 rows in set (0.03 sec)

mysql>
```

Joins

Q. What is joins?

When we combine more than one table in select query for fetch data and those tables are interconnected with each by using primary key and foreign key column for fetch data on output screen called as joins.

Some time we store data in different tables in database but on output screen we want to combine data from different tables then join come picture.

Example : Suppose consider we have two tables name as Course and Student and Course Table pass courseid as foreign key in Student table shown in following screen shot

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        3 | MERN       |
+----------+------------+
2 rows in set (0.02 sec)

mysql> select *from student;
+-----+---------+-------------------+---------+----------+
| sid | name    | email             | cotnact | courseid |
+-----+---------+-------------------+---------+----------+
|   3 | ganesh  | ganesh@gmail.com  | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com  | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com  | 133332  |        3 |
+-----+---------+-------------------+---------+----------+
4 rows in set (0.04 sec)
```

We want output like as

Output:

Student name	course name
ganesh	.NET
manesh	.NET
Sandeep	MERN
rajesh	MERN

Here we want to fetch student name from student table and course name from course table means we fetch data using single select query on output from two different tables so we required to write join in Database.

If we want to work with joins we have some types of joins

1. Left join: if we think about left join fetch all records from left hand side table and common records from right hand side tables shown in following diagram.

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.00 sec)
```

Output of left join

Course Name	Student Name
.NET	ganesh
.NET	manesh
MERN	Sandeep
MERN	rajesh
AWS	NULL
HADOOP	NULL

```
mysql> select *from student;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
4 rows in set (0.00 sec)
```

2. Right join: right join means fetch all records from right hand side table and common records from left hand side table.

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select *from student;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.00 sec)
```

Output of right join

Course Name	Student Name
.NET	ganesh
.NET	manesh
MERN	Sandeep
MERN	Rajesh
NULL	Shyam
NULL	raj

3. Inner join: inner join means fetch common records from left hand side table and right hand side tables.

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select *from student;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.00 sec)
```

Output of right join

Course Name	Student Name
.NET	ganesh
.NET	manesh
MERN	Sandeep
MERN	Rajesh

4. Outer join: if we think about outer join fetch all records from left hand side table as well as all records from right hand side tables called as outer join means we can say outer join combination of left join and right join.

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select *from student;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.00 sec)
```

Output of right join

Course Name	Student Name
.NET	ganesh
.NET	manesh
MERN	Sandeep
MERN	Rajesh
HADOOP	NULL
AWS	NULL
NULL	SHYAM
NULL	raj

5. Self join: when we perform join on same table called as self join

How to implement join practically using MYSQL

Syntax:

select ltref.column,rtref.columnname from lefttablename ltref jointype righttablename rtref on ltref.column=rtref.columnname

Example of left join

```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select *from student;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.00 sec)
```

```
mysql> select c.coursename,s.name from course c left join student s on c.courseid=s.courseid
```

```
+------------+---------+
| coursename | name    |
+------------+---------+
| .NET       | ganesh  |
| .NET       | manesh  |
| MERN       | sandeep |
| MERN       | rajesh  |
| AWS        | NULL    |
| HADOOP     | NULL    |
+------------+---------+
```

Example of right join

```
mysql> select *from course order by courseid;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        3 | MERN       |
|        4 | HADOOP     |
|        5 | AWS        |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select c.coursename,s.name from course c right join student s on c.courseid=s.courseid;
+------------+---------+
| coursename | name    |
+------------+---------+
| NULL       | shyam   |
| NULL       | raj     |
| .NET       | ganesh  |
| .NET       | manesh  |
| MERN       | sandeep |
| MERN       | rajesh  |
+------------+---------+
```

```
mysql> select *from student order by courseid;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.03 sec)
```

Example of inner join

```
mysql> select *from course order by courseid;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        3 | MERN       |
|        4 | HADOOP     |
|        5 | AWS        |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select c.coursename,s.name from course c inner join student s on c.courseid=s.courseid;
+------------+---------+
| coursename | name    |
+------------+---------+
| .NET       | ganesh  |
| .NET       | manesh  |
| MERN       | sandeep |
| MERN       | rajesh  |
+------------+---------+
4 rows in set (0.02 sec)
```

```
mysql> select *from student order by courseid;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.03 sec)
```

Example of outer join

Note: if we think about MYSQL there is no outer keyword for manage outer join so if we want to write outer join or implement outer join practically in MYSQL we have to use union keyword

So with union keyword you have to write left join at left hand side of union keyword and right join at right hand side of union keyword so it generate output for outer join

```
mysql> select *from course order by courseid;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        3 | MERN       |
|        4 | HADOOP     |
|        5 | AWS        |
+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select c.coursename,s.name from course c left join student s on c.courseid=s.courseid union
select c.coursename,s.name from course c right join student s on c.courseid=s.courseid;
+------------+---------+
| coursename | name    |
+------------+---------+
| .NET       | ganesh  |
| .NET       | manesh  |
| MERN       | sandeep |
| MERN       | rajesh  |
| AWS        | NULL    |
| HADOOP     | NULL    |
| NULL       | shyam   |
| NULL       | raj     |
+------------+---------+
8 rows in set (0.03 sec)
```

```
mysql> select *from student order by courseid;
+-----+---------+--------------------+---------+----------+
| sid | name    | email              | cotnact | courseid |
+-----+---------+--------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com    | 34444   |     NULL |
|   2 | raj     | raj@gmail.com      | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com   | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com   | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com  | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com   | 133332  |        3 |
+-----+---------+--------------------+---------+----------+
6 rows in set (0.03 sec)
```

Consider we have the Database

Course Table

Courseid	Course name	Fees
1	JAVA	26000
2	PHP	26000
3	.NET	26000

Student Table

Sid	Name	Email	Contact	Address
1	Ram	ram@gmail	123456	PUNE
2	Shyam	shyam@	343434	Nashik
3	Dinesh	dinesh@	43265	Mumbai

Placement Table

Pid	Comp name	Package	Pdate
1	TCS	360000	01/01/2020
2	Infosys	360000	01/02/2020

Csjoin table

Courseid	Sid	Pid
1	1	1
2	1	2
1	2	1
2	2	2

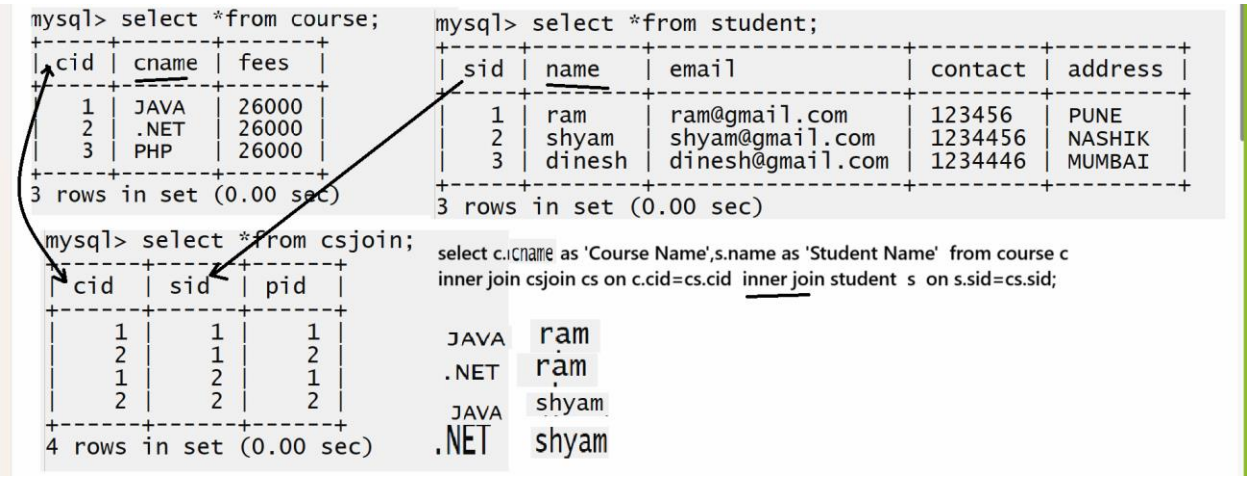
Example

```
mysql> create table course(cid int(5) primary key auto_increment,cname varchar(200) not null
unique,fees int(5) not null);
Query OK, 0 rows affected, 2 warnings (0.14 sec)

mysql> create table student(sid int(5) primary key auto_increment,name varchar(200),email
varchar(200),contact varchar(200),address varchar(200));
Query OK, 0 rows affected, 1 warning (0.10 sec)

mysql> create table placement(pid int(5) primary key auto_increment,compname varchar(200),package
int(5),pdate date);
Query OK, 0 rows affected, 2 warnings (0.06 sec)
```

mysql> create table csjoin(cid int(5),foreign key(cid) references course(cid) on delete cascade on update cascade, sid int(5),foreign key(sid) references student(sid) on delete cascade on update cascade,pid int(5),foreign key(pid) references placement(pid) on delete cascade on update cascade);
Query OK, 0 rows affected, 3 warnings (0.12 sec)

Perform following operations on above course

Q1. Write SQL Query to display the course wise student list?



Actual Output

```
mysql> select c.cname as 'Course Name',s.name as 'Student Name' from course c inner join
csjoin cs on c.cid=cs.cid inner join student s on s.sid=cs.sid;
+-------------+--------------+
| Course Name | Student Name |
+-------------+--------------+
| .NET        | ram          |
| .NET        | shyam        |
| JAVA        | ram          |
| JAVA        | shyam        |
+-------------+--------------+
4 rows in set (0.00 sec)
```

Q2. WA SQL query to display the course wise student count.

Syntax:

mysql> select c.cname as 'Course Name',count(cs.cid) from course c left join csjoin cs on c.cid=cs.cid left join student s on s.sid=cs.sid group by c.cid;

```
+-------------+---------------+
| Course Name | count(cs.cid) |
+-------------+---------------+
| JAVA        |             2 |
| .NET        |             2 |
| PHP         |             0 |
+-------------+---------------+
```

Q. Write SQL Query to show the course list who having more than 2 students?

Example:

mysql> select c.cname as 'Course Name',count(cs.cid) from course c left join csjoin cs on c.cid=cs.cid left join student s on s.sid=cs.sid group by c.cid having count(cs.cid)>2;

Output

```
+-------------+---------------+
| Course Name | count(cs.cid) |
+-------------+---------------+
| JAVA        |             3 |
+-------------+---------------+
1 row in set (0.00 sec)
```

Q. Write SQL Query to show the courses who having no admission?

mysql> select c.cname as 'Course Name',count(cs.cid) from course c left join csjoin cs on c.cid=cs.cid left join student s on s.sid=cs.sid group by c.cid having count(cs.cid)=0;

Output

```
+--------------+----------------+
| Course Name | count(cs.cid) |
+--------------+----------------+
| PHP          |              0 |
+--------------+----------------+
1 row in set (0.00 sec)
```

Q. Write SQL Query to show the student name with third highest package?

mysql> select s.name as 'Student name',p.package as 'Salary' from student s inner join csjoin cs on s.sid=cs.sid inner join placement p on p.pid=cs.pid order by p.package desc limit 1;

Output

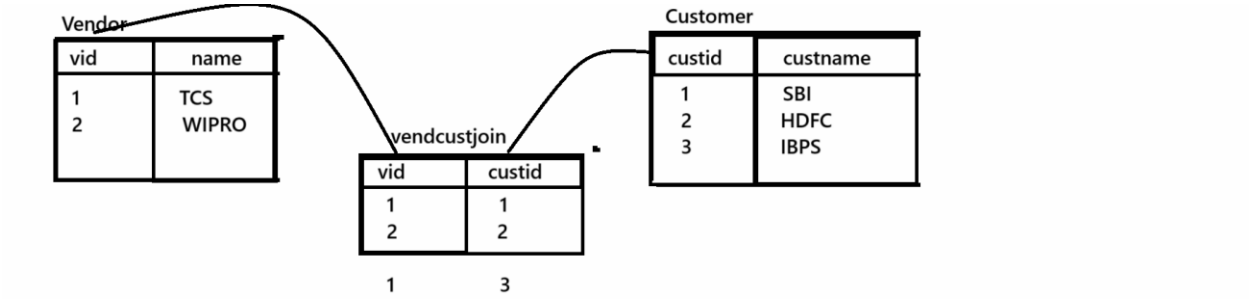
```
+--------------+--------+
| Student name | Salary |
+--------------+--------+
| dinesh       | 500000 |
+--------------+--------+
1 row in set (0.00 sec)
```

Example: Suppose consider we have following tables

Table1: Vendor

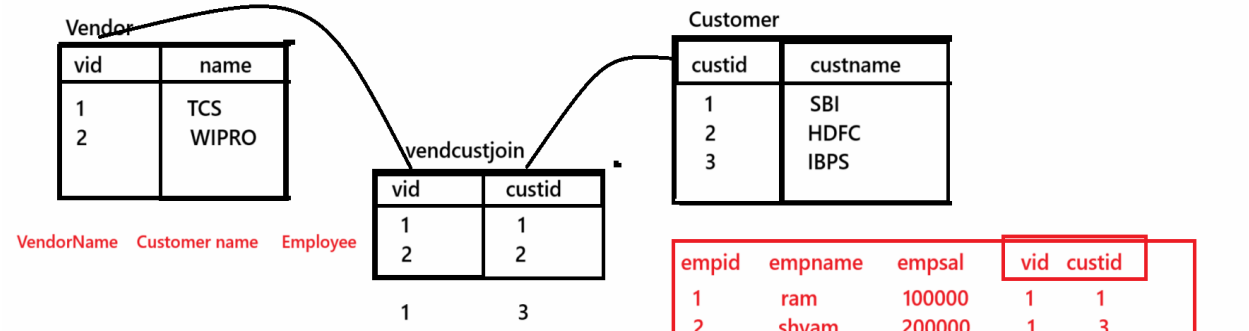
Table2: Customer

Table3: VendCustJoin




```
select v.name as 'Vendor Name',c.custname as 'Customer Name' from vendor v inner join vendcustjoin vcd on v.vid=vcd.vid inner join customer c
on c.custid=vcd.custid;
```

Example: find the join between vendor, employee and customer table

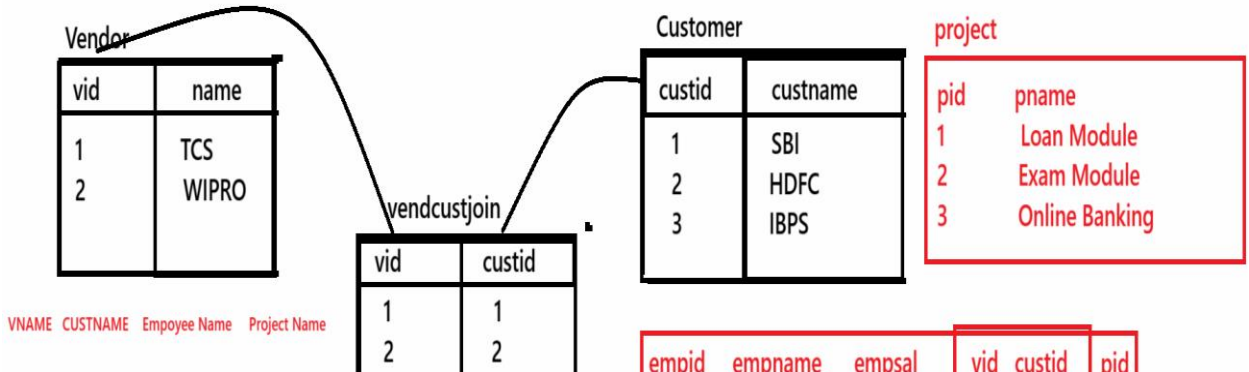


VendorName Customer name Employee

empid	empname	empsal	vid	custid
1	ram	100000	1	1
2	shyam	200000	1	3
3	ganesh	300000	2	2

```
select v.name as 'Vendor Name',e.empname as 'Employee Name',
c.custname as 'Customer name' from Vendor v inner join Employee e
on v.vid=e.vid inner join Customer c on c.custid=e.custiod
```

Example: find the vendor customer and employee with project list



VNAME CUSTNAME Empoyee Name Project Name

empid	empname	empsal	vid	custid	pid
1	ram	100000	1	1	1
2	shyam	200000	1	3	2
3	ganesh	300000	2	2	3

```
select v.name, c.custname,e.empname , p.pname  from Vendor v inner join Employee e on v.vid=e.vid inner join Customer c on c.custid=e.custid innerjoin project p on p.pid=e.pid
```

Normalization

Normalization is technique where we can decompose the large tables in to smaller tables for avoid data redundancy or data duplication.

Q. What happen if we not normalize database?

If we not normalize database then there is possibility following types of anomalies/confusion

a) Insertion Anomalies: Suppose consider we have following scenario

Admission

id	name	branch	HOD	Tel
1	ram	CSE	MR.X	334455
2	Shyam	IT	MR.Y	2233444
3	Ganesh	NULL	NULL	NULL

Note: if we think about above database table if we think student ganesh he is confuse about branch selection then there is insertion anomaly because when we want to insert ganesh data then we need to set branch,hod and tel as NULL or we cannot accept so it is not good approach so it is insertion anomaly.

b) Deletion anomalies:

Admission

id	name	branch	HOD	Tel
1	ram	CSE	MR.X	334455
2	Shyam	IT	MR.Y	2233444
3	Ganesh	CSE	MR.X	34455

delete from admission where HOD='MR.X';

Suppose consider MR.X leave the job and when we delete record of MR.X then student record also get deleted and branch record also get deleted so it is not real time solution because according this scenario we can say Dept is dependent on HOD means if MR.X leave the JOB then student get deleted means student also leave the college and dept also shut down it is not real life scenario because if HOD lave the dept then student data should persist or should not delete also dept info should not delete because we can replace HOD so above example say if we not normalize database then deletion anomalies may be occur.

C) Updating anomalies:

Admission

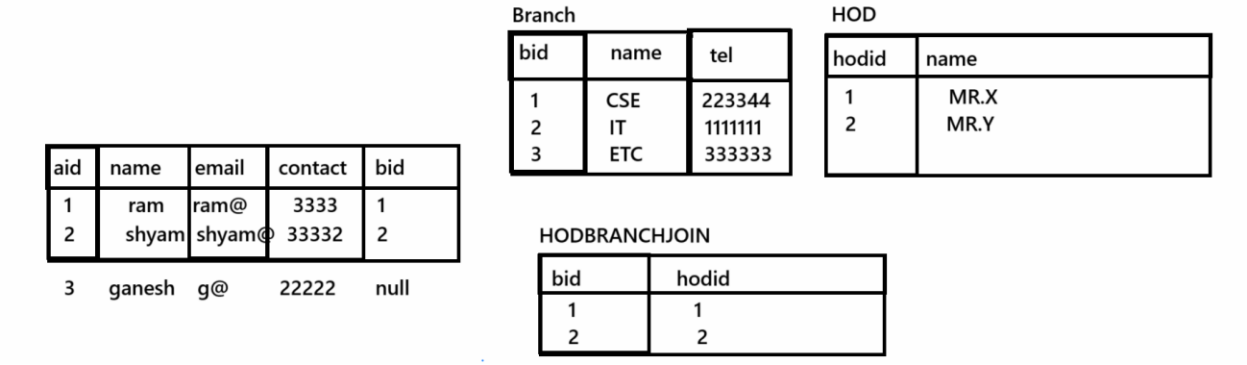
id	name	branch	HOD	Tel
1	ram	CSE	MR.X	334455
2	Shyam	IT	MR.Y	2233444
3	Ganesh	CSE	MR.X	34455

update Admission set HOD='MR.Z' where branch='CSE';

Note: suppose consider MR.X leave the JOB and college hire new HOD name as MR.Z then we need to update the MR.Z record as replacement of MR.X and if write update mention above then MR.Z update number of times according to student count in CSE DEPT so this updating anomalies means suppose if you have 1000 student for CSE then MR.Z record update 1000 times but it should be update only once so it is called as updating anomalies

Note: if we want to solve this type of problem we need to use normalization means divide the single tables in small tables and provide relationship between them

Example: Suppose if we consider our above table we have Admission table so if we divide this table in Two different tables like as Admission, Branch so above anomalies may not occur.



aid	name	email	contact	bid
1	ram	ram@	3333	1
2	shyam	shyam@	33332	2
3	ganesh	g@	22222	null

Branch

bid	name	tel
1	CSE	223344
2	IT	1111111
3	ETC	333333

HOD

hodid	name
1	MR.X
2	MR.Y

HODBRANCHJOIN

bid	hodid
1	1
2	2

If we want to work with normalization we have following types of normalization

1. First Normal Form:

Rules of First Normal Form

---

a) Column Name of table should be unique

b) Column must be atomic value

c) Column data and column type must be match.

d) Column Sequence does not matter.

Student

sid	name	coursename
1	Ganesh	JAVA
2	Manesh	JAVA,MERN
3	Dinesh	JAVA,MERN,.NET

Note: if we think about above table it is not in first normal form because if we think about coursename column it contain multiple values and we separate them by comm but the rule of first normal form is column must be atomic value and above table not satisfy above rule so we can say Student table not in first normal form.

If we want to organize your table in First Normal Form then you can organize like as

Student

sid	name	coursename
1	Ganesh	JAVA
2	Manesh	JAVA
2	Manesh	MERN
3	Dinesh	JAVA
3	Dinesh	MERN
3	Dinesh	.NET

If we think about above table we can say above Student table present in First Normal Form
But there data redundancy or duplication.
Means we can say If we think about First Normal form there is possibility of data duplication

2. Second Normal Form:

Rules of Second Normal Form

---

a) Table must be first normal form

b) Table should not have partial dependency

Q. What is partial dependency?

Before understanding partial dependency we need to know

Q. What is dependency?

Dependency means if particular column represent entire row uniquely in database table called as dependency.
Normally we can achieve dependency with the help of primary key column

Example of dependency

Employee Table

eid(pk)	name	salary	email	contact	age	desig
1	ram	100000	ram@	112233	24	Manager
2	shyam	40000	shyam@	445566	30	TL
3	ram	200000	rama@	112234	24	Manger

Note: if we think about table we can say if we want to identity employee uniquely we can identify it using eid cannot identity using name or salary or any other column and eid is our primary key so we can say employee identity is dependent on eid column called as dependency

Types of dependency

---

a) Partial dependency

b) Transitive dependency

c) Functional dependency etc

Q. What is partial dependency?

Partial dependency means if non primary key column is dependent on half portion of primary key columns column called as partial dependency.
Means if we have table which contain multiple foreign key(i.e primary key here we consider every foreign key is primary key somewhere so we cannot use multiple primary key table but can use multiple foreign key table means indirectly multiple primary key column in table in the form of foreign key) and if we have some non primary key which is dependent half portion of foreign key called as partial dependency

We want to understand above scenario with the help of following example.

Student

student_id	student_name	branch
1	RAM	CSE
2	SHYAM	IT
3	Ganesh	CSE

Subject

subject_id	subject_name
1	C
2	C++
3	DS

score

student_id	subject_id	marks	teacher_name
1	1	90	C Teacher
2	1	80	C Teacher
1	2	70	CPP Teacher

Suppose consider we have above database tables and if we think score table then we can say score table not in second normal form because it contain partial dependency

How we can prove score table contain partial dependency

Because if we think about score table it contain two non primary key column first is marks and second is teacher_name

Suppose consider if we think about marks column and if we have example student obtain 80 marks Then we have two questions

a) We need to know name of student : if identify student name we need to associate marks column with student_id means we need to know the student_id in score table because student_id of score table is associated with Student id in student table using student id we can fetch name of student for identify marks of student means here non primary key columns marks is dependent on student_id primary key column

b) We need to know subject name: if identify subject name we need to associate marks column with subject_id means we need to know the subject_id in score table because subject_id of score table is associate with subject id in Subject table using subject_id we can fetch name of subject of identify marks means here non primary key column marks is also dependent on subject_id means here we can say

marks column is completely dependent on student_id and subject_id so marks column not create partial dependency in score table.

So we have one more non primary key name as teacher_name.
Suppose consider we have scenario we want to identity C Teacher subject name here if we want to teacher subject then we need to know subject_id in score table not need to know the student id means teacher_name is non primary key column is only dependent on subject_id but we have two primary keys column in score table means teacher_name non primary column identification is not dependent on whole primary keys just it is dependent on half portion of primary key so here we can say teacher_name column generate the partial dependency so we can because teacher_name column score table not in second normal form.

If we want to arrange score table in second normal form then you have to create separate table teacher and provide association between teacher and subject with each other and suppose consider single teacher can teach more than one subject and single subject can have more than one teacher means we consider we have many to many relationship between teacher and subject then your database like as

Student

student_id	student_name	branch
1	RAM	CSE
2	SHYAM	IT
3	Ganesh	CSE

score

student_id	subject_id	marks
1	1	90
2	1	80
1	2	70

Subject

subject_id	subject_name
1	C
2	C++
3	DS

SubjectTeacherJoin

subject_id	tid
1	1
1	2
2	2
2	1

teacher

tid	teacher_name
1	C Teacher
2	CPP Teacher

3. Third Normal Form

Rules of third normal form

---

a) Table must be in second normal form
b) There should be no transitive dependency

Q. What is transitive dependency?

Transitive dependency means if non primary key is column is dependent on another non primary key column called as transitive dependency.

Now we want to discuss about the scenario for identify the transitive dependency

Student

student_id	student_name	branch
1	RAM	CSE
2	SHYAM	IT
3	Ganesh	CSE

score

student_id	subject_id	omarks	exam_type	totalmarks
1	1	90	Theory	100
2	1	80	Practical	100
1	2	70	Theory	100

Subject

subject_id	subject_name
1	C
2	C++
3	DS

SubjectTeacherJoin

subject_id	tid
1	1
1	2
2	2
2	1

teacher

tid	teacher_name
1	C Teacher
2	CPP Teacher

Note: if we think about table we have score table which is not in third normal form

Q. Why score not in third normal form?

Because score table maintain transitive dependency

How?

Suppose consider we have example we think student obtain 80 marks then we should have to know the following points.

a) We need to know Subject Name in which student obtain marks

---

if we want to identity the obtain marks subject then we can associate omarks column with subject_id in score using that we can identify in which subject student marks but here omarks is non primary key column and subject_ id primary key column so here is no transitive dependency

b) We need to know student name

---

if we want to identify the obtain marks student name we can associate omarks column with student_id in score table using that we can identify the student name here student_id is primary key column and omarks is non primary key here non primary key column is dependent on primary key column so here is no transitive dependency.

c) We need to exam type means in which exam student obtain marks

---

if we want to identity exam_type in which student obtain marks so we need to exam type and here omarks is also non primary key column and exam_type is also non primary key means here omarks no primary key column is dependent exam_type non primary key column so we can say these two columns create transitive dependency in score table so we can say score table not in third normal form.

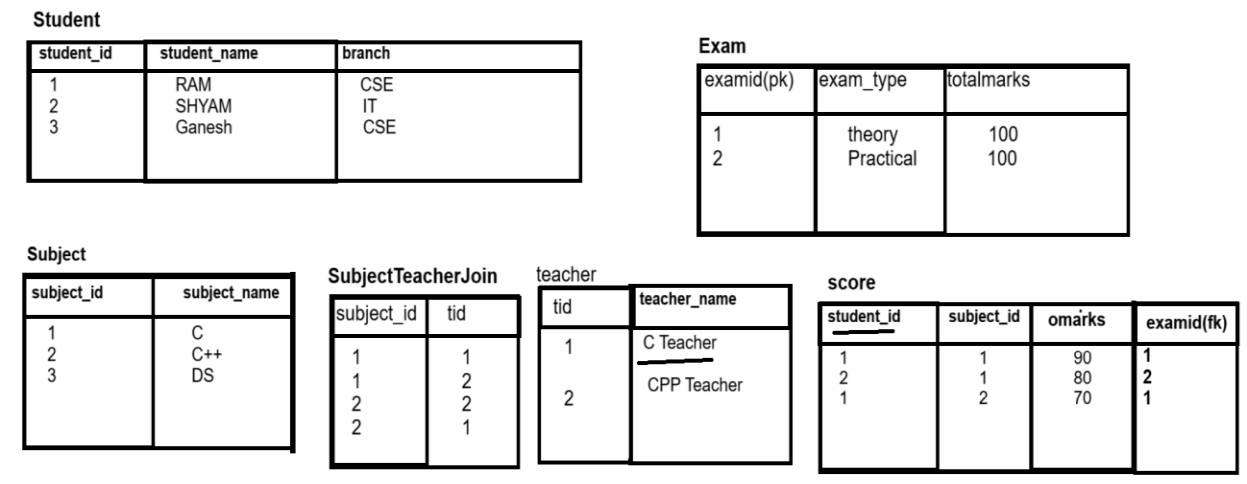
d) We need to total marks of exam:

---

if we want to check total marks of exam then we need to associate exam_type with totalmarks column and omarks column with totalmarks column means here exam_type is dependent on total marks and omarks is dependent on total marks here omarks is non primary key column and exam_type is also non

primary and total marks is also non primary here two non primary key columns dependent on non primary column so here also transitive dependency present so we can say score table not in third normal form

If you want to organize your score table in third normal form we have to create separate table for exam and store all exam info in that exam table just share examid in score for identify exam or obtain marks by student in which exam shown in following screen shot



Student

student_id	student_name	branch
1	RAM	CSE
2	SHYAM	IT
3	Ganesh	CSE

Exam

examid(pk)	exam_type	totalmarks
1	theory	100
2	Practical	100

Subject

subject_id	subject_name
1	C
2	C++
3	DS

SubjectTeacherJoin

subject_id	tid
1	1
1	2
2	2
2	1

teacher

tid	teacher_name
1	C Teacher
2	CPP Teacher

score

student_id	subject_id	omarks	examid(fk)
1	1	90	1
2	1	80	2
1	2	70	1

Note: remaining normal form we will discuss at end of SQL

4. Boyce code normal form or 3.5

5. Fourth Normal Form

6. Fifth Normal Form

Views

View is same like as virtual table in SQL the difference between table and view is view does not store data on disk just it is used for fetch data from table or already created table or perform other DML operation on table using SQL Statement and query.

Q. What are the benefits of view or why use the view if we have database table already?

---

1. Restricting data access: view provide additional level of table security by restricting access to predetermined set of rows and column of table.

Example: suppose consider we have Employee table with filed id,name,salary,email,contact,address etc and we provide to provide access to use only for id and name not other columns then we can apply view on employee table and provide access of particular column via view.

2. Hiding Data Complexity: if we have some complex query like as joins or sub query and if we want to use that query multiple time so you can hide that query in view means when we want to execute that query we not need to execute whole complex query just need to execute view.

3. Multiple view facility: means we can apply more than one view on single table or on same table. etc

How to create view practically

If we want to create view we have following syntax

Syntax: create view viewname as select column1,column2….column…n from tablename;

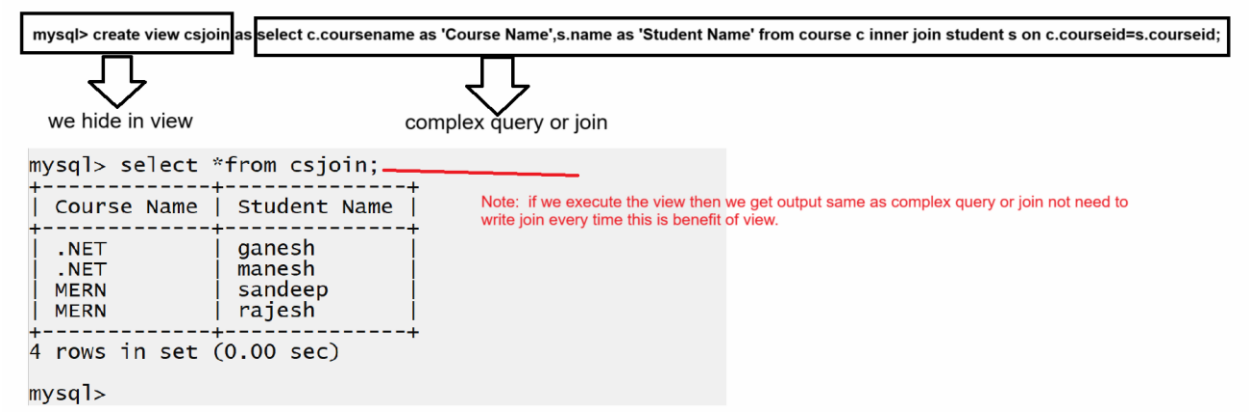
```
mysql> select *from employee;
+------+-------+--------+
| eid  | name  | salary |
+------+-------+--------+
|    1 | ram   |  20000 |
|    2 | shyam |  10000 |
+------+-------+--------+
2 rows in set (0.08 sec)
```

```
mysql> create view empview as select name,salary from employee;
Query OK, 0 rows affected (0.04 sec)

mysql> select *from empview;
+-------+--------+
| name  | salary |
+-------+--------+
| ram   |  20000 |
| shyam |  10000 |
+-------+--------+
2 rows in set (0.02 sec)

mysql>
```

Note: you can hide complex join view

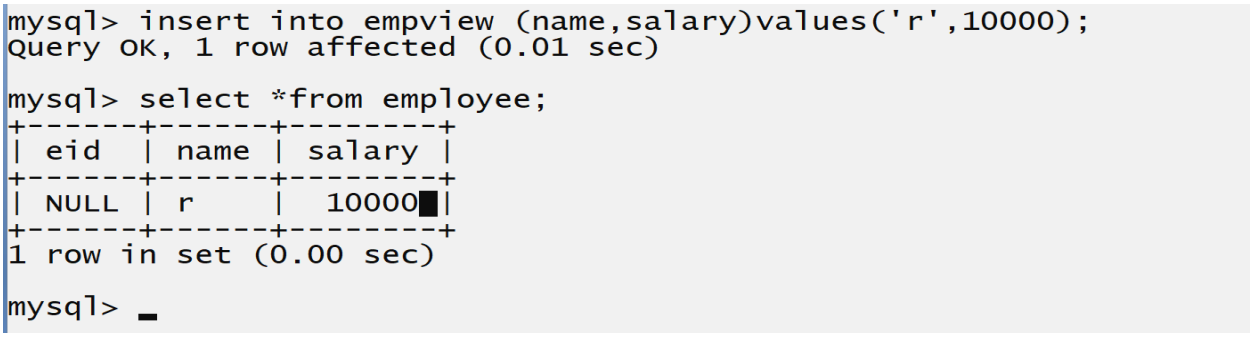
```
mysql> select *from course;
+----------+------------+
| courseid | coursename |
+----------+------------+
|        1 | .NET       |
|        5 | AWS        |
|        4 | HADOOP     |
|        3 | MERN       |
+----------+------------+
4 rows in set (0.03 sec)
mysql> select *from student;
+-----+---------+-------------------+---------+----------+
| sid | name    | email             | cotnact | courseid |
+-----+---------+-------------------+---------+----------+
|   1 | shyam   | shyam@gmail.com   | 34444   |     NULL |
|   2 | raj     | raj@gmail.com     | 344444  |     NULL |
|   3 | ganesh  | ganesh@gmail.com  | 1233255 |        1 |
|   4 | manesh  | manesh@gmail.com  | 13333   |        1 |
|   5 | sandeep | sandeep@gmail.com | 133333  |        3 |
|   6 | rajesh  | rajesh@gmail.com  | 133332  |        3 |
+-----+---------+-------------------+---------+----------+
6 rows in set (0.04 sec)
```

Example:



Note: you can perform insert or delete operation on table using join but only on that column which we mention in view

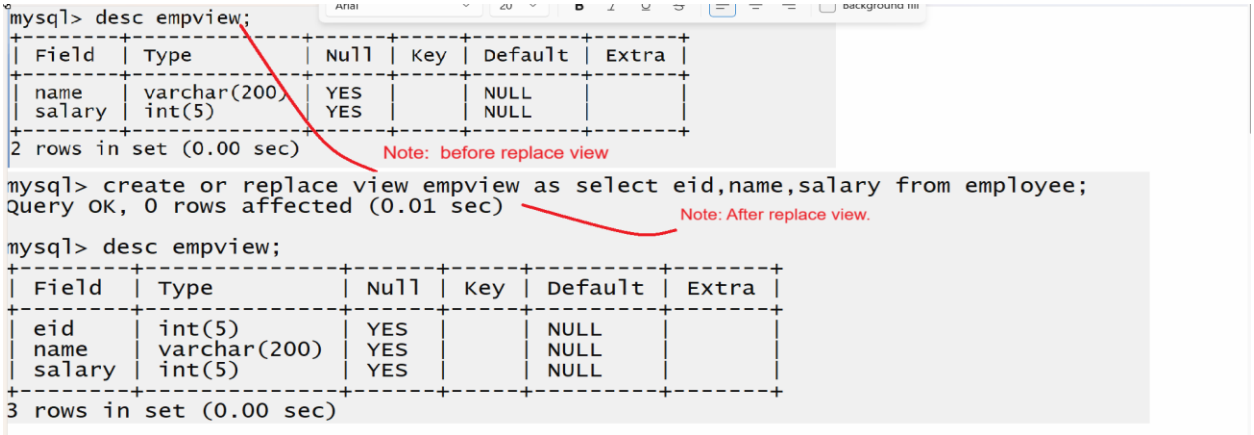
Note: join probably not recommend for non select operation it is specially recommended for select operation.



Note: if we want to update the view we have following command

Syntax: create or replace viewname as select query;

Note: create or replace command create view if it is not exist in database and if exist then override on previous view.



How to drop the view

If we want to delete view or drop view we have following command.

Syntax: drop view viewname;

Example: drop view empview;

```
mysql>  drop view empview;
Query OK, 0 rows affected (0.04 sec)

mysql> desc empview;
ERROR 1146 (42S02): Table 'aug2024.empview' doesn't exist
mysql>
```

Note: we get error because we drop the view means delete the view and after deletion we try to describe view so we get error and in erro we have Table does not exist because view consider as internally virtual table so database engine generate error as table with view.

SubQuery

SubQuery means query within query called as subquery we can use the sub query in all DML statements means in select query ,insert query,delete query etc

If we want to provide conjunction between outer query and inner query then we can use following operator

We can use following operator in some scenario

1. When sub query return single value as result then we can use following operator as conjunction between outer and inner query

< , > , = , <= ,>= ,!= etc

2. When Sub query return more than one value as result then we can use following operator as conjunction between outer and inner query.

in, any ,all or exists

Note: exists operator use when outer query reference wants to use in inner query

Generalize syntax of sub query

Syntax: select outer query conjunction (inner query)

Types of Sub Query

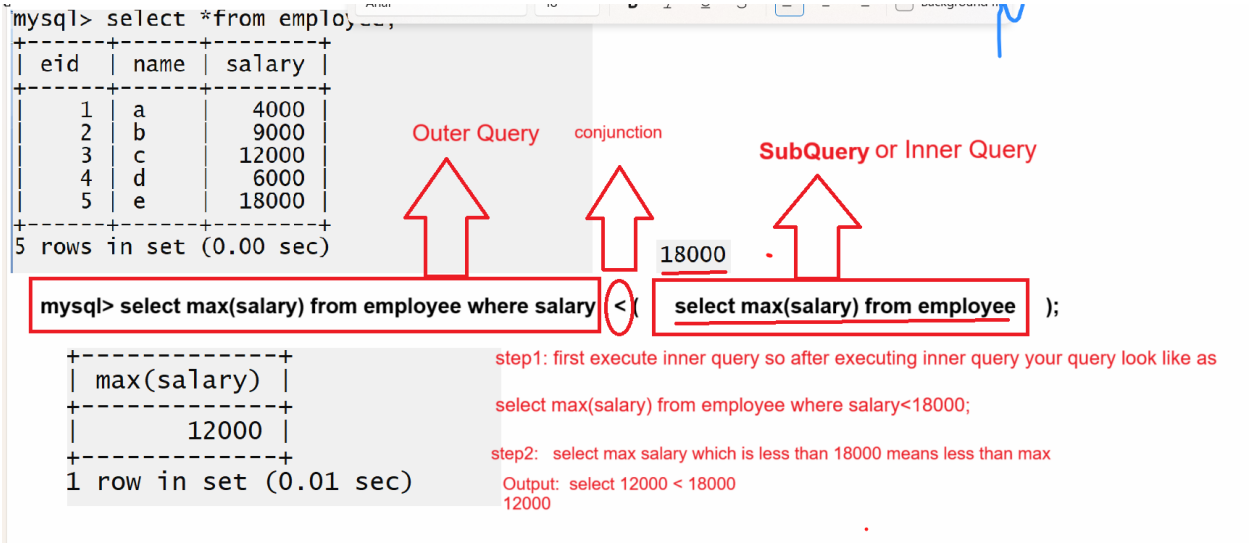
---

a) Inner Query: inner query is nested query which execute before outer query and inner query result pass as input to outer query.

b) Co-related SubQuery: co-related sub query means when we use the outer query reference as parameter in inner query called as co-related SubQuery.

Example of inner query

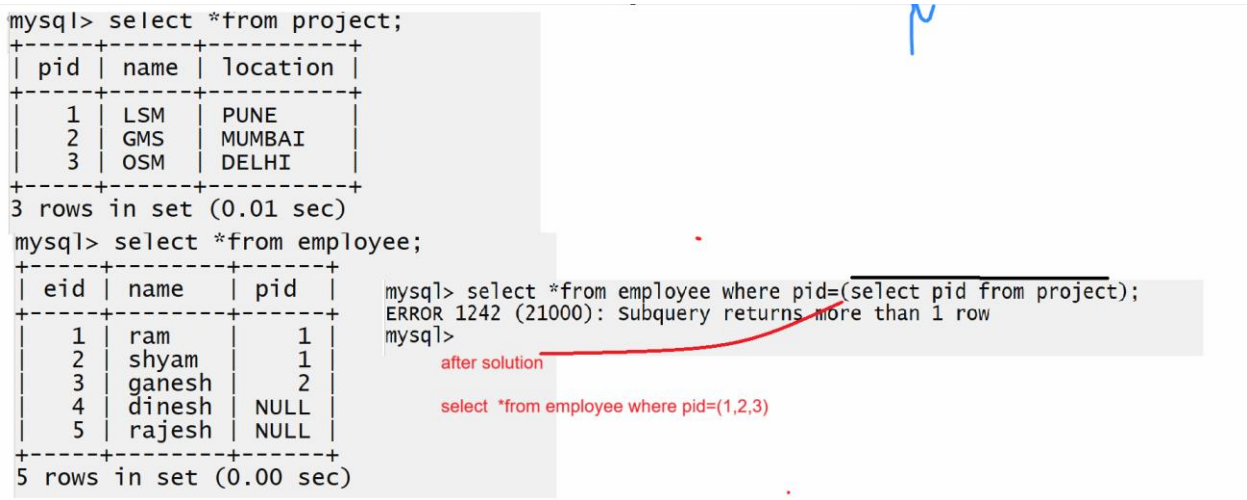
Example: Suppose consider we have employee table we want to find the second highest salary of employee table.



Example: print name of employee who is taking second highest salary
mysql> select *from employee where salary=(select max(salary) from employee where salary<(select max(salary) from employee));

```
+------+------+--------+
| eid  | name | salary |
+------+------+--------+
|    3 | c    |  12000 |
|    6 | f    |  12000 |
+------+------+--------+
2 rows in set (0.01 sec)
```

Example: Suppose consider we have following database tables.



Note: if we think about above query we get error

```
ERROR 1242 (21000): Subquery returns more than 1 row
mysql> select pid from project;
```

because if we think about given query

mysql> select *from employee where pid=(select pid from project)

here select pid execute first and this query generate result 1 , 2 3 according to our project table values and we cannot compare more than one values with single assignment or comparison operator because when we want to compare more than one values with condition then we can use logical operator like as OR , AND etc but we have IN operator is alternative for logical OR operator so if we want to solve above problem we required to use IN operator means when inner query return more than one values as result as output to outer query the we can use IN or any or all or exists but not here we IN operator for resolve above problem.

```
mysql> select *from project;
+-----+------+----------+
| pid | name | location |
+-----+------+----------+
|   1 | LSM  | PUNE     |
|   2 | GMS  | MUMBAI   |
|   3 | OSM  | DELHI    |
+-----+------+----------+
3 rows in set (0.01 sec)
```

mysql> select *from employee where pid IN(select pid from project);

solve inner query

select *from employee where pid IN(1,2,3); is equal with
select *from employee where pid=1 or pid=2 or pid=3

```
mysql> select *from employee;
+-----+--------+------+
| eid | name   | pid  |
+-----+--------+------+
|   1 | ram    |    1 |
|   2 | shyam  |    1 |
|   3 | ganesh |    2 |
|   4 | dinesh | NULL |
|   5 | rajesh | NULL |
+-----+--------+------+
5 rows in set (0.00 sec)
```

```
+-----+--------+------+
| eid | name   | pid  |
+-----+--------+------+
|   1 | ram    |    1 |
|   2 | shyam  |    1 |
|   3 | ganesh |    2 |
+-----+--------+------+
3 rows in set (0.02 sec)
```

Example: WA SQL Query find the employee who is working on same project

```
mysql> select *from employee;
+-----+--------+------+
| eid | name   | pid  |
+-----+--------+------+
|   1 | ram    |    1 |
|   2 | shyam  |    1 |
|   3 | ganesh |    2 |
|   4 | dinesh | NULL |
|   5 | rajesh | NULL |
+-----+--------+------+
5 rows in set (0.00 sec)
```

mysql> select *from employee where pid IN(select pid from employee group by pid having count(pid)>1);

select *from employee where pid IN(1)

```
+-----+-------+------+
| eid | name  | pid  |
+-----+-------+------+
|   1 | ram   |    1 |
|   2 | shyam |    1 |
+-----+-------+------+
2 rows in set (0.01 sec)
```

How to use exists() operator in SubQuery

---

when we use the exists() operator in subquery then we use the outer query reference in inner query and when we use exists() keyword in sub query then outer query execute first and compare outer query result with inner query and if outer query result match with inner query then return true otherwise return false.

Syntax:

select outer query where exists(inner query);

Example: WA SQL Query to find employee who is working on project by using exists operator

Syntax:

mysql> select *from employee e where exists(select pid from project p where e.pid=p.pid);

```
+-----+--------+------+
| eid | name   | pid  |
+-----+--------+------+
|   1 | ram    |    1 |
|   2 | shyam  |    1 |
|   3 | ganesh |    2 |
+-----+--------+------+
3 rows in set (0.00 sec)
```

Now we want to discuss about ANY and ALL operator

ANY or ALL operator allows you to perform comparison between single column value or range of column value.

The ANY Operator

The ANY Operator

a) Return Boolean value as result.

b) Return true if ANY of the sub query values meet the condition.

ANY means that the condition will be true if the operation is true for any of the value in the range.

Note: we can use ANY operator must be a standard comparison operator (=,<>,!=,>,>=,< or <=)

The ALL operator

---

a) Return Boolean value as result

b) Return TRUE if All SubQuery value meet the condition.

c) it used with select , where and HAVING condition also.

ALL means that the condition will be true only if the operation is true for all value in the range.

Suppose consider we have following tables

DEPT

deptid	deptname
1	DEV
2	PROD
3	TEST

select name and salary of employee whose salary is greater than all employee in deptid=1

select name,salary from employee where salary>ALL(select salary from employee where deptid=1)

eid	name	salary	contact	deptid
1	ram	10000	123456	1
2	shyam	20000	4444444	1
3	ganesh	50000	2222222	2
4	rajesh	6000	1211121	2
5	Sandeep	90000	3333333	3
6	sangram	15000	222222	3

select name,salary from employee where salary>ALL(10000,20000)

50000.

3 | ganesh | 50000 | 2222222 | 2

5 | Sandeep | 90000 | 3333333 |

mysql> create table dept(deptid int(5) primary key auto_increment,name varchar(200));
Query OK, 0 rows affected, 1 warning (0.05 sec)

mysql> create table employee(eid int(5) primary key auto_increment,name varchar(200),salary int(5),contact varchar(200),deptid int(5) ,foreign key(deptid) references dept(deptid));
Query OK, 0 rows affected, 3 warnings (0.08 sec)

Example: select employee detail whose salary is greater than all employee in deptid=1

```
mysql> select *from employee where salary>ALL(select salary from employee where deptid=1)
;
+-----+---------+--------+---------+--------+
| eid | name    | salary | contact | deptid |
+-----+---------+--------+---------+--------+
|   3 | ganesh  |  50000 | 1243445 |      2 |
|   5 | sandeep |  90000 | 1243444 |      3 |
+-----+---------+--------+---------+--------+
2 rows in set (0.04 sec)
```

Example: select the dept id with highest average salary

```
mysql> select *from employee;
+-----+---------+--------+---------+--------+
| eid | name    | salary | contact | deptid |
+-----+---------+--------+---------+--------+
|   1 | ram     |  10000 | 12345   |      1 |
|   2 | shyam   |  20000 | 123445  |      1 |
|   3 | ganesh  |  50000 | 1243445 |      2 |
|   4 | rajesh  |   6000 | 1243444 |      2 |
|   5 | sandeep |  90000 | 1243444 |      3 |
|   6 | sangram |  15000 | 1243444 |      3 |
+-----+---------+--------+---------+--------+
6 rows in set (0.00 sec)
```

select the dept id with highest average salary

10000
20000

mysql> select deptid,avg(salary) from employee group by deptid having avg(salary)>=ALL(select avg(salary) from employee group by deptid)

select deptid,avg(salary) from employee group by deptid having avg(salary)>=ALL(15000,28000,52500)

52500.

```
+--------+-------------+
| deptid | avg(salary) |
+--------+-------------+
|      3 |  52500.0000 |
+--------+-------------+
1 row in set (0.00 sec)
```

Example: select the dept name with highest average salary
mysql> select deptid,name from dept where deptid=(select deptid from employee group by deptid having avg(salary)>=ALL(select avg(salary) from employee group by deptid));

Output

```
+--------+------+
| deptid | name |
+--------+------+
|      3 | TEST |
+--------+------+
1 row in set (0.00 sec)
```

Example: select any employee who works with dept whose grade is A
mysql> select *from employee where deptid=ANY(select deptid from dept where grade='A');

```
+-----+---------+--------+---------+--------+
| eid | name    | salary | contact | deptid |
+-----+---------+--------+---------+--------+
|   1 | ram     |  10000 | 12345   |      1 |
|   2 | shyam   |  20000 | 123445  |      1 |
|   5 | sandeep |  90000 | 1243444 |      3 |
|   6 | sangram |  15000 | 1243444 |      3 |
+-----+---------+--------+---------+--------+
4 rows in set (0.00 sec)
```

Q. What is the difference between IN and exists?

IN	**Exists**
It can use for replacement of Multiple OR Condition	Determined any value are returned or not
Faster execution SubQuery executes only once	Slower operator executes force sub query re-execute again and again because in exists() outer query execute first and use its reference in inner query
Using IN operator SQL Engine compare all the values in the IN Clause	Once true is return in the exists condition then SQL engine will stop the process of further matching
IN execute from bottom to top approach	Exists execute top to bottom approach
IN operator cannot compare with null value	Exists can compare with null value

Example: using in() with null

```
mysql> select *from employee where deptid not IN(select deptid from dept);
Empty set (0.00 sec)
```

Note: There not output because IN () cannot compare with null values.

Example using exists() with null

mysql> select *from employee e where not exists(select deptid from dept d where e.deptid=d.deptid);

Output

```
+-----+---------+--------+---------+--------+
| eid | name    | salary | contact | deptid |
+-----+---------+--------+---------+--------+
|   7 | krushna |  50000 | 3434343 |   NULL |
|   8 | xyz     |  50000 | 3434343 |   NULL |
+-----+---------+--------+---------+--------+
```

Q. What is the difference between ANY and ALL?

ANY return true if any one condition matches in group of values otherwise return false and ALL can return true if all values in group match with conditions otherwise return false.

Q. What is the difference between ANY and IN?

IN	**ANY**
The IN operator is used to check if value matches any list or sub query and IN cannot use with relational operator	ANY operator is used as conjunction with a comparision = ,< ,> etc and is used for compare the value from list or sub query
IN specially design for avoid multiple OR	ANY specially use in SUB Query

Procedure IN SQL

Procedure is block of code or block of SQL which we can reuse more than one time and the goal of procedure is reuse SQL logic more than one time.

Means if we want to reuse SQL logics we need to call procedure

How to work with procedure practically

Steps to work with procedure

---

1. Set delimiter

Q. What is delimiter?

Delimiter is character which decide the end of SQL statement and the by default delimiter character is semi colon

```
mysql> delimiter //
mysql> select *from employee //
+-----+---------+--------+---------+--------+
| eid | name    | salary | contact | deptid |
+-----+---------+--------+---------+--------+
|   1 | ram     |  10000 | 12345   |      1 |
|   2 | shyam   |  20000 | 123445  |      1 |
|   3 | ganesh  |  50000 | 1243445 |      2 |
|   4 | rajesh  |   6000 | 1243444 |      2 |
|   5 | sandeep |  90000 | 1243444 |      3 |
|   6 | sangram |  15000 | 1243444 |      3 |
|   7 | krushna |  50000 | 3434343 |   NULL |
|   8 | xyz     |  50000 | 3434343 |   NULL |
+-----+---------+--------+---------+--------+
8 rows in set (0.00 sec)
```

end of SQL Statement

Note: when we work with Procedure, Trigger, Function or Cursor in SQL we need to set delimiter Other than semi colon when we write procedure then in procedure we have multiple SQL line of code and we close them using semi colon then SQL engine should not consider end of SQL statement so we required setting delimiter other than semi colon.

Syntax: delimiter character
Example: delimiter //

2. Create procedure:

If we want to create procedure using MYSQL then we have following syntax

```
delimter character

create procedure procedurename(IN | OUT | INOUT variablename datatype(size),variablename datatype(size))
begin
   write here your logics
end
delimiter character
```

If we think about procedure we have three types of parameter in procedure
IN: IN parameter normally used for accept the input parameter from procedure calling point to procedure definition point and by default procedure use the IN parameter
Note: when developer not specifies parameter type in procedure definition then MYSQL by default set IN as parameter in procedure.
OUT: OUT parameter specially used for get data from procedure definition to procedure calling point as output or resultant data.

INOUT: INOUT parameter is work as IN put and out parameter

Example: we want to create procedure name as saveDept with two IN parameters and save dept data in dept table using procedure

```
mysql> delimiter //
mysql> create procedure saveDept(IN deptName varchar(200),IN grade varchar(200))
    -> begin
    -> insert into dept values('0',deptName,grade);
    -> end
    -> //
Query OK, 0 rows affected (0.05 sec)

mysql>
```

Once we define procedure we can reuse it more than one time just we need to call the procedure

3. Call procedure

If we want to call procedure we have to use following syntax

Syntax: call procedurename(value1,value2,value…n);

Example:

```
mysql> delimiter ;
mysql> call saveDept('Director Body','A+');
Query OK, 1 row affected (0.01 sec)

mysql> select *from dept;
+--------+---------------+-------+
| deptid | name          | grade |
+--------+---------------+-------+
|      1 | DEV           | A     |
|      2 | PROD          | B     |
|      3 | TEST          | A     |
|      4 | HR            | A+    |
|      5 | Director Body | A+    |
+--------+---------------+-------+
5 rows in set (0.00 sec)

mysql>
```

How to get the out parameter using procedure

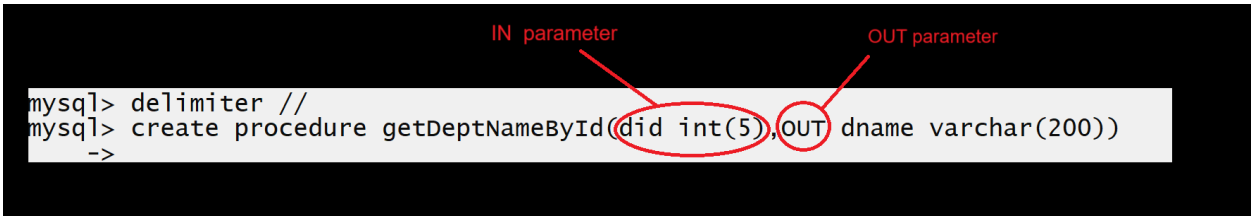
Suppose consider we want to create procedure which take id as input parameter and return name of dept as output parameter.

Note: Here dept id should be input parameter and dept name should be output parameter

Steps to work with Output parameter in procedure

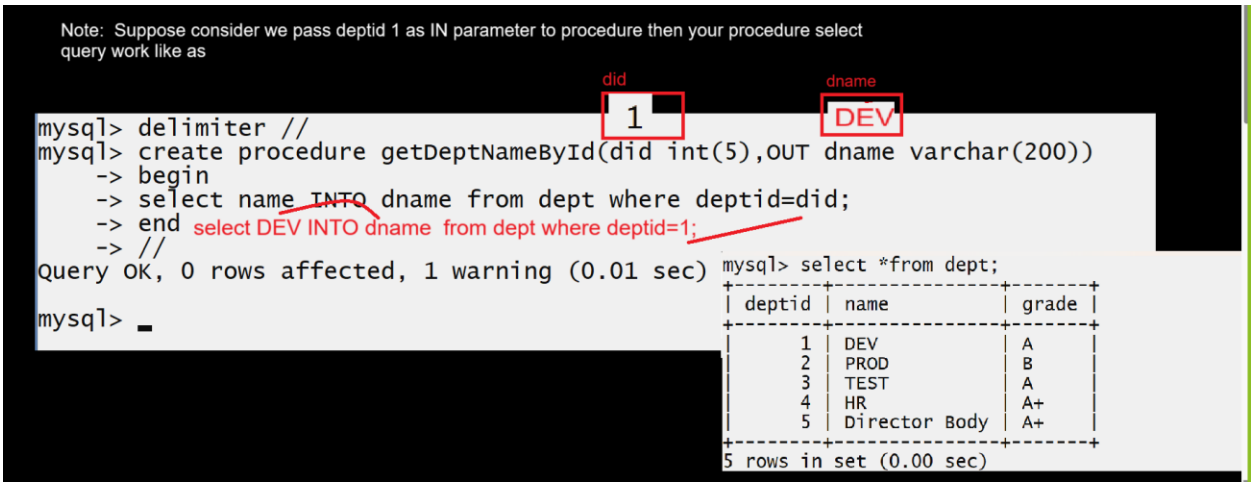
---

1. Declare procedure with output parameter using OUT keyword



2. Copy the value in out parameter using INTO keyword

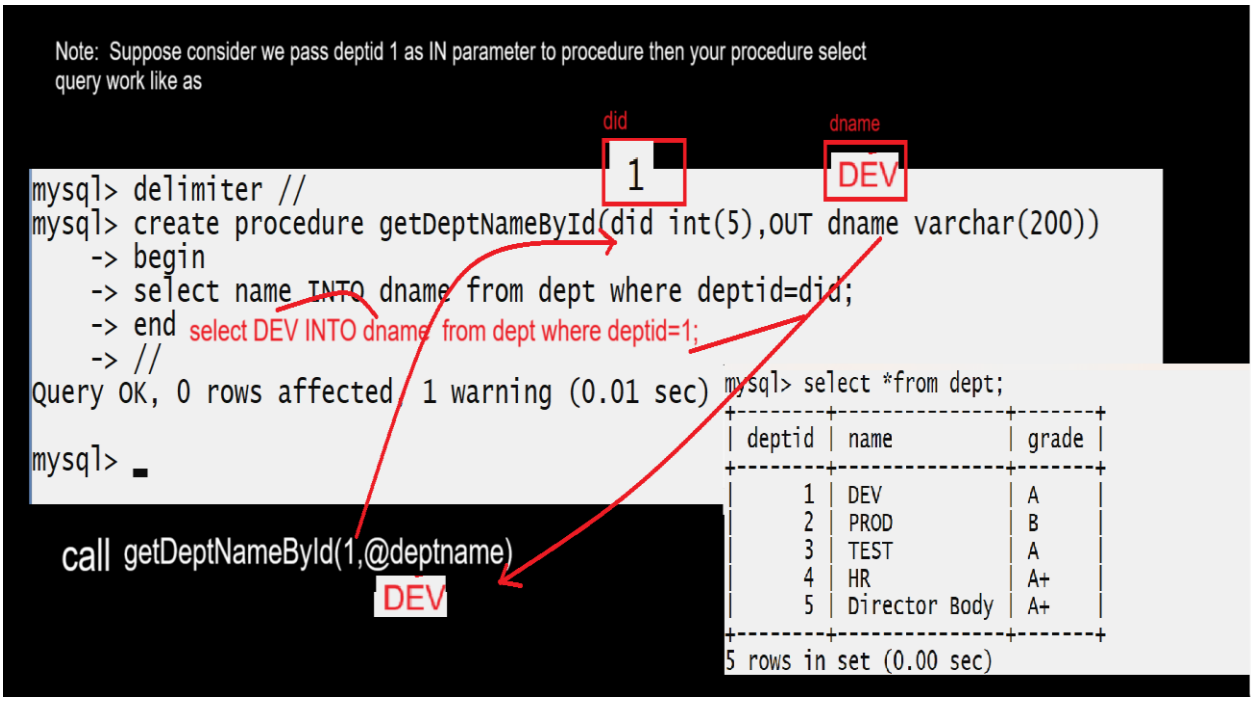
INTO keyword specially design for copy the value in out parameter



3. Get out parameter value at procedure calling point

Syntax: call procedurename(inparameter ,@outputparamvariablename);

Note: if we want to access out parameter value at procedure calling we can use variable using @symbol in procedure then we get output parameter



4. Display the output parameter value using select keyword

Final output given below

```
mysql> delimiter //
mysql> create procedure getDeptNameById(did int(5),OUT dname varchar(200))
    -> begin
    -> select name INTO dname from dept where deptid=did;
    -> end
    -> //
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> call getDeptNameById(1,@deptname)//
Query OK, 1 row affected (0.02 sec)

mysql> select @deptName;
    -> //
+------------+
| @deptName  |
+------------+
| DEV        |
+------------+
1 row in set (0.00 sec)

mysql>
```

Example: Create procedure to pass empid as IN parameter and get empname and empsal as output parameter

```
mysql> delimiter //
mysql> create procedure getEmpById(IN empid int(5),OUT empname varchar(200),OUT empsal int(5))
    -> begin
    -> select name,salary INTO empname,empsal from employee where eid=empid;
    -> end
    -> //
Query OK, 0 rows affected, 2 warnings (0.03 sec)

mysql>
```

```
mysql> select @ename as 'Employee Name', @esal as 'Employee Salary'//
+---------------+-----------------+
| Employee Name | Employee Salary |
+---------------+-----------------+
| ram           |           10000 |
+---------------+-----------------+
1 row in set (0.01 sec)

mysql>
```

Trigger in MYSQL

Q. What is trigger in MYSQL?

A MYSQL trigger is stored program (with queries) which executed automatically to respond to specific event such as insert/delete/updation occurring on table.

The Major difference between procedure and trigger is procedure need to call manually by developer but trigger can execute by database engine automatically on DML operation

There are six different types of trigger in MYSQL

---

1. before update trigger : this type of trigger execute automatically before update it means if we apply trigger on update operation on specified database then this type of trigger update automatically.
2. After Update trigger: this type of trigger executed automatically after update it means we apply trigger on update on specified table then this trigger automatically execute after updation.
3. Before Delete Trigger: this trigger automatically executed before deletion of operation**.**
4. After Delete trigger: this trigger automatically executed after deletion of operation.
5. before Insert Trigger: this trigger automatically execute after insertion operation
6. after Insert Trigger: this trigger automatically execute after deletion operation

How to create trigger practically

Delimiter character
Create trigger triggername [before | after] [insert | delete | update] on tablename
for each row
 begin
  Write her your logics
end
Character

Note: when we work with trigger in MYSQL then we need to use two keyword names as old and new
old: this keyword help us to access the values from target table column before change means here we consider target table means a table on which we apply trigger.
new: this keyword help us to access the values from target table column after change.

Example: Suppose consider we are working on banking application and we have two tables Account table and one is mini statement table so we want to track the activity of accounts means when someone credit the amount in account then mini statement should update with previous balance as well as new balance and if someone withdraw amount from account then mini statement should update
Note: mini statement should not update manually by bank employee we want to mini statement table automatically update by database engine automatically then we can use the trigger

mysql> create table account(aid int(5) primary key auto_increment,name varchar(200),balance int(5));
Query OK, 0 rows affected, 2 warnings (0.16 sec)

mysql> create table ministmt(tid int(5) primary key auto_increment,aid int(5),foreign key(aid) references account(aid), prevBal int(5),newBalance int(5),tdate date);

```
mysql> delimiter //
mysql> create trigger upministmt after update on account
    -> for each row
    -> begin
    -> insert into ministmt values('0',old.aid,old.balance,new.balance,curdate());
    -> end
    -> //
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> update account set balance=balance-9000 where aid=1;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select *from ministmt;
+-----+------+---------+------------+------------+
| tid | aid  | prevBal | newBalance | tdate      |
+-----+------+---------+------------+------------+
|   1 |    1 |   10000 |      15000 | 2025-01-24 |
|   2 |    1 |   15000 |       6000 | 2025-01-24 |
+-----+------+---------+------------+------------+
2 rows in set (0.00 sec)

mysql>
```

Q. What is index?

Index is specialized lookup table that are used by database search engine to fast data retrieval purpose. Means we can say index in SQL is tool used to quickly identify rows with specified column values if there is no index in SQL table server would have to start with first row and go through the entire table until it discovers the relevant rows this method known as full table scan which is not efficient way to search data in large table set.

Q. Why indexes in SQL are used or what are the benefits of index?

---

a) Improve Query performance: The primary reason for using index is to accelerate or speed up the query processing. Indexes can drastically reduce the amount of data the server needs to examine or check.

b) Efficient data access: index helps us to quick a way to access row data for select statement. This is particularly beneficial for table with large number of row.

c) Sorting and grouping speed: indexes improve the speed of data retrieval operation by providing sorting version of the data which is faster to process for order by and group by operations.

d) Unique Constraints: indexes help us to enforce uniqueness for column to ensure that no two rows of table have duplicate values in particular column or a combination of columns

e) Optimized Join operations: In database with multiple tables indexes improve the speed of join operations by quickly locating the joining row of each table.

Q. What are the limitations of indexes?

---

a) Storage space : index consume addition disk space in memory means we need to give space to index as well as space to disk space.

b) Maintenance overhead: index need to maintain and rebuild over the time which add overhead of database

c) Possibility of slow down insert/delete/updation operation: while indexes help us to faster retrieval operations they can slow down data input through insert/update/delete statement because index internally update when we update the data is modified.

Types of indexes

Clustered index

A clustered index sorts and store the row of table based on the values in one or more specified columns and each table can have only one clustered index and the choice of the clustering column impact how data internally store and retrieve

Example: primary key is an example of clustered index.

Non Clustered index: non clustered index is type of index used in relational database management system to improve the efficiency of data retrieval operations unlike clustered indexes which affect the physical order of data rows within a table non clustered index create separate data structure to allow fast access to data subsets means non clustered index not arrange in physical order but create separate data structure for fast data accessing purpose

Note: single table can have more than one non clustered indexes means developer can apply index on more than one column on database table.

Example: user defined is example of non clustered index

How to create using MYSQL

Syntax: create index indexname on tablename(columnname);

or

Syntax: create index indexname on tablename(columnname1,columnname,…column…n);

```
mysql> create index empind on employee(name);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> select name from employee;
+---------+
| name    |
+---------+
| ganesh  |
| krushna |
| rajesh  |
| ram     |
| sandeep |
| sangram |
| shyam   |
| xyz     |
+---------+
8 rows in set (0.00 sec)
```

Q. What is difference between clustered and non clustered index?

Non-Clustered Index	**Clustered Index**
Does not determine the physical order of data rows	Determines the physical order of data rows based on index column
Create separate data structure containing index entries with pointes to the corresponding data rows	Organizes the actual data rows in the table in specified order
Allow for multiple non clustered index in single table	Restrict table to having only one clustered index.
Suitable for optimizing query performance when the query does not align with the physical data order	Ideal for queries that frequently retrieve data in the same order as the cluster column

How to drop index using MYSQL

Syntax: drop index indexname;

How to declare variable in SQL

If we want to declare variable in SQL we have to use following syntax
Syntax: DECLARE variablename datatype default value ;
Example:DECLARE no INT;
How to initialize the value in variable

if we want to initialize value in variable then we have SET keyword in variable
Example: suppose we have no variable and we want to initialize value 100 in no variable then your syntax like as
SET variablename=value;
Note: here SET keyword indicate replace the value in variable
Example: SET a=100;

Example: we want to create procedure name as add() this procedure can declare two variables and set 100 and 200 values in it and calculate its addition and display it.

```
mysql> create procedure addTwoSum()
    -> begin
    -> DECLARE a INT;
    -> DECLARE b INT;
    -> SET a=100;
    -> SET b=200;
    -> select a+b;
    -> end
    -> //
Query OK, 0 rows affected (0.08 sec)

mysql> call addTwoSum();
    -> //
+------+
| a+b  |
+------+
|  300 |
+------+
1 row in set (0.04 sec)

Query OK, 0 rows affected (0.04 sec)
```

Example: we want to create procedure name as MulTwoVal() and declare two variable a and b and initialize value to them at the time of declaration and calculate its multiplication and display it.

```
mysql> create procedure MulTwoValDefault()
    -> begin
    -> DECLARE a INT DEFAULT 10;
    -> DECLARE b INT DEFAULT 20;
    -> DECLARE c INT;
    -> SET c=a*b;
    -> select c as "Multiplication";
    -> end
    -> $$
Query OK, 0 rows affected (0.06 sec)

mysql> call MulTwoValDefault()$$
+----------------+
| Multiplication |
+----------------+
|            200 |
+----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

How to use IF else in MYSQL

Syntax:

IF Condition THEN

Write here logics

ELSE

Write here logics

END IF

Example: we want to create procedure and pass two values in it and check which is greater

```
mysql> delimiter //
mysql> create procedure checkGreater(a INT(5),b int(5))
    -> begin
    -> IF a>b THEN
    ->  select "A is Greater";
    -> ELSE
    ->  select "B is Greater";
    -> END IF;
    -> END
    -> //
Query OK, 0 rows affected, 2 warnings (0.04 sec)

mysql> call checkGreater(100,5)//
+--------------+
| A is Greater |
+--------------+
| A is Greater |
+--------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

If we want to write switch case in MYSQL then we have to use following syntax

Syntax:

CASE

WHEN condition then

Logic

When condition then

Logic

ELSE

Logics

END

Example: Create procedure which accepts two parameter as input and 1 parameter as choice if choice is 1 then calculate the addition of two values if choice is 2 then calculate the multiplication of two values.

```
mysql> create procedure menuapp(a int(5),b int(5),c int(5))
    -> begin
    -> CASE
    ->  when c=1 THEN
    ->   select a+b as "Addition";
    ->  when c=2 THEN
    ->   select a*b as "Multiplication";
    ->  ELSE
    ->   select "wrong choice";
    -> END CASE;
    -> END//
Query OK, 0 rows affected, 3 warnings (0.17 sec)
```

```
mysql> call menuapp(10,20,2)//
+-----------------+
| Multiplication |
+-----------------+
|             200 |
+-----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Example: WA SQL Query to fetch employee name and salary from database table if salary is greater than 50000 then display salary is greater than 50000 otherwise display salary is less than 50000.

mysql> select name, CASE when salary>50000 THEN 'employee salary is greater than 50000' ELSE 'employee salary is less than 50000' END as 'employee salary message' from employee;

How to use loop in MYSQL

Syntax:

```
label_name:LOOP
    write here logic
   IF Condition THEN
       leave label_name;
  END IF
END LOOP;
```

Example: we want to design procedure and print the good morning 5 times using loop.

```
mysql> create procedure printGoodMsg()
    -> begin
    -> DECLARE i INT;
    -> SET i=1;
    -> msg_loop:LOOP
    -> SET i=i+1;//2
    ->  select "Good Morning";
    ->  IF i=5 THEN
    ->   LEAVE msg_loop;
    ->  END IF;
    -> END LOOP;
    -> end
    -> //
Query OK, 0 rows affected (0.04 sec)

mysql>
```

i
2

Example: Create procedure which accepts number as input and print the table of number.

```
mysql> delimiter //
mysql> create procedure tableNum(no int(5))
    -> begin
    -> DECLARE i INT DEFAULT 1;
    -> tab_loop:LOOP
    -> select i*no as "table";
    -> SET i=i+1;
    -> IF i=10 THEN
    ->  leave tab_loop;
    -> END IF;
    -> END LOOP;
    -> end//
Query OK, 0 rows affected, 1 warning (0.04 sec)
```

```
mysql> create procedure factorial(no int(5))
    -> begin
    -> DECLARE f INT DEFAULT 1;
    -> DECLARE i INT;
    -> SET i=1;
    -> f_loop:LOOP
    -> SET f=f*i;
    -> IF i=no THEN
    ->  leave f_loop;
    -> END IF;
    -> SET i=i+1;
    -> END LOOP;
    -> select f as "Result is ";
    -> END
    -> $$
Query OK, 0 rows affected, 1 warning (0.04 sec)

mysql> call factorial(5)$$
```

How create user define function using MYSQL

Function is block of statement like as procedure but the major difference between procedure and function is procedure cannot return value and function can return value as well as procedure can call manually and function cannot call manually we can use function within SQL statement means we can use function in insert/delete/update/select statement

Syntax to create function in MYSQL

---

delimiter character
create function functionname(parameter1,parameter2…..parameter…n) returns datatype[characteristics]
begin
  function body
end

Description about function in MYSQL

Create function: this is the keyword which helps us to create function using MYSQL
Function name: this is the function name and user can provide any function name according to his choice
parameter1….parametern…n: here we pass number of parameter IN,OUT,INOUT
returns datatype: here we specify what kind of value return by function or what kind of data type value return by function

Characteristics: this parameter decide what kind of value return by function means like as function returning value is SQL statement or normal value etc
When we use function then we have use following characteristics
{DETERMINISTICS, NO SQL or READS SQL} is specified in declaration

Q. What is DETERMINISTICS?

IN MYSQL a deterministic function is a function which returns same result when given the same input values.

Q. What is NOSQL?

NO SQL indicate routine or function contain no SQL Statements.

Example: Create function which accept number as parameter and return its square

```
mysql> create function getSq(val int(5)) returns int DETERMINISTIC
    -> begin
    -> DECLARE result INT;
    -> SET result=val*val;
    -> return result;
    -> END
    -> //
Query OK, 0 rows affected, 1 warning (0.03 sec)

mysql> select getSq(5)//
```

Note: If we use the function in SQL then we must be execute or call function within SQL Statement like as in select or insert or delete or update etc

Note: we cannot call function like as procedure and if we try to call it like as procedure we get following error.

```
mysql> call getSq(5)//
ERROR 1305 (42000): PROCEDURE mysql.getSq does not exist
mysql> _
```

Example: Create function which accept empid and fetch employee name and return employee name from function

Example with source code

mysql> create function getEmpNFun(eid int(5),empname varchar(200)) returns varchar(200)
DETERMINISTIC
-> begin
-> select name INTO empname from employee where empcode=eid;
-> return empname;
-> END$$
Query OK, 0 rows affected, 1 warning (0.04 sec)

```
mysql> select getEmpNFun(100,@empname)$$
+--------------------------+
| getEmpNFun(100,@empname) |
+--------------------------+
| Rajdeep                  |
+--------------------------+
1 row in set (0.04 sec)
```

Cursor in SQL

A MYSQL Cursor is database object that enables the end user to retrieve process and scroll through rows of the result one at time Using cursor we can perform operation on one row at time.
This can very useful for complicated data manipulsations and procedural logics

Steps to work with Cursor in MYSQL

---

1. Declare Cursor to intiailize the memory

Syntax: DECLARE cursorname CURSOR as select query

2. Open the cursor to allocate memory

Syntax: open cursorname;

Note: when we work with cursor we need to set flag variable for terminate condition
DECLARE variablename INT DEFAULT false
DECLARE CONTINUE handler For NOT FOUND SET variable=true

3. Fetch cursor to retrive data

Syntax: Fetch cursorname into variablenames ….
Note: fetch is used for retrive data from cursor

4. Close the cursor to release the memory
close cursorname;

Example with source code

```
mysql> create procedure empcurfetch2()
   -> begin
   -> DECLARE done INT DEFAULT false;
   -> DECLARE empname varchar(200);
   -> DECLARE sal INT;
   -> DECLARE empcursor CURSOR for select name,salary from employee;
   -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=true;
   -> open empcursor;
   -> read_loop:LOOP
   -> fetch empcursor into empname,sal;
   -> IF done=1 THEN
   ->  LEAVE read_loop;
```

-> END IF;

-> select empname,sal;

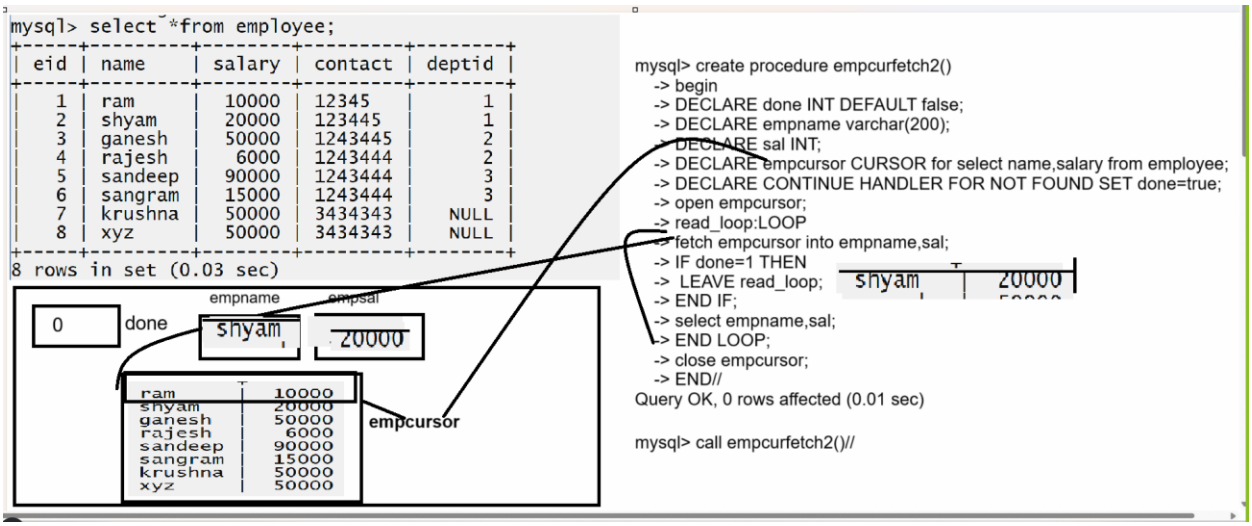
-> END LOOP;

-> close empcursor;

-> END//

Query OK, 0 rows affected (0.01 sec)

mysql> call empcurfetch2()//



ACID Properties in DBMS

ACID Stands for atomacity consistency isolation and durability called as ACID properties.