1. **foreign key** = It is used to relate two tables.

-> It can be applied column level and table level.

1. **Char of foreign key** :-

A table in which foreign key is present know as child table, and in which it is present as a primary key know as parent table.

1. Foreign key must and should be define as primary key in its parent table.
2. A column assign as a foreign key can be accept null or can accept duplicate value.
3. We can not insert a value in a column define as a foreign key which is not present in a parent table column.

**Check :-**

1. This constraints is used to restrict the value of a column between a range.
2. Its like a condition checking

* **Data types** :-

1. Data types selection is usually dictated by nature of data and by intented use.
2. Pay close attention to expected use of attribute for shorting and data retrieved purpose.

**Example**:- bit , tinynt , smallint, int ,bigint, decimal ,numeric, float ,real, char, varchar, text ,varchar(max)

* Char has fixed length of 8000 characters. means memory allocation.
* Varchar has variable length of 8000 characters.
* DATE :- YYYY-MM-DD
* TIME:- HH:MI:SS
* DATETIME:- YYYY-MM-DD HH:MI:SS
* TIMESTAMP
* YEAR
* CLOB:- character large object that can hold upto 2gb
* BLOG:- binary large object used to store audio video in terms of binary form.
* XML :- for storing xml data
* JSON:- for storing json data

**Note**:-while creating a table data type is mandatory to assign to the column but constraint are not mandatory but highly recomanded.

**SQL** is in-case-sensetive.

**History:-**

IBM was the first compony to develop DBMS which follow relational model which now known as system R model .

IBM develop a lag to communicate with sys R the lang.name is SEQUL stands for simple English query lang. due to its simplicity it was very popular at that time in 1980 ANSI acquires SEQUL lang. and after doing some modification it recommended then it is know as SQL . and made as a standard lang. for RDBMS.

* Sql is a programming lang where mysql is a database application.
* Structure query language it is a standard lang which is used for storing and managing data in database.
* Queries and other operation:- select ,add, update, insert ,delete ,alter ,trunk.
* Fives sql statements

DDL data definition : create,alter,drop,truncate,rename

DQL data query language

DCL data control language

TCL transactional control language

DML data manipulating language

* **For create database:**

mysql> create database ustglobal;

Query OK, 1 row affected (0.08 sec)

* **For see how many databases present:**

mysql> show databases;

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

| Database |

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

| information\_schema |

| mysql |

| performance\_schema |

| sys |

| ustglobal |

* **For use that database:**

mysql> use ustglobal;

Database changed

* **For cheak how many tables/data present in my database:**

mysql> show tables;

Empty set (0.01 sec)

* **For create the table ;**

**Syntax create table tablename (columnname datatype constrain(if any), columnname datatype constraint(if));**

mysql> create table student (id int,name varchar(120),emailId varchar(150));

Query OK, 0 rows affected (0.48 sec)

* **For see the table in the database:**

mysql> show tables;

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

| Tables\_in\_ustglobal |

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

| student |

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

1 row in set (0.00 sec)

* **For getting the description of the table:**

mysql> describe/desc student;

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

| Field | Type | Null | Key | Default | Extra |

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

| id | int(11) | YES | | NULL | |

| name | varchar(120) | YES | | NULL | |

| emailId | varchar(150) | YES | | NULL | |

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

3 rows in set (0.01 sec)

* **For drop the table (drop) will delete the whole table as well as data:**

mysql> drop table student;

Query OK, 0 rows affected (0.20 sec)

* **For drop the whole database:**

mysql> drop database ustglobal;

Query OK, 0 rows affected (0.16 sec)

* **For delete one column from the table:**

mysql> alter table friends drop column emailId;

Query OK, 0 rows affected (0.43 sec)

* **For add column in the table:**

mysql> alter table friends add emailId varchar(150);

Query OK, 0 rows affected (0.44 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> desc friends;

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

| Field | Type | Null | Key | Default | Extra |

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

| name | varchar(120) | YES | | NULL | |

| mobileNo | int(11) | YES | | NULL | |

| address | varchar(180) | YES | | NULL | |

| emailId | varchar(150) | YES | | NULL | |

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

* **To insert the data into the table:**

mysql> insert into simran(bff,mobileNo,DOB) values('luqhmaan',87,’1996-08-03’);

Query OK, 1 row affected (0.06 sec)

mysql> select \* from simran;

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

| bff | mobileNo | DOB |

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

| luqhmaan | 87 | 1985 |

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

1 row in set (0.00 sec)

* **To insert the two record at a same time:**

mysql>insert into simran values('ghfgh',768,68),('bjhds',687,878);

Query OK, 2 rows affected (0.05 sec)

Records: 2 Duplicates: 0 Warnings: 0

* **To insert the data into few column:**

mysql> insert into simran(bff,mobileNo) values('hj',677);

Query OK, 1 row affected (0.06 sec)

* **Alias :-** it is a temporary name which is give to the table or column.

***Syntax:-*** select column\_name AS alias\_name from table\_name

***Syntax:-*** select column\_name from table\_name AS alias\_name

Q= WAQTD empname ,anualsal,designation ?

mysql> select name,salary\*12 as anual\_salary,designation from employee;

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

| name | anual\_salary | designation |

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

| ranu | 653196 | tester |

| simu | 144000 | tester |

| luqhmaan | 720000 | HR |

| raju | 72000 | clerk |

| riya | 480000 | software developer |

| ritu | 360000 | analyst |

| rinku | 228000 | salesman |

| sukaniya | 96000 | clerk |

| sakshi | 300000 | software developer |

| sanju | 547920 | HR |

| saphu | 427920 | analyst |

| anju | 240000 | salesman |

| palak | 120000 | clerk |

| manik | 480000 | software developer |

| manish | 360000 | HR |

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

15 rows in set (0.05 sec)

Q: WAQ to give the alias name for all the column present in the employee table?

mysql> select id a,name b,emailId c,mobileNo d,age e,DOB F,DOJ G,Salary h,DeptNo i,BloodGroup j from employee;

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

| a | b | c | d | e | F | G | h | i | j |

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

| 1 | ranu | ranu@gmail | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 2 | simu | simu@gmail | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 3 | luqhmaan | luqh@gmail | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ |

| 4 | raju | raju@gmail | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- |

| 5 | riya | riya@gmail | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 6 | ritu | ritu@gmail | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 7 | rinku | riku@gmail | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 8 | sukaniya | suku@gmail | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 9 | sakshi | saku@gmail | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 10 | sanju | sanju@gmail | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 11 | saphu | saphu@gmail | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

| 12 | anju | anju@gmail | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- |

| 13 | palak | palu@gmail | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- |

| 14 | manik | manu@gmail | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- |

| 15 | manish | manish@gmail | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- |

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

15 rows in set (0.00 sec)

* **Distinct**

The select distinct statement is used to return only diff. value

***Syntax:-***select distinct column1,column2,… from table\_name;

**Q:- WAQTD the unique dept no form employee table?**

mysql> select distinct DeptNo from employee;

+--------+

| DeptNo |

+--------+

| 20 |

**|** 10 |

| 30 |

| 40 |

+--------+

4 rows in set (0.05 sec)

Q:-**WAQTD the unique deptno ,designation from employee table?**

mysql> select distinct DeptNo ,designation from employee;

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

| DeptNo | designation |

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

| 20 | tester |

| 10 | HR |

| 30 | clerk |

| 40 | software developer |

| 40 | analyst |

| 20 | salesman |

| 40 | clerk |

| 10 | software developer |

| 20 | HR |

| 20 | analyst |

| 10 | salesman |

| 30 | software developer |

| 30 | HR |

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

13 rows in set (0.00 sec)

**Q:-WAQTD the unique dept no,age,designation form employee table?**

mysql> select distinct DeptNo ,designation,age from employee;

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

| DeptNo | designation | age |

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

| 20 | tester | 22 |

| 10 | HR | 28 |

| 30 | clerk | 28 |

| 40 | software developer | 38 |

| 40 | analyst | 23 |

| 20 | salesman | 28 |

| 40 | clerk | 28 |

| 10 | software developer | 34 |

| 20 | HR | 43 |

| 20 | analyst | 43 |

| 10 | salesman | 33 |

| 30 | clerk | 22 |

| 30 | software developer | 33 |

| 30 | HR | 33 |

* **Operators:-**it is a reserved word or a character used prilimirary in on SQL statement.

Arithmetic, logical, comparision, special operators.

* **WAQTD name salary and salary increment by 5000?**
* **WAQTD name salary and salary decrement by 5000?**
* **WAQTD name salary and salary perday?**
* **WAQTD name salary and annual\_salary ?**
* **Divide 10%3 and change its name?**

**mysql> select 10%3 remainder;**

**+-----------+**

**| remainder |**

**+-----------+**

**| 1 |**

**+-----------+**

**1 row in set (0.00 sec)**

* **WAQTD all the records excluding dept10?**

mysql> select \* from employee

-> where DeptNo!=10;

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- |

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- |

| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- |

| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- |

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

12 rows in set (0.05 sec)

* **WAQTD name designation mobileno age whose sal<10000?**

mysql> select name , mobileNo ,age,designation ,salary from employee where salary<10000;

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

| name | mobileNo | age | designation | salary |

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

| raju | 8785 | 28 | clerk | 6000 |

| sukaniya | 8123 | 28 | clerk | 8000 |

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

2 rows in set (0.00 sec)

* **WAQTD salary <=10000?**

mysql> select name ,salary from employee where salary<=10000;

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

| name | salary |

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

| raju | 6000 |

| sukaniya | 8000 |

| palak | 10000 |

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

3 rows in set (0.00 sec)

* **WAQTD salary >=10000?**

mysql> select name ,salary from employee where salary>=10000;

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

| name | salary |

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

| ranu | 54433 |

| simu | 12000 |

| luqhmaan | 60000 |

| riya |40000 |

| ritu | 30000 |

| rinku | 19000 |

| sakshi | 25000 |

| sanju | 45660 |

| saphu | 35660 |

| anju | 20000 |

| palak | 10000 |

| manik | 40000 |

| manish | 30000 |

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

13 rows in set (0.00 sec)

* **Logical operators:-**

**And ,OR ,NOT!**

* WAQTD all the record who is working in deptno 20 only if salary is >10000?

mysql> select \* from employee where DeptNo = 20 and salary>10000;

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

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

5 rows in set (0.05 sec)

* **WAQTD all the records who is working as salesman and DOB is 1990-10-09?**
* **Waqtd all the records of tables whose salary = 10000 and designation is analyst?**

OR operator

Case1:- if the first condition is true the result of or operator is true it will not cheack for 2nd condition.

Case2:- if the first condition is false it will cheak for 2nd condition .

Case3:- if both the condition is false it will give the output false.

* **Syntax:- select \* from table name where column name =value or column name=value**
* **WAQTD all the records who is working in dept 20 or dept =60?**

**mysql>** select \* from employee where DeptNo=20 or DeptNo=60;

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

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

5 rows in set (0.00 sec)

* **WADTD name salary emailid who is working as a tester or age is greter then 20?**
* **WAQTD the name of all employee who is working as a HR in DEPtNO 20 or who is clerk where salary is greater then 10000?**
* **WAQTD salary>10000 and designation as software developer?**

mysql> select \* from employee where salary >10000 and designation='software developer';

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- |

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

3 rows in set (0.00 sec)

SYNTAX OF NOT : SELECT \* FROM TABLENAME WHERE NOT COLUMN NAME= VALUE

**WAQTD ALL THE EMPLOYEE EXCLUDING DEPTNO 20 OR 30?**

mysql> select \* from employee where not (DeptNo =20 or DeptNo=30);

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ |

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- |

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

6 rows in set (0.00 sec)

mysql> select \* from employee where (DeptNo <>20 and DeptNo<>30);

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ |

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- |

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

6 rows in set (0.00 sec)

ORDER OF EXECTION NOT AND OR

* **WAQTD ALL THE EMPLOYEE DETAILS EXCLUDING SALESMAN WHO ALL ARE WORKING IN DEPT NO 40?**
* select \* from employee where not (designation= 'salesman' and DeptNo=40);

**WAQTD ALL THE EMP DETAILSWHO IS WORKING AS A SALESMAN FOR DEPT 20 OR 30**

select \* from employee where designation = 'salesman' and (DeptNo=20 or DeptNo=30);

* **WAQTD ALL THE EMP WHO IS WORKING IN DEPT20 ONLY IF THERE SALARY IS > 10000 AND < 25000 including 10000 and 25000?**

select \* from employee where DeptNo=20 and (salary>=10000 and salary<=25000);

* WAQTD ALL THE RECORDS WHO IS WORKING AS A SOFTWARE DEVE AND DOB IS 1994?

select \* from employee where designation ='software developer' and (DOB>='1994-01-01' and DOB <='1994-12-31');

* **SPECIAL OPERATORS:-**
* **WAQTD THE EMP WHO IS WORKING IN DEPT10 OR 20 OR 30?**

select \* from employee where (DeptNo=20 or(DeptNo=10 or DeptNo=30));

**or**

select \* from employee where DeptNo in (10,20,30);

* WAQTD ALL THE EMP DETAILS WHOSE JOB DESIGNATION IS HR ANALYT TESTER?

select \* from employee where designation in ('tester','HR','analyst');

* **WAQTD all the emp details who r working as a clerk in dept 20 30 40 60 50?**

select \* from employee where designation = 'clerk' and DeptNo in (20,30,40,50,60);

**( NOT IN OPERATOR IS A MULTI VALUE OPERATOR THAT EXECUTE THE VALUES.)**

**SYNTAX :- NOT IN**

* **WAQTD ALL THE DETAILS OF EMPLOYEE WHO IS WORKING AS CLERK AND NOT IN DEPT 20 30 40 ?**

select \* from employee where designation ='clerk' and DeptNo not in (20,30,40);

* **WADTQ ALL THE DETAILS OF EMP WHO R WORKING IN DEPT 20 40 60 EXCLUDING THE SALESMAN OR ANALYST?**

select \* from employee where designation not in ('salesman','analyst') and DeptNo in(20,40,60);

* **WAQTD all the details whos salry is between 10000 to 25000?**

select \* from employee where salary between 10000 and 25000;

* **WAQTD ALL THE DETAIL OF THE EMP WHO HIRED IN 2017?**

select \* from employee where DOJ between '2017-01-01' and'2017-12-31';

* **WAQTD ALL THE DETAILS OF THE EMP WHO NOT HIRE IN 2017?**

select \* from employee where DOJ not between '2017-01-01' and '2017-12-31';

* **WAQTD ALL THE RECORDS OF EMP EXCLUDING WHO IS HIRED IN 2019?**

select \* from employee where DOJ not between '2019-01-01' and '2019-12-31';

* **WAQTD EMPNAME SALARY AND DEPNO OF ALL THE EMPLOYEE WHO IS WORKING IN DEPTNO 10 20 excluding whos salary >=15000 and <= 50000?**

select name, DeptNo, salary from employee where DeptNo in(10,20) and (salary not between 15000 and 50000);

* IS OPERATOR IS USED TO CHEAK WHEATHER THE VALUE IS NULL .
* IS NOT OPERTOR IS USED TO CHEAK WHEATHER THE VALUE IS NOT NULL.
* WAQTD emp name and annual salary if his designation is null?

select name , salary\*12 anual\_salary from employee where designation is null;

* **WAQTD emp name and annual salary if his designation is not null?**

select name , salary\*12 annual\_salary from employee where designation is not null;

**like operator:-** we use it to perform pattern matching

**syntax:-** select \* from tablename where column name like ‘%/\_’;

characters are of two types:- ordanary:- normal expression ,string.

**special characters:-**when we know only one character of the string.underscore is represent only one character.

And % represent n number of characters.

%,\_

* **WAQTD all the details of emp whose name start whith s?**

select \* from employee where name like's%';

* **WAQTD ALL THE DETAILS OF EMP WHOSE NAME CONTAINS 5CHAR?**

select \* from employee where name like'\_\_\_\_\_';

* **WAQTD WHOSE NAME START WITH S AND END WITH S?**

select \* from employee where name like's%s';

* **WAQTD ALL EMP WHOS 2ND CHAR IS i?**

select \* from employee where name like'\_i%';

* **WAQTD ALL THE INFO OF EMP WHERE DESIGNATION HAVE ATLEAST 2 e?**

select \* from employee where designation like'%e%e%';

* **WAQTD the name ALL THE STRING OF LENGTH 5 THAT STARTS WITH m AND THE 2ND LAST CHAR IS i?**

select \* from employee where name like'm\_\_i\_';

* **WAQTD ALL THE DETAIL OF EMP IF NAME OF EMP START WITH R OR START WHIT Z?**

select \* from employee where name like 'r%' or name like 'z%';

* **WAQTD ALL THE RECORD OF EMP WHOS NAME 3RD CHAR IS k ND 6TH CHAR IS I?**

select \* from employee where name like'\_\_k\_\_i%';

**NOT LIKE OPERATOR:**

* **WAQTD ALL THE RECORD OF EMP except WHOS NAME 3RD CHAR IS k ND 6TH CHAR IS I?**

select \* from employee where name not like'\_\_k\_\_i%';

* **EXCLUDING WHOS NAME START WITH A?**

select \* from employee where name not like'a%';

**FUNCTION :-** function is a block of code which is perform a special task.

**AGGREGATE FUNCTION:**

* **WAQTD AVG SALARY OF ALL THE EMPLOYEE?**

select avg(salary) from employee;

* **WAQTD MAXIMUM SALARY ?**

select max(salary) from employee;

* **WAQTD MIN SAL ?**

select mIN(salary) from employee;

* **WAQTD COUNT ?**

select count(salary) from employee;

* **WAQTD SUM OF SALARY?**

select sum(salary) from employee;

* **WAQTD ALL THE FUNCTION IN ONE ALSO PROVIDE THE ALIAS NAME ?**

select count(salary) count,max(salary) maximun,min(salary) minimun,sum(salary) sum,avg(salary) average from employee;

1. **WAQTD THE NO OF EMP HAD DATE OF JOINING AFTER 2015?**

select count(name) from employee where DOJ not between '0001-01-01' and '2015-12-31';

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

| count(name) |

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

| 10 |

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

1 row in set (0.00 sec)

1. **WAQTD AVG SAL & HiGEST SAL OF DEPT 20 30 40 AND 60?**

select avg(salary),max(salary) from employee where DeptNo in (40,20,30,60);

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

| avg(salary) | max(salary) |

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

| 27562.75 | 54433 |

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

1 row in set (0.00 sec)

1. **WAQTD DO FRST AND LAST HIRED EMPLOYEE?**

select min(DOJ) , max(DOJ) from employee;

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

| min(DOJ) | max(DOJ) |

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

| 2015-04-01 | 2019-09-10 |

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

1 row in set (0.00 sec)

OR

select min(DOJ) first\_hiring, max(DOJ) last\_hiring from employee;

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

| first\_hiring | last\_hiring |

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

| 2015-04-01 | 2019-09-10 |

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

1 row in set (0.00 sec)

**SCALAR FUNCTION**: it is a single row function.

It divided into two parts case manipulation single row func and secnd one is character

**1)WAQTD FOR UPPER?**

select upper(name) from employee;

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

| upper(name) |

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

| RANU |

| SIMU |

| LUQHMAAN |

| RAJU |

| RIYA |

| RITU |

| RINKU |

| SUKANIYA |

| SAKSHI |

| SANJU |

| SAPHU |

| ANJU |

| PALAK |

| MANIK |

| MANISH |

| RAJAN |

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

16 rows in set (0.05 sec)

**2)WAQTD LOWER CASE?**

select lower(name) from employee;

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

| lower(name) |

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

| ranu |

| simu |

| luqhmaan |

| raju |

| riya |

| ritu |

| rinku |

| sukaniya |

| sakshi |

| sanju |

| saphu |

| anju |

| palak |

| manik |

| manish |

| rajan |

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

16 rows in set (0.00 sec)

**CHARACTER MANUPULATION :**

**1)WAQTD ON LENGTH?**

select length(name) from employee;

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

| length(name) |

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

| 4 |

| 4 |

| 8 |

| 4 |

| 4 |

| 4 |

| 5 |

| 8 |

| 6 |

| 5 |

| 5 |

| 4 |

| 5 |

| 5 |

| 6 |

| 5 |

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

16 rows in set (0.03 sec)

**2)WAQ to merge name and salary FOR CONCAT()?**

select concat(name,'' ,salary) from employee;

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

| concat(name,'' ,salary) |

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

| ranu54433 |

| simu12000 |

| luqhmaan60000 |

| raju6000 |

| riya40000 |

| ritu30000 |

| rinku19000 |

| sukaniya8000 |

| sakshi25000 |

**|** sanju45660 |

| saphu35660 |

| anju20000 |

| palak10000 |

| manik40000 |

| manish30000 |

| NULL |

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

16 rows in set (0.00 sec)

**3)WAQ TO REVERSE THE NAME ?**

select reverse(name) from employee;

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

| reverse(name) |

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

| unar |

| umis |

| naamhqul |

| ujar |

| ayir |

| utir |

| uknir |

| ayinakus |

| ihskas |

| ujnas |

| uhpas |

| ujna |

| kalap |

| kinam |

| hsinam |

| najar |

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

16 rows in set (0.00 sec)

**4)WAQTD NAME OF EMP IN THREE CHARACTER?**

select substr(name,1,3) from employee;

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

| substr(name,1,3) |

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

| ran |

| sim |

| luq |

| raj |

| riy |

| rit |

| rin |

| suk |

| sak |

| san |

| sap |

| anj |

| pal |

| man |

| man |

| raj |

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

16 rows in set (0.00 sec)

**5)WAQTD ALL THE DETAIL OF EMP WHOS NAME BEGAIN WITH CONSONANT?**

select \* from employee where substr(name,1,1) not in('a','e','i','o','u');

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ |

| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- |

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- |

| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- |

| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- |

| 16 | rajan | NULL | clerk | 9834 | 32 | NULL | NULL | NULL | NULL | NULL |

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

15 rows in set (0.00 sec)

**3)WAQT SEARCH THE POSITION OF A IN THE NAME COLUMN?**

select instr(name,'a') from employee;

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

| instr(name,'a') |

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

| 2 |

| 0 |

| 6 |

| 2 |

| 4 |

| 0 |

| 0 |

| 4 |

| 2 |

| 2 |

| 2 |

| 1 |

| 2 |

| 2 |

| 2 |

| 2 |

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

16 rows in set (0.00 sec)

**5)WAQT REPLACE WHERE A IS PRESENT IN A NAME WITH Z?**

select replace(name,'a','z') from employee;

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

| replace(name,'a','z') |

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

| rznu |

| simu |

| luqhmzzn |

| rzju |

| riyz |

| ritu |

| rinku |

| sukzniyz |

| szkshi |

| sznju |

| szphu |

| znju |

| pzlzk |

| mznik |

| mznish |

| rzjzn |

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

16 rows in set (0.00 sec)

**5)WAQTD NO OF EMP WHO IS WORKING IN DEPT 20 AND SAL = 15000?**

select count(name) from employee where DeptNo=20 and salary=15000;

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

| count(name) |

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

| 0 |

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

1 row in set (0.00 sec)

**GROUP BY CLAUSE:** it is execute row by row and group the similar type data, and whatever the query execute after this will execute in group. It is not compulsory to have where clause during use of where clause.

It is used with aggregate funcn (count,min,max,sum,avg) to group the result- set by one or more column.

**Syntax:** select columnname from table name

Where condition

Group by columnnames(s);

**1)WAQTD NO OF EMP IN EACH DEPT?**

select count(name) from employee group by DeptNo;

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

| count(name) |

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

| 1 |

| 3 |

| 5 |

| 4 |

| 3 |

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

5 rows in set (0.03 sec)

**2)WAQTD NO OF EMP and designaton FOR EACH DESIGNATION?**

select designation ,count(name) from employee group by designation;

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

| designation | count(name) |

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

| analyst | 2 |

| clerk | 4 |

| HR | 3 |

| salesman | 2 |

| software developer | 3 |

| tester | 2 |

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

6 rows in set (0.00 sec)

**3)WAQTD highest sal of emp in each dept?**

select max(salary) from employee group by DeptNo;

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

| max(salary) |

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

| NULL |

| 60000 |

| 54433 |

| 40000 |

| 40000 |

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

5 rows in set (0.00 sec)

**4)WAQTD NO OF EMP IN EACH DEPT EXCEPT DESIGNATION HR?**

select count(name) from employee where designation not in ('HR') group by DeptNo;

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

| count(name) |

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

| 1 |

| 2 |

| 4 |

| 3 |

| 3 |

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

5 rows in set (0.00 sec)

**5)WAQTD NO OF EMP HIRED ON EACH DATE?**

select DOJ, count(\*) from employee group by DOJ;

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

| DOJ | count(\*) |

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

| NULL | 1 |

| 2015-04-01 | 1 |

| 2015-04-10 | 2 |

| 2015-06-12 | 1 |

| 2015-11-21 | 1 |

| 2016-01-12 | 2 |

| 2016-09-01 | 1 |

| 2017-01-12 | 1 |

| 2017-09-01 | 1 |

| 2018-01-12 | 1 |

| 2018-06-28 | 1 |

| 2019-07-03 | 1 |

| 2019-09-06 | 1 |

| 2019-09-10 | 1 |

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

**6)WAQTD NO OF SOFTWARE DEVE IN EACH DEPT?**

select count(\*) from employee where designation = 'software developer' group by DeptNo;

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

| count(\*) |

**+----------+**

| 1 |

| 1 |

| 1 |

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

3 rows in set (0.00 sec)

**7)WAQTD MIN SAL IN EACH DESIGNATION ?**

select min(salary) from employee group by designation;

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

| min(salary) |

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

| 30000 |

| 6000 |

| 30000 |

| 19000 |

| 25000 |

| 12000 |

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

6 rows in set (0.00 sec)

**HAVING CLAUSE**: It is used for filter group condition it will execute after group by clause .

In having clause we can only use an column i.e used in by clause.

**1)WAQTD DEPT NO WHICH HAS ATLEAST 4 WORKING EMPLOYEES?**

select DeptNo from employee group by DeptNo having count(name) >4;

+--------+

| DeptNo |

+--------+

| 20 |

+--------+

1 row in set (0.00 sec)

**2)WAQTD DEPTNO WHICH HAS TOTAL SAL > 50000?**

select DeptNo from employee group by DeptNo having sum(salary)> 50000;

+--------+

| DeptNo |

+--------+

| 10 |

| 20 |

| 30 |

| 40 |

+--------+

4 rows in set (0.00 sec)

**3)WAQTD DEPTNO AND SALARY OF ALL THE DEPT WHOS AVG SALARY IS > 20000 EXCLUDING THE EMPLOYEE WHOSE NAME BEGAINS WITH S?**

select Deptno,salary from employee where name not like 's%' group by DeptNo having avg(salary)>20000;

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

| Deptno | salary |

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

| 10 | 60000 |

| 20 | 54433 |

| 30 | 6000 |

| 40 | 40000 |

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

4 rows in set (0.00 sec)

**ORDER BY** CLAUSE IS USED TO SHORT THE RESULT IN ASSECNDING OG DECENDING ORDER AND BYDEFAULT IT IS ASC.

=>ASC/DESC.

=>It always executes after execution of select clause .

=>It should always last clause to be written in the syntax.

=>If the group by clause is used in order by clause we can use aggregate func or multi row func or only the column name which is used in the group by clause.

**1)WAQTD ALL THE RESULT SALARY SHOULD BE IN DESC ORDER?**

select \* from employee order by salary desc;

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

| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup |

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

| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ |

| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ |

| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- |

| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- |

| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- |

| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ |

| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ |

| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- |

| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ |

| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- |

| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- |

| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ |

| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- |

| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ |

| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- |

| 16 | rajan | NULL | clerk | 9834 | 32 | NULL | NULL | NULL | NULL | NULL |

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

16 rows in set (0.05 sec)

**2)WAQTD EMP NAME DESIGNATION DOJ FOR ALL EMP WHO R WORKING IN EITHER ONE OF THE DEPTS 10 20 50 60 SHORT THE RECORDS IN DESC ORDER OF THEIR SAALRY?**

select name,designation,DOJ from employee where DeptNo in(10,20,50,60) order by salary desc;

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

| name | designation | DOJ |

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

| luqhmaan | HR | 2018-01-12 |

| ranu | tester | 2019-09-06 |

| sanju | HR | 2016-09-01 |

| saphu | analyst | 2015-04-01 |

| sakshi | software developer | 2015-11-21 |

| anju | salesman | 2017-09-01 |

| rinku | salesman | 2015-06-12 |

| simu | tester | 2019-07-03 |

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

**3)WAQTD DEPTNO ALONG WITH NO OF EMP IN IT?**

select DeptNo,count(name) from employee group by DeptNo;

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

| DeptNo | count(name) |

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

| NULL | 1 |

| 10 | 3 |

| 20 | 5 |

| 30 | 4 |

| 40 | 3 |

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

5 rows in set (0.00 sec)

**4)WAQTD ALL THE DETAILS OF WHOS DESIGNATION IS SD AND TESTER?**

**SUB QUERY:** QUERY IINSIDE A QUERY IT IS A ANOTHER DATA TO FETCH THE DATA FROM MULTIPLE TABLES.

INNER QUERY MUST BE CLOSED WITH ().

ALWAYS INNER QUERY EXICUTED FIRST AND O/P OF INNER QUERY IS I/P OF OUTER QUERY AND FINAL RESULT WE GET FROM OUTER QUERY.

=>**WHEN TO USE SUBQUERY IN WHERE CLAUSE:**

* If the given condition contain unknow value we use subquery to obtain the unknow value.
* If the column to be selected and the condition to be filter as a given to different table we can use sub query.
* There are two types of subquery

1)single row subquery: which return single value.

2)and we use operators (=,>=,<=,<>,!=,)

* **Syntax:-select \* from table name where columnname operator (select \* from table name where condition);**
* **MULTI ROW SUBQUERY:- in this operator (in) are used.**

**1)it will return more then one record.**

**1)WAQTD NAME OF THE EMP WHO IS WORKING FOR RESEARCH DEPT?**

**select name from employee where DeptNo=(select deptno from department where dname='research');**

**+-------+**

**| name |**

**+-------+**

**| ranu |**

**| simu |**

**| rinku |**

**| sanju |**

**| saphu |**

**+-------+**

**5 rows in set (0.00 sec)**

**2)WAQTD THE NAME OF EMP WHOS WORKING LOCATION WHICH HAS ATLEAST TWO A CHAR IN ITS NAME?**

**select location from department where deptno in (select DeptNo from employee where name like '%a%a%');**

**+----------+**

**| location |**

**+----------+**

**| banglore |**

**| mumbai |**

**| delhi |**

**+----------+**

**3 rows in set (0.04 sec)**

**3)WAQT OBTAIN 2ND MAX SALARY FROM THE EMP TABLE?**

**select max(salary)from employee where salary<(select max(salary) from employee);**

**+-------------+**

**| max(salary) |**

**+-------------+**

**| 54433 |**

**+-------------+**

1. **row in set (0.00 sec**

**4)WAQT OBTAIN 4th MAX SALARY FROM THE EMP TABLE?**

**select max(salary)from employee where salary<(select max(salary)from employee where salary<(select max(salary)from employee where salary<(select max(salary) from employee)));**

**+-------------+**

**+-------------+**

**| 40000 |**

**+-------------+**

**5) WAQT OBTAIN 3rd MAX SALARY FROM THE EMP TABLE?**

**select max(salary)from employee where salary<(select max(salary)from employee where salary<(select max(salary) from employee));**

**+-------------+**

**| max(salary) |**

**+-------------+**

**| 45660 |**

**+-------------+**

**JOINTS:- it is used to fetch the data from two or more tables.**

* **Merging of more then one table horizontally is know joints.**

**Types of joints:-**

* **Cross join/cartision join:- a record from one table is merged with each and every record of the other table or joint two tables a record from left table will be merge with each and every record iin the right table such join is known as cartision join.**

**Syntax :**

**Select \* from tablename1 cross join tablename2**

**1)WAQTD ALL THE RECORDS USING CROSS JOIN?**

**Select \* from employee cross join department.**

* **Inner join: joining two tables with the help of joint is known as inner join.**

**This is give us a matched records only from both the tables.**

**SYNTAX{mysql}:- select \* from table1 inner join table2 on table1.colmunname = table2.columnname;**

**mysql> select \* from employee inner join department on employee.Deptno = department.deptno;**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup | deptno | dname | location |**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ | 20 | research | pune |**

**| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ | 20 | research | pune |**

**| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ | 10 | accounting | banglore |**

**| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- | 30 | sales | delhi |**

**| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- | 40 | admin | mumbai |**

**| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ | 40 | admin | mumbai |**

**| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- | 20 | research | pune |**

**| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ | 40 | admin | mumbai |**

**| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ | 10 | accounting | banglore |**

**| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- | 20 | research | pune |**

**| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ | 20 | research | pune |**

**| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- | 10 | accounting | banglore |**

**| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- | 30 | sales | delhi |**

**| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- | 30 | sales | delhi |**

**| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- | 30 | sales | delhi |**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**SYNTAX(ORACLE):- select \* from table1,table2 where condition**

* **Outer join :**

**Left join:- it gives inner join plus unmatched records of left table.**

**(Unmatched records) means the record do who do not have pair join the opposite table w.r.t join.**

**SYNTAX(my sql):- select \* from table1 left outer join table2 on condition;**

**SYNTAX(oracle):- select \* from table1 , table2 where table1.colname = table2.colname(+);**

**select \* from employee left outer join department on employee.Deptno = department.deptno;**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup | deptno | dname | location |**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ | 10 | accounting | banglore |**

**| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ | 10 | accounting | banglore |**

**| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- | 10 | accounting | banglore |**

**| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ | 20 | research | pune |**

**| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ | 20 | research | pune |**

**| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- | 20 | research | pune |**

**| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- | 20 | research | pune |**

**| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ | 20 | research | pune |**

**| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- | 30 | sales | delhi |**

**| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- | 30 | sales | delhi |**

**| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- | 30 | sales | delhi |**

**| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- | 30 | sales | delhi |**

**| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- | 40 | admin | mumbai |**

**| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ | 40 | admin | mumbai |**

**| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ | 40 | admin | mumbai |**

**| 16 | rajan | NULL | clerk | 9834 | 32 | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |**

**+----+----------+--------------+--------------------+----------+-----+------------+------------+--------+--------+------------+--------+------------+----------+**

**16 rows in set (0.00 sec)**

**Right join:- it gives inner join plus unmatched records of right table.**

**(Unmatched records) means the record do who do not have pair join the opposite table w.r.t join.**

**SYNTAX(my sql):- select \* from table1 right outer join table2 on condition;**

**SYNTAX(oracle):- select \* from table1 , table2 where table1.colname(+) = table2.colname;**

**select \* from employee right outer join department on employee.Deptno = department.deptno;**

**+------+----------+--------------+--------------------+----------+------+------------+------------+--------+--------+------------+--------+------------+----------+**

**| id | name | emailId | designation | mobileNo | age | DOB | DOJ | Salary | DeptNo | BloodGroup | deptno | dname | location |**

**+------+----------+--------------+--------------------+----------+------+------------+------------+--------+--------+------------+--------+------------+----------+**

**| 1 | ranu | ranu@gmail | tester | 9876 | 22 | 1997-07-02 | 2019-09-06 | 54433 | 20 | A+ | 20 | research | pune |**

**| 2 | simu | simu@gmail | tester | 8769 | 22 | 1997-07-02 | 2019-07-03 | 12000 | 20 | A+ | 20 | research | pune |**

**| 3 | luqhmaan | luqh@gmail | HR | 8855 | 28 | 1990-09-03 | 2018-01-12 | 60000 | 10 | B+ | 10 | accounting | banglore |**

**| 4 | raju | raju@gmail | clerk | 8785 | 28 | 1990-09-03 | 2016-01-12 | 6000 | 30 | B- | 30 | sales | delhi |**

**| 5 | riya | riya@gmail | software developer | 9823 | 38 | 1980-09-03 | 2016-01-12 | 40000 | 40 | AB- | 40 | admin | mumbai |**

**| 6 | ritu | ritu@gmail | analyst | 8021 | 23 | 1996-09-03 | 2018-06-28 | 30000 | 40 | O+ | 40 | admin | mumbai |**

**| 7 | rinku | riku@gmail | salesman | 9456 | 28 | 1990-10-09 | 2015-06-12 | 19000 | 20 | B- | 20 | research | pune |**

**| 8 | sukaniya | suku@gmail | clerk | 8123 | 28 | 1990-09-03 | 2017-01-12 | 8000 | 40 | AB+ | 40 | admin | mumbai |**

**| 9 | sakshi | saku@gmail | software developer | 9135 | 34 | 1985-03-02 | 2015-11-21 | 25000 | 10 | B+ | 10 | accounting | banglore |**

**| 10 | sanju | sanju@gmail | HR | 89675 | 43 | 1977-05-28 | 2016-09-01 | 45660 | 20 | O- | 20 | research | pune |**

**| 11 | saphu | saphu@gmail | analyst | 89235 | 43 | 1977-05-28 | 2015-04-01 | 35660 | 20 | O+ | 20 | research | pune |**

**| 12 | anju | anju@gmail | salesman | 99675 | 33 | 1987-05-28 | 2017-09-01 | 20000 | 10 | O- | 10 | accounting | banglore |**

**| 13 | palak | palu@gmail | clerk | 92345 | 22 | 1997-05-28 | 2019-09-10 | 10000 | 30 | B- | 30 | sales | delhi |**

**| 14 | manik | manu@gmail | software developer | 94178 | 33 | 1987-05-28 | 2015-04-10 | 40000 | 30 | AB- | 30 | sales | delhi |**

**| 15 | manish | manish@gmail | HR | 90976 | 33 | 1987-05-28 | 2015-04-10 | 30000 | 30 | AB- | 30 | sales | delhi |**

**+------+----------+--------------+--------------------+----------+------+------------+------------+--------+--------+------------+--------+------------+----------+**

**15 rows in set (0.00 sec)**

**Full outer join:- the full outer join gives inner join as well as all the records from both the table who do not have any pair.**

**NOTE: there is no syntax for this in oracle.**

**Then we go for union keyword.**

* **Self join**
* **Natural join**
* **Key in RDBMS**
* **Key attributes: a key attribute is used to find an attribute using which we can uniquely determine a record in a table .**
* **Non key attributes: all the attributes accept key attributes is known as non key attributes.**
* **Prime key attributes: a key attributes which is chosen to be a main attributes to determine the records of uniquely in the table.**
* **Non primary key attributes: a key attributes which is chosen to be an attributes to determine the records of non-uniquely in the table.**
* **Composite key: the combination of two and more attribute which determine the record uniquely in table.**

**Or**

**it is key which is eligible to be a primary key in a table.**

* **Foreign key: it is used to set the relation between two or more tables.**
* **FUNCTIONAL DEPENDANCY(FD):- in a FD a relation exist such that an attributes determine another attribute uniquely.**

**Ex:- R ->{x , y}**

**In this example x and y is an attribute in which x is a primary key so for getting the records we take the help of PK since *x is a determinant and y is a dependent***

***x->y***

* **TYPES OF FD: -**
* **TOTAL FD : if all the attributes of a relation is determine by a key attribute is known as totally FD.**

**R ->{A,B,C,D}**

**A->B**

**A->C**

**A->D**

**In which A is determinant and B is dependent.**

* **PARTIAL FD : a relation is said to have a partial FD if a) it consists of composite key attributes. B) their exist a dependency such as that attribute determine by another attribute which is the part of composite key.**

**Ex:- R ->{A,B,C,D}**

**(AB)->C**

**(AB)->D**

* **TRANSATIVE FD: relation is said to be transitive FD if their exist an relation such that am attribute is determine by non key attributes which in terms determine by key attribute.**

**R->{A,B,C,D}**

**A->C**

**C->D**

**A->D**

**NORMALIZATION :- the process of decomposing the table into a small table in order to remove redundancy and anamoly by identifying dependency.**

**The process of reducing**

**WE USE THE NORMALIZATION TO OVERCOME FOLLOWING PROBLEMS.**

* **DATA REPETATION**
* **ANAMOLY : the side effect is occur while performing the DML is known as anamoly.**
* **Normal form : a state of table without table redundancy is known as normal form**

**TYPES :-**

1. **First normal form :- a table is said to be in the first NF if they satisfy the following condition 1) tables should not have duplicate rows 2) every in the table should be single (atomic)value.**
2. **Second normal form :- if they follow the conditions 1) the table should be in first NF 2) the table should not have partial functional dependency (should not have composite key).**

**NOTE:-if the table consists partial FD the attribute which are responsible are remove from the table.**

1. **Third normal form :- the table has said to be third NF if the following condition are satisfy .**

**1)it should follow the second NF .**

**2) the table should not have transitive FD.**

**NOTE:- second and the third NF these are based on keys attributes and FD of relational schema .**

1. **BCNF normal form**
2. **Fourth normal form**

**ER DIAGRAM:- ENTITY RELATIONAL DIAGRAM describe the structure of database with the help of diagram which is known as ER diagram. It is a blue print of database that can be later implement as a database.**

**COMPONENT OF ER DIAGRAM**

1. **ENTITY**
2. **ATTRIBUTE**
3. **RELATIONSHIP**

**Rectangle represents the entity**

**elipse It represents the attribute.**

**diamond It represent the relationship.**

**single line it represents the connection b/w entity and attribute.**

1. **One to one.(in notes )**
2. **Many to many.**
3. **One to many.**

**VIEW :- it is a logical or virtual tables that can be created on the existing table.**

**View do not occupy memory.**

**SYNTAX:- create view view\_name**

**as**

**Select \* from table\_name;**

**TO CALL THE VIEW**

**Select \* from view;**

**Any DML operation perform on view a base table will got reflected.**

**TCL:-**

**All the command in mysql is aut6o commit.**

**So first we have to make it false .**

* **set autocommit = 0;**

**for rollback simply right**

**rollback;**

**after inserting the result.**

**Now for making the commit true simply right**

**Commit;**

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**+--------+------------+----------+**

**4 rows in set (0.00 sec)**

**mysql> insert into department values(55,'operation','agra');**

**Query OK, 1 row affected (0.06 sec)**

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**5 rows in set (0.00 sec)**

**mysql> rollback;**

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

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**5 rows in set (0.00 sec)**

**mysql> set autocommit = 0;**

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

**mysql> insert into department values(50,'operation','patna');**

**Query OK, 1 row affected (0.03 sec)**

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 50 | operation | patna |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**6 rows in set (0.00 sec)**

**mysql> rollback;**

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

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**5 rows in set (0.00 sec)**

**mysql> insert into department values(50,'operation','patna');**

**Query OK, 1 row affected (0.00 sec)**

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 50 | operation | patna |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**6 rows in set (0.00 sec)**

**mysql> commit;**

**Query OK, 0 rows affected (0.05 sec)**

**mysql> rollback;**

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

**mysql> select \* from department;**

**+--------+------------+----------+**

**| deptno | dname | location |**

**+--------+------------+----------+**

**| 10 | accounting | banglore |**

**| 20 | research | pune |**

**| 30 | sales | delhi |**

**| 40 | admin | mumbai |**

**| 50 | operation | patna |**

**| 55 | operation | agra |**

**+--------+------------+----------+**

**6 rows in set (0.00 sec)**

**SAVEPOINT:- it usually work with rollback.**

**For creating savepoint:- savepoint savepoint\_name;**

**Now update or insert the details**

**you can create more then 1 save point.**

**For rollback write**

**Rollback to savepoint\_name;**

**STORED PROCEDURE:-**

* **a store procedure is a collection or group of SQL statements stored in the databases data dictionary and called from either a remote program another stored procedure or the command line.**

**SYNTAX:-**

**Delimiter $$**

**Create procedure <procedure name>**

**As**

**Begin**

**<SQL statement>;**

**End $$**

**Delimiter;**

**For call the store procedure**

**SYNTAX:- call storeprocedure\_name;**