<u>Prerequisite:</u>

Joins:

The SQL Joins clause is used to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

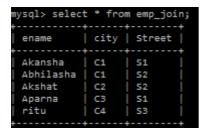
Types of Join:-

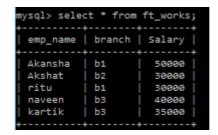
1. Cross Join :-

The CARTESIAN JOIN or CROSS JOIN returns the Cartesian product of the sets of records from the two or more joined tables. Thus, it equates to an inner join where the join-condition always evaluates to True or where the join-condition is absent from the statement.

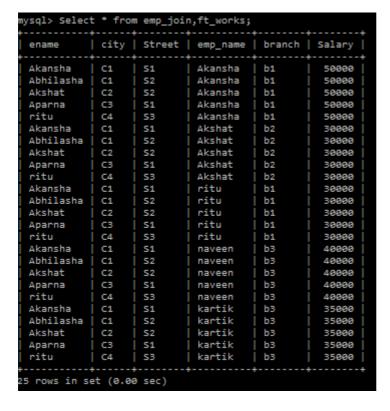
Example:

Consider the following two tables





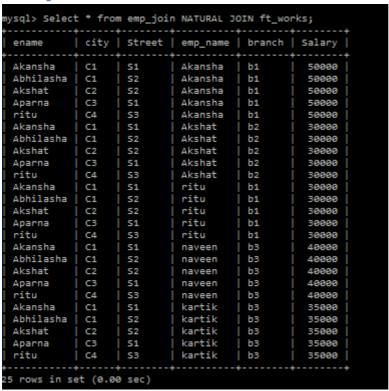
Now, let us join these two tables using CROSS JOIN



2. Natural Join:-

A **NATURAL JOIN** is a **JOIN** operation that creates an implicit **join** clause for you based on the common columns in the two tables being joined. Common columns are columns that have the same name in both tables.

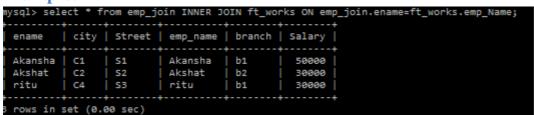
Example:



3. Inner Join:-

The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns in both tables.

Example:



4. Outer Join:-

A - Left Outer Join-

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

Example:

```
ysql> Select * from emp_join LEFT JOIN ft_works ON emp_join.ename=ft_works.emp_name;
           | city | Street | emp_name | branch | Salary
Akansha
                    51
                             Akansha
                                         b1
Abhilasha
                    52
                                         NULL
                                                    NULL
                             NULL
                    52
                             Akshat
Akshat
                                         b2
                                                   30000
Aparna
             C3
                             NULL
                                         NULL
                             ritu
                                                   30000
                                         b1
 rows in set (0.00 sec)
```

B - Right Outer Join-

he RIGHT JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

Example:

```
nysql> Select * from emp_join RIGHT OUTER JOIN ft_works ON emp_join.ename=ft_works.emp_name;
         | city | Street | emp_name | branch | Salary
 Akansha
           C1
                  51
                            Akansha
                                       b1
                                                  50000
 Akshat
                            Akshat
                                       b2
                                                  30000
           C4
                                                  30000
 ritu
                  S3
                            ritu
                                       b1
 NULL
           NULL
                  NULL
                                       ЬЗ
                                                  40000
                            naveen
                            kartik
           NULL
                  NULL
                                       bЗ
                                                  35000
 NULL
 rows in set (0.00 sec)
```

C - Full Outer Join-

The FULL OUTER JOIN keyword returns all rows from the left table (table1) and from the right table (table2).

The FULL OUTER JOIN keyword combines the result of both LEFT and RIGHT joins.

Example:

```
ysql> Select * from emp_join LEFT OUTER JOIN ft_works ON emp_join.ename=ft_works.emp_name UNION
      select * from emp_join RIGHT OUTER JOIN ft_works ON emp_join.ename=ft_works.emp_name;
            city | Street | emp_name | branch | Salary
Akansha
                    51
                             Akansha
                                                   50000
                                         b1
 Abhilasha
                    52
                             NULL
                                         NULL
                                                    NULL
 Akshat
                              Akshat
                                         b2
                                                    30000
 Aparna
             С3
                    51
                              NULL
                                         NULL
                                                    NULL
             C4
 ritu
                    53
                              ritu
                                         b1
                                                   30000
 NULL
             NULL
                    NULL
                              naveen
                                         ЬЗ
                                                   40000
             NULL
                    NULL
                              kartik
                                         bЗ
                                                   35000
 rows in set (0.00 sec)
```

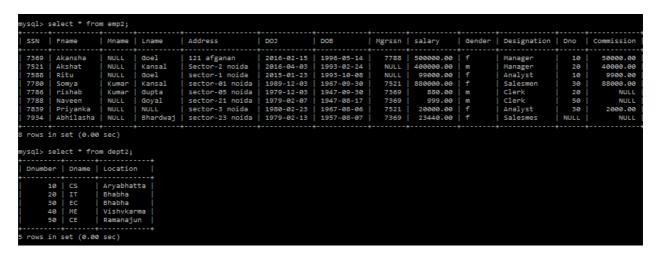
Prerequisite:

A Subquery or Inner query or Nested query is a query within another SQL query and embedded within the WHERE clause. A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN etc.

There are a few rules that subqueries must follow:

- Subqueries must be enclosed within parentheses.
- A subquery can have only one column in the SELECT clause, unless multiple columns are in the main query for the subquery to compare its selected columns.
- An ORDER BY cannot be used in a subquery, although the main query can use an ORDER BY. The GROUP BY can be used to perform the same function as the ORDER BY in a subquery.
- Subqueries that return more than one row can only be used with multiple value operators, such as the IN operator.
- A subquery cannot be immediately enclosed in a set function.
- The BETWEEN operator cannot be used with a subquery; however, the BETWEEN operator can be used within the subquery.

Consider the following two tables



Q- Display name of department where there is atleast one employee.

```
ysql> select Dname from dept2 where Dnumber in (select distinct dno from emp);
Dname
CS
     in set (0.00 sec)
```

Q- Display the fname, Iname, dno, salary of all employees who have same salary & the designation of all employees who work for manager 7788;

```
nysql> select fname,lname,dno,salary from emp2 where (salary,designation) IN (select salary,designation from emp where mgrssn=7788);
      | lname | dno | salary
Akansha | Goel | 10 | 500000.00 |
row in set (0.00 sec)
```

Q- Display the department name of the emp whose SSN=7788.

```
ysql> select dname from dept2 where dnumber in (select dno from emp2 where ssn=7788);
dname
row in set (0.00 sec)
```

Q- Display Iname, fname, Salary of all employee who work in same department as the employee whose last name is 'Gupta'

```
nysql> select lname,fname,salary from emp2 where Dno IN (select distinct Dno from emp where lname='Gupta');
       | fname | salary
 Kansal
        Akshat | 400000.00
                     880.00
        rishab
 rows in set (0.00 sec)
```

Q- Display the Fname, Lname & Designation of any employee employed by an existing department.

```
select fname, lname, designation from emp2 where Dno=ANY (select dnumber from dept2);
         | Iname | designation
Akansha
                  Manager
         Goel
Akshat
           Kansal
                    Manager
Ritu
           Goel
                    Analyst
           Kansal
                    Salesmen
Somya
rishab
           Gupta
                    Clerk
           Goyal
                    Clerk
           NULL
                    Analyst
rows in set (0.00 sec)
```

Q- Find the employee with lowest salary.

```
fname,lname from emp2 where salary <=ALL (select salary from emp2);
rishab | Gupta |
row in set (0.15 sec)
```

Q- Display the name of all department with no employee.

```
nysql> select dname from dept2 where Dnumber NOT IN( select dno from emp2);
Empty set (0.00 sec)
```

Prerequisite:

An operator is a reserved word or a character used primarily in an SQL statement's WHERE clause to perform operation(s), such as comparisons and arithmetic operations.

Various types of logical operator:-

Operator	Description
ALL	The ALL operator is used to compare a value to all values in another value set.
AND	The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
ANY	The ANY operator is used to compare a value to any applicable value in the list according to the condition.
BETWEEN	The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value.
EXISTS	The EXISTS operator is used to search for the presence of a row in a specified table that meets certain criteria.
IN	The IN operator is used to compare a value to a list of literal values that have been specified.
LIKE	The LIKE operator is used to compare a value to similar values using wildcard operators.
NOT	The NOT operator reverses the meaning of the logical operator with which it is used. Eg: NOT EXISTS, NOT BETWEEN, NOT IN, etc. This is a negate operator.
OR	The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.
IS NULL	The NULL operator is used to compare a value with a NULL value.
UNIQUE	The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates).