

Prerequisite :

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

```
mysql> select * from emp_join;
```

ename	city	Street
Akansha	C1	S1
Abhilasha	C1	S2
Akshat	C2	S2
Aparna	C3	S1
ritu	C4	S3

```
mysql> select * from ft_works;
```

emp_name	branch	Salary
Akansha	b1	50000
Akshat	b2	30000
ritu	b1	30000
naveen	b3	40000
kartik	b3	35000

Now, let us join these two tables using CROSS JOIN

```
mysql> Select * from emp_join,ft_works;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Abhilasha	C1	S2	Akansha	b1	50000
Akshat	C2	S2	Akansha	b1	50000
Aparna	C3	S1	Akansha	b1	50000
ritu	C4	S3	Akansha	b1	50000
Akansha	C1	S1	Akshat	b2	30000
Abhilasha	C1	S2	Akshat	b2	30000
Akshat	C2	S2	Akshat	b2	30000
Aparna	C3	S1	Akshat	b2	30000
ritu	C4	S3	Akshat	b2	30000
Akansha	C1	S1	ritu	b1	30000
Abhilasha	C1	S2	ritu	b1	30000
Akshat	C2	S2	ritu	b1	30000
Aparna	C3	S1	ritu	b1	30000
ritu	C4	S3	ritu	b1	30000
Akansha	C1	S1	naveen	b3	40000
Abhilasha	C1	S2	naveen	b3	40000
Akshat	C2	S2	naveen	b3	40000
Aparna	C3	S1	naveen	b3	40000
ritu	C4	S3	naveen	b3	40000
Akansha	C1	S1	kartik	b3	35000
Abhilasha	C1	S2	kartik	b3	35000
Akshat	C2	S2	kartik	b3	35000
Aparna	C3	S1	kartik	b3	35000
ritu	C4	S3	kartik	b3	35000

25 rows in set (0.00 sec)

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:

```
mysql> Select * from emp_join NATURAL JOIN ft_works;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Abhilasha	C1	S2	Akansha	b1	50000
Akshat	C2	S2	Akansha	b1	50000
Aparna	C3	S1	Akansha	b1	50000
ritu	C4	S3	Akansha	b1	50000
Akansha	C1	S1	Akshat	b2	30000
Abhilasha	C1	S2	Akshat	b2	30000
Akshat	C2	S2	Akshat	b2	30000
Aparna	C3	S1	Akshat	b2	30000
ritu	C4	S3	Akshat	b2	30000
Akansha	C1	S1	ritu	b1	30000
Abhilasha	C1	S2	ritu	b1	30000
Akshat	C2	S2	ritu	b1	30000
Aparna	C3	S1	ritu	b1	30000
ritu	C4	S3	ritu	b1	30000
Akansha	C1	S1	naveen	b3	40000
Abhilasha	C1	S2	naveen	b3	40000
Akshat	C2	S2	naveen	b3	40000
Aparna	C3	S1	naveen	b3	40000
ritu	C4	S3	naveen	b3	40000
Akansha	C1	S1	kartik	b3	35000
Abhilasha	C1	S2	kartik	b3	35000
Akshat	C2	S2	kartik	b3	35000
Aparna	C3	S1	kartik	b3	35000
ritu	C4	S3	kartik	b3	35000

25 rows in set (0.00 sec)

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:

```
mysql> select * from emp_join INNER JOIN ft_works ON emp_join.ename=ft_works.emp_Name;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Akshat	C2	S2	Akshat	b2	30000
ritu	C4	S3	ritu	b1	30000

3 rows in set (0.00 sec)

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:

```
mysql> Select * from emp_join LEFT JOIN ft_works ON emp_join.ename=ft_works.emp_name;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Abhilasha	C1	S2	NULL	NULL	NULL
Akshat	C2	S2	Akshat	b2	30000
Aparna	C3	S1	NULL	NULL	NULL
ritu	C4	S3	ritu	b1	30000

5 rows in set (0.00 sec)

B - Right Outer Join-

The 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:

```
mysql> Select * from emp_join RIGHT OUTER JOIN ft_works ON emp_join.ename=ft_works.emp_name;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Akshat	C2	S2	Akshat	b2	30000
ritu	C4	S3	ritu	b1	30000
NULL	NULL	NULL	naveen	b3	40000
NULL	NULL	NULL	kartik	b3	35000

5 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:

```
mysql> 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;
```

ename	city	Street	emp_name	branch	Salary
Akansha	C1	S1	Akansha	b1	50000
Abhilasha	C1	S2	NULL	NULL	NULL
Akshat	C2	S2	Akshat	b2	30000
Aparna	C3	S1	NULL	NULL	NULL
ritu	C4	S3	ritu	b1	30000
NULL	NULL	NULL	naveen	b3	40000
NULL	NULL	NULL	kartik	b3	35000

7 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

```
mysql> select * from emp2;
```

SSN	Fname	Mname	Lname	Address	DOJ	DOB	Mgrssn	salary	Gender	Designation	Dno	Commission
7369	Akansha	NULL	Goel	121 afganan	2016-02-15	1996-05-14	7788	500000.00	f	Manager	10	50000.00
7521	Akshat	NULL	Kansal	Sector-2 noida	2016-04-03	1993-02-24	NULL	400000.00	m	Manager	20	40000.00
7588	Ritu	NULL	Goel	sector-1 noida	2015-01-23	1993-10-08	NULL	90000.00	f	Analyst	10	9900.00
7780	Somya	Kumar	Kansal	sector-01 noida	1989-12-03	1967-09-30	7521	880000.00	f	Salesmen	30	88000.00
7786	rishab	Kumar	Gupta	sector-05 noida	1979-12-03	1947-09-30	7369	880.00	m	Clerk	20	NULL
7788	Naveen	NULL	Goyal	sector-21 noida	1979-02-07	1947-08-17	7369	999.00	m	Clerk	50	NULL
7839	Priyanka	NULL	NULL	sector-3 noida	1980-02-23	1967-08-06	7521	20000.00	f	Analyst	30	2000.00
7934	Abhilasha	NULL	Bhardwaj	sector-23 noida	1979-02-13	1957-08-07	7369	23440.00	f	Salesmes	NULL	NULL

```
8 rows in set (0.00 sec)
```

```
mysql> select * from dept2;
```

Dnumber	Dname	Location
10	CS	Aryabhatta
20	IT	Bhabha
30	EC	Bhabha
40	ME	Vishvkarma
50	CE	Ramanajun

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

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

```
mysql> select Dname from dept2 where Dnumber in (select distinct dno from emp);
+-----+
| Dname |
+-----+
| CS    |
| IT    |
| EC    |
| CE    |
+-----+
1 rows in set (0.00 sec)
```

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

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

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

```
mysql> select dname from dept2 where dnumber in (select dno from emp2 where ssn=7788);
+-----+
| dname |
+-----+
| CE    |
+-----+
1 row in set (0.00 sec)
```

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

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

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

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

Q- Find the employee with lowest salary.

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

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

```
mysql> 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).