# **Introduction to Joins in MySQL:**

Joins are a powerful feature of SQL that allows you to combine data from two or more tables in a database. With joins, you can retrieve data from related tables and combine it into a single result set. SQL supports different types of joins, each of which is used for specific scenarios.

Types of Joins in MySQL:

Inner join

Left join

Right join

Self join

Natural join

## **Inner join:**

The inner join is the most commonly used type of join in SQL. It returns only the matching rows between two tables based on the join condition. The syntax for an inner join is as follows:

SELECT column1, column2, ...

FROM table1

INNER JOIN table2

ON table1.column\_name = table2.column\_name;

In this syntax, the INNER JOIN keyword is used to join two tables based on the join condition specified by the ON clause. The result set will only contain the rows where the join condition is true for both tables.

## 

## **Left join:**

A left join returns all the rows from the left table and the matching rows from the right table. If there is no matching row in the right table, the result set will contain null values. The syntax for a left join is as follows:

SELECT column1, column2, ...

FROM table1

LEFT JOIN table2

ON table1.column\_name = table2.column\_name;

In this syntax, the LEFT JOIN keyword is used to join two tables based on the join condition specified by the ON clause. The result set will contain all the rows from the left table and the matching rows from the right table. If there is no matching row in the right table, the result set will contain null values.

## Right join:

A right join returns all the rows from the right table and the matching rows from the left table. If there is no matching row in the left table, the result set will contain null values. The syntax for a right join is as follows:

SELECT column1, column2, ...

FROM table1

RIGHT JOIN table2

ON table1.column\_name = table2.column\_name;

In this syntax, the RIGHT JOIN keyword is used to join two tables based on the join condition specified by the ON clause. The result set will contain all the rows from the right table and the matching rows from the left table. If there is no matching row in the left table, the result set will contain null values.

# 

# Examples of Joins

Joins are used to combine two or more tables to get the desired outputs. Joins can be broadly divided into two types. Inner and Outer

Suppose we have two tables:

Table 1: members

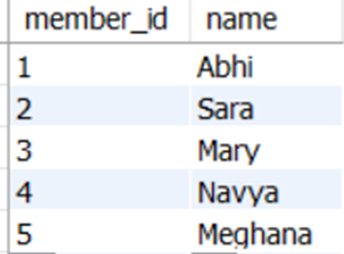
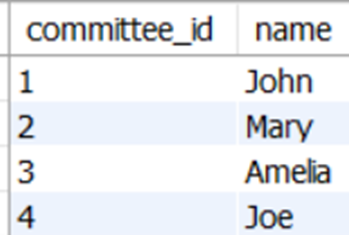
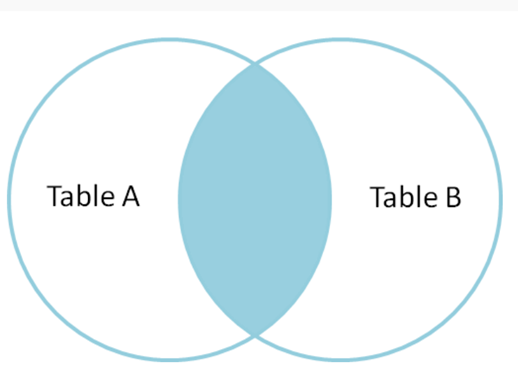


Table 2: committees



## Inner joins:

Combines the matching rows based on the condition. The output will be the intersection of the two tables.

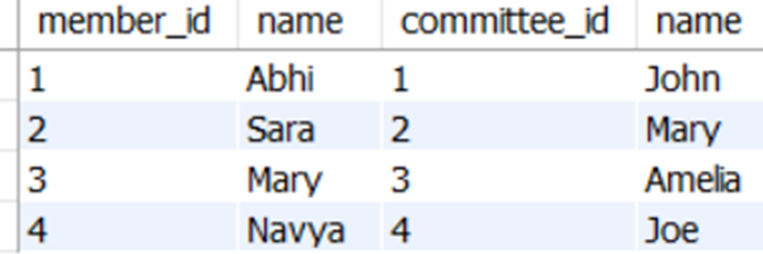


Suppose we do inner join on table1 and table2

SELECT \* FROM members m

INNER JOIN committees c ON m.member\_id = c.committee\_id

Output:



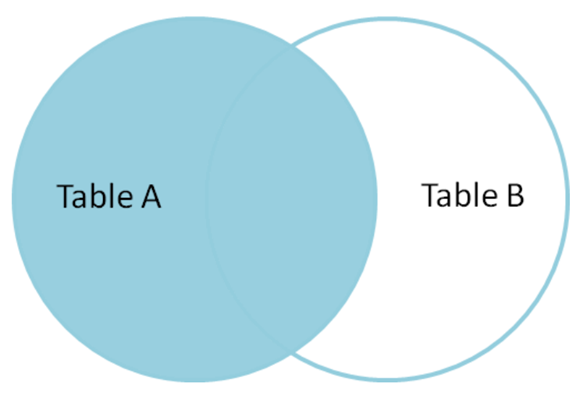
Here we are joining on member\_id and committee\_id columns. The values 1, 2, 3, 4 are common in both columns. So the corresponding rows are joined from the two tables.

## Outer Joins:

These are of two types: Left and Right

### Left Join:

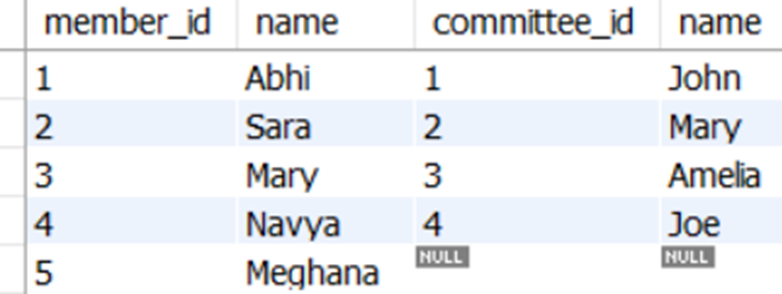
This join will take all the rows from the left table. The rows for which no values are available in right table will be null.



SELECT \* FROM members m

LEFT OUTER JOIN committees c ON m.member\_id = c.committee\_id

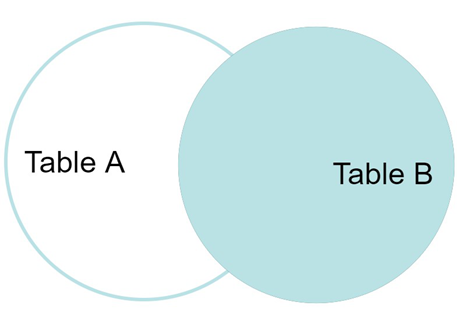
Output:



Here, in the left join we have members table to the left. So all the rows of members table will be joined with the right table. If the right table doesn’t have a specific row, null is added. The committee table do not have committee\_id = 5, so null values are returned.

### Right Join:

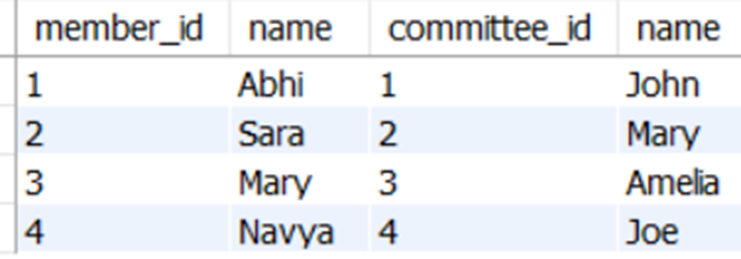
This join will take all the rows from the Right table. The rows for which no values are available in left table will be null.



SELECT \* FROM members m

RIGHT OUTER JOIN committees c ON m.member\_id = c.committee\_id

Here, in the Right join we have committee table to the right. So all the rows of committee table will be joined with the left table. As all the values 1, 2, 3, 4 are present in the left table, no null values are returned.

****

## Self join:

A self join is a join that is performed on a single table. It is useful when you want to combine data from the same table based on a join condition. The syntax for a self join is as follows:

SELECT column1, column2, ...

FROM table1 t1

JOIN table1 t2

ON t1.column\_name = t2.column\_name;

In this syntax, the self join is performed on the same table "table1". The table is given different aliases "t1" and "t2" to differentiate between the two instances of the table. The join condition is specified by the ON clause.

A self join can be used to find relationships between data in the same table, such as finding all employees who have the same manager, or finding all orders where the billing and shipping addresses are the same.

## Natural join:

A natural join is a join that is performed based on the common column names between two tables. It automatically matches the columns with the same name and returns the result set. The syntax for a natural join is as follows:

SELECT column1, column2, ...

FROM table1

NATURAL JOIN table2;

In this syntax, the NATURAL JOIN keyword is used to join two tables based on the common column names. The result set will contain only the columns that are common between the two tables.

A natural join can be useful when you have two tables with similar data and you want to combine them based on the common column names. However, it is important to be careful with natural joins, as they can sometimes produce unexpected results if the column names are not unique or if there are columns with the same name but different data types.

In conclusion, joins are a powerful feature of SQL that allow you to combine data from multiple tables. There are different types of joins available in SQL, each of which is used for specific scenarios. Understanding these different types of joins and when to use them can help you write more efficient and effective SQL queries.