**What is Join?**

Join is a binary operation which allows you to combine join product and selection in one single statement. The goal of creating a join condition is that it helps you to combine the data from multiple join tables.

## Types of Join

There are mainly two types of joins in DBMS:

* [Inner Join](https://www.guru99.com/joins-sql-left-right.html#3)
  + [Theta Join](https://www.guru99.com/joins-sql-left-right.html#4)
  + [EQUI join:](https://www.guru99.com/joins-sql-left-right.html#5)
  + [Natural Join (⋈)](https://www.guru99.com/joins-sql-left-right.html#6)
* [Outer Join](https://www.guru99.com/joins-sql-left-right.html#7)
  + [Left Outer Join (A [https://www.guru99.com/images/1/100518_0535_RelationalA1.png](https://www.guru99.com/joins-sql-left-right.html#8) B)](https://www.guru99.com/joins-sql-left-right.html#8)
  + [Right Outer Join (A [https://www.guru99.com/images/1/100518_0535_RelationalA2.png](https://www.guru99.com/joins-sql-left-right.html#9) B)](https://www.guru99.com/joins-sql-left-right.html#9)
  + [Full Outer Join (A [https://www.guru99.com/images/1/100518_0535_RelationalA3.png](https://www.guru99.com/joins-sql-left-right.html#10) B)](https://www.guru99.com/joins-sql-left-right.html#10)

## Inner Join

An inner join is the widely used join operation and can be considered as a default join-type. The inner JOIN is used to return rows from both tables which satisfy the given condition.

### Theta Join

Theta Join allows you to merge two tables based on the condition represented by theta. Theta joins work for all comparison operators.

The general case of JOIN operation is called a Theta join. It is denoted by symbol **θ**

Syntax:

A ⋈θ B

Theta join can use any conditions in the selection criteria.

***Equijoin(⋈):*** Equijoin is a **special case of conditional join** where only equality condition holds between a pair of attributes. As values of two attributes will be equal in result of equijoin, only one attribute will be appeared in result.

Example:Select students whose ROLL\_NO is equal to EMP\_NO of employees

**STUDENT⋈STUDENT.ROLL\_NO=EMPLOYEE.EMP\_NOEMPLOYEE**

**∏(STUDENT.ROLL\_NO, STUDENT.NAME, STUDENT.ADDRESS, STUDENT.PHONE, STUDENT.AGE EMPLOYEE.NAME, EMPLOYEE.ADDRESS, EMPLOYEE.PHONE, EMPLOYEE>AGE)(σ (STUDENT.ROLL\_NO=EMPLOYEE.EMP\_NO) (STUDENT×EMPLOYEE))**

**STUDENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ROLL\_NO** | **NAME** | **ADDRESS** | **PHONE** | **AGE** |
| 1 | RAM | DELHI | 9455123451 | 18 |
| 2 | RAMESH | GURGAON | 9652431543 | 18 |
| 3 | SUJIT | ROHTAK | 9156253131 | 20 |
| 4 | SURESH | DELHI | 9156768971 | 18 |

**EMPLOYEE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EMP\_NO** | **NAME** | **ADDRESS** | **PHONE** | **AGE** |
| 1 | RAM | DELHI | 9455123451 | 18 |
| 5 | NARESH | HISAR | 9782918192 | 22 |
| 6 | SWETA | RANCHI | 9852617621 | 21 |
| 4 | SURESH | DELHI | 9156768971 | 18 |

RESULT:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ROLL\_NO** | **NAME** | **ADDRESS** | **PHONE** | **AGE** | **NAME** | **ADDRESS** | **PHONE** | **AGE** |
| 1 | RAM | DELHI | 9455123451 | 18 | RAM | DELHI | 9455123451 | 18 |
| 4 | SURESH | DELHI | 9156768971 | 18 | SURESH | DELHI | 9156768971 | 18 |

### Natural Join (⋈)

Natural join does not utilize any of the comparison operators. In this type of join, the attributes should have the same name and domain. In this type of join, there should be at least one common attribute between two relations.

It performs selection forming equality on those attributes which appear in both relations and eliminates the duplicate attributes.

Example:

Consider the following two tables

|  |  |
| --- | --- |
| **C** | |
| **Num** | **Square** |
| 2 | 4 |
| 3 | 9 |

|  |  |
| --- | --- |
| **D** | |
| **Num** | **Cube** |
| 2 | 8 |
| 3 | 18 |

C ⋈ D

|  |  |  |
| --- | --- | --- |
| **C ⋈ D** | | |
| **Num** | **Square** | **Cube** |
| 2 | 4 | 8 |
| 3 | 9 | 18 |

## Outer Join

An outer join doesn't require each record in the two join tables to have a matching record. In this type of join, the table retains each record even if no other matching record exists.

### Left Outer Join (A <https://www.guru99.com/images/1/100518_0535_RelationalA5.png> B)

The LEFT JOIN returns all the rows from the table on the left even if no matching rows have been found in the table on the right. Where no matching record found in the table on the right, NULL is returned.

[](https://www.guru99.com/images/1/100518_0535_RelationalA6.png)

Consider the following 2 Tables

|  |  |
| --- | --- |
| **A** | |
| **Num** | **Square** |
| 2 | 4 |
| 3 | 9 |
| 4 | 16 |

|  |  |
| --- | --- |
| **B** | |
| **Num** | **Cube** |
| 2 | 8 |
| 3 | 18 |
| 5 | 75 |

A [https://www.guru99.com/images/1/100518_0535_RelationalA5.png](https://www.guru99.com/images/1/100518_0535_RelationalA5.png) B

|  |  |  |
| --- | --- | --- |
| **A ⋈ B** | | |
| **Num** | **Square** | **Cube** |
| 2 | 4 | 8 |
| 3 | 9 | 18 |
| 4 | 16 | - |

### Right Outer Join ( A <https://www.guru99.com/images/1/100518_0535_RelationalA7.png> B )

RIGHT outer JOIN is the opposite of LEFT JOIN. The RIGHT JOIN returns all the columns from the table on the right even if no matching rows have been found in the table on the left. Where no matches have been found in the table on the left, NULL is returned.

In our example, let's assume that you need to get the names of members and movies rented by them. Now we have a new member who has not rented any movie yet.

[](https://www.guru99.com/images/1/100518_0535_RelationalA8.png)

A [https://www.guru99.com/images/1/100518_0535_RelationalA7.png](https://www.guru99.com/images/1/100518_0535_RelationalA7.png) B

|  |  |  |
| --- | --- | --- |
| **A ⋈ B** | | |
| **Num** | **Cube** | **Square** |
| 2 | 8 | 4 |
| 3 | 18 | 9 |
| 5 | 75 | - |

### Full Outer Join ( A <https://www.guru99.com/images/1/100518_0535_RelationalA9.png> B)

In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

Example:

A [https://www.guru99.com/images/1/100518_0535_RelationalA9.png](https://www.guru99.com/images/1/100518_0535_RelationalA9.png) B

|  |  |  |
| --- | --- | --- |
| **A ⋈ B** | | |
| **Num** | **Square** | **Cube** |
| 2 | 4 | 8 |
| 3 | 9 | 18 |
| 4 | 16 | - |
| 5 | - | 75 |