SQL JOIN & Types of SQL JOIN

A SQL Join is used to fetch or combine data (rows or columns) from two or more tables based on the defined conditions.

Table 1: Order

OrderID	CustomerID	OrderName	ProductName
12025	101	Peter	ABC
12030	105	Robert	XYX
12032	110	James	XYZ
12034	115	Andrew	PQR
12035	120	Mathew	AAA

Table 2: Customer

CustomerID	CustomerName	Country
100	Messy	Maxico
101	Prince	Taiwan
103	Maria Fernandez	Turkey
105	Jasmine	Paris
110	Faf Weasel	Indonesia
120	Romen Rocket	Russia

Now,we have two tables **Order** and the **Customer**. There is a **CustomerID** column common in both tables. So, write the SQL query to define the general relationship to select the matches' records from both tables.

1. Select Order.OrderID, Customer.CustomerName, Customer.Country, Order.ProductName from Order INNER JOIN Customer ON Order.CustomerID = Customer.CustomerID;
   After executing the above SQL

queries, it produces the following output:

OrderID	CustomerName	Country	ProductName
12025	Prince	Taiwan	ABC
12030	Jasmine	Paris	XYX
12032	Faf Weasel	Indonesia	XYZ
12035	Romen Rocket	Russia	AAA

Types of SQL Join

There are different types of joins used in SQL:

1. Inner Join/Simple Join
2. Left Outer Join/Left Join
3. Right Outer Join/ Right Join
4. Full Outer Join
5. Cross Join
6. Self Join

Inner Join

The inner join is used to select all matching rows or columns in both tables or as long as the defined condition is valid in SQL.

Syntax:

1. Select column_1, column_2, column_3 FROM table_1 INNER JOIN table_2 ON table_1.column = table_2.column;
   We can represent the inner join through the Venn diagram, as follows:

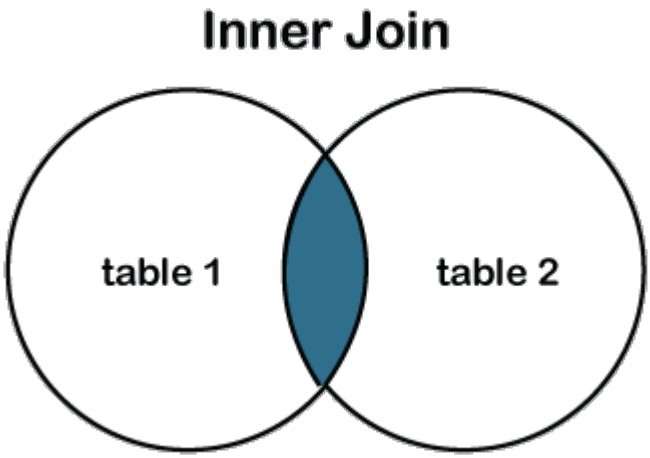


Table 1: Students

Student_ID	StudentName	Subject	TeacherID
101	Alexandra	Computer Science	T201
102	Charles	Economics	T202
103	Tom Cruise	Computer Science	T201
104	Aron Finch	Electronics	T203
105	Siemen Bajoff	Web designing	T204
106	Christopher	English Literature	T205
107	Denim	Fashion Designer	T206

Table 2: Teachers

TeacherID	TeacherName	TeacherEmail
T201	Mr Davis	abc@email.com
T202	Mrs Jonas	jonas@email.com
T201	Mr Davis	abc@email.com
T204	Mrs Lopez	lopez@email.com
T205	Mrs Wiley	wiley@email.com

T206	Mr Bean	bean@email.com

We have two tables: **Students** and the **Teachers** Tables. Let's write the SQL Queries to join the table using the **INNER JOIN** as follows:

1. Select Student_ID, StudentName, TeacherName, TeacherEmail FROM Students INNER JOIN Teachers ON Students.TeacherID = Teachers.TeacherID;
   After executing the query, it produces the below table.

```
+------------+---------------+-------------+-----------------+
| Student_ID | StudentName   | TeacherName | TeacherEmail    |
+------------+---------------+-------------+-----------------+
|        101 | Alexandra     | Mr Davis    | abc@email.com   |
|        101 | Alexandra     | Mr Davis    | abc@email.com   |
|        102 | Charles       | Mrs Jonas   | jonas@email.com |
|        103 | Tom Cruise    | Mr Davis    | abc@email.com   |
|        103 | Tom Cruise    | Mr Davis    | abc@email.com   |
|        105 | Siemen Bajoff | Mrs Lopez   | lopez@email.com |
|        106 | Chrishtopher  | Mrs Wiley   | wiley@email.com |
|        107 | Denim         | Mr Bean     | bean@email.com  |
+------------+---------------+-------------+-----------------+
8 rows in set (0.00 sec)
```

Natural Join

It is a type of inner type that joins two or more tables based on the same column name and has the same data type present on both tables.

Syntax:

1. Select * from tablename1 Natural JOIN tablename_2;
   We have two tables: **Students** and the **Teachers** Tables. Let's write the SQL Queries to join the table using the **Natural JOIN** as follows:

1. Select * from Students Natural JOIN Teachers;
   After executing the above query, it produces the following table.

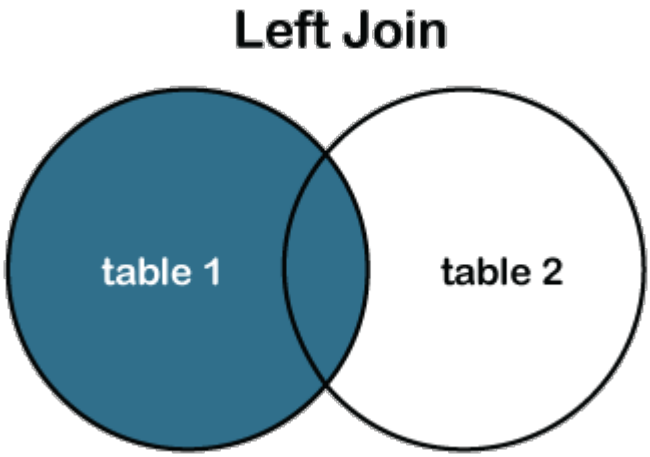
```
+-----------+------------+---------------+--------------------+-------------+-------------
| TeacherID | Student_ID | StudentName   | Subject            | TeacherName | TeacherEmail
+-----------+------------+---------------+--------------------+-------------+-------------
| T201      |        101 | Alexandra     | Computer Science   | Mr Davis    | abc@email.co
| T201      |        101 | Alexandra     | Computer Science   | Mr Davis    | abc@email.co
| T202      |        102 | Charles       | Economics          | Mrs Jonas   | jonas@email.
| T201      |        103 | Tom Cruise    | Computer Science   | Mr Davis    | abc@email.co
| T201      |        103 | Tom Cruise    | Computer Science   | Mr Davis    | abc@email.co
| T204      |        105 | Siemen Bajoff | Web Designing      | Mrs Lopez   | lopez@email.
| T205      |        106 | Chrishtopher  | English Literature | Mrs Wiley   | wiley@email.
| T206      |        107 | Denim         | Fashion Designer   | Mr Bean     | bean@email.c
+-----------+------------+---------------+--------------------+-------------+-------------
8 rows in set (0.10 sec)
```

LEFT JOIN

The LEFT Join is used to retrieve all records from the left table (table1) and the matched rows or columns from the right table (table2). If both tables do not contain any matched rows or columns, it returns the NULL.

Syntax:

1. Select column_1, column_2, column(s) FROM table_1 LEFT JOIN table_2 ON table_1.column_name = table_2.column_name;
   We can also represent the left join through the Venn diagram, as follows:



Note: In some databases, LEFT JOIN is also known as LEFT OUTER JOIN.

Table 1: Product_Details

ProductID	ProductName	Amount
Pro101	Laptop	56000
Pro102	Mobile	38000
Pro103	Headphones	5000
Pro104	Television	25000
Pro105	iPad	60000

Table 2: Customer_Details

CustomerName	CustomerAddress	CustomerAge	ProductID
Martin Guptill	San Francisco, USA	26	Pro101
James	Australia	29	Pro103

Ambati Williamson	New Zealand	27	Pro102
Jofra Archer	South Africa	24	Pro105
Kate Wiley	Australia	20	Pro103

We have two tables: **Product_Details** and the **Customer_Details** Tables. Let's write the SQL Queries to join the table using the **LEFT JOIN** as follows:

1. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details LEFT JOIN Customer_Details ON Product_Details.ID = Customer_Details.ProductID;
After executing the query, it produces the following table.

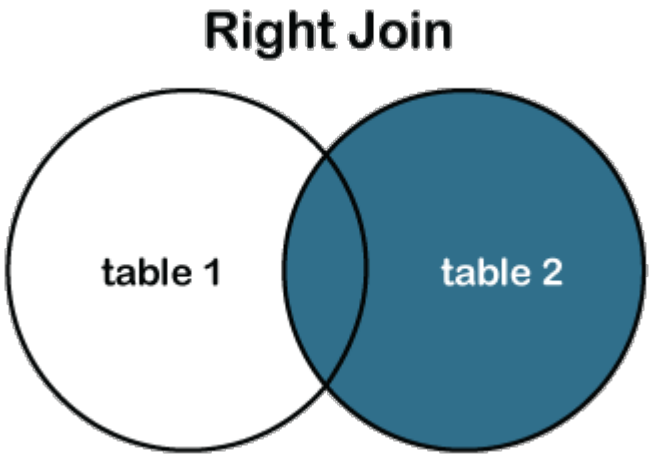
```
+--------+-------------+-------------------+---------------------+--------+
| ID     | ProductName | CustomerName      | CustomerAddress     | Amount |
+--------+-------------+-------------------+---------------------+--------+
| Pro101 | Laptop      | Martin Guptill    | San Francisco, USA  |  56000 |
| Pro102 | Mobile      | Ambati Williamson | New Zealand         |  38000 |
| Pro103 | HeadPhone   | James             | Australia           |   5000 |
| Pro104 | Television  | NULL              | NULL                |  25000 |
| Pro105 | iPad        | Jofra Archer      | South Africa        |  60000 |
+--------+-------------+-------------------+---------------------+--------+
5 rows in set (0.02 sec)
```

RIGHT JOIN or RIGHT Outer JOIN:

The RIGHT JOIN is used to retrieve all records from the right table (table2) and the matched rows or columns from the left table (table1). If both tables do not contain any matched rows or columns, it returns the NULL.

Syntax:

1. Select column_1, column_2, column(s) FROM table_1 RIGHT JOIN table_2 ON table_1.column_name = table_2.column_name;
We can also represent the right join through the Venn diagram, as follows:



Note: In some databases, the RIGHT JOIN is also known as the RIGHT OUTER JOIN.

Table 1: Product_Details

ID	ProductName	Amount
Pro101	Laptop	56000
Pro102	Mobile	38000
Pro103	Headphones	5000
Pro104	Television	25000
Pro105	iPad	60000

Table 2: Customer_Details

CustomerName	CustomerAddress	CustomerAge	ProductID
Martin Guptill	San Francisco, USA	26	Pro101
James	Australia	29	Pro103
Ambati Williamson	New Zealand	27	Pro102
Jofra Archer	South Africa	24	Pro105
Omen	England	29	Pro107
Morgan	England	20	Pro108

We have two tables: **Product_Details** and the **Customer_Details** Tables. Let's write the SQL Queries to join the table using the **RIGHT JOIN** as follows:

1. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details LEFT JOIN Customer_Details ON Product_Details.ID = Customer_Details.ProductID;
After executing the query, it produces the below table.

```
+--------+-------------+-------------------+--------------------+--------+
| ID     | ProductName | CustomerName      | CustomerAddress    | Amount |
+--------+-------------+-------------------+--------------------+--------+
| Pro101 | Laptop      | Martin Guptill    | San Francisco, USA |  56000 |
| Pro103 | HeadPhone   | James             | Australia          |   5000 |
| Pro102 | Mobile      | Ambati Williamson | New Zealand        |  38000 |
| Pro105 | iPad        | Jofra Archer      | South Africa       |  60000 |
| NULL   | NULL        | Omen              | England            |   NULL |
| NULL   | NULL        | Morgan            | England            |   NULL |
+--------+-------------+-------------------+--------------------+--------+
6 rows in set (0.01 sec)
```

FULL JOIN or FULL Outer JOIN:

It is a combination result set of both **LEFT JOIN** and **RIGHT JOIN**. The joined tables return all records from both the tables and if no matches are found in the table, it places NULL. It is also called a **FULL Outer JOIN.**

Syntax:

1. Select column_1, column_2, column(s) FROM table_1 FULL JOIN table_2 ON table_1.column_name = table_2.column_name;
   Or, **FULL OUTER JOIN**

1. Select column_1, column_2, column(s) FROM table_1 FULL OUTER JOIN table_2 ON table_1.column_name = table_2.column_name;
   We can also represent the full outer join through the Venn diagram, as follows:

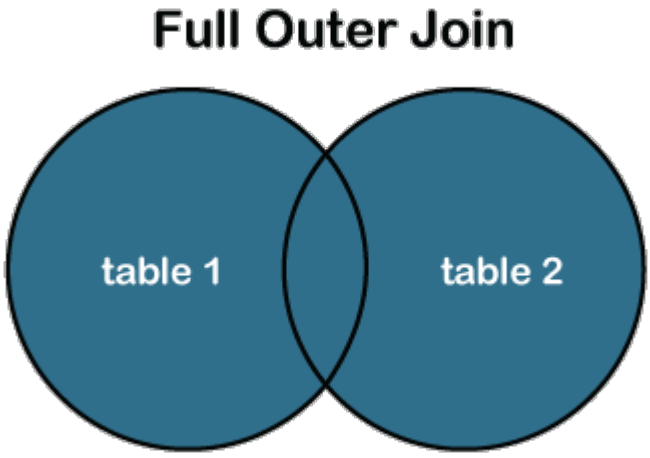


Table 1: Product_Details

ID	ProductName	Amount
Pro101	Laptop	56000
Pro10	Mobile	38000

2		
Pro103	Headphones	5000
Pro104	Television	25000
Pro105	iPad	60000

Table 2: Customer_Details

CustomerName	CustomerAddress	CustomerAge	ProductID
Martin Guptill	San Francisco, USA	26	Pro101
James	Australia	29	Pro103
Ambati Williamson	New Zealand	27	Pro102
Jofra Archer	South Africa	24	Pro105
Omen	England	29	Pro107
Morgan	England	20	Pro108

We have two tables: **Product_Details** and the **Customer_Details** Tables. Let's write the SQL Queries to join the table using the **FULL** JOIN as follows:

1. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details FULL JOIN Customer_Details ON Product_Details.ID = Customer_Details.ProductID;
After executing the query, it produces the below table.

```
+--------+-------------+-------------------+--------------------+--------+
| Pro101 | Laptop      | Martin Guptill    | San Francisco, USA |  56000 |
| Pro102 | Mobile      | Ambati Williamson | New Zealand        |  38000 |
| Pro103 | HeadPhone   | James             | Australia          |   5000 |
| Pro104 | Television  | NULL              | NULL               |  25000 |
| Pro105 | iPad        | Jofra Archer      | South Africa       |  60000 |
| Pro101 | Laptop      | Martin Guptill    | San Francisco, USA |  56000 |
| Pro103 | HeadPhone   | James             | Australia          |   5000 |
| Pro102 | Mobile      | Ambati Williamson | New Zealand        |  38000 |
| Pro105 | iPad        | Jofra Archer      | South Africa       |  60000 |
| NULL   | NULL        | Omen              | England            |   NULL |
| NULL   | NULL        | Morgan            | England            |   NULL |
+--------+-------------+-------------------+--------------------+--------+
```

> Note: MySQL does not support FULL JOIN concepts, so we can use UNION ALL clause to combine both tables.

Here is the Syntax for **UNION ALL** Clause to combine the tables.

1. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details LEFT JOIN Customer_Details ON Product_Details.ID = Customer_Details.ProductID
2. UNION ALL
3. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details RIGHT JOIN Customer_Details ON Product_Details.ID = Customer_Details.ProductID

```
+--------+-------------+--------------------+--------------------+--------+
| Pro101 | Laptop      | Martin Guptill     | San Francisco, USA |  56000 |
| Pro102 | Mobile      | Ambati Williamson  | New Zealand        |  38000 |
| Pro103 | HeadPhone   | James              | Australia          |   5000 |
| Pro104 | Television  | NULL               | NULL               |  25000 |
| Pro105 | iPad        | Jofra Archer       | South Africa       |  60000 |
| Pro101 | Laptop      | Martin Guptill     | San Francisco, USA |  56000 |
| Pro103 | HeadPhone   | James              | Australia          |   5000 |
| Pro102 | Mobile      | Ambati Williamson  | New Zealand        |  38000 |
| Pro105 | iPad        | Jofra Archer       | South Africa       |  60000 |
| NULL   | NULL        | Omen               | England            |   NULL |
| NULL   | NULL        | Morgan             | England            |   NULL |
+--------+-------------+--------------------+--------------------+--------+
```

CROSS JOIN

It is also known as **CARTESIAN JOIN**, which returns the Cartesian product of two or more joined tables. The **CROSS JOIN**
produces a table that merges each row from the first table with each second table row. It is not required to include any condition in CROSS JOIN.

Syntax:

1. Select * from table_1 cross join table_2;
   Or,

1. Select column1, column2, column3 FROM table_1, table_2;
   Table 1: Product_Details

ID	ProductName	Amount
Pro101	Laptop	56000
Pro102	Mobile	38000
Pro103	Headphones	5000

Pro104	Television	25000
Pro105	iPad	60000

Table 2: Customer_Details

CustomerName	CustomerAddress	CustomerAge	ProductID
Martin Guptill	San Francisco, USA	26	Pro101
James	Australia	29	Pro103
Ambati Williamson	New Zealand	27	Pro102
Jofra Archer	South Africa	24	Pro105
Omen	England	29	Pro107
Morgan	England	20	Pro108

We have two tables: **Product_Details** and the **Customer_Details** Tables. Let's write the SQL Queries to join the table using the **FULL** JOIN as follows:

1. Select ID, ProductName, CustomerName, CustomerAddress, Amount FROM Product_Details, Customer_Details;
   After executing the query, it produces the below table.

```
+--------+-------------+-------------------+--------------------+--------+
| ID     | ProductName | CustomerName      | CustomerAddress    | Amount |
+--------+-------------+-------------------+--------------------+--------+
| Pro101 | Laptop      | Martin Guptill    | San Francisco, USA |  56000 |
| Pro102 | Mobile      | Martin Guptill    | San Francisco, USA |  38000 |
| Pro103 | HeadPhone   | Martin Guptill    | San Francisco, USA |   5000 |
| Pro104 | Television  | Martin Guptill    | San Francisco, USA |  25000 |
| Pro105 | iPad        | Martin Guptill    | San Francisco, USA |  60000 |
| Pro101 | Laptop      | James             | Australia          |  56000 |
| Pro102 | Mobile      | James             | Australia          |  38000 |
| Pro103 | HeadPhone   | James             | Australia          |   5000 |
| Pro104 | Television  | James             | Australia          |  25000 |
| Pro105 | iPad        | James             | Australia          |  60000 |
| Pro101 | Laptop      | Ambati Williamson | New Zealand        |  56000 |
| Pro102 | Mobile      | Ambati Williamson | New Zealand        |  38000 |
| Pro103 | HeadPhone   | Ambati Williamson | New Zealand        |   5000 |
| Pro104 | Television  | Ambati Williamson | New Zealand        |  25000 |
| Pro105 | iPad        | Ambati Williamson | New Zealand        |  60000 |
| Pro101 | Laptop      | Jofra Archer      | South Africa       |  56000 |
| Pro102 | Mobile      | Jofra Archer      | South Africa       |  38000 |
| Pro103 | HeadPhone   | Jofra Archer      | South Africa       |   5000 |
| Pro104 | Television  | Jofra Archer      | South Africa       |  25000 |
| Pro105 | iPad        | Jofra Archer      | South Africa       |  60000 |
| Pro101 | Laptop      | Omen              | England            |  56000 |
| Pro102 | Mobile      | Omen              | England            |  38000 |
| Pro103 | HeadPhone   | Omen              | England            |   5000 |
| Pro104 | Television  | Omen              | England            |  25000 |
| Pro105 | iPad        | Omen              | England            |  60000 |
| Pro101 | Laptop      | Morgan            | England            |  56000 |
| Pro102 | Mobile      | Morgan            | England            |  38000 |
| Pro103 | HeadPhone   | Morgan            | England            |   5000 |
| Pro104 | Television  | Morgan            | England            |  25000 |
| Pro105 | iPad        | Morgan            | England            |  60000 |
+--------+-------------+-------------------+--------------------+--------+
30 rows in set (0.00 sec)
```

SELF JOIN

It is a SELF JOIN used to create a table by joining itself as there were two tables. It makes temporary naming of at least one table in an SQL statement.

Syntax:

1. Select column1, column2, column(s) FROM table_1 Tbl1, table_2 Tbl2 WHERE condition;
   Tbl1 and **Tbl2** are two different table aliases for the same table.

Table 1: Product_Details

ID	ProductName	Amount
Pro101	Laptop	56000
Pro102	Mobile	38000

Pro103	Headphones	5000
Pro104	Television	25000
Pro105	iPad	60000

Let's write the SQL Queries to join the table using the **SELF JOIN** as follows:

1. Select TB.ID, TB.ProductName FROM Product_Details TB, Product_Details TB2
2. WHERE TB.AMOUNT **< TB2.AMOUNT**;

After executing the query, it produces the below table.

```
+--------+-------------+
| ID     | ProductName |
+--------+-------------+
| Pro102 | Mobile      |
| Pro103 | HeadPhone   |
| Pro104 | Television  |
| Pro103 | HeadPhone   |
| Pro104 | Television  |
| Pro103 | HeadPhone   |
| Pro101 | Laptop      |
| Pro102 | Mobile      |
| Pro103 | HeadPhone   |
| Pro104 | Television  |
+--------+-------------+
10 rows in set (0.02 sec)
```