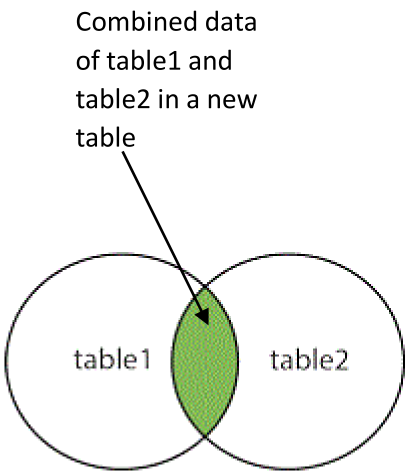
SQL JOINS

What is a SQL Join?

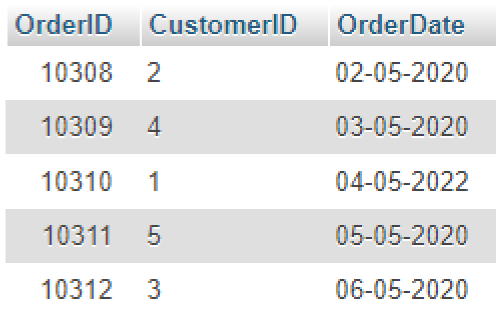
A SQL join is a Structured Query Language (**SQL**) instruction to combine data from two sets of data (i.e. two tables).

A join is a means for combining columns from one or more tables by using values common to each.



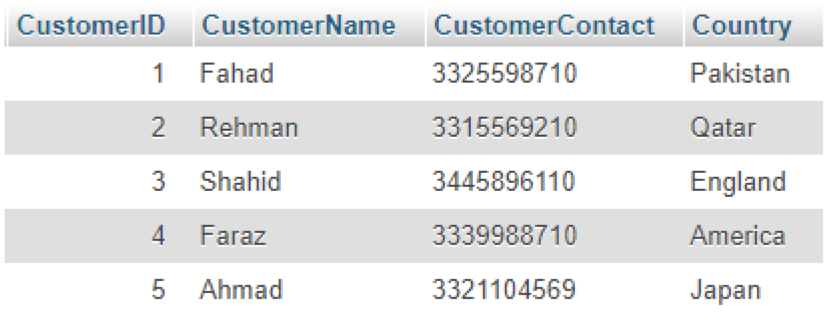
Note: we will use the following two tables while practicing on joins examples.

- “Orders” table



OrderID	CustomerID	OrderDate
10308	2	02-05-2020
10309	4	03-05-2020
10310	1	04-05-2022
10311	5	05-05-2020
10312	3	06-05-2020

- “Customer” table



CustomerID	CustomerName	CustomerContact	Country
1	Fahad	3325598710	Pakistan
2	Rehman	3315569210	Qatar
3	Shahid	3445896110	England
4	Faraz	3339988710	America
5	Ahmad	3321104569	Japan

There are three main types of SQL join:

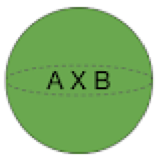
- Cross Join
- Inner Join
- Outer Join

CROSS JOIN:

In MySQL, the CROSS JOIN produced a result set which is the product of rows of two associated tables when no WHERE clause is used with CROSS JOIN.

In this join, the result set appeared by multiplying each row of the first table with all rows in the second table if no condition introduced with CROSS JOIN.

This kind of result is called as Cartesian product.



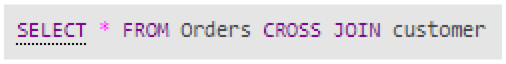
CARTESIAN
(CROSS) JOIN

Syntax:

SELECT * FROM *table1* CROSS JOIN *table2;*

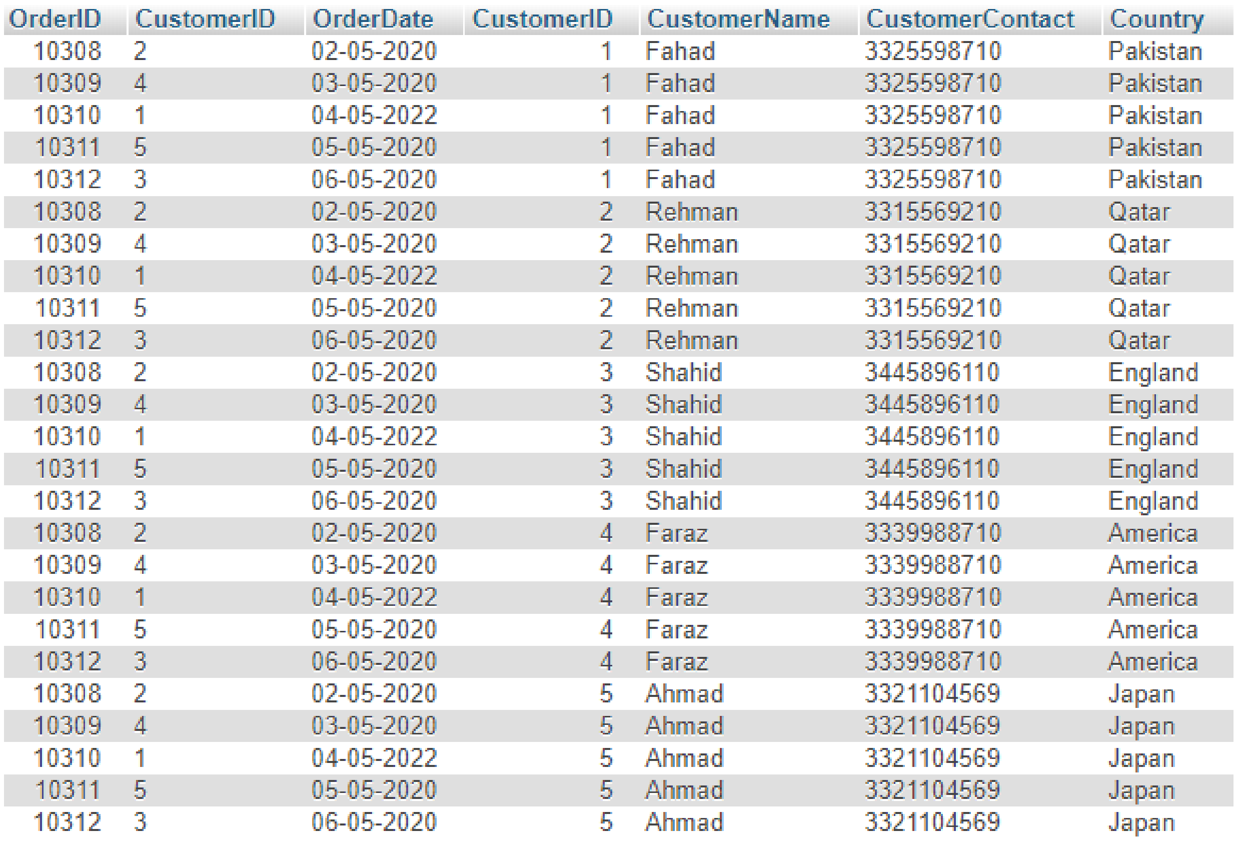
For example:

If we apply the query on “Order” and “Customer” tables. We get the following result:



```
SELECT * FROM Orders CROSS JOIN customer
```

Result:

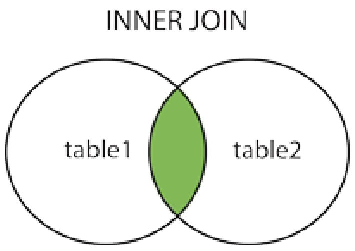


OrderID	CustomerID	OrderDate	CustomerID	CustomerName	CustomerContact	Country
10308	2	02-05-2020	1	Fahad	3325598710	Pakistan
10309	4	03-05-2020	1	Fahad	3325598710	Pakistan
10310	1	04-05-2022	1	Fahad	3325598710	Pakistan
10311	5	05-05-2020	1	Fahad	3325598710	Pakistan
10312	3	06-05-2020	1	Fahad	3325598710	Pakistan
10308	2	02-05-2020	2	Rehman	3315569210	Qatar
10309	4	03-05-2020	2	Rehman	3315569210	Qatar
10310	1	04-05-2022	2	Rehman	3315569210	Qatar
10311	5	05-05-2020	2	Rehman	3315569210	Qatar
10312	3	06-05-2020	2	Rehman	3315569210	Qatar
10308	2	02-05-2020	3	Shahid	3445896110	England
10309	4	03-05-2020	3	Shahid	3445896110	England
10310	1	04-05-2022	3	Shahid	3445896110	England
10311	5	05-05-2020	3	Shahid	3445896110	England
10312	3	06-05-2020	3	Shahid	3445896110	England
10308	2	02-05-2020	4	Faraz	3339988710	America
10309	4	03-05-2020	4	Faraz	3339988710	America
10310	1	04-05-2022	4	Faraz	3339988710	America
10311	5	05-05-2020	4	Faraz	3339988710	America
10312	3	06-05-2020	4	Faraz	3339988710	America
10308	2	02-05-2020	5	Ahmad	3321104569	Japan
10309	4	03-05-2020	5	Ahmad	3321104569	Japan
10310	1	04-05-2022	5	Ahmad	3321104569	Japan
10311	5	05-05-2020	5	Ahmad	3321104569	Japan
10312	3	06-05-2020	5	Ahmad	3321104569	Japan

INNER JOIN:

The INNER JOIN selects all rows from both participating tables as long as there is a match between the columns. An SQL INNER JOIN is same as JOIN clause, combining rows from two or more tables.

There must a same column in both the table to identify the relationship between two tables.



Syntax:

SELECT * FROM *table1* INNER JOIN *table2* ON *table1.column_name*

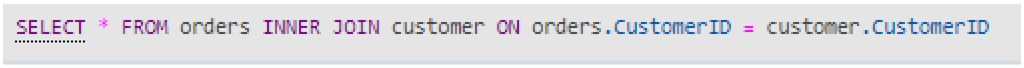
= *tabale2. column_name;*

Common column between two tables

For example:

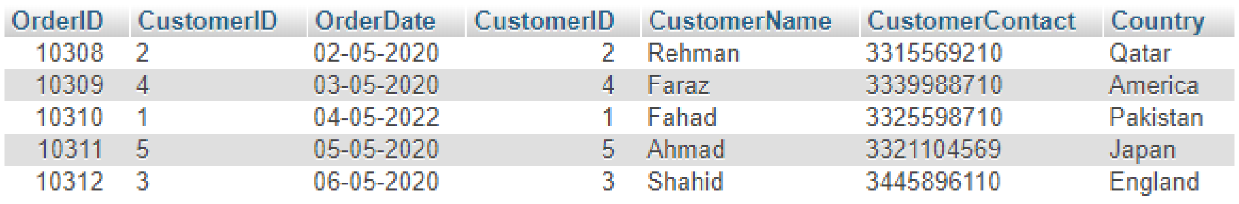
If we apply inner join query on both the tables ("Orders" and "Customer") we will get the following result.

Note: In our example "CustomerID" is the common column in between two tables.



```
SELECT * FROM orders INNER JOIN customer ON orders.CustomerID = customer.CustomerID
```

Result:



OrderID	CustomerID	OrderDate	CustomerID	CustomerName	CustomerContact	Country
10308	2	02-05-2020	2	Rehman	3315569210	Qatar
10309	4	03-05-2020	4	Faraz	3339988710	America
10310	1	04-05-2022	1	Fahad	3325598710	Pakistan
10311	5	05-05-2020	5	Ahmad	3321104569	Japan
10312	3	06-05-2020	3	Shahid	3445896110	England

OUTER JOIN:

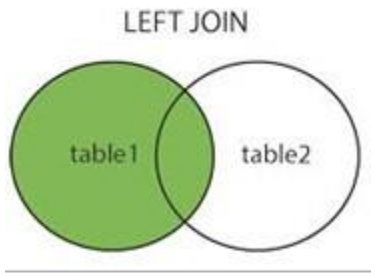
The Outer Join includes the matching rows as well as some of the non-matching rows between the two tables. An Outer join basically differs from the Inner join in how it handles the false match condition.

There are three main types of outer join:

- Left outer join or Left join
- Right outer join or Right join
- Full outer join or Full join

LEFT OUTER JOIN OR LEFT JOIN:

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

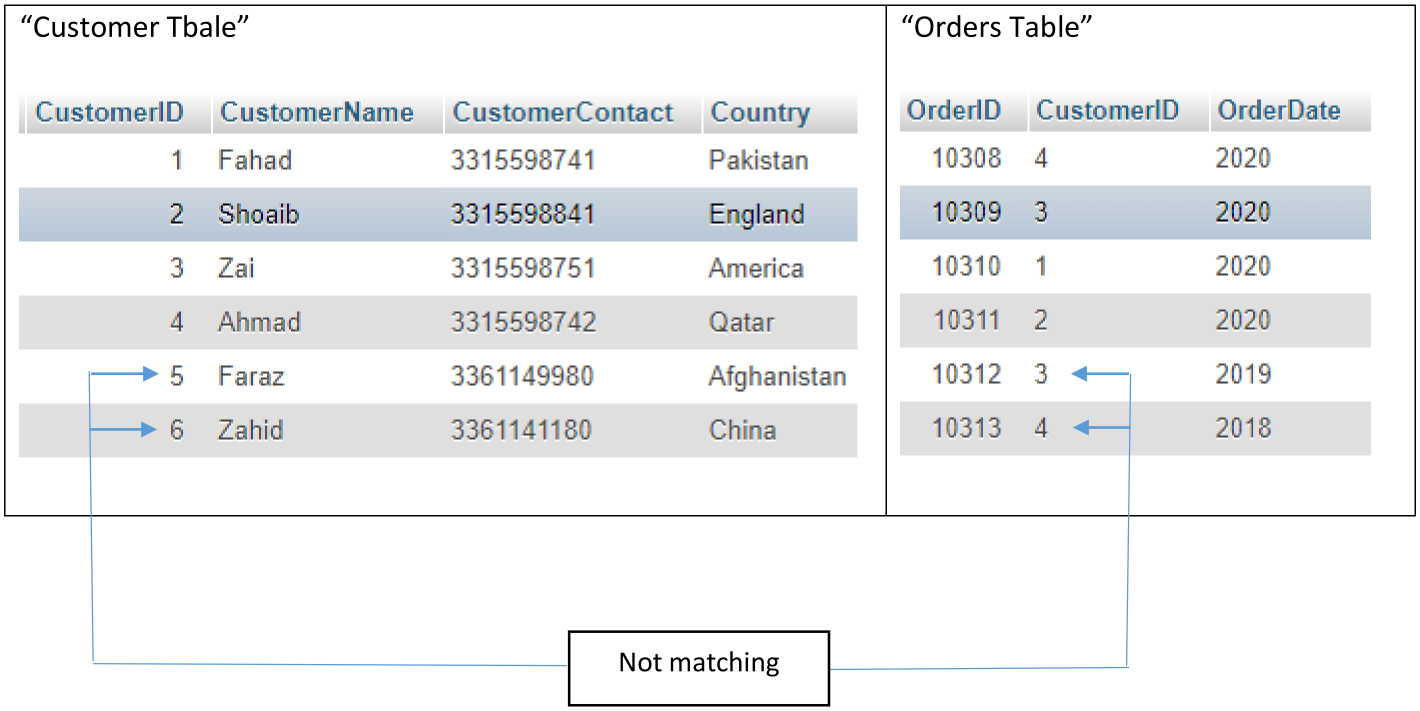


Syntax:

SELECT *column_name(s)*
FROM *table1*
LEFT JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

For example:

We have two tables "Customer" and "Orders". We entered two more row in both the table but here in "Customer" table 5th and 6th entry of CustomerID in not matching with the 5th and 6th entry of CustomerId column in "Orders" table.


Not matching

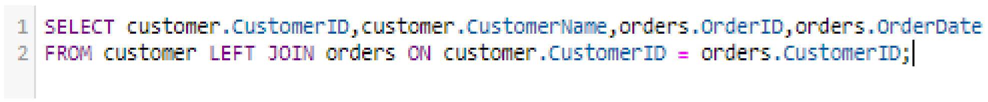
"Customer Tbale"

CustomerID	CustomerName	CustomerContact	Country
1	Fahad	3315598741	Pakistan
2	Shoaib	3315598841	England
3	Zai	3315598751	America
4	Ahmad	3315598742	Qatar
5	Faraz	3361149980	Afghanistan
6	Zahid	3361141180	China

"Orders Table"

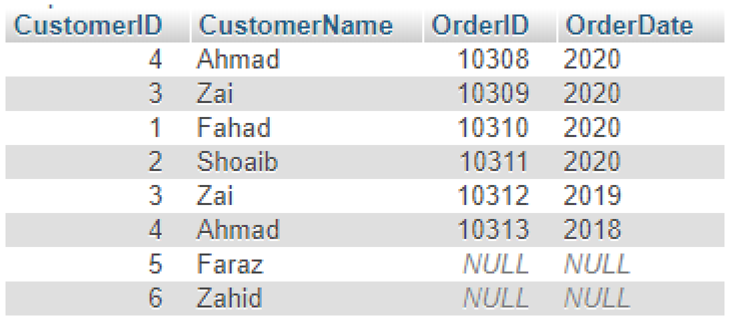
OrderID	CustomerID	OrderDate
10308	4	2020
10309	3	2020
10310	1	2020
10311	2	2020
10312	3	2019
10313	4	2018

After applying LEFT JOIN on both the tables we will get the results. In results we will get all the values from left table which is “Customer” and in “Orders” table we will get the null values for Faraz and Zahid because they do not have any order.



```
SELECT customer.CustomerID,customer.CustomerName,orders.OrderID,orders.OrderDate
FROM customer LEFT JOIN orders ON customer.CustomerID = orders.CustomerID;
```

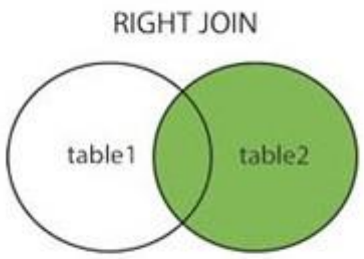
Result:



CustomerID	CustomerName	OrderID	OrderDate
4	Ahmad	10308	2020
3	Zai	10309	2020
1	Fahad	10310	2020
2	Shoaib	10311	2020
3	Zai	10312	2019
4	Ahmad	10313	2018
5	Faraz	NULL	NULL
6	Zahid	NULL	NULL

RIGHT OUTR JOIN OR RIGHT JOIN:

The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.



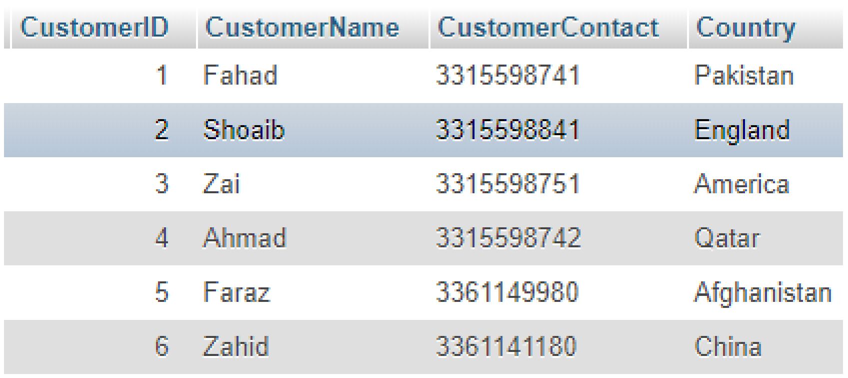
Syntax:

SELECT *column_name(s)*
FROM *table1*
RIGHT JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

For example:

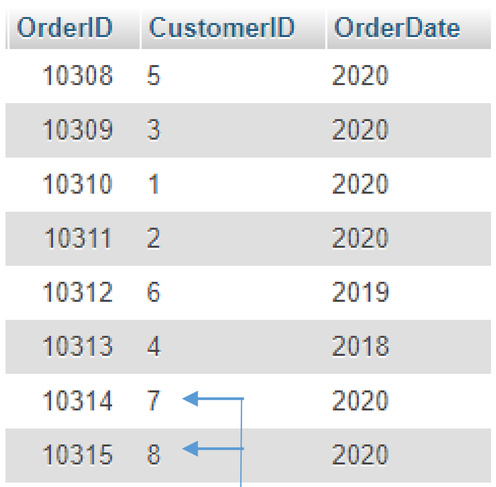
Considering above two table that are “Customer” and “Order”. Now we have two entries in “Orders” table that are not matching in CustomerID (7&8). Which means that there are two orders placed but they do not belong to any customer so it will show null in “Customer” table.

“Customer Table”



CustomerID	CustomerName	CustomerContact	Country
1	Fahad	3315598741	Pakistan
2	Shoaib	3315598841	England
3	Zai	3315598751	America
4	Ahmad	3315598742	Qatar
5	Faraz	3361149980	Afghanistan
6	Zahid	3361141180	China

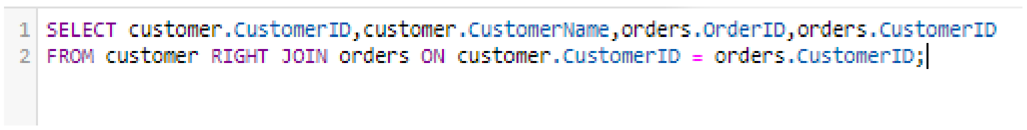
“Order Table”



OrderID	CustomerID	OrderDate
10308	5	2020
10309	3	2020
10310	1	2020
10311	2	2020
10312	6	2019
10313	4	2018
10314	7	2020
10315	8	2020

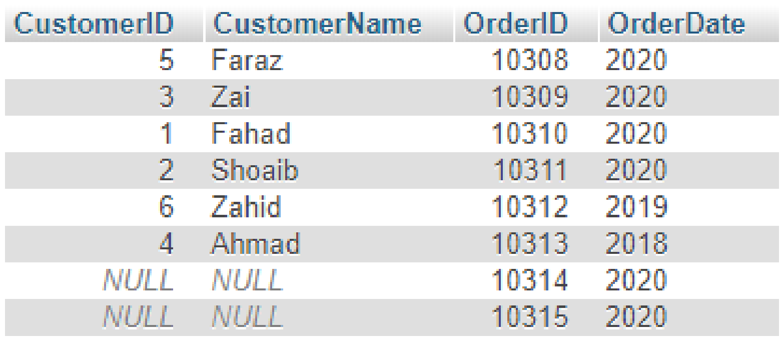
7 and 8 entry is not present in customer table

After applying the Right Join we will get the results. In results, we will get all the records of right table (“Orders”) and “Customer” will have two null records because 7th and 8th customer is not present there.



```sql
SELECT customer.CustomerID,customer.CustomerName,orders.OrderID,orders.CustomerID
FROM customer RIGHT JOIN orders ON customer.CustomerID = orders.CustomerID;
```

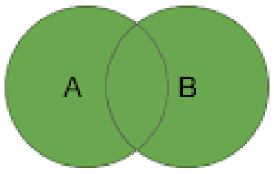
Result:



CustomerID	CustomerName	OrderID	OrderDate
5	Faraz	10308	2020
3	Zai	10309	2020
1	Fahad	10310	2020
2	Shoaib	10311	2020
6	Zahid	10312	2019
4	Ahmad	10313	2018
NULL	*NULL*	10314	2020
NULL	*NULL*	10315	2020

FULL OUTER JOIN OR FULL JOIN:

In SQL the FULL OUTER JOIN combines the results of both left and right outer joins and returns all (matched or unmatched) rows from the tables on both sides of the join clause.

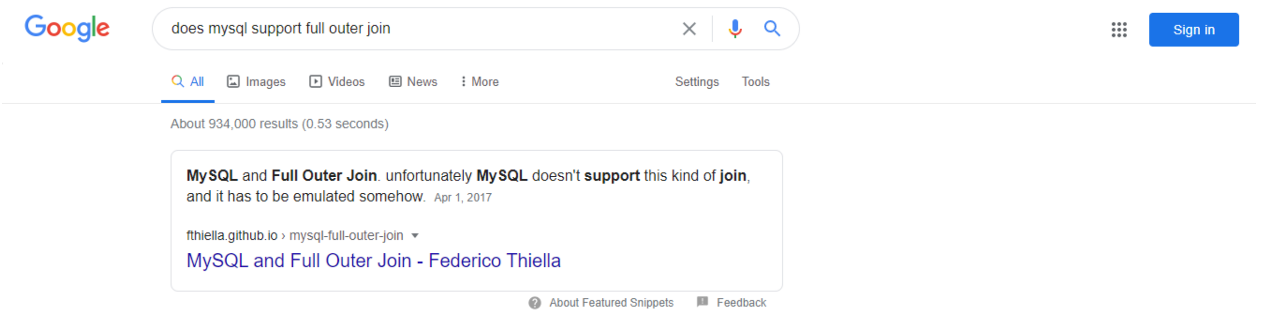


FULL OUTER JOIN

Syntax:

SELECT *column_name(s)*
FROM *table1*
FULL JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

NOTE: Unfortunately MySQL does not support full join. We will cover it in further topics using different join techniques.



About 934,000 results (0.53 seconds)

MySQL and **Full Outer Join**. unfortunately **MySQL** doesn't **support** this kind of **join**, and it has to be emulated somehow. Apr 1, 2017

fthiella.github.io › mysql-full-outer-join

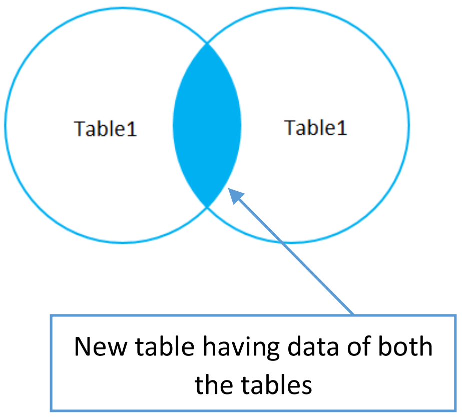
MySQL and Full Outer Join - Federico Thiella

SELF JOIN:

A SELF JOIN is a join that is used to join a table with **itself**. In the previous sections, we have learned about the joining of the table with the other tables using different JOINS, such as INNER, LEFT, RIGHT, and CROSS JOIN. However, there is a need to combine data with other data in the same table itself. In that case, we use Self Join.

We can perform Self Join using **table aliases**. The table aliases allow us not to use the same table name twice with a single statement. If we use the same table name more than one time in a single query without table aliases, it will throw an error.

The table aliases enable us to use the **temporary name** of the table that we are going to use in the query. Let us understand the table aliases with the following explanation.

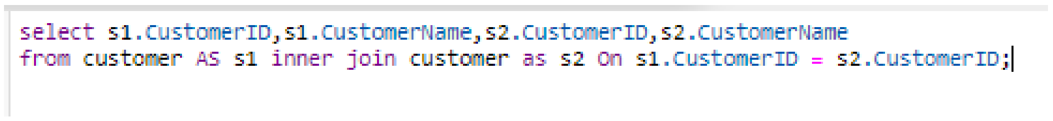


Syntax:

Select *s1.cloumn(s), s2.column(s)* from *table1* AS *s1* INNER JOIN *table1* AS *s2*;

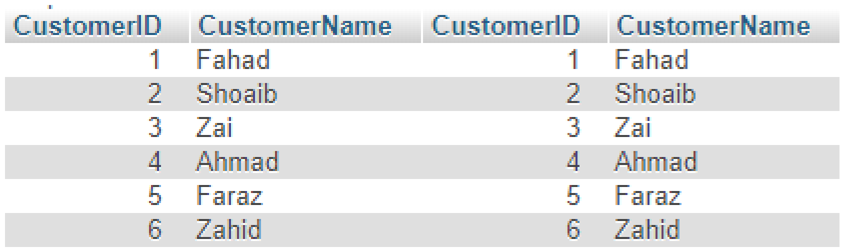
For example:

If we apply SELF JOIN on “Customer” table we will get the results. In results, we will get “Customer” table joins with itself.



```
select s1.CustomerID,s1.CustomerName,s2.CustomerID,s2.CustomerName
from customer AS s1 inner join customer as s2 On s1.CustomerID = s2.CustomerID;
```

Results:



CustomerID	CustomerName	CustomerID	CustomerName
1	Fahad	1	Fahad
2	Shoaib	2	Shoaib
3	Zai	3	Zai
4	Ahmad	4	Ahmad
5	Faraz	5	Faraz
6	Zahid	6	Zahid