Praneeth Kumar Thiruveedhula SeniorSoftware Engineer

Database JOINS

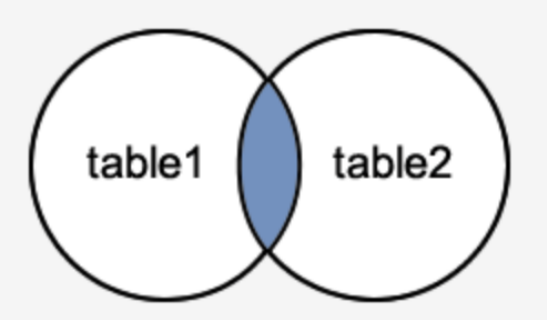
This page explains about joins in SQL. with examples. I used Oracle database for this example.

Joins used to retrieve the data from multiple Tables (Two or more).

Joins are 4 Types
- INNER JOIN (also called simple join)
- LEFT OUTER JOIN (LEFT JOIN)
- RIGHT OUTER JOIN (RIGHT JOIN)
- FULL OUTER JOIN (FULL JOIN)

INNER JOIN (simple join)

It returns all the rows from Multiple tables where the join condition is met.



Let's understand more detail

We have the following tables Orders, Suppliers. It contains the following data.

supplier_id	supplier_name
10000	IBM
10001	Hewlett Packard
10002	Microsoft
10003	NVIDIA

suppliers Table

order_id	supplier_id	order_date
500125	10000	2003/05/12
500126	10001	2003/05/13
500127	10004	2003/05/14

orders Table

Here is the **Inner Join** Query

```
SELECT suppliers.supplier_id, suppliers.supplier_name, orders.order_date
FROM suppliers
INNER JOIN orders
ON suppliers.supplier_id = orders.supplier_id;
```

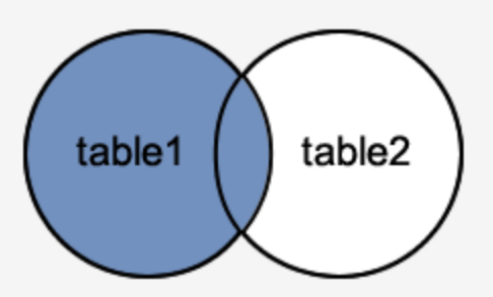
We get the below output.

supplier_id	name	order_date
10000	IBM	2003/05/12
10001	Hewlett Packard	2003/05/13

LEFT OUTER JOIN (Left join)

It returns all rows from the LEFT-hand table specified in the ON condition and only those rows from the other table where the joined fields are equal (join condition is met).

In some databases, the LEFT OUTER JOIN keywords are replaced with LEFT JOIN.



In some databases, the LEFT OUTER JOIN keywords are replaced with LEFT JOIN.

Let's understand in detail. We have the two tables as below.

supplier_id	supplier_name
10000	IBM
10001	Hewlett Packard
10002	Microsoft
10003	NVIDIA

Suppliers Table

order_id	supplier_id	order_date
500125	10000	2003/05/12
500126	10001	2003/05/13

Orders Table

If we run the SELECT statement (that contains a LEFT OUTER JOIN) below:

```
SELECT suppliers.supplier_id, suppliers.supplier_name, orders.order_date
FROM suppliers
LEFT OUTER JOIN orders
ON suppliers.supplier_id = orders.supplier_id;
```

We get the below Output.

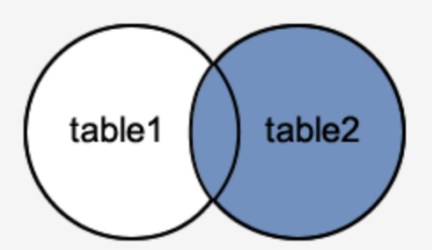
supplier_id	supplier_name	order_date
10000	IBM	2003/05/12
10001	Hewlett Packard	2003/05/13
10002	Microsoft	<null>
10003	NVIDIA	<null>

Old Syntax: We can write the same query another way by using + Operator as below. Which returns the same result as above.

```
SELECT suppliers.supplier_id, suppliers.supplier_name, orders.order_date
FROM suppliers, orders
WHERE suppliers.supplier_id = orders.supplier_id(+);
```

RIGHT OUTER JOIN (Right join)

It returns all rows from the RIGHT-hand table specified in the ON condition and only those rows from the other table where the joined fields are equal (join condition is met).



In some databases, the RIGHT OUTER JOIN keywords are replaced with RIGHT JOIN.

The RIGHT OUTER JOIN would return the all records from table2 and only those records from table1 that intersect with table2.

supplier_id	supplier_name
10000	Apple
10001	Google

Suppliers Table

order_id	supplier_id	order_date
500125	10000	2013/08/12
500126	10001	2013/08/13
500127	10002	2013/08/14

Orders Table

```
SELECT orders.order_id, orders.order_date, suppliers.supplier_name
FROM suppliers
RIGHT OUTER JOIN orders
ON suppliers.supplier_id = orders.supplier_id;
```

If we run the above query we get the below Result

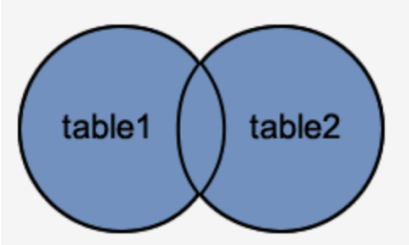
order_id	order_date	supplier_name
500125	2013/08/12	Apple
500126	2013/08/13	Google
500127	2013/08/14	<null>

Old Syntax: We can write the same query another way by using + Operator as below. Which returns the same result as above.

```
SELECT orders.order_id, orders.order_date, suppliers.supplier_name
FROM suppliers, orders
WHERE suppliers.supplier_id(+) = orders.supplier_id;
```

FULL OUTER JOIN (Full join)

It returns all rows from the LEFT-hand table and RIGHT-hand table with nulls in place where the join condition is not met.



In some databases, the FULL OUTER JOIN keywords are replaced with FULL JOIN.

supplier_id	supplier_name
10000	IBM
10001	Hewlett Packard
10002	Microsoft
10003	NVIDIA

Suppliers Table

order_id	supplier_id	order_date
500125	10000	2013/08/12
500126	10001	2013/08/13
500127	10004	2013/08/14

Orders Table

This FULL OUTER JOIN example would return all rows from the suppliers table and all rows from the orders table and whenever the join condition is not met, <nulls> would be extended to those fields in the result set.

If a supplier_id value in the suppliers table does not exist in the orders table, all fields in the orders table will display as <null> in the result set. If a supplier_id value in the orders table does not exist in the suppliers table, all fields in the suppliers table will display as <null> in the result set.

```
SELECT suppliers.supplier_id, suppliers.supplier_name, orders.order_date
FROM suppliers
FULL OUTER JOIN orders
ON suppliers.supplier_id = orders.supplier_id;
```

We get the below output for the above FULL OUTER JOIN query

supplier_id	supplier_name	order_date
10000	IBM	2013/08/12
10001	Hewlett Packard	2013/08/13
10002	Microsoft	<null>
10003	NVIDIA	<null>
<null>	<null>	2013/08/14

The rows for Microsoft and NVIDIA would be included because a FULL OUTER JOIN was used. However, you will notice that the order_date field for those records contains a <null> value.

The row for supplier_id 10004 would be also included because a FULL OUTER JOIN was used. However, you will notice that the supplier_id and supplier_name field for those records contain a <null> value.