**-- SQL Joins**

-----------------------------------------------------------------------------------------------------------------------------

/\* SQL supports different joins like INNER, LEFT, RIGHT, FULL, CROSS, and SELF JOIN. INNER returns only matches, LEFT and RIGHT include all rows from one side with NULLs for missing matches, FULL returns everything, CROSS gives Cartesian product, and SELF is used for hierarchical queries. I’ve mainly used INNER and LEFT joins in data pipelines, especially in Snowflake, when combining fact and dimension tables.

|  |  |
| --- | --- |
| **Join Type** | **What it Does** |
| **INNER JOIN** | Only matching rows |
| **LEFT JOIN** | All from left + matches from right |
| **RIGHT JOIN** | All from right + matches from left |
| **FULL JOIN** | All rows from both tables |
| **CROSS JOIN** | Cartesian product (all combinations) |
| **SELF JOIN** | Joins a table to itself |
| **NATURAL JOIN** | Joins by automatically matched column names |

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

Let's look at a selection from the "Orders" table:

OrderID CustomerID OrderDate

10308 2 1996-09-18

10309 37 1996-09-19

10310 77 1996-09-20

Then, look at a selection from the "Customers" table:

CustomerID CustomerName ContactName Country

1 Alfreds Futterkiste Maria Anders Germany

2 Ana Trujillo Emparedados y helados Ana Trujillo Mexico

3 Antonio Moreno Taquería Antonio Moreno Mexico

Notice that the "CustomerID" column in the "Orders" table refers to the "CustomerID" in the "Customers" table. The relationship between the two tables above is the "CustomerID" column.

2 Ana Trujillo Emparedados y helados Ana Trujillo Mexico 10308 2 1996-09-18

Then, we can create the following SQL statement (that contains an INNER JOIN), that selects records that have matching values in both tables:

\*/

SELECT customers.customerid,Orders.OrderID, Customers.CustomerName, Orders.OrderDate

FROM Orders

INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;

-- and it will produce something like this:

/\*

OrderID CustomerName OrderDate

10308 Ana Trujillo Emparedados y helados 9/18/1996

10365 Antonio Moreno Taquería 11/27/1996

10383 Around the Horn 12/16/1996

10355 Around the Horn 11/15/1996

10278 Berglunds snabbköp 8/12/1996

Different Types of SQL JOINs

Here are the different types of the JOINs in SQL:

(INNER) JOIN: Returns records that have matching values in both tables

LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table

RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table

FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

SELF JOIN:

CROSS JOIN:

\*/

-----------------------------------------------------------------------------------------------------------------------------

-- INNER JOIN

-----------------------------------------------------------------------------------------------------------------------------

/\*The INNER JOIN keyword selects records that have matching values in both tables.

Let's look at a selection of the Products table:

ProductID ProductName CategoryID Price

1 Chais 1 18

2 Chang 1 19

3 Aniseed Syrup 2 10

And a selection of the Categories table:

CategoryID CategoryName Description

1 Beverages Soft drinks, coffees, teas, beers, and ales

2 Condiments Sweet and savory sauces, relishes, spreads, and seasonings

3 Confections Desserts, candies, and sweet breads

We will join the Products table with the Categories table, by using the CategoryID field from both tables:\*/

-- Join Products and Categories with the INNER JOIN keyword:

SELECT ProductID, ProductName, CategoryName

FROM Products P

INNER JOIN Categories C ON P.CategoryID = C.CategoryID;

select count(\*) from Products; -- 79

select count(\*) from Categories; -- 8

select categoryid from Categories; -- 77

SELECT \*

FROM Products;

-- Note: The INNER JOIN keyword returns only rows with a match in both tables. Which means that if you have a product with no CategoryID, or with a CategoryID that is not present in the Categories table, that record would not be returned in the result.

-- Syntax

SELECT column\_name(s)

FROM table1

INNER JOIN table2

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

-- Naming the Columns

-- It is a good practice to include the table name when specifying columns in the SQL statement.

-- Specify the table names:

SELECT Products.ProductID, Products.ProductName, Categories.CategoryName

FROM Products

INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID;

-- The example above works without specifying table names, because none of the specified column names are present in both tables. If you try to include CategoryID in the SELECT statement, you will get an error if you do not specify the table name (because CategoryID is present in both tables).

-- JOIN or INNER JOIN

-- JOIN and INNER JOIN will return the same result.

-- INNER is the default join type for JOIN, so when you write JOIN the parser actually writes INNER JOIN.

-- JOIN is the same as INNER JOIN:

SELECT Products.ProductID, Products.ProductName, Categories.CategoryName

FROM Products

JOIN Categories ON Products.CategoryID = Categories.CategoryID;

/\*

JOIN Three Tables

The following SQL statement selects all orders with customer and shipper information:

Here is the Shippers table:

ShipperID ShipperName Phone

1 Speedy Express (503) 555-9831

2 United Package (503) 555-3199

3 Federal Shipping (503) 555-9931

\*/

SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName

FROM ((Orders

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)

INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);

-----------------------------------------------------------------------------------------------------------------------------

-- SQL LEFT JOIN Keyword

-----------------------------------------------------------------------------------------------------------------------------

/\*

SQL LEFT JOIN Keyword

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

LEFT JOIN Syntax

SELECT column\_name(s)

FROM table1

LEFT JOIN table2

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

Note: In some databases LEFT JOIN is called LEFT OUTER JOIN.

SQL LEFT JOIN

Below is a selection from the "Customers" table:

CustomerID CustomerName ContactName Address City PostalCode Country

1

Alfreds Futterkiste Maria Anders Obere Str. 57 Berlin 12209 Germany

2 Ana Trujillo Emparedados y helados Ana Trujillo Avda. de la Constitución 2222 México D.F. 05021 Mexico

3 Antonio Moreno Taquería Antonio Moreno Mataderos 2312 México D.F. 05023 Mexico

And a selection from the "Orders" table:

OrderID CustomerID EmployeeID OrderDate ShipperID

10308 2 7 1996-09-18 3

10309 37 3 1996-09-19 1

10310 77 8 1996-09-20 2

The following SQL statement will select all customers, and any orders they might have:

\*/

SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

ORDER BY Customers.CustomerName; -- 213

select count(\*) from customers; -- 91

select count(\*) from orders; -- 196

-- Note: The LEFT JOIN keyword returns all records from the left table (Customers), even if there are no matches in the right table (Orders).

-----------------------------------------------------------------------------------------------------------------------------

-- SQL RIGHT JOIN Keyword

-----------------------------------------------------------------------------------------------------------------------------

/\*

SQL RIGHT JOIN Keyword

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

RIGHT JOIN Syntax

SELECT column\_name(s)

FROM table1

RIGHT JOIN table2

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

Note: In some databases RIGHT JOIN is called RIGHT OUTER JOIN.

SQL RIGHT JOIN

Below is a selection from the "Orders" table:

OrderID CustomerID EmployeeID OrderDate ShipperID

10308 2 7 1996-09-18 3

10309 37 3 1996-09-19 1

10310 77 8 1996-09-20 2

And a selection from the "Employees" table:

EmployeeID LastName FirstName BirthDate Photo

1 Davolio Nancy 12/8/1968 EmpID1.pic

2 Fuller Andrew 2/19/1952 EmpID2.pic

3 Leverling Janet 8/30/1963 EmpID3.pic

SQL RIGHT JOIN Example

The following SQL statement will return all employees, and any orders they might have placed:

\*/

SELECT Orders.OrderID, Employees.LastName, Employees.FirstName

FROM Orders

RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID

ORDER BY Orders.OrderID; -- 197

-- Note: The RIGHT JOIN keyword returns all records from the right table (Employees), even if there are no matches in the left table (Orders).

-----------------------------------------------------------------------------------------------------------------------------

-- SQL FULL OUTER JOIN Keyword

-----------------------------------------------------------------------------------------------------------------------------

/\*

SQL FULL OUTER JOIN Keyword

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

Tip: FULL OUTER JOIN and FULL JOIN are the same.

FULL OUTER JOIN Syntax

SELECT column\_name(s)

FROM table1

FULL OUTER JOIN table2

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

WHERE condition;

Below is a selection from the "Customers" table:

CustomerID CustomerName ContactName Address City PostalCode Country

1

Alfreds Futterkiste Maria Anders Obere Str. 57 Berlin 12209 Germany

2 Ana Trujillo Emparedados y helados Ana Trujillo Avda. de la Constitución 2222 México D.F. 05021 Mexico

3 Antonio Moreno Taquería Antonio Moreno Mataderos 2312 México D.F. 05023 Mexico

And a selection from the "Orders" table:

OrderID CustomerID EmployeeID OrderDate ShipperID

10308 2 7 1996-09-18 3

10309 37 3 1996-09-19 1

10310 77 8 1996-09-20 2

The following SQL statement selects all customers, and all orders:

\*/

SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID

ORDER BY Customers.CustomerName; -- 213

/\*A selection from the result set may look like this:

CustomerName OrderID

Null 10309

Null 10310

Alfreds Futterkiste Null

Ana Trujillo Emparedados y helados 10308

Antonio Moreno Taquería Null

\*/

-- Note: The FULL OUTER JOIN keyword returns all matching records from both tables whether the other table matches or not. So, if there are rows in "Customers" that do not have matches in "Orders", or if there are rows in "Orders" that do not have matches in "Customers", those rows will be listed as well.

-----------------------------------------------------------------------------------------------------------------------------

-- SQL Self Join

-----------------------------------------------------------------------------------------------------------------------------

/\*

SQL Self Join

A self join is a regular join, but the table is joined with itself.

Self Join Syntax

SELECT column\_name(s)

FROM table1 T1, table1 T2

WHERE condition;

T1 and T2 are different table aliases for the same table.

Below is a selection from the "Customers" table:

CustomerID CustomerName ContactName Address City PostalCode Country

1

Alfreds Futterkiste Maria Anders Obere Str. 57 Berlin 12209 Germany

2 Ana Trujillo Emparedados y helados Ana Trujillo Avda. de la Constitución 2222 México D.F. 05021 Mexico

3 Antonio Moreno Taquería Antonio Moreno Mataderos 2312 México D.F. 05023 Mexico

The following SQL statement matches customers that are from the same city:

\*/

SELECT \*

FROM PRACTICE\_DB.SQL.PRODUCTS A

JOIN PRACTICE\_DB.SQL.PRODUCTS B

ON A.productid = B.productid;

select \* from PRACTICE\_DB.SQL.PRODUCTS;

-----------------------------------------------------------------------------------------------

-- CROSS JOIN

------------------------------------------------------------------------------------------------

select count(\*) from PRACTICE\_DB.SQL.PRODUCTS; -- 79

select count(\*) from PRACTICE\_DB.SQL.ORDERS; -- 196

select 79\*196; -- 15,484

select \* from PRACTICE\_DB.SQL.PRODUCTS

CROSS JOIN PRACTICE\_DB.SQL.ORDERS; -- 15,484