Joins in MsSQL

Joins in MSSQL are used to combine data from two or more tables based on related columns between them. Joins are used to retrieve data from multiple tables in a single query, instead of querying each table individually. There are several types of joins in MSSQL, including:

**Basic Syntax:**

SELECT column\_name(s)

FROM table1

JOIN table2

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

1. Inner Join: Returns only the rows that have matching values in both tables being joined.
2. Left Join: Returns all rows from the left table and the matching rows from the right table. If there is no match in the right table, NULL values are returned.
3. Right Join: Returns all rows from the right table and the matching rows from the left table. If there is no match in the left table, NULL values are returned.
4. Full Outer Join: Returns all rows from both tables, with NULL values in the columns where there is no match.
5. Cross Join: Returns the Cartesian product of the two tables, where each row from the first table is matched with every row from the second table.

Joins are typically specified using the JOIN keyword followed by the name of the table to be joined and the ON keyword followed by the columns used for the join.

1. Inner join : gives Matching Record in both tables

Suppose we have two tables, **orders** and **customers**. The **orders** table has columns **order\_id**, **customer\_id**, **order\_date**, and **order\_total**. The **customers** table has columns **customer\_id**, **first\_name**, **last\_name**, and **email**.

To join these tables on the **customer\_id** column and retrieve the **first\_name**, **last\_name**, **order\_date**, and **order\_total** columns, we can use the following SQL query:

create table customers(

id int primary key identity,

f\_name varchar(25),

l\_name varchar(25),

email varchar(50)

)

insert into customers values ('saqib','Ali','m4377535'),('Noman','A','b43577535@gmail.com'),('Billal','Ali','m4377535'),

('Rakham','akram','4343@gmail.com')

create table orders(

id int primary key identity,

order\_date date,

customer\_id int foreign key references customers(id)

)

insert into orders values ('2022-5-20',1),('2023-1-20',2)

insert into orders values ('2022-11-19',1),('2020-4-20',3)

select \* from orders;

select \* from customers;

--\_\_\_\_\_\_\_\_\_\_ Inner Join \_\_\_\_\_\_\_\_\_\_\_\_

select \* from orders o

inner join customers c on c.id = o.customer\_id

select (c.f\_name+' '+c.l\_name) 'FullName',o.order\_date,c.email from orders o

inner join customers c on c.id = o.customer\_id

2. Right join : gives Similar & right table data

To join these tables on the **department\_id** column and retrieve the **first\_name**, **last\_name**, and **department\_name** columns, we can use the following SQL query:

select \* from employee e

right join departments d on e.department\_id = d.department\_id

select (e.first\_name+''+e.last\_name) 'Employee Name',d.department\_name

from employee e

right join departments d on e.department\_id = d.department\_id

In the above query, the left table is **department** and the right table is **employee**.

3. left join : gives Similar & left table data.

create table student(

st\_id int primary key identity,

st\_name varchar(25),

st\_email varchar(25)

)

insert into student values ('Sajid','m4357753@gmail.com'),('Rehman','rehman@gmail.com'),('Tahir','23423@gmail.com')

create table course(

c\_id int primary key identity,

c\_name varchar(25),

c\_studentId int foreign key references student(st\_id)

)

insert into course values ('Cours1',1)

insert into course values ('Cours2',1)

insert into course values ('co1',2)

insert into course values ('co2',2)

SELECT s.st\_name, s.st\_email ,c.c\_name

FROM student s

LEFT JOIN course c ON s.st\_id= c.c\_id

In the above query, the left table is **students** and the right table is **courses**.

4. Full join : Give both right and left tables records

select \*

from employee e

full outer join departments d on e.department\_id = d.department\_id

5. Cross join : Gives all possible combinations of rows from two tables

A cross join, also known as a Cartesian product, is a type of join in which each row from one table is joined with every row from another table. This results in a result set that is the product of the two tables. A cross join does not require any matching columns between the two tables.

Here is an example of a cross join in MSSQL:

Suppose we have two tables, **colors** and **sizes**. The **colors** table has columns **color\_id** and **color\_name**. The **sizes** table has columns **size\_id** and **size\_name**.

To join these tables and retrieve all possible combinations of color and size, we can use the following SQL query:

SELECT colors.color\_name, sizes.size\_name FROM colors CROSS JOIN sizes;

In this query, we specify that we want to select the **color\_name** and **size\_name** columns from the **colors** and **sizes** tables. We use the **CROSS JOIN** keyword to specify that we want to join all possible combinations of rows from the two tables.

The result of this query will be a table that lists all possible combinations of color and size, with each row containing a color from the **colors** table and a size from the **sizes** table.

6. Self join : joins of table with its Own

A self join is a type of join in which a table is joined with itself, typically based on a foreign key relationship between two columns within the same table.

Here is an example of a self join in MSSQL:

Suppose we have a table called **employees** with columns **employee\_id**, **employee\_name**, and **manager\_id**. The **manager\_id** column contains the **employee\_id** of the employee's manager. We want to retrieve the name of each employee and their manager's name.

To achieve this, we can use a self join on the **employees** table:

SELECT e1.employee\_name, e2.employee\_name AS manager\_name FROM employees e1 INNER JOIN employees e2 ON e1.manager\_id = e2.employee\_id;

In this query, we specify that we want to select the **employee\_name** from the **employees** table, and the **employee\_name** of the employee's manager. We use the **INNER JOIN** keyword to join the **employees** table with itself based on the **manager\_id** column, joining each employee with their respective manager.