SQL JOIN

As the name shows, JOIN means *to combine something*. In case of SQL, JOIN means **"to combine two or more tables"**.

The SQL JOIN clause takes records from two or more tables in a database and combines it together.

**ANSI standard SQL** defines five types of JOIN :

1. inner join,
2. left outer join,
3. right outer join,
4. full outer join, and
5. cross join.

In the process of joining, rows of both tables are combined in a single table.

Why SQL JOIN is used?

If you want to access more than one table through a select statement.

If you want to combine two or more table then SQL JOIN statement is used .it combines rows of that tables in one table and one can retrieve the information by a SELECT statement.

The joining of two or more tables is based on common field between them.

SQL INNER JOIN also known as simple join is the most common type of join.

How to use SQL join or SQL Inner Join?

Let an example to deploy SQL JOIN process:

1.Staff table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Staff\_NAME** | **Staff\_AGE** | **STAFF\_ADDRESS** | **Monthley\_Package** |
| 1 | ARYAN | 22 | MUMBAI | 18000 |
| 2 | SUSHIL | 32 | DELHI | 20000 |
| 3 | MONTY | 25 | MOHALI | 22000 |
| 4 | AMIT | 20 | ALLAHABAD | 12000 |

2.Payment table

|  |  |  |  |
| --- | --- | --- | --- |
| **Payment\_ID** | **DATE** | **Staff\_ID** | **AMOUNT** |
| 101 | 30/12/2009 | 1 | 3000.00 |
| 102 | 22/02/2010 | 3 | 2500.00 |
| 103 | 23/02/2010 | 4 | 3500.00 |

So if you follow this JOIN statement to join these two tables ?

1. **SELECT** Staff\_ID, Staff\_NAME, Staff\_AGE, AMOUNT
2. **FROM** STAFF s, PAYMENT p
3. **WHERE** s.ID =p.STAFF\_ID;

This will produce the result like this:

|  |  |  |  |
| --- | --- | --- | --- |
| **STAFF\_ID** | **NAME** | **Staff\_AGE** | **AMOUNT** |
| 3 | MONTY | 25 | 2500 |
| 1 | ARYAN | 22 | 3000 |
| 4 | AMIT | 25 | 3500 |
| 1 | ARYAN | 22 | 3000 |

SQL OUTER JOIN

In the SQL outer JOIN all the content of the both tables are integrated together either they are matched or not.

If you take an example of employee table

**Outer join of two types:**

**1.Left outer join** (also known as left join): this join returns all the rows from left table combine with the matching rows of the right table. If you get no matching in the right table it returns NULL values.

**2.Right outer join** (also known as right join): this join returns all the rows from right table are combined with the matching rows of left table .If you get no column matching in the left table .it returns null value.

This diagram shows the different type of joins:

SQL LEFT JOIN

The SQL left join returns all the values from the left table and it also includes matching values from right table, if there are no matching join value it returns NULL.

**BASIC SYNTAX FOR LEFT JOIN:**

1. **SELECT** table1.column1, table2.column2....
2. **FROM** table1
3. LEFTJOIN table2
4. **ON** table1.column\_field = table2.column\_field;

let us take two tables in this example to elaborate all the things:

**CUSTOMER TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **SALARY** |
| 1 | ARYAN | 51 | 56000 |
| 2 | AROHI | 21 | 25000 |
| 3 | VINEET | 24 | 31000 |
| 4 | AJEET | 23 | 32000 |
| 5 | RAVI | 23 | 42000 |

This is second table

**ORDER TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **O\_ID** | **DATE** | **CUSTOMER\_ID** | **AMOUNT** |
| 001 | 20-01-2012 | 2 | 3000 |
| 002 | 12-02-2012 | 2 | 2000 |
| 003 | 22-03-2012 | 3 | 4000 |
| 004 | 11-04-2012 | 4 | 5000 |

**join these two tables with LEFT JOIN:**

1. SQL **SELECT** ID, **NAME**, AMOUNT,**DATE**
2. **FROM** CUSTOMER
3. LEFT JOIN **ORDER**
4. **ON** CUSTOMER.ID = **ORDER**.CUSTOMER\_ID;

This will produce the following result:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AMOUNT** | **DATE** |
| 1 | ARYAN | NULL | NULL |
| 2 | AROHI | 3000 | 20-01-2012 |
| 2 | AROHI | 2000 | 12-02-2012 |
| 3 | VINEET | 4000 | 22-03-2012 |
| 4 | AJEET | 5000 | 11-04-2012 |
| 5 | RAVI | NULL | NULL |

SQL RIGHT JOIN

The SQL right join returns all the values from the rows of right table. It also includes the matched values from left table but if there is no matching in both tables, it returns NULL.

Basic syntax for right join:

1. **SELECT** table1.column1, table2.column2.....
2. **FROM** table1
3. RIGHT JOIN table2
4. **ON** table1.column\_field = table2.column\_field;

let us take an example with 2 tables table1 is CUSTOMERS table and table2 is ORDERS table.

**CUSTOMER TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **SALARY** |
| 1 | ARYAN | 51 | 56000 |
| 2 | AROHI | 21 | 25000 |
| 3 | VINEET | 24 | 31000 |
| 4 | AJEET | 23 | 32000 |
| 5 | RAVI | 23 | 42000 |

and this is the second table:

**ORDER TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **O\_ID** | **CUSTOMER\_ID** | **AMOUNT** |
| 20-01-2012 | 001 | 2 | 3000 |
| 12-02-2012 | 002 | 2 | 2000 |
| 22-03-2012 | 003 | 3 | 4000 |
| 11-04-2012 | 004 | 4 | 5000 |

Here we will join these two tables with SQL RIGHT JOIN:

1. SQL> **SELECT** ID,**NAME**,AMOUNT,**DATE**
2. **FROM** CUSTOMER
3. RIGHT JOIN **ORDER**
4. **ON** CUSTOMER.ID = **ORDER**.CUSTOMER\_ID;

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AMOUNT** | **DATE** |
| 2 | AROHI | 3000 | 20-01-2012 |
| 2 | AROHI | 2000 | 12-02-2012 |
| 3 | VINEET | 4000 | 22-03-2012 |
| 4 | AJEET | 5000 | 11-04-2012 |

SQL FULL JOIN

The SQL full join is the result of combination of both left and right outer join and the join tables have all the records from both tables. It puts NULL on the place of matches not found.

SQL full outer join and SQL join are same. generally it is known as SQL FULL JOIN.

SQL full outer join:

What is SQL full outer join?

SQL full outer join is used to combine the result of both left and right outer join and returns all rows (don?t care its matched or unmatched) from the both participating tables.

**Syntax for full outer join:**

1. **SELECT** \*
2. **FROM** table1
3. **FULL** OUTER JOIN table2
4. **ON** table1.column\_name = table2.column\_name;

**Note:**here table1 and table2 are the name of the tables participating in joining and column\_name is the column of the participating tables.

Let us take two tables to demonstrate full outer join:

**table\_A**

|  |  |
| --- | --- |
| **A** | **M** |
| 1 | m |
| 2 | n |
| 4 | o |

**table\_B**

|  |  |
| --- | --- |
| **A** | **N** |
| 2 | p |
| 3 | q |
| 5 | r |

**Resulting table**

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **M** | **A** | **N** |
| 2 | n | 2 | p |
| 1 | m | - | - |
| 4 | o | - | - |
| - | - | 3 | q |
| - | - | 5 | r |

Because this is a full outer join so all rows (both matching and non-matching) from both tables are included in the output. Here only one row of output displays values in all columns because there is only one match between table\_A and table\_B.

Pictorial representation of full outer join:

    

SQL Cross Join

When each row of first table is combined with each row from the second table, known as Cartesian join or cross join. In general words we can say that SQL CROSS JOIN returns the Cartesian product of the sets of rows from the joined table.

**We can specify a CROSS JOIN in two ways:**

1. Using the JOIN syntax.
2. the table in the FROM clause without using a WHERE clause.

**SYNTAX of SQL Cross Join**

1. **SELECT** \* **FROM** [TABLE1] CROSS JOIN [TABLE2]
2. OR
3. **SELECT** \* **FROM** [ TABLE1] , [TABLE2]

Let us take an example of two tables,

**Table1 - MatchScore**

|  |  |  |
| --- | --- | --- |
| **Player** | **Department\_id** | **Goals** |
| Franklin | 1 | 2 |
| Alan | 1 | 3 |
| Priyanka | 2 | 2 |
| Rajesh | 3 | 5 |

**Table2 - Departments**

|  |  |
| --- | --- |
| **Department\_id** | **Department\_name** |
| 1 | IT |
| 2 | HR |
| 3 | Marketing |

**SQL Statement:**

1. **SELECT** \* **FROM** MatchScore CROSS JOIN Departments

**After executing this query , you will find the following result:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Player** | **Department\_id** | **Goals** | **Depatment\_id** | **Department\_name** |
| Franklin | 1 | 2 | 1 | IT |
| Alan | 1 | 3 | 1 | IT |
| Priyanka | 2 | 2 | 1 | IT |
| Rajesh | 3 | 5 | 1 | IT |
| Franklin | 1 | 2 | 2 | HR |
| Alan | 1 | 3 | 2 | HR |
| Priyanka | 2 | 2 | 2 | HR |
| Rajesh | 3 | 5 | 2 | HR |
| Franklin | 1 | 2 | 3 | Marketing |
| Alan | 1 | 3 | 3 | Marketing |
| Priyanka | 2 | 2 | 3 | Marketing |
| Rajesh | 3 | 5 | 3 | Marketing |