Oracle Joins

Join is a query that is used to combine rows from two or more tables, views, or materialized views. It retrieves data from multiple tables and creates a new table.

* Inner Joins (Simple Join)
* Outer Joins
  + Left Outer Join (Left Join)
  + Right Outer Join (Right Join)
  + Full Outer Join (Full Join)
* Equijoins
* Self Joins
* Cross Joins (Cartesian Products)
* Antijoins
* Semijoins

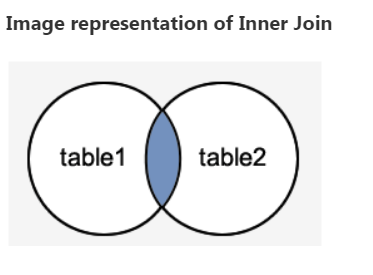
Oracle INNER JOIN

Inner Join is the simplest and most common type of join. It is also known as simple join. It returns all rows from multiple tables where the join condition is met.

**Syntax**

1. **SELECT** columns
2. **FROM** table1
3. **INNER** JOIN table2
4. **ON** table1.**column** = table2.**column**;

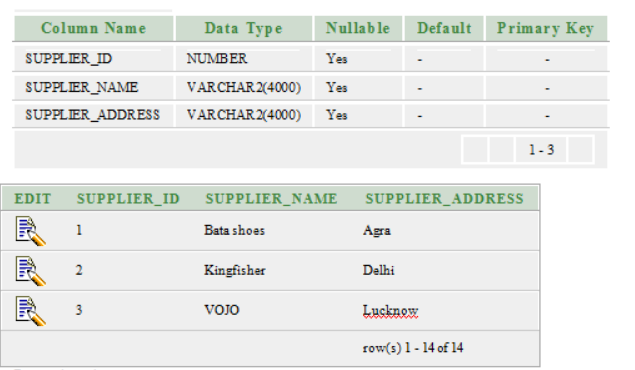
**Image representation of Inner Join**

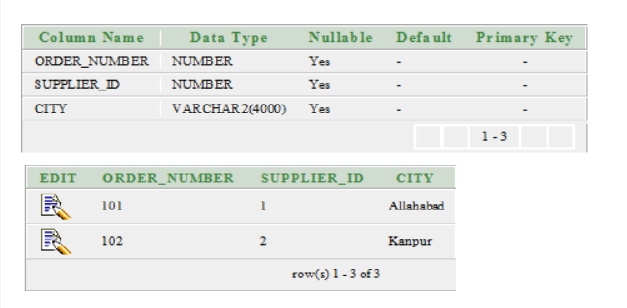


Oracle INNER JOIN Example

Let's take an example to perform Inner Join on two tables "Suppliers" and "Order1”.

**Suppliers**



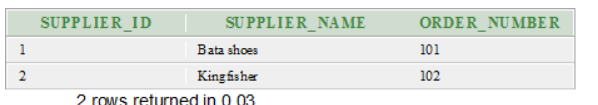
**Order1**  


his example will return all rows from "suppliers" and "order1" table where there is a matching supplier\_id value in both the suppliers and order1 tables.

Execute the following query

1. **SELECT** suppliers.supplier\_id, suppliers.supplier\_name, order1.order\_number
2. **FROM** suppliers
3. **INNER** JOIN order1
4. **ON** suppliers.supplier\_id = order1.supplier\_id;

**Output**



Oracle OUTER JOIN

An outer join is similar to equijoin but it gets also the non-matched rows from the table. It is categorized in Left Outer Join, Right Outer Join and Full Outer Join by Oracle 9i ANSI/ISO 1999 standard.

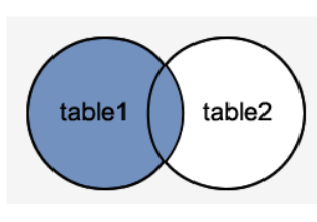
Left Outer Join

Left Outer Join returns all rows from the left (first) table specified in the ON condition and only those rows from the right (second) table where the join condition is met.

**Syntax**

1. **SELECT** columns
2. **FROM** table1
3. LEFT [OUTER] JOIN table2
4. **ON** table1.**column** = table2.**column**;

**Image representation of left outer join**



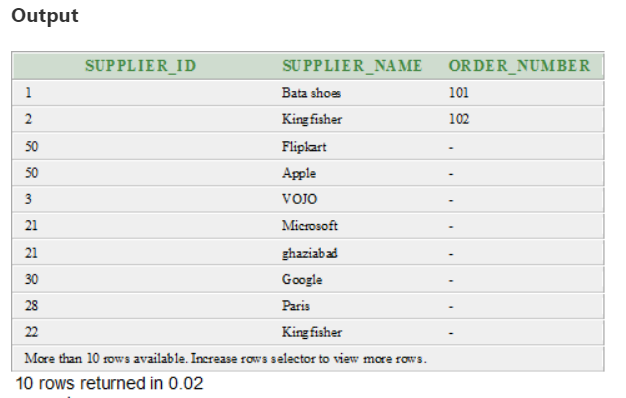
**Example**

In this example, we are performing left outer join on the already created tables ?suppliers? and ?order1?.

The following example would return all records from table ?suppliers? and only those records from table ?order1? where the join fields are equal.

**Execute this query**

1. **SELECT** suppliers.supplier\_id, suppliers.supplier\_name, order1.order\_number
2. **FROM** suppliers
3. LEFT OUTER JOIN order1
4. **ON** suppliers.supplier\_id = order1.supplier\_id;



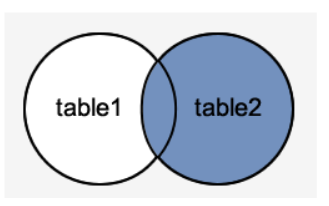
Right Outer Join

The Right Outer Join returns all rows from the right-hand table specified in the ON condition and only those rows from the other table where the join condition is met.

**Syntax**

1. **SELECT** columns
2. **FROM** table1
3. RIGHT [OUTER] JOIN table2
4. **ON** table1.**column** = table2.**column**;

**Image representation of Right Outer Join**



**Example**

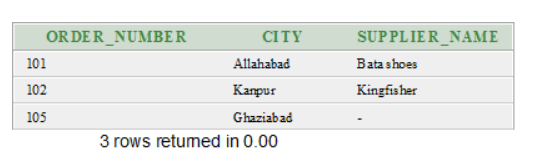
In this example, we are performing right outer join on the already created tables ?suppliers? and ?order1?.

The following example would return all rows from the order1 table and only those rows from the suppliers table where the join condition is met.

**Execute this query**

1. **SELECT** order1.order\_number, order1.city, suppliers.supplier\_name
2. **FROM** suppliers
3. RIGHT OUTER JOIN order1
4. **ON** suppliers.supplier\_id = order1.supplier\_id;

**Output**



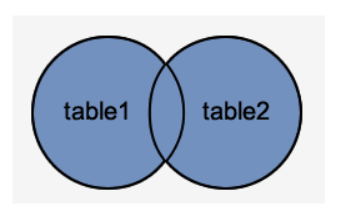
Full Outer Join

The Full Outer Join returns all rows from the left hand table and right hand table. It places NULL where the join condition is not met.

**Syntax**

1. **SELECT** columns
2. **FROM** table1
3. **FULL** [OUTER] JOIN table2
4. **ON** table1.**column** = table2.**column**;

**Image representation of Full Outer Join**



**Example**

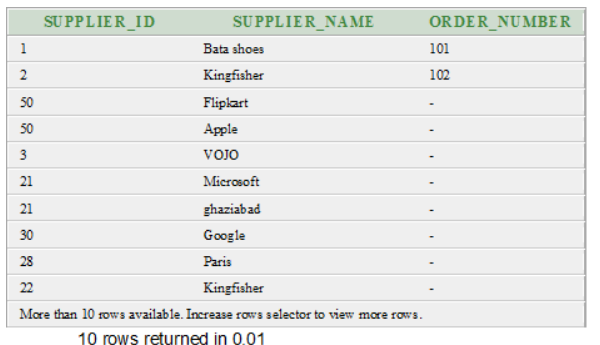
In this example, we are performing full outer join on the already created tables ?suppliers? and ?order1?.

The following example will return all rows from the ?suppliers? table and all rows from the ?order1? table and whenever the join condition is not met, it places the NULL value.

**Execute this query**

1. **SELECT** suppliers.supplier\_id, suppliers.supplier\_name, order1.order\_number
2. **FROM** suppliers
3. **FULL** OUTER JOIN order1
4. **ON** suppliers.supplier\_id = order1.supplier\_id;

**Output**



Oracle EQUI JOIN

Oracle Equi join returns the matching column values of the associated tables. It uses a comparison operator in the WHERE clause to refer equality.

**Syntax**

1. **SELECT** column\_list
2. **FROM** table1, table2....
3. **WHERE** table1.column\_name =
4. table2.column\_name;

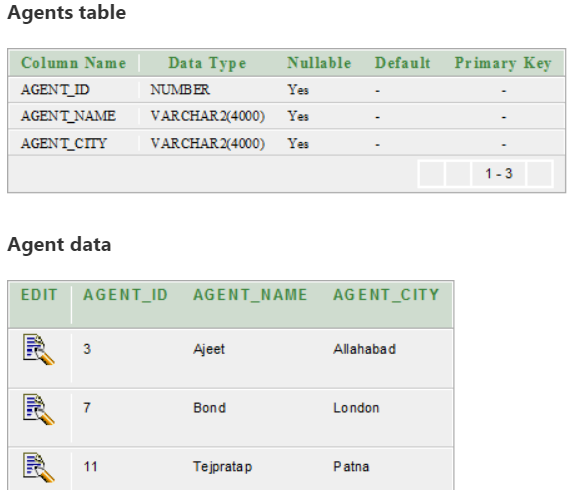
Equijoin also can be performed by using JOIN keyword followed by ON keyword and then specifying names of the columns along with their associated tables to check equality.

**Syntax**

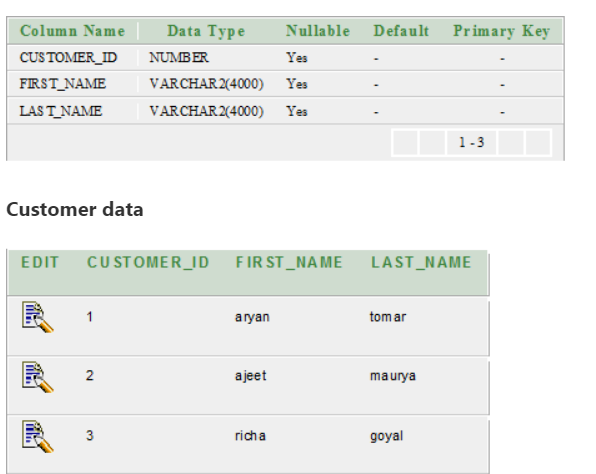
1. **SELECT** \*
2. **FROM** table1
3. JOIN table2
4. [**ON** (join\_condition)]

Oracle EQUI JOIN Example

Let' take two tables "agents" and "customer".



**Customer table**



**SQL | EQUI Join and NON EQUI JOIN**

Table name — Student

In this table, you have I’d, name, class and city are the fields.

Select \* from Student;

| **id** | **name** | **class** | **city** |
| --- | --- | --- | --- |
| 3 | Hina | 3 | Delhi |
| 4 | Megha | 2 | Delhi |
| 6 | Gouri | 2 | Delhi |
|  |  |  |  |

Table name — Record

In this table, you have I’d, class and city are the fields.

Select \* from Record;

| **id** | **class** | **city** |
| --- | --- | --- |
| 9 | 3 | Delhi |
| 10 | 2 | Delhi |
| 12 | 2 | Delhi |
|  |  |  |

**1. EQUI JOIN :**

EQUI JOIN creates a JOIN for equality or matching column(s) values of the relative tables. EQUI JOIN also create JOIN by using JOIN with ON and then providing the names of the columns with their relative tables to check equality using equal sign (=).

**Example –**

SELECT student.name, student.id, record.class, record.city

FROM student, record

WHERE student.city = record.city;

Or

**Example –**

SELECT student.name, student.id, record.class, record.city

FROM student

JOIN record

ON student.city = record.city;

**Output :**

| **name** | **id** | **class** | **city** |
| --- | --- | --- | --- |
| Hina | 3 | 3 | Delhi |
| Megha | 4 | 3 | Delhi |
| Gouri | 6 | 3 | Delhi |
| Hina | 3 | 2 | Delhi |
| Megha | 4 | 2 | Delhi |
| Gouri | 6 | 2 | Delhi |
| Hina | 3 | 2 | Delhi |
| Megha | 4 | 2 | Delhi |
| Gouri | 6 | 2 | Delhi |

**2. NON EQUI JOIN :**

NON EQUI JOIN performs a JOIN using comparison operator other than equal(=) sign like >, <, >=, <= with conditions.

**Syntax:**

SELECT \*

FROM table\_name1, table\_name2

WHERE table\_name1.column [> | < | >= | <= ] table\_name2.column;

**Example –**

SELECT student.name, record.id, record.city

FROM student, record

WHERE Student.id < Record.id ;

**Output :**

| **name** | **id** | **city** |
| --- | --- | --- |
| Hina | 9 | Delhi |
| Megha | 9 | Delhi |
| Gouri | 9 | Delhi |
| Hina | 10 | Delhi |
| Megha | 10 | Delhi |
| Gouri | 10 | Delhi |
| Hina | 12 | Delhi |
| Megha | 12 | Delhi |
| Gouri | 12 | Delhi |

Oracle SELF JOIN

Self Join is a specific type of Join. In Self Join, a table is joined with itself (Unary relationship). A self join simply specifies that each rows of a table is combined with itself and every other row of the table.

**Syntax**

1. **SELECT** a.column\_name, b.column\_name...
2. **FROM** table1 a, table1 b
3. **WHERE** a.common\_filed = b.common\_field;

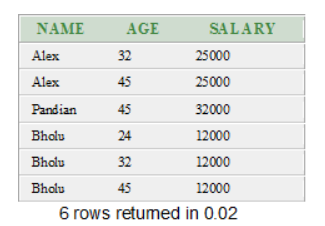
Oracle SELF JOIN Example

Let's take a table "customers".



1. **SELECT**  a.**name**, b.age, a.SALARY
2. **FROM** CUSTOMERS a, CUSTOMERS b
3. **WHERE** a.SALARY < b.SALARY;

**Output**



create table Employees (

    EmployeeID int primary key,

    Name varchar(100),

    ManagerID int null

        );

insert into Employees Values(1,'Kevin ',null)

insert into Employees Values(2,'Akshay',1)

insert into Employees Values(3,'Sandeep',2)

insert into Employees Values(4,'Swati',2)

insert into Employees Values(5,'Sunil',1)

To get the Employees List, with the manager’s name we need to join the Employees table with itself as shown below.

|  |  |
| --- | --- |
|  | Select e.EmployeeID,e.Name, manager.EmployeeID as ManagerID, manager.Name As Manager  from Employees e  left join Employees manager  on (e.ManagerID=manager.EmployeeID) |

