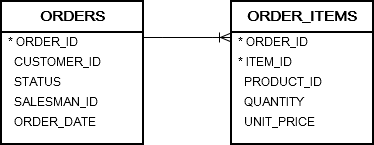
**Introduction to Oracle INNER JOIN syntax**

In a relational database, data is distributed in many related tables. For example, in the PRODUCT set of tables, the sales orders data is mainly stored in both **orders** and **order\_items** tables.



The **orders** table links to the **order\_items** table via the **order\_id** column. It means that for each row in the orders table, you can find one or more rows in the order\_items with the same values in the order\_id column.

To query data from two or more related tables, you use the **INNER JOIN** clause. The following statement illustrates how to join two tables T1 and T2.

SELECT

\*

FROM

T1

INNER JOIN T2

ON join\_predicate;

Let’s examine the statement above in detail:

* First, specify the main table in the FROM clause, T1 in this case.
* Second, specify the joined table in the INNER JOIN clause followed by a join\_predicate. The joined table is T2 in the above statement.
* Third, a join predicate specifies the condition for joining tables. Only rows that satisfy the join predicate are included in the result set.

The query returns a result set by combining column values of both tables T1 and T2 based on the join predicate. It compares each row of table T1 with rows of table T2 to find all pairs of rows that satisfy the join predicate.

Whenever the join predicate is satisfied by matching non-NULL values, column values for each matching pair of rows of T1 and T2 tables are combined into a row in the result set.

**Oracle INNER JOIN example**

The following query uses a **INNER JOIN** clause to retrieve data from the **orders** and **order\_items** tables. Word INNER is optional in the syntax.

SELECT \*

FROM

Orders o

INNER JOIN

Order\_items i ON

o.order\_id = i.order\_id

ORDER BY

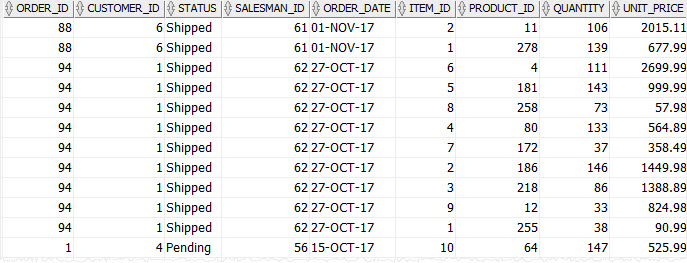
order\_date DESC;

In this example, the join predicate is

**o.order\_id = i.order\_id** and o and i are **Table Aliases**.

The query compares each row in the orders table with rows in the order\_items table. When rows from both tables have the same values in the order\_id columns, the query combines column values from rows of both tables into a result row and include it in the result set.

The following picture illustrates the partial result:



**Oracle INNER JOIN with USING clause**

Besides the ON clause, it is possible to use the **USING** clause to specify which columns to test for equality when joining tables. The following example uses the INNER JOIN with USING clause to retrieve data from **orders** and **order\_items** tables:

SELECT \*

FROM

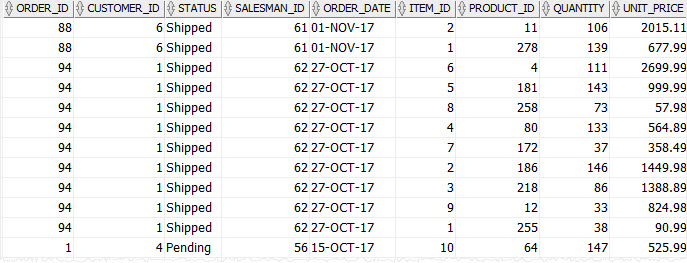
Orders

JOIN order\_items

USING( order\_id )

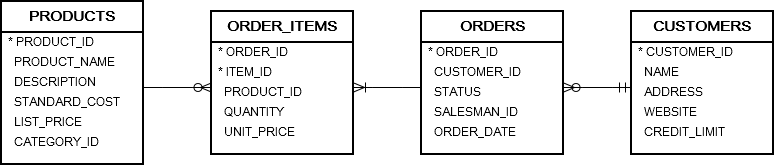
ORDER BY

order\_date DESC;



**Oracle INNER JOIN – joining multiple tables**

Consider the following tables **orders,  order\_items, customers,** and **products** from the PRODUCT set.



The inner join clause can join more than two tables. In practice, you should limit the number of joined tables to avoid the performance issue.  The following statement shows how to join three tables: orders, order\_items, and customers.

SELECT

name AS customer\_name,

order\_id,

order\_date,

item\_id,

quantity,

unit\_price

FROM

orders

JOIN order\_items USING(order\_id)

JOIN customers USING(customer\_id)

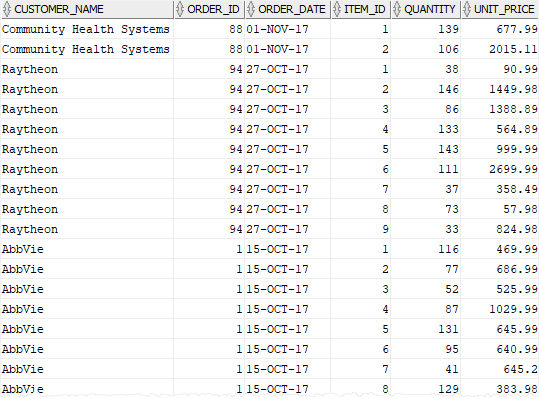
ORDER BY

order\_date DESC,

order\_id DESC,

item\_id ASC;

Here is the partial result set:



The following example illustrates how to join four tables: orders,  order\_items, customers, and products. It is followed by the partial result set;

SELECT

name AS customer\_name,

order\_id,

order\_date,

item\_id,

product\_name,

quantity,

unit\_price

FROM

orders

JOIN order\_items

USING(order\_id)

JOIN customers

USING(customer\_id)

JOIN products

USING(product\_id)

ORDER BY

order\_date DESC,

order\_id DESC,

item\_id ASC;

