

SQL Introduction

>

SQL SELECT (I)

>

SQL SELECT (II)

⌵

✓ SQL ORDER BY

✓ SQL GROUP BY

✓ SQL LIKE

✓ SQL Wildcards

✓ SQL UNION

✓ SQL Subquery

○ SQL ANY and ALL

✓ SQL CASE

✓ SQL HAVING

✓ SQL EXISTS

SQL JOIN

>

SQL DATABASE & TABLE

>

SQL Insert, Update and Delete

>

SQL Constraints

>

SQL Additional Topics

>

Related Topics

SQL IN Operator

SQL EXISTS

SQL JOIN

SQL FULL OUTER JOIN

SQL RIGHT JOIN

SQL SELECT INTO Statement

## SQL Subquery

In this tutorial, we'll learn about subqueries in SQL with the help of examples.

In SQL, it's possible to place a SQL query inside another query known as subquery. For example,

```
SELECT *
FROM Customers
WHERE age = (
  SELECT MIN(age)
  FROM Customers
);
```

Run Code >>

In a subquery, the outer query's result is dependent on the result-set of the inner subquery. That's why subqueries are also called nested queries.

Here, the SQL command

- executes the subquery first; selects minimum `age` from the `Customers` table.
- executes the outer query; selects rows where `age` is **equal to the result of subquery**.



### Example 2: SQL Subquery

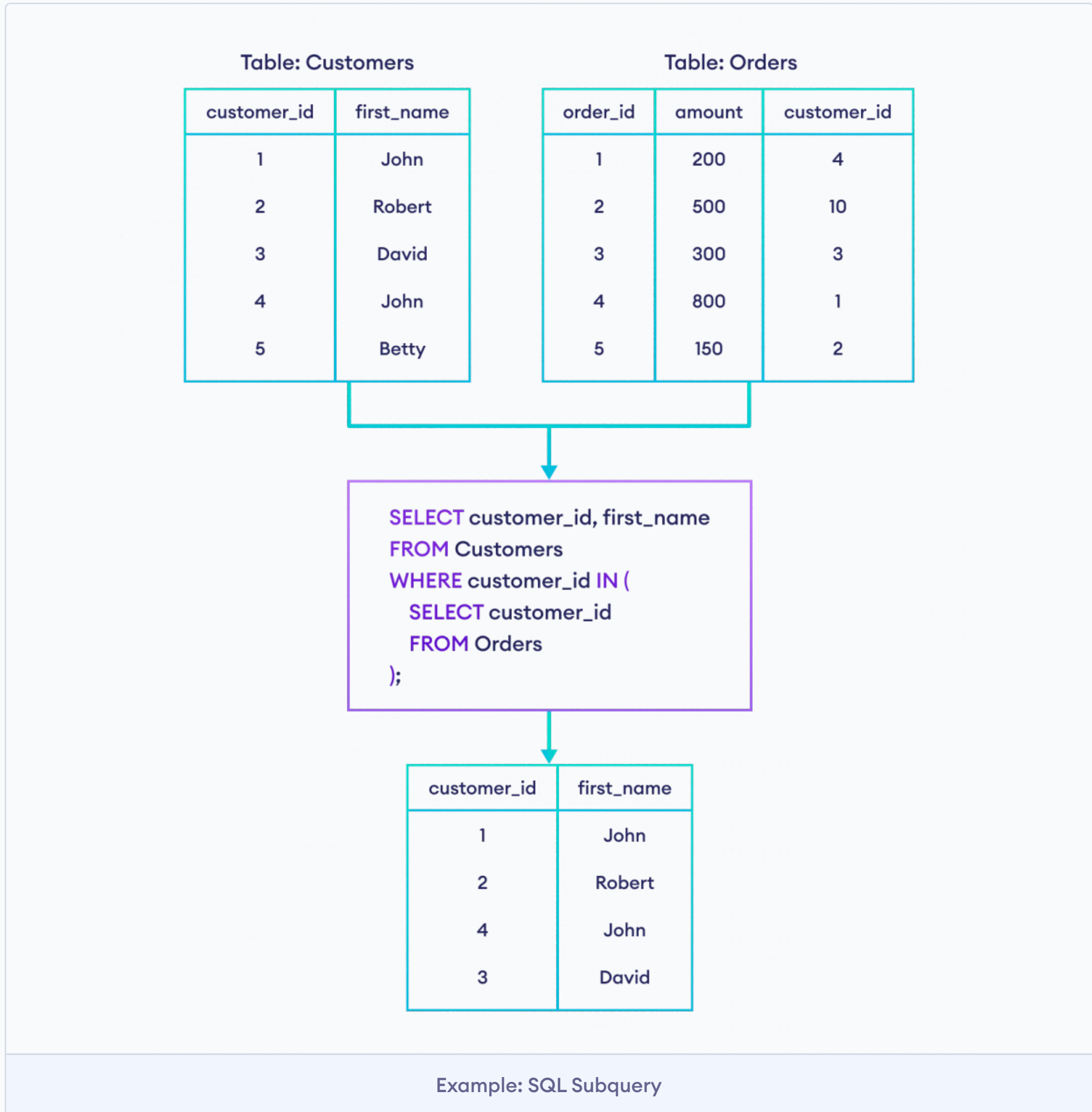
Suppose we want details of **customers** who have placed an **order**. Here's how we can do that using a subquery:

```
SELECT customer_id, first_name
FROM Customers
WHERE customer_id IN (
  SELECT customer_id
  FROM Orders
);
```

Run Code >>

Here, the SQL command

- selects `customer_id` from `Orders` table
- select rows from `Customers` table where `customer_id` is in the result set of subquery



### SQL Subquery and JOIN

In some scenarios, we can get the same result set using a subquery and **the JOIN clause**. For example,

The result set of this query

```
SELECT DISTINCT Customers.customer_id, Customers.first_name
FROM Customers
INNER JOIN Orders
ON Customers.customer_id = Orders.customer_id
ORDER BY Customers.customer_id;
```

Run Code >>

will be the same as

```
SELECT customer_id, first_name
FROM Customers
WHERE customer_id IN (
  SELECT customer_id
  FROM Orders
);
```

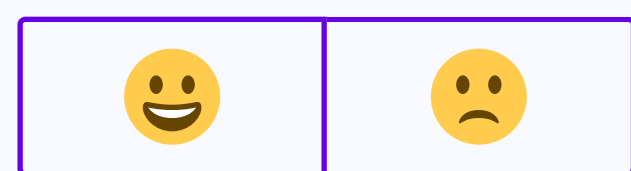
Run Code >>

**Note:** It's preferred to use the `JOIN` clause instead of a subquery whenever possible. It's because the execution speed of `JOIN` is faster and more optimized than a subquery.

Previous Tutorial:  
**SQL UNION**

Next Tutorial:  
**SQL ANY and ALL** →

Did you find this article helpful?



#### Related Tutorials

Programming

SQL IN Operator

Programming

SQL EXISTS

Programming

SQL JOIN

Programming

SQL FULL OUTER JOIN

Programiz



#### Tutorials

- Python 3 Tutorial
- JavaScript Tutorial
- SQL Tutorial
- C Tutorial
- Java Tutorial
- Kotlin Tutorial
- C++ Tutorial
- Swift Tutorial
- C# Tutorial
- Go Tutorial
- DSA Tutorial

#### Examples

- Python Examples
- JavaScript Examples
- C Examples
- Java Examples
- Kotlin Examples
- C++ Examples

#### Company

- About
- Advertising
- Privacy Policy
- Terms & Conditions
- Contact
- Blog
- Youtube

#### Apps

- Learn Python
- Learn C Programming
- Learn Java