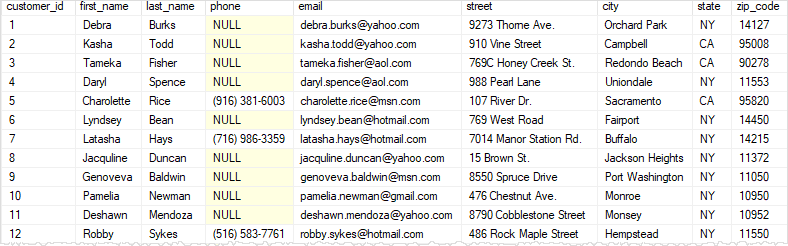
**Basic sales.customers Table**



1. **SQL Server SELECT – retrieve some columns of a table example**

**SELECT**

**first\_name,**

**last\_name**

**FROM**

**sales.customers;**



**SELECT**

**first\_name,**

**last\_name,**

**email**

**FROM**

**sales.customers;**



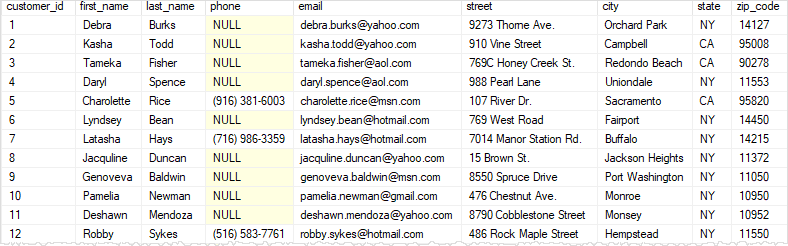
1. **SQL Server SELECT – retrieve all columns from a table example**

**SELECT**

**\***

**FROM**

**sales.customers;**



1. **SQL Server SELECT – sort the result set**

**SELECT**

**\***

**FROM**

**Sales.customers**

**WHERE**

**state = 'CA';**



**SELECT**

**\***

**FROM**

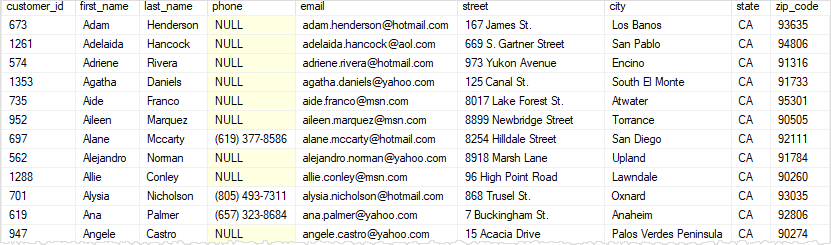
**sales.customers**

**WHERE**

**state = 'CA'**

**ORDER BY**

**first\_name;**



**Introduction to the SQL Server ORDER BY clause**

When you use the [SELECT](https://www.sqlservertutorial.net/sql-server-basics/sql-server-select/) statement to query data from a table, the order of rows in the result set is not guaranteed. It means that SQL Server can return a result set with an unspecified order of rows.

**SELECT**

**first\_name,**

**last\_name**

**FROM**

**sales.customers**

**ORDER BY**

**first\_name;**



Sort a result set by one column in descending order

**SELECT**

**firstname,**

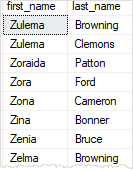
**lastname**

**FROM**

**sales.customers**

**ORDER BY**

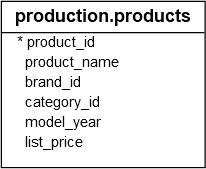
**first\_name DESC;**



## **Introduction to SQL Server WHERE clause**

## **SQL Server WHERE examples**

We will use the production.products table from the [sample database](https://www.sqlservertutorial.net/sql-server-sample-database/) for the demonstration.



### A) Finding rows by using a simple equality

**SELECT**

**product\_id,**

**product\_name,**

**category\_id,**

**model\_year,**

**list\_price**

**FROM**

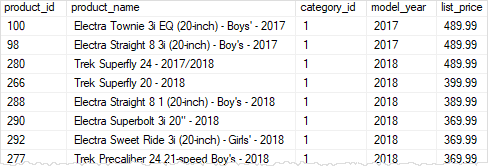
**Production.products**

**WHERE**

**category\_id = 1**

**ORDER BY**

**list\_price DESC;**



### Finding rows that meet two conditions

### SELECT

### product\_id,

### product\_name,

### category\_id,

### model\_year,

### list\_price

### FROM

### production.products

### WHERE

### category\_id = 1 AND model\_year = 2018

### ORDER BY

### list\_price DESC;

### SQL Server WHERE - match two conditions

### Finding rows with the value between two values

### SELECT

### product\_id,

### product\_name,

### category\_id,

### model\_year,

### list\_price

### FROM

### production.products

### WHERE

### list\_price BETWEEN 1899.00 AND 1999.99

### ORDER BY

### list\_price DESC;

### SQL Server WHERE - between operator

### Finding rows that have a value in a list of values

**SELECT**

**product\_id,**

**product\_name,**

**category\_id,**

**model\_year,**

**list\_price**

**FROM**

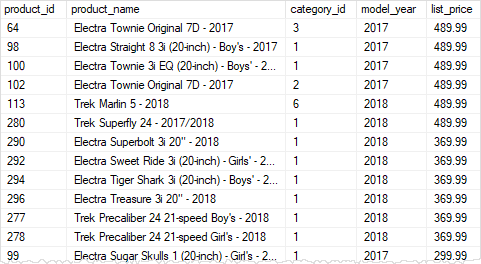
**production.products**

**WHERE**

**list\_price IN (299.99, 369.99, 489.99)**

**ORDER BY**

**list\_price DESC;**



### Finding rows whose values contain a string

**SELECT**

**product\_id,**

**product\_name,**

**category\_id,**

**model\_year,**

**list\_price**

**FROM**

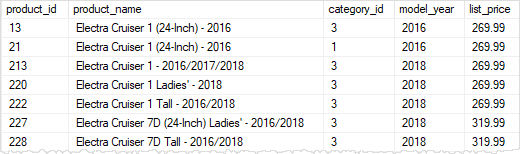
**production.products**

**WHERE**

**product\_name LIKE '%Cruiser%'**

**ORDER BY**

**list\_price;**



## **Introduction to SQL Server AND operator**

### Using AND operator example

**SELECT**

**\***

**FROM**

**production.products**

**WHERE**

**category\_id = 1**

**AND list\_price > 400**

**ORDER BY**

**list\_price DESC;**



### Using multiple AND operators example

**SELECT**

**\***

**FROM**

**production.products**

**WHERE**

**category\_id = 1**

**AND list\_price > 400**

**AND brand\_id = 1**

**ORDER BY**

**list\_price DESC;**

SQL Server AND multiple operators example

### Using the AND operator with other logical operators

**SELECT**

**\***

**FROM**

**production.products**

**WHERE**

**brand\_id = 1**

**OR brand\_id = 2**

**AND list\_price > 1000**

**ORDER BY**

**brand\_id DESC;**



## **Introduction to SQL Server OR operator**

**SELECT**

**product\_name,**

**list\_price**

**FROM**

**production.products**

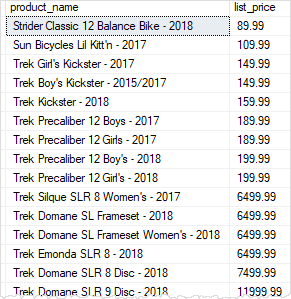
**WHERE**

**list\_price < 200**

**OR list\_price > 6000**

**ORDER BY**

**list\_price;**



**SELECT**

**product\_name,**

**brand\_id**

**FROM**

**production.products**

**WHERE**

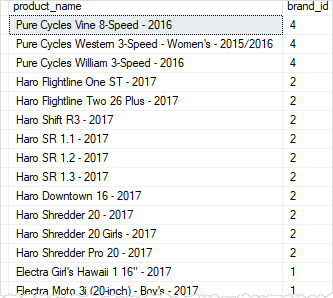
**brand\_id = 1**

**OR brand\_id = 2**

**OR brand\_id = 4**

**ORDER BY**

**brand\_id DESC;**



### Using OR operator with AND operator example

**SELECT**

**product\_name,**

**brand\_id,**

**list\_price**

**FROM**

**production.products**

**WHERE**

**brand\_id = 1**

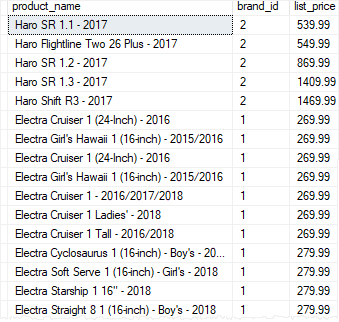
**OR brand\_id = 2**

**AND list\_price > 500**

**ORDER BY**

**brand\_id DESC,**

**list\_price;**



## **SQL Server IN operator overview**

### Using SQL Server IN with a list of values example

**SELECT**

**product\_name,**

**list\_price**

**FROM production.products**

**WHERE**

**list\_price IN (89.99, 109.99, 159.99)**

**ORDER BY**

**list\_price;**



## **Overview of the SQL Server BETWEEN operator**

**SELECT**

**product\_id,**

**product\_name,**

**list\_price**

**FROM**

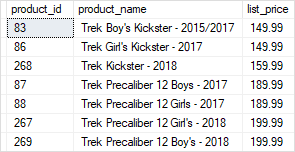
**production.products**

**WHERE**

**list\_price BETWEEN 149.99 AND 199.99**

**ORDER BY**

**list\_price;**



**CREATE TABLE hr.candidates(**

**id INT PRIMARY KEY IDENTITY,**

**fullname VARCHAR(100) NOT NULL**

**);**

**CREATE TABLE hr.employees(**

**id INT PRIMARY KEY IDENTITY,**

**fullname VARCHAR(100) NOT NULL**

**);**

**INSERT INTO**

**hr.candidates(fullname)**

**VALUES**

**('John Doe'),**

**('Lily Bush'),**

**('Peter Drucker'),**

**('Jane Doe');**

**INSERT INTO**

**hr.employees(fullname)**

**VALUES**

**('John Doe'),**

**('Jane Doe'),**

**('Michael Scott'),**

**('Jack Sparrow');**

## **SQL Server Inner Join**

**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

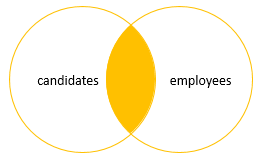
**e.fullname employee\_name**

**FROM**

**hr.candidates c**

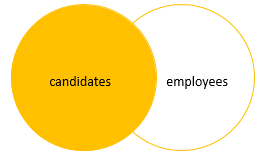
**INNER JOIN hr.employees e**

**ON e.fullname = c.fullname;**



SQL Server Joins - Inner Join

## **SQL Server Left Join**



**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

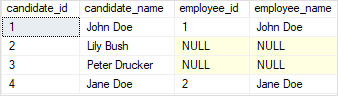
**e.fullname employee\_name**

**FROM**

**hr.candidates c**

**LEFT JOIN hr.employees e**

**ON e.fullname = c.fullname;**



To get the rows that are available only in the left table but not in the right table, you add a WHERE clause to the above query:

**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

**e.fullname employee\_name**

**FROM**

**hr.candidates c**

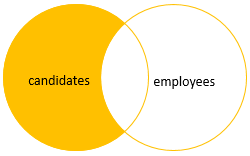
**LEFT JOIN hr.employees e**

**ON e.fullname = c.fullname**

**WHERE**

**e.id IS NULL;**

SQL Server Joins - left Join with a where clause



## **SQL Server Right Join**

**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

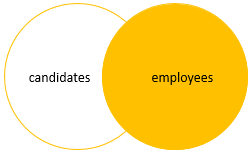
**e.fullname employee\_name**

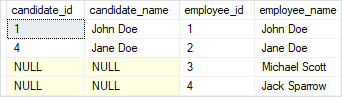
**FROM**

**hr.candidates c**

**RIGHT JOIN hr.employees e**

**ON e.fullname = c.fullname;**





Similarly, you can get rows that are available only in the right table by adding a [WHERE](https://www.sqlservertutorial.net/sql-server-basics/sql-server-where/) clause to the above query as follows:

**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

**e.fullname employee\_name**

**FROM**

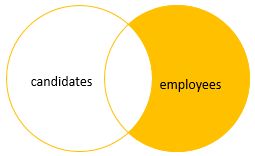
**hr.candidates c**

**RIGHT JOIN hr.employees e**

**ON e.fullname = c.fullname**

**WHERE**

**c.id IS NULL;**



SQL Server Joins - right Join with a where clause

## **SQL Server full join**

**SELECT**

**c.id candidate\_id,**

**c.fullname candidate\_name,**

**e.id employee\_id,**

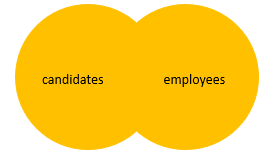
**e.fullname employee\_name**

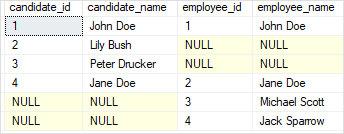
**FROM**

**hr.candidates c**

**FULL JOIN hr.employees e**

**ON e.fullname = c.fullname;**





## **Introduction to SQL Server GROUP BY clause**

The GROUP BY clause allows you to arrange the rows of a [query](https://www.sqlservertutorial.net/sql-server-basics/sql-server-select/) in groups. The groups are determined by the columns that you specify in the GROUP BY clause.

**SELECT**

**customer\_id,**

**YEAR (order\_date) order\_year**

**FROM**

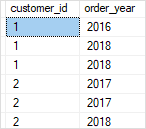
**sales.orders**

**WHERE**

**customer\_id IN (1, 2)**

**ORDER BY**

**customer\_id;**



**SELECT**

**customer\_id,**

**YEAR (order\_date) order\_year**

**FROM**

**sales.orders**

**WHERE**

**customer\_id IN (1, 2)**

**GROUP BY**

**customer\_id,**

**YEAR (order\_date)**

**ORDER BY**

**customer\_id;**

