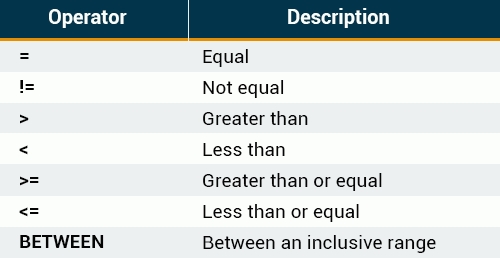
**The WHERE Statement**

The **WHERE**clause is used to extract only those records that fulfill a specified criterion.  
**The syntax for the WHERE clause:**

SELECT column\_list   
FROM table\_name   
**WHERE** condition;

**SQL Operators**

**Comparison Operators**and **Logical Operators** are used in the WHERE clause to filter the data to be selected.



**The BETWEEN Operator**

The BETWEEN operator selects values within a range. The first value must be lower bound and the second value, the upper bound.  
  
**The syntax for the BETWEEN clause is as follows:**

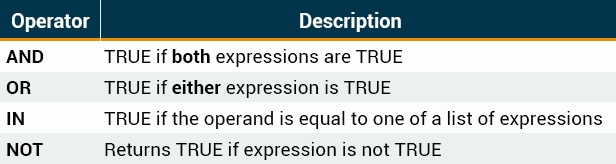
SELECT column\_name(s)   
FROM table\_name   
WHERE column\_name **BETWEEN** value1 **AND** value2;

# **Text Values**

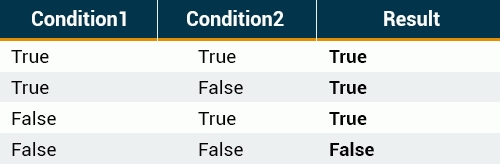
When working with text columns, surround any text that appears in the statement with**single quotation marks (')**.  
  
The following SQL statement selects all records in which the *City*is equal to 'New York'.

SELECT ID, FirstName, LastName, City   
FROM customers   
WHERE City = **'New York'**;

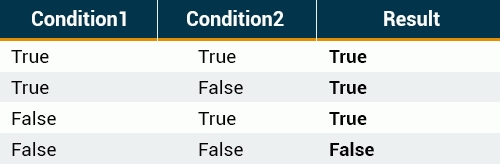
**Logical Operators**

Logical operators can be used to combine two Boolean values and return a result of **true**, **false**, or **null**.  
**The following operators can be used:**When retrieving data using a SELECT statement, use logical operators in the WHERE clause to combine multiple conditions.  
  
If you want to select rows that satisfy all of the given conditions, use the logical operator, **AND**.

**OR**

If you want to select rows that satisfy at least one of the given conditions, you can use the logical **OR**operator.  
  
**The following table describes how the logical OR operator functions:**

**OR**

If you want to select rows that satisfy at least one of the given conditions, you can use the logical **OR**operator.  
  
**The following table describes how the logical OR operator functions:**

**Combining AND & OR**

The SQL **AND**and **OR**conditions may be combined to test multiple conditions in a query.  
These two operators are called **conjunctive operators**.  
  
When combining these conditions, it is important to use **parentheses**, so that the order to evaluate each condition is known.

**The IN Operator**

The **IN**operator is used when you want to compare a column with more than one value.  
  
For example, you might need to select all customers from New York, Los Angeles, and Chicago.  
With the **OR**condition, your SQL would look like this:

**The IN Operator**

You can achieve the same result with a single IN condition, instead of the multiple **OR** conditions:

SELECT \* FROM customers   
WHERE City **IN ('New York', 'Los Angeles', 'Chicago')**;

**The NOT IN Operator**

The **NOT IN** operator allows you to exclude a list of specific values from the result set.  
  
If we add the **NOT**keyword before **IN**in our previous query, customers living in those cities will be excluded:

SELECT \* FROM customers   
WHERE City **NOT IN** ('New York', 'Los Angeles', 'Chicago');

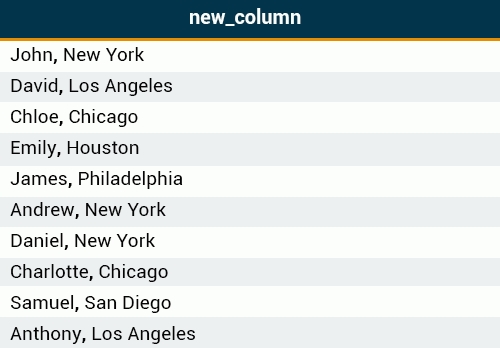
**The CONCAT Function**

The **CONCAT**function is used to concatenate two or more text values and returns the concatenating string.  
  
**Let's concatenate the *FirstName*with the *City*, separating them with a *comma*:**

**The AS Keyword**

A concatenation results in a new column. The default column name will be the CONCAT function.  
You can assign a custom name to the resulting column using the **AS**keyword:

SELECT CONCAT(FirstName,', ', City) **AS** new\_column   
FROM customers;

And when you run the query, the column name appears to be changed.

A concatenation results in a new column.

**Arithmetic Operators**

Arithmetic operators perform arithmetical operations on numeric operands. The Arithmetic operators include addition (+), subtraction (-), multiplication (\*) and division (/).

Parentheses can be used to force an operation to take priority over any other operators. They are also used to improve code readability.

**The UPPER Function**

The **UPPER**function converts all letters in the specified string to uppercase.  
The **LOWER**function converts the string to lowercase.  
  
**The following SQL query selects all *LastNames*as uppercase:**

SELECT FirstName, **UPPER(LastName)** AS LastName   
FROM employees;

https://downloadly.net/2020/09/1041/03/full-stack-angular-and-spring-boot/12/?#/1041-udemy-092135024728.html