SQL for Citizen Data Scientists

SQL LIKE Operator - Learn By Example

SETScholars & WACAMLDS

Until now you have been using operators such as conditional operators (for testing equality), BETWEEN operator (for checking against a range of values) or IN operator (for checking against a set of values) in which one thing is common; They all search for an exact match.

Filtering data that way is all well and good. But sometimes a situation can arise when you want to search for partial string matches, for example, you want to search for all products that include a text 'Chocolate Cookie' within their name. You cannot achieve this with simple comparison operators; This is where you need a wildcard search.

Wildcards are special characters with which you can create search patterns that can be compared against your data.

The LIKE Operator

To use wildcards in search clauses, the LIKE operator is used. LIKE instructs the DBMS to search using a wildcard match rather than a straight equality match.

The LIKE operator has the following syntax:

```
SELECT column_name(s)
FROM table_name
WHERE column_name LIKE search_pattern;
```

The search_pattern is a sequence of characters to search for in the column. It is made up of literal text, wildcard characters, or a combination of both.

The Following wildcard characters are used to create a search pattern:

Wildcard character	Description
%	The percent sign matches any string of zero or more characters.
_	The underscore matches a single character.

Let's take a look at them one by one.

Sample Table

To help you better understand the examples, and enable you to follow along with the tutorial, we are going to use the following sample table.

This table is part of an 'Product Management System' used by an imaginary distributor of cookies.

ID	Name	Price ($)
1	Vanilla Cookies	3.00
2	Cashew Nut Cookies	3.00
3	Chocolate Cookies	4.00
4	Chocolate Orange Cookies	5.00
5	White Chocolate Cookies	6.00
6	Dark Chocolate Chip Cookies	6.00
7	Double Chocolate Chip Cookies	7.00
8	4 Ct. Cookies Variety Pack	6.00
9	8 Ct. Cookies Variety Pack	9.00
10	12 Ct. Cookies Variety Pack	12.00

The Percent Sign % Wildcard

The percent sign `%` is the most commonly used wildcard. `%` Matches zero, one or more characters at a specified location within a search string. For example, if you wanted to search for all products starting with the word 'Chocolate', you could write a query like:

```
SELECT *
FROM Products
WHERE Name LIKE 'Chocolate%';
```

ID	Name	Price ($)
3	Chocolate Cookies	4.00
4	Chocolate Orange Cookies	5.00

In this example the search pattern `Chocolate%` tells the DBMS to match any value starting with the text 'Chocolate', regardless of any character after it.

You can use several wildcards to create a search pattern. The following example uses two wildcards, one at either end:

```
SELECT *
FROM Products
WHERE Name LIKE '%Chocolate%';
```

ID	Name	Price ($)
3	Chocolate Cookies	4.00
4	Chocolate Orange Cookies	5.00
5	White Chocolate Cookies	6.00
6	Dark Chocolate Chip Cookies	6.00
7	Double Chocolate Chip Cookies	7.00

The above example uses the search pattern `%Chocolate%` which matches any value that contains the text 'Chocolate' anywhere within it, regardless of any characters before or after that text. It is important to note that `%` also matches zero characters; that's why the result-set also includes products whose name starts with 'Chocolate'.

Wildcards can be used anywhere within the search pattern, including in the middle, although this is rarely useful. The following example finds all products that start with a 'D' and end with 'Cookies'.

```
SELECT *
FROM Products
WHERE Name LIKE 'D%Cookies';
```

ID	Name	Price ($)
6	Dark Chocolate Chip Cookies	6.00
7	Double Chocolate Chip Cookies	7.00

The Underscore _ Wildcard

Like the percent sign `%`, the underscore `_` is another wildcard character that is used extensively. The underscore is used the same way as the `%`, but the underscore matches only one character, rather than matching multiple characters.

Below example specifies the search pattern containing a single underscore wildcard followed by literal text.

```
SELECT *
FROM Products
WHERE Name LIKE '_ Ct. Cookies Variety Pack';
```

ID	Name	Price ($)
8	4 Ct. Cookies Variety Pack	6.00
9	8 Ct. Cookies Variety Pack	9.00

As you can see only two rows are retrieved as the underscore matches 4 in the first row and 8 in the second row. But the '12 Ct. Cookies Variety Pack' did not match because the search pattern required only one wildcard match, not two.

More Examples

Below table shows some more search expressions and their interpretation.

Search expression	Interpretation
B%	Finds any values that start with 'B'
%t	Finds any values that ends with 't'
%bas%	Finds any values that have 'bas' in any position
_ir%	Finds any values that have 'ir' in the second and third positions
A__%	Finds any values that start with 'A' and are at least three characters in length
%0	Finds any values that end with '0'
_c%1	Finds any values that have a 'c' in the second position and end with a '1'
2___3	Finds any values in a five-digit number that start with '2' and end with '3'
a%o	Finds any values that start with 'a' and ends with 'o'
__t_	Finds any four-character string with a 't' in the third position
___-__-____	Finds any 11-character string with dashes in the fourth and seventh positions

The NOT LIKE Operator

The LIKE operator is negated as NOT LIKE. When NOT LIKE is used, only values that are not similar are returned. For example, if you wanted to search for all products that do not have the word 'chocolate' anywhere in their name, you could write a query like:

```
SELECT *
FROM Products
WHERE Name NOT LIKE '%Chocolate%';
```

ID	Name	Price ($)
1	Vanilla Cookies	3.00
2	Cashew Nut Cookies	3.00
8	4 Ct. Cookies Variety Pack	6.00
9	8 Ct. Cookies Variety Pack	9.00
10	12 Ct. Cookies Variety Pack	12.00