MySQL Wildcards Tutorial: Like, NOT Like, Escape, (%), (_)

What are MySQL Wildcards?

MySQL Wildcards are characters that help search data matching complex criteria. Wildcards are used in conjunction with the LIKE comparison operator or with the NOT LIKE comparison operator.

Why use WildCards ?

If you are familiar with using SQL, you may think that you can search for any complex data using the SELECT and WHERE clause . Then why use Wildcards ?

Before we answer that question, let's look at an example. Suppose that the marketing department of Myflix video library carried out marketing promotions in the city of Texas and would like to get some feedback on the number of members

that registered from Texas, you can use the following SELECT statement together with the WHERE clause to get the desired information.

```
SELECT * FROM members WHERE postal_address = 'Austin ,
TX' OR  postal_address = Dallas , TX OR postal_address =
Iola,TX OR postal_adress = Houston ,TX';
```

As you can see from the above query, the "WHERE clause" becomes complex. Using wildcards however, simplifies the query as we can use something simple like the script shown below.

```
SELECT * FROM  members  WHERE postal_address  like '%
TX';
```

In short, wildcards allow us to develop power search engines into our data driven applications.

Types of wildcards

% the percentage

% the percentage character is used to specify a pattern of **zero (0) or more characters**. It has the following basic syntax.

```
SELECT statements... WHERE fieldname LIKE 'xxx%';
```

HERE

- "SELECT statement..." is the standard SQL SELECT command.
- "WHERE" is the key word used to apply the filter.
- "LIKE" is the comparison operator that is used in conjunction with wildcards
- 'xxx' is any specified starting pattern such as a single character or more and "%" matches any number of characters starting from zero (0).

To fully appreciate the above statement, let's look at a practical example

Suppose we want to get all the movies that have the word "code" as part of the title, we would use the percentage wildcard to perform a pattern match on both sides of the word "code". Below is the SQL statement that can be used to achieve the desired results.

```
SELECT * FROM movies WHERE title LIKE '%code%';
```

Executing the above script in MySQL workbench against the myflixdb gives us the results shown below.

movie_id	**title**	**director**	**year_released**	**category_id**

4	Code Name Black	Edgar Jimz	2010	NULL
7	Davinci Code	NULL	NULL	6

Notice that even if the search key word "code" appears on the beginning or end of the title, it is still returned in our result set. This is because our code includes any number of characters at the beginning then matches the pattern "code" followed by any number of characters at the end.

Let's now modify our above script to include the percentage wildcard at the beginning of the search criteria only.

```
SELECT * FROM movies WHERE title LIKE '%code';
```

Executing the above script in MySQL workbench against the myflixdb gives us the results shown below.

movie_id	**title**	**director**	**year_released**	**category_id**
7	Davinci Code	NULL	NULL	6

Notice that only one record has been returned from the database. This is because our code matches any number of characters at the beginning of the movie title and gets only records that end with the pattern "code".

Let's now shift the percentage wildcard to the end of the specified pattern to be matched. The modified script is shown below.

```
SELECT * FROM movies WHERE title LIKE 'code%';
```

Executing the above script in MySQL workbench against the myflixdb gives us the results shown below.

movie_id	**title**	**director**	**year_released**	**category_id**

4	Code Name Black	Edgar Jimz	2010	NULL

Notice only one record has been returned from the database. This is because our code matches all titles that start with the pattern "code" followed by any number of characters.

_ underscore wildcard

The underscore wildcard is used to **match exactly one character**. Let's suppose that we want to search for all the movies that were released in the years 200x where x is exactly one character that could be of any value. We would use the underscore wild card to achieve that. The script below select all the movies that were released in the year "200x"

```
SELECT * FROM movies WHERE year_released LIKE '200_';
```

Executing the above script in MySQL workbench against the myflixdb gives us the results shown below.

movie_id	title	director	year_released	category_id
2	Forgetting Sarah Marshall	Nicholas Stoller	2008	2
9	Honey mooners	Jhon Shultz	2005	8

Notice that only movies that have 200 followers by any character in the field year have been returned in our result set. This is because the underscore wildcard matched the pattern 200 followed by any single character

NOT Like

The NOT logical operator can be used together with the wildcards to return rows that do not match the specified pattern.

Suppose we want to get movies that were not released in the year 200x. We would use the NOT logical operator together with the underscore wildcard to get our results. Below is the script that does that.

```
SELECT * FROM movies WHERE year_released NOT LIKE '200_';
```

movie_id	**title**	**director**	**year_released**	**category_id**
1	Pirates of the Caribbean 4	Rob Marshall	2011	1
4	Code Name Black	Edgar Jimz	2010	NULL
8	Underworld-Awakeninh	Michahel Eal	2012	6

Notice only movies that do not start with 200 in the year released have been returned in our result set. This is because we used the NOT logical operator in our wildcard pattern search.

Escape keyword.

The ESCAPE keyword is used to **escape pattern matching characters** such as the (%) percentage and underscore (_) if they form part of the data.

Let's suppose that we want to check for the string "67%" we can use;

```
LIKE '67#%%' ESCAPE '#';
```

If we want to search for the movie "67% Guilty", we can use the script shown below to do that.

```
SELECT * FROM movies WHERE title LIKE '67#%%' ESCAPE '#';
```

Note the double "**%**%" in the LIKE clause, the first one in red "**%**" is treated as part of the string to be searched for. The other one is used to match any number of characters that follow.

The same query will also work if we use something like

```
SELECT * FROM movies WHERE title LIKE '67=%%' ESCAPE '=';
```

Summary

- Like & Wildcards powerful tools that help search data matching complex patterns.
- There are a number of wildcards that include the percentage, underscore and charlist(not supported by MySQL) among others
- The percentage wildcard is used to match any number of characters starting from zero (0) and more.
- The underscore wildcard is used to match exactly one character