

Chapter 11: LIKE operator

Section 11.1: Match open-ended pattern

The % wildcard appended to the beginning or end (or both) of a string will allow 0 or more of any character before the beginning or after the end of the pattern to match.

Using '%' in the middle will allow 0 or more characters between the two parts of the pattern to match.

We are going to use this Employees Table:

Id	FName	LName	PhoneNumber	ManagerId	DepartmentId	Salary	Hire_date
1	John	Johnson	2468101214	1	1	400	23-03-2005
2	Sophie	Amudsen	2479100211	1	1	400	11-01-2010
3	Ronny	Smith	2462544026	2	1	600	06-08-2015
4	Jon	Sanchez	2454124602	1	1	400	23-03-2005
5	Hilde	Knag	2468021911	2	1	800	01-01-2000

Following statement matches for all records having FName **containing** string 'on' from Employees Table.

```
SELECT * FROM Employees WHERE FName LIKE '%on%';
```

Id	FName	LName	PhoneNumber	ManagerId	DepartmentId	Salary	Hire_date
3	Ronny	Smith	2462544026	2	1	600	06-08-2015
4	Jon	Sanchez	2454124602	1	1	400	23-03-2005

Following statement matches all records having PhoneNumber **starting with** string '246' from Employees.

```
SELECT * FROM Employees WHERE PhoneNumber LIKE '246%';
```

Id	FName	LName	PhoneNumber	ManagerId	DepartmentId	Salary	Hire_date
1	John	Johnson	246 8101214	1	1	400	23-03-2005
3	Ronny	Smith	246 2544026	2	1	600	06-08-2015
5	Hilde	Knag	246 8021911	2	1	800	01-01-2000

Following statement matches all records having PhoneNumber **ending with** string '11' from Employees.

```
SELECT * FROM Employees WHERE PhoneNumber LIKE '%11';
```

Id	FName	LName	PhoneNumber	ManagerId	DepartmentId	Salary	Hire_date
2	Sophie	Amudsen	24791002 11	1	1	400	11-01-2010
5	Hilde	Knag	24680219 11	2	1	800	01-01-2000

All records where FName **3rd character** is 'n' from Employees.

```
SELECT * FROM Employees WHERE FName LIKE '__n%';
```

(two underscores are used before 'n' to skip first 2 characters)

Id	FName	LName	PhoneNumber	ManagerId	DepartmentId	Salary	Hire_date
3	Ronny	Smith	2462544026	2	1	600	06-08-2015
4	Jon	Sanchez	2454124602	1	1	400	23-03-2005

Section 11.2: Single character match

To broaden the selections of a structured query language (SQL-SELECT) statement, wildcard characters, the percent sign (%) and the underscore (_), can be used.

The _ (underscore) character can be used as a wildcard for any single character in a pattern match.

Find all employees whose FName start with 'j' and end with 'n' and has exactly 3 characters in FName.

```
SELECT * FROM Employees WHERE FName LIKE 'j_n'
```

_ (underscore) character can also be used more than once as a wild card to match patterns.

For example, this pattern would match "jon", "jan", "jen", etc.

These names will not be shown "jn","john","jordan", "justin", "jason", "julian", "jillian", "joann" because in our query one underscore is used and it can skip exactly one character, so result must be of 3 character FName.

For example, this pattern would match "LaSt", "LoSt", "HaLt", etc.

```
SELECT * FROM Employees WHERE FName LIKE '_A_T'
```

Section 11.3: ESCAPE statement in the LIKE-query

If you implement a text-search as [LIKE](#)-query, you usually do it like this:

```
SELECT *  
FROM T_Whatever  
WHERE SomeField LIKE CONCAT('%', @in_SearchText, '%')
```

However, (apart from the fact that you shouldn't necessarily use [LIKE](#) when you can use fulltext-search) this creates a problem when somebody inputs text like "50%" or "a_b".

So (instead of switching to fulltext-search), you can solve that problem using the [LIKE](#)-escape statement:

```
SELECT *  
FROM T_Whatever  
WHERE SomeField LIKE CONCAT('%', @in_SearchText, '%') ESCAPE '\'
```

That means \ will now be treated as ESCAPE character. This means, you can now just prepend \ to every character in the string you search, and the results will start to be correct, even when the user enters a special character like % or _.

e.g.

```
string stringToSearch = "abc_def 50%";  
string newString = "";  
foreach(char c in stringToSearch)  
    newString += @"\" + c;  
  
sqlCmd.Parameters.Add("@in_SearchText", newString);  
// instead of sqlCmd.Parameters.Add("@in_SearchText", stringToSearch);
```

Note: The above algorithm is for demonstration purposes only. It will not work in cases where 1 grapheme consists out of several characters (utf-8). e.g. string stringToSearch = "Les Mise\u0301rables"; You'll need to do this

for each grapheme, not for each character. You should not use the above algorithm if you're dealing with Asian/East-Asian/South-Asian languages. Or rather, if you want correct code to begin with, you should just do that for each graphemeCluster.

See also [ReverseString, a C# interview-question](#)

Section 11.4: Search for a range of characters

Following statement matches all records having FName that starts with a letter from A to F from Employees Table.

```
SELECT * FROM Employees WHERE FName LIKE '[A-F]%'
```

Section 11.5: Match by range or set

Match any single character within the specified range (e.g.: [a-f]) or set (e.g.: [abcdef]).

This range pattern would match "gary" but not "mary":

```
SELECT * FROM Employees WHERE FName LIKE '[a-g]ary'
```

This set pattern would match "mary" but not "gary":

```
SELECT * FROM Employees WHERE FName LIKE '[lmnop]ary'
```

The range or set can also be negated by appending the ^ caret before the range or set:

This range pattern would *not* match "gary" but will match "mary":

```
SELECT * FROM Employees WHERE FName LIKE '[^a-g]ary'
```

This set pattern would *not* match "mary" but will match "gary":

```
SELECT * FROM Employees WHERE FName LIKE '[^lmnop]ary'
```

Section 11.6: Wildcard characters

wildcard characters are used with the SQL LIKE operator. SQL wildcards are used to search for data within a table.

Wildcards in SQL are: %, _, [charlist], [^charlist]

% - A substitute for zero or more characters

```
Eg: //selects all customers with a City starting with "Lo"
SELECT * FROM Customers
WHERE City LIKE 'Lo%';

//selects all customers with a City containing the pattern "es"
SELECT * FROM Customers
WHERE City LIKE '%es%';
```

_ - A substitute for a single character

```
Eg://selects all customers with a City starting with any character, followed by "erlin"
SELECT * FROM Customers
```

```
WHERE City LIKE '_erlin';
```

[charlist] - Sets and ranges of characters to match

```
Eg://selects all customers with a City starting with "a", "d", or "l"  
SELECT * FROM Customers  
WHERE City LIKE '[adl]';
```

```
//selects all customers with a City starting with "a", "d", or "l"  
SELECT * FROM Customers  
WHERE City LIKE '[a-c]';
```

[^charlist] - Matches only a character NOT specified within the brackets

```
Eg://selects all customers with a City starting with a character that is not "a", "p", or "l"  
SELECT * FROM Customers  
WHERE City LIKE '[^apl]';
```

or

```
SELECT * FROM Customers  
WHERE City NOT LIKE '[apl]%' and city like '_%';
```
