文章
姚 鑫 · 十二月 14, 2021 阅读大约需 7 分钟

第十五章 SQL谓词 LIKE

第十五章 SQL谓词 LIKE

用包含字面值和通配符的模式字符串匹配值。

大纲

```
scalar-expression LIKE pattern [ESCAPE char]
```

参数

- scalar-expression - 一个标量表达式(最常见的是数据列)，它的值正在与模式进行比较。
- pattern - 一个带引号的字符串，表示要与标量表达式中的每个值匹配的字符模式。
  模式字符串可以包含字面字符、下划线(_)和百分比(%)通配符。
- ESCAPE char 可选-包含单个字符的字符串。
  这个字符字符可以在模式中用于指定紧跟在它后面的字符将被视为文字。

描述

LIKE谓词允许选择那些匹配模式中指定的字符的数据值。
模式可以包含通配符。
如果pattern不匹配任何标量表达式值，LIKE返回空字符串。

LIKE可以在任何可以指定谓词条件的地方使用，如本手册的谓词概述页面所述。

LIKE谓词支持以下通配符:

- _ - 任何单个字符
- % - 由0个或多个字符组成的序列。
  (根据SQL标准，NULL不被认为是一个0字符的序列，因此不被这个通配符选中。)

在动态SQL或嵌入式SQL中，模式可以将通配符和输入参数或输入主机变量表示为连接的字符串，如示例部分所示。

**注意:当在运行时提供谓词值时(使用?
输入参数或:var输入主机变量)，结果谓词%STARTSWITH 'abc'提供了比等价的结果谓词'abc%'更好的性能。**

排序类型

模式字符串使用与它匹配的列相同的排序规则类型。
默认情况下，字符串数据类型字段是用SQLUPPER排序规则定义的，它不区分大小写。

如果LIKE应用于具有SQLUPPER默认排序类型的字段，则LIKE子句返回忽略字母大小写的匹配项。
可以使用SQLSTRING排序规则类型执行区分大小写的LIKE字符串比较。

下面的示例返回包含子字符串 “ Ro ” 的所有名称。
因为LIKE不区分大小写，LIKE '%Ro%'返回Robert, Rogers, deRocca, LaRonga, Brown, Mastroni等:

```
SELECT Name FROM Sample.Person
WHERE Name LIKE '%Ro%'
```

将其与Contains操作符([)进行比较，后者使用EXACT(区分大小写)排序:

```
SELECT Name FROM Sample.Person
WHERE Name [ 'Ro'
```

通过使用%SQLSTRING排序类型，可以使用LIKE只返回那些包含区分大小写的子字符串 “ Ro ” 的名称。
Mastroni 和Brown都不会回来:

```
SELECT Name FROM Sample.Person
WHERE %SQLSTRING(Name) LIKE '%Ro%'
```

在上面的示例中，%SQLSTRING附加到Name值的前导空格由%通配符处理。
一个更健壮的例子是在谓词两边指定排序规则类型:

```
SELECT Name FROM Sample.Person
WHERE %SQLSTRING(Name) LIKE %SQLSTRING('%Ro%')
```

所有值，空字符串值，和NULL

如果模式值是percent (%)， LIKE选择指定字段的所有值，包括空字符串值:

```
SELECT Name,FavoriteColors FROM Sample.Person
WHERE FavoriteColors LIKE '%'
```

它不选择NULL字段。

指定空字符串的模式值将返回空字符串值。

```
SELECT Name,FavoriteColors FROM Sample.Person
WHERE FavoriteColors LIKE ''
```

指定模式值为NULL不是一个有意义的操作。
它成功完成，但没有返回值。

```
SELECT Name,FavoriteColors FROM Sample.Person
WHERE FavoriteColors LIKE NULL
```

像大多数谓词一样，Like可以使用NOT逻辑运算符进行反转。
LIKE和NOT LIKE都不能用于返回NULL字段。
返回NULL字段使用IS NULL。

ESCAPE子句

ESCAPE允许在模式中使用通配符作为文本字符。
如果提供了ESCAPE字符并且它是单个字符，则表示模式中直接跟在它后面的任何字符都应该被理解为文字字符，而不是通配符或格式化字符。
下面的例子展示了如何使用ESCAPE返回包含字符串 “ SYS ” 的值:

```
SELECT * FROM MyTable
WHERE symbol_field LIKE '%\_SYS%' ESCAPE '\'
```

%SelectMode

LIKE谓词不使用当前的%SelectMode设置。
应该以逻辑格式指定模式，无论%SelectMode设置如何。
尝试以ODBC格式或Display格式指定模式通常会导致没有数据匹配或意外的数据匹配。

可以使用%EXTERNAL或%ODBCOUT格式转换函数来转换谓词操作的标量表达式字段。
这允许以Display格式或ODBC格式指定模式。
但是，使用格式转换函数会阻止对字段使用索引，因此会对性能产生重大影响。

在下面的动态SQL示例中，LIKE谓词以逻辑格式指定日期模式，而不是%SelectMode=1 (ODBC)格式。
从41开始的逻辑值(日期从1953年4月4日($HOROLOG 41000)到1955年12月28日($HOROLOG 41999))被选中:

```
/// d ##class(PHA.TEST.SQLCommand).Like()
ClassMethod Like()
{
    s q1 = "SELECT Name,DOB FROM Sample.Person "
    s q2 = "WHERE DOB LIKE '41%'"
    s myquery = q1_q2
    s tStatement = ##class(%SQL.Statement).%New()
    s tStatement.%SelectMode=1
    s qStatus = tStatement.%Prepare(myquery)
    if qStatus'=1 {
        w "%Prepare failed:"
        d $System.Status.DisplayError(qStatus)
        q
    }
    s rset = tStatement.%Execute()
    d rset.%Display()
    w !,"End of data"
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLCommand).Like()
Name    DOB
Houseman,Martin D.      1955-09-25
Ingrahm,Yan S.  1954-06-15
Smith,Elvis Y.  1955-06-29
Ng,Liza Z.      1955-10-05
Zweifelhofer,Zelda J.   1954-02-19
```

```
Zampitello,Josephine Q. 1953-08-14
Hertz,Uma C.    1954-07-25
Davis,Jane E.   1953-07-28
Vanzetti,Alexandra O.   1953-12-29

9 Rows(s) Affected
End of data
```

下面的动态SQL示例使用%ODBCOUT格式转换函数来转换谓词匹配的DOB字段。
这允许以ODBC格式指定LIKE模式。
它选择DOB字段ODBC值以195开头的行(日期范围从1950年到1959年)。
但是，指定格式转换函数会阻止对DOB字段值使用索引:

```
ClassMethod Like1()
{
    s q1 = "SELECT Name,DOB FROM Sample.Person "
    s q2 = "WHERE %ODBCOUT(DOB) LIKE '195%'"
    s myquery = q1_q2
    s tStatement = ##class(%SQL.Statement).%New()
    s tStatement.%SelectMode=1
    s qStatus = tStatement.%Prepare(myquery)
    if qStatus'=1 {
        w "%Prepare failed:"
        d $System.Status.DisplayError(qStatus)
        q
    }
    s rset = tStatement.%Execute()
    d rset.%Display()
    w !,"End of data"
}
```

```
HC-APP>d ##class(PHA.TEST.SQLCommand).Like1()
Name    DOB
Houseman,Martin D.      1955-09-25
Ingrahm,Yan S.  1954-06-15
Smith,Elvis Y.  1955-06-29
Gore,Alfred M.  1958-09-15
Yoders,Liza U.  1959-06-05
Ng,Liza Z.      1955-10-05
Yeats,Debby G.  1951-12-06
Zweifelhofer,Zelda J.   1954-02-19
Solomon,Emily D.        1953-01-28
Isaacs,Elvis V. 1952-04-05
Pantaleo,Robert U.      1950-03-29
Zampitello,Josephine Q. 1953-08-14
Xiang,Molly F.  1953-03-21
Nichols,Heloisa M.      1957-07-19
Hertz,Uma C.    1954-07-25
LaRocca,David X.        1956-01-11
Houseman,Alice R.       1957-12-07
Alton,Phil T.   1953-02-25
Davis,Jane E.   1953-07-28
Vanzetti,Alexandra O.   1953-12-29
Uhles,Dmitry P. 1951-08-23
Jafari,Christine Z.     1950-04-11
```

```
22 Rows(s) Affected
End of data
```

文字替换覆盖

在编译预解析期间，可以用双括号将LIKE谓词参数括起来，从而重写文字替换。
例如，WHERE Name LIKE (('Mc%'))或WHERE Name LIKE (('%son%'))。
这可以通过改善整体选择性和/或下标边界选择性来提高查询性能。
但是，当使用不同的值多次调用同一个查询时，应该避免使用这种方法，因为这将导致为每个查询调用创建一个单独的缓存查询。

示例

下面的示例使用WHERE子句选择包含 “ son ” 的Name值，包括以 “ son ” 开头或结尾的值。
默认情况下，LIKE字符串比较是不区分大小写的:

```
SELECT %ID,Name FROM Sample.Person
WHERE Name LIKE '%son%'
```

下面的嵌入式SQL示例返回与前一个示例相同的结果集。
注意如何在LIKE模式中使用连接操作符指定输入主机变量(:subname):

```
ClassMethod Like2()
{
    s subname = "son"
    &sql(
        DECLARE C1 CURSOR FOR SELECT %ID,Name INTO :id,:nameout FROM Sample.Person
        WHERE Name LIKE '%'_:subname_'%'
    )
    &sql(OPEN C1)
    q:(SQLCODE'=0)
    &sql(FETCH C1)
    while (SQLCODE = 0) {
        w id," ",nameout,!
        &sql(FETCH C1)
    }
    &sql(CLOSE C1)
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLCommand).Like2()
86 Anderson,Mario L.
131 Anderson,Valery N.
67 Donaldson,Julie I.
50 Emerson,Edgar T.
43 Hanson,George C.
164 Jackson,Ralph V.
119 Jackson,Terry L.
100 Johnson,Danielle I.
54 Larson,Nataliya Z.
103 Nathanson,Norbert Z.
52 Nelson,Neil E.
143 Nelson,Paul O.
```

```
214 Peterson,Alice E.
118 Peterson,Janice N.
196 Peterson,Kirsten R.
140 Peterson,Sophia A.
123 Sorenson,Samantha X.
149 Thompson,Umberto Q.
184 Wilson,Andrew O.
58 Wilson,Quentin Z.
```

下面的动态SQL示例返回与前一个示例相同的结果集。
注意如何在LIKE模式中使用连接操作符指定输入参数(?):

```
ClassMethod Like3()
{
    s myquery = "SELECT %ID,Name FROM Sample.Person WHERE Name LIKE '%'_?_'%'"
    s tStatement = ##class(%SQL.Statement).%New()
    s qStatus = tStatement.%Prepare(myquery)
    if qStatus'=1 {
        w "%Prepare failed:"
        d $System.Status.DisplayError(qStatus)
        q
    }
    s rset = tStatement.%Execute("son")
    d rset.%Display()
}
```

下面的示例使用WHERE子句选择包含 “ blue ” 的FavoriteColors值。
FavoriteColors字段是一个%List字段;
%通配符处理%List格式字符:

```
SELECT Name,FavoriteColors FROM Sample.Person
WHERE FavoriteColors LIKE '%blue%'
```

下面的示例使用HAVING子句为年龄以1开头后跟一个字符的人选择记录。
它显示所有年龄的平均值和HAVING子句选择的年龄的平均值。
它根据年龄对结果排序。
所有返回值的年龄从10到19。

```
SELECT Name,
       Age,
       AVG(Age) AS AvgAge,
       AVG(Age %AFTERHAVING) AS AvgTeen
FROM Sample.Person
HAVING Age LIKE '1_'
ORDER BY Age
```

#SQL #Caché

源 URL:

https://cn.community.intersystems.com/post/%E7%AC%AC%E5%8D%81%E4%BA%94%E7%AB%A0-sql%E8%B0%93%E8%AF%8D