SQL通配符

在搜索数据库中的数据时，SQL 通配符可以替代一个或多个字符。SQL 通配符必须与 LIKE 运算符一起使用，而like用在Where子句里面。

SELECT \* FROM Persons WHERE City LIKE 'N%'

SELECT \* FROM Persons WHERE City LIKE '%g'

SELECT \* FROM Persons WHERE City LIKE '%lon%'

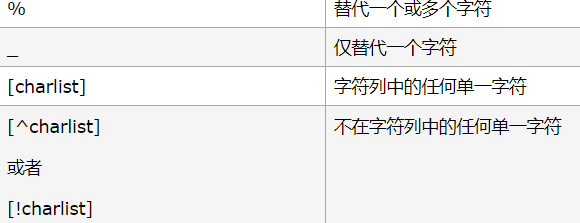
SELECT \* FROM Persons WHERE City NOT LIKE '%lon%'

SELECT \* FROM Persons WHERE LastName LIKE 'C\_r\_er'

**SELECT \* FROM Persons WHERE City LIKE '[ALN]%'**

**SELECT \* FROM Persons WHERE City LIKE '[!ALN]%' / '[^ALN]'**

**选取居住的城市以/不以 "A" 或 "L" 或 "N" 开头的人**



正则表达式

*==正则表达式****是一个用来描述和匹配一系列符合某个句法规则的字符串的单个字符串***

*==为什么要在前端开发中使用正则表达式*

1. 防止SQL注入

通过在form表单检验中使用正则表达式，能够验证用户的输入信息，保证数据的合法性

1. 采集数据

通过使用正则表达式可以从互联网上按照某种句法规则搜索想要的数据，对于分析复杂的html标签，RegExp是一个很好用的方法

*==正则表达式 简单的四种匹配规则*

**模式pattern**，是RegExp最基本的元素，它们是一组**描述字符串特征**的字符。

往往用特殊的字符表示**一个范围内的字符、重复出现、上下文**。

^once

该模式只匹配：以once开头的字符串

$once

该模式只匹配：以once结尾的字符串

^once$

该模式表示精确匹配：字符串与模式一样，只匹配该字符串。

Tips：如果一个模式不包含^ or $，那么它与任何包含该模式的字符串匹配。

^\t

该模式中\t表示制表符的**转移序列**。检测一个字符串是否以制表符开头。

Tips：如标点符号和白字符（空格、制表符等）等都要用**转义序列**。还有如\n换行 \r回车 [\\反斜杠](file:///\\反斜杠) \’单引号 \”双引号 \?问号 \ddd1~3位八进制数对应的字符（如’\141’代表字符常量’a’） \xhh1~2位十六进制数对应的字符（如’\x41’代表字符常量’A’）

*转义字符*

**所有的ASCII码都可以用“\”加数字（一般是8进制数字，用\0hh表示；而十六进制数字，用\xhh表示）来表示**。

1.在而C中定义了**一些字母前加"\"**来表示常见的那些不能显示的ASCII字符，如\0,\t,\n等，就称为**转义字符**。

2. url

web 开发中，通过问号（?）方式在浏览器地址栏中传值时，浏览器是通过“&”来区分问号后的参数个数的。如果出现传值参数中带有“&”时，在接受页面就会出现错误，类似如下请求路径：/next.jsp?param1=hendhs89**&**furej **&** param2=sss。因为参数param1中含有转义字符“&”，这样会导致被请求页的参数接收错误。

所以，常常在传值前通过 **java.net.URLEncoder.encode(param1)** 编码处理后将转义字符转为16进制;

*==正则表达式 字符簇/方括号/验证一个用户的输入*

比如当一个用户提交一个form表单，要判断输入的电话、地址、Email地址、信用卡号码等是否有效，用普通的基于字面的字符是不够的。**所以，要用一种更自由的描述我们要的模式的方法——字符簇**。

要建立一个表示所有元音字符的字符簇，就要把所有元音字符放在一个方括号里：

[AaEeIiOoUu] 该模式匹配任何一个元音字符

[a-z] 该模式匹配所有的任何一个小写字母。

用连字符表示一个字符的范围

[A-Z] 该模式所有的任何一个大写字母

[a-zA-Z] 该模式匹配所有的任何一个字母

[0-9] 匹配所有的任何一个数字

[0-9\.\-] 匹配所有的数字、句号和减号（任何一个）

[ \f\r\t\n] 匹配所有的任何一个白字符，f换页r回车t制表 n换行

Tips：上面的都只表示一个字符。

匹配一个由小写字母和一位数字组成的字符串：

^[a-z][0-9]$

^[^0-9][0-9]$ 该模式要求第一个字符不能是数字

Tips：^在一组方括号里面使用时，表示“非”或“排除”的意思，常用来剔除某个字符。

[^a-z] 该模式匹配除了小写字母以外的所有字符

[^\\\/\^] 该模式匹配除了(\)(/)(^)之外的所有字符

[^\"\'] 该模式匹配除了双引号(")和单引号(')之外的所有字符

模式“^.5$”匹配任何两个字符的、以其他非“新行”字符开头和以数字5结尾的字符串

Tips："."可以匹配任何字符串，除了空串和只包括一个"新行"的字符串（即表示除了“新行”之外的所有字符）

*==正则表达式 匹配一个单词/匹配一组数字*

^[a-zA-Z\_]$ 该模式匹配所有的字母和下划线

^a{4}$ 该模式匹配aaaa

Tips：跟在字符或字符簇后面的花括号({})用来确定前面的内容的重复出现的次数

^a{2,4}$ 该模式匹配aa,aaa或aaaa

^a{2,}$ 该模式匹配多于两个a的字符串

^a{2,} 该模式匹配如aardvark和aaab，但apple不行

a{2,} 该模式匹配如：baad和aaa，但Nantucket不行

.{2} 该模式匹配所有的两个字符

总结：

1. 一个数字{x} 的意思是前面的字符或字符簇只出现x次
2. 一个数字加逗号 {x,} 的意思是前面的内容出现x或更多的次数
3. 两个数字用逗号分隔的数字 {x,y} 表示 前面的内容至少出现x次，但不超过y次。

*==正则表达式 ?和{0, 1} 、 \*和{0,} 、 +和{1,}是相等的*

普通形式：

所有的正整数

^[1-9][0-9]{0,}$

所有的整数

^\-{0, 1}[0-9]{1,}$

**所有的浮点数**

**^ [-]? [0-9]+ \. ?[0-9]+ $**

Tips：

以一个可选的负号开头、跟着一个或更多的数字、和一个小数点、再跟上一个或多个数字。

进行简化：

所有的正整数

^[1-9][0-9]\*$

所有的整数

^\-?[0-9]+$

所有的浮点数

^ [-]? [0-9]+(\.[0-9]+)?$

\*

{0,}

+

{1,}

?

{0,1}