Material descargable

Funciones de texto

Las funciones de texto en MySQL son herramientas fundamentales que simplifican la manipulación y análisis de datos de tipo textual mediante consultas SQL. Estas funciones permiten realizar una variedad de operaciones, como la búsqueda de subcadenas, la concatenación de cadenas, la conversión de texto a minúsculas o mayúsculas, la extracción de fragmentos específicos de una cadena y muchas otras operaciones relacionadas con el procesamiento de texto. Estas funciones son fundamentales para trabajar con datos de texto de manera eficiente y efectiva en bases de datos MySQL, permitiéndote extraer información relevante y formatearla según tus necesidades.

Aquí tienes una tabla con las funciones más comunes de cadenas texto y una breve descripción:

Nombre	**Descripción**
CHAR_LENGTH()	Devuelve el número de caracteres en el argumento
CHARACTER_LENGTH()	Sinónimo de CHAR_LENGTH()
CONCAT()	Devuelve una cadena concatenada
CONCAT_WS()	Devuelve la concatenación con separador
FORMAT()	Devuelve un número formateado con un número especificado de lugares decimales
INSERT()	Inserta una subcadena en una posición específica hasta un número específico de caracteres

INSTR()	Devuelve el índice de la primera aparición de una subcadena
LCASE()	Sinónimo de LOWER()
LEFT()	Devuelve el número especificado de caracteres más a la izquierda
LENGTH()	Devuelve la longitud de una cadena en bytes
LIKE	Coincidencia de patrón simple
LOCATE()	Devuelve la posición de la primera aparición de una subcadena
LOWER()	Devuelve el argumento en minúsculas
LPAD()	Devuelve el argumento de cadena con relleno a la izquierda con la cadena especificada
LTRIM()	Elimina los espacios iniciales
MAKE_SET()	Devuelve un conjunto de cadenas separadas por comas que tienen el bit correspondiente en bits establecido
MATCH()	Realiza una búsqueda de texto completo
MID()	Devuelve una subcadena a partir de la posición especificada
NOT LIKE	Negación de coincidencia de patrón simple
NOT REGEXP	Negación de REGEXP
POSITION()	Sinónimo de LOCATE()
QUOTE()	Escapa el argumento para su uso en una declaración SQL
REGEXP	Si la cadena coincide con una expresión regular

REGEXP_INSTR()	Índice de inicio de la subcadena que coincide con una expresión regular
REGEXP_LIKE()	Si la cadena coincide con una expresión regular
REGEXP_REPLACE()	Reemplaza subcadenas que coinciden con una expresión regular
REGEXP_SUBSTR()	Devuelve una subcadena que coincide con una expresión regular
REPEAT()	Repite una cadena el número especificado de veces
REPLACE()	Reemplaza las ocurrencias de una cadena especificada
REVERSE()	Invierte los caracteres de una cadena
RIGHT()	Devuelve el número especificado de caracteres más a la derecha
RLIKE	Si la cadena coincide con una expresión regular
RPAD()	Añade una cadena el número especificado de veces
RTRIM()	Elimina los espacios finales
SPACE()	Devuelve una cadena del número especificado de espacios
STRCMP()	Compara dos cadenas
SUBSTR()	Devuelve la subcadena como se especifica
SUBSTRING()	Devuelve la subcadena como se especifica
SUBSTRING_INDEX()	Devuelve una subcadena de una cadena antes del número especificado de ocurrencias del delimitador
TRIM()	Elimina los espacios iniciales y finales

UCASE()	Sinónimo de UPPER()
UPPER()	Convierte a mayúsculas
GROUP_CONCAT()	concatena un grupo de datos

📍*Las funciones con valor de cadena devuelven NULL si la longitud del resultado es mayor que el valor de la variable de sistema max_allowed_packet.*

CHAR_LENGTH(str) o CHARACTER_LENGTH(str)

Devuelve la longitud de la cadena str, medida en puntos de código (caracteres). Un carácter multibyte se cuenta como un solo punto de código. Esto significa que, para una cadena que contiene dos caracteres de 3 bytes, LENGTH() devuelve 6, mientras que CHAR_LENGTH() devuelve 2, como se muestra aquí:

```
SELECT CHAR_LENGTH('海豚');
2
SELECT LENGTH('海豚');
6
```

CHAR_LENGTH() devuelve NULL si str es NULL.

CONCAT(str1, str2, ...)

Devuelve la cadena resultante de concatenar los argumentos. Puede tener uno o más argumentos.

```
SELECT CONCAT('h', 'o', 'l', 'a');
hola
```

CONCAT() devuelve NULL si alguno de los argumentos es NULL.

CONCAT_WS(separator, str1, str2, ...)

CONCAT_WS() significa Concatenar con Separador. El primer argumento es el separador para el resto de los argumentos. El separador se agrega entre las cadenas que se van a concatenar. El separador puede ser una cadena, al igual que el resto de los argumentos. Si el separador es NULL, el resultado es NULL.

```
SELECT CONCAT_WS(',', 'hola', ' como estas?');
hola, como estas?
```

FORMAT(X, D, [locale])

Formatea el número X en un formato como '#,###,###.##', redondeado a D lugares decimales, y devuelve el resultado como una cadena. Si D es 0, el resultado no tiene punto decimal ni parte fraccional. Si X o D es NULL, la función devuelve NULL. El tercer parámetro opcional permite especificar una configuración regional a utilizar para el punto decimal, separador de miles y agrupación entre separadores.

```
SELECT FORMAT(12332.123456, 4);
'12,332.1235'
SELECT FORMAT(12332.1,4);
'12,332.1000'
SELECT FORMAT(12332.2,0);
'12,332'
SELECT FORMAT(12332.2,2,'de_DE');
'12.332,20'
```

Aquí tienes un link donde encontraras los distintos “locale” permitidos:
MySQL Server Locale Support

INSERT(str, pos, len, newstr)

Devuelve la cadena str, con el substring que comienza en la posición pos y tiene len caracteres de longitud reemplazados por la cadena newstr. Devuelve la cadena original si pos no está dentro de la longitud de la cadena. Reemplaza el resto de la cadena a partir de la posición pos si len no está dentro de la longitud de la cadena restante. Devuelve NULL si alguno de los argumentos es NULL.

```
SELECT INSERT('Quadratic', 3, 4, 'What');
QuWhattic
SELECT INSERT('Quadratic', -1, 4, 'What');
Quadratic
SELECT INSERT('Quadratic', 3, 100, 'What');
QuWhat
```

INSTR(str, substr)

Devuelve la posición de la primera ocurrencia de la subcadena substr en la cadena str. Esta función es segura para cadenas multibyte y distingue mayúsculas de minúsculas. Si alguno de los argumentos es NULL, la función devuelve NULL.

```
SELECT INSTR('foobarbar', 'bar');
4
SELECT INSTR('xbar', 'foobar');
0
```

Es lo mismo que la forma de dos argumentos de LOCATE(), excepto que el orden de los argumentos está invertido

LCASE(str) o LOWER()

Devuelve la cadena str con todos los caracteres convertidos a minúsculas según la asignación de caracteres o NULL si str es NULL.

```
SELECT LOWER('HoLa');
hola
```

LEFT(str, len)

Devuelve los primeros len caracteres de la cadena str, o NULL si alguno de los argumentos es NULL. Esta función es segura para cadenas multibyte.

```
SELECT LEFT('foobarbar', 5);
fooba
```

LENGTH(str)

Devuelve la longitud de la cadena str, medida en bytes. Un carácter multibyte se cuenta como varios bytes. Esto significa que para una cadena que contiene cinco caracteres de 2 bytes, LENGTH() devuelve 10, mientras que CHAR_LENGTH() devuelve 5. Devuelve NULL si str es NULL.

```
SELECT LENGTH('海豚');
6
SELECT CHAR_LENGTH('海豚');
```

```
2
```

LOCATE(substr, str), LOCATE(substr, str, pos)

La primera sintaxis devuelve la posición de la primera ocurrencia de la subcadena substr en la cadena str. La segunda sintaxis devuelve la posición de la primera ocurrencia de la subcadena substr en la cadena str, comenzando en la posición pos. Devuelve 0 si substr no se encuentra en str. Devuelve NULL si alguno de los argumentos es NULL. Esta función es segura para cadenas multibyte y distingue mayúsculas de minúsculas.

```
SELECT LOCATE('bar', 'foobarbar');
4
SELECT LOCATE('xbar', 'foobar');
0
SELECT LOCATE('bar', 'foobarbar', 5);
7
```

LPAD(str, len, padstr)

Devuelve la cadena str rellenada a la izquierda con la cadena padstr hasta una longitud de len caracteres. Si str es más largo que len, el valor de retorno se acorta a len caracteres. Devuelve NULL si alguno de sus argumentos es NULL.

```
SELECT LPAD('hi',4,'??');
??hi
SELECT LPAD('hi',1,'??');
h
SELECT LPAD('hi',10,'??');
????????hi
```

LTRIM(str)

Devuelve la cadena str con los espacios en blanco iniciales eliminados. Devuelve NULL si str es NULL. Esta función es segura para cadenas multibyte.

```
SELECT LTRIM('  barbar');
barbar
```

MAKE_SET(bits, str1, str2, ...)

Devuelve un valor de conjunto (una cadena que contiene subcadenas separadas por comas) que consiste en las cadenas que tienen el bit correspondiente en bits establecido. str1 corresponde al bit 0, str2 al bit 1 y así sucesivamente. Los valores NULL en str1, str2, ... no se agregan al resultado.

```
SELECT MAKE_SET(1,'a','b','c');
'a'
SELECT MAKE_SET(1 | 4,'hello','nice','world');
'hello,world'
SELECT MAKE_SET(1 | 4,'hello','nice',NULL,'world');
'hello'
SELECT MAKE_SET(0,'a','b','c');
''
```

MID(str, pos, len)

MID(str, pos, len) es un sinónimo de SUBSTRING(str, pos, len).

Devuelve una subcadena de la cadena str de len caracteres de longitud, comenzando en la posición pos.

```
SELECT MID('Quadratically',5,6);
ratica
```

POSITION(substr IN str)

POSITION(substr IN str) es un sinónimo de LOCATE(substr, str).

```
SELECT POSITION('bar' IN 'foobarbar');
4
SELECT POSITION('xbar' IN 'foobar');
0
```

QUOTE(str)

Citar una cadena con la función QUOTE significa envolver la cadena con comillas simples (' '), y además, las comillas simples que aparezcan dentro de la cadena, las barras invertidas (\), los ASCII NUL y Control+Z se escapan con una barra invertida (\) adicional, lo que garantiza que la cadena se pueda utilizar de

manera segura en declaraciones SQL sin causar problemas de sintaxis o interpretación incorrecta.

```
SELECT QUOTE('Don\'t!');
'Don\'t!'
SELECT QUOTE(NULL);
NULL
SELECT 'Don\'t!';
Don't!
```

Si el argumento es NULL, el valor de retorno es la palabra "NULL" sin comillas simples.

REPEAT(str, count)

Devuelve una cadena que consiste en la cadena str repetida count veces. Si count es menor que 1, devuelve una cadena vacía. Devuelve NULL si str o count es NULL.

```
SELECT REPEAT('MySQL', 3);
MySQLMySQLMySQL
```

REPLACE(str, from_str, to_str)

Devuelve la cadena str con todas las ocurrencias de la cadena from_str reemplazadas por la cadena to_str. REPLACE() realiza una coincidencia sensible a mayúsculas y minúsculas al buscar from_str. Devuelve NULL si alguno de sus argumentos es NULL. Esta función es segura para cadenas multibyte.

```
SELECT REPLACE('www.mysql.com', 'w', 'Ww');
WwWwWw.mysql.com
```

REVERSE(str)

Devuelve la cadena str con el orden de los caracteres invertido, o NULL si str es NULL. Esta función es segura para cadenas multibyte.

```
SELECT REVERSE('abc');
cba
```

RIGHT(str, len)

Devuelve los últimos len caracteres de la cadena str, o NULL si alguno de los argumentos es NULL. Esta función es segura para cadenas multibyte.

```
SELECT RIGHT('foobarbar', 4);
rbar
```

RPAD(str, len, padstr)

Devuelve la cadena str rellenada a la derecha con la cadena padstr hasta una longitud de len caracteres. Si str es más largo que len, el valor de retorno se acorta a len caracteres. Si str, padstr o len es NULL, la función devuelve NULL.

```
SELECT RPAD('hi',5,'?');
hi???
SELECT RPAD('hi',1,'?');
h
SELECT RPAD('hi',10,'??');
hi????????
```

RTRIM(str)

Devuelve la cadena str con los espacios en blanco finales eliminados. Esta función es segura para cadenas multibyte y devuelve NULL si str es NULL.

```
SELECT RTRIM('barbar   ');
'barbar'
```

SPACE(N)

Devuelve una cadena que consiste en N caracteres de espacio, o NULL si N es NULL.

```
SELECT SPACE(6);
'      '
```

SUBSTR(str, pos), SUBSTR(str FROM pos), SUBSTR(str, pos, len), SUBSTR(str FROM pos FOR len)

SUBSTR() es un sinónimo de SUBSTRING().

SUBSTRING(str, pos), SUBSTRING(str FROM pos), SUBSTRING(str, pos, len), SUBSTRING(str FROM pos FOR len)

Las formas sin un argumento len devuelven una subcadena de la cadena str que comienza en la posición pos. Las formas con un argumento len devuelven una subcadena de len caracteres de longitud de la cadena str, comenzando en la posición pos. Las formas que utilizan FROM son sintaxis estándar de SQL. También es posible utilizar un valor negativo para pos. En este caso, el inicio de la subcadena es pos caracteres desde el final de la cadena, en lugar del principio. Un valor negativo se puede utilizar para pos en cualquiera de las formas de esta función. Un valor de 0 para pos devuelve una cadena vacía.

```
SELECT SUBSTRING('Quadratically',5);
ratically
SELECT SUBSTRING('foobarbar' FROM 4);
barbar
SELECT SUBSTRING('Quadratically',5,6);
ratica
SELECT SUBSTRING('Sakila', -3);
ila
SELECT SUBSTRING('Sakila', -5, 3);
aki
SELECT SUBSTRING('Sakila' FROM -4 FOR 2);
ki
```

Para todas las formas de SUBSTRING(), la posición del primer carácter en la cadena de la cual se extraerá la subcadena se cuenta como 1.

SUBSTRING_INDEX(str, delim, count)

Devuelve la subcadena de la cadena str antes de count ocurrencias del delimitador delim. Si count es positivo, se devuelve todo a la izquierda del último delimitador (contando desde la izquierda). Si count es negativo, se devuelve todo a la derecha del último delimitador (contando desde la derecha). SUBSTRING_INDEX() realiza una coincidencia sensible a mayúsculas y minúsculas al buscar delim. Devuelve NULL si alguno de los argumentos es NULL.

```
SELECT SUBSTRING_INDEX('www.mysql.com', '.', 2);
www.mysql
SELECT SUBSTRING_INDEX('www.mysql.com', '.', -2);
mysql.com
```

TRIM([{BOTH | LEADING | TRAILING} [remstr] FROM] str), TRIM([remstr FROM] str)

Devuelve la cadena str con todos los prefijos o sufijos remstr eliminados. Si ninguno de los especificadores LEADING (lado izquierdo), TRAILING (lado derecho) o BOTH (ambos lados), se proporciona, se asume BOTH. remstr es opcional y, si no se especifica, se eliminan los espacios en blanco.

```
SELECT TRIM('  bar   ');
'bar'
SELECT TRIM(LEADING 'x' FROM 'xxxbarxxx');
'barxxx'
SELECT TRIM(BOTH 'x' FROM 'xxxbarxxx');
'bar'
SELECT TRIM(TRAILING 'xyz' FROM 'barxxyz');
'barx'
```

UCASE(str) o UPPER(str)

Devuelve la cadena str con todos los caracteres convertidos a mayúsculas según la asignación de caracteres actual o NULL si str es NULL. El conjunto de caracteres predeterminado es utf8mb4.

```
SELECT UPPER('HoLa');
HOLA
```

GROUP_CONCAT

GROUP_CONCAT es una función de MySQL que se utiliza para combinar valores de una columna en una única cadena de texto. A menudo se asocia con datos de texto y es útil cuando deseas mostrar información de una manera más legible o analizar datos de texto.

```
SELECT GROUP_CONCAT(column_0) FROM (VALUES ROW(85), ROW(98), ROW(100),
ROW(85), ROW(90), ROW(95)) tabla;
-- Resultado: '85,98,100,85,90,95'

SELECT GROUP_CONCAT(DISTINCT column_0 ORDER BY column_0 ASC) FROM
(VALUES ROW(85), ROW(98), ROW(100), ROW(85), ROW(90), ROW(95)) tabla;
-- Resultado: '85,90,95,98,100'
```

```
SELECT GROUP_CONCAT(column_0 SEPARATOR ' | ') FROM (VALUES ROW(85),
ROW(98), ROW(100), ROW(85), ROW(90), ROW(95)) tabla;
-- Resultado: '85 | 98 | 100 | 85 | 90 | 95'
```

📍Si bien es una función de agregación en términos técnicos, su función principal está relacionada con la manipulación de datos de texto, lo que la hace más relevante para entenderla junto con otras funciones de texto.

REGEXP

Las expresiones regulares (REGEXP) son una herramienta avanzada para buscar y manipular patrones de texto en las bases de datos. Si bien son potentes, también pueden ser complicadas de dominar. Debido a su complejidad no explicaremos su uso y funcionamiento, porque podría abarcar un curso completo aprender expresiones regulares.
