Apuntes accesibles de Funciones

Sitio: FRANCISCO DE GOYA
Curso: Bases de datos ASIR1, DAM1 y DAW1
Libro: Apuntes accesibles de Funciones

Imprimido por: Alberto Bollo Rodríguez
Día: martes, 6 de febrero de 2024, 11:45

Tabla de contenidos

1. Funciones

En ocasiones será necesario utilizar funciones que serán muy útiles, sino indispensables para lograr ciertos objetivos.

Cada una tendrá un nombre y una lista de parámetros.

- El nombre ha de ser el correcto, suele identificar bien en inglés el objetivo de la función.
- Los parámetros se indican entre paréntesis y se separan por comas: nombre_funcion(parametro1, parametro2, parametro3).
- Los parámetros de la mayoría de las funciones son un número concreto. Por ejemplo year() recibe uno, substring() recibe tres. Pero unas pocas pueden recibir un número variable de parámetros como concat().
- En muchos casos los parámetros han de ser de un tipo concreto, por ejemplo sqrt() debe tener un número, no un varchar. Aunque hay que tener cuidado porque no da error, sino warnings. La consulta se realiza.

Las funciones se pueden llamar dentro de la lista de campos de una consulta así como en la lista de condiciones. Por ejemplo:

```
select YEAR(fechaNac),  fechaNac from Persona;
```

Este ejemplo devolverá una tabla como la siguiente:

YEAR(fechaNac)	fechaNac
2009	2009-5-13
2017	2017-9-30
2011	2011-2-21

En las páginas siguientes se definirán algunas de las funciones que se utilizarán en los primeros ejercicios, pero siempre será útil (y accesible durante los exámenes) visitar las páginas de internet relacionadas, como la propia de MySql u otras como w3schools.
Además, es muy recomendable probar las funciones previamente a su uso final si no se entiende del todo. La mejor forma de probarlo es con un select y solo la función, sin tabla, como podría ser:

```
select 2+3;
```

2+3
5

1.1. Funciones numéricas

Las operaciones básicas se realizan mediante los operadores habituales:

- suma +
- resta -
- multiplicación *
- división /
- módulo % (resto de la división entera).

Hay múltiples funciones matemáticas que aplican sobre números, ya sean escritos, sean campos de una tabla, o resultados de otras funciones.

Un primer ejemplo es abs, que es el valor absoluto. abs(5) = 5 y abs(-5) = 5.

```
select abs(-5);
```

abs(-5)
5

Otras funciones,entre muchas otras, son:

- sqrt(int): devuelve la raíz cuadrada.
- ceil(int): devuelve el siguiente entero.
  - ceil(4) -> 4
  - ceil(4.01) -> 5
- round(int[,int]): devuelve el número redondeado. Si se pasa un parámetro redondea al entero más cercano. Si se pasan dos parámetros, dejará tantos decimales como el segundo parámetro.
  - round(4.4) -> 4
  - round(4.8) -> 5
  - round(4.012,2) -> 4.01
  - round(4.81274,3) -> 4.813
- truncate(int,int): devuelve el número con los decimales especificados en el segundo parámetro.
  - truncate(4.918,0) -> 4
  - truncate(4.918,1) -> 4.9
  - truncate(4.918,2) -> 4.91
  - truncate(4.918,3) -> 4.918
- greatest(campo1,campo2,...,campoN): recibe cuantos campos pongamos y devuelve el mayor de ellos:
  - select greatest(4,6,2,1,33,23) -> 33
  - select greatest(4,7) -> 7
  - Hacerlo con diferentes campos de una tabla devolverá el mayor valor de los campos seleccionados para cada fila. No es una función de agregación, por lo que no necesita group by.

1.2. Funciones de fechas

Consulta y modificación de fechas

Las funciones más sencillas de fechas son day(), month() y year(). Estas devuelven un número como resultado, y no una fecha.

```
select year("2001-12-13"), day("2001-12-13"), month("2001-12-13");
```

year("2001-12-13")	day("2001-12-13")	month("2001-12-13")
2001	13	12

Por otro lado tenemos la función timestampdiff, que lo que hace es calcular la diferencia entre dos fechas. Se le ha de indicar en qué concepto queremos la diferencia: días, meses o años. Devolverá un número.

Al igual que está la función year() hay funciones para las demás partes como month(), dayofweek(), day(), hour() ...

```
select timestampdiff(YEAR,"2008-02-05","2010-02-08");
```

timestampdiff(YEAR,"2008-02-05","2010-02-08")
2

```
select timestampdiff(YEAR,"2010-02-05","2008-02-08");
```

timestampdiff(YEAR,"2010-02-05","2008-02-08")
-2

Otra función útil es curdate(). Devuelve la fecha actual, para así poder usarla en comparaciones con fechas, con timestampdiff o date_add().

```
select curdate();
```

curdate()
2020-01-09

La función date_add() nos permitirá añadir una cantidad de tiempo concreta a una fecha:

```
select date_add("2009-05-25",interval 250 day);
```

date_add("2009-05-25",interval 250 day)
2010-01-30

Tenemos dos funciones para trabajar con los días de la semana, weekday() y dayofweek()

weekday() recibe una fecha y devuelve un número del 0 al 6 siendo el 0 el lunes y el 6 el domingo. En el siguiente ejemplo, la fecha introducida es un jueves.

```
select weekday("2022-01-20");
```

weekday(2022-01-20)
3

Por otro lado, dayofweek() devuelve valores del 1 al 7, pero donde el 1 es domingo y el 7 es el sábado (semana inglesa). Con la misma fecha, que es jueves, devuelve un 5:

```
select dayofweek("2022-01-20");
```

dayofweek("2022-01-20")
5

Formato de fecha

Una fecha puede mostrarse por defecto tal y como se ve en los ejemplos anteriores, o usar todos los datos que se pueden sacar de una fecha, y no solo día, mes y año: día de la semana, hora, nombre del día de la semana en inglés.

Esto habrá que indicarlo mediante unas variables precedidas de "%" en la función. Todo lo que no sea una variable se mostrará literal, por ejemplo si escribimos "%m del %y" aparecerá "del" entre el mes y el año.

Algunas de las opciones son las siguientes:

Format	Descripción
%a	Abreviatura del día de la semana en inglés (sun, sat...)
%c	Número del mes.
%H	Hora de 0 a 23
%i	Minutos, de 0 a 59
%j	Día del año, de 0 a 366
%M	Nombre del mes en inglés completo (March)
%u	Semana del año, comenzando en lunes, de 0 a 53
%w	Día de la semana del 0 al 6, siendo 0 domingo, 1 lunes y 6 sábado
%Y	Año numérico con 4 dígitos
%y	Año numérico con 2 dígitos
%p	AM o PM

Hay más opciones disponibles aquí.
Ejemplo:

```
select date_format("2020-01-29","%d de %m de %Y");
```

date_format("2020-01-29","%d de %m de %Y")
29 de 01 de 2020

Formato de fecha en español

Se puede cambiar el formato fecha para que los días aparezcan como lunes, martes, etc, y los meses como enero, febrero, etc mediante el siguiente comando;

```
SET lc_time_names = 'es_ES';
```

Si bien esto habría que ejecutarlo cada vez que se inicie un cliente mysql (una conexión a base de datos). Para hacerse permanente debería cambiarse el archivo de configuración del servidor y reiniciar.

Otras funciones

Hay muchas otras funciones que pueden utilizarse, entre ellas se puede destacar:

- pow(base,exponente): devuelve el resultado de elevar la base al exponente.
- sign(num) devuelve si el número es positivo (1), negativo (-1) o cero (0)
-

1.3. Funciones de varchar

Hay múltiples funciones de varchar, también llamados string.

Length

La función length, que devuelve la longitud de una cadena de caracteres (un número):

```
select length("Hola");
```

length("Hola)"
4

Concat

Otra que es útil es concat, que recibe cualquier número de parámetros, los cuales pueden ser varchar o no, admitiendo números y fechas. Devuelve una única cadena de caracteres.

```
select concat("Hola",5,"Adios");
```

concat("Hola",5,"Adios")
Hola5Adios

Substring, right y left

También están las funciones que seleccionan parte de la cadena de caracteres. La más versátil es substring(). Substring recibe tres parámetros: la cadena sobre la que se trabaja, la posición desde donde se va a seleccionar y por último el número de caracteres que se seleccionan:

```
select substring("Enero",2,3);
```

substring("Enero",2,3)
ner

Es importante tener en cuenta que el segundo parámetro, que indica la posición desde donde se empieza a seleccionar, empieza por el 1 y no por el 0 como en la mayoría de lenguajes. También que si es negativo se contará desde la derecha, y no desde la izquierda, aunque la longitud seguirá contando en el sentido habitual:

```
select substring("Enero",-3,2);
```

substring("Enero",-3,2)
er

Si bien es una función muy versátil, hay otras que la simplifican para hacer tareas concretas:

```
select substring("Enero",1,3);
```

es lo mismo que

```
select left("Enero",3);
```

Resultando ambas:

left("Enero",3)
Ene

La función right() funciona igual que left, pero seleccionando desde la derecha, e igualmente, todo lo que se puede hacer con right() se puede hacer con substring().

Substring_index

La función substring_index logra reunir en una sola función lo que se puede hacer juntando locate, substring y otras. Esta función devuelve el varchar que encuentra antes de una instancia indicada:

```
select substring_index("Este es un texto aleatorio","t",1) as txt1;
```

El resultado es:

txt1
Es

Cambiamos el número:

```
select substring_index("Este es un texto aleatorio","t",2) as txt2;
```

El resultado es:

txt2
Este es un

Si es negativo, se buscará desde el final, seleccionando todo lo que hay después:

```
substring_index("Este es un texto aleatorio","t",-2)
```

El resultado es:

txt3
o aleatorio

Y con -1:

```
substring_index("Este es un texto aleatorio","t",-1)
```

El resultado es:

txt4
orio

Pero hay que tener cuidado: substring_index es "case sensitive", esto es, diferencia mayúsculas de minúsculas:

```
select substring_index("Hola _Hey, hola hey","h",2) as resultado;
```

El resultado es:

resultado
Hola _Hey, hola

Y por otro lado:

```
select substring_index("Hola _Hey, hola hey","H",2) as resultado2;
```

El resultado es:

resultado2
Hola _

Locate

La función locate es útil para localizar un substring dentro de un string y obtener la posición en la que comienza la primera ocurrencia del substring. Si no se encuentra devuelve un 0.

```
select locate("on","Acondicionador");
```

El resultado es:

locate("on","Acondicionador")
3

Y con la siguiente selección:

```
select locate("es","Terráqueo");
```

El resultado es:

locate("es","Terráqueo")
0

```
select locate("on","Acondicionador");
```

locate("es","Terráqueo")
0

Otras funciones

Hay muchas otras funciones que pueden utilizarse, entre ellas se puede destacar:

- LOWER y UPPER: cambia un texto a minúsculas o mayúsculas respectivamente.
- LPAD y RPAD: rellena con caracteres a izquierda o derecha hasta alcanzar una longitud especificada.
- REVERSE: invierte una cadena.
- TRIM, LTRIM: elimina espacios supérfluos (repetidos, anteriores a los caracteres, etc).
- STUFF: sustituye cierto contenido en una cadena, en ciertas posiciones, cambiándolo por otra cadena (solo en las posiciones indicadas).
- REPLACE(cad1,cad2,cad3): en una cadena cad1 busca dónde encuentra cad2 y lo sustituye por cad3.

1.4. Funciones de agregación

Una vez definidas las selecciones de agregación en la Unidad de Trabajo 6, en Apuntes accesibles DQL, (selects unitabla), se definen aquí una serie de funciones que se pueden utilizar:

- max(columna) selecciona el máximo de una columna de una tabla, devolviendo un único valor, o un valor por cada columna agrupada.

```
select max(sueldo) from empleados;
```

max(sueldo)
1800

En vez de aplicarlo con un número, se puede aplicar a una fecha obteniendo la más reciente:

```
select max(fecha_fac) from facturas;
```

max(fecha_fac)
2005-06-30

- count(columna) cuenta el número de elementos. Si es count(*) contará el número de filas totales, y si es count(columna) las cuenta omitiendo los nulos.

```
select count(*) from facturas
```

count(*)
114

Se puede seleccionar también cuántas diferentes hay en algún concepto. Como el siguiente ejemplo:

```
select count(distinct cliente) from facturas;
```

count(distinct cliente)
7

- min(columna) al igual que la función max() pero con el valor mínimo.

```
select min(sueldo) from empleados;
```

min(sueldo)
914.9

- min(expresión) calcula el mínimo tras evaluar la expresión, aunque al igual que en el resto de funciones numéricas, ignorará los nulos.

```
select min(sueldo+dietas+comision) from empleados;
```

min(sueldo+dietas+comision)
1352.85

Como se puede observar con un `select(*) from empleados",` se tiene solo en cuenta filas que tengan sueldo, dietas y comisión como no nulos. Por ello el mínimo no es 914.9 + 36.5, ya que esa fila tiene dietas en nulo, sino la fila que tenga los tres valores como números y cuya suma sea mínima.

- avg(columna) calcula la media de todos los valores de una columna que se pase por parámetro.

```
select avg(sueldo) from empleados;
```

avg(sueldo)
1236.974996

- avg(expresion) realiza la media después de evaluar la expresión.
- sum(columna) suma todos los valores numéricos de una columna.

```
select sum(sueldo) from empleados;
```

sum(sueldo)
7421.85

- sum(distinct columna) suma todos los valores numéricos de una columna que sean diferentes (no suma repetidos).

```
select sum(distinct precio_unidad) from facturas
```

sum(distinct precio_unidad)
18.71

El resultado aquí viene de sumar 0.15+0.17+3.00+3.12+6.02+4.5+0.25+1.5, ignorando que 0.15 aparece 26 veces, o que ya solo las 8 veces que aparece 6.02 ya es superior a toda la suma. Únicamente suma una vez cada precio diferente.

- group_concat(columna) agrupa cadenas de caracteres en una sola línea, separadas por una coma.

```
select group_concat(distinct cliente) from facturas;
```

group_concat(distinct cliente)
Ahorra Muchisimo,Baratolandia,Blanca Lecheria,Compraqui,El Barato,El Campo,Super Paquito

Otro ejemplo:

```
select propietario, group_concat(especie) from mascotas group by propietario;
```

nombre	group_concat(especie)
Benito	Perro, Serpiente
David	Gato, Perro
Diana	Perro
Juan	Gato
Omar	Ave
Tomás	Ave