Bases de Datos

Unidad 5:

Realización de consultas avanzadas

Sesión 6

FUNCIONES MYSQL

Funciones en MySQL 5.7

•Todos los SGBD incluyen un conjunto de funciones que pueden ser usadas para obtener fácilmente determinados resultados.

•Cada SGBD incluye un conjunto propio de funciones. Las funciones no forman parte del lenguaje SQL.

•La llamada a una función puede realizarse en las sentencias SELECT, INSERT, UPDATE y DELETE.

•Toda función devuelve un valor y opera con unos datos recibidos o parámetros.

•Para llamar a una función siempre se usa la sintaxis:

Nombre_funcion(param1, param1,…)

•Como parámetros pueden darse valores constantes, nombres de columnas, llamadas a otras funciones y operaciones entre los anteriores.

Funciones en MySQL 5.7

Las funciones MySQL pueden clasificarse en función de los tipos de datos con los que trabajan o del tipo de operación que realizan en:

- **Funciones matemáticas o numéricas**
- **Funciones de cadena de caracteres**
- Funciones de fecha y hora
- Funciones de búsqueda de texto
- Funciones de control de flujo
- Funciones de conversión
- Funciones de agregado o agrupación
- Otras funciones

FUNCIONES MATEMÁTICAS PRINCIPALES

- **pow(X,Y) :** Devuelve el resultado X elevado a Y.
- **sqrt(X) :** Devuelve la raíz cuadrada de X.
- **ceil(X) :** Redondea al entero más cercano por arriba.
- **floor(X):** Redondea al entero más cercano por abajo.
- **round(X) :** Redondea al entero más cercano.
- **round(X,D) :** Redondea al número más cercano usando D decimales.
- **truncate(X,D):** Obtiene el número X truncado a D decimales.
- **rand()** : Devuelve un número coma flotante aleatorio mayor o igual que cero y menor que 1.0.

FUNCIONES DE CADENA DE CARACTERES

char_length(cadena) : Devuelve el número de caracteres que tiene el contenido de la cadena.

concat(cad1, cad2,…) : Devuelve la cadena resultado de concatenar todas las cadenas pasadas. Se pueden pasar otros tipos de datos en cuyo caso los trata como cadenas de caracteres.

left(cad, N) : Devuelve los N primeros caracteres de cad.

right(cad, N) : Devuelve los N últimos caracteres de cad.

insert(cadena, posicion, longitud, nueva_cadena): Devuelve el resultado de sustituir con la nueva cadena los caracteres de cadena expresados en longitud desde la posición indicada.

FUNCIONES DE CADENA DE CARACTERES

Ejemplo: Obtener los nombres y apellidos de todos los clientes en una sola columna con el formato apellidos, nombre.

SELECT concat(apellidos,", ",nombre) AS nombrecompleto FROM clientes;

EJEMPLO: Suponiendo que la columna localidad de CLIENTES contiene erróneamente SANTRDER para todos los alumnos de Santander, hacer lo necesario modificar el valor de esa columna usando la función insert.

```
SET SQL_SAFE_UPDATES = 0; # Esta instrucción desactiva el modo seguro para UPDATES y permite actualizar valores sin usar clave
UPDATE clientes SET localidad= insert(localidad, 5, 1, 'AN')   WHERE localidad='SANTRDER';
SET SQL_SAFE_UPDATES = 1; # Esta instrucción reactiva el modo seguro para UPDATES
```

Aunque, por lógica, esto cualquiera lo haría así:

UPDATE clientes SET localidad= 'SANTANDER' WHERE localidad='SANTRDER';

FUNCIONES DE CADENA DE CARACTERES

locate(subcadena, cadena): Devuelve la posición a partir de la cual se encuentra subcadena en cadena, cero si no la encuentra.
locate(subcadena, cadena, pos): igual que la anterior buscando a partir de la posición pos.

lcase(cadena): Devuelve la cadena en minúsculas.
ucase(cadena): Devuelve cadena en mayúsculas.

lpad(cadena, N, subcadena): Devuelve cadena ocupando N caracteres, rellenando por la izquierda con subcadena si fuese necesario.
rpad(cadena, N, subcadena): igual que la anterior por la derecha.

ltrim(cadena): Devuelve cadena tras eliminarle los espacios por la izquierda si los tuviera.
rtrim(cadena): igual que la anterior por la derecha.
trim(subcadena FROM cadena): Devuelve la cadena tras eliminarle las apariciones de subcadena por la izquierda y por la derecha. Esta función admite otras sintaxis.

replace(cadena, subcadena a sustituir, subcadena a incluir): Sustituye una cadena de caracteres por otra.

FUNCIONES DE CADENA DE CARACTERES

Ejemplo: Obtener la calle en la que vive cada cliente. No hay que escribir otros datos de la dirección.

SELECT LEFT(direccion, LOCATE(',', direccion)-1) FROM clientes;

EJEMPLO: Obtener las matrículas y precios de los automóviles de forma que los precios ocupen 20 posiciones rellenando las sobrantes con '+.'

SELECT matricula, LPAD(precio, 20, '+.') FROM automoviles;

Ejemplo: Sustituir dos espacios en blanco por uno solo

SELECT replace(direccion, ' ', ' ') FROM clientes;

Funciones en MySQL 5.7

- Funciones de fecha y hora
- Funciones de búsqueda de texto
- Funciones de control de flujo
- Funciones de conversión
- Funciones de agregado o agrupación
- Otras funciones

Funciones de fecha y hora en MySQL 5.7

Adddate(fecha, INTERVAL N tipo_intervalo): Devuelve la fecha incrementada en N el tipo de intervalo indicado.
El tipo de intervalo para fechas puede ser DAY, WEEK, MONTH, QUARTER, YEAR

EJEMPLO: Suponiendo que todas las fechas de los contratos tienen como año el 2007, modificarlas para que tengan como año el 2017.

```
UPDATE contratos SET finicial=adddate(finicial,INTERVAL 10 YEAR),
ffinal=adddate(ffinal,INTERVAL 10 YEAR);
```

Funciones en MySQL 5.7

FUNCIONES DE FECHA Y HORA

Addtime(tiempo1, tiempo2): Devuelve el resultado de sumar los dos tiempos.
Subtime(tiempo1, tiempo2): Devuelve el resultado de tiempo1-tiempo2.
Curtime(): Devuelve la hora actual.

EJEMPLO: Obtener la hora que será dentro de 1 hora y 20 minutos y la que era hace 3 horas y 15 minutos.

```
SELECT addtime(curtime(),'1:20:0'), subtime(curtime(),'3:15:0');
```

Funciones en MySQL 5.7

FUNCIONES DE FECHA Y HORA

datediff(fecha1, fecha2): Devuelve los días transcurridos entre fecha2 y fecha1.
Subdate(fecha, INTERVAL N tipo_periodo): Devuelve la fecha resultado de restarle a fecha el tipo de periodo Nveces.
Curdate(): Devuelve la fecha actual.

EJEMPLO: Obtener la fecha que era hace dos trimestres y cuantos días han transcurrido desde esa fecha.

```
SELECT subdate(curdate(), INTERVAL 2 QUARTER), datediff( curdate(), subdate(curdate(), INTERVAL 2 QUARTER));
```

Funciones en MySQL 5.7

FUNCIONES DE FECHA Y HORA

date(fechahora): Devuelve la fecha de una dato DATETIME.
time(fechahora): Devuelve la parte TIME de una dato DATETIME.
year(fecha): Devuelve el año de una fecha.
quarter(fecha): Devuelve el trimestre de una fecha.
month(fecha): Devuelve el mes numérico de una fecha.
monthname(fecha): Devuelve el nombre del mes de una fecha.
day(fecha): Devuelve el día del mes de una fecha.
dayname(fecha): Devuelve el nombre del día de la semana de una fecha.
dayofweek(fecha): Devuelve el número de día de la semana de una fecha. Semana comienza en Domingo con número 1.
dayofyear(fecha): Devuelve el número de día del año de una fecha.
weekofyear(fecha): Devuelve el número de semana del año de la fecha dada. Las semanas comienzan en domingo y la primera del año es la primera con comienzo en domingo.

Funciones en MySQL 5.7

FUNCIONES DE FECHA Y HORA

now(): Devuelve la fecha y hora actuales.
hour(tiempo): Devuelve la parte horas de tiempo.
minute(tiempo): Devuelve la parte minutos de tiempo.
second(tiempo): Devuelve la parte segundos de tiempo.
sec_to_time(segundos): Convierte los segundos pasados a dato TIME
time_to_sec(tiempo): Opuesta a la anterior

Ejemplo: Obtener la hora actual y cuantos minutos faltan para la siguiente hora en punto.

```
SELECT curtime();

SELECT 60-minute(curtime());
```

Funciones en MySQL 5.7

FUNCIONES DE CONVERSIÓN

convert(valor,tipo): Convierte el valor al tipo de dato que se indique en tipo.
BINARY valor: Realmente BINARY no es una función de MySQL sino un operador especial de MySQL. Hace que valor se interprete conforme a su codificación interna. Su principal aplicación es la de servir para comparar dos cadenas de caracteres diferenciando entre mayúsculas y minúsculas, entre palabras con acentos y sin acentos, etc.

Si ejecutamos lo siguiente para obtener los clientes cuyo nombre está guardado como SANDRA, obtenemos una clienta Sandra.

SELECT * FROM clientes WHERE nombre='SANDRA';

Sin embargo, si ejecutamos los siguiente, no obtenemos ningún resultado pues no hay ningún nombre que sea exactamente SANDRA.

SELECT * FROM clientes WHERE BINARY nombre='SANDRA';

Funciones en MySQL 5.7

FUNCIONES DE CONTROL DE FLUJO

CASE *valor* WHEN [*valor1*] THEN *resultado1* [WHEN [*valor2*] THEN *resultado2* ...] [ELSE *resultado*] END

devuelve el resultado correspondiente al primer valorN que coincida con *valor*. Si ningún valorN coincide con *valor* se devuelve el resultado que hay tras la cláusula ELSE, y si no tuviera esta cláusula se devuelve NULL.

Ejemplo: Obtener el día de la semana que es hoy en español.

```
SELECT case dayofweek(curdate()) when 1 then 'domingo' when
2 then 'lunes' when 3 then 'martes' when 4 then 'miercoles'
when 5 then 'jueves' when 6 then 'viernes' when 7 then 'sabado'
end;
```

Funciones en MySQL 5.7

FUNCIONES DE CONTROL DE FLUJO

CASE WHEN [*condicion1*] THEN *resultado1* [WHEN [*condicion2*] THEN *resultado2* ...] [ELSE *resultado*] END
devuelve el resultado correspondiente a la primera condición que se cumpla.

Ejemplo: Obtener la calificación de los alumnos en formato alfanumérico. Debe preverse una calificación incorrecta.

```
SELECT nombre, apellidos, case when nota>=0 and nota<5 then
'suspenso'  when nota<6 then 'aprobado' when nota<7 then
'bien' when nota<9 then 'notable' when nota<10 then
'sobresaliente' else 'calificacion incorrecta' end FROM alumnos;
```

Funciones en MySQL 5.7

FUNCIONES DE CONTROL DE FLUJO

IF(*expr1*,*expr2*,*expr3*)
Si *expr1* es verdadera (*expr1* <> 0 and *expr1* <> NULL), devuelve *expr2*, si no devuelve *expr3*.

Ejemplo: Obtener la matrícula marca y modelo de los automóviles junto con su estado (escribiendo alquilado o disponible).

```
SELECT matricula, marca, modelo, if(alquilado, 'alquilado',
'disponible') FROM automoviles;
```

Funciones en MySQL 5.7

OTRAS FUNCIONES

aes_encrypt(*texto*,clave):Permite encriptar información usando una clave de encriptación. Utiliza la técnica AES
aes_decrypt(texto,clave): Para desencriptar.
md5(texto): Para encriptar con algoritmo MD5. NO es reversible , es decir, no hay una función para desencriptar.
connection_id(): Devuelve el número de identificador de la conexión cliente MySQL al servidor.
current_user(): Devuelve el nombre del usuario y del equipo donde éste ha sido autenticado.
last_insert_id(): Devuelve el último valor insertado en una columna AUTO_INCREMENT.
row_count(): Devuelve el número de filas que se vieron afectadas por la operación precedente de borrado, inserción o modificación.
version(): Devuelve la versión del servidor MySQL

VARIABLES DEFINIDAS POR EL USUARIO DE MYSQL

Variables definidas por el usuario de MYSQL

A veces, desea **pasar un valor de una instrucción SQL a otra** instrucción SQL. Para ello, **almacenará el valor en una variable definida por el usuario de MySQL en la primera instrucción y hará referencia a él en las instrucciones siguientes**.

Para crear una variable definida por el usuario **se debe respetar el formato** de caracteres alfanuméricos con una longitud máxima de 64 caracteres (a partir de MySQL 5.7.5): @nombre_variable

Las variables definidas por el usuario **no distinguen entre mayúsculas y minúsculas**. Significa que @id y @ID son lo mismo

Se puede asignar la variable definida por el usuario a determinados **tipos de datos** como **integer, floating point, decimal, string o NULL**.

Otros clientes no pueden ver una variable definida por el usuario definida por un cliente. En otras palabras, **una variable definida por el usuario es específica de la sesión**.

Hay que tener en cuenta que las variables definidas por el usuario son una **extensión específica de MySQL** del estándar SQL. Es posible que no estén disponibles en otros sistemas de bases de datos.

Asignación de variables MySQL con SET

Hay **dos maneras de asignar un valor a una variable** definida por el usuario.

La primera forma es utilizar la instrucción **SET** de la siguiente manera:

SET @nombre_variable:= valor;

Se puede utilizar := o = como operador de asignación en la instrucción SET.

Por ejemplo, la instrucción asigna el número 100 a la variable @numcontrato.

SET @numcontrato := 100;

O el valor del numero de contrato más alto

SET @numcontrato = (select max(numcontrato) from contratos);

~~SET @numcontrato = max(numcontrato) from contratos~~; (esto no funciona)

Asignación de variables MySQL con SELECT

La segunda forma de asignar un valor a una variable es utilizar la instrucción **SELECT**.

En este caso, se debe utilizar el operador de asignación :=

SELECT @nombre_variable:= valor;

Después de la asignación, se puede utilizar la variable en la instrucción posterior donde se permite una expresión, por ejemplo, en la cláusula WHERE, la instrucción INSERT o UPDATE.

La siguiente instrucción **obtiene el valor** del numero de contrato más alto **y lo asigna a la variable** definida por el usuario (compárese con SET)
SELECT @maximo:=MAX(numcontrato) FROM contratos;

La siguiente instrucción **utiliza la variable** @maximo para consultar la información del contrato con el número más alto.
SELECT * FROM contratos WHERE numcontrato=@maximo;

Asignación de variables MySQL con SELECT

A veces, desea insertar una fila en una tabla, obtener el último identificador de inserción y usarlo para insertar datos en otra tabla.

En este sentido, hay que indicar que **una variable definida por el usuario solo puede contener un único valor**. **Si la instrucción SELECT devuelve varios valores, la variable tomará el valor de la última fila del resultado**.

```
SELECT    @numcontrato:=numcontrato as result
FROM    contratos
WHERE
fini > '2018-01-06'
ORDER BY fini;
```

result
14
15
18
19
20
21

```
SELECT @numcontrato;
```

@numcontrato
21