BD UD6 Funciones

## **Funciones**

[Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Proc_15.jpg)

Imagen obtenida de [https://dev.mysql.com](https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html)

El conjunto de instrucciones van en el 'routine\_body'. Si lo forman varias (lo normal) deben ir entre las etiquetas BEGIN y END.

Fijarse que el END debe llevar punto y coma si a continuación le siguen nuevas instrucciones.

Una función lo conforman un bloque de instrucciones que se ejecutan y que 'devuelven' un dato el cual es 'recogido' por el que realiza la llamada a la función.

Podríamos ver una función como un procedimiento con un parámetro de tipo OUT.

Con respecto a los procedimientos, la función se diferencia (a mayores del hecho de devolver un valor) en que se pueden usar dentro de instrucciones SQL, como por ejemplo un SELECT, de la forma SELECT funcion(),col1 FROM Tabla o SELECT \* FROM Tabla WHERE col = funcion()...

En estos casos, la función se va a llamar **por cada fila de la tabla** que cumpla los criterios de selección indiciados en la consulta SELECT.

Los tipos de datos que va a poder devolver una función son: STRING, INTEGER, REAL o DECIMAL.

Nota: Indicar que en el caso de que la función devuelva un número de tipo DECIMAL, es como si devolviera un String.

Ejemplo de función:

**DELIMITER** //

**CREATE** FUNCTIONejemploFuncion() **RETURNS** VARCHAR(20)

**BEGIN**

**RETURN** 'Ejemplo';

**END**

//

Línea 3: Como vemos, en la definición de la función indicamos el **tipo de dato** que va a devolver (que si consultamos la sintaxis puede ser uno de estos tipos:STRING|INTEGER|REAL|DECIMAL).

Línea 5: En cualquier parte de la función hacemos uso de la instrucción **return** pasando un dato que debe ser del mismo tipo que el indicado en la definición.

**Ejemplos de llamada:**

**SELECT** ejemploFuncion();

**SET** @valorDevuelto = ejemploFuncio();

Mostrará la cadena 'Ejemplo'.

Una de las ventajas que tienen las funciones con respecto a los procedimientos es la posibilidad de emplearlas dentro de consultas SQL y emplear columnas de la consulta como parámetros de entrada para la función.

Por ejemplo, si tengo definida una función que devuelve el nombre de un empleado en base a su id, de la forma: employees\_getName(id Integer), puedo llamarla de la forma:

**SELECT** employees\_getName(emp\_no)

**FROM** salaries;

Como vemos, la función tiene como parámetro un id el cual se obtiene de **cada fila de la consulta** select. Es decir, la sentencia SELECT devolverá un conjunto de filas, **por cada fila** de dicha consulta, se ejecutará la función enviando como parámetro el id de la fila.

# Aclaraciones importantes

## **Aclaración 1 - Sobre funciones deterministas**

A la hora de crear una función, **si tenemos activado el**[**registro binario de log**](https://wiki.cifprodolfoucha.es/index.php?title=Mysql_Ficheros_de_log#El_registro_binario), debemos indicar que dicha función es determinista. Una función se denomina determinista si **siempre va a devolver el mismo resultado al aplicar la misma entrada**. Si no fuera así y realizamos un procedimiento de restauración de la base de datos empleado el fichero de log binario, podría darse el caso de que los datos 'recuperados' no fueran los mismos que antes de la recuperación.

Para hacerlo, en la definición de la función debemos poner:

CREATE FUNCTION nombreFuncion() RETURNS tipoDato **DETERMINISTIC**

Por defecto, tanto los procedimientos como las funciones, son consideradas como NO DETERMINISTAS.

Otros modificadores que podemos añadir a la función y que van después del tipo de dato son:

* CONTAINS SQL: La función o procedimiento no contiene sentencias SQL que lean (como SELECT) o escriban datos (como INSERT).
* NO SQL: La función o procedimiento no contiene sentencias SQL.
* READS SQL DATA: La función o procedimiento contiene sentencias que leen datos (como SELECT) pero no sentencias que modifiquen datos.
* MODIFIES SQL DATA: La función o procedimiento contiene sentencias que modifican datos (como INSERT, DELETE, UPDATE).

**Importante:**

Para comprobar si el registro binario de log está activado podemos ejecutar la orden: SELECT @@log\_bin;

En caso de estarlo (devuelve el valor 1 la llamada anterior), debemos de realizar alguna de las siguientes acciones:

* Añadir a la definición de la función al menos uno de los modificadores siguientes: DETERMINISTIC, NO SQL o READS SQL DATA.
* Desactivar el registro binario de log.
* Modificar la variable del sistema log\_bin\_trust\_function\_creators con la orden: SET GLOBAL log\_bin\_trust\_function\_creators = 1;

## **Aclaración 2 - Sobre Error code 1267. Illegal mix of collations**

A la hora de realizar comparaciones de cadena empleando funciones (por ejemplo en la cláusula WHERE al comparar una expresión con un valor concreto), Mysql puede dar un error de tipo **Error code 1267. Illegal mix of collations ...**.

Por ejemplo: SELECT \* FROM Tabla WHERE funcion(col1) = 'VALOR'

Esto es debido a que la expresión que devuelve Mysql al emplear la función tendrá el mismo [Collate](https://dev.mysql.com/doc/refman/8.0/en/charset-mysql.html) que el definido en la tabla mientras que a la cadena con la que estamos comparando Mysql no le da el mismo collate.

Para eliminar este error, tendremos que hacer uso de la [orden COLLATE](https://dev.mysql.com/doc/refman/8.0/en/charset-collate.html) de la forma:

SELECT \* FROM Tabla WHERE funcion(col1) = 'VALOR' **COLLATE collate\_de\_la\_tabla**

Para saber que COLLATE tiene una tabla, podemos ejecutar la orden 'SHOW CREATE TABLE nombre\_tabla;' o ir desde MysqlWorkBench a la información de la tabla.

# Ejercicios propuestos Funciones

Nota: Después de crear la función llama a la misma de diferentes formas (dentro de una orden SQL o directamente).

## **Ejercicio 1**

Crea una función de nombre utilidades\_getMesEnLetra a la que se le pase un número y devuelva el nombre del mes. En caso de que el número no se corresponda con ningún mes, debe devolver null.

Fijarse que esta función es determinista.

* Llama a la función directamente y guarda el resultado en una variable de sesión.
* Llama a la función para que muestre los meses en letra en los que se celebró la atracción 'El gran felino'.
* Llama a la función para que muestre las atracciones que se celebraron en ABRIL (busca por la cadena ABRIL) (recuerda hacer uso de COLLATE).

Nota: Indicar que Mysql ya dispone de dicha función, a la que se le pasa una fecha y devuelve el mes en forma de cadena: [monthname()](https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html" \l "function_monthname).

## **Ejercicio 2**

Modifica un ejercicio anterior en el que creamos el procedimiento de nombre 'atracciones\_getNumPorMes' y crea una función de nombre utilidades\_getMesEnNumero a la que se le pase el nombre de un mes y devuelva el número que se corresponde con ese nombre. Si el mes no existe debe devolver -1. Modifica el procedimiento para que haga uso de la función.

## **Ejercicio 3**

Crea una función de nombre animales\_getEstadoPorAnhos que devuelva la cadena:

* Si tipo = León
* anhos < 2: 'JOVEN'
* anhos >=2 y <=5: 'MADURO'
* anhos > 5: 'VIEJO'
* Cualquier otro tipo:
* anhos < 1: 'JOVEN'
* anhos >=1 y <=3: 'MADURO'
* anhos > 3: 'VIEJO'

Llama a la función para mostrar el estado por años de cada uno de los animales del CIRCO.