

**PRACTICA3 - Funciones & *Triggers***

**Nombre**: Marco Antonio Roca Montenegro.

**Institución**: Universidad Privada Boliviana.

**Materia**: Bases de datos Avanzadas.

**Docente**: Msc. Carlos Aníbarro.

**Fecha:** 12/06/2023.

Contenido

[INTRODUCCIÓN 3](#_Toc137423349)

[Función “clientesSucursal” 3](#_Toc137423350)

[Función “eval\_salario” 4](#_Toc137423351)

[Función “miFactorial” 5](#_Toc137423352)

[Trigger “datosNotas” 6](#_Toc137423353)

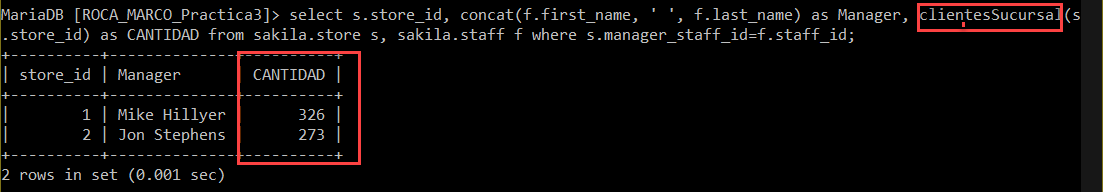
# INTRODUCCIÓN

El presente documento contiene las pruebas realizadas con respecto a los procedimientos y funciones implementados dentro de la base de datos ROCA\_MARCO\_Practica3, realizado en MariaDB – Versión: 15.1.

# Función “clientesSucursal”

La función “clientesSucursal” tiene como objetivo el mostrar la cantidad de clientes por sucursal de la tabla “**sakila.store”** (únicamente 2 tiendas). La base de datos “**sakila**” cuenta con una tabla “**customer**”, por lo cual para saber la cantidad la función hace uso de un “**count**” tomando en cuenta el id de la tienda como parámetro de igualdad. En conclusión, nos retorna una respuesta con tipo de dato ***INT***.

* **Para su uso:** select s.store\_id, concat(f.first\_name, ' ', f.last\_name) as Manager, **clientesSucursal(s.store\_id)** as CANTIDAD from sakila.store s, sakila.staff f where s.manager\_staff\_id=f.staff\_id;



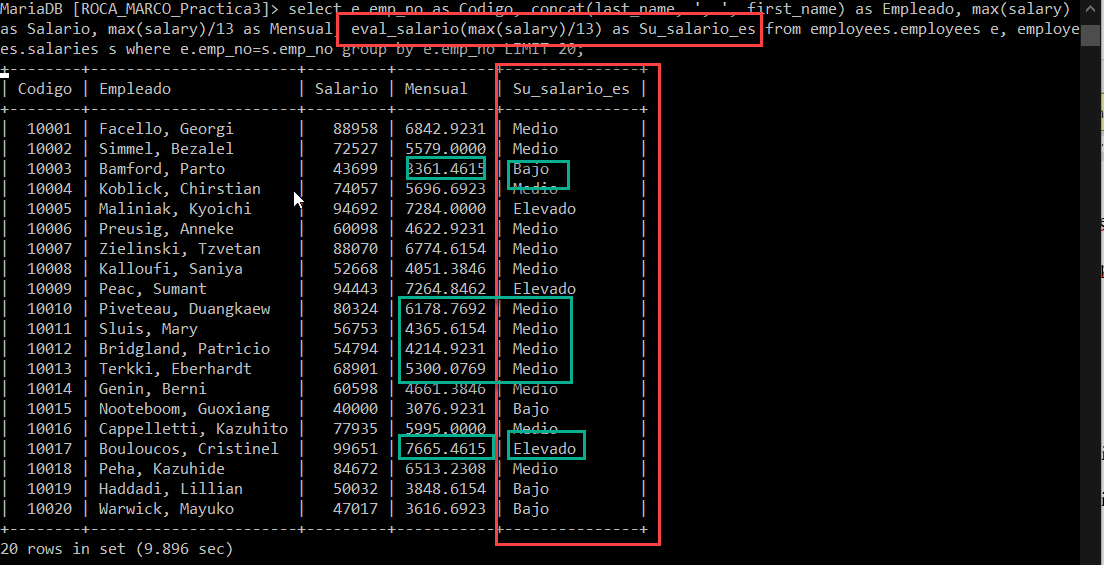
# Función “eval\_salario”

Recibe como parámetro un dato de tipo ***float***, el cual es una cantidad la cual en la práctica denomina la evaluación de un salario (evaluamos el salario mensual). Lo que nos debe retornar es un mensaje el cual responde a:

Si el salario es < 4000 ? ‘Bajo’ : salario >=4000 AND salairo <=7000 ? ‘Medio’ : ‘Elevado’.

Adicionalmente se establece un “**LIMIT 20**”, para controlar que se muestre en consola únicamente 20 resultados.

* **Para su uso:** select e.emp\_no as Codigo, concat(last\_name, ', ', first\_name) as Empleado, max(salary) as Salario, max(salary)/13 as Mensual, eval\_salario(max(salary)/13) as Su\_salario\_es from employees.employees e, employees.salaries s where e.emp\_no=s.emp\_no group by e.emp\_no LIMIT 20;

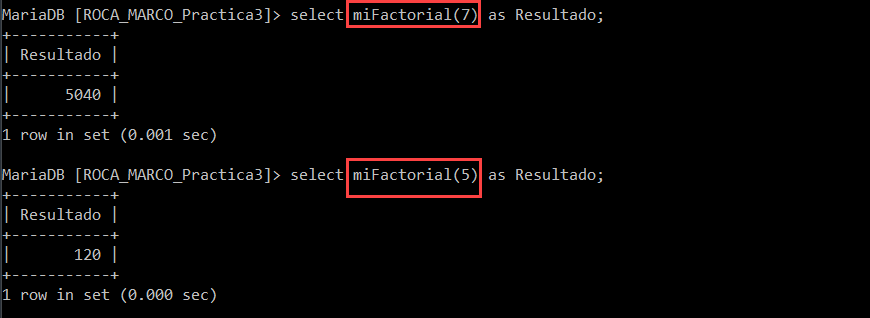


# Función “miFactorial”

Calcula el factorial del parámetro el cual propiciemos (tipo de entrada y salida ***INT***). Internamente se maneja una variable de **iteración** = al parámetro de entrada y una variable denominada **resultado** (definida con valor 1). Luego se realiza un bucle ***WHILE***, multiplica resultado = 1 \* el valor de iteración, luego restamos iteración - 1 siempre y cuando iteración > 1.

Con esto, una vez iteración sea igual a 1 se rompe el ciclo WHILE (Considerando que el primer paso es multiplicar por un valor 1).

* **Para su uso:** select miFactorial(7) as Resultado;



# Trigger “datosNotas”

El siguiente trigger realiza un control de inserción de notas. Entonces el disparador se ejecuta **DESPUES** de haber hecho un insert. Si la nota es menor a 0 y mayor a 100, El disparador (***AFTER INSERT***) realiza un ***DELETE*** de los datos ingresados.

**Datos de prueba:**

insert into notas values(5,'roca','piter','biologia',90);

insert into notas values(6,'roca','marco','fisica',40);

insert into notas values(7,'roca','marco','cultura',60);

insert into notas values(8,'roca','marco','deporte',10);

insert into notas values(9,'roca','marco','canto',70);

insert into notas values(10,'roca','marco','musica',90);

insert into notas values(11,'roca','marco','fisica',-120);

insert into notas values(12,'roca','marco','cultura',-20);

insert into notas values(13,'roca','marco','deporte',0);

insert into notas values(14,'roca','marco','canto',120);

insert into notas values(15,'roca','marco','musica',190);

