

Aplicación de Procedimientos Almacenados

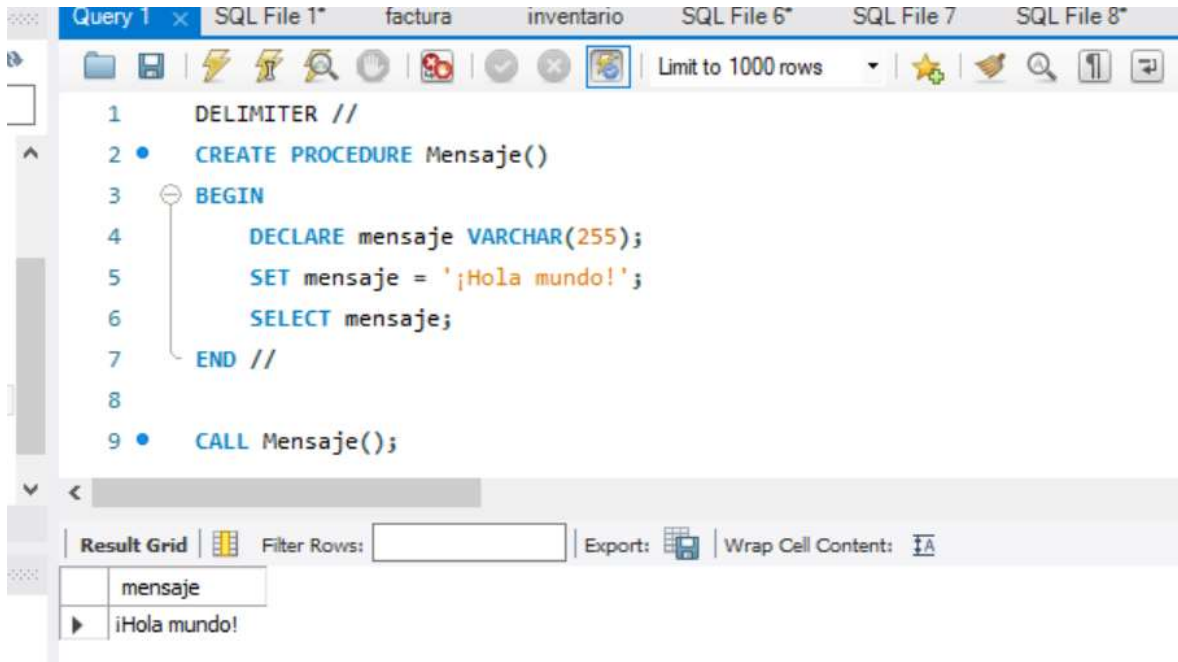
Elaborado por:
María Fernanda Rios Rodriguez

Entregado a:
Mary Luz Rubiano

Universidad de San Buenaventura, Bogotá
Facultad de Ingeniería
Tecnología en Desarrollo de Software
2024

Taller – Procedimientos almacenados

1. Escribe un procedimiento que no tenga ningún parámetro de entrada ni de salida y que muestre el texto ¡Hola mundo!

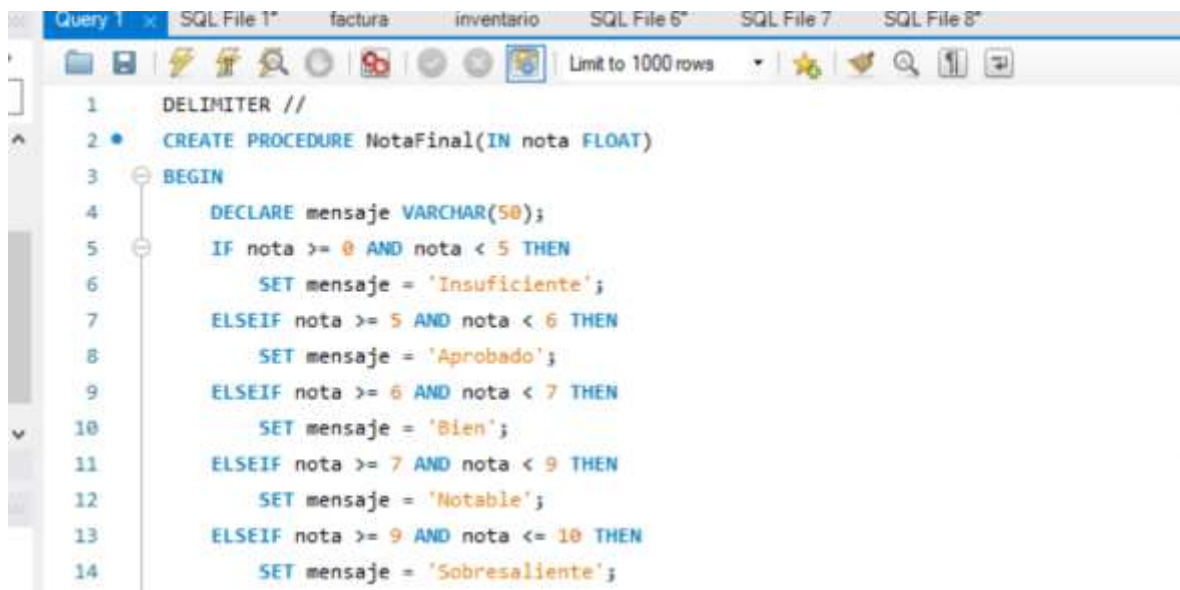


```
1 DELIMITER //
2 CREATE PROCEDURE Mensaje()
3 BEGIN
4     DECLARE mensaje VARCHAR(255);
5     SET mensaje = '¡Hola mundo!';
6     SELECT mensaje;
7 END //
8
9 CALL Mensaje();
```

Result Grid

| mensaje |
|--------------|
| ¡Hola mundo! |

2. Escribe un procedimiento que reciba un número real de entrada, que representa el valor de la nota de un alumno, y muestre un mensaje indicando qué nota ha obtenido teniendo en cuenta las siguientes condiciones: [0,5) = Insuficiente [5,6) = Aprobado [6, 7) = Bien [7, 9) = Notable [9, 10] = Sobresaliente En cualquier otro caso la nota no será válida.



```
1 DELIMITER //
2 CREATE PROCEDURE NotaFinal(IN nota FLOAT)
3 BEGIN
4     DECLARE mensaje VARCHAR(50);
5     IF nota >= 0 AND nota < 5 THEN
6         SET mensaje = 'Insuficiente';
7     ELSEIF nota >= 5 AND nota < 6 THEN
8         SET mensaje = 'Aprobado';
9     ELSEIF nota >= 6 AND nota < 7 THEN
10        SET mensaje = 'Bien';
11    ELSEIF nota >= 7 AND nota < 9 THEN
12        SET mensaje = 'Notable';
13    ELSEIF nota >= 9 AND nota <= 10 THEN
14        SET mensaje = 'Sobresaliente';
```

```
16     ELSE
17         SET mensaje = 'La Nota no es válida';
18     END IF;
19     SELECT mensaje AS 'Resultado';
20 END //
21
22 • CALL NotaFinal(9);
```

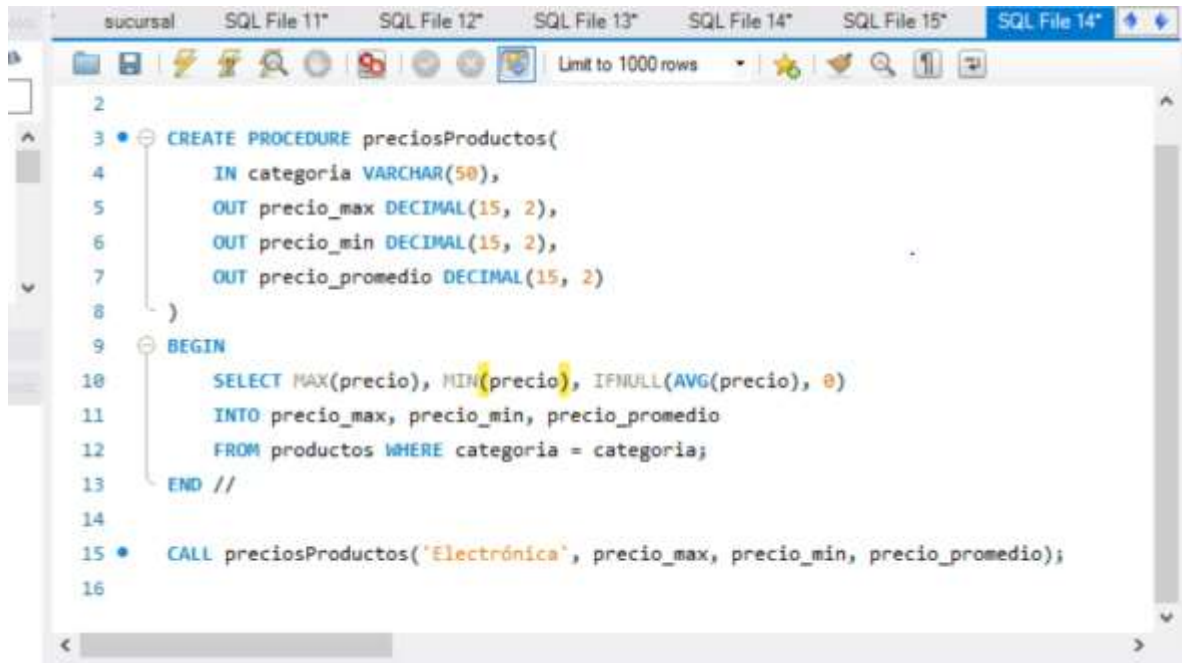
Result Grid

| Resultado |
|---------------|
| Sobresaliente |

3. Escriba un procedimiento llamado **cantidadProductos** que reciba como entrada el nombre del tipo de producto y devuelva el número de productos que existen dentro de esa categoría.

```
1 DELIMITER //
2
3 • CREATE PROCEDURE cantidadProductos(IN tipoproductos VARCHAR(100), OUT cantidad INT)
4 BEGIN
5     SELECT COUNT(*) INTO cantidad
6     FROM productos
7     WHERE tipo = tipoproductos;
8 END //
9 • CALL cantidadProductos (frutas, cantidad);
10
```

4. Escribe un procedimiento que se llame preciosProductos, que reciba como parámetro de entrada el nombre del tipo de producto y devuelva como salida tres parámetros. El precio máximo, el precio mínimo y la media de los productos que existen en esa categoría.



```
2
3 • CREATE PROCEDURE preciosProductos(
4     IN categoria VARCHAR(50),
5     OUT precio_max DECIMAL(15, 2),
6     OUT precio_min DECIMAL(15, 2),
7     OUT precio_promedio DECIMAL(15, 2)
8 )
9 BEGIN
10     SELECT MAX(precio), MIN(precio), IFNULL(AVG(precio), 0)
11     INTO precio_max, precio_min, precio_promedio
12     FROM productos WHERE categoria = categoria;
13 END //
14
15 • CALL preciosProductos('Electrónica', precio_max, precio_min, precio_promedio);
16
```

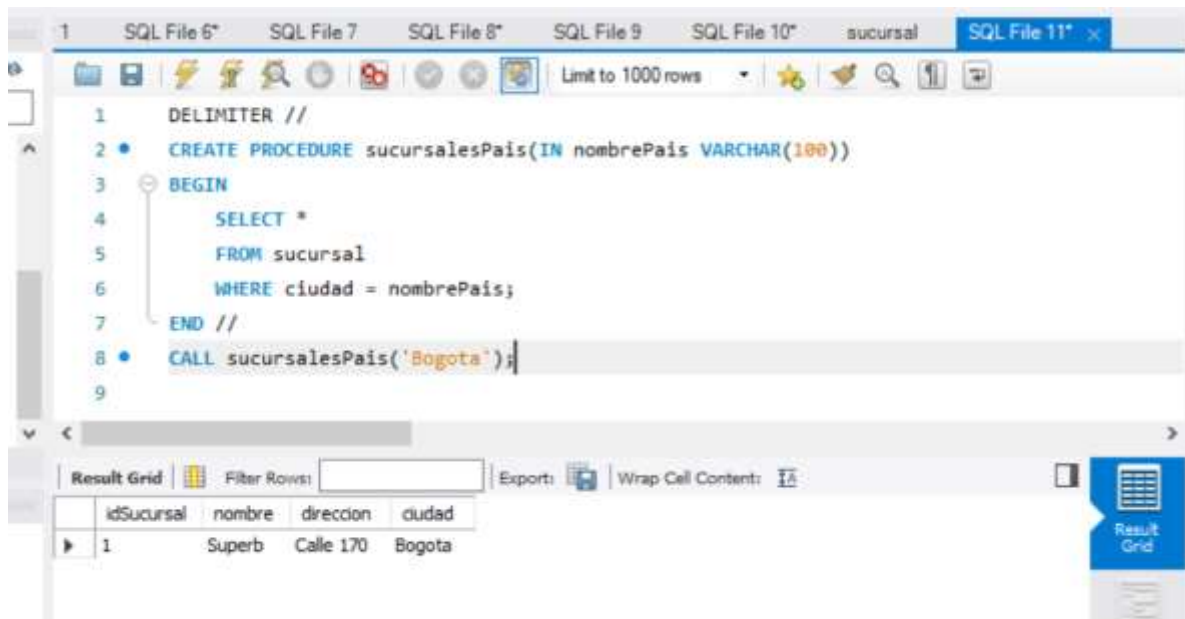
5. Realice un procedimiento que se llame funcionIVA que incluya una función que calcule el total con el incremento del iva.



```
1 DELIMITER //
2
3 • CREATE PROCEDURE funcionIVA(
4     IN subtotal DECIMAL(15, 2),
5     OUT totalIva DECIMAL(15, 2)
6 )
7 BEGIN
8     DECLARE iva DECIMAL(15, 2);
9
10
11     SET iva = 0.19;
12     SET totalIva = calcularTotalConIVA(subtotal, iva);
13 END //
```

```
14
15 • CREATE FUNCTION calcularTotalIVA(subtotal DECIMAL(15, 2), iva DECIMAL(15, 2))
16 RETURNS DECIMAL(15, 2)
17 BEGIN
18     DECLARE total DECIMAL(15, 2);
19     SET total = subtotal + (subtotal * iva);
20
21     RETURN total;
22 END //
```

6. Escribe un procedimiento que reciba el nombre de un país como parámetro de entrada y realice una consulta sobre la tabla sucursal para obtener todos las sucursales que existen en la tabla de ese país.



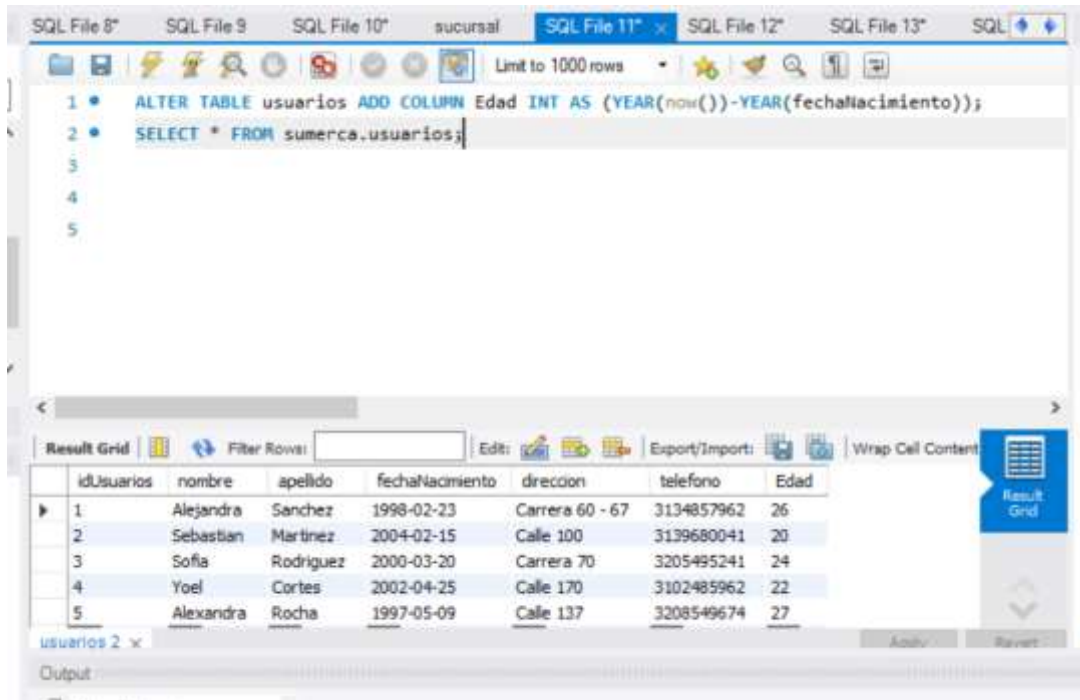
The screenshot shows a SQL IDE with multiple tabs. The active tab is 'SQL File 11*' containing the following SQL code:

```
1 DELIMITER //
2 • CREATE PROCEDURE sucursalesPais(IN nombrePais VARCHAR(100))
3 BEGIN
4     SELECT *
5     FROM sucursal
6     WHERE ciudad = nombrePais;
7 END //
8 • CALL sucursalesPais('Bogota');
9
```

Below the code editor, the 'Result Grid' is displayed, showing the results of the procedure call. The grid has columns: idSucursal, nombre, direccion, ciudad. The first row shows the result for Bogota.

| idSucursal | nombre | direccion | ciudad |
|------------|--------|-----------|--------|
| 1 | Superb | Calle 170 | Bogota |

7. Una vez creada la tabla se decide añadir una nueva columna a la tabla llamada edad que será un valor calculado a partir de la columna fecha_nacimiento. Escriba la sentencia SQL necesaria para modificar la tabla y añadir la nueva columna.



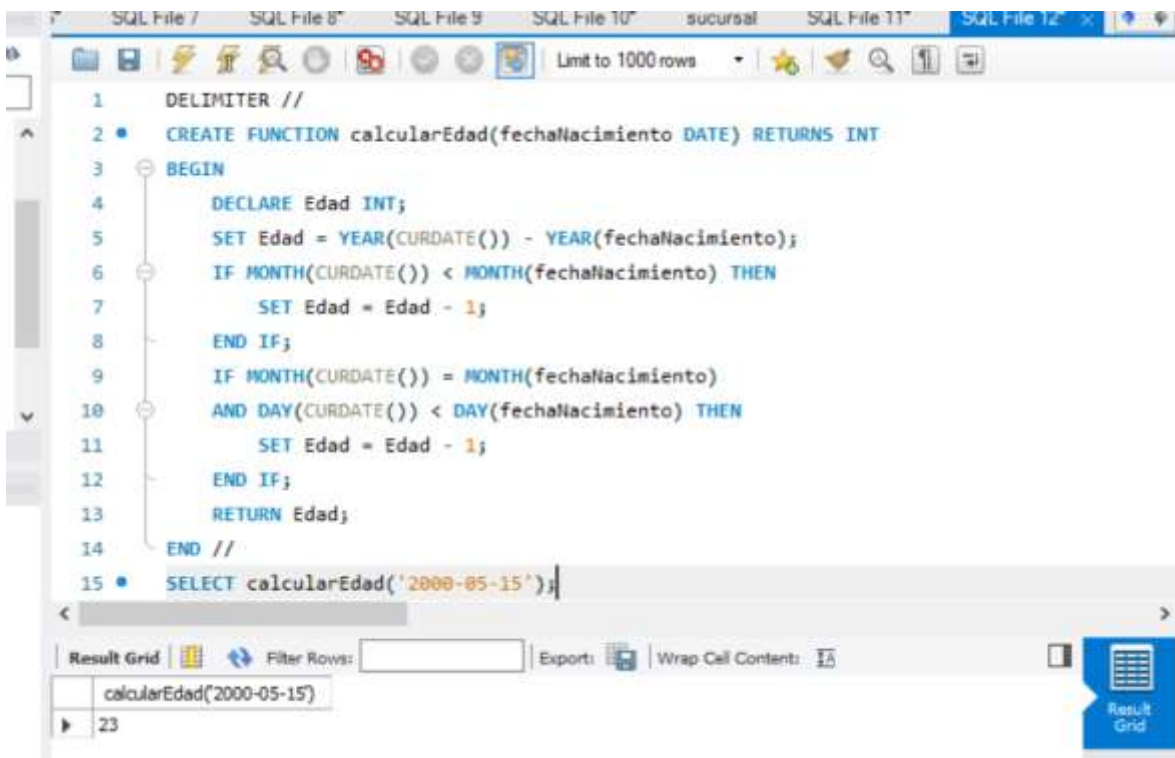
```

1 • ALTER TABLE usuarios ADD COLUMN Edad INT AS (YEAR(now())-YEAR(fechaNacimiento));
2 • SELECT * FROM sumerca.usuarios;
3
4
5

```

| idUsuarios | nombre | apellido | fechaNacimiento | direccion | telefono | Edad |
|------------|-----------|-----------|-----------------|-----------------|------------|------|
| 1 | Alejandra | Sanchez | 1998-02-23 | Carrera 60 - 67 | 3134857962 | 26 |
| 2 | Sebastian | Martinez | 2004-02-15 | Calle 100 | 3139680041 | 20 |
| 3 | Sofia | Rodriguez | 2000-03-20 | Carrera 70 | 3205495241 | 24 |
| 4 | Yoel | Cortes | 2002-04-25 | Calle 170 | 3102485962 | 22 |
| 5 | Alexandra | Rocha | 1997-05-09 | Calle 137 | 3208549674 | 27 |

8. Escriba una función llamada **calcularEdad** que reciba una fecha y devuelva el número de años que han pasado desde la fecha actual hasta la fecha pasada como parámetro.



```

1 DELIMITER //
2 • CREATE FUNCTION calcularEdad(fechaNacimiento DATE) RETURNS INT
3 BEGIN
4     DECLARE Edad INT;
5     SET Edad = YEAR(CURDATE()) - YEAR(fechaNacimiento);
6     IF MONTH(CURDATE()) < MONTH(fechaNacimiento) THEN
7         SET Edad = Edad - 1;
8     END IF;
9     IF MONTH(CURDATE()) = MONTH(fechaNacimiento)
10    AND DAY(CURDATE()) < DAY(fechaNacimiento) THEN
11         SET Edad = Edad - 1;
12    END IF;
13    RETURN Edad;
14 END //
15 • SELECT calcularEdad('2000-05-15');

```

| calcularEdad('2000-05-15') |
|----------------------------|
| 23 |

9. Escriba un procedimiento que permita calcular la edad de todos los usuarios que ya existen en la tabla. Para esto será necesario crear un procedimiento llamado actualizarColumnaEdad que calcule la edad de cada usuario y actualice la tabla. Este procedimiento hará uso de la función calcularEdad que hemos creado en el paso anterior

```
SQL File 14*  SQL File 15*  SQL File 14*  SQL File 15*  SQL File 16*  SQL File 17*  usuarios
Limit to 1000 rows

1 DELIMITER //
2 • CREATE PROCEDURE actualizarColumnaEdad()
3 BEGIN
4     DECLARE done INT DEFAULT FALSE;
5     DECLARE userId INT;
6     DECLARE userFechaNacimiento DATE;
7     DECLARE cur CURSOR FOR SELECT id, fechaNacimiento FROM usuarios;
8     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
9
10    OPEN cur;
11    read_loop: LOOP
12        FETCH cur INTO userId, userFechaNacimiento;
13        IF done THEN
14            LEAVE read_loop;
15        END IF;
16        UPDATE usuarios
17        SET Edad = calcularEdad(userFechaNacimiento)
18        WHERE id = userId;
19    END LOOP;
20    CLOSE cur;
21 END//
22 • CALL actualizarColumnaEdad();
```

SQL File 14* SQL File 15* SQL File 14* SQL File 15* SQL File 16* SQL File 17* usuarios

Limit to 1000 rows

```
1 • SELECT * FROM sumerca.usuarios;
```

| idUsuarios | nombre | apellido | fechaNacimiento | direccion | telefono | Edad |
|------------|-----------|-----------|-----------------|-----------------|------------|------|
| 1 | Alejandra | Sanchez | 1998-02-23 | Carrera 60 - 67 | 3134857962 | 23 |
| 2 | Sebastian | Martinez | 2004-02-15 | Calle 100 | 3139680041 | 31 |
| 3 | Sofia | Rodriguez | 2000-03-20 | Carrera 70 | 3205495241 | |
| 4 | Yoel | Cortes | 2002-04-25 | Calle 170 | 3102485962 | |
| 5 | Alexandra | Rocha | 1997-05-09 | Calle 137 | 3208549674 | |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content

Result Grid

Form Editor

10. Escribe un procedimiento almacenado para su proyecto integrador que sea útil.

```

53 • #HECHO POR DALLIAT
54 CREATE PROCEDURE agregarProducto(
55     IN p_nombre VARCHAR(255),
56     IN p_descripcion TEXT,
57     IN p_precio DECIMAL(10, 2),
58     IN p_categoria_id INT,
59     IN p_vendedor_id INT
60 )
61 BEGIN
62     INSERT INTO productos (nombre, descripcion, precio, categoria_id, vendedor_id)
63     VALUES (p_nombre, p_descripcion, p_precio, p_categoria_id, p_vendedor_id);
64
65     SELECT LAST_INSERT_ID() AS ProductoID;
66 END //
67
68 DELIMITER ;
69
70 • CALL agregarProducto(
71     'iphone 15 pro max',
72     'celular con muy buena camara y con 126 gb',
73     3500000,
74     1,
75     1
76 );

```

```

1 • SELECT * FROM dalliat.productos;

```

| Result Grid | | | | | | |
|--------------------|------|-------------------|---|------------|--------------|-------------|
| Filter Rows: | | | | | | |
| Edit: | | | | | | |
| Export/Import: | | | | | | |
| Wrap Cell Content: | | | | | | |
| | id | nombre | descripcion | precio | categoria_id | vendedor_id |
| ▶ | 1 | Smartphone XYZ | Un smartphone de última generación | 799.99 | 1 | 1 |
| | 2 | Laptop ABC | Una laptop potente | 1199.99 | 1 | 2 |
| | 3 | iphone 15 pro max | celular con muy buena camara y con 126 gb | 3500000.00 | 1 | 1 |
| * | NULL | NULL | NULL | NULL | NULL | NULL |