



# ESCUELA POLITÉCNICA NACIONAL

## ESCUELA DE FORMACIÓN DE TECNÓLOGOS



### BASE DE DATOS

PROFESOR:

Ing. Yadira Franco R

PERÍODO ACADÉMICO:

2024-B

### TAREA

TÍTULO:

INVESTIGACIÓN Y PRACTICA



Estudiante

Edwin Sarango

2024-B

## INVESTIGAR QUE SON Procedimientos Almacenados en Bases de Datos

- Entender qué son los procedimientos almacenados y cómo funcionan.
- Aprender a crear procedimientos almacenados sencillos.
- PRACTICA - Realizar operaciones de **INSERT**, **SELECT**, **DELETE** y **UPDATE** usando procedimientos almacenados.
- **Revisión de Buenas Prácticas**

## Introducción a los Procedimientos Almacenados **MSQL- PostgreSQL – Sql Server**

### 1. Concepto y Beneficios de los Procedimientos Almacenados

- **Explicación:** Los procedimientos almacenados son conjuntos de instrucciones SQL que se guardan y ejecutan en el servidor de base de datos. Permiten ejecutar operaciones complejas, con seguridad, rendimiento optimizado y reutilización de código.
- **Beneficios:**
  - Reutilización de código.
  - Mejora en la seguridad (al evitar inyecciones SQL).
  - Optimización en el rendimiento de consultas frecuentes.
  - Consistencia en las operaciones realizadas.

```
CREATE PROCEDURE Nombre_Procedimiento (parámetros)
BEGIN
    -- Instrucciones SQL aquí
END //

DELIMITER ;
```

### 2. ESPECIFICAR LA Sintaxis Básica de un Procedimiento Almacenado

- **Explicación:** El delimitador se cambia temporalmente para permitir el uso de **;** dentro del procedimiento.

#### Crear la tabla de cliente:

```
CREATE TABLE cliente (
```

```
    ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID único del cliente
```

```
    Nombre VARCHAR(100), -- Campo para el nombre del cliente
```

Estatura DECIMAL(5,2),                      -- Campo para la estatura del cliente con dos decimales  
 FechaNacimiento DATE,                      -- Campo para la fecha de nacimiento del cliente  
 Sueldo DECIMAL(10,2)                      -- Campo para el sueldo del cliente con dos decimales  
 );

```

CREATE TABLE cliente (
  ClienteID INT AUTO_INCREMENT PRIMARY KEY,
  Nombre VARCHAR(100),
  Estatura DECIMAL(5,2),
  FechaNacimiento DATE,
  Sueldo DECIMAL(10,2)
);
  
```

Output		
Action Output		
#	Time	Action
✓ 1	21:25:57	create database Tienda
✓ 2	21:26:00	use Tienda
✓ 3	21:26:16	CREATE TABLE cliente ( ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID único del cliente...

### 3. Ejercicio 1: Crear un procedimiento simple que seleccione datos de la tabla cliente

#### 4. Ejercicio: Ejecutar - LLAMAR el procedimiento

### Inserción, Actualización y Eliminación de Datos

#### 1. Procedimiento de Inserción (INSERT)

- Crear un procedimiento que permita insertar un nuevo cliente en la tabla cliente

- Ejecutar - LLAMAR el procedimiento

```

3 DELIMITER //
4
5 CREATE PROCEDURE SeleccionarClientes()
6 BEGIN
7   SELECT * FROM cliente;
8 END //
9
10 DELIMITER ;
  
```

```

20 DELIMITER ;
21 • CALL SeleccionarClientes();
22
Result Grid | Filter Rows: | Export: | W
+-----+-----+-----+-----+-----+
| ClienteID | Nombre | Estatura | FechaNacimiento | Sueldo |
+-----+-----+-----+-----+-----+

DELIMITER //

CREATE PROCEDURE InsertarCliente(
    IN p_Nombre VARCHAR(100),
    IN p_Estatura DECIMAL(5,2),
    IN p_FechaNacimiento DATE,
    IN p_Sueldo DECIMAL(10,2)
)
BEGIN
    INSERT INTO cliente (Nombre, Estatura, FechaNacimiento, Sueldo)
    VALUES (p_Nombre, p_Estatura, p_FechaNacimiento, p_Sueldo);
END //

DELIMITER ;
36 DELIMITER ;
37 • CALL InsertarCliente('Juan Pérez', 1.75, '1990-05-15', 30000.50);
38

```

## 2. Procedimiento de Actualización (UPDATE)

Actualizar la edad de un cliente específico:

```

-- update
DELIMITER //

CREATE PROCEDURE ActualizarSueldo(
    IN p_ClienteID INT,
    IN p_NuevoSueldo DECIMAL(10,2)
)
BEGIN
    UPDATE cliente
    SET Sueldo = p_NuevoSueldo
    WHERE ClienteID = p_ClienteID;
END //

DELIMITER ;

52 DELIMITER ;
53 • CALL ActualizarSueldo(1, 35000.00);
54

```

## 3. Procedimiento de Eliminación (DELETE)

Eliminar un cliente de la base de datos usando su ClienteID:

```

DELIMITER //

CREATE PROCEDURE EliminarCliente(
    IN p_ClienteID INT
)
BEGIN
    DELETE FROM cliente
    WHERE ClienteID = p_ClienteID;
END //

DELIMITER ;

```

66 DELIMITER ;  
67 • CALL EliminarCliente(1);  
68  
69

Output  
Action Output  

#	Time	Action
11	21:40:46	CREATE PROCEDURE ActualizarSue
12	21:40:50	CALL ActualizarSueldo(1, 35000.00)
13	21:41:45	CREATE PROCEDURE EliminarClient
14	21:41:48	CALL EliminarCliente(1)

## Introducción a Condiciones en Procedimientos Almacenados

### Uso de Condicionales (IF)

El uso de condicionales dentro de los procedimientos es fundamental para tomar decisiones basadas en los datos.

Verifica si la edad de un cliente es mayor o igual a 22:

### Creación de la Tabla de Órdenes CON RELACIÓN CON EL CLIENTE - FORANEA

Para almacenar las órdenes de los clientes, se debe crear la tabla **ordenes**:

- Procedimientos de Órdenes -Insertar Orden
- Procedimientos Actualizar Orden
- Procedimientos Eliminar Orden

```

72 DELIMITER //
73
74 • CREATE FUNCTION CalcularEdad(p_FechaNacimiento DATE)
75 RETURNS INT
76 DETERMINISTIC
77 BEGIN
78     RETURN TIMESTAMPDIFF(YEAR, p_FechaNacimiento, CURDATE());
79 END //
80
81 DELIMITER ;
82

```

DELIMITER //

```
CREATE PROCEDURE VerificarEdadCliente(  
    IN p_ClienteID INT  
)  
BEGIN  
    DECLARE edad INT;  
    SELECT CalcularEdad(FechaNacimiento) INTO edad  
    FROM cliente  
    WHERE ClienteID = p_ClienteID;  
  
    IF edad >= 22 THEN  
        SELECT CONCAT('El cliente con ID ', p_ClienteID, ' tiene ', edad, ' años, y es mayor o igual a 22.');    ELSE  
        SELECT CONCAT('El cliente con ID ', p_ClienteID, ' tiene ', edad, ' años, y es menor de 22.');    END IF;  
END //
```

DELIMITER ;

```
103  
104 • CALL VerificarEdadCliente(1);  
105
```

Result Grid	Filter Rows:	Export
CONCAT('El cliente con ID ', p_ClienteID, ' tiene ', edad, ' años, y es menor de 22.')		
NULL		

```
CREATE TABLE ordenes (  
    OrdenID INT AUTO_INCREMENT PRIMARY KEY, -- ID único de la orden  
    ClienteID INT, -- ID del cliente (relación)  
    FechaOrden DATE, -- Fecha de la orden  
    MontoTotal DECIMAL(10,2), -- Monto total de la orden  
    Estado VARCHAR(50), -- Estado de la orden (ej: 'Pendiente', 'Completada')  
    FOREIGN KEY (ClienteID) REFERENCES cliente(ClienteID) -- Relación con la tabla cliente  
);
```

```
CREATE PROCEDURE InsertarOrden(  
    IN p_ClienteID INT,  
    IN p_FechaOrden DATE,  
    IN p_MontoTotal DECIMAL(10,2),  
    IN p_Estado VARCHAR(50)  
)  
BEGIN  
    INSERT INTO ordenes (ClienteID, FechaOrden, MontoTotal, Estado)  
    VALUES (p_ClienteID, p_FechaOrden, p_MontoTotal, p_Estado);  
END //
```

DELIMITER ;

```
28 DELIMITER ;  
29 • CALL InsertarOrden(1, '2024-12-17', 500.00, 'Pendiente');  
30
```

```

DELIMITER //

CREATE PROCEDURE ActualizarOrden(
    IN p_OrdenID INT,
    IN p_NuevoEstado VARCHAR(50)
)
BEGIN
    UPDATE ordenes
    SET Estado = p_NuevoEstado
    WHERE OrdenID = p_OrdenID;
END //

DELIMITER ;

CALL ActualizarOrden(1, 'Completada');

```

```

149 DELIMITER //
150
151 CREATE PROCEDURE EliminarOrden(
152     IN p_OrdenID INT
153 )
154 BEGIN
155     DELETE FROM ordenes
156     WHERE OrdenID = p_OrdenID;
157 END //
158
159 DELIMITER ;
160 CALL EliminarOrden(1);

```

Output			
Action Output			
#	Time	Action	
✓ 28	21:51:44	CREATE PROCEDURE ActualizarOrden( IN p_OrdenID INT, IN p_NuevoEstado VARCHAR(50) ) BEGIN U...	
✓ 29	21:51:46	CALL ActualizarOrden(1, 'Completada')	
✓ 30	21:52:32	CREATE PROCEDURE EliminarOrden( IN p_OrdenID INT ) BEGIN DELETE FROM ordenes WHERE Orde...	
✓ 31	21:52:35	CALL EliminarOrden(1)	

## Entrega Final

### Instrucciones de Entrega:

#### 1. Objetivos:

Crear procedimientos almacenados para **insertar, actualizar, eliminar y consultar** registros en las tablas cliente y ordenes.

#### 2. Archivo de Script:

Los estudiantes deben escribir y guardar el código SQL con todos los procedimientos mencionados.

#### 3. Documento PDF:

Incluir las capturas de pantalla y explicaciones detalladas de los pasos realizados durante la tarea.

4. **Subida a GitHub:**

Subir el script .sql y el documento PDF a un repositorio en GitHub para su REVISIÓN