



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

XXXXXXXXXXXXXXXXXX

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.

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  
);
```

```
4 • CREATE TABLE cliente (  
5     ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID único del cliente  
6     Nombre VARCHAR(100), -- Campo para el nombre del cliente  
7     Estatura DECIMAL(5,2), -- Campo para la estatura del cliente con dos decimales  
8     FechaNacimiento DATE, -- Campo para la fecha de nacimiento del cliente  
9     Sueldo DECIMAL(10,2) -- Campo para el sueldo del cliente con dos decimales  
10 );  
11
```

Output

#	Time	Action	Message
✓ 1	18:25:11	CREATE DATABASE tarea	1 row(s) affected
✓ 2	18:25:17	USE tarea	0 row(s) affected
✓ 3	18:28:43	CREATE TABLE cliente (ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID ...	0 row(s) affected

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

```
12 -- PROCEDIMIENTO PARA SELECCIONAR TODOS LOS DATOS DE LA TABLA  
13 DELIMITER //  
14 • CREATE PROCEDURE verDatos()  
15 BEGIN  
16     SELECT * FROM cliente;  
17 END //  
18 DELIMITER ;  
19
```

Output

#	Time	Action
✓ 1	18:33:39	CREATE PROCEDURE verDatos() BEGIN SELECT * FROM cliente; END

4. Ejercicio: Ejecutar - LLAMAR el procedimiento

```

20 • -- LLAMAR AL PROCEDIMIENTO
21 CALL verDatos();

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	ClienteID	Nombre	Estatura	FechaNacimiento	Sueldo
--	-----------	--------	----------	-----------------	--------

Result 1 x

Output

Action Output

#	Time	Action
✓ 1	18:33:39	CREATE PROCEDURE verDatos() BEGIN SELECT * FROM cliente; END
✓ 2	18:34:26	CALL verDatos()

Inserción, Actualización y Eliminación de Datos

1. Procedimiento de Inserción (INSERT)

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

```

29 DELIMITER //
30 • CREATE PROCEDURE agregarCliente(
31     IN nombre VARCHAR(100),
32     IN estatura DECIMAL(5,2),
33     IN fechaNacimiento DATE,
34     IN sueldo DECIMAL(10,2)
35 )
36 BEGIN
37     INSERT INTO cliente (Nombre, Estatura, FechaNacimiento, Sueldo)
38     VALUES (nombre, estatura, fechaNacimiento, sueldo);
39 END //
40 DELIMITER ;
41 • -- LLAMAR AL PROCEDIMIENTO

```

Output

Action Output

#	Time	Action
1	19:56:12	CREATE PROCEDURE agregarCliente(IN nombre VARCHAR(100), IN estatura DECIMAL(5,2...

- Ejecutar - LLAMAR el procedimiento

```
41 -- LLAMAR AL PROCEDIMIENTO
42 CALL agregarCliente('Kevin Flores', 1.75, '2003-05-15', 980.50);
```

Output

#	Time	Action
1	19:56:12	CREATE PROCEDURE agregarCliente(IN nombre VARCHAR(100), IN estatura DECIMAL(5,2,
2	19:56:38	CALL agregarCliente('Kevin Flores', 1.75, '2003-05-15', 980.50)

2. Procedimiento de Actualización (UPDATE)

Actualizar la edad de un cliente específico:

```
44 -- CREAR EL PROCEDIMIENTO PARA ACTUALIZAR LA FECHA DE NACIMIENTO
45 DELIMITER //
46 CREATE PROCEDURE actualizarEdadCliente(
47     IN clienteID INT,
48     IN nuevaFechaNacimiento DATE
49 )
50 BEGIN
51     UPDATE cliente
52     SET FechaNacimiento = nuevaFechaNacimiento
53     WHERE ClienteID = clienteID;
54 END //
55 DELIMITER ;
56
```

Output

#	Time	Action
1	20:00:22	DROP PROCEDURE `tarea`.`ActualizarEdadCliente`
2	20:00:31	CREATE PROCEDURE actualizarEdadCliente(IN clienteID INT, IN nuevaFechaNacimiento

3. Procedimiento de Eliminación (DELETE)

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

```
57 -- CREAR EL PROCEDIMIENTO PARA ELIMINAR UN CLIENTE POR ID
58 DELIMITER //
59 CREATE PROCEDURE EliminarCliente(
60     IN clienteID INT
61 )
62 BEGIN
63     DELETE FROM cliente
64     WHERE ClienteID = clienteID;
65 END //
66 DELIMITER ;
67
```

Output

Action Output

#	Time	Action
✓ 1	20:00:22	DROP PROCEDURE `tarea`.`ActualizarEdadCliente`
✓ 2	20:00:31	CREATE PROCEDURE actualizarEdadCliente(IN clienteID INT, IN nuevaFechaNacimiento
✓ 3	20:01:20	CREATE PROCEDURE EliminarCliente(IN clienteID INT) BEGIN DELETE FROM cliente

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:

```
68 -- CONDICIONAL IF
69 DELIMITER //
70 CREATE PROCEDURE verificarEdadCliente(
71     IN clienteID INT,
72     OUT esMayor22 BOOLEAN
73 )
74 BEGIN
75     DECLARE edad INT;
76
77     -- Calcular la edad en años basada en la fecha de nacimiento
78     SELECT TIMESTAMPTDIFF(YEAR, FechaNacimiento, CURDATE()) INTO edad
79     FROM cliente
80     WHERE ClienteID = clienteID;
```

Output :

Action Output

#	Time	Action
✓ 1	20:00:22	DROP PROCEDURE `tarea`.`ActualizarEdadCliente`
✓ 2	20:00:31	CREATE PROCEDURE actualizarEdadCliente(IN clienteID INT, IN nuevaFechaNacimiento D
✓ 3	20:01:20	CREATE PROCEDURE EliminarCliente(IN clienteID INT) BEGIN DELETE FROM cliente ..
✗ 4	20:09:30	CREATE PROCEDURE EliminarCliente(IN clienteID INT) BEGIN DELETE FROM cliente ..
✓ 5	20:09:32	CREATE PROCEDURE verificarEdadCliente(IN clienteID INT, OUT esMayor22 BOOLEAN) .

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

```
91 • -- CREAR LA TABLA ORDENES CON SUS RELACIONES
92 CREATE TABLE ordenes (
93     OrdenID INT AUTO_INCREMENT PRIMARY KEY,
94     ClienteID INT,
95     FechaOrden DATE,
96     Monto DECIMAL(10,2),
97     Descripcion VARCHAR(255),
98     FOREIGN KEY (ClienteID) REFERENCES cliente(ClienteID) ON DELETE CASCADE
99 );
100
```

Output

Action Output

#	Time	Action
✓ 1	20:17:19	CREATE TABLE ordenes (OrdenID INT AUTO_INCREMENT PRIMARY KEY, ClienteID INT, ...

```
100 -- PROCEDIMIENTO PARA INGRESAR UNA ORDEN
101 DELIMITER //
102 • CREATE PROCEDURE ingresartarOrden(
103     IN clienteID INT,
104     IN fechaOrden DATE,
105     IN monto DECIMAL(10,2),
106     IN descripcion VARCHAR(255)
107 )
108 BEGIN
109     INSERT INTO ordenes (ClienteID, FechaOrden, Monto, Descripcion)
110     VALUES (clienteID, fechaOrden, monto, descripcion);
111 END //
112 DELIMITER ;
113
```

Output

Action Output

#	Time	Action
✓ 1	20:17:19	CREATE TABLE ordenes (OrdenID INT AUTO_INCREMENT PRIMARY KEY, ClienteID INT, ...
✗ 2	20:19:11	CREATE TABLE ordenes (OrdenID INT AUTO_INCREMENT PRIMARY KEY, ClienteID INT, ...
✓ 3	20:19:23	CREATE PROCEDURE ingresartarOrden(IN clienteID INT, IN fechaOrden DATE, IN mont...

- Procedimientos Actualizar Orden


```
114 -- PROCEDIMIENTO PARA ACTUALIZAR UNA ORDEN
```

```
115 DELIMITER //
```

```
116 CREATE PROCEDURE ActualizarOrden(  
117     IN ordenID INT,  
118     IN nuevoMonto DECIMAL(10,2),  
119     IN nuevaDescripcion VARCHAR(255)  
120 )  
121 BEGIN  
122     UPDATE ordenes  
123     SET Monto = nuevoMonto, Descripcion = nuevaDescripcion  
124     WHERE OrdenID = ordenID;  
125 END //
```

```
126 DELIMITER ;
```

```
127
```

Output

Action Output

	#	Time	Action
✖	2	20:19:11	CREATE TABLE ordenes (OrdenID INT AUTO_INCREMENT PRIMARY KEY, ClienteID IN...
✔	3	20:19:23	CREATE PROCEDURE ingresarOrden(IN clienteID INT, IN fechaOrden DATE, IN mon...
✔	4	20:20:12	CREATE PROCEDURE ActualizarOrden(IN ordenID INT, IN nuevoMonto DECIMAL(10,2), ...

- Procedimientos Eliminar Orden

```
128 -- PROCEDIMIENTO PARA ELIMINAR UNA ORDEN
129 DELIMITER //
130 CREATE PROCEDURE eliminarOrden(
131     IN ordenID INT
132 )
133 BEGIN
134     DELETE FROM ordenes
135     WHERE OrdenID = ordenID;
136 END //
137 DELIMITER ;
138
```

Output

Action Output

#	Time	Action
3	20:19:23	CREATE PROCEDURE ingresarOrden(IN clienteID INT, IN fechaOrden DATE, IN mon.
4	20:20:12	CREATE PROCEDURE ActualizarOrden(IN ordenID INT, IN nuevoMonto DECIMAL(10,2), .
5	20:21:13	CREATE PROCEDURE eliminarOrden(IN ordenID INT) BEGIN DELETE FROM ordenes .

Entrega Final

Instrucciones de Entrega:

1. Objetivos:

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

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