

ACTIVIDAD 3
PROCEDIMIENTOS Y TRIGGERS

Realizado por:

Francy Julieth Ramírez Rodríguez

Presentado a:

Juan Esteban Pineda Ángel

SOFKA U

2023

1. Elabore 4 procedimientos almacenados que me permitan agregar, actualizar, consultar y borrar, en una de las tablas de la librería (primera actividad)

Para realizar el ejercicio se selecciona la tabla cliente

USE libreriaescalibre;

1.1. PROCEDIMIENTO PARA AGREGAR UN REGISTRO

```
DELIMITER //
CREATE PROCEDURE agregarCliente
(IN _cedula VARCHAR (10),
IN _nombre VARCHAR(45))
BEGIN
INSERT INTO cliente (cedula,nombre)
values (_cedula,_nombre);
END
//
```

Llamar el procedimiento agregar cliente

- CALL agregarCliente('100','Flor Jimenez');

1.2. PROCEDIMIENTO PARA CONSULTAR CLIENTE

```
DELIMITER //
CREATE PROCEDURE buscar_cliente(IN _cedula VARCHAR (10))
BEGIN
SELECT * FROM cliente WHERE cedula = _cedula;
END
//
```

Llamar procedimiento buscar cliente

- CALL buscar_cliente(100);

1.3. PROCEDIMIENTO ACTUALIZAR CLIENTE

```
DELIMITER //
CREATE PROCEDURE modificar_cliente
(IN _cedula VARCHAR (10),
IN _nombre VARCHAR(45))
BEGIN
UPDATE cliente
SET nombre = _nombre
WHERE cedula = _cedula;
END
//
```

Llamar procedimiento modificar cliente

- `CALL modificar_cliente ('100','Carlos Torres');`

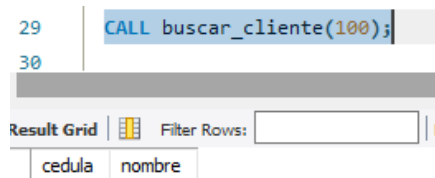
1.4. PROCEDIMIENTO BORRAR CLIENTE

```
DELIMITER //  
CREATE PROCEDURE borrar_cliente  
(IN _cedula VARCHAR (10))  
BEGIN  
DELETE FROM cliente WHERE cedula = _cedula;  
END  
//
```

Llamar procedimiento borrar cliente

- `CALL borrar_cliente(100);`

Se busca el cliente con id 100 en la tabla, la tabla no muestra ningún registro ya que no existen clientes con este id:



```
29 CALL buscar_cliente(100);  
30
```

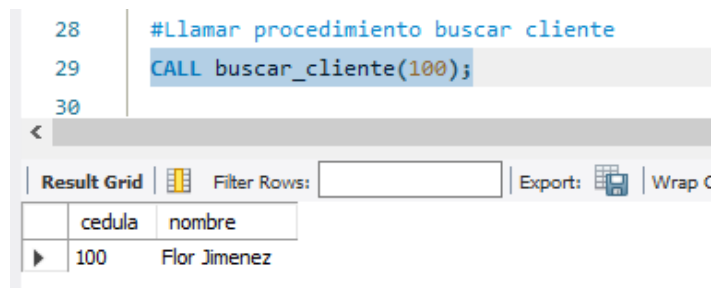
Result Grid | Filter Rows:

cedula	nombre
--------	--------

Se realiza registro de cliente:

```
#Llamar el procedimiento agregar cliente  
CALL agregarCliente('100','Flor Jimenez');
```

Se consulta registros agregados:



```
28 #Llamar procedimiento buscar cliente  
29 CALL buscar_cliente(100);  
30
```

Result Grid | Filter Rows: | Export: | Wrap C

cedula	nombre
100	Flor Jimenez

Se modifica cliente:

```
#Llamar procedimiento modificar cliente  
CALL modificar_cliente ('100','Carlos Torres');
```

Se consulta estado de actualización de cliente

```
28 #Llamar procedimiento buscar cliente  
29 CALL buscar_cliente(100);  
30
```

Result Grid | Filter Rows: | Export: | W

cedula	nombre
100	Carlos Torres

Se borra cliente:

```
57 #Llamar procedimiento borrar cliente  
58 CALL borrar_cliente(100);
```

Output

Action Output

#	Time	Action
133	18:49:30	CALL buscar_cliente(100)
134	18:50:39	CALL borrar_cliente(100)

Se consulta estado del cliente borrado:

```
28 #Llamar procedimiento buscar cliente  
29 CALL buscar_cliente(100);  
30
```

Result Grid | Filter Rows: | Export: | W

cedula	nombre
--------	--------

2. Elabore una nueva tabla llamada "control_de_cambios_librería" la cual debe contener 3 columnas (usuario, acción, fecha) y guarde utilizando 2 Triggers el nombre del usuario que agrego o elimino un registro en la tabla seleccionada en el punto anterior.

CREACION DE LA TABLA control_de_cambios_libreria

```
CREATE TABLE control_de_cambios_libreria (  
  usuario VARCHAR(45),  
  accion VARCHAR (45),
```

```
fecha DATETIME DEFAULT current_timestamp  
);
```

CREACION DE TRIGGER PARA REGISTRO DE ACCION AL AGREGAR CLIENTES

```
DELIMITER //  
CREATE TRIGGER registro_accion_agregar  
AFTER INSERT  
ON cliente FOR EACH ROW  
BEGIN  
INSERT INTO control_de_cambios_libreria (usuario,accion,fecha)  
VALUES ('Sergio Paez','Agregar cliente',now());  
END  
//
```

Se usa el procedimiento agregar cliente para cargar el registro

- CALL agregarCliente('40','Andres Ramirez');

CREACION DE TRIGGER PARA REGISTRO DE ACCION AL ELIMINAR CLIENTES

```
DELIMITER //  
CREATE TRIGGER registro_accion_borrar  
AFTER DELETE  
ON cliente FOR EACH ROW  
BEGIN  
INSERT INTO control_de_cambios_libreria (usuario,accion,fecha)  
VALUES ('Antonio Pardo','Borrar paciente',now());  
END  
//
```

Se llama el procedimiento creado para borrar cliente

- CALL borrar_cliente(40);

Se lista la tabla control_de_cambios_libreria

- SELECT * FROM control_de_cambios_libreria;

Se ejecuta el procedimiento para agregar clientes y automáticamente se activa el trigger que registra el usuario que realiza la acción y carga esta información en la tabla control_de_cambios_libreria, al realizar la misma acción con el procedimiento para borrar clientes se activa automáticamente el trigger que registra esta ocurrencia en la tabla control_de_cambios_libreria.

Lo anteriormente mencionado se puede ver en la siguiente tabla:

usuario	accion	fecha
Sergio Paez	Agregar cliente	2023-02-14 19:16:58
Antonio Pardo	Borrar paciente	2023-02-14 19:17:19

3. Elabore 4 procedimientos almacenados que me permitan agregar, actualizar, consultar y borrar, en una de las tablas del Hospital (segunda actividad).

Para realizar el ejercicio se selecciona la tabla paciente

```
USE hospitalqa;
```

3.1. PROCEDIMIENTO PARA AGREGAR UN REGISTRO

```
DELIMITER //
CREATE PROCEDURE agregarPaciente
(IN _id_paciente VARCHAR (10),
IN _nombre VARCHAR(45),
IN _direccion VARCHAR(45),
IN _id_procedimient VARCHAR(45))
BEGIN
INSERT INTO paciente (id_paciente,nombre,direccion,id_procedimient)
values (_id_paciente,_nombre,_direccion,_id_procedimient);
END
//
Llamar el procedimiento agregar paciente
```

- CALL agregarPaciente('20','Flor Ramirez','Fusagasuga','3');

3.2. PROCEDIMIENTO PARA CONSULTAR PACIENTE

```
DELIMITER //
CREATE PROCEDURE buscar_paciente(IN _id_paciente VARCHAR (10))
BEGIN
SELECT * FROM paciente WHERE id_paciente = _id_paciente;
END
//
Llamar procedimiento buscar paciente
```

- CALL buscar_paciente(20);

3.3. PROCEDIMIENTO ACTUALIZAR PACIENTE

```

DELIMITER //
CREATE PROCEDURE modificar_paciente
(IN _id_paciente VARCHAR(10),
IN _nombre VARCHAR(45),
IN _direccion VARCHAR(45),
IN _id_procedimient VARCHAR(45))
BEGIN
UPDATE paciente
SET nombre = _nombre,
direccion = _direccion,
id_procedimient = _id_procedimient
WHERE id_paciente = _id_paciente;
END
//
Llamar procedimiento modificar paciente

```

- CALL modificar_paciente ('20','Carla Mendez','Fusagasuga','2');

3.4. PROCEDIMIENTO BORRAR PACIENTE

```

DELIMITER //
CREATE PROCEDURE borrar_paciente
(IN _id_paciente VARCHAR(10))
BEGIN
DELETE FROM paciente WHERE id_paciente = _id_paciente;
END
//
Llamar procedimiento borrar paciente

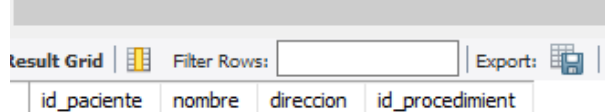
```

CALL borrar_paciente(20);

```

30  #Llamar procedimiento buscar paciente
31  CALL buscar_paciente(20);
32

```



- Se busca paciente con id 20: La tabla no muestra ningún registro porque no existe paciente con ese id.

- Se crea el paciente con id = 20 y se registran todos los demás datos del paciente:

```

#Llamar el procedimiento agregar paciente
CALL agregarPaciente('20','Flor Ramirez','Fusagasuga','3');

```

- Se confirma que los datos se hayan registrado e la tabla correctamente:

```

30 #Llamar procedimiento buscar paciente
31 CALL buscar_paciente(20);
32

```

Result Grid	Filter Rows:	Export:	
id_paciente	nombre	direccion	id_procedimient
20	Flor Ramirez	Fusagasuga	3

- Se modifican los datos del paciente con id 20.

```

#Llamar procedimiento modificar paciente
CALL modificar_paciente ('20','Carla Mendez','Fusagasuga','2');

```

- Se comprueba estado de la modificación:

```

30 #Llamar procedimiento buscar paciente
31 CALL buscar_paciente(20);
32

```

Result Grid	Filter Rows:	Export:	
id_paciente	nombre	direccion	id_procedimient
20	Carla Mendez	Fusagasuga	2

- Se elimina paciente con id 20

```

#Llamar procedimiento borrar paciente
CALL borrar_paciente(20);

```

- Se confirma estado de la eliminación del paciente con id 20

```

30 #Llamar procedimiento buscar paciente
31 CALL buscar_paciente(20);
32

```

Result Grid	Filter Rows:	Export:	
id_paciente	nombre	direccion	id_procedimient

Observamos que la tabla se muestra vacía por lo cual podemos deducir que el paciente fue borrado de la tabla.

4. Elabore una nueva tabla llamada "control_de_cambios_hospital" la cual debe contener 3 columnas (usuario, accion, fecha) y guarde usando 2 Triggers el nombre del usuario que agrego o elimino un registro en la tabla seleccionada en el punto anterior.

- **CREACION DE LA TABLA control_de_cambios_hospital**


```
CREATE TABLE control_de_cambios_hospital (  
usuario VARCHAR(45),  
accion VARCHAR (45),  
fecha DATETIME DEFAULT current_timestamp  
);
```

- **CREACION DE TRIGGER PARA REGISTRO DE ACCION AL AGREGAR PACIENTES**

```
DELIMITER //  
CREATE TRIGGER registro_accion_agregar  
AFTER INSERT  
ON paciente FOR EACH ROW  
BEGIN  
INSERT INTO control_de_cambios_hospital (usuario,accion,fecha)  
VALUES ('Santiago Peñalosa','Agregar paciente',now());  
END  
//
```

- **Se usa el procedimiento agregar paciente para cargar el registro**

```
CALL agregarPaciente('40','Andrea Ramirez','Fusagasuga','4');
```

- **CREACION DE TRIGGER PARA REGISTRO DE ACCION AL ELIMINAR PACIENTES**

```
DELIMITER //  
CREATE TRIGGER registro_accion_borrar  
AFTER DELETE  
ON paciente FOR EACH ROW  
BEGIN  
INSERT INTO control_de_cambios_hospital (usuario,accion,fecha)  
VALUES ('Santiago Peñalosa','Borrar paciente',now());  
END  
//
```

- **Se llama el procedimiento creado para borrar pacientes**

```
CALL borrar_paciente(40);
```

Se ejecuta el procedimiento para agregar pacientes y automáticamente se activa el trigger que registra el usuario que realiza la acción y carga esta información en la tabla control_de_cambios_hospital, al realizar la misma acción con el procedimiento para borrar pacientes se activa automáticamente el trigger que registra esta ocurrencia en la tabla control_de_cambios_hospital.

Lo anteriormente mencionado se puede ver en la siguiente tabla:

```
27 SELECT * FROM control_de_cambios_hospital;
28
```

Result Grid				Filter Rows:	Export:	Wrap C
	usuario	accion	fecha			
▶	Santiago Peñalosa	Agregar paciente	2023-02-14 17:57:48			
	Santiago Peñalosa	Borrar paciente	2023-02-14 17:58:47			