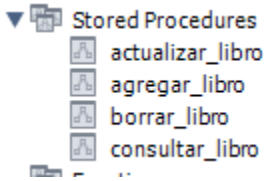
Triggers y procedimientos

1. **Procedimientos almacenados para la librería**



- Agregar registro: se explica el código del procedimiento almacenado que permite agregar un registro a una tabla específica de la librería.

- Consulta

```
DELIMITER //

CREATE PROCEDURE agregar_libro(
    IN isbn VARCHAR(10),
    IN titulo VARCHAR(45),
    IN num_paginas VARCHAR(45),
    IN nombre_editorial VARCHAR(50)
)
BEGIN
    INSERT INTO libro (ISBN, titulo, numero_paginas, nombre_editorial)
    VALUES (isbn, titulo, num_paginas, nombre_editorial);
END //

DELIMITER ;
```

```
CALL agregar_libro('978-8498', 'Cien años de soledad', '400', 'Editorial A');
```

- Resultado

ISBN	titulo	numero_paginas	nombre_editorial
978-8498	Cien años de soledad	400	Editorial A
ISBN1	Título 1	100	Editorial A
ISBN10	Título 10	1000	Editorial B
ISBN11	Título 11	1100	Editorial C

- Actualizar registro: se explica el código del procedimiento almacenado que permite actualizar un registro existente en una tabla específica de la librería.

- Consulta

```
CREATE PROCEDURE actualizar_libro(
    IN isbn VARCHAR(10),
    IN titulo VARCHAR(45),
    IN num_paginas VARCHAR(45),
    IN nombre_editorial VARCHAR(50)
)
BEGIN
    UPDATE libro
    SET titulo = titulo,
        numero_paginas = num_paginas,
        nombre_editorial = nombre_editorial
    WHERE ISBN = isbn;
END //
```

```
CALL actualizar_libro('978-8498', 'Cien años de felicidad', '200', 'Editorial A');
```

- Resultado

ISBN	titulo	numero_paginas	nombre_editorial
978-8498	Cien años de felicidad	200	Editorial A

- Consultar registro: se explica el código del procedimiento almacenado que permite consultar un registro en una tabla específica de la librería.

- Consulta

```
DELIMITER //

CREATE PROCEDURE consultar_libro(IN isbn VARCHAR(10))
BEGIN
    SELECT * FROM libro WHERE ISBN = isbn;
END //

DELIMITER ;
```

```
CALL consultar_libro('978-8498');
```

- Resultado

ISBN	titulo	numero_paginas	nombre_editorial
978-8498	Cien años de felicidad	200	Editorial A

- Borrar registro: se explica el código del procedimiento almacenado que permite borrar un registro existente en una tabla específica de la librería.

- Consulta

```
DELIMITER //

CREATE PROCEDURE borrar_libro(IN isbn VARCHAR(10))
BEGIN
    DELETE FROM libro WHERE ISBN = isbn;
END //

DELIMITER ;
```

```
CALL borrar_libro('978-8498');
```

- Resultado

ISBN	titulo	numero_paginas	nombre_editorial
ISBN1	Título 1	100	Editorial A
ISBN2	Título 2	200	Editorial B
ISBN3	Título 3	300	Editorial C
ISBN12	Título 4	400	Editorial D

2. Tabla de control de cambios para la librería

- Se crea la nueva tabla "control_de_cambios_librería" con las columnas "usuario", "acción" y "fecha".

- Consulta

```
CREATE TABLE control_de_cambios_librería (
  usuario VARCHAR(50),
  accion VARCHAR(10),
  fecha TIMESTAMP
);
```

- Se crea el primer Trigger que guarda el nombre del usuario que agrega un registro en la tabla seleccionada en el punto anterior.

- Consulta

```
CREATE TRIGGER insertar_libro_trigger
  BEFORE INSERT ON libro
  FOR EACH ROW
  INSERT INTO control_de_cambios_librería (usuario, accion, fecha)
  VALUES (USER(), 'agregar', NOW());
```

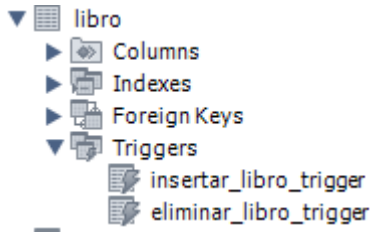
- Se explica cómo crear el segundo Trigger que guarda el nombre del usuario que elimina un registro en la tabla seleccionada en el punto anterior.

- Consulta

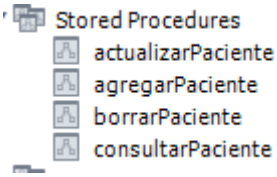
```
CREATE TRIGGER eliminar_libro_trigger
BEFORE DELETE ON libro
FOR EACH ROW
INSERT INTO control_de_cambios_librería (usuario, accion, fecha)
VALUES (USER(), 'eliminar', NOW());
```

- Resultados

usuario	accion	fecha
root@localhost	agregar	2023-02-14 15:44:02
root@localhost	eliminar	2023-02-14 16:35:54



Procedimientos almacenados para el hospital



Agregar registro: se explica el código del procedimiento almacenado que permite agregar un registro a una tabla específica del hospital.

- Consulta

```
DELIMITER $$
CREATE PROCEDURE agregarPaciente (IN nombre_paciente VARCHAR(45), IN direccion_paciente VARCHAR(45),
IN telefono_paciente VARCHAR(45), IN id_procedimiento INT)
BEGIN
    INSERT INTO Paciente (nombre, direccion, telefono, id_procedimiento)
    VALUES (nombre_paciente, direccion_paciente, telefono_paciente, id_procedimiento);
END $$
DELIMITER ;
```

```
CALL agregarPaciente('Juan Pérez', 'Calle 123', '555-1234', 1);
```

- Resultado

id_paciente	nombre	direccion	teléfono	id_procedimiento
5	Juan Pérez	Calle 123	555-1234	1
4	Ana	Calle 4	555-555-5558	2
3	Pedro	Calle 3	555-555-5557	2

Actualizar registro: se explica el código del procedimiento almacenado que permite actualizar un registro existente en una tabla específica del hospital.

- Consulta

```
DELIMITER //
CREATE PROCEDURE actualizarPaciente (IN id_paciente INT, IN nombre_paciente VARCHAR(45),
IN direccion_paciente VARCHAR(45), IN telefono_paciente VARCHAR(45), IN id_procedimiento INT)
BEGIN
    UPDATE Paciente
    SET nombre = nombre_paciente,
        direccion = direccion_paciente,
        telefono = telefono_paciente,
        id_procedimiento = id_procedimiento
    WHERE id_paciente = id_paciente;
END //
DELIMITER ;
```

```
CALL actualizarPaciente(5, 'eros jose', 'Calle 124', 'Nuevo teléfono', 2);
```

- Resultado

id_paciente	nombre	direccion	teléfono	id_procedimiento
5	eros jose	Calle 124	Nuevo teléfono	2
4	Ana	Calle 4	555-555-5558	2
3	Pedro	Calle 3	555-555-5557	2

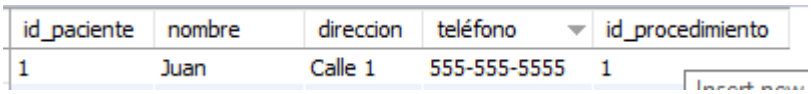
Consultar registro: se explica el código del procedimiento almacenado que permite consultar un registro en una tabla específica del hospital.

- Consultar

```
DELIMITER //
CREATE PROCEDURE consultarPaciente (IN id_paciente INT)
BEGIN
    SELECT *
    FROM Paciente
    WHERE id_paciente = id_paciente;
END //
DELIMITER ;
```

```
CALL consultarPaciente(1);
```

- Resultado



id_paciente	nombre	direccion	teléfono	id_procedimiento
1	Juan	Calle 1	555-555-5555	1

Borrar registro: se explica el código del procedimiento almacenado que permite borrar un registro existente en una tabla específica del hospital.

- Consulta

```
DELIMITER //
CREATE PROCEDURE borrarPaciente (IN id_paciente INT)
BEGIN
    DELETE FROM Paciente
    WHERE id_paciente = id_paciente;
END //
DELIMITER ;

CALL borrarPaciente(1);
```

Se explica cómo crear la nueva tabla "control_de_cambios_hospital" con las columnas "usuario", "acción" y "fecha".

- Consulta

```
CREATE TABLE control_de_cambios_hospital (
  usuario VARCHAR(45),
  accion VARCHAR(45),
  fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Se explica cómo crear el primer Trigger que guarda el nombre del usuario que agrega un registro en la tabla seleccionada en el punto anterior.

- Consulta

```
CREATE TRIGGER tr_control_de_cambios_insert
AFTER INSERT ON `consultorio_doctor`.`Paciente`
FOR EACH ROW
BEGIN
    INSERT INTO `consultorio_doctor`.`control_de_cambios_hospital` (`usuario`, `accion`)
    VALUES (USER(), 'Insertar en tabla Paciente');
END
```

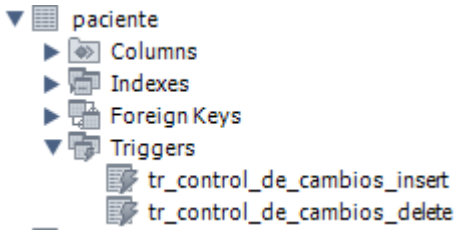
Se explica cómo crear el segundo Trigger que guarda el nombre del usuario que elimina un registro en la tabla seleccionada en el punto anterior.

- Consulta

```
CREATE TRIGGER tr_control_de_cambios_delete
AFTER DELETE ON `consultorio_doctor`.`Paciente`
FOR EACH ROW
BEGIN
    INSERT INTO `consultorio_doctor`.`control_de_cambios_hospital` (`usuario`, `accion`)
    VALUES (USER(), 'Eliminar de tabla Paciente');
END
```

- Resultados

Tabla de control de cambios para el hospital



id	usuario	accion
1	root@localhost	Eliminar de tabla Paciente
2	root@localhost	Insertar en tabla Paciente

Se realiza una breve reflexión sobre lo aprendido y la utilidad de las herramientas utilizadas en la tarea.

Temas tratados:

La creación de bases de datos y tablas

El uso de comandos básicos como SELECT, INSERT, UPDATE y DELETE para manipular datos dentro de las tablas

La utilización de cláusulas WHERE para filtrar datos específicos dentro de una tabla

La creación y utilización de procedimientos almacenados para realizar operaciones complejas y repetitivas en la base de datos

El uso de restricciones de integridad referencial y llaves foráneas para asegurar la integridad de los datos

La implementación de disparadores (triggers) para realizar operaciones adicionales como el registro de cambios y eventos en la base de datos.

Anexo:

procedimientos_y_trigger_consultorio.sql

procedimientos_y_trigger_Libreria.sql