

## TALLER 6

### Tercera actividad (TALLER 6):

- Elabore 4 procedimientos almacenados que me permitan agregar, actualizar, consultar y borrar, en una de las tablas de la librería (primera actividad).
- 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.
- Elabore 4 procedimientos almacenados que me permitan agregar, actualizar, consultar y borrar, en una de las tablas del Hospital (segunda actividad).
- Elabore una nueva tabla llamada "control\_de\_cambios\_hospital" 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.

### Solución:

En esta actividad trabajaremos con la base de datos Librería Busca Libre, la cual fue utilizada en la primera actividad.



La actividad nos pide que escojamos una tabla para trabajar, en esta ocasión utilizaremos la tabla libro y crearemos 4 procedimientos.

### Tabla Libros

	ISBN	titulo	numero_paginas	nombre_editorial
▶	000-000	La isla del tesoro	150	Editorial A
	001-001	La montaña mágica	320	Viva libre
	101-101	El Gran Inquisidor	200	Libro al viento

### Procedimiento Agregar:

Comenzamos creando el procedimiento insertar\_libro en el cual almacenamos una sentencia que permite insertar distintos libros a la tabla.

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

### Llamada del procedimiento agregar:

```
CALL insertar_libro("011-011", "Harry Potter I", "450", "Libro al viento");
```

### Resultado:

	ISBN	titulo	numero_paginas	nombre_editorial
▶	000-000	La isla del tesoro	150	Editorial A
	001-001	La montaña mágica	320	Viva libre
	011-011	Harry Potter I	450	Libro al viento
	101-101	El Gran Inquisidor	200	Libro al viento

### Procedimiento actualizar:

Seguidamente creamos un procedimiento llamado Actualizar\_libro en el cual buscaremos un libro por ID y luego cambiaremos sus valores.

```
DELIMITER //
```

- ```
CREATE PROCEDURE Actualizar_Libro(IN p_ISBN VARCHAR(10), IN p_titulo VARCHAR(45),  
  IN p_numero_paginas VARCHAR(45), IN p_nombre_editorial VARCHAR(50))  
BEGIN  
  UPDATE libro  
  SET titulo = p_titulo, numero_paginas = p_numero_paginas, nombre_editorial = p_nombre_editorial  
  WHERE ISBN = p_ISBN;  
END //
```

```
DELIMITER ;
```

### Llamada del método actualizar:

```
CALL Actualizar_Libro("011-011", "Harry Potter II", "394", "Libro al viento");
```

### Resultados:

|   | ISBN    | titulo             | numero_paginas | nombre_editorial |
|---|---------|--------------------|----------------|------------------|
| ▶ | 000-000 | La isla del tesoro | 150            | Editorial A      |
|   | 001-001 | La montaña mágica  | 320            | Viva libre       |
|   | 011-011 | Harry Potter II    | 394            | Libro al viento  |
|   | 101-101 | El Gran Inquisidor | 200            | Libro al viento  |

### Procedimiento consultar:

Siguiendo con la actividad propuesta el siguiente es el método consultar, existen 2 formas de consulta bastante comunes por ID y trayendo todos los registros, en esta ocasión utilizaremos búsqueda por ID, debido a que no sería muy lógico crear un procedimiento para el método global que literalmente se puede hacer en una sola línea:

```
SELECT * FROM libreriaescalibre.libro;
```

### ConsultarLibroPorISBN:

```
1  DELIMITER //
```

```
2  CREATE PROCEDURE consultarLibroPorISBN (IN idLibro VARCHAR(10))
```

- 3 

```
BEGIN
```
- 4 

```
  SELECT * FROM libro WHERE ISBN = idLibro;
```
- 5 

```
END //
```
- 6 

```
DELIMITER ;
```

### Llamada procedimiento consultar:

```
8 • CALL consultarLibroPorISBN('000-000');
```

### Resultados:

| Result Grid | Filter Rows:       | Export:        | Wrap Cell Content: |
|-------------|--------------------|----------------|--------------------|
| ISBN        | titulo             | numero_paginas | nombre_editorial   |
| 000-000     | La isla del tesoro | 150            | Editorial A        |

### Procedimiento borrar:

Este procedimiento cuenta con una sentencia que permite borrar un registro, que fue dado como parámetro.

```
DELIMITER //
```

- ```
CREATE PROCEDURE borrarLibro (IN idLibro VARCHAR(10))
```

```
BEGIN
DELETE FROM libro WHERE ISBN = idLibro;
END //
```

```
DELIMITER ;
```

### Llamada del procedimiento:

```
10 • CALL borrarLibro ('011-011');
```

### Resultado:

De esta no pongo resultados, si salió bien el registro seleccionado debería haber sido eliminado de la tabla.

## Triggers

Continuando con los puntos propuestos en el taller el siguiente paso es crear una tabla llamada "control\_de\_cambios\_librería".

```
1 • ○ create table control_de_cambios_librería(  
2     nombre varchar(50),  
3     accion varchar(30),  
4     fecha datetime default current_timestamp  
5 );
```

Una vez creada esta table en la cual se almacenaran los resultados de los triggers que se realicen, procedemos a realizar el primer trigger, el cual nos avisara quien y cuando se realizó una inserción de datos en la tabla libro.

### Trigger insertar:

```
-- Creacion trigger insercion de datos a libro  
DELIMITER //  
• create trigger trigger_ins_libro after insert on libro  
  for each row  
  begin  
    insert into control_de_cambios_librería values (  
      user(), 'Insertar', now()  
    );  
  end;  
//  
DELIMITER ;
```

### Resultados:

Result Grid			
Filter Rows:			
	nombre	accion	fecha
▶	root@localhost	Insertar	2023-02-14 14:32:56

### Trigger eliminar:

Ahora crearemos un trigger que permita guardar un registro cuando se eliminen datos de la tabla libro:

```
-- Creacion trigger eliminacion de datos a libro
DELIMITER //
create trigger trigger_Del_libro after delete on libro
for each row
begin
    insert into control_de_cambios_librería values (
        user(), 'Eliminar', now()
    );
end;
//
DELIMITER ;
```

### Resultados:

Result Grid			
Filter Rows:			
	nombre	accion	fecha
▶	root@localhost	Insertar	2023-02-14 14:32:56
	root@localhost	Eliminar	2023-02-14 14:40:10

Ahora ya podemos pasar al siguiente punto, este punto es bastante similar al que ya completamos, la única diferencia es que este proceso será realizado en la base de datos Hospital.

Para ello en esta oportunidad usaremos la tabla Medico:

#### Procedimiento agregar:

```
-- Procedimiento insertar medico
DELIMITER //
CREATE PROCEDURE insertarMedico (IN idMedico VARCHAR(10), IN nombreMedico VARCHAR(45),
IN apellidoMedico VARCHAR(45), IN especialidad VARCHAR(45))
BEGIN
    INSERT INTO Medico (idMedico, Nombre_Medico, apellido_medico, especialidad)
    VALUES (idMedico, nombreMedico, apellidoMedico, especialidad);
END //
DELIMITER ;
```

#### Llamada del procedimiento:

```
12 • CALL insertarMedico("001", "Juan", "Pérez", "Pediatra");
```

#### Resultado:

	idMedico	Nombre_Medico	apellido_medico	especialidad
▶	001	Juan	Pérez	Pediatra
	11-00	David	Pineda	Pediatricia
	11-01	Raul	Gomez	Neurocirugia



#### Procedimiento actualizar:

```
-- Procedimiento actualizar medico
DELIMITER //
CREATE PROCEDURE actualizarMedico (IN p_idMedico VARCHAR(10), IN p_Nombre_Medico VARCHAR(45),
IN p_apellido_medico VARCHAR(45), IN p_especialidad VARCHAR(45))
BEGIN
    UPDATE Medico
    SET Nombre_Medico = p_Nombre_Medico, apellido_medico = p_apellido_medico, especialidad = p_especialidad
    WHERE idMedico = p_idMedico;
END //
DELIMITER ;
```

#### Llamada procedimiento:

```
• CALL actualizarMedico("001", "Juan", "Chavez", "Bacteriologo");
```

### Resultados:

Result Grid			Filter Rows:	<input type="text"/>	Edit: 
	idMedico	Nombre_Medico	apellido_medico	especialidad	
▶	001	Juan	Chavez	Bacteriologo	
	11-00	David	Pineda	Pediatricia	
	11-01	Raul	Gomez	Neurocirugia	

### Procedimiento consultar:

```
DELIMITER //
```

- **CREATE PROCEDURE** consultarMedicoPorID (IN buscar\_idMedico VARCHAR(10))  
BEGIN  
SELECT \* FROM Medico WHERE idMedico = buscar\_idMedico;  
END //

```
DELIMITER ;
```

### Llamada procedimiento consultar:

- **CALL** consultarMedicoPorID("001");

### Resultado:

Result Grid		Filter Rows:	Export:	
	idMedico	Nombre_Medico	apellido_medico	especialidad
▶	001	Juan	Chavez	Bacteriologo

### Procedimiento eliminar medico:

```
-- Procedimiento eliminar medico
```

```
DELIMITER //
```

- **CREATE PROCEDURE** eliminarMedico (IN del\_idMedico VARCHAR(10))  
BEGIN  
DELETE FROM Medico WHERE idMedico = del\_idMedico;  
END //

```
DELIMITER ;
```

### Llamada del procedimiento eliminar:

- **CALL** eliminarMedico("001");

### Resultado:

De esta no pongo resultados, si salió bien el registro seleccionado debería haber sido eliminado de la tabla.



## Triggers

Continuando con los puntos propuestos en el taller el siguiente paso es crear una tabla llamada "control\_de\_cambios\_hospital".

```
1  -- Creacion tabla cambios
2  • create table control_de_cambios_hospital(
3      nombre varchar(50),
4      accion varchar(30),
5      fecha datetime default current_timestamp
6  );
```

Ahora procedemos creando los triggers para la inserción y eliminación de datos en la tabla médico, guardando los registros en la tabla "control\_de\_cambios\_hospital".

### Trigger insertar medico:

```
    -- Creacion trigger insercion de datos a medico
    DELIMITER //
    • create trigger trigger_ins_medico after insert on medico
      for each row
      begin
          insert into control_de_cambios_hospital values (
              user(), 'Insertar', now()
          );
      end;
      //
      DELIMITER ;
```

### Resultado:

Result Grid			
Filter Rows:			
	nombre	accion	fecha
▶	root@localhost	Insertar	2023-02-14 15:22:05

### Trigger eliminar medico:

```
-- Creacion trigger eliminacion de datos a medico
DELIMITER //
• create trigger trigger_Del_medico after delete on medico
  for each row
  begin
    insert into control_de_cambios_hospital values (
      user(), 'Eliminar', now()
    );
  end;
//
DELIMITER ;
```

### Resultado:

Result Grid			
Filter Rows:			
	nombre	accion	fecha
▶	root@localhost	Insertar	2023-02-14 15:22:05
	root@localhost	Eliminar	2023-02-14 15:22:31

¡Por recomendación en clase, despedida solemne!  
Gracias por la atención.