Tercera Actividad – Procedimientos y Trigger

Requerimientos de la actividad

- 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, accion, fecha) y guarde utilizando 2 Triggers el nombre del usuario que agrego o elimino un registro en la tabla seleccionada en el punto anterior.

PROCEDIMIENTOS DE LA TABLA EDITORIAL

AGREGAR

1. Se crea el procedimiento almacenado de la tabla editorial en este caso se llamara agregar editorial
2. Especificamos los parámetros que se van agregar (nombre editorial, ciudad editorial, complemento editorial, teléfono editorial) se tiene en cuenta que los parámetros que pasemos tengan el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL insert into dentro del cuerpo del procedimiento (begin, end)
4. Se ejecuta el procedimiento mediante la sentencia call y se especifican los datos que se van agregar (agregar_editorial)

```
delimiter //
create procedure agregar_editorial (in Nom_editorial varchar(50), Cdad_editorial varchar(30),
Compl_editorial varchar(100), tel_editorial varchar(20))
begin
    insert into editorial (nombre,ciudad,complemento,Telefono)
    values (Nom_editorial,Cdad_editorial,Compl_editorial,tel_editorial);
end;    //

call agregar_editorial ('Norma','Bogota','cr 6 12-25','320675489');

select * from editorial;
```

5. Se consulta la tabla editorial para mirar si la información quedo agregada mediante la sentencia select * from.

	nombre	ciudad	complemento	Telefono
▶	Altea	Bogota	Cr 11 98-50	310675893
	Edicion Gamma	Bogota	calle 85 18-32	3185642311
	Luna libros	Bogota	calle 97 16-50	3155126886
	Norma	Bogota	cr 6 12-25	320675489
	Tragaluz Editores	Medellin	calle 9 43c-50	3214568950

ACTUALIZAR

1. Se crea el procedimiento almacenado de la tabla editorial en este caso se llamará actualizar editorial
2. Especificamos los parámetros que se van actualizar (nombre editorial, ciudad editorial, complemento editorial, teléfono editorial) se tiene en cuenta que los parámetros que pasemos tengan el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL update dentro del cuerpo del procedimiento (begin, end) seguido de la sentencia set para decir si el campo de la tabla es igual a la variable local que se definió anteriormente (ciudad, complemento, teléfono) donde la clave primaria va ser igual a la variable local (nombre)
4. Se ejecuta el procedimiento mediante la sentencia call y se especifican los datos que se van actualizar (actualizar_editorial)

```
delimiter //
create procedure actualizar_editorial(in Nom_editorial varchar(50), Cdad_editorial varchar(30),
Compl_editorial varchar(100), tel_editorial varchar(20))
begin
    update editorial set ciudad=Cdad_editorial, complemento=Compl_editorial, Telefono= tel_editorial
    where nombre=Nom_editorial;
end //
delimiter;

call actualizar_editorial ('Altea','Cundinamarca','calle 5 13-25','3182953467')
```

5. Se consulta la tabla editorial para mirar si la información quedo actualizada mediante la sentencia select * from.

	nombre	ciudad	complemento	Telefono
▶	Altea	Cundinamarca	calle 5 13-25	3182953467
	Edicion Gamma	Bogota	calle 85 18-32	3185642311
	Luna libros	Bogota	calle 97 16-50	3155126886
	Norma	Bogota	cr 6 12-25	320675489
	Tragaluz Editores	Medellin	calle 9 43c-50	3214568950

CONSULTAR

1. Se crea el procedimiento almacenado de la tabla editorial en este caso se llamará consultar por nombre
2. Como vamos a consultar mediante el nombre se crea la variable local de nombre editorial con el mismo tipo de dato que la tabla original

3. Se escribe la sentencia SQL select * from dentro del cuerpo del procedimiento (begin, end) de editorial donde se iguala el campo de la tabla con la variable local

```
delimiter //
create procedure consultar_por_nombre (in nombre_editorial varchar (50))
begin
select * from editorial where nombre=nombre_editorial;
end //
delimiter ;

call consultar_por_nombre('Altea')
```

4. Se ejecuta el procedimiento mediante la sentencia call y se especifican el nombre de la editorial que se va a consultar

	nombre	ciudad	complemento	Telefono
▶	Altea	Bogota	Cr 11 98-50	310675893

ELIMINAR

1. Se crea el procedimiento almacenado de la tabla editorial en este caso se llamará eliminar editorial
2. Se va eliminar el registro mediante el nombre por lo tanto se crea la variable local de nombre editorial con el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL delete dentro del cuerpo del procedimiento (begin, end) donde se iguala el campo de la tabla con la variable local
4. Antes de ejecutar el procedimiento vamos a consultar la tabla editorial con la sentencia select * from

	nombre	ciudad	complemento	Telefono
▶	Altea	Cundinamarca	calle 5 13-25	3182953467
	Edicion Gamma	Bogota	calle 85 18-32	3185642311
	Luna libros	Bogota	calle 97 16-50	3155126886
	Norma	Bogota	cr 6 12-25	320675489
	Tragaluz Editores	Medellin	calle 9 43c-50	3214568950

5. Ahora si vamos a ejecutar el procedimiento mediante la sentencia call y se especifican el nombre de la editorial que se va a eliminar

```
delimiter //
create procedure eliminar_editorial (in Nombre_editorial varchar(50))
begin
    delete from editorial where nombre=Nombre_editorial;
end; //

call eliminar_editorial('Norma')
```

6. Se consulta la tabla editorial para mirar si el registro quedo eliminado mediante la sentencia select * from.

	nombre	ciudad	complemento	Telefono
▶	Altea	Cundinamarca	calle 5 13-25	3182953467
	Edicion Gamma	Bogota	calle 85 18-32	3185642311
	Luna libros	Bogota	calle 97 16-50	3155126886
	Tragaluz Editores	Medellin	calle 9 43c-50	3214568950

TRIGGERS DE LA TABLA EDITORIAL

1. Se crea la tabla control de cambios librería para llevar registro de los usuarios que realizan diferentes acciones en la tabla editorial

```
create table control_de_cambios_libreria(
    usuario varchar(45),
    accion varchar(45),
    fecha datetime default current_timestamp
);
```

2. Se crea el trigger que va generar un registro en la tabla control de cambios librería cada vez que se agregue una editorial

```
delimiter //
create trigger nom_editorial after insert on editorial
    for each row
    begin
        insert into control_de_cambios_libreria (usuario,accion,fecha)
        values ('Laura alba','Agregar', now());
    end;
    //
```

3. Se agrega un numero registro en la tabla editorial

```
insert into editorial (nombre,ciudad,complemento,Telefono)
values ('Norma','Bogota','cr 7 12-25','3205642857');
```

4. Se consulta la tabla control de cambios librería para mirar si el registro quedo agregado mediante la sentencia select * from.

	usuario	accion	fecha
▶	Laura alba	Agregar	2023-02-14 21:02:34

5. Se crea el trigger que va generar un registro en la tabla control de cambios librería cada vez que se elimine una editorial

```
delimiter //
create trigger eliminar_editorial after delete on editorial
    for each row
    begin
        insert into control_de_cambios_libreria (usuario,accion,fecha)
        values ('Pedro perez','eliminar', now());
    end;
    //
```

6. Se elimina un registro de la tabla editorial

```
delete from editorial where nombre='Norma';
```

7. Nuevamente consultamos la tabla control de cambios librería con la sentencia select * from

	usuario	accion	fecha
▶	Laura alba	Agregar	2023-02-14 21:02:34
	Pedro perez	eliminar	2023-02-14 21:08:49

PROCEDIMIENTOS DE LA TABLA MEDICO

AGREGAR

1. Se crea el procedimiento almacenado de la tabla medico en este caso se llamará agregar medico
2. Especificamos los parámetros que se van agregar (cedula médico, nombres médico, apellidos médico, especialidad) se tiene en cuenta que los parámetros que pasemos tengan el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL insert into dentro del cuerpo del procedimiento (begin, end)
4. Se ejecuta el procedimiento mediante la sentencia call y se especifican los datos que se van agregar (agregar_medico)

```
delimiter //
create procedure agregar_medico (in Ced_medico varchar(15), Nom_medico varchar(60),
    Apel_medico varchar(60), Espe_medico varchar(55))
begin
    insert into medico(cedula_medico,nombres_medico,apellidos_medico,especialidad)
    values (Ced_medico,Nom_medico,Apel_medico,Espe_medico);
end;  //

call agregar_medico('51734675','Juan David','Mazo Rivera','Cardiologia')
```

5. Se consulta la tabla medico para mirar si la información quedo agregada mediante la sentencia select * from.

cedula_medico	nombres_medico	apellidos_medico	especialidad
10587675	Yeferson	Ortiz Bolivar	Cirujano
10832356	Sergio Esteben	Ramirez Contreras	Odontologo
10835676	Deiver Adrian	Gonzalez Hernandez	Medico General
10987864	Maria Fernanda	Zuñiga Zalazar	Pediatra
19078654	Karen	Rivera Lasso	Neurocirugia
51734675	Juan David	Mazo Rivera	Cardiologia

ACTUALIZAR

1. Se crea el procedimiento almacenado de la tabla editorial en este caso se llamará actualizar medico
2. Especificamos los parámetros que se van actualizar (cedula médico, nombres médicos, apellidos médico, especialidad) se tiene en cuenta que los parámetros que pasemos tengan el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL update dentro del cuerpo del procedimiento (begin, end) seguido de la sentencia set para decir si el campo de la tabla es igual a la variable local que se definió anteriormente (nombres médico, apellidos médico, especialidad) donde la clave primaria va ser igual a la variable local (cedula medico)
4. Se ejecuta el procedimiento mediante la sentencia call y se especifican los datos que se van actualizar (actualizar_medico), en este caso se va añadir una especialidad a un medico

```
delimiter //
create procedure actualizar_medico (in Ced_medico varchar(15), Nom_medico varchar(60),
    Apel_medico varchar(60), Espe_medico varchar(55))
begin
    update medico set nombres_medico=Nom_medico, apellidos_medico=Apel_medico,especialidad=Espe_medico
    where cedula_medico=Ced_medico;
end; //

call actualizar_medico ('10987864','Maria Fernanda','Zuñiga Zalazar','Pediatra-Cardiologia')
```

5. Se consulta la tabla medico para mirar si la información quedo actualizada mediante la sentencia select * from.

cedula_medico	nombres_medico	apellidos_medico	especialidad
10587675	Yeferson	Ortiz Bolivar	Cirujano
10832356	Sergio Esteben	Ramirez Contreras	Odontologo
10835676	Deiver Adrian	Gonzalez Hernandez	Medico General
10987864	Maria Fernanda	Zuñiga Zalazar	Pediatra-Cardiologia
19078654	Karen	Rivera Lasso	Neurocirugia
51734675	Juan David	Mazo Rivera	Cardiologia

CONSULTAR

1. Se crea el procedimiento almacenado de la tabla médico en este caso se llamará consultar por cedula
2. Como vamos a consultar mediante la cedula se crea la variable local de cedula medico con el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL select * from dentro del cuerpo del procedimiento (begin, end) de medico donde se iguala el campo de la tabla con la variable local

```
delimiter //
create procedure consultar_cedula (in Ced_medico varchar(15))
begin
    select * from medico where cedula_medico=Ced_medico;
end; //

call consultar_cedula ('10832356')
```

4. Se ejecuta el procedimiento mediante la sentencia call y se especifican la cedula del médico que se va a consultar

	cedula_medico	nombres_medico	apellidos_medico	especialidad
▶	10832356	Sergio Esteben	Ramirez Contreras	Odontologo

ELIMINAR

1. Se crea el procedimiento almacenado de la tabla medico en este caso se llamará eliminar medico
2. Se va eliminar el registro mediante la cedula por lo tanto se crea la variable local de cedula medico con el mismo tipo de dato que la tabla original
3. Se escribe la sentencia SQL delete dentro del cuerpo del procedimiento (begin, end) donde se iguala el campo de la tabla con la variable local
4. Antes de ejecutar el procedimiento vamos a consultar la tabla editorial con la sentencia select * from

cedula_medico	nombres_medico	apellidos_medico	especialidad
10587675	Yeferson	Ortiz Bolivar	Cirujano
10832356	Sergio Esteben	Ramirez Contreras	Odontologo
10835676	Deiver Adrian	Gonzalez Hernandez	Medico General
10987864	Maria Fernanda	Zuñiga Zalazar	Pediatra-Cardiologia
19078654	Karen	Rivera Lasso	Neurocirugia
51734675	Juan David	Mazo Rivera	Cardiologia

5. Ahora si vamos a ejecutar el procedimiento mediante la sentencia call y se especifican la cedula del medico que se va a eliminar

```
delimiter //
create procedure eliminar_medico(in Ced_medico varchar(15))
begin
    delete from medico where cedula_medico=Ced_medico;
end; //

call eliminar_medico('51734675')
```

6. Se consulta la tabla medico para mirar si el registro quedo eliminado mediante la sentencia select * from.

	cedula_medico	nombres_medico	apellidos_medico	especialidad
▶	10587675	Yeferson	Ortiz Bolivar	Cirujano
	10832356	Sergio Esteben	Ramirez Contreras	Odontologo
	10835676	Deiver Adrian	Gonzalez Hernandez	Medico General
	10987864	Maria Fernanda	Zuñiga Zalazar	Pediatra-Cardiologia
	19078654	Karen	Rivera Lasso	Neurocirugia

TRIGGERS DE LA TABLA MEDICO

1. Se crea la tabla control de cambios hospital para llevar registro de los usuarios que realizan diferentes acciones en la tabla editorial

```
create table control_de_cambios_hospital(
    usuario varchar(45),
    accion varchar(45),
    fecha datetime default current_timestamp
);
```

2. Se crea el trigger que va generar un registro en la tabla control de cambios hospital cada vez que se agregue un medico

```
delimiter //
create trigger agregar_medico after insert on medico
    for each row
    begin
        insert into control_de_cambios_hospital (usuario,accion,fecha)
        values ('Luisa mendez','Agregar', now());
    end;
    //
```

3. Se agrega un numero registro en la tabla medico

```
insert into medico(cedula_medico,nombres_medico,apellidos_medico,especialidad)
values ('516784523','Alexander','Gonzalez Ramirez','Nutricionista');
```

4. Se consulta la tabla control de cambios hospital para mirar si el registro quedo agregado mediante la sentencia select * from.

	usuario	accion	fecha
▶	Luisa mendez	Agregar	2023-02-14 21:29:57

5. Se crea el trigger que va generar un registro en la tabla control de cambios hospital cada vez que se elimine un medico

```
delimiter //
create trigger eliminar_medico after delete on medico
    for each row
    begin
        insert into control_de_cambios_hospital (usuario,accion,fecha)
        values ('Vivian Gonzalez','eliminar', now());
    end;
    //
```

6. Se elimina un registro de la tabla editorial

```
delete from medico where cedula_medico='516784523';
```

7. Nuevamente consultamos la tabla control de cambios librería con la sentencia select * from

	usuario	accion	fecha
▶	Luisa mendez	Agregar	2023-02-14 21:29:57
	Vivian Gonzalez	eliminar	2023-02-14 21:38:59