



Universidad Politécnica de Tlaxcala Región Poniente

Ingeniería en Sistemas Computacionales

Materia: Administración de Base de Datos

Docente: Ing. Vanesa Tenopala Zavala

Alumnos: 22SIC008

Isaac Brandon Martínez Ramírez

Tema: Reporte Ejercicios de Disparadores

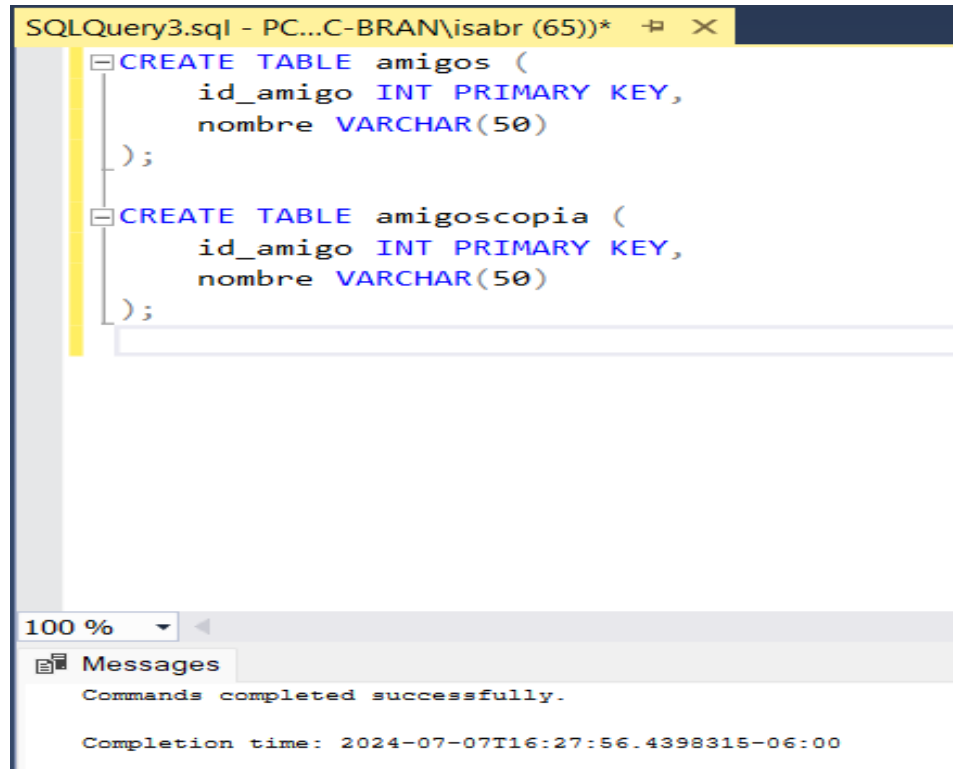
Ciclo escolar: mayo-agosto 2024

Fecha: 7 de julio de 2024

Reporte de Ejercicios y Ejemplos de Disparadores en SQL Server

Ejemplo 1: Estructura y Disparador de Inserción

1. Creación de tablas amigos y amigoscopia



```
SQLQuery3.sql - PC...C-BRAN\isabr (65))*  X
CREATE TABLE amigos (
    id_amigo INT PRIMARY KEY,
    nombre VARCHAR(50)
);
CREATE TABLE amigoscopia (
    id_amigo INT PRIMARY KEY,
    nombre VARCHAR(50)
);
```

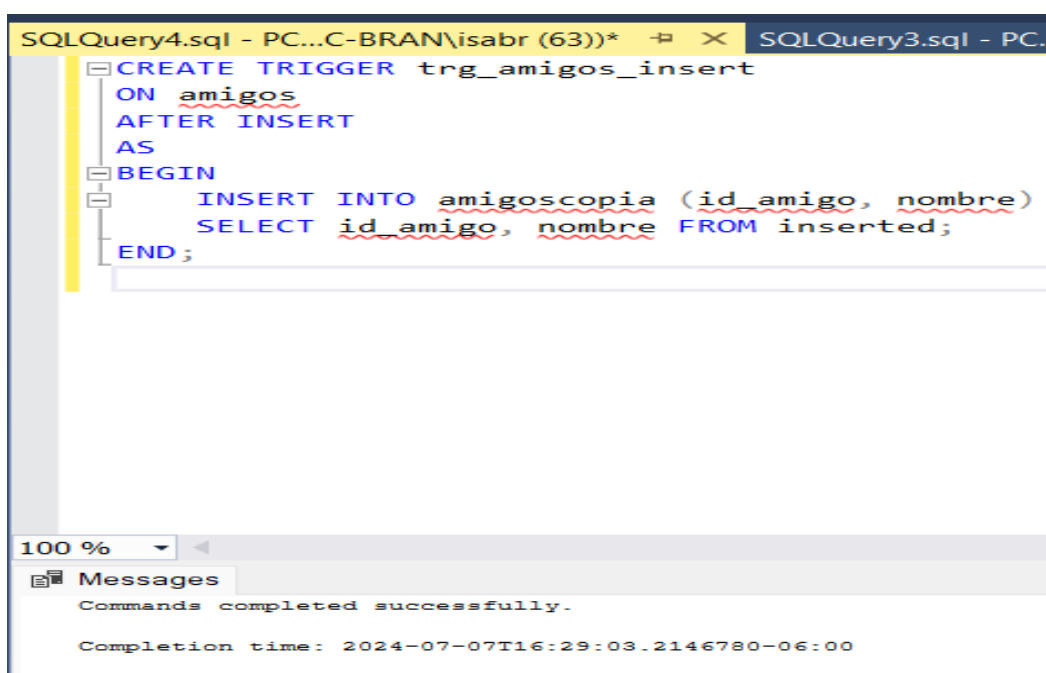
100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T16:27:56.4398315-06:00

2. Creación del disparador para insertar en amigoscopia al insertar en amigos



```
SQLQuery4.sql - PC...C-BRAN\isabr (63))*  X  SQLQuery3.sql - PC...
CREATE TRIGGER trg_amigos_insert
ON amigos
AFTER INSERT
AS
BEGIN
    INSERT INTO amigoscopia (id_amigo, nombre)
    SELECT id_amigo, nombre FROM inserted;
END;
```

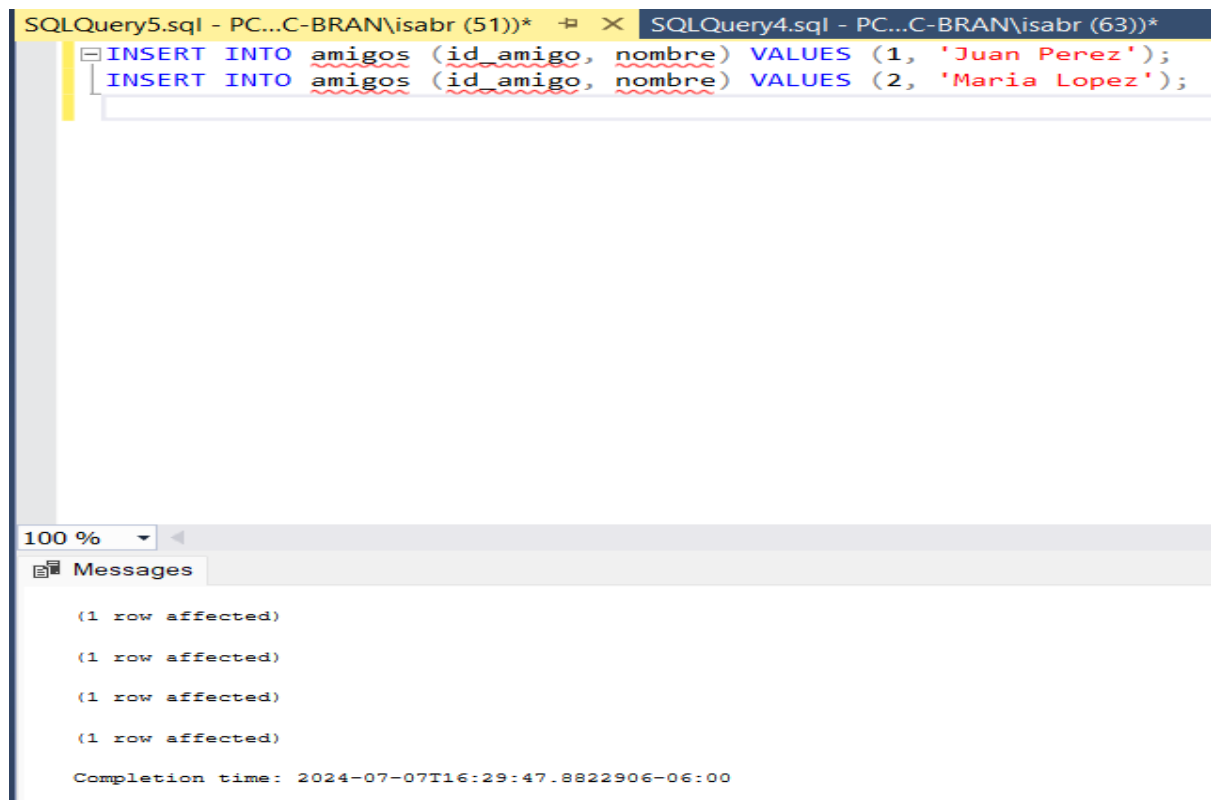
100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T16:29:03.2146780-06:00

3. Inserción de datos en la tabla amigos



```
SQLQuery5.sql - PC...C-BRAN\isabr (51))* X SQLQuery4.sql - PC...C-BRAN\isabr (63))*  
INSERT INTO amigos (id_amigo, nombre) VALUES (1, 'Juan Perez');  
INSERT INTO amigos (id_amigo, nombre) VALUES (2, 'Maria Lopez');
```

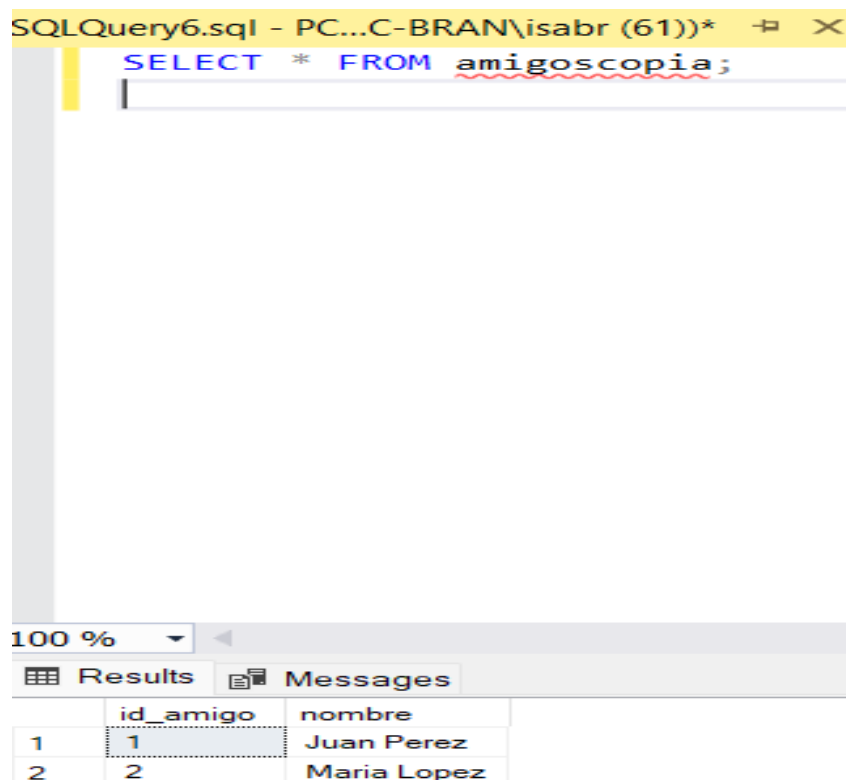
100 %

Messages

(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)

Completion time: 2024-07-07T16:29:47.8822906-06:00

4. Verificación del contenido de la tabla amigoscopia



```
SQLQuery6.sql - PC...C-BRAN\isabr (61))* X  
SELECT * FROM amigoscopia;
```

100 %

Results Messages

	id_amigo	nombre
1	1	Juan Perez
2	2	Maria Lopez

Ejemplo 2: Disparador de Actualización de Precios

1. Creación de las tablas libros y control

```
SQLQuery7.sql - PC...C-BRAN\isabr (68))* X
CREATE TABLE libros (
    codigo INT PRIMARY KEY,
    titulo VARCHAR(50),
    autor VARCHAR(50),
    editorial VARCHAR(50),
    precio DECIMAL(10, 2)
);

CREATE TABLE control (
    usuario VARCHAR(50),
    fecha DATETIME,
    codigo_libro INT,
    precio_anterior DECIMAL(10, 2),
    precio_nuevo DECIMAL(10, 2)
);
```

data type decimal(10, 2)

100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T16:47:11.6971803-06:00

2. Inserción de registros en la tabla libros

```
SQLQuery8.sql - PC...C-BRAN\isabr (52))* X SQLQuery7.sql - PC...C-BRAN\isabr (68))*
INSERT INTO libros VALUES (100, 'Uno', 'Richard Bach', 'Planeta', 25);
INSERT INTO libros VALUES (103, 'El aleph', 'Borges', 'Emece', 28);
INSERT INTO libros VALUES (105, 'Matematica estas ahi', 'Paenza', 'Nuevo siglo', 12);
INSERT INTO libros VALUES (120, 'Aprenda PHP', 'Molina Mario', 'Nuevo siglo', 55);
INSERT INTO libros VALUES (145, 'Alicia en el pais de las maravillas', 'Carroll', 'Planeta', 35);
```

100 %

Messages

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2024-07-07T16:51:03.0108815-06:00

3. Creación del disparador de actualización de precios

```
SQLQuery9.sql - PC...C-BRAN\isabr (61))* X SQLQuery8.sql - PC...C-BRAN\isabr (52))* SQLQuery7.sql - PC...C-BRA
CREATE TRIGGER trg_update_precio_libros
ON libros
FOR UPDATE
AS
BEGIN
    IF UPDATE(precio)
    BEGIN
        INSERT INTO control (usuario, fecha, codigo_libro, precio_anterior, precio_nuevo)
        SELECT SYSTEM_USER, GETDATE(), i.codigo, d.precio, i.precio
        FROM inserted i
        JOIN deleted d ON i.codigo = d.codigo;
    END
END;

100 %
Messages
Commands completed successfully.
Completion time: 2024-07-07T16:51:36.8151179-06:00
```

4. Actualización del precio de un libro

```
SQLQuery10.sql - P...C-BRAN\isabr (62))* X SQLQuery9.sql - PC...C-B
UPDATE libros SET precio = 30 WHERE codigo = 100;

100 %
Messages
(1 row affected)
(1 row affected)
Completion time: 2024-07-07T16:52:39.0250615-06:00
```

5. Verificación del contenido de la tabla control

SQLQuery11.sql - P...C-BRAN\isabr (64))* X SQLQuery10.sql - P...C-BRAN\isabr (6

```
SELECT * FROM control;
```

100 %

Results Messages

	usuario	fecha	codigo_libro	precio_anterior	precio_nuevo
1	PC-Bran\isabr	2024-07-07 16:52:39.000	100	25.00	30.00

Ejemplo 3: Control de Precio en Disparador

1. Eliminamos el disparador existente

SQLQuery13.sql - P...C-BRAN\isabr (62))* X

```
IF OBJECT_ID('tr_actualizarPrecioLibros', 'TR') IS NOT NULL
BEGIN
    DROP TRIGGER tr_actualizarPrecioLibros;
END;
GO
```

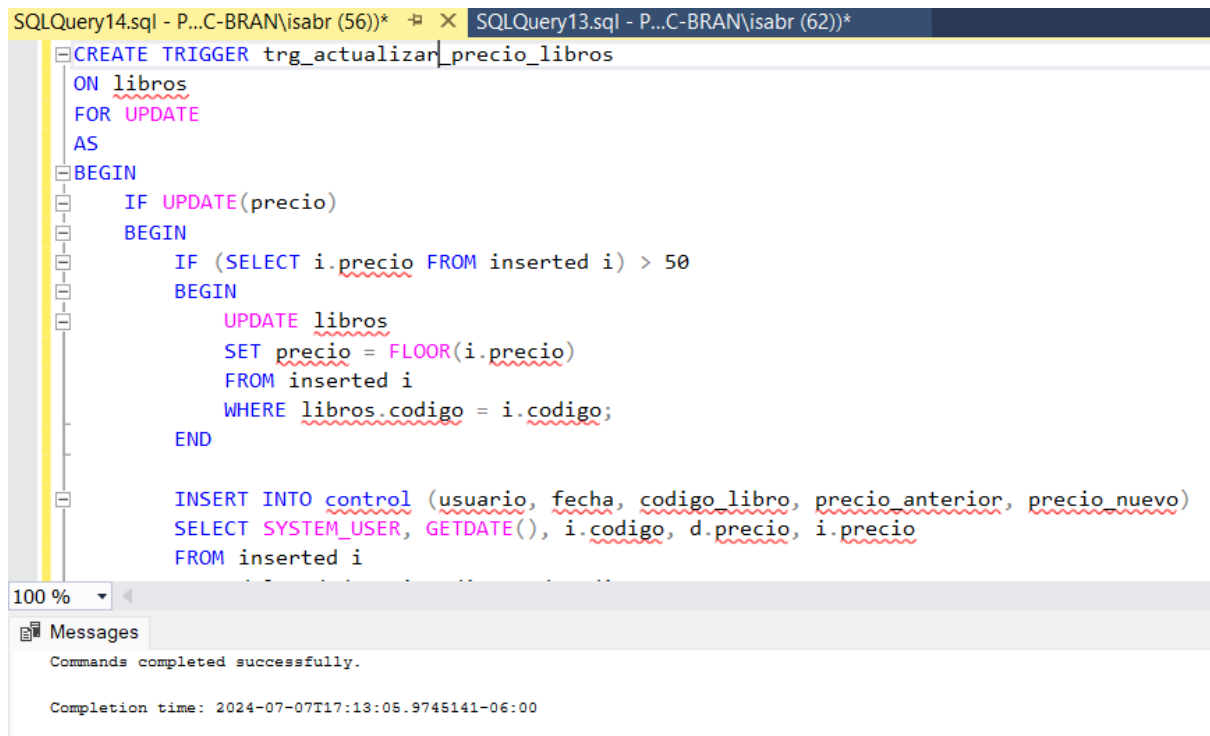
100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:08:52.0221133-06:00

2. Modificación del disparador para controlar el precio



```
CREATE TRIGGER trg_actualizar_precio_libros
ON libros
FOR UPDATE
AS
BEGIN
    IF UPDATE(precio)
    BEGIN
        IF (SELECT i.precio FROM inserted i) > 50
        BEGIN
            UPDATE libros
            SET precio = FLOOR(i.precio)
            FROM inserted i
            WHERE libros.codigo = i.codigo;
        END

        INSERT INTO control (usuario, fecha, codigo libro, precio anterior, precio nuevo)
        SELECT SYSTEM_USER, GETDATE(), i.codigo, d.precio, i.precio
        FROM inserted i
    END
END
```

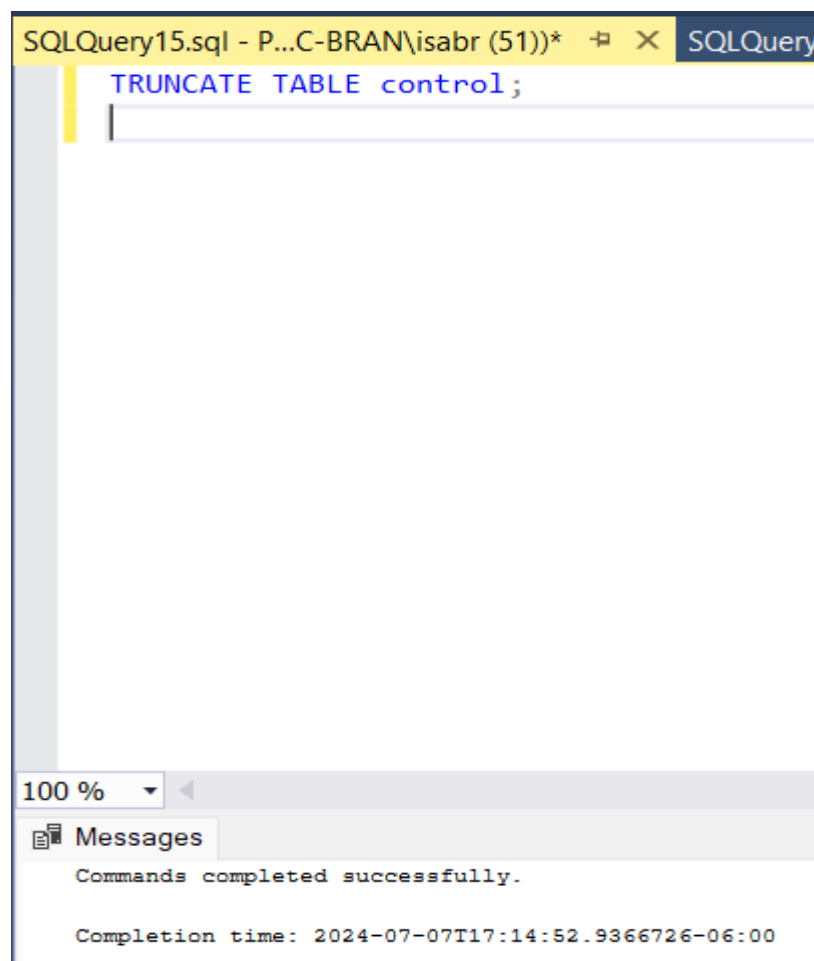
100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:13:05.9745141-06:00

3. Vaciado de la tabla control



```
TRUNCATE TABLE control;
```

100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:14:52.9366726-06:00

3. Actualización del precio de un libro

```
SQLQuery16.sql - P...C-BRAN\isabr (57))*  SQLQuery15.sql - P...C-BRAI
UPDATE libros SET precio = 54.99 WHERE codigo = 100;

100 %
Messages
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
Completion time: 2024-07-07T17:16:02.7189254-06:00
```

4. Verificación del contenido de la tabla control

```
SQLQuery17.sql - P...C-BRAN\isabr (73))*  SQLQuery16.sql - P...C-BRAN\isabr
SELECT * FROM control;
```

100 %

Results Messages

	usuario	fecha	codigo_libro	precio_anterior	precio_nuevo
1	PC-Bran\isabr	2024-07-07 17:16:02.700	100	30.00	54.99
2	PC-Bran\isabr	2024-07-07 17:16:02.703	100	54.99	54.00
3	PC-Bran\isabr	2024-07-07 17:16:02.703	100	30.00	54.99

Ejemplo 4: Disparador para Múltiples Eventos

1. Creación de un disparador para múltiples eventos

```
SQLQuery18.sql - P...C-BRAN\isabr (56))* X
CREATE TRIGGER trg_libros_all_events
ON libros
FOR INSERT, UPDATE, DELETE
AS
BEGIN
    DECLARE @event VARCHAR(10), @precio_anterior DECIMAL(10, 2), @precio_nuevo DECIMAL(10, 2);

    IF EXISTS (SELECT * FROM inserted) AND EXISTS (SELECT * FROM deleted)
    BEGIN
        SET @event = 'UPDATE';
        SELECT @precio_anterior = d.precio, @precio_nuevo = i.precio
        FROM inserted i
        JOIN deleted d ON i.codigo = d.codigo;
    END
    ELSE IF EXISTS (SELECT * FROM inserted)
    BEGIN
        SET @event = 'INSERT';
        SELECT @precio_nuevo = precio FROM inserted;
    END
    ELSE IF EXISTS (SELECT * FROM deleted)
    BEGIN
        SET @event = 'DELETE';
        SELECT @precio_anterior = precio FROM deleted;
    END

    -- Actualizar la tabla control
    UPDATE control SET evento = @event, precio_ant = @precio_anterior, precio_nue = @precio_nuevo;
END
```

100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:24:29.5048115-06:00

2. Realización de operaciones INSERT, DELETE y UPDATE y verificación del funcionamiento del disparador

```
SQLQuery19.sql - P...C-BRAN\isabr (52))* X SQLQuery18.sql - P...C-BRAN\isabr (56))*
-- Insertar un nuevo libro
INSERT INTO libros VALUES (150, 'Nuevo Libro', 'Nuevo Autor', 'Nueva Editorial', 45);

-- Eliminar un libro
DELETE FROM libros WHERE codigo = 103;

-- Actualizar el precio de un libro
UPDATE libros SET precio = 40 WHERE codigo = 105;

-- Verificar el contenido de la tabla control
SELECT * FROM control;
```

100 %

Results Messages

	usuario	fecha	codigo_libro	precio_anterior	precio_nuevo
1	PC-Branlisabr	2024-07-07 17:16:02.700	100	30.00	54.99
2	PC-Branlisabr	2024-07-07 17:16:02.703	100	54.99	54.00
3	PC-Branlisabr	2024-07-07 17:16:02.703	100	30.00	54.99
4	PC-Branlisabr	2024-07-07 17:25:08.630	150	NULL	45.00
5	PC-Branlisabr	2024-07-07 17:25:08.640	103	28.00	NULL
6	PC-Branlisabr	2024-07-07 17:25:08.643	105	12.00	40.00
7	PC-Branlisabr	2024-07-07 17:25:08.647	105	12.00	40.00
8	PC-Branlisabr	2024-07-07 17:25:08.647	105	12.00	40.00
9	PC-Branlisabr	2024-07-07 17:25:08.647	105	12.00	40.00

Análisis de Resultados

1. Creación de un disparador que se active al modificar empleados

```
SQLQuery20.sql - P...C-BRAN\isabr (67))*  X
-- Crear la tabla empleados
CREATE TABLE empleados (
    documento CHAR(8) NOT NULL,
    nombre VARCHAR(50) NOT NULL,
    domicilio VARCHAR(50),
    seccion VARCHAR(20)
);
GO

-- Crear la tabla controlCambios
CREATE TABLE controlCambios (
    usuario VARCHAR(50),
    fecha DATETIME,
    datoanterior VARCHAR(50),
    datonuevo VARCHAR(50)
);
GO

100 %
Messages
Commands completed successfully.

Completion time: 2024-07-07T17:29:31.3550337-06:00
```

2. Inserción de registros en empleados

```
SQLQuery21.sql - P...C-BRAN\isabr (54))*  X SQLQuery20.sql - P...C-BRAN\isabr (67))*
INSERT INTO empleados VALUES ('22222222', 'Ana Acosta', 'Bulnes 56', 'Secretaria');
INSERT INTO empleados VALUES ('23333333', 'Bernardo Bustos', 'Bulnes 188', 'Contaduria');
INSERT INTO empleados VALUES ('24444444', 'Carlos Caseres', 'Caseros 364', 'Sistemas');
INSERT INTO empleados VALUES ('25555555', 'Diana Duarte', 'Colon 1234', 'Sistemas');
INSERT INTO empleados VALUES ('26666666', 'Diana Duarte', 'Colon 897', 'Sistemas');
INSERT INTO empleados VALUES ('27777777', 'Matilda Morales', 'Colon 542', 'Gerencia');
GO

100 %
Messages
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)

Completion time: 2024-07-07T17:30:24.8177273-06:00
```

3. Creación de un disparador que se active al insertar en empleados

```
SQLQuery22.sql - P...C-BRAN\isabr (53))* X SQLQuery21.sql - P...C-BRAN\isabr (54))* SQLQ

CREATE TRIGGER trg_empleados_insert
ON empleados
FOR INSERT
AS
BEGIN
    INSERT INTO controlCambios (usuario, fecha, datoanterior, datonuevo)
    SELECT SYSTEM_USER, GETDATE(), NULL, i.documento
    FROM inserted i;
END;
GO
```

100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:31:01.1439958-06:00

4. Creación de un disparador que se active al eliminar en empleados

```
SQLQuery23.sql - P...C-BRAN\isabr (69))* X SQLQuery22.sql - P...C-BRAN\isabr (53))* SQLQ

CREATE TRIGGER trg_empleados_delete
ON empleados
FOR DELETE
AS
BEGIN
    INSERT INTO controlCambios (usuario, fecha, datoanterior, datonuevo)
    SELECT SYSTEM_USER, GETDATE(), d.documento, NULL
    FROM deleted d;
END;
GO
```

100 %

Messages

Commands completed successfully.

Completion time: 2024-07-07T17:31:41.2135841-06:00