TRIGGER

AFTER TRIGGER

Descontar stock a travez de un insert

Sobre que tabla:

```
ON VENTAS
```

1) Se ejecuta el insert → AFTER INSERT
2) Se descuenta el stock

Se genera un TRIGGER en la tabla de VENTAS, este reacciona después de hacer el INSERT

```
CREATE TRIGGER TR_AGREGAR_VENTA ON VENTAS
AFTER INSERT
AS
BEGIN

END
```

1) Registrar la VENTA (hecho)
2) Continuar con los siguientes:
   - Tabla temporal : INSERTED

Al ejecutar este TRIGGER genera un objeto en la base de datos que dispara este código siempre y cuando se ejecute un INSERT en la tabla VENTAS previamente.

```
CREATE TRIGGER TR_AGREGAR_VENTA ON VENTAS
AFTER INSERT
AS
BEGIN

    BEGIN TRY
        BEGIN TRANSACTION
            -- 1. REGISTRAR LA VENTA (OK)
            -- 2. DESCONTAR EL STOCK
                -- CONOCER EL IDARTICULO Y LA CANTIDAD A DESCONTAR
                -- REALIZAR EL UPDATE AL ARTÍCULO CORRESPONDIENTE

            DECLARE @IDARTICULO BIGINT
            DECLARE @CANTIDAD INT
            SELECT @IDARTICULO = IDARTICULO, @CANTIDAD = CANTIDAD FROM INSERTED

            UPDATE ARTICULOS SET STOCK = STOCK - @CANTIDAD WHERE IDARTICULO = @IDARTICULO

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH

END
```

```
SELECT * FROM ARTICULOS
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	1	ROUTER	550,00	17	5	I	1
2	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
3	3	MOUSE INALÁMBRICO	100,00	0	12	N	1
4	4	PENDRIVE 8GB	130,00	5	10	N	1
5	5	PENDRIVE 16GB	170,00	5	10	N	1
6	6	PENDRIVE 32GB	240,00	6	10	N	1
7	7	PENDRIVE 64GB	320,00	4	10	N	1
8	8	DISCO RÍGIDO EXTERNO 1TB	980,00	5	2	I	1
9	9	DISCO RÍGIDO EXTERNO 2TB	1400,00	5	2	I	1
10	10	JOYSTICK INALÁMBRICO	250,00	10	10	N	1
11	11	NOTEBOOK	12000,00	3	1	N	1
12	12	ULTRABOOK	12000,00	2	1	I	1
13	13	RASPBERRY PI	1100,00	0	0	I	1

Al ejecutar el INSERT podemos ver que hay 2 filas afectadas:

```
INSERT INTO VENTAS(IDARTICULO, FECHA, DNI, CANTIDAD, IMPORTE)
VALUES(12, GETDATE(), 1, 1, 0)
```

Una para tabla VENTAS y la otra para tabla ARTICULOS:

```
(1 filas afectadas)

(1 filas afectadas)
```

Verificamos

```
SELECT * FROM VENTAS
```

	IDVENTA	FECHA	DNI	CANTIDAD	IDARTICULO	IMPORTE
1	25	2016-05-27 00:13:00	1	1	12	0,00

```
SELECT * FROM ARTICULOS
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
	1	ROUTER	550,00	17	5	I	1
	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
	3	MOUSE INALÁMBRICO	100,00	0	12	N	1
	4	PENDRIVE 8GB	130,00	5	10	N	1
	5	PENDRIVE 16GB	170,00	5	10	N	1
	6	PENDRIVE 32GB	240,00	6	10	N	1
	7	PENDRIVE 64GB	320,00	4	10	N	1
	8	DISCO RÍGIDO EXTERNO 1TB	980,00	5	2	I	1
	9	DISCO RÍGIDO EXTERNO 2TB	1400,00	5	2	I	1
	10	JOYSTICK INALÁMBRICO	250,00	10	10	N	1
	11	NOTEBOOK	12000,00	3	1	N	1
	12	ULTRABOOK	12000,00	1	1	I	1
	13	RASPBERRY PI	1100,00	0	0	I	1

Al intentar generar este INSERT, nos va a hacer ROLLBACK por tener 1 de stock y querer hacer una venta de 3 unidades. Generando el stock en negativo:

```
INSERT INTO VENTAS(IDARTICULO, FECHA, DNI, CANTIDAD, IMPORTE)
VALUES(12, GETDATE(), 1, 3, 0)
```

```
Mensajes

(0 filas afectadas)
Mens. 3609, Nivel 16, Estado 1, Línea 1
La transacción terminó en el desencadenador. Se anuló el lote.
```

De esta forma al generar error en el INSERT, se anula el TRIGGER sin provocar cambios en ninguna de las 2 tablas.

GENERAR REGISTRO DE FECHA Y PRECIO SIN IMPORTAR SI GENERA ERROR EL INSERT

Por mas que genere error, el TRIGGER tiene que realizar un registro en la tabla de VENTA con el importe que iba a tener y la FECHA, sin perjudicar el stock y la cantidad.

Borramos el TRIGGER anterior:

```
DROP TRIGGER TR_AGREGAR_VENTA
```

Vamos a querer que nuestra línea de INSERT no se ejecute, para luego reescribir el insert como nosotros queremos y descontar el stock:

Luego del INSERT erróneo, se va a ejecutar el lugar del insert, captura el lugar del insert en la tabla de ventas y ejecuta este código:

1) Registramos la venta manualmente
2) Luego descontar el stock

```
CREATE TRIGGER TR_AGREGAR_VENTA ON VENTAS
INSTEAD OF INSERT
AS
BEGIN

    BEGIN TRY
        BEGIN TRANSACTION
            -- 1. REGISTRAR LA VENTA
            DECLARE @IDARTICULO BIGINT
            DECLARE @CANTIDAD INT
            SELECT @IDARTICULO = IDARTICULO, @CANTIDAD = CANTIDAD FROM INSERTED

            DECLARE @PU MONEY
            SELECT @PU = PRECIO FROM ARTICULOS WHERE IDARTICULO = @IDARTICULO

            INSERT INTO VENTAS(IDARTICULO, CANTIDAD, FECHA, DNI, IMPORTE)
            SELECT IDARTICULO, CANTIDAD, GETDATE(), DNI, @PU*@CANTIDAD FROM INSERTED

            -- 2. DESCONTAR EL STOCK
                -- REALIZAR EL UPDATE AL ARTICULO CORRESPONDIENTE

            UPDATE ARTICULOS SET STOCK = STOCK - @CANTIDAD WHERE IDARTICULO = @IDARTICULO

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH

END
```

No se ejecuto correctamente el insert pero los valores quedaron guardados en la tabla INSERTED.Si hay un error en cualquiera de las dos consultas genera el rollback sin ser ejecutadas el insert y update.

Verificar:

```
SELECT * FROM ARTICULOS
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	1	ROUTER	550,00	17	5	I	1
2	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
3	3	MOUSE INALÁMBRICO	100,00	0	12	N	1
4	4	PENDRIVE 8GB	130,00	5	10	N	1
5	5	PENDRIVE 16GB	170,00	5	10	N	1
6	6	PENDRIVE 32GB	240,00	6	10	N	1
7	7	PENDRIVE 64GB	320,00	4	10	N	1
8	8	DISCO RÍGIDO EXTERNO 1TB	980,00	5	2	I	1
9	9	DISCO RÍGIDO EXTERNO 2TB	1400,00	5	2	I	1
10	10	JOYSTICK INALÁMBRICO	250,00	10	10	N	1
11	11	NOTEBOOK	12000,00	3	1	N	1
12	12	ULTRABOOK	12000,00	1	1	I	1
13	13	RASPBERRY PI	1100,00	0	0	I	1

Realizamos el INSERT colocando una fecha X y de importe 0. De esta forma vamos a poder ver que se ejecuta correctamente pero la fecha se va a insertar GETDATE() y el importe va ser PU*Cantidad del TRIGGER

```
INSERT INTO VENTAS(IDARTICULO, FECHA, DNI, CANTIDAD, IMPORTE)
VALUES (10, '1/1/2000', 1, 2, 0)
```

Como alcanza el stock se pudo realizar correctamente:

```
(1 filas afectadas)

(1 filas afectadas)

(1 filas afectadas)
```

```
SELECT * FROM ARTICULOS
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	1	ROUTER	550,00	17	5	I	1
2	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
3	3	MOUSE INALÁMBRIICO	100,00	0	12	N	1
4	4	PENDRIVE 8GB	130,00	5	10	N	1
5	5	PENDRIVE 16GB	170,00	5	10	N	1
6	6	PENDRIVE 32GB	240,00	6	10	N	1
7	7	PENDRIVE 64GB	320,00	4	10	N	1
8	8	DISCO RIGIDO EXTERNO 1TB	980,00	5	2	I	1
9	9	DISCO RIGIDO EXTERNO 2TB	1400,00	5	2	I	1
10	10	JOYSTICK INALÁMBRICO	250,00	8	10	N	1
11	11	NOTEBOOK	12000,00	3	1	N	1
12	12	ULTRABOOK	12000,00	1	1	I	1
13	13	RASPBERRY PI	1100,00	0	0	I	1

```
SELECT * FROM VENTAS
```

	IDVENTA	FECHA	DNI	CANTIDAD	IDARTICULO	IMPORTE
1	26	2016-05-27 11:02:00	1	2	10	500,00

Fecha del sistema siempre

PU*cantidad

El insert original que ejecutamos no se registro pero disparó el TRIGGER en la tabla venta reescribiendo el insert y descontando el stock correctamente.

TRIGGER a partir de un DELETE

Baja lógica: Una vez de ejecutar el DELETE, va a reemplazarlo por un UPDATE con el TRIGGER modificando el estado de 1 a 0.

```
CREATE TRIGGER TR_BAJALOGICA_ARTICULO ON ARTICULOS
INSTEAD OF DELETE
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH
END
```

```
CREATE TRIGGER TR_BAJALOGICA_ARTICULO ON ARTICULOS
INSTEAD OF DELETE
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION

            DECLARE @IDARTICULO BIGINT
            SELECT @IDARTICULO = IDARTICULO FROM DELETED

            UPDATE ARTICULOS SET ESTADO = 0 WHERE IDARTICULO = @IDARTICULO

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH
END
```

```
DELETE FROM ARTICULOS WHERE IDARTICULO = 1
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	1	ROUTER	550,00	17	5	I	1

```
(1 filas afectadas)

(1 filas afectadas)
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
	1	ROUTER	550,00	17	5	I	0

Para desactivar el TRIGGER para poder borrar el registro de la tabla podemos hacer:

```
DISABLE TRIGGER TR_BAJALOGICA_ARTICULO ON ARTICULOS
```

Volvemos a ejecutar el DELETE y elimina el articulo:

```
DELETE FROM ARTICULOS WHERE IDARTICULO = 1
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
2	3	MOUSE INALÁMBRICO	100,00	0	12	N	1

Luego habilitamos el TRIGGER nuevamente para que siga funcionando:

```
ENABLE TRIGGER TR_BAJALOGICA_ARTICULO ON ARTICULOS
```

TRIGGER por UPDATE

Luego de un UPDATE el stock queda por debajo del stock mínimo entonces genere automáticamente una orden de pedido

Un trigger puede desencadenar otro trigger cuando hacemos una venta descueta el stock y dispara otro trigger para que verifique si se realiza la modificación o no.

```
CREATE TRIGGER TR_MODIFICAR_ARTICULO ON ARTICULOS
AFTER UPDATE
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH
END
```

```
CREATE TRIGGER TR_MODIFICAR_ARTICULO ON ARTICULOS
AFTER UPDATE
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION
            DECLARE @STOCK INT
            DECLARE @STOCK_MIN INT
            DECLARE @IDARTICULO BIGINT

            SELECT @IDARTICULO = IDARTICULO, @STOCK = STOCK FROM INSERTED
            SELECT @STOCK_MIN = STOCK_MIN FROM ARTICULOS WHERE IDARTICULO = @IDARTICULO

            IF @STOCK < @STOCK_MIN BEGIN
                INSERT INTO PEDIDOS(IDARTICULO, CANTIDAD, FECHA)
                VALUES (@IDARTICULO, @STOCK_MIN-@STOCK, GETDATE())
            END
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION
    END CATCH
END
```

```
SELECT * FROM ARTICULOS
```

Resultados | Mensajes

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	2	TECLADO INALÁMBRICO	190,00	20	10	N	1
2	3	MOUSE INALÁMBRICO	100,00	15	12	N	1

```
UPDATE ARTICULOS SET STOCK = 9 WHERE IDARTICULO = 2
```

```
SELECT * FROM ARTICULOS
```

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	2	TECLADO INALÁMBRICO	190,00	9	10	N	1
2	3	MOUSE INALÁMBRICO	100,00	15	12	N	1

```
SELECT * FROM PEDIDOS
```

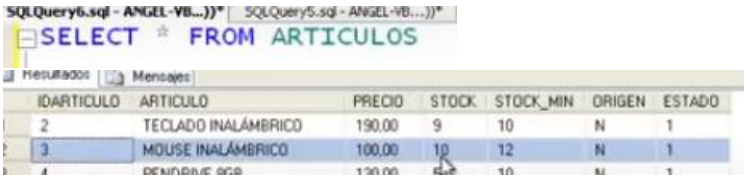
	IDPEDIDO	IDARTICULO	CANTIDAD	FECHA
1	2	2	1	2016-05-27

TRIGGER EN CADENA

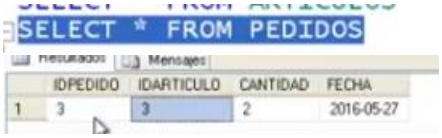
Genera un trigger donde pide una venta de 10 unidades pero al tener stock de 15 y tener 12 de stock_minimo, ejecuta el trigger anterior para generar una orden de pedido:

	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
1	2	TECLADO INALÁMBRICO	190,00	9	10	N	1
2	3	MOUSE INALÁMBRICO	100,00	15	12	N	1
3	4	PENDRIVE 8GB	130,00	5	10	N	1
4	5	PENDRIVE 16GB	170,00	5	10	N	1
5	6	PENDRIVE 32GB	240,00	6	10	N	1

	IDVENTA	FECHA	DNI	CANTIDAD	IDARTICULO	IMPORTE
1	27	2016-05-27 12:14:00	1	5	3	500,00



	IDARTICULO	ARTICULO	PRECIO	STOCK	STOCK_MIN	ORIGEN	ESTADO
	2	TECLADO INALÁMBRICO	190,00	9	10	N	1
	3	MOUSE INALÁMBRICO	100,00	10	12	N	1



	IDPEDIDO	IDARTICULO	CANTIDAD	FECHA
1	3	3	2	2016-05-27

- TRIGGER Es un fragmento de código que se va ejecutar automáticamente si ocurre cierta consulta de acción en alguna tabla en particular.
- El código de nuestro trigger puede ejecutarse en lugar de la consulta de acción en esa tabla o luego de la consulta de acción.
- Un rollback transaction en nuestro trigger va a retrotraer los cambios tanto del trigger como de la consulta de acción que lo desencadeno.
- Tablas INSERTED y DELETED son tablas temporales que contienen los datos que van a ser ingresados en nuestra tabla, va a figurar en la tabla INSERTED o los datos que van a ser eliminados se encuentran en la tabla DELETED
- TRIGGER puede desencadenar a otro, por eso hay que ser cuidadoso y documentar bien el código para que no traiga problemas.