Ejemplo triggers

15.Anidados.sql
16.Evitar anidados.sql

Ejemplo triggers

Ejemplo

Definimos desencadenadores para realizar el mantenimiento de totales de ventas en diferentes niveles:

1. Insertamos, modificamos o eliminamos de la tabla "Order Details". Esta modificación de datos hace que se ejecute el trigger AFTER UPDATE.

2. EL desencadenador AFTER UPDATE de la tabla "Order Details" actualiza la columna TotalVentas en la tabla Orders.

3. Como la columna TotalVentas en la tabla Orders ha sufrido modificaciones, el trigger AFTER UPDATE definido en la tabla Orders se ejecuta automáticamente y actualiza la columna TotalVentas en las tablas Employees y Customers.

Tabla: Employees

Descripción
Nombres de empleados, títulos y información personal.

Name	Type	Descripción
Address	Text(60)	Street or post-office box.
BirthDate	Date/Time	Employee's date of birth.
City	Text(15)	Name of city where employee lives
Country	Text(15)	Name of country where employee lives.
EmployeeID	AutoNumber	Number automatically assigned to new employee.
Extension	Text(4)	Internal telephone extension number.
FirstName	Text(10)	First name of employee.
HireDate	Date/Time	Date that employee was hired.
HomePhone	Text(24)	Phone number includes country code or area code.
LastName	Text(20)	Last name of employee.
Notes	Memo	General information about employee's background.
Photo	Text(255)	Picture of employee.
PostalCode	Text(10)	ZIP code where employee lives.
Region	Text(15)	State or province
ReportsTo	Long Integer	Employee's supervisor.
Title	Text(30)	Employee's title.
TitleOfCourtesy	Text(25)	Title used in salutations.

Order Details

Descripción
Detalles de productos, cantidades, y precios de cada pedido de la tabla orders.

Column_name	Data type	Nullable	Default	Check	Key/index
OrderID	**int**	no			Composite PK, clust[1], FK **Orders**(**OrderID**)[2]
ProductID	**int**	no			Composite PK, clust[1], FK **Products**(**ProductID**)[3]
UnitPrice	**money**	no	0	yes[4]	
Quantity	**smallint**	no	1	yes[5]	
Discount	**real**	no	0		

Tabla: Orders

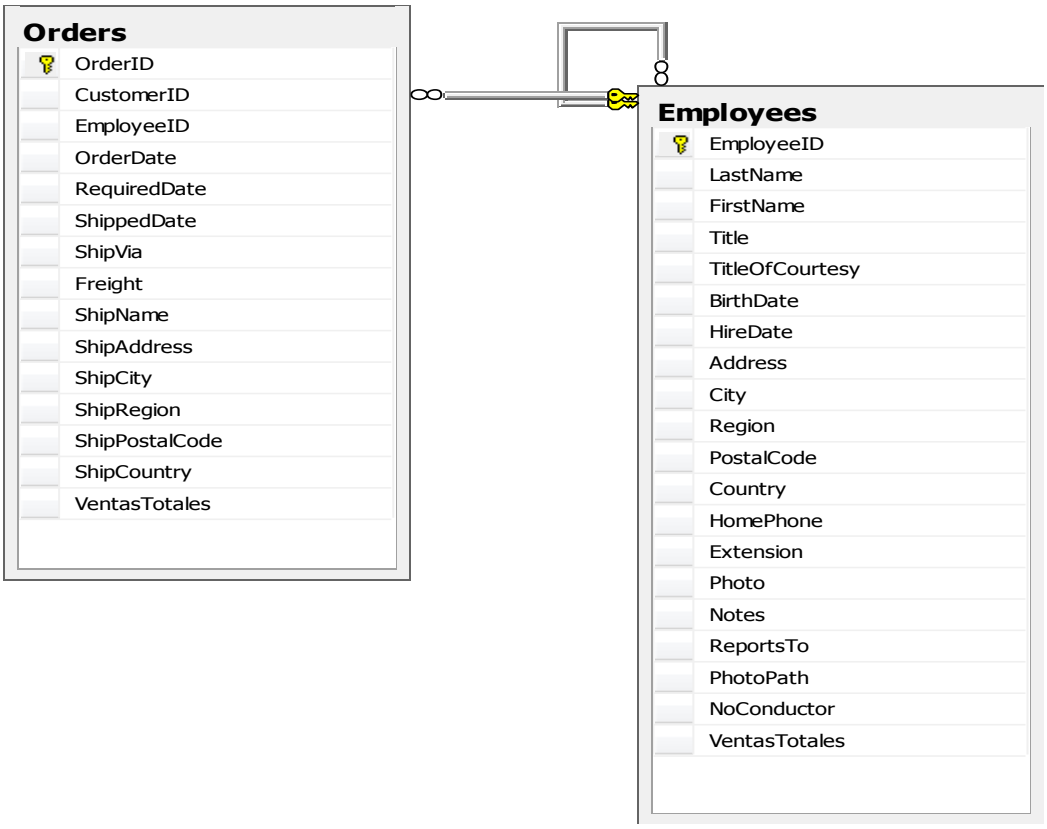
Description
Customer name, order date, and freight charge for each order.

Name	**Tipo**	**Descripción**
CustomerID	Text(5)	
EmployeeID	Long Integer	Same entry as in Employees table.
Freight	Currency	Shipping cost.
OrderDate	Date/Time	Date of order.
OrderID	AutoNumber	Unique order number.
RequiredDate	Date/Time	Required by date.
ShipAddress	Text(60)	Street address only -- no post-office box allowed.
ShipCity	Text(15)	Name of city where order was shipped.
ShipCountry	Text(15)	Name of country where order was shipped.
ShipName	Text(40)	Name of person or company to receive the shipment.
ShippedDate	Date/Time	Date that order was shipped.
ShipPostalCode	Text(10)	ZIP code where order was shipped.
ShipRegion	Text(15)	State or province.
ShipVia	Long Integer	Same as Shipper ID in Shippers table.

Table: Customers

Nombre, dirección y teléfono de Clientes.

Name	Type	Description
Address	Text(60)	Street or post-office box.
City	Text(15)	Name of city where customer is located.
CompanyName	Text(40)	Name of customer company.
ContactName	Text(30)	Name of contact person.
ContactTitle	Text(30)	Title of contact person.
Country	Text(15)	Name of country where customer is located.
CustomerID	Text(5)	Unique five-character code based on customer name.
Fax	Text(24)	Phone number includes country code or area code.
Phone	Text(24)	Phone number includes country code or area code.
PostalCode	Text(10)	ZIP code where customer is located.
Region	Text(15)	State or province.

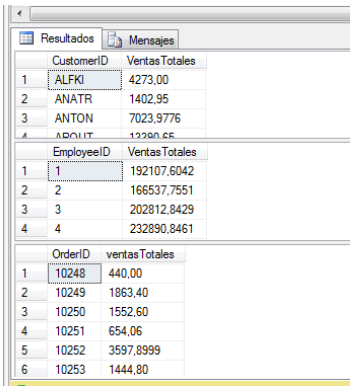


Creación del ejemplo

```
USE Northwind
GO
-- Añadimos la columna VentasTotales a la tabla Orders
ALTER TABLE Orders
ADD VentasTotales money NULL
-- Añadimos la columna VentasTotales a la tabla Employees
LTER TABLE Employees
ADD VentasTotales money NULL
-- Añadimos la columna VentasTotales a la tabla Customers
ALTER TABLE Customers
ADD VentasTotales money NULL
GO
-- Inicializamos los datos
UPDATE Orders
SET VentasTotales =(
      SELECT SUM([Order Details].UnitPrice * Quantity * (1 - Discount))
      FROM [Order Details]
      WHERE [Order Details].OrderID = Orders.OrderID
      )
UPDATE Employees
SET VentasTotales =(
      SELECT SUM(Orders.VentasTotales)
      FROM Orders
      WHERE Orders.EmployeeID = Employees.EmployeeID
      )

UPDATE Customers
SET VentasTotales =(
      SELECT SUM(Orders.VentasTotales)
      FROM Orders
      WHERE Orders.CustomerID = Customers.CustomerID
      )

GO
```

Prueba de los datos creados

```
-- Muestra la información de vetas totales de cada cliente, empleado y
pedido
Select CustomerID, VentasTotales from Customers
GO
Select EmployeeID,VentasTotales from Employees
GO
Select OrderID, ventasTotales FROM Orders
```



Resultados | Mensajes

	CustomerID	VentasTotales
1	ALFKI	4273,00
2	ANATR	1402,95
3	ANTON	7023,9776
4	AROUT	13290,65

	EmployeeID	VentasTotales
1	1	192107,6042
2	2	166537,7551
3	3	202812,8429
4	4	232890,8461

	OrderID	ventasTotales
1	10248	440,00
2	10249	1863,40
3	10250	1552,60
4	10251	654,06
5	10252	3597,8999
6	10253	1444,80

Crear trigger para Order details

```
CREATE TRIGGER TR_TotalOrderDetails
ON [Order details]
AFTER INSERT, DELETE, UPDATE
AS
      IF @@rowcount = 1
            -- Operación sobre una sola fila
            UPDATE Orders
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Inserted
                        WHERE Inserted.OrderID = Orders.OrderID), 0)
                  - ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Deleted
                        WHERE Deleted.OrderID = Orders.OrderID), 0)

      ELSE
            -- Operación sobre varias filas
            UPDATE Orders
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Inserted
                        WHERE Inserted.OrderID = Orders.OrderID), 0)
                  - ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Deleted
                        WHERE Deleted.OrderID = Orders.OrderID), 0)

GO
```

Creación de trigger para Orders

```
CREATE TRIGGER TR_TotalOrders
ON Orders
AFTER INSERT, DELETE, UPDATE
AS
      IF @@rowcount = 1
      BEGIN
            -- Operación sobre una sola fila
            UPDATE Employees
            SET VentasTotales = VentasTotales
                  + ISNULL(
                  (SELECT VentasTotales
                  FROM Inserted
                  WHERE Inserted.EmployeeID = Employees.EmployeeID), 0)
                  - ISNULL(
                  (SELECT VentasTotales
                  FROM Deleted
                  WHERE Deleted.EmployeeID = Employees.EmployeeID), 0)

            UPDATE Customers
            SET VentasTotales = VentasTotales
                  + ISNULL(
                  (SELECT VentasTotales
                  FROM Inserted
                  WHERE Inserted.CustomerID = Customers.CustomerID), 0)
            - ISNULL(
                  (SELECT VentasTotales
```

```
                FROM Deleted
                WHERE Deleted.CustomerID = Customers.CustomerID), 0)

        END
        ELSE
        BEGIN
              -- Operación sobre varias filas
              UPDATE Employees
              SET VentasTotales = VentasTotales
                    + ISNULL(
                    (SELECT SUM(VentasTotales)
                    FROM Inserted
                    WHERE Inserted.EmployeeID = Employees.EmployeeID), 0)
                    - ISNULL(
                    (SELECT SUM(VentasTotales)
                    FROM Deleted
                    WHERE Deleted.EmployeeID = Employees.EmployeeID), 0)

              UPDATE Customers
              SET VentasTotales = VentasTotales
                    + ISNULL(
                    (SELECT SUM(VentasTotales)
                    FROM Inserted
                    WHERE Inserted.CustomerID = Customers.CustomerID), 0)
                    - ISNULL(
                    (SELECT SUM(VentasTotales)
                    FROM Deleted
                    WHERE Deleted.CustomerID = Customers.CustomerID), 0)
        END
```

Test de triggers

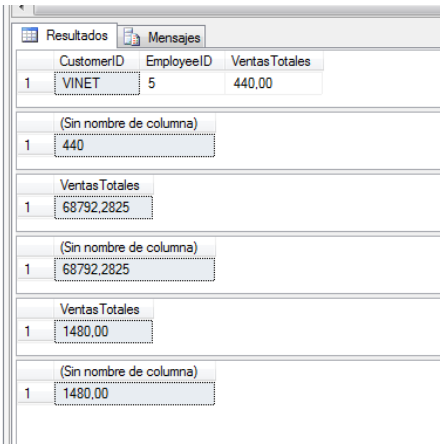
Vamos a realizar una actualización del pedido 10248. Este pedido corresponde al empleado número 5 y al cliente VNET. Por lo que con la actualización del pedido se deverian actualizar Los totales del pedido de la tabla orders. Los totales del empleado y los totales del Clliente.

Vemos la información que tenmos antes de realizar los cambios:

```
SELECT CustomerID
      , EmployeeID
      , VentasTotales
FROM orders
WHERE OrderID =  10248

SELECT SUM([Order Details].UnitPrice * Quantity * (1 - Discount))
FROM [Order Details]
WHERE OrderID =  10248

-- Comprobamos los totales en la tabla Employees

SELECT VentasTotales
FROM Employees
WHERE EmployeeID = 5

SELECT SUM(VentasTotales)
FROM Orders
WHERE EmployeeID =  5

-- Comprobamos los totales en la tabla Customers
```

```
SELECT VentasTotales
FROM Customers
WHERE CustomerID = 'VINET'

SELECT SUM(VentasTotales)
FROM Orders
WHERE CustomerID =  'VINET'
GO
```

Resultado:



Realizamos la actualización:

```
UPDATE [Order Details]
SET quantity = 100
WHERE orderid = 10248
      AND productid = 11
```

Volvemos a ver los datos:

```
SELECT CustomerID
      , EmployeeID
      , VentasTotales
FROM orders
WHERE OrderID =  10248

SELECT SUM([Order Details].UnitPrice * Quantity * (1 - Discount))
FROM [Order Details]
WHERE OrderID =  10248
-- Comprobamos los totales en la tabla Employees
SELECT VentasTotales
FROM Employees
WHERE EmployeeID = 5

SELECT SUM(VentasTotales)
FROM Orders
WHERE EmployeeID =  5
-- Comprobamos los totales en la tabla Customers
SELECT VentasTotales
FROM Customers
WHERE CustomerID = 'VINET'

SELECT SUM(VentasTotales)
FROM Orders
WHERE CustomerID =  'VINET'
GO
```

Deshacer el ejemplo:

```
DROP TRIGGER isrTotalOrderDetails

DROP TRIGGER isrTotalOrders

UPDATE [Order Details]
SET quantity = 12
WHERE orderid = 10248
      AND productid = 11

ALTER TABLE Orders
DROP VentasTotales

-- Añadimos la columna VentasTotales a la tabla Employees

ALTER TABLE Employees
DROP VentasTotales

-- Añadimos la columna VentasTotales a la tabla Customers

ALTER TABLE Customers
DROP VentasTotales

GO
```

Ejemplo2

EVITAR TRIGGERS ANIDADOS

Deberíamos evitar el uso de los triggers anidados. Es más recomendable crear un trigger para cada acción lógica. Cada tabla puede tener varios triggers para cada acción.

```
Nota:
Los ejemplos del Tema 15. y 16. crean un único trigger para tres acciones
(INSERT, UPDATE Y DELETE).
En términos de rendimiento, es más eficiente crear triggers individuales
para cada acción.
```

Crear trigger para Order details

```
USE Northwind
GO

CREATE TRIGGER tr_OrderDetails_TotalOrders
ON [Order details]
AFTER INSERT, DELETE, UPDATE
AS

      IF @@rowcount = 1

            -- Operación sobre una sola fila

            UPDATE Orders
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
```

```
                        FROM Inserted
                        WHERE Inserted.OrderID = Orders.OrderID), 0)
                  - ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Deleted
                        WHERE Deleted.OrderID = Orders.OrderID), 0)

      ELSE

            -- Operación sobre varias filas

            UPDATE Orders
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Inserted
                        WHERE Inserted.OrderID = Orders.OrderID), 0)
                  - ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Deleted
                        WHERE Deleted.OrderID = Orders.OrderID), 0)

GO


CREATE TRIGGER tr_OrderDetails_TotalEmployees
ON [Order details]
AFTER INSERT, DELETE, UPDATE
AS

      IF @@rowcount = 1

            -- Operación sobre una sola fila

            UPDATE Employees
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Inserted
                              JOIN Orders
                                    ON Inserted.OrderID = Orders.OrderID
                        WHERE Orders.EmployeeID = Employees.EmployeeID), 0)
                  - ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Deleted
                              JOIN Orders
                                    ON Deleted.OrderID = Orders.OrderID
                        WHERE Orders.EmployeeID = Employees.EmployeeID), 0)

      ELSE

            -- Operación sobre varias filas

            UPDATE Employees
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Inserted
                              JOIN Orders
                                    ON Inserted.OrderID = Orders.OrderID
                        WHERE Orders.EmployeeID = Employees.EmployeeID), 0)
                  - ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
```

```
                        FROM Deleted
                              JOIN Orders
                                    ON Deleted.OrderID = Orders.OrderID
                        WHERE Orders.EmployeeID = Employees.EmployeeID), 0)

GO

CREATE TRIGGER tr_OrderDetails_TotalCustomers
ON [Order details]
AFTER INSERT, DELETE, UPDATE
AS

      IF @@rowcount = 1

            -- Operación sobre una sola fila

            UPDATE Customers
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Inserted
                              JOIN Orders
                                    ON Inserted.OrderID = Orders.OrderID
                        WHERE Orders.CustomerID = Customers.CustomerID), 0)
                  - ISNULL(
                        (SELECT UnitPrice * Quantity * (1 - Discount)
                        FROM Deleted
                              JOIN Orders
                                    ON Deleted.OrderID = Orders.OrderID
                        WHERE Orders.CustomerID = Customers.CustomerID), 0)

      ELSE

            -- Operación sobre varias filas

            UPDATE Customers
            SET VentasTotales = VentasTotales
                  + ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Inserted
                              JOIN Orders
                                    ON Inserted.OrderID = Orders.OrderID
                        WHERE Orders.CustomerID = Customers.CustomerID), 0)
                  - ISNULL(
                        (SELECT SUM(UnitPrice * Quantity * (1 - Discount))
                        FROM Deleted
                              JOIN Orders
                                    ON Deleted.OrderID = Orders.OrderID
                        WHERE Orders.CustomerID = Customers.CustomerID), 0)
GO

DROP TRIGGER tr_OrderDetails_TotalOrders

DROP TRIGGER tr_OrderDetails_TotalCustomers

DROP TRIGGER tr_OrderDetails_TotalEmployees

GO
```

Más información

C/ Miracruz, 10 (Bº de Gros) 20001 Donostia

Telf.: 943 275819

email: seim@centroseim.com

S.E.I.M.