|  |
| --- |
|  |
| Ejemplo triggers |
|  |
| *Mod\_columnas.sql* |

Ejemplo triggers

# Ejemplo1

El ejemplo siguiente crea un desencadenador que impide la realización de cambios en el primarykey ORDERID e imprime un mensaje por cada una de los siguientes campos que se actualiza:

* ProductID
* UnitPrice
* Quantity

Razón por la que crear el trigger:

No se quiere evitar el cumplimiento de la instrucción, sino que se quiere enviar un mensaje por cada campo actualizado.

# 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, clust1, FK  **Orders**(**OrderID**)2 |
| **ProductID** | **int** | no |  |  | Composite PK, clust1, FK  **Products**(**ProductID**)3 |
| **UnitPrice** | **money** | no | 0 | yes4 |  |
| **Quantity** | **smallint** | no | 1 | yes5 |  |
| **Discount** | **real** | no | 0 |  |  |

1 The composite, primary key, clustered index is defined on **OrderID** and **ProductID**.  
2 There are also two nonclustered indexes on **OrderID**.  
3 There are also two nonclustered indexes on **ProductID**.  
4 The **UnitPrice** CHECK constraint is defined as (**UnitPrice** >= 0).  
5 The **Quantity** CHECK constraint is defined as (**Quantity** > 0).   
The table-level CHECK constraint is defined as (**Discount** >= 0 and **Discount** < = 1).

#### Creación del trigger:

USE Northwind

GO

CREATE TRIGGER tr\_OrderDetails

ON [Order Details]

AFTER UPDATE

AS

-- Miramos si se han realizado cambios en la PRIMARY KEY

IF UPDATE(OrderID)

BEGIN

PRINT 'Los cambios sobre la PRIMARY KEY nom están permitidos'

ROLLBACK TRAN

END

-- Miramos si se han realizado cambios en las columnas 2, 3 y 4

IF ((COLUMNS\_UPDATED() & (2 + 4 + 8)) > 0)

BEGIN

IF ((COLUMNS\_UPDATED() & 2) = 2)

PRINT 'ProductID actualizado'

IF ((COLUMNS\_UPDATED() & 4) = 4)

PRINT 'UnitPrice actualizado'

IF ((COLUMNS\_UPDATED() & 8) = 8)

PRINT 'Quantity actualizada'

END

GO

#### Testear el trigger y sus consecuencias:

-- Test the trigger.

PRINT CHAR(10) + 'Actualizando ProductID y UnitPrice'

UPDATE [Order Details]

SET ProductID = ProductID

, UnitPrice = UnitPrice

PRINT CHAR(10) + 'Actualizando Quantity only'

UPDATE [Order Details]

SET Quantity = Quantity

PRINT CHAR(10) + 'Actualizando OrderID'

UPDATE [Order Details]

SET OrderID = OrderID

#### Resultado:

Actualizando ProductID y UnitPrice

ProductID actualizado

UnitPrice actualizado

(9 filas afectadas)

(9 filas afectadas)

(2025 filas afectadas)

Actualizando Quantity

Quantity actualizada

(9 filas afectadas)

(9 filas afectadas)

#### Deshacer el ejemplo:

--deshacer ejemplo

UPDATE Person.Address

SET PostalCode = 19107

WHERE PostalCode = '99999';

USE AdventureWorks;

GO

IF EXISTS (SELECT name FROM sys.objects

WHERE name = 'reminder' AND type = 'TR')

DROP TRIGGER Person.reminder;

GO

# Ejemplo2 (Mod\_columnas2.sql)

El ejemplo siguiente crea un desencadenador que imprime un mensaje si alguien intenta actualizar (UPDATE) las columnas StateProvinceID o PostalCode de la tabla Address. Levantará una excepción. RAISERROR.

Razón por la que crear el trigger:

No se quiere evitar el cumplimiento de la instrucción si no que se quiere enviar un mensaje.

Descripción de la tabla Address:

Contiene información sobre las direcciones de todos los clientes, proveedores y empleados de Adventure Works Cycles. Los clientes y los proveedores pueden tener más de una dirección. Por ejemplo, un cliente puede tener una dirección para facturación y otra dirección para envíos.

La tabla Address se encuentra en el esquema Person.

|  |  |  |  |
| --- | --- | --- | --- |
| Columna | Tipo de datos | Nulabilidad | Descripción |
| AddressID | int | No NULL | Clave principal para las filas de dirección. |
| AddressLine1 | nvarchar(60) | No NULL | Primera línea de la dirección postal. |
| AddressLine2 | nvarchar(60) | NULL | Segunda línea de la dirección postal. |
| City | nvarchar(30) | No NULL | Nombre de la ciudad. |
| StateProvinceID | int | No NULL | Número de identificación único para el estado o la provincia. Clave externa para StateProvince.StateProvinceID. |
| PostalCode | nvarchar(15) | No NULL | Código postal para la dirección postal. |
| rowguid | uniqueidentifier ROWGUIDCOL | No NULL | Número de ROWGUIDCOL que identifica la fila de forma exclusiva. Se utiliza para admitir un ejemplo de replicación de mezcla. |
| ModifiedDate | datetime | No NULL | Fecha y hora de la última actualización de la fila. |

#### Creación del trigger:

--Creación del mensaje de error

USE master

EXEC sp\_addmessage

@msgnum = 50009,

@severity = 16,

@msgtext = N'Se han producido los cambios satisfactoriamente, utilizando Triggers.',

@lang = 'us\_english';

GO

--Ejemplo 2

USE AdventureWorks;

GO

IF EXISTS (SELECT name FROM sys.objects

WHERE name = 'reminder' AND type = 'TR')

DROP TRIGGER Person.reminder;

GO

CREATE TRIGGER reminder

ON Person.Address

AFTER UPDATE

AS

IF ( UPDATE (StateProvinceID) OR UPDATE (PostalCode) )

BEGIN

RAISERROR (50009, 16, 10)

END;

GO

#### Testear el trigger y sus consecuencias:

-- Test the trigger.

select \* FROM Person.Address

WHERE PostalCode = '19107';

UPDATE Person.Address

SET PostalCode = 99999

WHERE PostalCode = '19107';

select \* FROM Person.Address

where PostalCode = '19107' or PostalCode = '99999'

GO

#### Deshacer el ejemplo:

--deshacer ejemplo

UPDATE Person.Address

SET PostalCode = 19107

WHERE PostalCode = '99999';

USE AdventureWorks;

GO

IF EXISTS (SELECT name FROM sys.objects

WHERE name = 'reminder' AND type = 'TR')

DROP TRIGGER Person.reminder;

GO

USE master

EXEC sp\_dropmessage

@msgnum = 50009,

@lang = 'us\_english';

Go

# Más información

|  |  |
| --- | --- |
| C/ Miracruz, 10 (Bº de Gros) 20001 Donostia  Telf.: 943 275819  email: [seim@centroseim.com](mailto:seim@centroseim.com) |  |

Logotipo SEIM