
	GUÍA DE TRABAJO N° 10 - SQL SERVER		
	Educación Media Fortalecida SED/SENA	Programación de Software GRADO 11	
	Ing. Néstor Raúl Suarez Perpiñan Página 1 de 6		

TEMA: VISTAS Y DESENCADENADORES (TRIGGER) EN SQL SERVER

OBJETIVO:

- ✓ Adquirir los conocimientos necesarios para desarrollar e implementar vistas y desencadenadores utilizando SQL Server

I. VISTAS EN BASES DE DATOS

Una vista es términos generales es una forma alternativa de recopilar y mostrar datos procedentes de varias tablas; una vista es similar a una tabla virtual dentro de la base de datos que almacena el resultado de una consulta. Una vista almacena la consulta como un objeto de la base de datos que puede utilizarse posteriormente de forma similar a una tabla (aunque no lo es). Las tablas consultadas en una vista se llaman tablas base; en la creación de vistas se puede plantear cualquier tipo de consulta (preferiblemente multitabla).

Una vista suele llamarse también tabla virtual porque los resultados que retorna y la manera de referenciarlas es prácticamente igual que para una tabla. Se pueden crear vistas con un subconjunto de registros y campos de una tabla; una unión de varias tablas; una combinación de varias tablas; un resumen estadístico de una tabla; un subconjunto de otra vista, combinación de vistas, tablas y otras alternativas similares



La sintaxis básica para crear una vista en Sql Server es:

```
CREATE VIEW NOMBREVISTA AS
SENTENCIAS SELECT ...
```

El contenido de una vista una vez ha sido creada se muestra con un "SELECT" de forma similar a como se consultaría una tabla:

```
SELECT * FROM NOMBREVISTA;
```

Las vistas en una base de datos ofrecen múltiples ventajas entre las cuales se destacan:

	GUÍA DE TRABAJO N° 10 - SQL SERVER		
	Educación Media Fortalecida SED/SENA	Programación de Software GRADO 11	
	Ing. Néstor Raúl Suarez Perpiñan Página 2 de 6		

- ✓ Permiten ocultar información: permitiendo el acceso a algunos datos y manteniendo oculto el resto de la información que no se incluye en la vista. El usuario opera con los datos de una vista como si se tratara de una tabla.
- ✓ Permiten simplificar la administración de los permisos de usuario: se pueden dar al usuario permisos para que solamente pueda acceder a los datos a través de vistas, en lugar de concederle permisos para acceder a ciertos campos, así se protegen las tablas base de cambios en su estructura.
- ✓ Permiten mejorar el rendimiento: se puede evitar escribir instrucciones repetidamente almacenando en una vista el resultado de una consulta compleja que incluya información de varias tablas.

DESENCADENADORES (TRIGGER) EN BASES DE DATOS

Un "trigger" (disparador o desencadenador) es un tipo de procedimiento almacenado que se ejecuta cuando se intenta modificar los datos de una tabla. Un Trigger se define para una tabla (o vista) específica, si se intenta modificar (agregar, actualizar o eliminar) datos de una tabla en la que se definió un disparador para alguna de estas acciones (inserción, actualización y eliminación), el disparador se ejecuta (se dispara) en forma automática. En ese orden de ideas un trigger se asocia a un evento (inserción, actualización o borrado) sobre una tabla.

Las principales diferencias entre triggers y procedimientos almacenados es que los triggers no pueden ser invocados directamente; únicamente al intentar modificar los datos de una tabla para la que se ha definido un disparador, es el disparador se ejecuta automáticamente, además de que los triggers no reciben ni retornan parámetros.

Los disparadores pueden hacer referencia a campos de otras tablas. Por ejemplo, puede crearse un trigger de inserción en la tabla "ventas" que compruebe el campo "inventario" de un artículo en la tabla "articulos"; el disparador controlaría que, cuando el valor de "inventario" sea menor a la cantidad que se intenta vender, la inserción del nuevo registro en "ventas" no se realice.

Los disparadores se ejecutan DESPUES de la ejecución de una instrucción "insert", "update" o "delete" en la tabla en la que fueron definidos. Las restricciones se comprueban ANTES de la

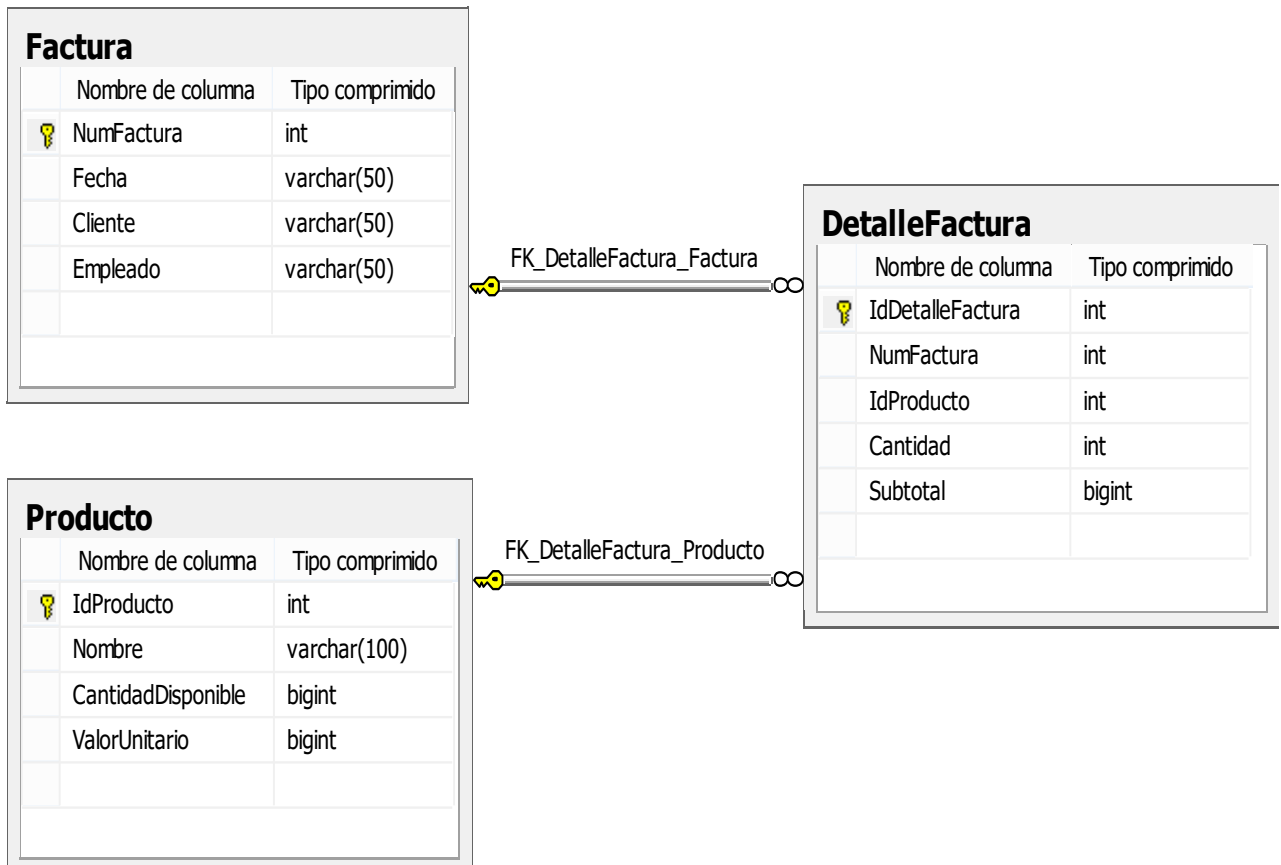
ejecución de una instrucción "insert", "update" o "delete". Por lo tanto, las restricciones se comprueban primero, si se infringe alguna restricción, el desencadenador no llega a ejecutarse.



Los triggers se crean usando la instrucción "CREATE TRIGGER". Esta instrucción especifica la tabla en la que se define el disparador, los eventos para los que se ejecuta y las instrucciones que contiene. La sintaxis básica para crear un trigger en sql server es:

```
CREATE TRIGGER NOMBREDISPARADOR
on NOMBRETABLA
for EVENTO- insert, update o delete
AS
SENTENCIAS A EJECUTAR
```

EJEMPLO – VISTAS Y TRIGGERS EN SQL SERVER

PASO 1: Cree en SQL SERVER una nueva base de datos llamada “BDDEjemploFactura” con la distribución de tablas y campos que se muestran a continuación:



	GUÍA DE TRABAJO N° 10 - SQL SERVER		
	Educación Media Fortalecida SED/SENA	Programación de Software GRADO 11	
	Ing. Néstor Raúl Suarez Perpiñan Página 4 de 6		

PASO 2: Inserte 5 productos diferentes (Leche, Huevo, Pan, Verdura, Fruta) en la tabla “Productos”

PASO 3: Inserte 2 registros en la tabla “Factura”, numérelas como NumFactura = 1 y NumFactura=2

PASO 4: Inserte en la Tabla “DetalleFatura” registros donde asocie leche, huevo y pan a la factura número 1 y verdura y fruta a la factura número 2, coloque el dato subtotal de acuerdo a la cantidad y valor unitario de cada producto

PASO 6: Cree una nueva Vista (Click Derecho sobre la carpeta “Vistas” y luego seleccionar la opción llamada “Nueva Vista”) y utilizando el asistente de creación de vistas genere un código similar al que se muestra a continuación:

```
CREATE VIEW Vista_Detallefactura
AS
SELECT
Factura.NumFactura,
DetalleFactura.IdProducto,
Producto.Nombre,
Producto.ValorUnitario,
DetalleFactura.Cantidad,
DetalleFactura.Subtotal
FROM DetalleFactura
INNER JOIN Factura ON DetalleFactura.NumFactura = Factura.NumFactura
INNER JOIN Producto ON DetalleFactura.IdProducto =Producto.IdProducto
```

Nota: Este código permite crear una nueva vista llamada “Vista_Detallefactura”

PASO 7: Verifique que en la carpeta “Vistas” de la base de datos aparezca disponible la vista “Vista_Detallefactura” .

A continuación visualice que el contenido de la vista corresponda con los datos ya almacenados. Use una consulta básica tipo:



```
“SELECT * FROM Vista_Detallefactura”
```

PASO 8: Visualice únicamente el detalle de factura de la factura número 1, use una consulta básica tipo:

```
“SELECT * FROM Vista_Detallefactura WHERE NumFactura = 1”
```

PASO 9: Calcule el total a pagar de la factura número 1, use la consulta sql que se muestra a continuación:

```
SELECT SUM(Subtotal) AS Total FROM Vista_DetalleFactura
WHERE NumFactura = 1
```

	GUÍA DE TRABAJO N° 10 - SQL SERVER		
	Educación Media Fortalecida SED/SENA	Programación de Software GRADO 11	
	Ing. Néstor Raúl Suarez Perpiñan Página 5 de 6		

PASO 10: Crear un desencadenador(trigger) que permita actualizar automáticamente la Cantidad Disponible en la tabla Producto cada vez que se inserte o actualice un registro en la tabla DetalleFactura.

Para crear el trigger solicitado abra una ventana de nueva consulta y digite el código que se muestra a continuación:

```
CREATE TRIGGER ActualizarInventario
ON DetalleFactura
AFTER INSERT, UPDATE
AS

DECLARE
@idproducto int,
@CantidadVendida int,
@CantidadDisponible int,
@NuevaCantidad int

SET @idproducto= (Select idproducto From inserted);
SET @CantidadVendida= (Select Cantidad From inserted);
SET @CantidadDisponible = (Select CantidadDisponible From Producto where
idproducto = @idproducto);

SET @NuevaCantidad = @CantidadDisponible - @CantidadVendida
Update Producto Set CantidadDisponible = @NuevaCantidad where idproducto
= @idproducto
```

NOTA: Para verificar que este desencadenador(Trigger) funciona consulte la tabla Productos (Select * From Productos), luego inserte nuevos registros en la tabla DetalleFactura y finalmente vuelva a consultar nuevamente la tabla Productos en donde debe ver que la cantidad disponible a cambiado de forma automática.

NOTA: Los desencadenadores (Trigger) aparecen disponibles para su verificación en la carpeta llamada “Desencadenadores” de cada una de las tablas.

TALLER

1. Sobre la misma base de datos del ejemplo de esta guía cree una factura numero 3 donde el detalle de factura contenga todos los 5 productos ya creados
2. Cree una nueva vista de nombre “Vista_DetalleFactura2” cuyo resultado debe visualizar el contenido tal cual como se muestra a continuación:

NumFactura	Nombre	ValorUnitario	Cantidad	Subtotal
1	Leche	2000	6	10000
1	Huevo	1000	2	2000
1	Pan	200	5	10000
2	Verdura	100	2	20000
2	Fruta	1500	3	30000
3	Leche	2000	2	4000
3	Huevo	1000	3	3000
3	Pan	200	4	800
3	Verdura	100	5	500
3	Fruta	1500	6	9000
<i>NULL</i>	<i>NULL</i>	<i>NULL</i>	<i>NULL</i>	<i>NULL</i>

3. Realice una consulta sobre la “Vista_DetalleFactura2” de únicamente la factura número 3, la consulta resultante debe salir tal cual como aparece a continuación:

Nombre	ValorUnitario	Cantidad	Subtotal
Leche	2000	2	4000
Huevo	1000	3	3000
Pan	200	4	800
Verdura	100	5	500
Fruta	1500	6	9000

4. Calcule el total a pagar en esta factura número 3 usando código SQL
5. Verifique que el desencadenador(Trigger) “ActualizarInventario” sigue aún funcionando