Material Imprimible

Curso de SQL Server

Módulo 8: Disparadores (Triggers)

TRIGGER

Un trigger es una clase especial de procedimiento almacenado que se ejecuta automáticamente cuando se produce un evento en el servidor de bases de datos. Los trigger se ejecutan cuando un usuario intenta modificar datos mediante un evento de lenguaje de manipulación de datos (DML). Los eventos DML son instrucciones **INSERT**, **UPDATE** o **DELETE** de una tabla o vista. Estos triggers se activan cuando se desencadena cualquier evento válido, con independencia de que las filas de la tabla se vean o no afectadas.
Sintaxis básica:

```
create trigger NOMBREDISPARADOR
 on NOMBRETABLA
 for EVENTO- insert, update o delete
as
 SENTENCIAS
```

Analizamos la sintaxis:
- "create trigger" junto al nombre del disparador.
- "on" seguido del nombre de la tabla o vista para la cual se establece el trigger.
- luego de "for", se indica la acción (evento, el tipo de modificación) sobre la tabla o vista que activará el trigger. Puede ser "insert", "update" o "delete". Debe colocarse al menos UNA acción, si se coloca más de una, deben separarse con comas.
- luego de "as" viene el cuerpo del trigger, se especifican las condiciones y acciones del disparador; es decir, las condiciones que determinan cuando un intento de inserción, actualización o borrado provoca las acciones que el trigger realizará.

Consideraciones generales:
- "create trigger" debe ser la primera sentencia de un bloque y sólo se puede aplicar a una tabla.
- un disparador se crea solamente en la base de datos actual, pero puede hacer referencia a objetos de otra base de datos.
- Las siguientes instrucciones no están permitidas en un desencadenador: create database, alter database, drop database, load database, restore database, load log, reconfigure, restore log, disk init, disk resize.
- Se pueden crear varios triggers para cada evento, es decir, para cada tipo de modificación (inserción, actualización o borrado) para una misma tabla. Por ejemplo, se puede crear un "insert trigger" para una tabla que ya tiene otro "insert trigger".

TRIGGER INSERT

Podemos crear un disparador para que se ejecute siempre que una instrucción "insert" ingrese datos en una tabla.

Sintaxis básica:

```
create trigger NOMBREDISPARADOR
 on NOMBRETABLA
 for insert
as
  SENTENCIAS
```

Analizamos la sintaxis:
"create trigger" junto al nombre del disparador; "on" seguido del nombre de la tabla para la cual se establece el trigger.
Luego de "for" se coloca el evento (en este caso "insert"), lo que indica que las inserciones sobre la tabla activarán el trigger.
Luego de "as" se especifican las condiciones y acciones, es decir, las condiciones que determinan cuando un intento de inserción provoca las acciones que el trigger realizará.
Creamos un trigger sobre la tabla "ventas" para el evento de inserción. Cada vez que se realiza un "insert" sobre "ventas", el disparador se ejecuta. El disparador controla que la cantidad que se intenta vender sea menor o igual al stock del libro y actualiza el campo "stock" de "libros", restando al valor anterior la cantidad vendida:

```
create trigger DIS_ventas_insertar
 on ventas
 for insert
 as
  declare @stock int
  select @stock= stock from libros
                    join inserted
                     on inserted.codigolibro=libros.codigo
                     where libros.codigo=inserted.codigolibro
 if (@stock>=(select cantidad from inserted))
   update libros set stock=stock-inserted.cantidad
    from libros
    join inserted
    on inserted.codigolibro=libros.codigo
    where codigo=inserted.codigolibro
 else
 begin
   raiserror ('Hay menos libros en stock de los solicitados para la venta', 16, 1)
   rollback transaction
 end
```

Entonces, creamos el disparador ("create trigger") dándole un nombre ("DIS_ventas_insertar") sobre ("on") una tabla específica ("ventas") para ("for") el suceso de inserción ("insert"). Luego de "as" colocamos las sentencias, las acciones que el trigger realizará cuando se ingrese un registro en "ventas" (en este caso, controlar que haya stock y disminuir el stock de "libros").
Cuando se activa un disparador "insert", los registros se agregan a la tabla del disparador y a una tabla denominada "inserted". La tabla "inserted" es una tabla virtual que contiene una copia de los registros insertados; tiene una estructura similar a la tabla en que se define el disparador, es decir, la tabla en que se intenta la acción. La tabla "inserted" guarda los valores nuevos de los registros.

TRIGGER UPDATE

Podemos crear un disparador para que se ejecute siempre que una instrucción "update" actualice los datos de una tabla.
Sintaxis básica:

```
create trigger NOMBREDISPARADOR
 on NOMBRETABLA
 for update
as
  SENTENCIAS
```

Analizamos la sintaxis:
"create trigger" junto al nombre del disparador; "on" seguido del nombre de la tabla para la cual se establece el trigger.
Luego de "for" se coloca el evento (en este caso "update"), lo que indica que las actualizaciones sobre la tabla activarán el trigger.
Luego de "as" se especifican las condiciones y acciones, es decir, las condiciones que determinan cuando un intento de modificación provoca las acciones que el trigger realizará.
El siguiente disparador de actualización se crea para evitar que se modifiquen los datos de la tabla "libros":

```
create trigger DIS_libros_actualizar
 on libros
 for update
 as
   raiserror('Los datos de la tabla "libros" no pueden modificarse', 10, 1)
   rollback transaction
```

Entonces, creamos el disparador ("create trigger") dándole un nombre ("DIS_libros_actualizar") sobre una tabla específica ("libros") para ("for") el suceso de actualización ("update"). Luego de "as" colocamos las sentencias, las acciones que el trigger realizará cuando se intente actualizar uno o varios registros en "libros" (en este caso, impedir las modificaciones).

Cuando se ejecuta una instrucción "update" en una tabla que tiene definido un disparador, los registros originales (antes de ser actualizados) se mueven a la tabla virtual "deleted" y los registros actualizados (con los nuevos valores) se copian a la tabla virtual "inserted". Dentro del trigger se puede acceder a estas tablas.
En el cuerpo de un trigger se puede emplear la función "update(campo)" que recibe un campo y retorna verdadero si el evento involucra actualizaciones (o inserciones) en ese campo; en caso contrario retorna "false".

TRIGGER DELETE

Podemos crear un disparador para que se ejecute siempre que una instrucción "delete" elimine datos en una tabla.
Sintaxis básica:

```
create trigger NOMBREDISPARADOR
 on NOMBRETABLA
 for delete
as
  SENTENCIAS
```

Analizamos la sintaxis:
"create trigger" junto al nombre del disparador; "on" seguido del nombre de la tabla para la cual se establece el trigger.
Luego de "for" se coloca el evento (en este caso "delete"), lo que indica que las eliminaciones sobre la tabla activarán el trigger.
Luego de "as" se especifican las condiciones que determinan cuando un intento de eliminación causa las acciones que el trigger realizará.
El disparador del siguiente ejemplo se crea para la tabla "ventas", para que cada vez que se elimine un registro de "ventas", se actualice el campo "stock" de la tabla "libros" (por ejemplo, si el comprador devuelve los libros comprados):

```
create trigger DIS_ventas_borrar
 on ventas
 for delete
as
  update libros set stock= libros.stock+deleted.cantidad
```

```
from libros
join deleted
on deleted.codigolibro=libros.codigo;
```

Entonces, creamos el disparador ("create trigger") dándole un nombre ("DIS_ventas_borrar") sobre ("on") una tabla específica ("ventas") para ("for") el evento de borrado ("delete"). Luego de "as" colocamos las sentencias, las acciones que el trigger realizará cuando se elimine un registro en "ventas" (en este caso, aumentar el stock de "libros").
Cuando se activa un disparador "delete", los registros eliminados en la tabla del disparador se agregan a una tabla llamada "deleted". La tabla "deleted" es una tabla virtual que conserva una copia de los registros eliminados; tiene una estructura similar a la tabla en que se define el disparador, es decir, la tabla en que se intenta la acción.

HABILITAR Y DESHABILITAR

Se puede deshabilitar o habilitar un disparador específico de una tabla o vista, o todos los disparadores que tenga definidos.
Si se deshabilita un disparador, éste sigue existiendo, pero al ejecutar una instrucción "insert", "update" o "delete" en la tabla, no se activa.
Sintaxis para deshabilitar o habilitar un disparador:

```
alter table NOMBRETABLA
ENABLE | DISABLE trigger NOMBREDISPARADOR;
```

El siguiente ejemplo deshabilita un trigger:

```
alter table empleados
disable trigger dis_empleados_borrar;
```

Se pueden deshabilitar (o habilitar) varios disparadores en una sola sentencia, separando sus nombres con comas. El siguiente ejemplo deshabilitamos dos triggers definidos sobre la tabla empleados:

```
alter table empleados
disable trigger dis_empleados_actualizar, dis_empleados_insertar;
```

Sintaxis para habilitar (o deshabilitar) todos los disparadores de una tabla específica:

```
alter table NOMBRETABLA
```

ENABLE | DISABLE TRIGGER all;

La siguiente sentencia habilita todos los triggers de la tabla "empleados":

alter table empleados

enable trigger all;

PUBLICAR UNA BASE DE DATOS (SQL SERVER MANAGEMENT STUDIO)

Puede utilizar el **Asistente** para **generar y publicar scripts** para publicar una base de datos completa u objetos de base de datos individuales en un proveedor de alojamiento web.

El Asistente para generar y publicar scripts se puede utilizar para publicar una base de datos u objetos de base de datos seleccionados en un proveedor de alojamiento web. Un proveedor de alojamiento web de SQL Server es una interfaz de conectividad a un servicio web. El servicio web se crea mediante el proyecto de servicios de publicación de bases de datos del kit de herramientas de alojamiento de SQL Server en CodePlex. El servicio web facilita a los clientes de alojamiento web publicar sus bases de datos en el servicio mediante el Asistente para generar y publicar scripts. Para obtener más información sobre cómo descargar el kit de herramientas de hospedaje de SQL Server, consulte Servicios de publicación de bases de datos de SQL Server.

El Asistente para generar y publicar scripts también se puede utilizar para crear un script para transferir una base de datos.

Para publicar una base de datos en un servicio web

1. En el Explorador de objetos, expanda **Bases de datos**, haga clic con el botón derecho en una base de datos, seleccione **Tareas** y luego haga clic en **Generar y publicar scripts**. Siga los pasos del asistente para crear un script de los objetos de la base de datos para su publicación.
2. En la página **Elegir objetos**, seleccione los objetos que se publicarán en el servicio de alojamiento web.
3. En la página **Establecer opciones de secuencias de comandos**, seleccione **Publicar en servicio web**.
    1. En el cuadro **Proveedor**, especifique el proveedor de su servicio web. Si no ha configurado un proveedor de alojamiento web, seleccione **Administrar proveedores** y use el cuadro de diálogo **Administrar proveedores** para configurar un proveedor para su servicio web.

2. Para especificar opciones de publicación avanzadas, seleccione el botón **Avanzado** en la sección **Publicar en servicio web**.

4. En la página **Resumen**, revise sus selecciones. Haga clic en **Anterior** para cambiar sus selecciones. Haga clic en **Siguiente** para publicar los objetos que seleccionó.
5. En la página **Guardar o publicar secuencias de comandos**, supervise el progreso de la publicación.

Fuente: https://www.estradawebgroup.com/Post/Que-es-un-trigger-o-desencadenador-y-como-crearlo-/.
https://docs.microsoft.com/es-es/sql/relational-databases/databases/publish-a-database-sql-server-management-studio?view=sql-server-ver15
https://www.tutorialesprogramacionya.com/sqlserverya/temarios/descripcion.php?inicio=125&cod=150&punto=144