```
-- SQL Server Syntax
Trigger on an INSERT, UPDATE, or DELETE statement to a table or view (DML
Trigger)

CREATE TRIGGER [ schema_name . ]trigger_name
ON { table | view }
[ WITH <dml_trigger_option> [ ,...n ] ]
{ FOR | AFTER | INSTEAD OF }
{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }
[ WITH APPEND ]
[ NOT FOR REPLICATION ]
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME <method specifier [ ;
] > }

<dml_trigger_option> ::=
    [ ENCRYPTION ]
    [ EXECUTE AS Clause ]

<method_specifier> ::=
    assembly_name.class_name.method_name



Trigger on a CREATE, ALTER, DROP, GRANT, DENY, REVOKE, or UPDATE
STATISTICS statement (DDL Trigger)
CREATE TRIGGER trigger_name
ON { ALL SERVER | DATABASE }
[ WITH <ddl_trigger_option> [ ,...n ] ]
{ FOR | AFTER } { event_type | event_group } [ ,...n ]
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier >
[ ; ] }

<ddl_trigger_option> ::=
    [ ENCRYPTION ]
    [ EXECUTE AS Clause ]
```

Trigger on a LOGON event (Logon Trigger)

```
CREATE TRIGGER trigger_name
ON ALL SERVER
[ WITH <logon_trigger_option> [ ,...n ] ]
{ FOR| AFTER } LOGON
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier >
[ ; ] }

<logon_trigger_option> ::=
    [ ENCRYPTION ]
    [ EXECUTE AS Clause ]
```

A.Usar desencadenador DML con un mensaje de aviso

El siguiente desencadenador DML imprime un mensaje en el cliente cuando alguien intenta agregar o cambiar datos en la tabla `Customer` de la base de datos AdventureWorks2012.

```
IF OBJECT_ID ('Sales.reminder1', 'TR') IS NOT NULL
   DROP TRIGGER Sales.reminder1;
GO
CREATE TRIGGER reminder1
ON Sales.Customer
AFTER INSERT, UPDATE
AS RAISERROR ('Notify Customer Relations', 16, 10);
GO
```

B.Usar un desencadenador DML con un mensaje de correo electrónico de aviso

Este ejemplo envía un mensaje de correo electrónico a una persona especificada (`MaryM`) cuando cambia la tabla `Customer`.

```
IF OBJECT_ID ('Sales.reminder2','TR') IS NOT NULL
    DROP TRIGGER Sales.reminder2;
GO
CREATE TRIGGER reminder2
ON Sales.Customer
AFTER INSERT, UPDATE, DELETE
AS
   EXEC msdb.dbo.sp_send_dbmail
        @profile_name = 'AdventureWorks2012 Administrator',
        @recipients = 'danw@Adventure-Works.com',
        @body = 'Don''t forget to print a report for the sales force.',
        @subject = 'Reminder';
GO
```

C.Usar un desencadenador DML AFTER para exigir una regla de negocios entre las tablas PurchaseOrderHeader y Vendor

Debido a que las restricciones CHECK solo pueden hacer referencia a las columnas en las que se han definido las restricciones de columna o de tabla, cualquier restricción entre tablas, en este caso, reglas de negocios, debe definirse como desencadenadores.

En el ejemplo siguiente se crea un desencadenador DML en la base de datos AdventureWorks2012. El desencadenador comprueba que la solvencia del proveedor es satisfactoria cuando se intenta insertar un nuevo pedido de compra en la tabla `PurchaseOrderHeader`. Para obtener la solvencia del proveedor, debe hacerse referencia a la tabla `Vendor`. Si la solvencia no es satisfactoria, se obtiene un mensaje y no se ejecuta la inserción.

```
IF OBJECT_ID ('Purchasing.LowCredit','TR') IS NOT NULL
   DROP TRIGGER Purchasing.LowCredit;
GO
-- This trigger prevents a row from being inserted in the
Purchasing.PurchaseOrderHeader table
-- when the credit rating of the specified vendor is set to 5 (below
average).
```

```
CREATE TRIGGER Purchasing.LowCredit ON Purchasing.PurchaseOrderHeader
AFTER INSERT
AS
IF EXISTS (SELECT *
           FROM Purchasing.PurchaseOrderHeader AS p
           JOIN inserted AS i
           ON p.PurchaseOrderID = i.PurchaseOrderID
           JOIN Purchasing.Vendor AS v
           ON v.BusinessEntityID = p.VendorID
           WHERE v.CreditRating = 5
          )
BEGIN
RAISERROR ('A vendor''s credit rating is too low to accept new
purchase orders.', 16, 1);
ROLLBACK TRANSACTION;
RETURN
END;
GO

-- This statement attempts to insert a row into the PurchaseOrderHeader
table
-- for a vendor that has a below average credit rating.
-- The AFTER INSERT trigger is fired and the INSERT transaction is rolled
back.

INSERT INTO Purchasing.PurchaseOrderHeader (RevisionNumber, Status,
EmployeeID,
VendorID, ShipMethodID, OrderDate, ShipDate, SubTotal, TaxAmt, Freight)
VALUES (
2
,3
,261
,1652
,4
,GETDATE()
,GETDATE()
,44594.55
,3567.564
,1114.8638 );
GO
```

D.Usar un desencadenador DDL con ámbito de base de datos

En el ejemplo siguiente se usa un desencadenador DDL para impedir que se quiten sinónimos de una base de datos.

```
IF EXISTS (SELECT * FROM sys.triggers
    WHERE parent_class = 0 AND name = 'safety')
    DROP TRIGGER safety
         ON DATABASE;
```

```
GO
CREATE TRIGGER safety
ON DATABASE
FOR DROP_SYNONYM
AS
   RAISERROR ('You must disable Trigger "safety" to drop synonyms!',10,
1)
   ROLLBACK
GO
DROP TRIGGER safety
ON DATABASE;
GO
```

E.Usar un desencadenador DDL con ámbito de servidor

En el ejemplo siguiente se utiliza un desencadenador DDL para imprimir un mensaje si se produce un evento CREATE DATABASE en la instancia actual del servidor, y se utiliza la función `EVENTDATA` para recuperar el texto de la instrucción Transact-SQL correspondiente. Para obtener más ejemplos que usan EVENTDATA en desencadenadores DDL, vea Usar la función EVENTDATA.

> **Se aplica a**: SQL Server 2008 a SQL Server 2014.

```
IF EXISTS (SELECT * FROM sys.server_triggers
    WHERE name = 'ddl_trig_database')
DROP TRIGGER ddl_trig_database
ON ALL SERVER;
GO
CREATE TRIGGER ddl_trig_database
ON ALL SERVER
FOR CREATE_DATABASE
AS
    PRINT 'Database Created.'
    SELECT
EVENTDATA().value('(/EVENT_INSTANCE/TSQLCommand/CommandText)[1]','nvarcha
r(max)')
GO
DROP TRIGGER ddl_trig_database
ON ALL SERVER;
GO
```

F.Usar un desencadenador LOGON

El ejemplo siguiente de desencadenador logon rechaza un intento de iniciar sesión en SQL Server como miembro del inicio de sesión *login_test* si ya hay tres sesiones de usuario ejecutándose con ese inicio de sesión.

> **Se aplica a**: SQL Server 2008 a SQL Server 2014.

```
USE master;
GO
CREATE LOGIN login_test WITH PASSWORD = '3KHJ6dhx(0xVYsdf' MUST_CHANGE,
```

```
    CHECK_EXPIRATION = ON;
GO
GRANT VIEW SERVER STATE TO login_test;
GO
CREATE TRIGGER connection_limit_trigger
ON ALL SERVER WITH EXECUTE AS 'login_test'
FOR LOGON
AS
BEGIN
IF ORIGINAL_LOGIN()= 'login_test' AND
    (SELECT COUNT(*) FROM sys.dm_exec_sessions
            WHERE is_user_process = 1 AND
                original_login_name = 'login_test') > 3
    ROLLBACK;
END;
```

G.Ver los eventos que hacen que se active un desencadenador

En el ejemplo siguiente se efectúa una consulta en las vistas de catálogo `sys.triggers` y `sys.trigger_events` para determinar qué eventos de lenguaje Transact-SQL hacen que se active el desencadenador `safety`. `safety` se ha creado en el ejemplo anterior.

```
SELECT TE.*
FROM sys.trigger_events AS TE
JOIN sys.triggers AS T ON T.object_id = TE.object_id
WHERE T.parent_class = 0 AND T.name = 'safety';
GO
```

Ejemplos

```
CREATE TRIGGER ActualizaVentasEmpleados
ON pedidos FOR INSERT
AS
UPDATE  empleados SET ventas=ventas+inserted.importe
FROM  empleados, inserted
WHERE  numemp=inserted.rep;
```

```
create trigger DIS_libros_borrar
  on libros
  for delete
  as
   if (select count(*) from deleted) > 1
   begin
    raiserror('No puede borrar más de un libro',16,1)
    rollback transaction
   end;
```

CREATE TRIGGER del_cust
ON customer
AFTER DELETE
AS
BEGIN
 DELETE FROM customer_repl
 WHERE customer_num in (
 SELECT customer_num FROM DELETED)
END

DELETE FROM CUSTOMER where customer_num=130
select * from customer_borrados

	customer_num	fname	lname	company	address1	address2	city	state	zipcode	phone
1	130	Alfred	Grant	Gold Medal Sports	776 Gary Avenue		Menlo Park	CA	99999	415-356-1123

insert into customer select * from customer_borrados

DELETE FROM CUSTOMER where customer_num in (129,130)
select * from customer_borrados

	customer_num	fname	lname	company	address1	address2	city	state	zipcode	phone
1	130	Alfred	Grant	Gold Medal Sports	776 Gary Avenue		Menlo Park	CA	99999	415-356-1123
2	129	Lana	Beatty	Sportstown	654 Oak Grove		Menlo Park	CA	99999	415-356-9982