**Introduction to Triggers in SQL Server**

SQL Server Triggers are used to execute after or before an INSERT, DELETE, or an UPDATE operation on a table. You can use these SQL triggers on Views, or Tables to perform any of the above-specified activities. Remember, you can associate a trigger to a single table only.

There are three main characteristics that make triggers different than stored procedures:

* Triggers cannot be manually executed by the user.
* There is no chance for triggers to receive parameters.
* You cannot commit or rollback a transaction inside a trigger.

**Types of Triggers in SQL Server**

There are two types of Triggers in SQL Server, and they are AFTER TRIGGERS, and INSTEAD OF TRIGGERS.

For the Sql Server triggers demonstration, we are working with the Employee table and Employee audit table

**AFTER TRIGGERS in SQL Server**

The after/for triggers in SQL runs after an [INSERT](https://www.tutorialgateway.org/sql-insert-statement/), [DELETE](https://www.tutorialgateway.org/sql-delete-statement/), or an [UPDATE](https://www.tutorialgateway.org/sql-update-statement/) on a table. It means, Before the trigger starts running, all the operations should execute, and the statement has to succeed in the constraint [check](https://www.tutorialgateway.org/sql-check-constraint/) as well.

After triggers in SQL Server are not supported on [Views](https://www.tutorialgateway.org/views-in-sql-server/) so, use them on [tables](https://www.tutorialgateway.org/sql-create-table/) only. It can further divide into

1. [AFTER INSERT TRIGGERS](https://www.tutorialgateway.org/after-insert-triggers-in-sql-server/): This trigger will fire after the completion of Insert operation on the Employee table. Once it completes inserting into the Employee table, it will start inserting into the audit table. Say, if it fails to insert into the Employee table, then it won’t insert into the Audit table.
2. [AFTER UPDATE TRIGGERS](https://www.tutorialgateway.org/after-update-triggers-in-sql-server/): This SQL Server trigger will fire after the Update operation completed on the Employees table. Once it ends Updating the Employee table, it will start inserting/updating into the audit table. For instance, if it fails to update the Employees table, then it won’t insert into the Audit table.
3. [AFTER DELETE TRIGGERS](https://www.tutorialgateway.org/after-delete-triggers-in-sql-server/): The After delete trigger will fire after the completion of Delete operation on the Employee table. Once it completes deleting the records from the Employees, it will start inserting/deleting from the audit table. Say, if it fails to remove from the Employee table, then it won’t insert into the Audit table

**INSTEAD OF TRIGGERS in SQL Server**

SQL Server Instead of Triggers fired before the execution of an INSERT, DELETE, or an UPDATE on a table start.

It means, Before the trigger starts running, it does not need any condition constraint check. So, this trigger will execute even the constraint check fails. It can further divide into

* [INSTEAD OF INSERT TRIGGERS](https://www.tutorialgateway.org/instead-of-insert-triggers-in-sql-server/): This instead of insert trigger will fire before the Insert operation on Employee table starts. Once it completes inserting into the Employee Audit table, it will begin to insert into the Employee table. And if it fails for some reason, it won’t insert into the Employee table.
* [INSTEAD OF UPDATE TRIGGERS](https://www.tutorialgateway.org/instead-of-update-triggers-in-sql-server/): This update trigger will fire before updating the records in the Employee table. Once it completes the trigger execution, it will start updating the records in the Employee table. And if it fails, it won’t update the table.
* [INSTEAD OF DELETE TRIGGERS](https://www.tutorialgateway.org/instead-of-delete-triggers-in-sql-server/): This delete trigger will fire before deleting records from the Employee table starts. Once the trigger executed successfully, it will begin deleting records from the Employees table. And if it fails, it won’t remove any record from the Employee table.

**Sql Server Triggers Syntax**

The syntax of After Triggers in [SQL Server](https://www.tutorialgateway.org/sql/) is

-- Create Triggers in SQL Server

CREATE [OR ALTER] TRIGGER [Schema\_Name].Trigger\_Name

ON Table

AFTER INSERT | UPDATE | DELETE

AS

BEGIN

-- Trigger Statements

-- Insert, Update, Or Delete Statements

END

* Schema\_name: Please specify the schema name. For example, dbo, or Human Resource, etc.
* Trigger\_Name: You can specify any name you wish to give other than the system reserved keywords. Please try to use meaningful names so that you can identify them easily.

The basic syntax of Instead Of Triggers in SQL Server is as shown below:

-- Create Triggers in SQL Server

CREATE [OR ALTER] TRIGGER [Schema\_Name].Trigger\_Name

ON Table | View

INSTEAD OF INSERT | UPDATE | DELETE

AS

BEGIN

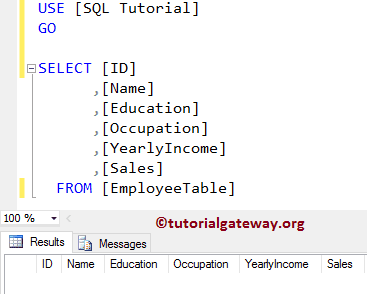
-- Trigger Statements

-- Insert, Update, Or Delete Statements

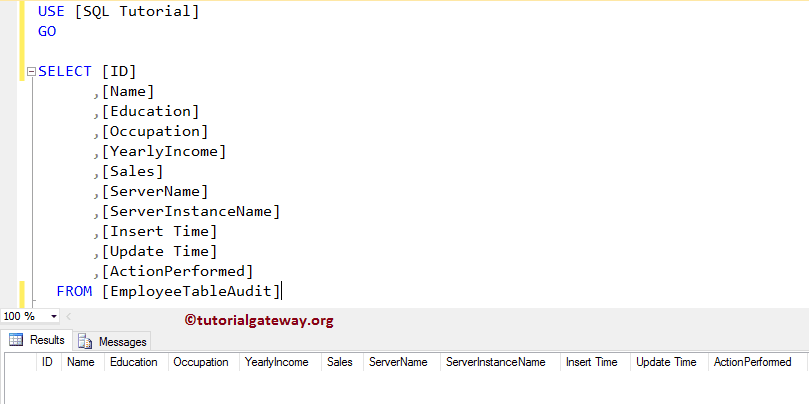
END

Let us see how to Create a Trigger, Modify the existing triggers, and delete triggers in SQL Server with an example.

For this SQL Server triggers demonstration, We are going to use the below-shown tables. As you can see, the table is Empty



and our Audit table is also empty



### Create SQL Triggers using SQL Query

Let us see how to create the Triggers in SQL Server using the SQL Create Trigger Statement.

For this SQL triggers demonstration purpose, we are creating an After Insert Trigger to Insert the records into Employee Table Audit, along with the audit information

-- Triggers in SQL Server Introduction

USE [SQL Tutorial]

GO

CREATE TRIGGER triggers\_in\_sql

ON EmployeeTable

AFTER INSERT

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for trigger here

INSERT INTO [EmployeeTableAudit](

[ID]

,[Name]

,[Education]

,[Occupation]

,[YearlyIncome]

,[Sales]

,[ServerName]

,[ServerInstanceName]

,[Insert Time])

SELECT ID,

Name,

Education,

Occupation,

YearlyIncome,

Sales,

CAST( SERVERPROPERTY('MachineName') AS VARCHAR(50)),

CAST( SERVERPROPERTY('ServerName') AS VARCHAR(50)),

GETDATE()

FROM INSERTED;

PRINT 'We Successfully Fired the AFTER INSERT Triggers in SQL Server.'

END

GO

We already explained the statements inside the SQL triggers in our previous article. I suggest you refer [AFTER INSERT Triggers](https://www.tutorialgateway.org/after-insert-triggers-in-sql-server/) article to under this code.

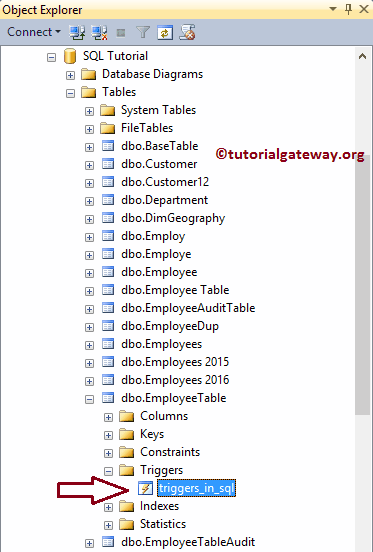
Run the above Create trigger query

Messages

--------

Command(s) completed successfully.

Let me show you the newly created trigger by opening the Object Explorer -> Go to the SQL Tutorial Database -> Go and expand the Employee Table -> and then expand the Triggers Folder



Let me insert 5 random records into the Employee table for the demonstration purpose. It will help us to check whether the After insert Trigger is triggered or not.

USE [SQL Tutorial]

GO

INSERT INTO [EmployeeTable] (

[Name]

,[Education]

,[Occupation]

,[YearlyIncome]

,[Sales]

)

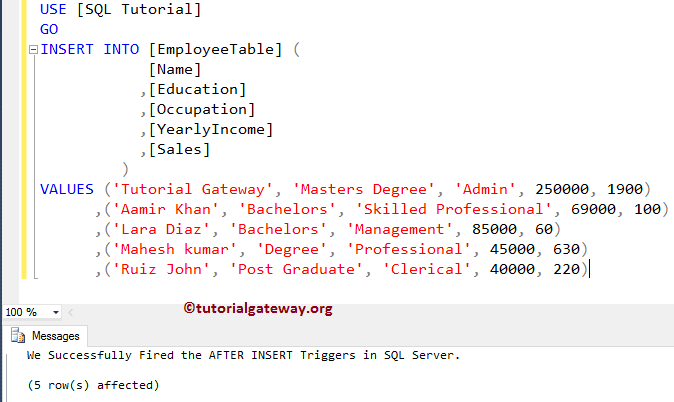
VALUES ('Tutorial Gateway', 'Masters Degree', 'Admin', 250000, 1900)

,('Aamir Khan', 'Bachelors', 'Skilled Professional', 69000, 100)

,('Lara Diaz', 'Bachelors', 'Management', 85000, 60)

,('Mahesh kumar', 'Degree', 'Professional', 45000, 630)

,('Ruiz John', 'Post Graduate', 'Clerical', 40000, 220)



From the above screenshot, you can see that our trigger is triggered. And also, it is printing the message that we placed in our trigger. Please use the following Query to check the inserted records in the Employee table

USE [SQL Tutorial]

GO

SELECT [ID]

,[Name]

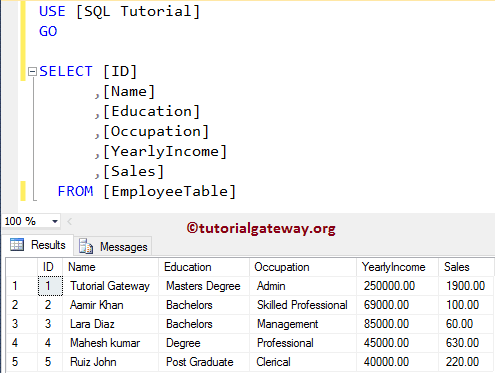
,[Education]

,[Occupation]

,[YearlyIncome]

,[Sales]

FROM [EmployeeTable]



Next, check the records in the Audit table using the following query.

USE [SQL Tutorial]

GO

SELECT [ID]

,[Name]

,[Education]

,[Occupation]

,[YearlyIncome]

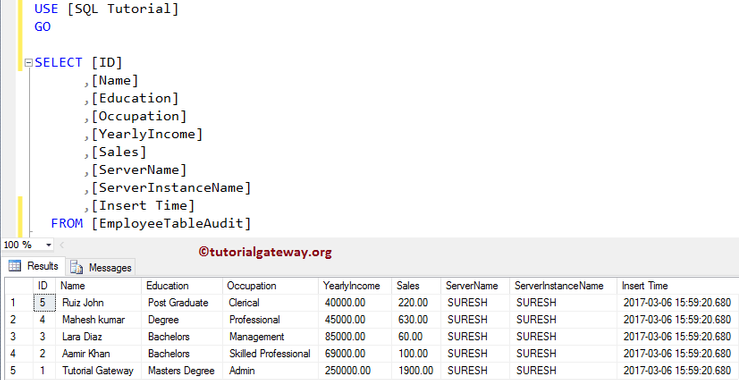
,[Sales]

,[ServerName]

,[ServerInstanceName]

,[Insert Time]

FROM [EmployeeTableAudit]

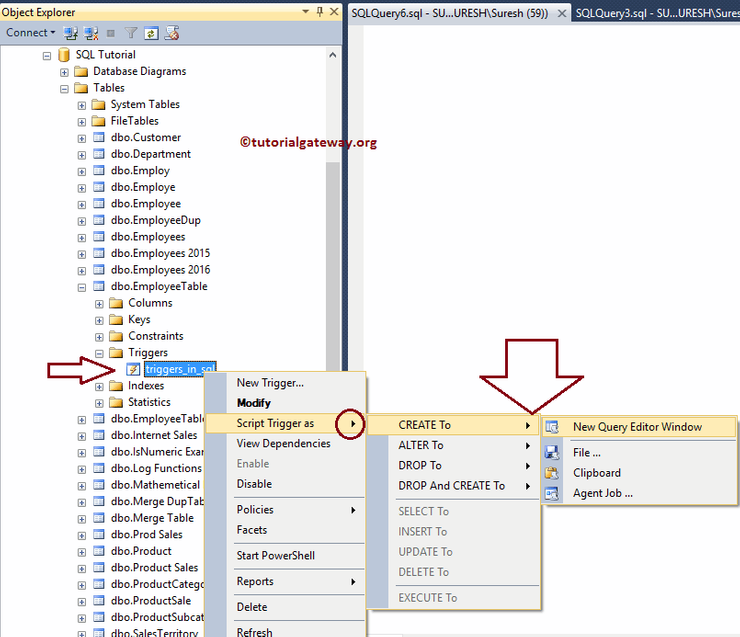


## Modify Triggers in SQL Server Example

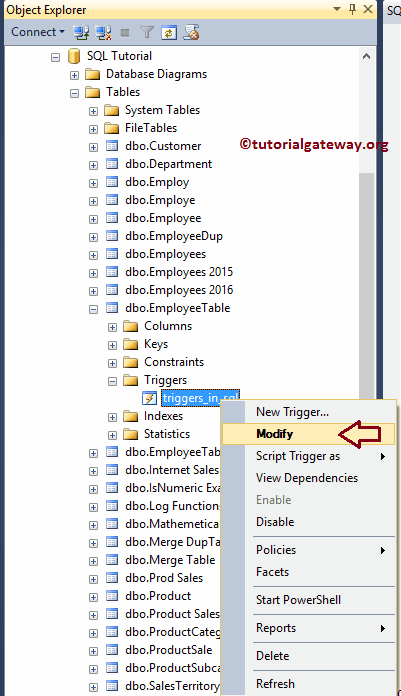
The following examples will help you understand the steps involved in modifying the existing SQL Triggers using the [SQL Management Studio](https://www.tutorialgateway.org/sql-server-management-studio/), and Transact-SQL Query.

### Modify SQL Triggers example using SSMS

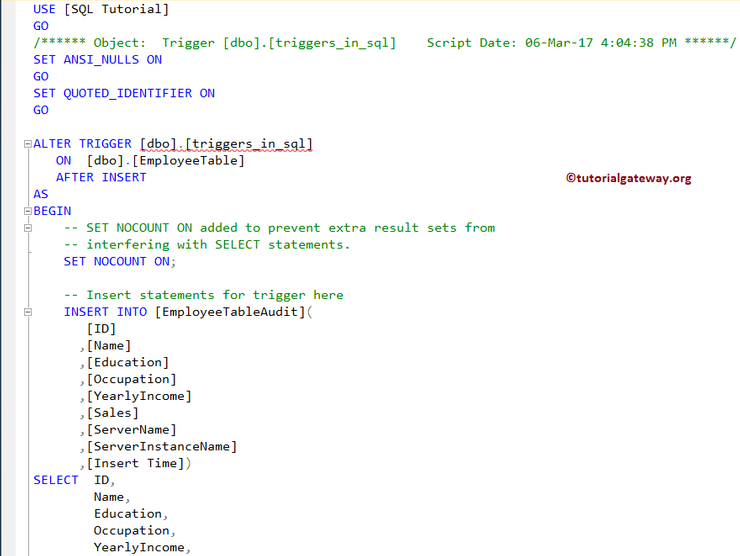
Before we get into the SQL Trigger modification, let me show you the list of available options that the Management Studio is providing to you. As we show below, you can drop it, drop and recreate, alter it, and create it.



To modify the Trigger using the [Management Studio](https://www.tutorialgateway.org/sql-server-management-studio/), Please navigate to the table that holds the Trigger. Next, select the Trigger that you want to modify (triggers\_in\_sql), and right-click on it will open the context menu. Here, select the Modify option



Once you choose the Modify option, a new query window will open with auto-generated ALTER TRIGGER code. You can modify it as per your requirement.



### Use Alter Trigger to Modify Sql Server Triggers example

The steps involved in modifying the existing triggers using the ALTER TRIGGER statement. To do so, Click New query and return the following query.

ALTER TRIGGER [dbo].[triggers\_in\_sql]

ON [dbo].[EmployeeTable]

AFTER INSERT

AS

BEGIN

-- Modify as per Your requirement

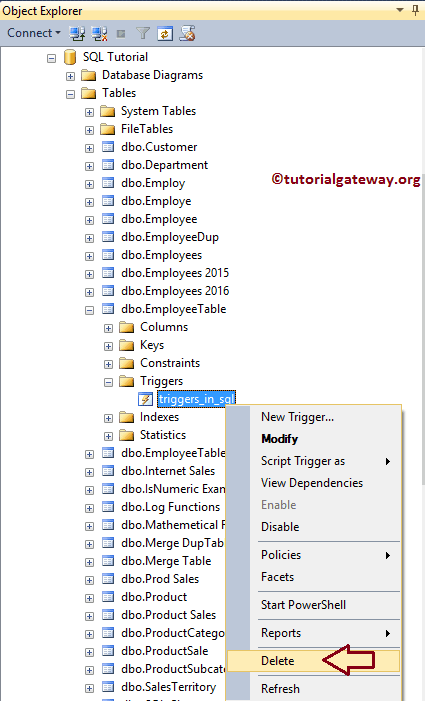
END

## Delete Triggers in SQL Server Example

The following examples will show you, How to drop, or delete the triggers using the Transact-SQL Query, and Management Studio (SSMS).

### Delete Trigger in Management Studio

Here, we show you how to delete a trigger using the SSMS. To delete a trigger using the SSMS, right-click on the trigger name that you want to delete will open the context menu. You can click on the Delete option



### Delete SQL Trigger using DROP TRIGGER

We will show you how to delete triggers using the Drop Trigger statement.

USE [SQL Tutorial]

GO

DROP TRIGGER [dbo].[triggers\_in\_sql]