**Trigger**

**✅ What is a Trigger in SQL Server?**

* A **Trigger** is a **special kind of stored procedure** that **automatically runs** when something happens in a table.
* It works like an **"auto-response"**.
* It runs **when you insert, update, or delete** data in a table.

👉 Think of a trigger like:

"**If something happens, then do this automatically**"

**📌 Simple Example: Trigger on Insert**

Let’s say we have two tables:

1. **Students** – where we insert student info
2. **StudentLogs** – where we want to keep a log of every new student added

**Step1** :

CREATE TABLE Students (

Id INT PRIMARY KEY,

Name NVARCHAR(100)

);

CREATE TABLE StudentLogs (

LogId INT IDENTITY(1,1) PRIMARY KEY,

StudentName NVARCHAR(100),

LogDate DATETIME

);

**Step2 :**

CREATE TRIGGER trg\_AfterInsertStudent

ON Students

AFTER INSERT

AS

BEGIN

INSERT INTO StudentLogs (StudentName, LogDate)

SELECT Name, GETDATE() FROM inserted;

END;

* This trigger will run **automatically** after a new student is added to the Students table.
* It will **copy the student name and date** into the StudentLogs table.

**Step3 :**

INSERT INTO Students (Id, Name) VALUES (1, 'Apsara');

SELECT \* FROM StudentLogs;

Trigger = Automatic action  
Like: "If I insert data, then do something else behind the scenes"

 ✅ **AFTER UPDATE Trigger**

 ✅ **AFTER DELETE Trigger**

✅ 1. AFTER UPDATE Trigger (log changes when a student is updated)

CREATE TRIGGER trg\_AfterUpdateStudent

ON Students

AFTER UPDATE

AS

BEGIN

INSERT INTO StudentLogs (StudentName, LogDate)

SELECT Name + ' (Updated)', GETDATE() FROM inserted;

END;

 This trigger runs **after an update** in the Students table.

 It logs the updated name into StudentLogs, with the word **(Updated)** to show it was changed.

UPDATE Students SET Name = 'Apsara R' WHERE Id = 1;

✅ 2. AFTER DELETE Trigger (log deleted student name)

CREATE TRIGGER trg\_AfterDeleteStudent

ON Students

AFTER DELETE

AS

BEGIN

INSERT INTO StudentLogs (StudentName, LogDate)

SELECT Name + ' (Deleted)', GETDATE() FROM deleted;

END;

 This trigger runs **after a student is deleted** from the Students table.

 It logs the deleted name with **(Deleted)** into StudentLogs.

DELETE FROM Students WHERE Id = 1;

### 1. ****Modify a Trigger****

Use ALTER TRIGGER if you want to **edit an existing trigger** (without dropping it):

ALTER TRIGGER trg\_AfterInsertStudent

ON Students

AFTER INSERT

AS

BEGIN

-- Change the logic here

INSERT INTO StudentLogs (StudentName, LogDate)

SELECT Name + ' (New Student)', GETDATE() FROM inserted;

END;

### 2. ****Delete a Trigger****

Use DROP TRIGGER to **remove** a trigger completely:

DROP TRIGGER trg\_AfterInsertStudent;

You can also drop multiple:

DROP TRIGGER trg\_AfterInsertStudent, trg\_AfterUpdateStudent;

## ✅ Types of Triggers in SQL Server

| **Trigger Type** | **When it Runs** | **Usage** |
| --- | --- | --- |
| **AFTER** | After the actual INSERT/UPDATE/DELETE happens | Most common |
| **INSTEAD OF** | **Before** the action happens — and can **stop or replace** the action | Used when you want to **control or block** the action |

### 🔹 2. ****INSTEAD OF Trigger****

* It **stops the original action** and **does something else instead**.

Suppose you **don’t want students to be deleted**, just log the attempt instead.

CREATE TRIGGER trg\_InsteadOfDelete

ON Students

INSTEAD OF DELETE

AS

BEGIN

INSERT INTO StudentLogs (StudentName, LogDate)

SELECT Name + ' (Delete Attempted)', GETDATE() FROM deleted;

-- No delete happens here

END;

DELETE FROM Students WHERE Id = 2;

Student is **not deleted**, but the **log is created**.