A **SQL Server trigger** is a special type of stored procedure that automatically executes in response to certain events on a specified table or view. These triggers are fired when INSERT, UPDATE, or DELETE operations occur on a table.

**Real-Life Example: Maintaining an Audit Log with an INSERT Trigger**

**Scenario:**

In a **hospital management system**, we want to track changes in the **Patients** table. Whenever a new patient is registered, we log the action in an **AuditLog** table.

**Database Schema**

1. **Patients Table:** Stores patient details.
2. **AuditLog Table:** Keeps a record of all insert operations for tracking purposes.

**SQL Script**

-- Creating the Patients table

CREATE TABLE Patients (

PatientID INT IDENTITY(1,1) PRIMARY KEY,

FullName VARCHAR(100),

Age INT,

Gender CHAR(1),

ContactNumber VARCHAR(15),

CreatedAt DATETIME DEFAULT GETDATE()

);

-- Creating the AuditLog table

CREATE TABLE AuditLog (

LogID INT IDENTITY(1,1) PRIMARY KEY,

ActionPerformed VARCHAR(50),

PatientID INT,

PerformedAt DATETIME DEFAULT GETDATE()

);

**Trigger to Log Inserts**

CREATE TRIGGER trg\_PatientAudit

ON Patients

AFTER INSERT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO AuditLog (ActionPerformed, PatientID, PerformedAt)

SELECT 'New Patient Registered', PatientID, GETDATE()

FROM inserted;

END;

**Testing the Trigger**

**Insert a New Patient**

INSERT INTO Patients (FullName, Age, Gender, ContactNumber)

VALUES ('Alice Johnson', 30, 'F', '123-456-7890');

**Check Audit Log**

SELECT \* FROM AuditLog;

**Expected Output:**

| **LogID** | **ActionPerformed** | **PatientID** | **PerformedAt** |
| --- | --- | --- | --- |
| 1 | New Patient Registered | 1 | 2025-03-15 10:30:00 |

**Explanation**

1. When a new patient is inserted into the **Patients** table, the trigger fires.
2. It automatically inserts a log entry into the **AuditLog** table.
3. The **AuditLog** helps track when a new patient was added.

**Real-World Applications**

* **User Registration Systems:** Log when new users sign up.
* **Inventory Management:** Track new stock additions.
* **Employee Onboarding:** Log employee hires for compliance.

**Scenario: Audit Log for Employee Salary Updates**

**Business Requirement**

A company wants to maintain a record of any changes made to employee salaries. Whenever an employee's salary is updated in the Employees table, the old salary and the updated salary should be recorded in an EmployeeSalaryAudit table.

**SQL Server Implementation**

**Step 1: Create the Employees Table**

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

Name NVARCHAR(100),

Salary DECIMAL(10,2)

);

**Step 2: Create the EmployeeSalaryAudit Table**

CREATE TABLE EmployeeSalaryAudit (

AuditID INT IDENTITY(1,1) PRIMARY KEY,

EmployeeID INT,

OldSalary DECIMAL(10,2),

NewSalary DECIMAL(10,2),

ChangeDate DATETIME DEFAULT GETDATE()

);

**Step 3: Create an AFTER UPDATE Trigger**

CREATE TRIGGER trg\_AuditSalaryChange

ON Employees

AFTER UPDATE

AS

BEGIN

-- Insert old and new salary values into the audit table

INSERT INTO EmployeeSalaryAudit (EmployeeID, OldSalary, NewSalary, ChangeDate)

SELECT i.EmployeeID, d.Salary, i.Salary, GETDATE()

FROM inserted i

INNER JOIN deleted d ON i.EmployeeID = d.EmployeeID

WHERE i.Salary <> d.Salary; -- Only log changes where salary is modified

END;

**Step 4: Test the Trigger**

**Insert an Employee Record**

INSERT INTO Employees (EmployeeID, Name, Salary)

VALUES (1, 'John Doe', 50000);

**Update the Employee’s Salary**

UPDATE Employees

SET Salary = 55000

WHERE EmployeeID = 1;

**Check the Audit Log**

SELECT \* FROM EmployeeSalaryAudit;

**How This Works**

1. The **trigger** automatically executes **after an UPDATE** on the Employees table.
2. It logs the **old and new salary values** in the EmployeeSalaryAudit table.
3. If an update is made but the salary remains unchanged, no entry is added to the audit table.

**Real-Life Applications of SQL Triggers**

* **Audit Logs** – Track changes to critical financial or user data.
* **Automatic Notifications** – Send emails or alerts when important changes occur.
* **Enforcing Business Rules** – Prevent invalid data modifications.
* **Data Synchronization** – Keep data updated across multiple tables or databases.

A **DELETE trigger** in SQL Server is a type of trigger that executes automatically when a DELETE operation is performed on a table. A real-life example of using a DELETE trigger is maintaining a **soft delete** or **audit log** when records are deleted.

**Scenario: Audit Log for Deleted Employees**

Imagine you have an **Employee** table, and you want to maintain a record of deleted employees in a separate **EmployeeAudit** table instead of losing the data permanently.

**Step 1: Create Tables**

First, create the Employee and EmployeeAudit tables.

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY,

Name NVARCHAR(100),

Department NVARCHAR(50),

Salary DECIMAL(10,2)

);

CREATE TABLE EmployeeAudit (

AuditID INT IDENTITY(1,1) PRIMARY KEY,

EmployeeID INT,

Name NVARCHAR(100),

Department NVARCHAR(50),

Salary DECIMAL(10,2),

DeletedAt DATETIME DEFAULT GETDATE()

);

**Step 2: Create DELETE Trigger**

Create a trigger that will capture deleted employee details and insert them into the EmployeeAudit table before deletion.

CREATE TRIGGER trg\_AuditEmployeeDeletion

ON Employee

AFTER DELETE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO EmployeeAudit (EmployeeID, Name, Department, Salary, DeletedAt)

SELECT EmployeeID, Name, Department, Salary, GETDATE()

FROM deleted;

PRINT 'Audit log created for deleted employee(s)';

END;

**Step 3: Test the Trigger**

**Insert Sample Data**

INSERT INTO Employee (EmployeeID, Name, Department, Salary)

VALUES (1, 'John Doe', 'IT', 75000),

(2, 'Jane Smith', 'HR', 65000);

**Delete an Employee**

DELETE FROM Employee WHERE EmployeeID = 1;

**Check the Audit Log**

SELECT \* FROM EmployeeAudit;

**How It Works**

* When an Employee is deleted, the trigger captures the deleted row(s).
* The data is inserted into EmployeeAudit before being removed from Employee.
* The audit log keeps track of deleted employees along with the timestamp.

This method helps **track deleted records** for compliance, auditing, or data recovery purposes.