upGrad ENTERPRISE

SQL Server – Triggers and Transactions

By
Narasimha Rao T
Microsoft.Net FSD Trainer
Professional Development Trainer
tnrao.trainer@gmail.com

1. What are Triggers?

Definition:

A **trigger** in SQL Server is a special type of stored procedure that automatically executes in response to certain events on a particular table or view.

Example:

```
CREATE TRIGGER trg_AuditInsert
ON Employees
AFTER INSERT
AS
BEGIN
    INSERT INTO AuditLog (Action, ActionDate)
    VALUES ('Insert into Employees', GETDATE())
END
```

2. Why Do We Need Triggers?

- **To enforce business rules** automatically.
- **To maintain audit trails** (e.g., log when data is changed).
- **To prevent invalid transactions**.
- **To execute cascading changes**.
- **To enhance automation** in database operations.

3. Advantages of Triggers

- Enforce **data integrity** without modifying application logic.
- Keep an **automatic history** of changes (for auditing).
- Can be used to **prevent changes** when certain conditions aren't met.
- Run **automatically** — no need for explicit call.

4. How to Create Triggers in SQL Server

Syntax:

```
CREATE TRIGGER trigger_name
ON table_name
AFTER | INSTEAD OF [INSERT | UPDATE | DELETE]
AS
BEGIN
    -- SQL statements
END
```

Example – Trigger to prevent deleting records:

```
CREATE TRIGGER trg_PreventDelete
ON Employees
INSTEAD OF DELETE
AS
BEGIN
    RAISERROR('Delete operation not allowed on Employees table.', 16, 1);
END
```

5. AFTER vs. INSTEAD OF Triggers

Type	Description
AFTER	Executes **after** the triggering DML operation completes.
INSTEAD OF	**Replaces** the DML operation – original does NOT run.

Example: AFTER Trigger

```
CREATE TRIGGER trg_AfterUpdate
ON Employees
AFTER UPDATE
AS
BEGIN
    INSERT INTO AuditLog (Action, ActionDate)
    VALUES ('Employee record updated', GETDATE())
END
```

Example: INSTEAD OF Trigger

```
CREATE TRIGGER trg_InsteadOfDelete
ON Employees
INSTEAD OF DELETE
AS
BEGIN
    PRINT 'Delete operation is restricted.'
END
```

6. INSERTED and DELETED Pseudo-Tables

- SQL Server creates **virtual tables** during DML operations:
  - `INSERTED` – holds **new data** for INSERT/UPDATE.
  - `DELETED` – holds **old data** for DELETE/UPDATE.

Example:

```
CREATE TRIGGER trg_LogUpdate
ON Employees
AFTER UPDATE
AS
BEGIN
    INSERT INTO EmployeeAudit(EmployeeID, OldSalary, NewSalary, ModifiedDate)
    SELECT d.EmployeeID, d.Salary, i.Salary, GETDATE()
    FROM DELETED d
    JOIN INSERTED i ON d.EmployeeID = i.EmployeeID
END
```

7. Real-Time Case Study: Auditing Employee Updates

Scenario: HR wants to track salary changes in the `Employees` table.

Solution:
Create an `AFTER UPDATE` trigger using `INSERTED` and `DELETED` to record old and new salaries in an audit table.

Audit Table:

```
CREATE TABLE EmployeeAudit (
    AuditID INT IDENTITY,
    EmployeeID INT,
    OldSalary MONEY,
    NewSalary MONEY,
    ModifiedDate DATETIME
);
```

Trigger: Refer to `trg_LogUpdate` above.

8. Working with Transactions in SQL Server

What is a Transaction?

A **transaction** is a group of operations treated as a single unit.
It **ensures data integrity** using **ACID** properties (Atomicity, Consistency, Isolation, Durability).

Real-Time Case Study: E-Commerce Checkout

Scenario: A customer buys items from an e-commerce store.

Transaction includes:

- Deduct stock from Products table.
- Create a new order in Orders.
- Deduct amount from UserWallet.

ACID Application:

- **Atomicity**: All steps succeed or none.
- **Consistency**: Inventory and payment constraints must be respected.
- **Isolation**: Another user can’t order the same product while it's being processed.
- **Durability**: Once the order is confirmed, it survives power failure.

9. Using COMMIT and ROLLBACK

Example:

```
BEGIN TRANSACTION;

BEGIN TRY
    UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = 1;
    UPDATE Accounts SET Balance = Balance + 100 WHERE AccountID = 2;

    COMMIT TRANSACTION;
END TRY
BEGIN CATCH
    ROLLBACK TRANSACTION;
    PRINT ERROR_MESSAGE();
END CATCH
```

Real-Time Case Study: Bank Fund Transfer

Scenario: Transfer ₹500 from Account A to B. Both updates must succeed, or none.

Example:

```
BEGIN TRANSACTION;

BEGIN TRY
    UPDATE Accounts SET Balance = Balance - 500 WHERE AccountID = 101;
    UPDATE Accounts SET Balance = Balance + 500 WHERE AccountID = 102;

    COMMIT;
    PRINT 'Transaction Successful';
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Transaction Failed: ' + ERROR_MESSAGE();
END CATCH
```

upGrad ENTERPRISE

Q & A

Narasimha Rao T
tnrao.trainer@gmail.com