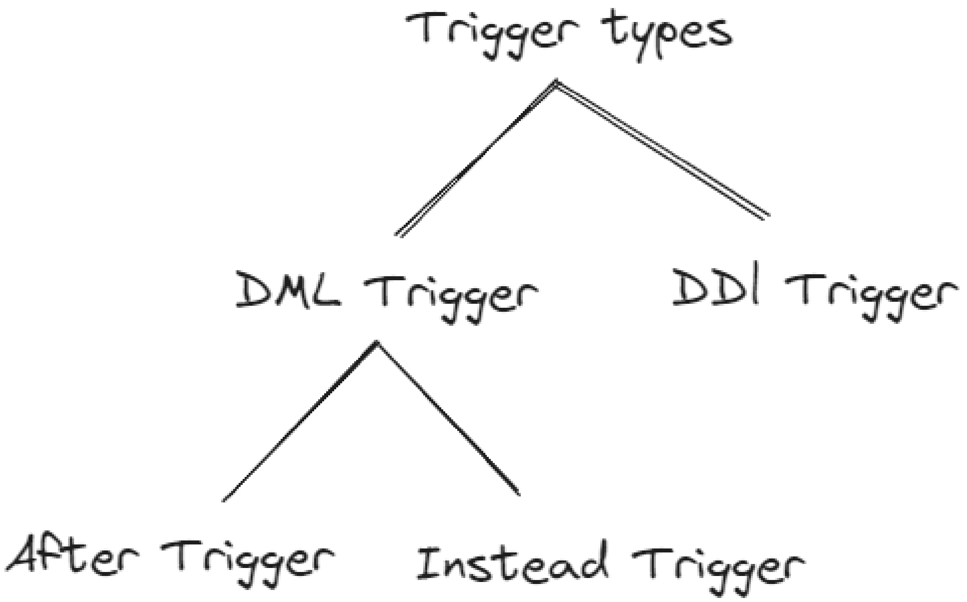
Triggers in SQL Server

SQL Server triggers are special stored procedures that are executed automatically in response to the database object, database, and server events. SQL Server provides Two type of triggers



1- Data definition language (DDL) triggers which fire in response to CREATE, ALTER, and DROP statements. DDL triggers also fire in response to some system stored procedures that perform DDL-like operations.

One of Purpose ➔ Make data base secure , trigger can prevent any one to make DDL in your data base ,any one cannot Create ,Alter and Drop .

Note ➔ We can use trigger at many times and many purposes

Ex:

```
  i. CREATE TRIGGER trigger_name
 ii. ON { DATABASE |  ALL SERVER}
iii. [WITH ddl_trigger_option]
 iv. For➔crate ,alter ,drop table
  v. As print➔Don't allow make this
 vi. Roolback ;
```

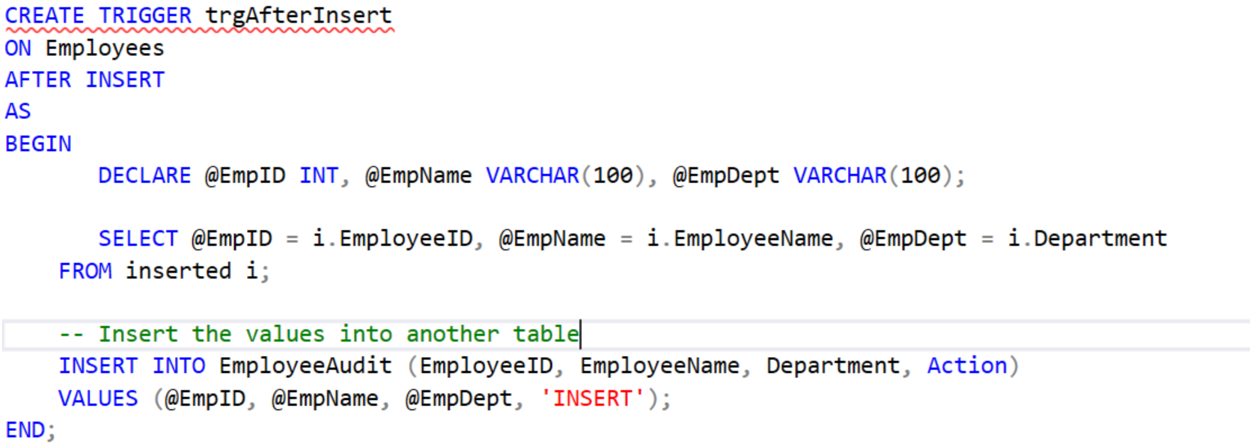
Summaries : After execute this trigger , no one can crate ,alter ,drop table.

DML Triggers

2- **AFTER Triggers:** These triggers execute after an insert, update, or delete operation.
3- **INSTEAD OF Triggers:** These triggers execute in place of an insert, update, or delete operation.

Ex :

AFTER Triggers Ex :



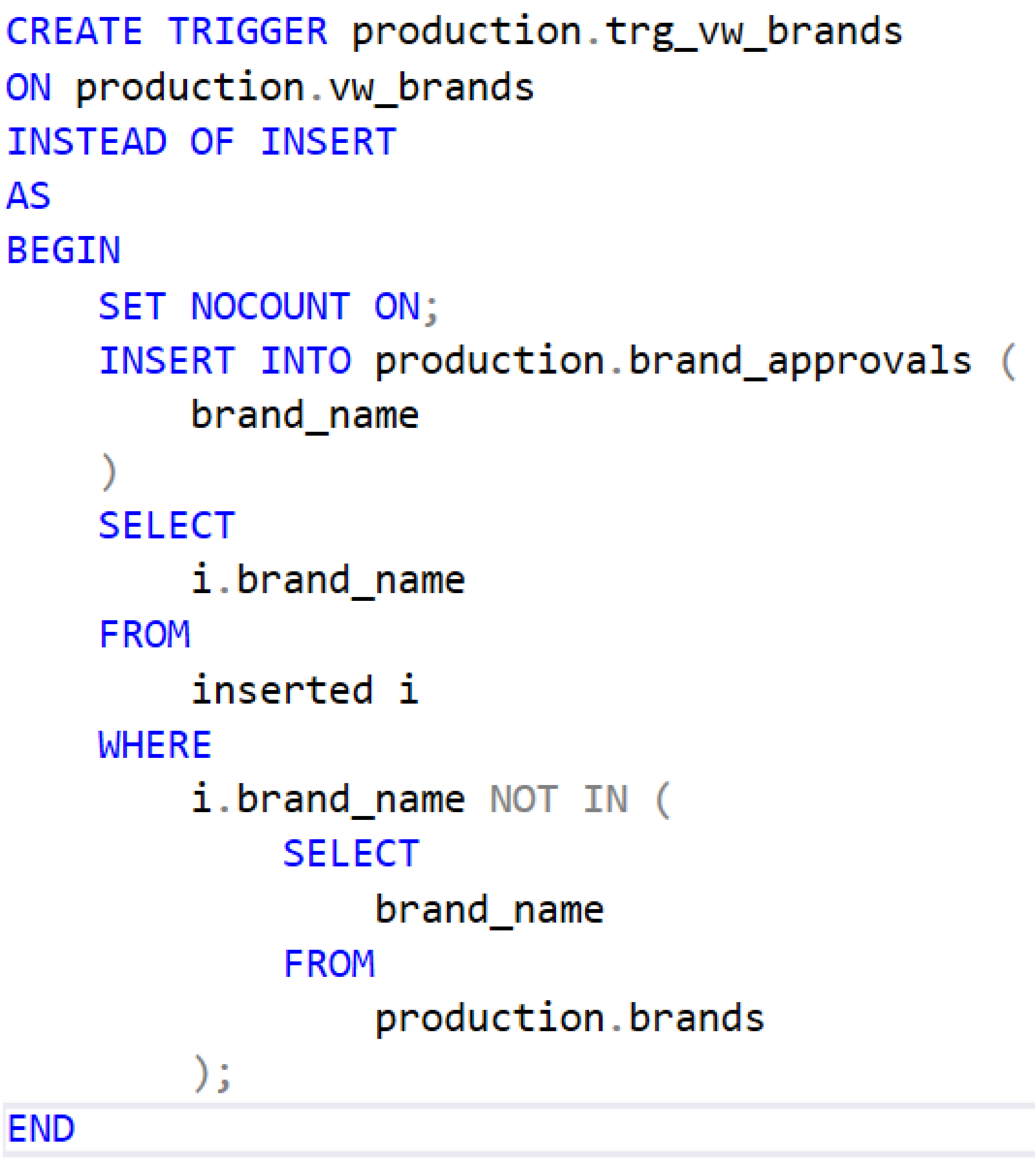
```
CREATE TRIGGER trgAfterInsert
ON Employees
AFTER INSERT
AS
BEGIN
       DECLARE @EmpID INT, @EmpName VARCHAR(100), @EmpDept VARCHAR(100);

       SELECT @EmpID = i.EmployeeID, @EmpName = i.EmployeeName, @EmpDept = i.Department
    FROM inserted i;

    -- Insert the values into another table
    INSERT INTO EmployeeAudit (EmployeeID, EmployeeName, Department, Action)
    VALUES (@EmpID, @EmpName, @EmpDept, 'INSERT');
END;
```

Summaries : When make insert in Employee table ,After trigger make automatic insert in Employee Adult Table .

INSTEAD OF Triggers Ex :



```
CREATE TRIGGER production.trg_vw_brands
ON production.vw_brands
INSTEAD OF INSERT
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO production.brand_approvals (
        brand_name
    )
    SELECT
        i.brand_name
    FROM
        inserted i
    WHERE
        i.brand_name NOT IN (
            SELECT
                brand_name
            FROM
                production.brands
        );
END
```

Summaries :before execute the query check that a brand not existed in a Brand table