What is Triggers

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

- Data manipulation language (DML) triggers which are invoked automatically in response to INSERT, UPDATE, and DELETE events against tables.
- 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.
- Logon triggers which fire in response to LOGON events

Types of Triggers

Triggers are generally classified based on the timing of their execution relative to the event that causes them to fire:

- BEFORE Triggers: These triggers execute before the triggering event. They are typically used to validate or modify data before it is inserted or updated in the database. For example, a BEFORE INSERT trigger can be used to ensure that no duplicate entries are added to a table.
- AFTER Triggers: These triggers execute after the triggering event. They are often used for auditing, logging changes, or maintaining related data in other tables. For example, an AFTER UPDATE trigger can log the old and new values of a modified record.
- INSTEAD OF Triggers: Commonly used in views, these triggers replace the triggering event with their own logic. For example, an INSTEAD OF INSERT trigger on a view can allow inserting data into multiple underlying tables.

Use Cases for Triggers

Triggers can be applied in various scenarios to enhance database functionality:

- Data Integrity Enforcement: Triggers ensure that data modifications adhere to business rules. For example, a trigger might ensure that the salary of an employee does not exceed a certain limit based on their job title.
- Audit and Logging: Triggers are useful for automatically recording changes to sensitive data. An AFTER UPDATE trigger can log modifications to employee records, including who made the changes and when.
- Automatic Calculations: Triggers can automate the calculation of values based on data changes. For instance, a BEFORE INSERT trigger could calculate and set a TotalPrice column based on the quantity and unit price of an item.
- Cascading Operations: Triggers can enforce cascading actions across related tables. For example, an AFTER DELETE trigger might automatically remove related records from other tables when a primary record is deleted.

How To Create Triggers

Creating triggers in SQL involves defining an automatic set of actions that will be executed in response to certain events on a table, such as INSERT, UPDATE, or DELETE.

Example:-

```
CREATE TRIGGER trigger_name
ON table_name
AFTER INSERT, UPDATE, DELETE
AS
BEGIN
   -- Trigger logic here
   UPDATE table_name
   SET column_name = value
   WHERE condition;
END;
```