Intelligent Data Management with SQL Server

Session: 12

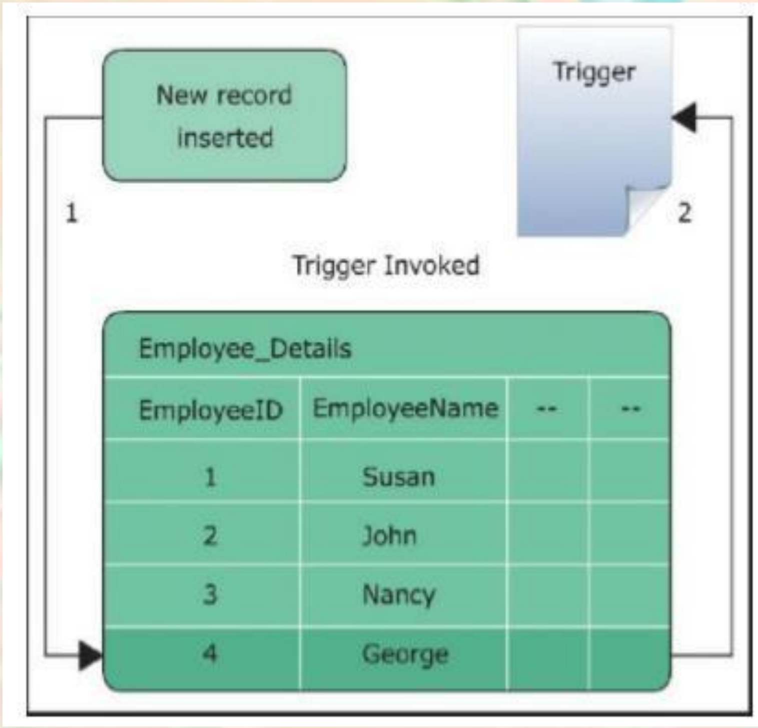
Triggers

Objectives

- Explain triggers
- Explain the procedure to create and alter DML triggers
- Describe nested triggers
- Describe update functions
- Explain the handling of multiple rows in a session
- Explain the performance implication of triggers implication

Introduction

- Trigger is a special type of stored procedure that is executed when an attempt is made to modify data in a table for which triggers are created.
- Unlike standard system stored procedures, triggers cannot be executed directly, nor do they pass or receive parameters.
- Triggers will get executed automatically when INSERT, UPDATE, or DELETE operations are performed.
- In SQL Server, triggers are created using the CREATE TRIGGER statement.



EmployeeID	EmployeeName	--	--
1	Susan		
2	John		
3	Nancy		
4	George		

Triggers

Use of Triggers 1-4

- ➢ Triggers can contain complex processing logic and are generally used for maintaining low-level data integrity.
- ➢ The primary uses of triggers can be classified as follows:

Cascade changes in related tables

- ➢ Users can use a trigger to cascade changes through related tables.

For example,

Consider a table Sales.Customer in AdventureWorks2019 database having a foreign key, PersonID referencing the primary key, BusinessEntityID of the Person.Person table. If an update or a delete event occurs in the Person.Person table, an update or delete trigger can be defined to cascade these changes to the Sales.Customer table.

Use of Triggers 2-4

Enforce complex data integrity

DML Triggers

- DML triggers execute when data is inserted, modified, or deleted in a table or a view using the INSERT, UPDATE, or DELETE statements.

DDL Triggers

- DDL triggers execute when a table or a view is created, modified, or deleted using the CREATE, ALTER, or DROP statements.

Logon Triggers

- Logon triggers execute stored procedures when a session is established with a LOGON event. These triggers are invoked after the login authentication is complete and before the actual session is established. Logon triggers control server sessions by restricting invalid logins or limiting the number of sessions.

Use of Triggers 3-4

Define custom error messages

- Used for providing more suitable or detailed explanations in certain error situations.
- Triggers can be used to invoke such predefined custom error messages when the relevant error conditions occur.

Maintain denormalized data

- Low-level data integrity can be maintained in denormalized database environments using triggers.
- It generally refers to redundant or derived data.

For example, if the value of the year is to be checked against complete dates, a trigger can be used to perform the check.

Use of Triggers 4-4

Compare before and after states of data

Triggers provide the option to reference changes that are made to data by INSERT, UPDATE, and DELETE statements.

This allows users to reference the affected rows when modifications are carried out through triggers.

Types of Triggers

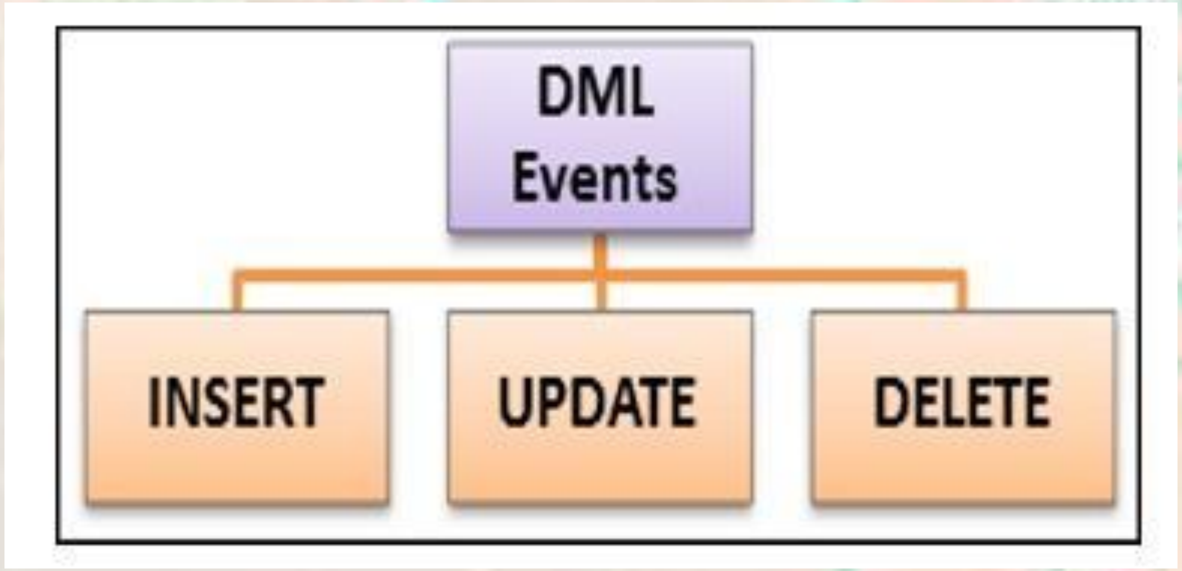
Triggers are automatically executed when a language event occurs in a table or a view.

Language events can be classified as DML and DDL events.

- Triggers associated with DML events are known as DML triggers, whereas triggers associated with DDL events are known as DDL triggers.

Creating DML Triggers

- DML triggers are executed either on completion of the DML events or in place of the DML events.
- They enforce referential integrity by cascading changes to related tables when a row is modified.



Introduction to Inserted and Deleted Tables

SQL statements in DML triggers use two special types of tables to modify data in the database.

When the data is inserted, updated, or deleted, SQL Server creates and manages these tables automatically.

Tables temporarily store the original as well as the modified data.

Inserted Table

- New rows are added to both the inserted table and the trigger table.
- The rows in the inserted table are copies of the new rows in the trigger table.

Deleted Table

- During the execution of a DELETE or UPDATE statement, rows are deleted from the trigger table and transferred to the deleted table.

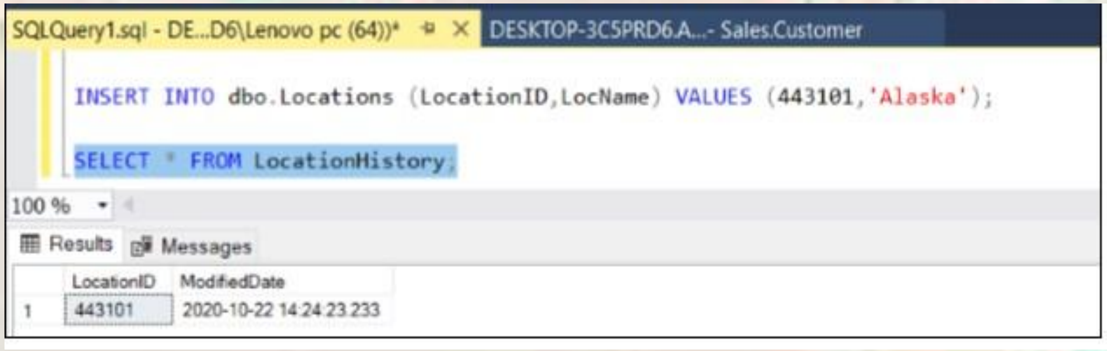
Insert Triggers

An INSERT trigger is executed when a new record is inserted in a table.

INSERT trigger ensures that the value being entered conforms to the constraints defined on that table.

When a user inserts a record in the table, INSERT trigger saves a copy of that record in inserted table.

It then checks whether the new value in the inserted table conforms to the specified constraints or not.


SQLQuery1.sql - DE...D6\Lenovo pc (64))*
DESKTOP-3C5PRD6.A...- Sales.Customer
100 %
Results
Messages

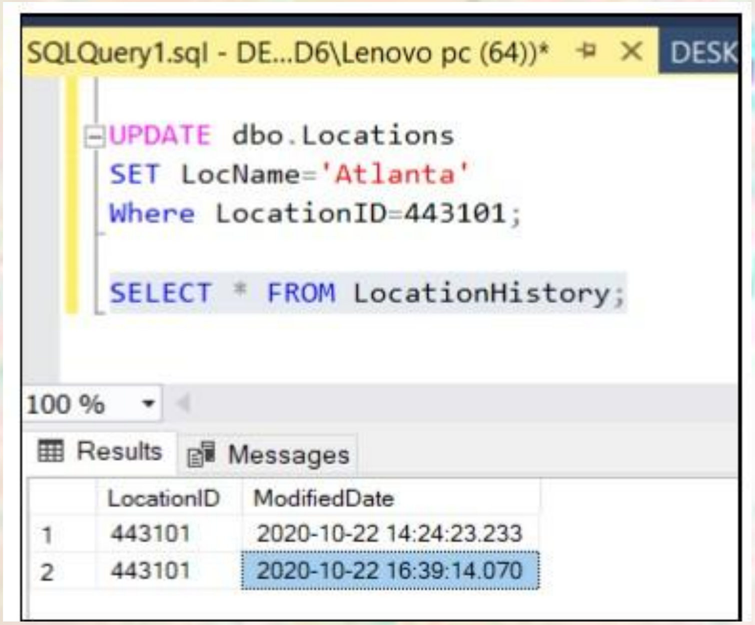
```
INSERT INTO dbo.Locations (LocationID,LocName) VALUES (443101,'Alaska');

SELECT * FROM LocationHistory;
```

```
   LocationID  ModifiedDate
1  443101      2020-10-22 14:24:23.233
```

Insert Trigger Test

Update Triggers

- The UPDATE trigger copies the original record in the Deleted table and the new record into the Inserted table when a record is updated.


SQLQuery1.sql - DE...D6\Lenovo pc (64))* DESK

```
UPDATE dbo.Locations
SET LocName='Atlanta'
Where LocationID=443101;

SELECT * FROM LocationHistory;
```

	LocationID	ModifiedDate
1	443101	2020-10-22 14:24:23.233
2	443101	2020-10-22 16:39:14.070

Update Trigger Test

Delete Triggers

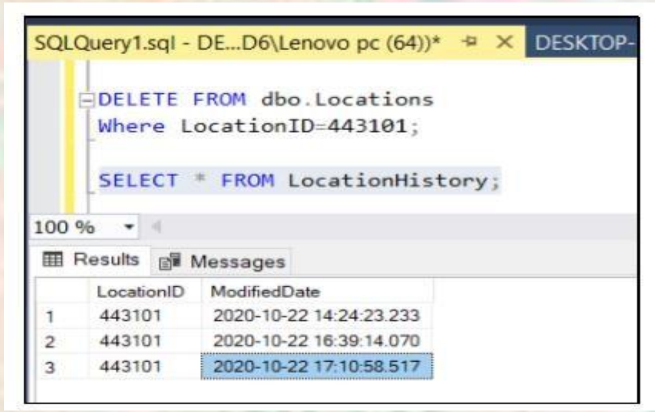
The DELETE trigger can be created to restrict a user from deleting a particular record in a table.

Following will happen if the user tries to delete the record:

The record is deleted from the trigger table and inserted in the Deleted table.

If there is a constraint on the record to prevent deletion, the DELETE trigger displays an error message.

The deleted record stored in the Deleted table is copied back to the trigger table.



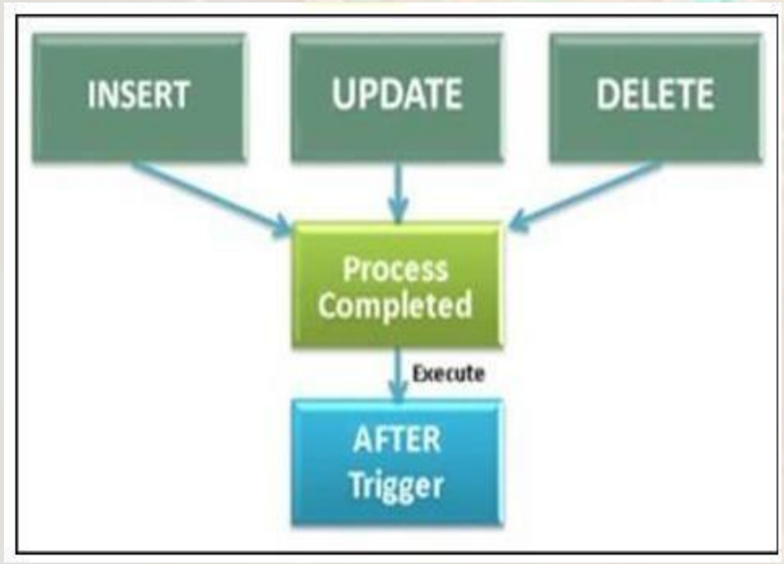
```
DELETE FROM dbo.Locations
Where LocationID=443101;

SELECT * FROM LocationHistory;
```

	LocationID	ModifiedDate
1	443101	2020-10-22 14:24:23.233
2	443101	2020-10-22 16:39:14.070
3	443101	2020-10-22 17:10:58.517

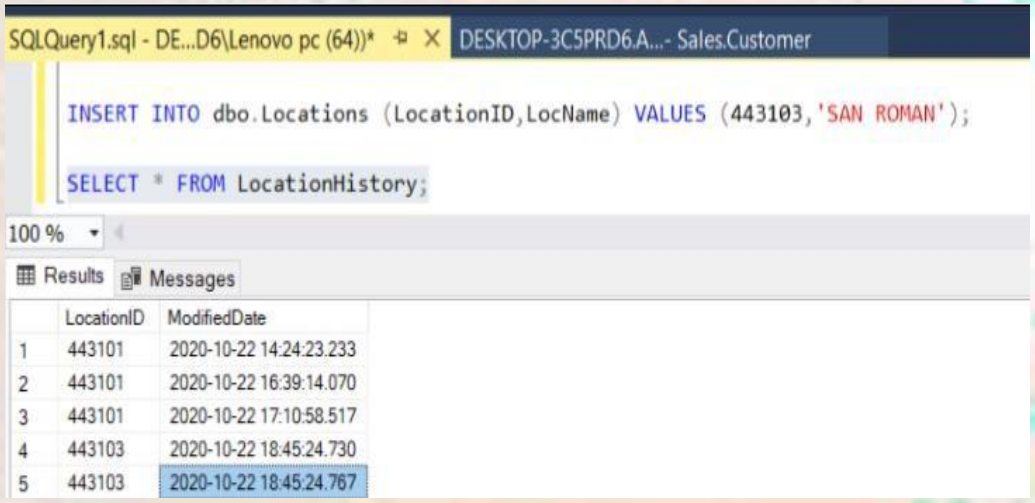
Delete Trigger Test

AFTER Triggers

- An AFTER trigger is executed on completion of INSERT, UPDATE, or DELETE operations.
- An AFTER trigger is executed when the constraint check in the table is completed.



AFTER Triggers


SQLQuery1.sql - DE...D6\Lenovo pc (64))*
DESKTOP-3C5PRD6.A...- Sales.Customer
100 %
Results
Messages

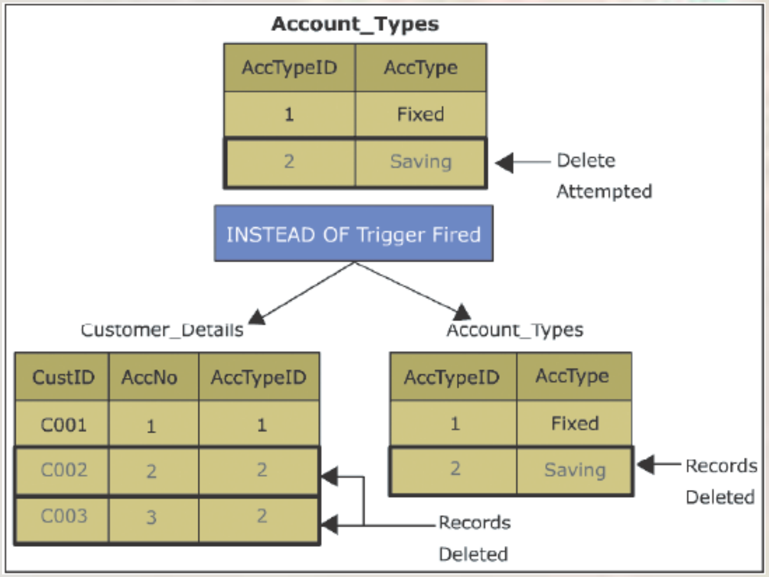
```
INSERT INTO dbo.Locations (LocationID,LocName) VALUES (443103,'SAN ROMAN');

SELECT * FROM LocationHistory;
```

	LocationID	ModifiedDate
1	443101	2020-10-22 14:24:23.233
2	443101	2020-10-22 16:39:14.070
3	443101	2020-10-22 17:10:58.517
4	443103	2020-10-22 18:45:24.730
5	443103	2020-10-22 18:45:24.767

After Insert TriggerTest

INSTEAD OF Triggers

- An INSTEAD OF trigger is executed in place of the INSERT, UPDATE, or DELETE operations.
- They are executed before constraint checks are performed on the table.
- These triggers are executed after the creation of the Inserted and Deleted tables.



Account_Types

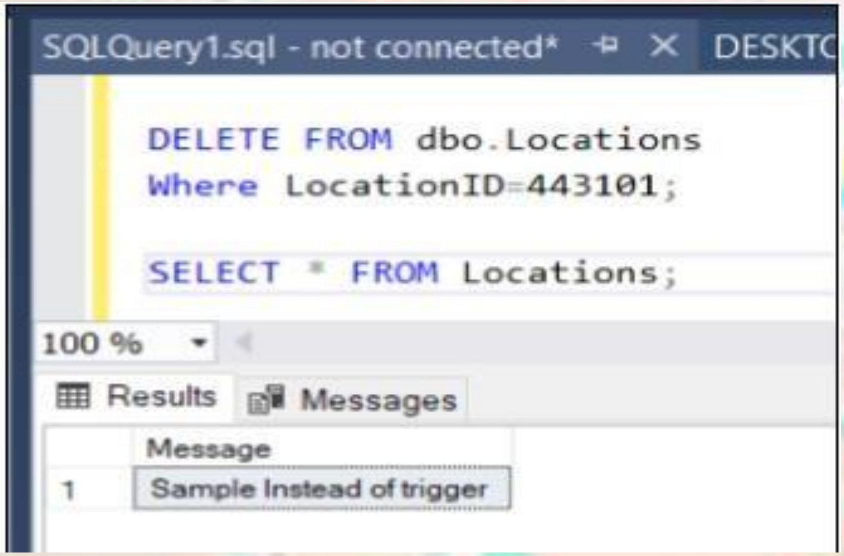
AccTypeID	AccType
1	Fixed
2	Saving

Customer_Details

CustID	AccNo	AccTypeID
C001	1	1
C002	2	2
C003	3	2

Account_Types

AccTypeID	AccType
1	Fixed
2	Saving



SQLQuery1.sql - not connected*

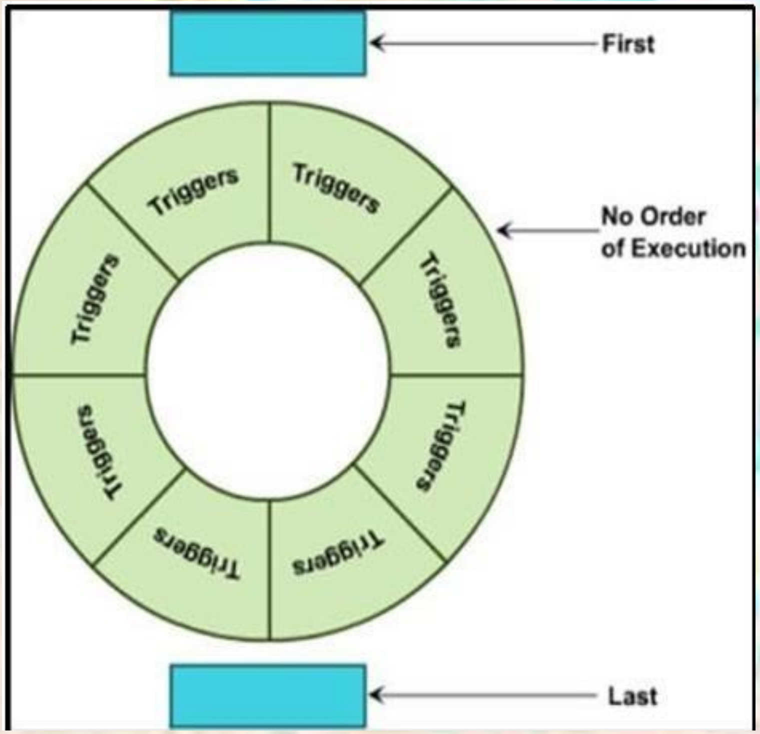
```
DELETE FROM dbo.Locations
Where LocationID=443101;

SELECT * FROM Locations;
```

	Message
1	Sample Instead of trigger

Instead of Trigger Test

Execution Order of DML Triggers

- SQL Server allows users to specify which AFTER trigger is to be executed first and which is to be executed last.
- All the triggering actions have a first and last trigger defined for them.
- However, no two triggering actions on a table can have the same first and last triggers.



Execution Order of DML Triggers

Viewing Definitions of DML Triggers

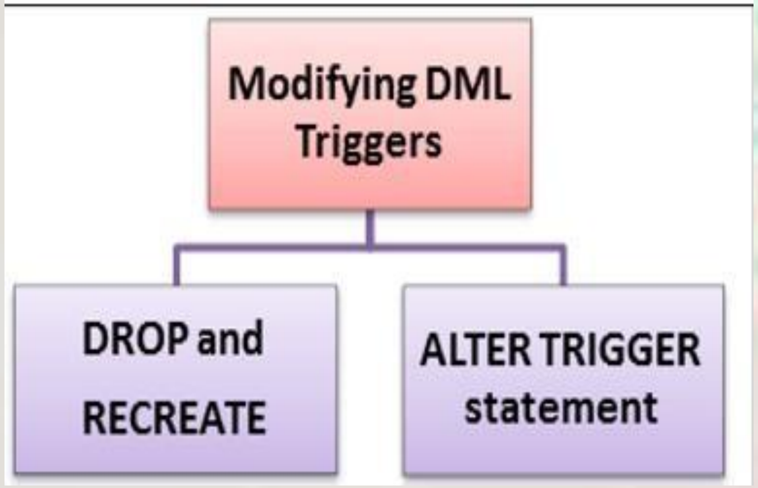
A trigger definition includes the trigger name, the table on which the trigger is created, the triggering actions, and the SQL statements that are executed.

SQL Server provides sp_helptext stored procedure to retrieve the trigger definitions.

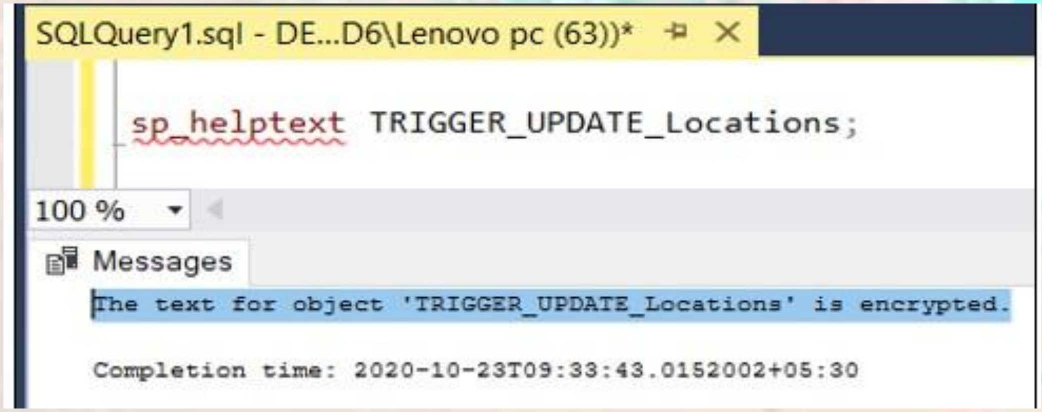
Modifying Definitions of DML Triggers

Trigger parameters are defined at the time of creating a trigger.

These parameters include the type of triggering action that invokes the trigger and the SQL statements that are executed.



Modifying DML Triggers



```
sp_helptext TRIGGER_UPDATE_Locations;
```

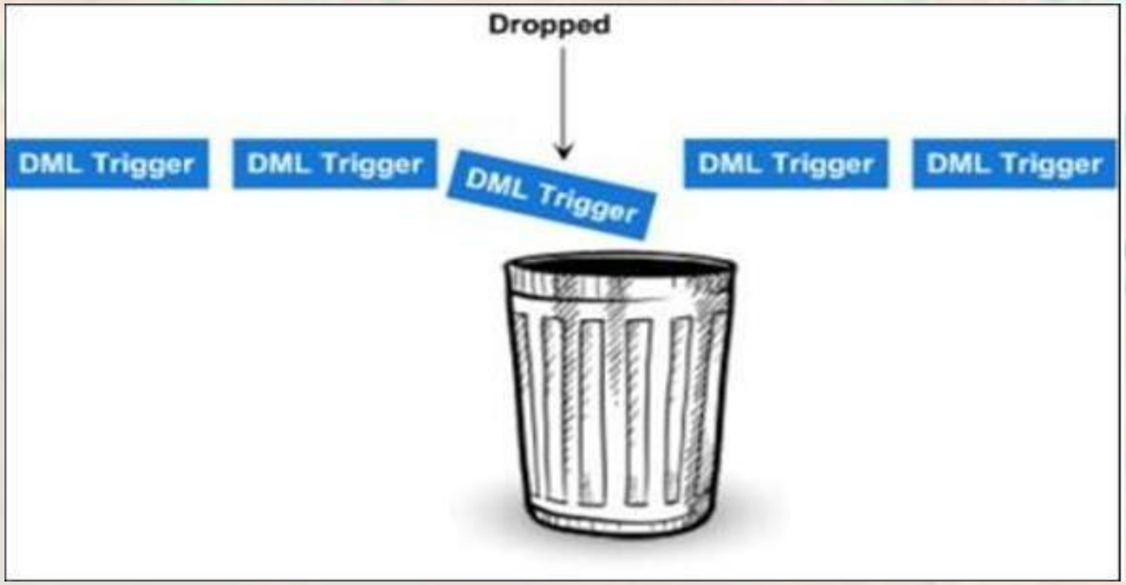
```
The text for object 'TRIGGER_UPDATE_Locations' is encrypted.

Completion time: 2020-10-23T09:33:43.0152002+05:30
```

Encrypted Trigger

Dropping DML Triggers

SQL Server provides the option of dropping a DML trigger created on a table if the trigger is no longer required.

When a table is dropped, all the triggers defined on that table are also dropped.



Dropping DML Triggers

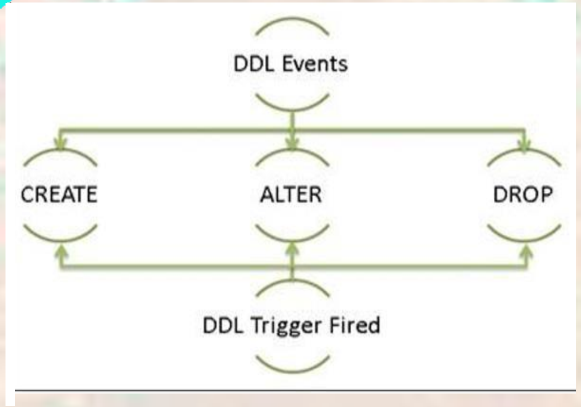
DDL Triggers

A Data Definition Language (DDL) triggers execute stored procedures when DDL events such as CREATE, ALTER, and DROP statements occur in the database or the server.

It can be used to prevent modifications in the database schema.

A schema is a collection of objects such as tables, views, and so forth in a database.

These are defined either at the database level or at the server level.

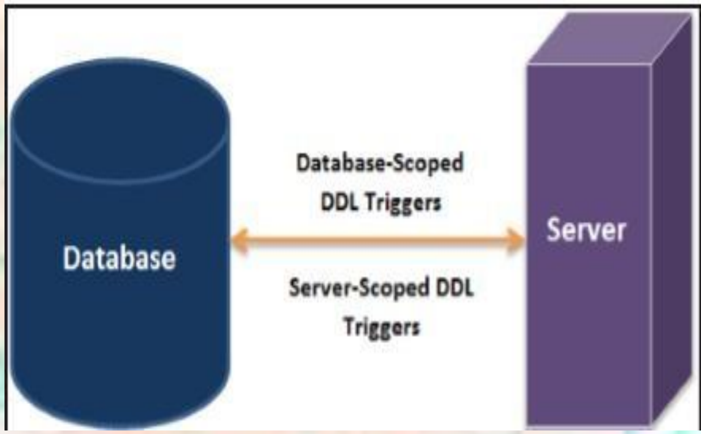


DDL Triggers

Scope of DDL Triggers 1-2

DDL triggers are invoked by SQL statements executed either in the current database or on the current server.

The scope of the DDL trigger depends on whether the trigger executes for database events or server events



Scope of DDL Triggers

DDL Triggers are classified into two types

- Database-Scoped DDL Triggers
- Server-Scoped DDL Triggers

Scope of DDL Triggers 2-2

Database-Scoped DDL Triggers

- They are invoked by the events that modify the database schema.
- Stored in the database and execute on DDL events, except those related to temporary tables.

Server-Scoped DDL Triggers

- Server-scoped DDL triggers are invoked by DDL events at the server level.
- Stored in the master database.

Nested Triggers

Both DDL and DML triggers are nested when a trigger implements an action that initiates another trigger.

DDL and DML triggers can be nested up to 32 levels.

If the nested triggers are allowed then, the triggers in the sequence start an infinite loop.

Nested triggers can be used to perform the functions such as storing the backup of the rows that are affected by the previous actions.

UPDATE()

- UPDATE( ) function returns a Boolean value that specifies whether an UPDATE or INSERT action was performed on a specific view or column of a table.

- UPDATE( ) function can be used anywhere inside the body of a Transact-SQL UPDATE or INSERT trigger to test whether the trigger should execute some actions.

Handling of Multiple Rows in a Session

When a user writes the code for a DML trigger then, the statement that causes the trigger to fire will be single statement.

This single statement will affect multiple rows of data, instead of a single row.

This is a common behavior for DELETE and UPDATE triggers as these statements often affect multiple rows.

LOGON Triggers

- Logon triggers fire stored procedures in response to a LOGON event.
- This event is raised when a user session is established with an instance of SQL Server.

LOGON triggers are created at the server level and are useful in following cases:

- To audit login activity
- To control the login activity

Performance Implication of Triggers

Triggers do not carry overheads, rather they are quite responsive.

Many performance issues can occur because of the logic present inside the trigger.

Similarly, consider that the trigger executes various SQL statements against other tables separate from the Inserted and Deleted tables.

This will again result in the slowdown of speed of SQL statements that are within the trigger.

Summary

- A trigger is a stored procedure that is executed when an attempt is made to insert, update, or delete data in a table that is protected by the trigger.
- Logon triggers execute stored procedures when a session is established with a LOGON event.
- DML triggers are executed when DML events occur in tables or views.
- The INSERT trigger is executed when a new record is inserted in a table.
- The UPDATE trigger copies the original record in the Deleted table and the new record into the Inserted table when a record is updated.
- The DELETE trigger can be created to restrict a user from deleting a particular record in a table.
- The AFTER trigger is executed on completion of INSERT, UPDATE, or DELETE operations.