**Trigger**

A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server. DML, DDL, or logon trigger can be created.

**Why Use Triggers?**

To improve data integrity, trigger can be used. When an action is performed on data, it is possible to check if the manipulation of the data concurs with the underlying business rules, and thus avoids erroneous entries in a table. For example:We might want to ship a free item to a client with the order, if it totals more than $1000. A trigger will be built to check the order total upon completion of the order, to see if an extra order line needs to be inserted.

In a banking scenario, when a request is made to withdraw cash from a cash point, the stored procedure will create a record on the client's statement table for the withdrawal, and the trigger will automatically reduce the balance as required. The trigger may also be the point at which a check is made on the client's balance to verify that there is enough balance to allow the withdrawal. By having a trigger on the statement table, we are secure in the knowledge that any statement entry made, whether withdrawal or deposit, will be validated and processed in one central place.

DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view. These triggers fire when any valid event fires, whether table rows are affected or not. For more information, see DML Triggers.

DDL triggers run in response to a variety of data definition language (DDL) events. These events primarily correspond to Transact-SQL CREATE, ALTER, and DROP statements, and certain system stored procedures that perform DDL-like operations.

Logon triggers fire in response to the LOGON event that's raised when a user's session is being established. You can create triggers directly from Transact-SQL statements or from methods of assemblies that are created in the Microsoft .NET Framework common language runtime (CLR) and uploaded to an instance of SQL Server. SQL Server lets you create multiple triggers for any specific statement.

1. -- Trigger on an INSERT, UPDATE, or DELETE statement to a table or view (DML Trigger)

CREATE [ OR ALTER ] TRIGGER [ schema\_name . ]trigger\_name

ON { table | view }

[ WITH <dml\_trigger\_option> [ ,...n ] ]

{ FOR | AFTER | INSTEAD OF }

{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }

[ WITH APPEND ]

[ NOT FOR REPLICATION ]

AS { sql\_statement [ ; ] [ ,...n ] | EXTERNAL NAME <method specifier [ ; ] > }

<dml\_trigger\_option> ::=

[ ENCRYPTION ]

[ EXECUTE AS Clause ]

<method\_specifier> ::=

assembly\_name.class\_name.method\_name

2) -- Trigger on an INSERT, UPDATE, or DELETE statement to a

-- table (DML Trigger on memory-optimized tables)

CREATE [ OR ALTER ] TRIGGER [ schema\_name . ]trigger\_name

ON { table }

[ WITH <dml\_trigger\_option> [ ,...n ] ]

{ FOR | AFTER }

{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }

AS { sql\_statement [ ; ] [ ,...n ] }

<dml\_trigger\_option> ::=

[ NATIVE\_COMPILATION ]

[ SCHEMABINDING ]

[ EXECUTE AS Clause ]

3) - Trigger on a CREATE, ALTER, DROP, GRANT, DENY,

-- REVOKE or UPDATE statement (DDL Trigger)

CREATE [ OR ALTER ] TRIGGER trigger\_name

ON { ALL SERVER | DATABASE }

[ WITH <ddl\_trigger\_option> [ ,...n ] ]

{ FOR | AFTER } { event\_type | event\_group } [ ,...n ]

AS { sql\_statement [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier > [ ; ] }

<ddl\_trigger\_option> ::=

[ ENCRYPTION ]

[ EXECUTE AS Clause ]