/\*

What is a trigger?

A trigger is a special type of a procedure that 'fires' when an event such as INSERT, DELETE or UPDATE occurs occurs

They are regared as 'event-driven specialized procedures', and as such are stored in and managed by the RDBMS

A trigger fires when we modify the data modification to the associated table by suing one of the events INSERT, DELETE or UPDATE

Triggers are primarily used for referential integrity of data. When creating a trigger we attach it to the table

Triggers can be 'nested' in another trigger thereby firing another trigger

Two typs of triggers in this demo. After and Instead Of.

After triggers fire immediately after the event

Instead Of fires preventing the event from happening

\*/

Use TSQL

Go

-- Select Grades table to put trigger on

SELECT [GRADEID]

,[FNAME]

,[GRADE]

FROM [TSQL].[dbo].[GRADES]

-- Create a simple trigger to print a statement with current date concatinated when an Isert occurs

CREATE TRIGGER Trg\_Grades\_Insert --<< Trigger name

ON Grades --<< attaching the table with a trigger

AFTER INSERT --<< for type of event (either insert, update or delete) Will run immediately AFTER the event has occured

AS

BEGIN

PRINT 'Note that an insert has been made for grades' + ' ' + Convert(varchar (20),GETDATE(), 101) --<< PRINT keyword prints whatever is in single quotes

END

GO

--Insert values in grades table (this modification (event-INSERT) to the attached table will cause the trigger to fire (execute)

INSERT GRADES

VALUES ('Samson', 95)

--View insert via normal table

SELECT \*

FROM [TSQL].[dbo].[GRADES]

-- When you need to alter the trigger, use the ALTER keyword. EX: Changing the text

ALTER TRIGGER [dbo].[Trg\_Grades\_Insert]

ON [dbo].[GRADES]

AFTER INSERT

AS

BEGIN

PRINT 'We have altered the print statement using ALTER' + ' - ' + Convert(varchar (20),GETDATE(), 101)

END

GO

--Insert values in grades table (this modification (event-INSERT) to the attached table will cause the trigger to fire (execute)

INSERT GRADES VALUES ('Lucy', 56)

--View insert via normal table

SELECT \*

FROM [TSQL].[dbo].[GRADES]

--Creating a trigger for all the events (Insert, Delete, Update) --<< Data definition language (DML)

CREATE TRIGGER Trg\_Grades\_Insert\_Delete\_Update --<< Trigger name

ON Grades --<< attaching the table with a trigger

AFTER INSERT,DELETE, UPDATE --<< for type of event (insert, update or delete) wilf fire for any type of modification to the table

AS

BEGIN

PRINT 'You either had an insert, delete or an update against your table' + ' ' + Convert(varchar (20),GETDATE(), 101) --<< PRINT keyword prints whatever is in single quotes

END

GO

--Inserting, updating and deleting from the Grades table

INSERT GRADES VALUES ('Jacqueline', 87)

--View insert via normal table

SELECT \*

FROM [TSQL].[dbo].[GRADES]

UPDATE GRADES

SET GRADE = 91

WHERE FNAME = 'JACQUELINE'

DELETE GRADES

WHERE FNAME = 'JACQUELINE'

--View insert via normal table

SELECT \*

FROM [TSQL].[dbo].[GRADES]

--Using a Instead Of trigger. When using this trigger, note that no data has been inserted by the insert command!!

CREATE TRIGGER Trg\_Grades\_Insert\_Delete\_Updates --<< Trigger name

ON Grades --<< attaching the table with a trigger

INSTEAD OF INSERT, UPDATE --<< for type of event (insert, update or delete) wilL fire for any type of modification to the table

AS

BEGIN

RAISERROR ('You either had an insert, delete or an update against your table',16,1) --<< PRINT keyword prints whatever is in single quotes

END

GO

INSERT GRADES VALUES ('Jacqueline', 87) --<< this insert 'failed' to insert a record because of the Instead Of trigger

Select \* from GRADES

--Delete the trigger with the Drop keyword

DROP TRIGGER [dbo].[Trg\_Grades\_Insert]

GO

DROP TRIGGER Trg\_Grades\_Insert\_Delete\_Update

GO

DROP TRIGGER Trg\_Grades\_Insert\_Delete\_Updates

DROP TRIGGER Trg\_Grades\_Inserted\_Table

GO

DROP Trg\_Grades\_Delete\_Table

GO

/\*

Two system controlled tables called Inserted table and Deleted table is where all the records of inserts, deletes and updates are recorded by the system

you CANNOT directly query against the tables via a select statement as they are embedded in the system memory. However, you can see the results of the

two inserted and deleted tables when 'encapsulated' in a trigger

\*/

--Inserting a new record in the INSERTED table can be viewed using the trigger

CREATE TRIGGER Trg\_Grades\_Inserted\_Table --<< Trigger name

ON Grades --<< attaching the table with a trigger

AFTER INSERT

AS

BEGIN

SELECT \* FROM INSERTED --<< Select statement against the Inserted table (Runnning the Select statement itself causes an error)

END

GO

-- Insert a record

INSERT GRADES VALUES ('Jackie', 100) --Note that when I run this insert statement, I see the result of the insert statement

--Deleteing a record in the DELETED table can be viewed using the trigger

SELECT \* FROM GRADES --<< 13 RECORDS

CREATE TRIGGER Trg\_Grades\_Delete\_Table --<< Trigger name

ON Grades --<< attaching the table with a trigger

AFTER DELETE

AS

BEGIN

SELECT \* FROM DELETED --<< Select statement against the Inserted table (Runnning the Select statement itself causes an error)

END

GO

--Record Jackquline is deleted and is shown when running the trigger

DELETE GRADES

WHERE FNAME = 'Jackie'

SELECT \* FROM GRADES --<< 12 RECORDS