TRIGGER INSTEAD OF (VIEW)

Nguyen Minh Dao

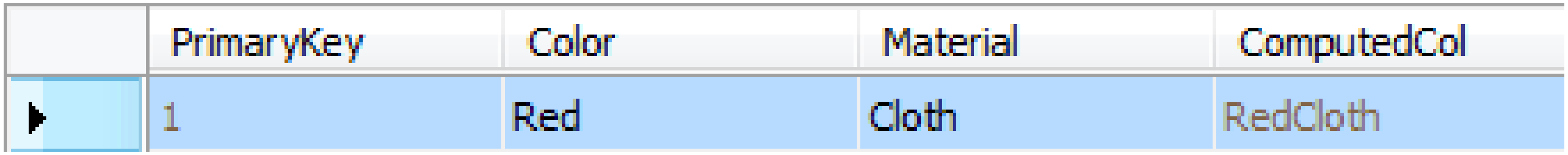
INSTEAD OF INSERT Triggers

```
CREATE TABLE BaseTable
  (PrimaryKey     int PRIMARY KEY IDENTITY(1,1),
   Color          nvarchar(10) NOT NULL,
   Material       nvarchar(10) NOT NULL,
   ComputedCol AS (Color + Material)
  )
GO

--Create a view that contains all columns from the base table.
CREATE VIEW InsteadView
AS SELECT PrimaryKey, Color, Material, ComputedCol
FROM BaseTable
GO
```

INSTEAD OF INSERT Triggers

- --A correct INSERT statement that skips the PrimaryKey and ComputedCol columns.
  - INSERT INTO **BaseTable** (Color, Material)
  - VALUES (N'Red', N'Cloth')
- --View the results of the INSERT statement.
  - SELECT PrimaryKey, Color, Material, ComputedCol
  - FROM **BaseTable**



	PrimaryKey	Color	Material	ComputedCol
▶	1	Red	Cloth	RedCloth

INSTEAD OF INSERT Triggers

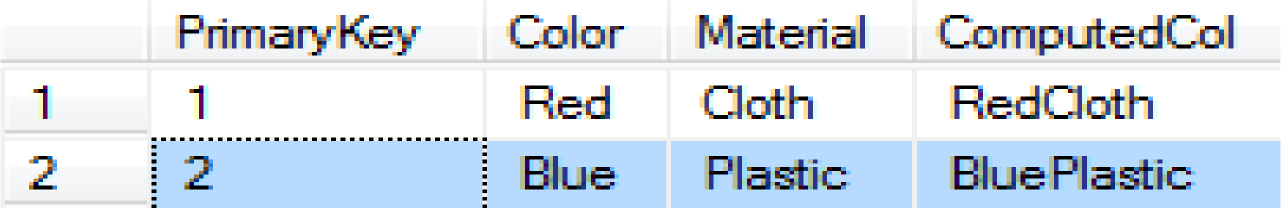
- --An incorrect statement that tries to supply a value for the PrimaryKey and ComputedCol columns.
  - INSERT INTO BaseTable
  - VALUES (**2**, N'Green', N'Wood', **N'GreenWood'**)
- Error!!!
  - *Msg 8101, Level 16, State 1, Line 3*
  - *An **explicit value** for **the identity column** in table **'BaseTable'** can only be specified when a column list is used and IDENTITY_INSERT is ON.*

INSTEAD OF INSERT Triggers

- --Create an INSTEAD OF INSERT trigger on the view.
  - CREATE TRIGGER **InsteadTrigger** on **InsteadView**
  - **INSTEAD OF INSERT**
  - AS
    - BEGIN
    -   --Build an INSERT statement ignoring inserted.PrimaryKey and
    -   --inserted.ComputedCol.
    -   INSERT INTO BaseTable
    -          SELECT Color, Material
    -          FROM inserted
    - END
    - GO

INSTEAD OF INSERT Triggers

- --A correct INSERT statement supplying dummy values for the PrimaryKey and ComputedCol columns.
  - *INSERT INTO InsteadView (PrimaryKey, Color, Material, ComputedCol) VALUES (999, N'Blue', N'Plastic', N'XXXXXX')*
- --View the results of the INSERT statement.
  - *SELECT PrimaryKey, Color, Material, ComputedCol*
  - *FROM InsteadView*



	PrimaryKey	Color	Material	ComputedCol
1	1	Red	Cloth	RedCloth
2	2	Blue	Plastic	BluePlastic

Example

- USE **AdventureWorks**
- GO
- -- Create table for employees
  - CREATE TABLE **Employees**
  - (EmpCode VARCHAR(8) PRIMARY KEY, Name VARCHAR(50) NOT NULL,
  - Designation VARCHAR(50) NOT NULL, QualificationCode TINYINT,
  - Deleted BIT NOT NULL DEFAULT 0)
- GO
- -- Create look up table for employees qualification
  - CREATE TABLE **Lib_Qualification**
  - (QualificationCode TINYINT PRIMARY KEY, Qualification VARCHAR(20) NOT NULL)
- GO

Example

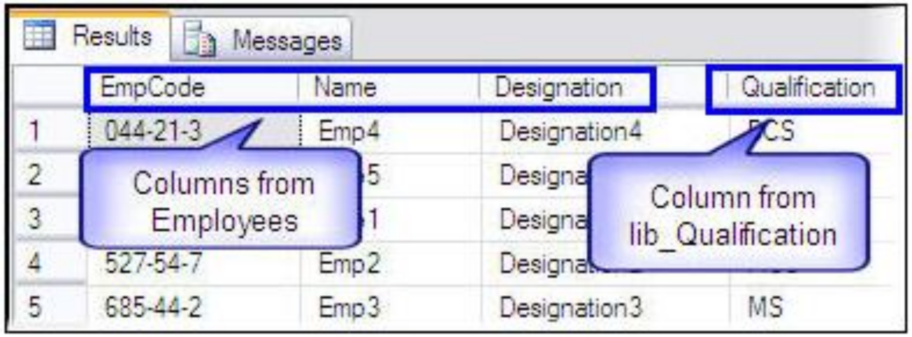
- -- Add constraint to lib_qualification
  - ALTER TABLE **dbo.Lib_Qualification** ADD CONSTRAINT
  - FK_Lib_Qualification_Lib_Qualification FOREIGN KEY
  - (QualificationCode) REFERENCES dbo.Lib_Qualification
  - (QualificationCode) ON UPDATE NO ACTION ON DELETE NO ACTION
- GO
- -- Add constraint to employees
  - ALTER TABLE **dbo.EMPLOYEES** ADD CONSTRAINT
  - FK_EMPLOYEES_Lib_Qualification FOREIGN KEY
  - (QualificationCode) REFERENCES dbo.Lib_Qualification
  - (QualificationCode) ON UPDATE NO ACTION ON DELETE NO ACTION
- GO

Example

- -- Insert data into lib_qualification table
  - Insert into lib_qualification VALUES (1, 'MS')
  - Insert into lib_qualification VALUES (2, 'MCS')
  - Insert into lib_qualification VALUES (3, 'BCS')
  - Insert into lib_qualification VALUES (4, 'MBA')
- GO
- -- Insert data into employees table
  - Insert into Employees VALUES ('405-21-1' ,'Emp1' ,'Designation1' ,1 ,0)
  - Insert into Employees VALUES ('527-54-7' ,'Emp2' ,'Designation2' ,2 ,0)
  - Insert into Employees VALUES ('685-44-2' ,'Emp3' ,'Designation3' ,1 ,0)
  - Insert into Employees VALUES ('044-21-3' ,'Emp4' ,'Designation4' ,3 ,0)
  - Insert into Employees VALUES ('142-21-9' ,'Emp5' ,'Designation5' ,2 ,0)
- GO

Example

- -- Create view by two base tables
  - CREATE VIEW **vw_EmpQualification** AS
    - SELECT EmpCode, Name, Designation, Qualification
    - FROM **employees E inner join lib_qualification Q**
    - ON **E.qualificationCOde = Q.QualificationCode** WHERE deleted = 0
  - GO
    - Select * from **vw_EmpQualification**
  - GO



	EmpCode	Name	Designation	Qualification
1	044-21-3	Emp4	Designation4	CS
2	[illegible]	5	Designa	[illegible]
3	[illegible]	1	Designa	[illegible]
4	527-54-7	Emp2	Designa	[illegible]
5	685-44-2	Emp3	Designation3	MS

Example

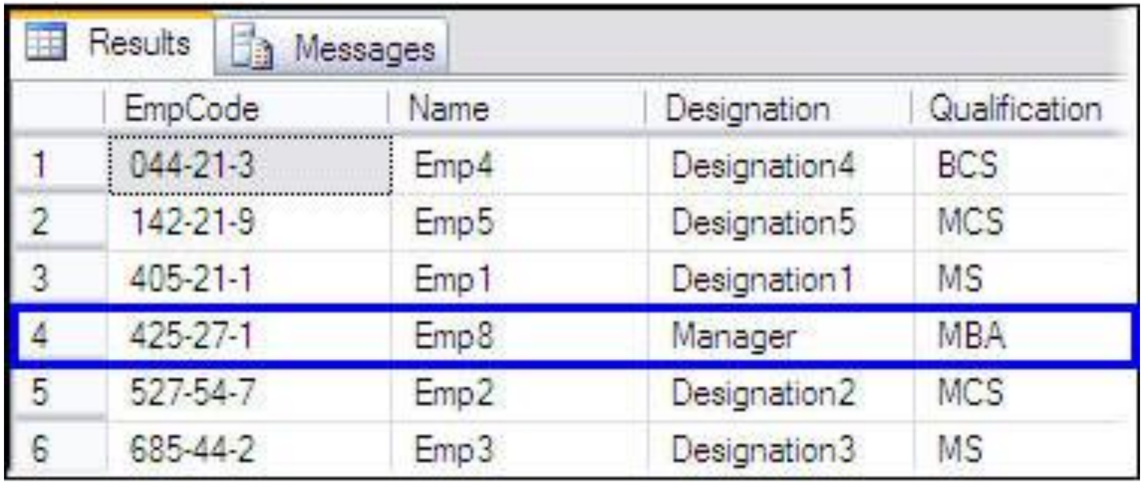
- **INSTEAD OF INSERT Trigger for Insert operation**
- Our view is comprised of two base tables. If someone tries to insert values using the view the following error will be generated and the insert will fail.
  - INSERT INTO **vw_EmpQualification** VALUES ('425-27-1', 'Emp8','Manager','MBA')
    - Msg 4405, Level 16, State 1, Line 1
      View or function 'vw_EmpQualification' is not updatable because the modification affects multiple base tables.

Example

```
CREATE TRIGGER INSTEADOF_TR_I_EmpQualification ON vw_EmpQualification
INSTEAD OF INSERT AS
    BEGIN
    DECLARE @Code TINYINT
    SELECT @Code = qualificationCode  FROM lib_Qualification L INNER JOIN
    INSERTED I
    ON L.qualification = I.qualification
        IF (@code is NULL )
        BEGIN
        RAISERROR (N'The provided qualification does not exist in qualification library‘,16, 1)
        RETURN
        END
    INSERT INTO employees (empcode, name,
    designation,qualificationCode,deleted)
    SELECT empcode, name, designation, @code, 0  FROM inserted
    END
GO
```

Example

- -- Insert data in view
  - INSERT INTO vw_EmpQualification VALUES ('425-27-1', 'Emp8','Manager','MBA')
- -- To confirm the data insertion
  - SELECT * FROM vw_EmpQualification



	EmpCode	Name	Designation	Qualification
1	044-21-3	Emp4	Designation4	BCS
2	142-21-9	Emp5	Designation5	MCS
3	405-21-1	Emp1	Designation1	MS
4	425-27-1	Emp8	Manager	MBA
5	527-54-7	Emp2	Designation2	MCS
6	685-44-2	Emp3	Designation3	MS

INSTEAD OF UPDATE Triggers

- There may be several scenarios where using INSTEAD of triggers can solve this problem. In the case of views with multiple base tables, you may only issue update statements that affect a single base table at a time.
- If any update statement on our view affects multiple base tables at a time then the following error would be generated.

INSTEAD OF UPDATE Triggers

- -- Update data in view
  - UPDATE **vw_EmpQualification**
  - SET designation = 'Designation4 Updated', Qualification = 'MCS'
  - WHERE empcode = '044-21-3’
- -- Error
  - Msg 4405, Level 16, State 1, Line 2
  - View or function 'vw_EmpQualification' is not updatable because the modification affects multiple base tables.

INSTEAD OF UPDATE Triggers

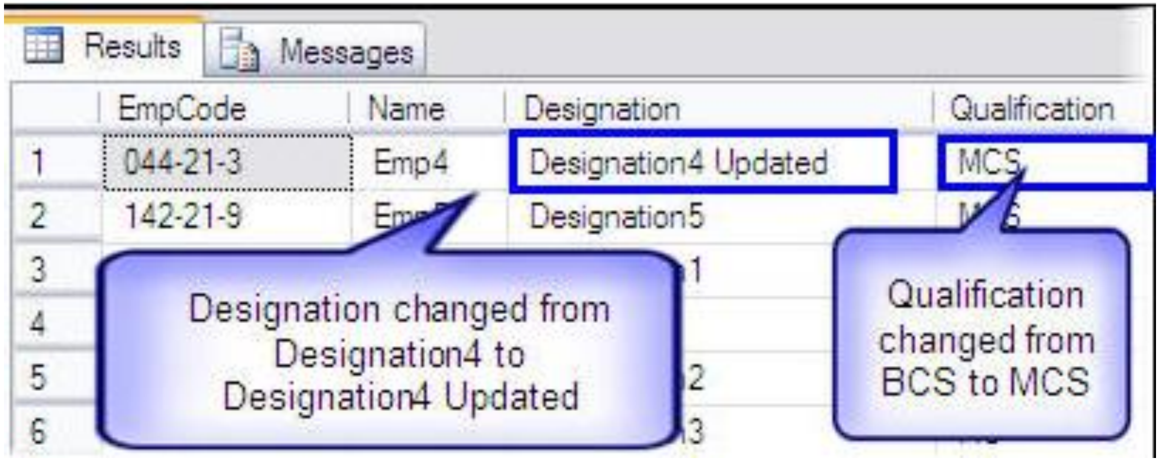
- CREATE **TRIGGER INSTEADOF_TR_U_EmpQualification** ON **vw_EmpQualification**
- INSTEAD OF UPDATE AS
- BEGIN
  - IF (UPDATE(qualification)) -- If qualification is updated
    - BEGIN
    - DECLARE @code TINYINT
    - UPDATE employees
    - SET @code = L.qualificationcode
    - FROM lib_qualification L INNER JOIN inserted I
    - ON L.qualification = I.qualification
      - IF (@code is NULL)
        - BEGIN
          - RAISERROR (N'The provided qualification does not exist in qualification library',
          - 16, 1)
          - RETURN
        - END

INSTEAD OF UPDATE Triggers

  - UPDATE employees
    - SET qualificationCode = @code
    - FROM inserted I INNER JOIN employees E ON I.empcode = E.empcode
- END
- IF (UPDATE(EmpCode)) -- If employee code is updated
  - BEGIN
    - RAISERROR (N'You can not edit employee code, Transaction has been failed', 16, 1)
    - RETURN
  - END
- IF (UPDATE(name)) -- If name is updated
  - BEGIN
    - UPDATE employees
    - SET name = I.name
    - FROM inserted I INNER JOIN employees E ON I.empcode = E.empcode
    - WHERE E.empcode = I.empcode
  - END

INSTEAD OF UPDATE Triggers

- -- Update data in view
  - UPDATE **vw_EmpQualification**
    - SET designation = 'Designation4 Updated', Qualification = 'MCS'
    - WHERE empcode = '044-21-3'
- -- To confirm the data update
  - SELECT * FROM vw_EmpQualification



	EmpCode	Name	Designation	Qualification
1	044-21-3	Emp4	Designation4 Updated	MCS
2	142-21-9	Em	Designation5	M S

Designation changed from Designation4 to Designation4 Updated

Qualification changed from BCS to MCS

INSTEAD OF trigger for delete

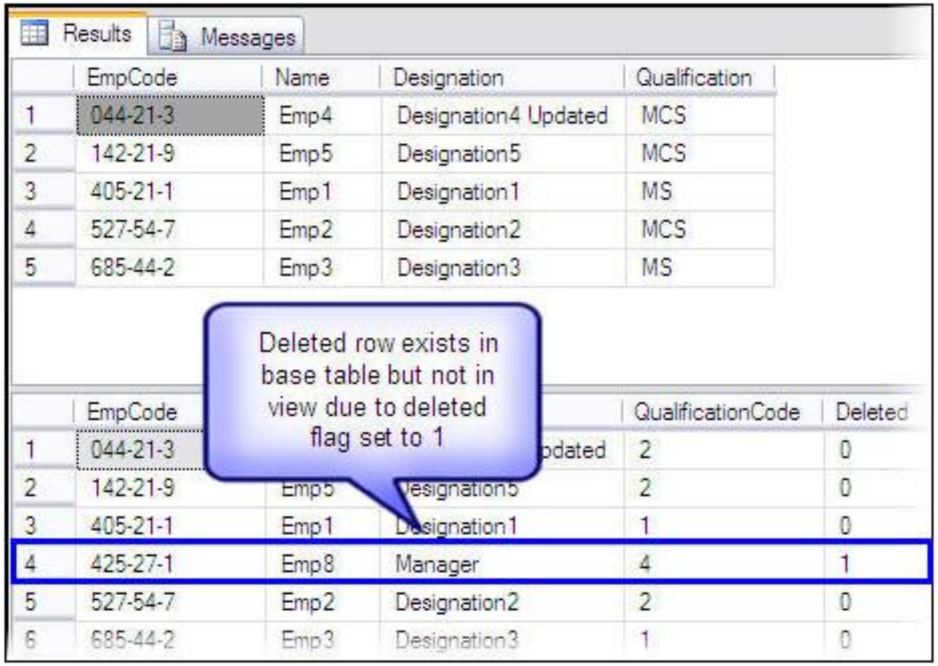
- INSTEAD OF trigger may be attached for delete operations.
- In our case we are required that when rows are deleted through the view, a deleted flag in the table should be marked "1" for those rows, but rows should not actually be deleted.
- Such rows may be deleted in bulk later at specified time if needed. For this we may create the following INSTEAD OF DELETE trigger.

INSTEAD OF trigger for delete

- CREATE TRIGGER **INSTEADOF_TR_D_EmpQualification**
- ON **vw_EmpQualification**
- INSTEAD OF DELETE AS
  - BEGIN
    - update employees
    - set deleted = 1
    - where empcode in (select empcode from deleted)
  - END
- GO

INSTEAD OF trigger for delete

- -- Delete data in view
  - DELETE FROM vw_EmpQualification
    - WHERE Designation = 'Manager‘
- -- To confirm the data update
  - SELECT * FROM vw_EmpQualification
  - SELECT * FROM Employees



	EmpCode	Name	Designation	Qualification
1	044-21-3	Emp4	Designation4 Updated	MCS
2	142-21-9	Emp5	Designation5	MCS
3	405-21-1	Emp1	Designation1	MS
4	527-54-7	Emp2	Designation2	MCS
5	685-44-2	Emp3	Designation3	MS

Deleted row exists in base table but not in view due to deleted flag set to 1

	EmpCode			QualificationCode	Deleted
1	044-21-3		pdated	2	0
2	142-21-9	Emp5	esignation5	2	0
3	405-21-1	Emp1	Designation1	1	0
4	425-27-1	Emp8	Manager	4	1
5	527-54-7	Emp2	Designation2	2	0
6	685-44-2	Emp3	Designation3	1	0

Thank You!