

1:

```
/* Point a)
Создаем таблицу Production.ProductModelHst,
которая будет хранить информацию об изменениях в таблице Production.ProductModel.
Поля, которые присутствуют в таблице:
ID – первичный ключ IDENTITY(1,1);
Action – совершенное действие (insert, update или delete);
ModifiedDate – дата и время, когда была совершена операция;
SourceID – первичный ключ исходной таблицы;
UserName – имя пользователя, совершившего операцию.
*/
CREATE TABLE Production.ProductModelHst
(
    ID [INT] IDENTITY (1,1) NOT NULL,
    [Action] VARCHAR(10) NOT NULL
        CHECK ([Action] IN ('insert', 'update', 'delete')),
    ModifiedDate DATETIME NOT NULL,
    SourceID [INT],
    UserName VARCHAR(256) NOT NULL
);
```

0 %

Результаты Сообщения

ID	Action	ModifiedDate	SourceID	UserName
----	--------	--------------	----------	----------

	Имя столбца	Тип данных	Разрешить знач...
►	ID	int	<input type="checkbox"/>
	Action	varchar(10)	<input type="checkbox"/>
	ModifiedDate	datetime	<input type="checkbox"/>
	SourceID	int	<input checked="" type="checkbox"/>
	UserName	varchar(256)	<input type="checkbox"/>
			<input type="checkbox"/>

```

/* Point b)
Создаем один AFTER триггер для трех операций INSERT, UPDATE, DELETE для таблицы Production.ProductModel.
Триггер заполняет таблицу Production.ProductModelHst
с указанием типа операции в поле Action в зависимости от оператора, вызвавшего триггер.
*/
CREATE
TRIGGER onProductModelChanged
ON Production.ProductModel
AFTER
INSERT, UPDATE, DELETE AS
BEGIN
    DECLARE @actionType varchar(20);
    DECLARE @sourceID int;
    IF EXISTS(SELECT * FROM inserted)
    BEGIN
        SELECT @sourceID = ProductModelID
        FROM inserted;
        IF EXISTS(SELECT * FROM deleted)
        SELECT @actionType = 'update';
        ELSE
        SELECT @actionType = 'insert';
    END;
    ELSE
    BEGIN
        IF EXISTS(SELECT * FROM deleted)
        SELECT @actionType = 'delete';
        SELECT @sourceID = ProductModelID
        FROM deleted;
    END;
    INSERT INTO Production.ProductModelHst([Action], ModifiedDate, SourceID, UserName)
    VALUES (@actionType, GETDATE(), @sourceID, USER_NAME());
END;
GO

```

% <

Сообщения

Выполнение команд успешно завершено.

- [-] Production.ProductModel
  - [+] Столбцы
  - [+] Ключи
  - [+] Ограничения
  - [-] Триггеры
    - onProductModelChanged
  - [+] Индексы
  - [+] Статистика

```

USE [AdventureWorks2012]
GO
/***** Object: Trigger [Production].[onProductModelChanged]    Script Date: 15.10.2019 18:08:26 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER
    TRIGGER [Production].[onProductModelChanged]
    ON [Production].[ProductModel]
    AFTER
        INSERT, UPDATE, DELETE AS
    BEGIN
        DECLARE @actionType varchar(20);
        DECLARE @sourceID int;
        IF EXISTS(SELECT * FROM inserted)
            BEGIN
                SELECT @sourceID = ProductModelID
                FROM inserted;
                IF EXISTS(SELECT * FROM deleted)
                    SELECT @actionType = 'update';
                ELSE
                    SELECT @actionType = 'insert';
            END;
        ELSE
            BEGIN
                IF EXISTS(SELECT * FROM deleted)
                    SELECT @actionType = 'delete';
                SELECT @sourceID = ProductModelID
                FROM deleted;
            END;
        INSERT INTO Production.ProductModelHst([Action], ModifiedDate, SourceID, UserName)
        VALUES (@actionType, GETDATE(), @sourceID, USER_NAME());
    END;

```

```

/* Point c)
   Создаем представление VIEW, отображающее все поля таблицы Production.ProductModel.
*/
CREATE VIEW ProductModelView AS
    SELECT *
    FROM Production.ProductModel;
GO

```

100 %

Сообщения

Выполнение команд успешно завершено.

Представления

Системные представления

dbo.ProductModelView

Столбцы

ProductModelID (int, He NULL)

Name (Name(nvarchar(50)), He NULL)

CatalogDescription (XML(Production.ProductDescriptionSchemaCollection), NULL)

Instructions (XML(Production.ManulInstructionsSchemaCollection), NULL)

rowguid (uniqueidentifier, He NULL)

ModifiedDate (datetime, He NULL)

Триггеры

Индексы

Статистика

```

/* Point d)
Вставляю новую строку в Production.ProductModel через представление.
Обновляю вставленную строку.
Удаляю вставленную строку.
Все три операции отображены в Production.ProductModelHst.
*/
INSERT
    INTO Production.ProductModel(Name)
    VALUES ('Roma');
UPDATE Production.ProductModel
    SET Name = 'Roman'
    WHERE Name = 'Roma';
DELETE
    FROM Production.ProductModel
    WHERE Name = 'Roman';
SELECT *
    FROM Production.ProductModelHst;

```

100 %

Результаты Сообщения

	ID	Action	ModifiedDate	Source...	UserNa...
1	1	insert	2019-10-16 12:55:41.480	129	dbo
2	2	update	2019-10-16 12:55:41.483	129	dbo
3	3	delete	2019-10-16 12:55:41.487	129	dbo

2:

```

/* Point a)
    Создаю представление VIEW, отображающее данные из таблиц
    Production.ProductModel,
    Production.ProductModelProductDescriptionCulture,
    Production.Culture и
    Production.ProductDescription.
    Делаю невозможным просмотр исходного кода представления.
    Создаю уникальный кластерный индекс в представлении по полям ProductModelID, CultureID.
*/
CREATE VIEW dbo.ProductModelClusterView
    WITH ENCRYPTION, SCHEMABINDING
AS
SELECT C.CultureID,
       C.Name AS C_Name,
       C.ModifiedDate AS C_ModifiedDate,
       PM.CatalogDescription,
       PM.Instructions,
       PM.Name AS PM_Name,
       PM.ProductModelID,
       PM.ModifiedDate AS PM_ModifiedDate,
       PD.Description,
       PD.ProductDescriptionID,
       PD.rowguid,
       PD.ModifiedDate AS PD_ModifiedDate,
       PMPDC.ModifiedDate AS PMPDC_ModifiedDate
FROM Production.ProductModel AS PM
     JOIN Production.ProductModelProductDescriptionCulture AS PMPDC
         ON PM.ProductModelID = PMPDC.ProductModelID
     JOIN Production.Culture AS C
         ON C.CultureID = PMPDC.CultureID
     JOIN Production.ProductDescription AS PD
         ON PD.ProductDescriptionID = PMPDC.ProductDescriptionID;
GO

CREATE UNIQUE CLUSTERED INDEX PRODUCT_MODEL_INDX
    ON dbo.ProductModelClusterView (ProductModelID, CultureID);

```

3 %

Сообщения

Выполнение команд успешно завершено.

- [-] Представления
  - [+] Системные представления
  - [-] dbo.ProductModelClusterView
    - [-] Столбцы
      - [+] CultureID (nchar(6), He NULL)
      - [+] C\_Name (Name(nvarchar(50)), He NULL)
      - [+] C\_ModifiedDate (datetime, He NULL)
      - [+] CatalogDescription (XML(Production.ProductDescriptionSchemaCollection), NULL)
      - [+] Instructions (XML(Production.ManulInstructionsSchemaCollection), NULL)
      - [+] PM\_Name (Name(nvarchar(50)), He NULL)
      - [+] ProductModelID (int, He NULL)
      - [+] PM\_ModifiedDate (datetime, He NULL)
      - [+] Description (nvarchar(400), He NULL)
      - [+] ProductDescriptionID (int, He NULL)
      - [+] rowguid (uniqueidentifier, He NULL)
      - [+] PD\_ModifiedDate (datetime, He NULL)
      - [+] PMPDC\_ModifiedDate (datetime, He NULL)
    - [+] Триггеры
    - [+] Индексы
    - [+] Статистика

```

/* Point b)
Создаю три INSTEAD OF триггера для представления на операции INSERT, UPDATE, DELETE.
Каждый триггер выполняет соответствующие операции в таблицах
    Production.ProductModel,
    Production.ProductModelProductDescriptionCulture,
    Production.Culture и
    Production.ProductDescription.
Обновляю не происходит в таблице Production.ProductModelProductDescriptionCulture.
Удаление строк из таблиц
    Production.ProductModel,
    Production.Culture и
    Production.ProductDescription
произвожу только в том случае, если удаляемые строки больше не ссылаются на Production.ProductModelProductDescriptionCulture.
*/
CREATE TRIGGER onInsertIntoProductModelView
ON dbo.ProductModelClusterView
INSTEAD OF
INSERT
AS
BEGIN
    INSERT INTO Production.Culture(CultureID, Name)
    SELECT CultureID, C_Name
    FROM inserted;
    INSERT INTO Production.ProductModel(Name)
    SELECT PM_Name
    FROM inserted;
    INSERT INTO Production.ProductDescription([Description])
    SELECT [Description]
    FROM inserted;
    INSERT INTO Production.ProductModelProductDescriptionCulture(CultureID, ProductModelID, ProductDescriptionID)
    VALUES ((SELECT CultureID FROM inserted),
    IDENT_CURRENT('Production.ProductModel'),
    IDENT_CURRENT('Production.ProductDescription'));
END;
GO

```

```

CREATE TRIGGER onUpdateProductModelView
ON dbo.ProductModelClusterView
INSTEAD OF
UPDATE
AS
BEGIN
    UPDATE Production.Culture
    SET Name = (SELECT C_Name FROM inserted),
        ModifiedDate = GETDATE()
    WHERE Name = (SELECT C_Name FROM deleted);
    UPDATE Production.ProductModel
    SET Name = (SELECT PM_Name FROM inserted),
        ModifiedDate = GETDATE()
    WHERE Name = (SELECT PM_Name FROM deleted);
    UPDATE Production.ProductDescription
    SET [Description] = (SELECT [Description] FROM inserted),
        ModifiedDate = GETDATE()
    WHERE [Description] = (SELECT [Description] FROM deleted);
END;
GO

```

11 %

Сообщения

Выполнение команд успешно завершено.

```

CREATE TRIGGER onDeleteFromProductModelView
ON dbo.ProductModelClusterView
INSTEAD OF
DELETE
AS
BEGIN
    IF (SELECT CultureID FROM deleted) NOT IN
        (SELECT CultureID FROM Production.ProductModelProductDescriptionCulture)
    BEGIN
        DELETE
        FROM Production.Culture
        WHERE CultureID = (SELECT CultureID FROM deleted);
    END;

    IF (SELECT ProductDescriptionID FROM deleted) NOT IN
        (SELECT ProductDescriptionID FROM Production.ProductModelProductDescriptionCulture)
    BEGIN
        DELETE
        FROM Production.ProductDescription
        WHERE ProductDescriptionID = (SELECT ProductDescriptionID FROM deleted);
    END;





    IF (SELECT ProductModelID FROM deleted) NOT IN
        (SELECT ProductModelID FROM Production.ProductModelProductDescriptionCulture)
    BEGIN
        DELETE
        FROM Production.ProductModel
        WHERE ProductModelID = (SELECT ProductModelID FROM deleted);
    END;
END;

```

% <

Сообщения

Выполнение команд успешно завершено.

- [-] Представления
  - [+] Системные представления
  - [-]  dbo.ProductModelClusterView
    - [+] Столбцы
    - [-] Триггеры
      -  onDeleteFromProductModelView
      -  onInsertIntoProductModelView
      -  onUpdateProductModelView
  - [+] Индексы
  - [+] Статистика

```

-- Вставляем новую строку в представление, указав новые данные для
-- ProductModel,
-- Culture и
-- ProductDescription.
-- Триггер добавляет новые строки в таблицы
-- Production.ProductModel,
-- Production.ProductModelProductDescriptionCulture,
-- Production.Culture и
-- Production.ProductDescription.
-- Обновляем вставленные строки через представление.
-- Удаляем строку.
*/

```

```

-- INSERT
-- INTO dbo.ProductModelClusterView(CultureID, C_Name, PH_Name, [Description])
-- VALUES ('ru', 'Russian', 'Roga I Kopita', 'Have fun');

-- SELECT * FROM Production.ProductModel;
-- SELECT * FROM Production.ProductModelProductDescriptionCulture;
-- SELECT * FROM Production.Culture;
-- SELECT * FROM Production.ProductDescription;

```

91 %

Результаты Свойства

ProductModel	Name	CatalogDescription	Instructions	rowguid	ModifiedDate
128	128	Rear Brakes	NULL	71D47AFD-DA3A-43F1-83AD-69C71F96EF33	2007-06-01 00:00:00.000
129	131	Roga I Kopita	NULL	801AC1B0-126C-4DAF-A331-FAEA07B5E8FF	2019-10-16 19:32:54.853

ProductModel	ProductDescription	Culture	ModifiedDate	
763	131	2011	ru	2019-10-16 19:32:54.853

CultureID	Name	ModifiedDate	
7	ru	Russian	2019-10-16 19:32:54.853

ProductDescription	Description	rowguid	ModifiedDate	
762	2010	宽连杆设计。	46541761-9053-4B6F-80F0-68AAE2695A8D	2007-06-01 00:00:00.000
763	2011	Have fun	960CF3BF-2462-4B7B-8698-46405E9A2C5	2019-10-16 19:32:54.853

```

-- UPDATE dbo.ProductModelClusterView
-- SET C_Name = 'ModifiedRussian',
-- PH_Name = 'Super Roga I Kopita',
-- [Description] = 'Have crazy fun'
-- WHERE CultureID = 'ru'
-- AND ProductModelID = IDENT_CURRENT('Production.ProductModel')
-- AND ProductDescriptionID = IDENT_CURRENT('Production.ProductDescription');

-- SELECT * FROM Production.ProductModel;
-- SELECT * FROM Production.ProductModelProductDescriptionCulture;
-- SELECT * FROM Production.Culture;
-- SELECT * FROM Production.ProductDescription;

```

91 %

Результаты | Свойства

ProductModel	Name	CatalogDescription	Instructions	rowguid	ModifiedDate
128	128	Rear Brakes	NULL	71D47AFD-DA3A-43F1-83AD-69C71F96EF33	2007-06-01 00:00:00.000
129	131	Super Roga I Kopita	NULL	801AC1B0-126C-4DAF-A331-FAEA07B5E8FF	2019-10-16 19:38:59.333

ProductModel	ProductDescription	Culture	ModifiedDate	
763	131	2011	ru	2019-10-16 19:32:54.853

CultureID	Name	ModifiedDate	
7	ru	ModifiedRussian	2019-10-16 19:38:59.333

ProductDescription	Description	rowguid	ModifiedDate	
762	2010	宽连杆设计。	46541761-9053-4B6F-80F0-68AAE2695A8D	2007-06-01 00:00:00.000
763	2011	Have crazy fun	960CF3BF-2462-4B7B-8698-46405E9A2C5	2019-10-16 19:38:59.333

```

-- DELETE
-- FROM dbo.ProductModelClusterView
-- WHERE CultureID = 'ru'
-- AND ProductModelID = IDENT_CURRENT('Production.ProductModel')
-- AND ProductDescriptionID = IDENT_CURRENT('Production.ProductDescription');

```

91 %

Сообщения

(строк обработано: 1)