

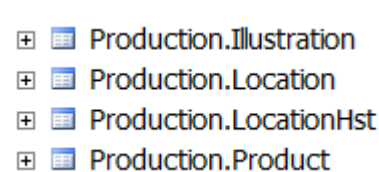
Лабораторная работа №4

Вариант 2

Задание №1

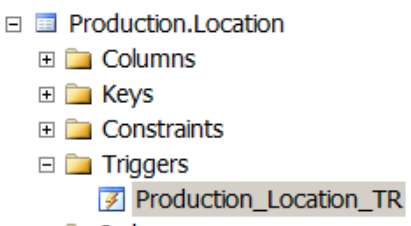
а) Создайте таблицу Production.LocationHst, которая будет хранить информацию об изменениях в таблице Production.Location. Обязательные поля, которые должны присутствовать в таблице: ID — первичный ключ IDENTITY(1,1); Action — совершенное действие (insert, update или delete); ModifiedDate — дата и время, когда была совершена операция; SourceID — первичный ключ исходной таблицы; UserName — имя пользователя, совершившего операцию. Создайте другие поля, если считаете их нужными.

```
CREATE TABLE Production.LocationHst (  
    ID INT IDENTITY(1,1) PRIMARY KEY,  
    Action NVARCHAR(8) NOT NULL,  
    ModifiedDate DATETIME NOT NULL,  
    SourceID INT NOT NULL,  
    UserName NVARCHAR(25) NOT NULL  
);  
GO
```

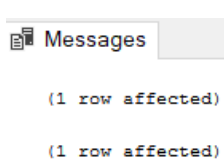


б) Создайте один AFTER триггер для трех операций INSERT, UPDATE, DELETE для таблицы Production.Location. Триггер должен заполнять таблицу Production.LocationHst с указанием типа операции в поле Action в зависимости от оператора, вызвавшего триггер.

```
CREATE TRIGGER Production_Location_TR  
ON Production.Location  
AFTER INSERT, UPDATE, DELETE  
AS  
IF EXISTS (SELECT * FROM inserted) AND EXISTS (SELECT * FROM deleted)  
    INSERT INTO Production.LocationHst  
        SELECT  
            'update',  
            CURRENT_TIMESTAMP,  
            LocationID,  
            CURRENT_USER  
        FROM inserted  
ELSE IF EXISTS (SELECT * FROM inserted)  
    INSERT INTO Production.LocationHst  
        SELECT  
            'insert',  
            CURRENT_TIMESTAMP,  
            LocationID,  
            CURRENT_USER  
        FROM inserted  
ELSE IF EXISTS (SELECT * FROM deleted)  
    INSERT INTO Production.LocationHst  
        SELECT  
            'delete',  
            CURRENT_TIMESTAMP,  
            LocationID,  
            CURRENT_USER  
        FROM deleted;  
GO
```



```
INSERT INTO Production.Location (  
    LocationID,  
    Name,  
    CostRate,  
    Availability,  
    ModifiedDate  
) VALUES (  
    401,  
    'Loca',  
    10,  
    0.6,  
    CURRENT_TIMESTAMP  
);  
GO
```



```

UPDATE Production.Location
    SET Name = 'name'
    WHERE LocationID = 401;
GO

DELETE FROM Production.Location
    WHERE LocationID = 401;
GO

SELECT * FROM Production.LocationHst
    WHERE SourceID = 401;
GO

```

	ID	Action	ModifiedDate	Source...	UserNa...
1	1	insert	2020-11-07 19:24:49.437	401	dbo
2	2	update	2020-11-07 19:24:49.443	401	dbo
3	3	delete	2020-11-07 19:24:49.670	401	dbo

c) Создайте представление VIEW, отображающее все поля таблицы Production.Location.

```

CREATE VIEW LocationView AS
    SELECT * FROM Production.Location;
GO

```

Views
System Views
dbo.LocationView

d) Вставьте новую строку в Production.Location через представление. Обновите вставленную строку. Удалите вставленную строку. Убедитесь, что все три операции отображены в Production.LocationHst.

```

SET IDENTITY_INSERT Production.Location ON;
GO

```

```

INSERT INTO LocationView (
    LocationID,
    Name,
    CostRate,
    Availability,
    ModifiedDate
) VALUES (
    401,
    'loca',
    10,
    0.6,
    CURRENT_TIMESTAMP
);
GO

```

```

SET IDENTITY_INSERT Production.Location OFF;
GO

```

Results Messages
(1 row(s) affected)
(1 row(s) affected)

```

SELECT * FROM LocationView
    WHERE LocationID = 401;
GO

```

	Location...	Name	CostR...	Availabil...	ModifiedDate
1	401	loca	10,00	0.60	2020-11-08 16:21:25.387

```

UPDATE LocationView
    SET Name = 'Loca'
    WHERE LocationID = 401;
GO

```

```

DELETE FROM LocationView
    WHERE LocationID = 401;
GO

```

Results Messages
(1 row(s) affected)
(1 row(s) affected)
(1 row(s) affected)
(1 row(s) affected)

```
SELECT * FROM Production.LocationHist  
WHERE SourceID = 401;  
GO
```

	ID	Action	ModifiedDate	Source...	UserNa...
1	1	insert	2020-11-08 16:21:24.520	401	dbo
2	2	update	2020-11-08 16:21:24.703	401	dbo
3	3	delete	2020-11-08 16:21:24.837	401	dbo
4	4	insert	2020-11-08 16:21:25.390	401	dbo
5	5	update	2020-11-08 16:21:25.510	401	dbo

Задание №2

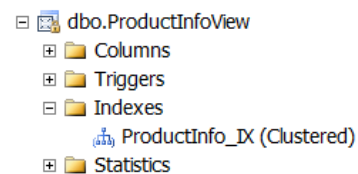
а) Создайте представление VIEW, отображающее данные из таблиц Production.Location и Production.ProductInventory, а также Name из таблицы Production.Product. Сделайте невозможным просмотр исходного кода представления. Создайте уникальный кластерный индекс в представлении по полям LocationID, ProductID.

```
CREATE VIEW ProductInfoView
WITH SCHEMABINDING, ENCRYPTION AS
SELECT
    l.LocationID,
    l.Name AS LocationName,
    l.CostRate,
    l.Availability,
    l.ModifiedDate AS LocationModifiedDate,
    pi.ProductID,
    pi.Shelf,
    pi.Bin,
    pi.Quantity,
    pi.rowguid,
    pi.ModifiedDate AS ProductInventoryModifiedDate,
    p.Name
FROM Production.Location AS l
    INNER JOIN Production.ProductInventory AS pi
ON l.LocationID = pi.LocationID
    INNER JOIN Production.Product AS p
ON pi.ProductID = p.ProductID;
```

GO

```
CREATE UNIQUE CLUSTERED INDEX ProductInfo_IX
ON ProductInfoView(LocationID, ProductID);
```

GO



б) Создайте три INSTEAD OF триггера для представления на операции INSERT, UPDATE, DELETE. Каждый триггер должен выполнять соответствующие операции в таблицах Production.Location и Production.ProductInventory для указанного Product Name. Обновление и удаление строк производите только в таблицах Production.Location и Production.ProductInventory, но не в Production.Product.

```
CREATE TRIGGER ProductInfo_Ins_TR
ON ProductInfoView
INSTEAD OF INSERT AS
BEGIN
    INSERT INTO Production.Location
    SELECT
        LocationName,
        CostRate,
        Availability,
        LocationModifiedDate
    FROM inserted
    INNER JOIN Production.Product AS p
    ON inserted.Name = p.Name;
    INSERT INTO Production.ProductInventory
    SELECT
        p.ProductID,
        l.LocationID,
        Shelf,
        Bin,
        Quantity,
        inserted.rowguid,
        ProductInventoryModifiedDate
    FROM inserted
    INNER JOIN Production.Product AS p
    ON inserted.Name = p.Name
    INNER JOIN Production.Location AS l
    ON inserted.LocationName = l.Name;
END;
```

GO

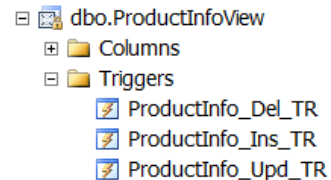
```
CREATE TRIGGER ProductInfo_Del_TR
ON ProductInfoView
INSTEAD OF DELETE AS
BEGIN
    DECLARE @pID INT;
    SELECT @pID = (SELECT ProductID FROM deleted);
    CREATE TABLE #locations (
        LocationID SMALLINT NOT NULL
    );
    INSERT INTO #locations
    SELECT DISTINCT p.LocationID
    FROM Production.ProductInventory AS p
    INNER JOIN deleted
    ON deleted.ProductID = p.ProductID
    WHERE p.LocationID NOT IN (
        SELECT DISTINCT pi.LocationID
        FROM Production.ProductInventory as pi
        WHERE pi.ProductID != @pID
    );
    DELETE p
    FROM Production.ProductInventory AS p
    WHERE p.ProductID = @pID;
    DELETE l
    FROM Production.Location AS l
    WHERE LocationID IN (SELECT * FROM #locations);
END;
```

GO

```

CREATE TRIGGER ProductInfo_Upd_TR
ON ProductInfoView
INSTEAD OF UPDATE AS
BEGIN
    IF UPDATE(LocationID) OR UPDATE(ProductID)
    BEGIN
        RAISERROR ('UPDATE of Primary Key through ProductInfoView is prohibited.', 16, 1);
        ROLLBACK;
    END
    ELSE
    BEGIN
        UPDATE Production.Location
        SET
            Name = inserted.LocationName,
            CostRate = inserted.CostRate,
            Availability = inserted.Availability,
            ModifiedDate = inserted.LocationModifiedDate
        FROM Production.Location AS l
        INNER JOIN inserted
        ON inserted.LocationID = l.LocationID;
        UPDATE Production.ProductInventory
        SET
            Shelf = inserted.Shelf,
            Bin = inserted.Bin,
            Quantity = inserted.Quantity,
            rowguid = inserted.rowguid,
            ModifiedDate = inserted.ProductInventoryModifiedDate
        FROM Production.ProductInventory AS ppi
        INNER JOIN inserted
        ON ppi.ProductID = inserted.ProductID;
    END;
END;
GO

```



с) Вставьте новую строку в представление, указав новые данные для Location и ProductInventory, но для существующего Product (например для 'Adjustable Race'). Триггер должен добавить новые строки в таблицы Production.Location и Production.ProductInventory для указанного Product Name. Обновите вставленные строки через представление. Удалите строки.

```

INSERT INTO ProductInfoView (
    LocationName,
    CostRate,
    Availability,
    LocationModifiedDate,
    Shelf,
    Bin,
    Quantity,
    rowguid,
    ProductInventoryModifiedDate,
    Name
) VALUES (
    'Oloe',
    20.1,
    0.5,
    CURRENT_TIMESTAMP,
    'A',
    5,
    445,
    NewID(),
    CURRENT_TIMESTAMP,
    'Lock Nut 5'
);
GO

```

```

SELECT * FROM Production.Location
WHERE Name = 'Oloe';
GO

```

	Location...	Name	CostR...	Availabil...	ModifiedDate
1	408	Oloe	20,10	0.50	2020-11-08 18:22:56.863

```

SELECT TOP 2 * FROM Production.ProductInventory
ORDER BY ModifiedDate DESC;
GO

```

	Product...	Locatio...	Sh...	Bin	Quant...	rowguid	ModifiedDate
1	438	408	A	5	445	83804B82-0289-4223-9D29-3F10326981A3	2020-11-08 18:2
2	316	10	A	6	8	47A24246-6C43-48EB-968F-025738A8A410	2020-11-08 16:3

```

UPDATE ProductInfoView
SET LocationID = 7
WHERE Name = 'Lock Nut 5';
GO

```

Msg 50000, Level 16, State 1, Procedure ProductInfo_Upd_TR, Line 8
 UPDATE of Primary Key through ProductInfoView is prohibited.
 Msg 3609, Level 16, State 1, Line 2
 The transaction ended in the trigger. The batch has been aborted.

```

UPDATE ProductInfoView
SET CostRate = 5.4
WHERE Name = 'Lock Nut 5';
GO

```

```

SELECT ppi.LocationID, p.Name, CostRate
FROM Production.Location
INNER JOIN Production.ProductInventory AS ppi
ON ppi.LocationID = Production.Location.LocationID
INNER JOIN Production.Product AS p
ON p.ProductID = ppi.ProductID
WHERE p.Name = 'Lock Nut 5';
GO

```

	Location...	Name	CostRate
1	1	Lock Nut 5	5,40
2	6	Lock Nut 5	5,40
3	50	Lock Nut 5	5,40
4	404	Lock Nut 5	5,40
5	407	Lock Nut 5	5,40
6	408	Lock Nut 5	5,40

```

UPDATE ProductInfoView
SET Quantity = 7
WHERE Name = 'Lock Nut 5';
GO

```

```

SELECT Name, Quantity
FROM Production.ProductInventory AS ppi
INNER JOIN Production.Product AS p
ON ppi.ProductID = p.ProductID
WHERE Name = 'Lock Nut 5';
GO

```

	Name	Quant...
1	Lock Nut 5	7
2	Lock Nut 5	7
3	Lock Nut 5	7
4	Lock Nut 5	7
5	Lock Nut 5	7
6	Lock Nut 5	7

```

DELETE FROM ProductInfoView
WHERE Name = 'Lock Nut 5';
GO

```

```

SELECT COUNT(*) AS recCount
FROM Production.Location AS l
INNER JOIN Production.ProductInventory AS ppi
ON ppi.LocationID = l.LocationID
INNER JOIN Production.Product AS p
ON p.ProductID = ppi.ProductID
WHERE p.Name = 'Lock Nut 5';
GO

```

Results		Messages	
	recCount		
1	0		