

Лабораторная работа 3

Задание 1

а) Создайте таблицу Production.WorkOrderHst, которая будет хранить информацию об изменениях в таблице Production.WorkOrder.

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.WorkOrderHst
(
    [id] int IDENTITY (1,1) primary key,
    [ACTION] CHAR(6) NOT NULL CHECK (Action IN ('INSERT', 'UPDATE', 'DELETE')),
    [ModifiedDate] DATETIME NOT NULL,
    [SourceID] INT NOT NULL,
    [UserName] VARCHAR(50) NOT NULL
);
GO
```

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

```
CREATE TRIGGER Production.MYTrigger
ON AdventureWorks2012.Production.WorkOrder
AFTER INSERT, UPDATE, DELETE AS
INSERT INTO Production.WorkOrderHst ([ACTION], [ModifiedDate], [SourceID], [UserName])
SELECT CASE
    WHEN inserted.ProductID IS NULL THEN 'DELETE'
    WHEN deleted.ProductID IS NULL THEN 'INSERT'
    ELSE 'UPDATE' END,
    GETDATE(),
    COALESCE(inserted.ProductID, deleted.ProductID),
    USER_NAME()
FROM inserted
FULL OUTER JOIN [deleted] ON inserted.ProductID = deleted.ProductID;
GO;
```

с) Создайте представление VIEW, отображающее все поля таблицы Production.WorkOrder.

```
CREATE VIEW Production.MyView AS SELECT * FROM  
AdventureWorks2012.Production.WorkOrder;  
GO
```

д) Вставьте новую строку в Production.WorkOrder через представление. Обновите вставленную строку. Удалите вставленную строку. Убедитесь, что все три операции отображены в Production.WorkOrderHst.

```
SET IDENTITY_INSERT Production.WorkOrder ON
```

```
INSERT INTO Production.MyView (WorkOrderID, ProductID, OrderQty, ScrappedQty,  
StartDate, EndDate, DueDate, ScrapReasonID)  
VALUES (72599, 329, 39, 0, '2011-10-14 00:00:00.000', '2011-11-24 00:00:00.000', '2020-06-25  
00:00:00.000', null);
```

```
SET IDENTITY_INSERT Production.WorkOrder OFF
```

```
UPDATE Production.MyView SET EndDate = '2020-10-14 00:00:00.000' WHERE WorkOrderID =  
72599;
```

```
DELETE FROM Production.MyView WHERE WorkOrderID = 72599;
```

```
SELECT * FROM Production.WorkOrderHst;  
Go
```

Задание 2

а) Создайте представление VIEW, отображающее данные из таблиц Production.WorkOrder и Production.ScrapReason, а также Name из таблицы Production.Product. Сделайте невозможным просмотр исходного кода представления. Создайте уникальный кластерный индекс в представлении по полю WorkOrderID.

```
CREATE VIEW Production.WorkOrderView  
(  
    [WorkOrderId],  
    [ProductId],  
    [OrderQty],  
    [StockedQty],  
    [ScrappedQty],  
    [ScrapReasonId],  
    [ProductLineName])  
AS SELECT w.WorkOrderID, w.ProductID, w.OrderQty, w.StockedQty, w.ScrappedQty, s.ScrapReasonID, p.Name  
FROM Production.WorkOrder w  
JOIN Production.ScrapReason s ON w.ScrapReasonID = s.ScrapReasonID  
JOIN Production.Product p ON w.ProductID = p.ProductID
```

```

        [StartDate],
        [EndDate],
        [DueDate],
        [ScrapReasonID],
        [ModifiedDate],
        [SRName],
        [SRModifiedDate]
    ) WITH ENCRYPTION, SCHEMABINDING
AS
SELECT [WO].[WorkOrderId],
       [WO].[ProductId],
       [WO].[OrderQty],
       [WO].[StockedQty],
       [WO].[ScrappedQty],
       [WO].[StartDate],
       [WO].[EndDate],
       [WO].[DueDate],
       [WO].[ScrapReasonID],
       [WO].[ModifiedDate],
       [SR].[Name],
       [SR].[ModifiedDate]
FROM Production.WorkOrder AS WO
     JOIN Production.ScrapReason as SR ON WO.ScrapReasonID = SR.ScrapReasonID
GO

CREATE UNIQUE CLUSTERED INDEX [AK_WorkOrderIdView_WorkOrderId] ON
Production.WorkOrderView ([WorkOrderId]);

```

b) Создайте три INSTEAD OF триггера для представления на операции INSERT, UPDATE, DELETE. Каждый триггер должен выполнять соответствующие операции в таблицах Production.WorkOrder и Production.ScrapReason для указанного Product Name. Обновление и удаление строк производите только в таблицах Production.WorkOrder и Production.ScrapReason, но не в Production.Product. В UPDATE триггере не указывайте обновление поля OrderQty для таблицы Production.WorkOrder.

```

CREATE TRIGGER Production.WorkOrderViewInsteadInsertTrigger
ON Production.WorkOrderView
INSTEAD OF INSERT AS
BEGIN
    BEGIN
        INSERT INTO Production.ScrapReason ([Name], [ModifiedDate])

```

```

    SELECT [SRName], [SRModifiedDate]
    from inserted
end;

BEGIN
    INSERT INTO Production.WorkOrder (ProductID, OrderQty, ScrappedQty, StartDate,
    EndDate, DueDate, ScrapReasonID,
        ModifiedDate)
    SELECT [ProductID],
        [OrderQty],
        [ScrappedQty],
        [StartDate],
        [EndDate],
        [DueDate],
        [SR].[ScrapReasonID],
        GETDATE()
    from inserted
        JOIN Production.ScrapReason AS SR
        On SR.Name = inserted.SRName
end;
END;
Go

```

```

CREATE TRIGGER Production.WorkOrderScrapReasonVIEW_Update
ON Production.WorkOrderView
INSTEAD OF UPDATE AS
BEGIN
    UPDATE Production.ScrapReason
    SET Name      = inserted.SRName,
        ModifiedDate = inserted.SRModifiedDate
    FROM inserted
    WHERE Production.ScrapReason.ScrapReasonID = inserted.ScrapReasonID

    UPDATE Production.WorkOrder
    SET DueDate      = inserted.DueDate,
        EndDate      = inserted.EndDate,
        ModifiedDate = inserted.ModifiedDate,
        ScrappedQty  = inserted.ScrappedQty,
        StartDate    = inserted.StartDate
    FROM inserted
    WHERE Production.WorkOrder.ProductID = inserted.ProductID
END;
GO

```

