

УО «Белорусский государственный университет информатики и  
радиоэлектроники»

Кафедра ПОИТ

Отчёт по лабораторной работе № 4

Базы данных

Вариант 4

Выполнил:

Бобко А.В.

751001

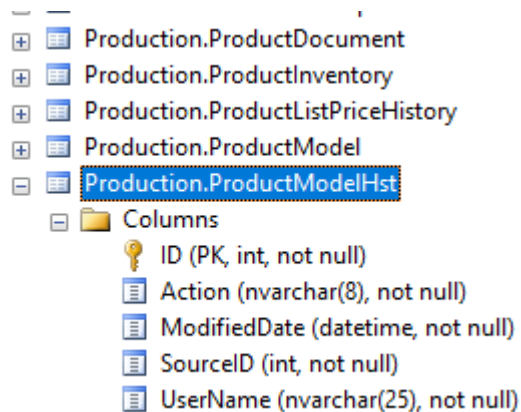
Минск 2020

Задание №1. Работа с представлением VIEW. Изменение данных в таблице через представление. Создание AFTER DML триггера для таблицы. Логирование изменений в history таблицу.

а) Создайте таблицу Production.ProductModelHst, которая будет хранить информацию об изменениях в таблице Production.ProductModel.

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.ProductModelHst (  
    ID INT IDENTITY(1,1) PRIMARY KEY,  
    [Action] NVARCHAR(8) NOT NULL CHECK(  
        [Action] IN (  
            'insert',  
            'update',  
            'delete'  
        )  
    ),  
    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.ProductModel. Триггер должен заполнять таблицу Production.ProductModelHst с указанием типа операции в поле Action в зависимости от оператора, вызвавшего триггер.

```

CREATE TRIGGER prod_model_operation_change_trigger
ON Production.ProductModel
AFTER
INSERT, UPDATE, DELETE
AS
| IF EXISTS (SELECT * FROM inserted) AND EXISTS (SELECT * FROM deleted)
|     INSERT INTO Production.ProductModelHst
|     SELECT
|         'update',
|         CURRENT_TIMESTAMP,
|         ProductModelID,
|         CURRENT_USER
|     FROM inserted
| ELSE IF EXISTS (SELECT * FROM inserted)
|     INSERT INTO Production.ProductModelHst
|     SELECT
|         'insert',
|         CURRENT_TIMESTAMP,
|         ProductModelID,
|         CURRENT_USER
|     FROM inserted
| ELSE IF EXISTS (SELECT * FROM deleted)
|     INSERT INTO Production.ProductModelHst
|     SELECT
|         'delete',
|         CURRENT_TIMESTAMP,
|         ProductModelID,
|         CURRENT_USER
|     FROM deleted;

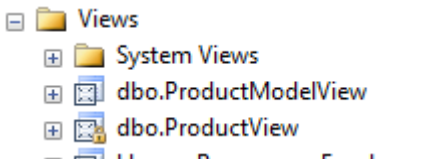
```

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

```

CREATE VIEW ProductModelView AS
| SELECT * FROM Production.ProductModel;
GO
|

```



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

```

)INSERT INTO ProductModelView (
    Name,
    rowguid,
    ModifiedDate
) VALUES (
    'test',
    'A35598F6-B413-4138-8081-5DC7D4C64B64',
    CURRENT_TIMESTAMP
);
GO

)UPDATE ProductModelView
SET Name = 'test2'
WHERE rowguid = 'A35598F6-B413-4138-8081-5DC7D4C64B64';
GO

)DELETE FROM ProductModelView
WHERE rowguid = 'A35598F6-B413-4138-8081-5DC7D4C64B64';
GO

```

129	134	hello2	NULL	NULL	3A1EED3A-1A82-4855-99CB-2A
130	135	prod1	NULL	NULL	A21EED3A-1A82-4855-99CB-2A
131	136	test	NULL	NULL	A35598F6-B413-4138-8081-5DC

129	134	hello2	NULL	NULL	
130	135	prod1	NULL	NULL	
131	136	test2	NULL	NULL	

129	134	hello2	NULL	NULL	3A1EED3A-1A82-4855-99CB-2A
130	135	prod1	NULL	NULL	A21EED3A-1A82-4855-99CB-2A

Query executed successfully. DESKTOP-UQEBPDI\SQLEXPRESS1... DESKTOP-UQEBPDI\hp (56) AdventureWorks2012 | 0

```

SELECT * FROM Production.ProductModelHst
GO

```

ID	Action	ModifiedDate	SourceID	UserName
1	insert	2020-11-01 00:16:43.373	136	dbo
2	update	2020-11-01 00:17:09.640	136	dbo
3	delete	2020-11-01 00:17:29.553	136	dbo

**Задание №2.** Работа с представлением VIEW. Индексированное представление. Создание INSTEAD OF DML триггера для представления.

- Создайте представление VIEW, отображающее данные из таблиц Production.ProductModel, Production.ProductModelProductDescriptionCulture, Production.Culture и Production.ProductDescription. Сделайте невозможным просмотр исходного кода представления. Создайте уникальный кластерный индекс в представлении по полям ProductModelID, CultureID.

```

CREATE VIEW dbo.ProductView
WITH ENCRYPTION, SCHEMABINDING
AS
SELECT
    culture.CultureID,
    culture.Name AS CultName,
    culture.ModifiedDate AS Cult_ModifiedDate,
    prod_model.CatalogDescription,
    prod_model.Instructions,
    prod_model.Name AS ProdModelName,
    prod_model.ProductModelID,
    prod_model.ModifiedDate AS ProdModel_ModifiedDate,
    prod_desc.[Description],
    prod_desc.ProductDescriptionID,
    prod_desc.rowguid,
    prod_desc.ModifiedDate AS ProdDesc_ModifiedDate,
    p_mod_p_desc_cult.ModifiedDate AS ProdModProdDescCult_ModifiedDate
FROM Production.ProductModel AS prod_model
JOIN Production.ProductModelProductDescriptionCulture AS p_mod_p_desc_cult
    ON prod_model.ProductModelID = p_mod_p_desc_cult.ProductModelID
JOIN Production.Culture AS culture
    ON culture.CultureID = p_mod_p_desc_cult.CultureID
JOIN Production.ProductDescription AS prod_desc
    ON prod_desc.ProductDescriptionID = p_mod_p_desc_cult.ProductDescriptionID;
GO

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

```

	CultureID	CultName	Cult_ModifiedDate	CatalogDescription	Instructions	ProdModelName	ProductModelID	ProdModel_ModifiedDate
1	en	English	2002-06-01 00:00:00.000	NULL	NULL	LL Bottom Bracket	95	2007-06-01 00:00:00
2	en	English	2002-06-01 00:00:00.000	NULL	NULL	ML Bottom Bracket	96	2007-06-01 00:00:00
3	en	English	2002-06-01 00:00:00.000	NULL	NULL	HL Bottom Bracket	97	2007-06-01 00:00:00
4	en	English	2002-06-01 00:00:00.000	<?xml-stylessheet ...	NULL	Mountain-500	23	2006-11-20 09:56:38
5	en	English	2002-06-01 00:00:00.000	NULL	NULL	Mountain-400-W	22	2007-06-01 00:00:00
6	en	English	2002-06-01 00:00:00.000	NULL	NULL	Mountain-300	21	2006-06-01 00:00:00
7	en	English	2002-06-01 00:00:00.000	NULL	NULL	Mountain-200	20	2006-06-01 00:00:00
8	en	English	2002-06-01 00:00:00.000	<?xml-stylessheet ...	NULL	Mountain-100	19	2005-06-01 00:00:00
9	en	English	2002-06-01 00:00:00.000	NULL	NULL	Mountain-400	39	2005-06-01 00:00:00
10	en	English	2002-06-01 00:00:00.000	NULL	NULL	Road-750	31	2006-11-20 09:56:38
11	en	English	2002-06-01 00:00:00.000	NULL	NULL	Road-650	30	2005-06-01 00:00:00

- b) Создайте три INSTEAD OF триггера для представления на операции INSERT, UPDATE, DELETE. Каждый триггер должен выполнять соответствующие операции в таблицах Production.ProductModel, Production.ProductModelProductDescriptionCulture, Production.Culture и Production.ProductDescription. Обновление не должно происходить в таблице Production.ProductModelProductDescriptionCulture. Удаление строк из таблиц Production.ProductModel, Production.Culture и Production.ProductDescription производите только в том случае, если удаляемые строки больше не ссылаются на Production.ProductModelProductDescriptionCulture.

```

--- insert trigger
CREATE TRIGGER ProductViewInsertTrigger
ON dbo.ProductView
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @modified_date DATETIME = CURRENT_TIMESTAMP;
    INSERT INTO Production.Culture(CultureID, Name, ModifiedDate)
        SELECT
            CultureID,
            CultName,
            @modified_date
        FROM inserted;
    INSERT INTO Production.ProductModel(Name, CatalogDescription, Instructions, rowguid, ModifiedDate)
        SELECT
            ProdModelName,
            CatalogDescription,
            Instructions,
            rowguid,
            @modified_date
        FROM inserted;
    INSERT INTO Production.ProductDescription([Description], rowguid, ModifiedDate)
        SELECT
            [Description],
            rowguid,
            @modified_date
        FROM inserted;
    INSERT INTO Production.ProductModelProductDescriptionCulture(
        CultureID,
        ProductModelID,
        ProductDescriptionID
    ) VALUES (
        (SELECT CultureID FROM inserted),
        IDENT_CURRENT('Production.ProductModel'),
        IDENT_CURRENT('Production.ProductDescription')
    );
END;
GO

```

```

--- update trigger
CREATE TRIGGER ProductViewUpdateTrigger
ON dbo.ProductView
INSTEAD OF UPDATE
AS
BEGIN
    DECLARE @modified_date DATETIME = CURRENT_TIMESTAMP;

    UPDATE Production.Culture
        SET Name = (SELECT CultName FROM inserted),
            ModifiedDate = @modified_date
    WHERE Name = (SELECT CultName FROM deleted);

    UPDATE Production.ProductModel
        SET Name = (SELECT ProdModelName FROM inserted),
            ModifiedDate = @modified_date
    WHERE Name = (SELECT ProdModelName FROM deleted);

    UPDATE Production.ProductDescription
        SET [Description] = (SELECT [Description] FROM inserted),
            ModifiedDate = @modified_date
    WHERE [Description] = (SELECT [Description] FROM deleted);
END;
GO

```

```

--- delete trigger
CREATE TRIGGER ProductViewDeleteTrigger
ON dbo.ProductView
INSTEAD OF DELETE
AS
BEGIN
    DECLARE @CultureID NCHAR(6);
    DECLARE @ProductDescriptionID [int];
    DECLARE @ProductModelID [int];
    SELECT
        @CultureID = CultureID,
        @ProductDescriptionID = ProductDescriptionID,
        @ProductModelID = ProductModelID
    FROM deleted;

    IF @CultureID NOT IN (SELECT CultureID FROM Production.ProductModelProductDescriptionCulture)
        DELETE FROM Production.Culture
        WHERE @CultureID = CultureID

    IF @ProductModelID NOT IN (SELECT @ProductModelID FROM Production.ProductModelProductDescriptionCulture)
        DELETE FROM Production.ProductModel
        WHERE @ProductModelID = ProductModelID

    IF @ProductDescriptionID NOT IN (SELECT ProductDescriptionID
        FROM Production.ProductModelProductDescriptionCulture)

        DELETE FROM Production.ProductDescription
        WHERE @ProductDescriptionID = ProductDescriptionID
END;
GO

```

- c) Вставьте новую строку в представление, указав новые данные для ProductModel, Culture и ProductDescription. Триггер должен добавить новые строки в таблицы Production.ProductModel, Production.ProductModelProductDescriptionCulture, Production.Culture и Production.ProductDescription. Обновите вставленные строки через представление. Удалите строки.

```

INSERT INTO ProductView(
    CultureID, CultName, ProdModelName, rowguid, [Description]
) VALUES (
    'test3', 'cult3', 'prod3', 'A51EED3A-1A82-4855-99CB-2AFE8290D641', 'hello'
)

SELECT * FROM ProductView

UPDATE ProductView
SET
    CultName = 'help'
WHERE CultureID = 'test3'

DELETE FROM ProductView
WHERE CultureID = 'test3'

```

759	fr	French	2002-06-01 00:00:00.000	NULL	NULL	Rear Derailleur	127
760	th	Thai	2002-06-01 00:00:00.000	NULL	NULL	Rear Derailleur	127
761	he	Hebrew	2002-06-01 00:00:00.000	NULL	NULL	Rear Derailleur	127
762	zh-cht	Chinese	2002-06-01 00:00:00.000	NULL	NULL	Rear Derailleur	127
763	test1	cult1	2020-10-31 23:31:41.900	NULL	NULL	prod1	135
764	test3	cult3	2020-11-01 00:25:55.010	NULL	NULL	prod3	137

761	he	Hebrew	2002-06-01 00:00:00.000	NULL			
762	zh-cht	Chinese	2002-06-01 00:00:00.000	NULL			
763	test1	cult1	2020-10-31 23:31:41.900	NULL			
764	test3	help	2020-11-01 00:28:29.180	NULL			

row	name	service	date and time	note
763	test1	cult1	2020-10-31 23:31:41.900	NULL
764	test3	help	2020-11-01 00:28:29.180	NULL