УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 4

по предмету

«Базы данных»

Выполнил:
Войтешонок А. Л.
группа 751001

Проверил:
Салей О. А.

Минск 2020

Вариант 6

Задание №1

Работа с представлением VIEW. Изменение данных в таблице через представление. Создание AFTER DML триггера для таблицы. Логгирование изменений в history таблицу.

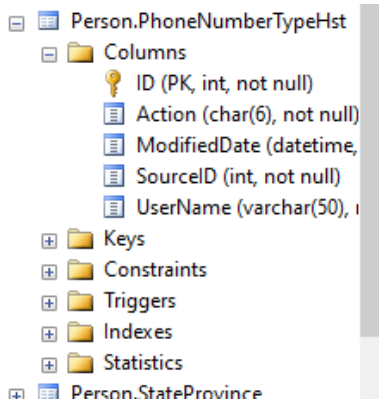
a) Создайте таблицу Person.PhoneNumberTypeHst, которая будет хранить информацию об изменениях в таблице Person.PhoneNumberType.

   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.

SQL запрос:

```
/*
a) Создайте таблицу Person.PhoneNumberTypeHst, которая будет хранить информацию об изменениях
в таблице Person.PhoneNumberType.
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 [Person].[PhoneNumberTypeHst] (
    [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
);
```

Результат выполнения:



b) Создайте три AFTER триггера для трех операций INSERT, UPDATE, DELETE для таблицы Person.PhoneNumberType. Каждый триггер должен заполнять таблицу Person.PhoneNumberTypeHst с указанием типа операции в поле Action.

.

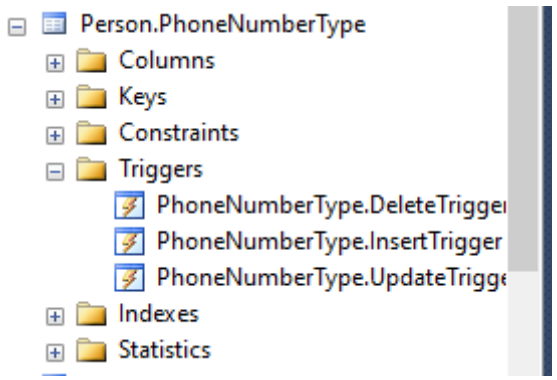
SQL запрос:

```
/*
b) Создайте три AFTER триггера для трех операций INSERT, UPDATE, DELETE
для таблицы Person.PhoneNumberType. Каждый триггер должен заполнять таблицу
Person.PhoneNumberTypeHst с указанием типа операции в поле Action.
*/
CREATE TRIGGER [Person].[PhoneNumberType.InsertTrigger]
ON [Person].[PhoneNumberType]
AFTER INSERT AS
    INSERT INTO [Person].[PhoneNumberTypeHst]([Action], [ModifiedDate], [SourceID], [UserName])
    SELECT 'INSERT', GETDATE(), [ins].[PhoneNumberTypeID], USER_NAME()
    FROM [inserted] AS [ins]

CREATE TRIGGER [Person].[PhoneNumberType.UpdateTrigger]
ON [Person].[PhoneNumberType]
AFTER UPDATE AS
    INSERT INTO [Person].[PhoneNumberTypeHst]([Action], [ModifiedDate], [SourceID], [UserName])
    SELECT 'UPDATE', GETDATE(), [ins].[PhoneNumberTypeID], USER_NAME()
    FROM [inserted] AS [ins]

CREATE TRIGGER [Person].[PhoneNumberType.DeleteTrigger]
ON [Person].[PhoneNumberType]
AFTER DELETE AS
    INSERT INTO [Person].[PhoneNumberTypeHst]([Action], [ModifiedDate], [SourceID], [UserName])
    SELECT 'DELETE', GETDATE(), [ins].[PhoneNumberTypeID], USER_NAME()
    FROM [deleted] AS [ins]
```

Результат выполнения:

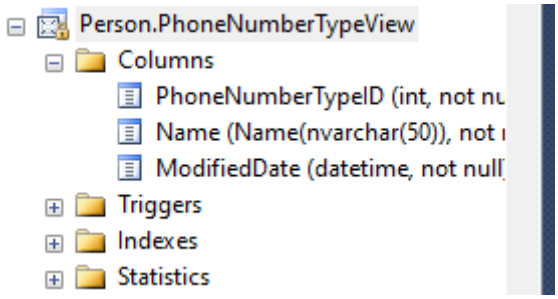


c) Создайте представление VIEW, отображающее все поля таблицы Person.PhoneNumberType. Сделайте невозможным просмотр исходного кода представления.

SQL запрос:

```sql
/*
c) Создайте представление VIEW, отображающее все поля таблицы
Person.PhoneNumberType. Сделайте невозможным просмотр исходного кода представления.
*/
CREATE VIEW [Person].[PhoneNumberTypeView]
WITH ENCRYPTION
AS SELECT * FROM [Person].[PhoneNumberType];
```

Результат выполнения:



d) Вставьте новую строку в Person.PhoneNumberType через представление. Обновите вставленную строку. Удалите вставленную строку. Убедитесь, что все три операции отображены в Person.PhoneNumberTypeHst..

SQL запрос:

```sql
/*
d) Вставьте новую строку в Person.PhoneNumberType через представление. Обновите вставленную
строку. Удалите вставленную строку. Убедитесь, что все три операции отображены в Person.PhoneNumberTypeHst.
*/
INSERT INTO [Person].[PhoneNumberTypeView]  ([Name], [ModifiedDate])
VALUES ('FirstName', GETDATE());

SELECT * FROM [Person].[PhoneNumberTypeHst]

UPDATE [Person].[PhoneNumberTypeView]
SET [Name] = 'SecondName'
WHERE [PhoneNumberTypeID] = (
    SELECT MAX([PhoneNumberTypeID])
    FROM [Person].[PhoneNumberTypeView]);

SELECT * FROM [Person].[PhoneNumberTypeHst]

DELETE FROM [Person].[PhoneNumberTypeView]
WHERE [PhoneNumberTypeID] = (
    SELECT MAX([PhoneNumberTypeID])
    FROM [Person].[PhoneNumberTypeView]);

SELECT * FROM [Person].[PhoneNumberTypeHst];
```

Результат выполнения:

Results | Messages

	ID	Action	ModifiedDate	SourceID	UserName
1	1	INSERT	2020-10-31 17:48:10.297	4	dbo
2	2	UPDATE	2020-10-31 17:48:26.710	4	dbo
3	3	DELETE	2020-10-31 17:48:52.460	4	dbo
4	4	INSERT	2020-10-31 17:52:42.627	5	dbo
5	5	UPDATE	2020-10-31 17:53:21.510	5	dbo
6	6	DELETE	2020-10-31 17:54:19.553	5	dbo
7	7	INSERT	2020-10-31 19:34:19.527	6	dbo

Задание №2

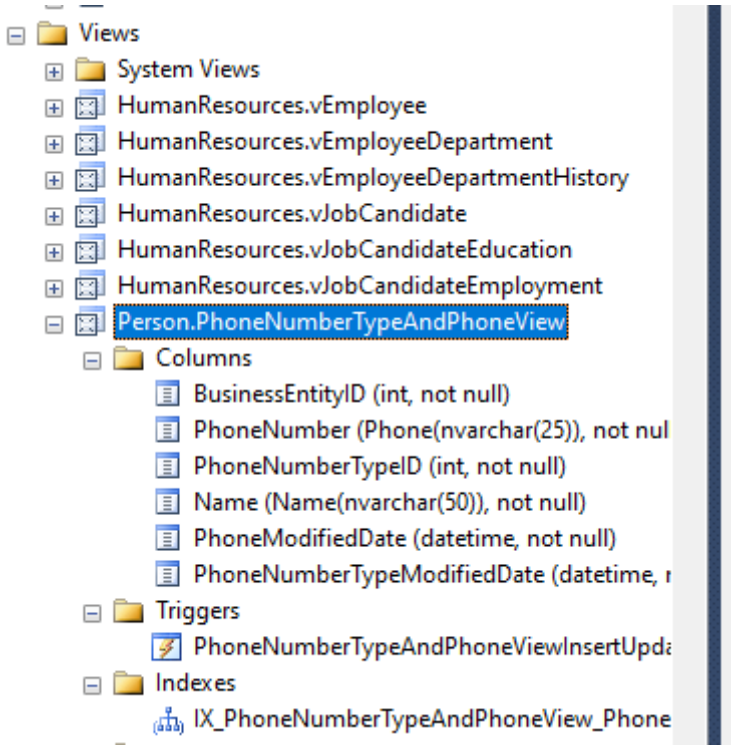
Индексированное представление. Создание AFTER DML триггера для представления.

а) Создайте представление VIEW, отображающее данные из таблиц Person.PhoneNumberType и Person.PersonPhone. Создайте уникальный кластерный индекс в представлении по полям PhoneNumberTypeID и BusinessEntityID.

SQL запрос:

```
/*
а) Создайте представление VIEW, отображающее данные из таблиц Person.PhoneNumberType
и Person.PersonPhone. Создайте уникальный кластерный индекс в представлении
по полям PhoneNumberTypeID и BusinessEntityID.
*/
CREATE VIEW [Person].[PhoneNumberTypeAndPhoneView] (
    [BusinessEntityID],
    [PhoneNumber],
    [PhoneNumberTypeID],
    [Name],
    [PhoneModifiedDate],
    [PhoneNumberTypeModifiedDate]
)
WITH SCHEMABINDING
AS SELECT
    [pp].[BusinessEntityID],
    [pp].[PhoneNumber],
    [pnt].[PhoneNumberTypeID],
    [pnt].[Name],
    [pp].[ModifiedDate],
    [pnt].[ModifiedDate]
FROM [Person].[PersonPhone] AS [pp]
INNER JOIN [Person].[PhoneNumberType] AS [pnt] ON [pnt].[PhoneNumberTypeID] = [pp].[PhoneNumberTypeID]
GO
```

Результат выполнения:



b) Создайте один INSTEAD OF триггер для представления на три операции INSERT, UPDATE, DELETE. Триггер должен выполнять соответствующие операции в таблицах Person.PhoneNumberType и Person.PersonPhone для указанного BusinessEntityID.

SQL запрос:

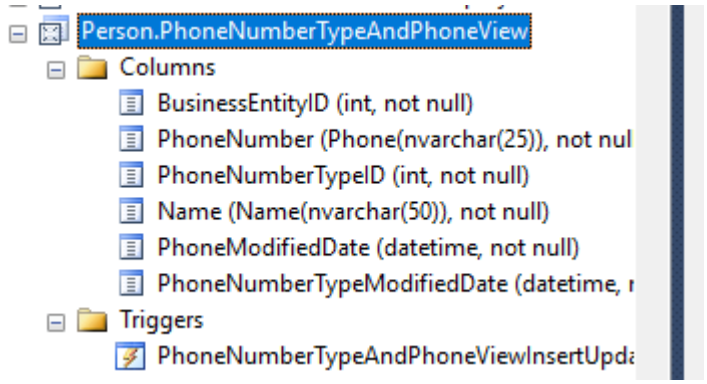
```sql
/*
b) Создайте один INSTEAD OF триггер для представления на три операции INSERT, UPDATE, DELETE.
Триггер должен выполнять соответствующие операции в таблицах Person.PhoneNumberType и Person.PersonPhone
для указанного BusinessEntityID.
*/
CREATE TRIGGER [Person].[PhoneNumberTypeAndPhoneViewInsertUpdateDeleteTrigger]
ON [Person].[PhoneNumberTypeAndPhoneView]
INSTEAD OF INSERT, UPDATE, DELETE AS
BEGIN
    IF EXISTS (SELECT * FROM [inserted])
    BEGIN
        IF NOT EXISTS (SELECT * FROM [deleted])
        BEGIN
            INSERT INTO [Person].[PhoneNumberType] (
                [Name],
                [ModifiedDate])
            SELECT
                [inserted].[Name],
                GETDATE()
            FROM [inserted]

            INSERT INTO [Person].[PersonPhone] (
                [BusinessEntityID],
                [PhoneNumber],
                [PhoneNumberTypeID],
                [ModifiedDate])
            SELECT
                [inserted].[BusinessEntityID],
                [inserted].[PhoneNumber],
                [pnt].[PhoneNumberTypeID],
                GETDATE()
            FROM [inserted]
            INNER JOIN [Person].[PhoneNumberType] [pnt] ON [pnt].[Name] = [inserted].[Name];
        END
        ELSE
        BEGIN
            UPDATE [Person].[PhoneNumberType] SET
                [Name] = [inserted].[Name],
                [ModifiedDate] = GETDATE()
            FROM [inserted], [deleted]
            WHERE [Person].[PhoneNumberType].[PhoneNumberTypeID] = [deleted].[PhoneNumberTypeID]

            UPDATE [Person].[PersonPhone] SET
                [BusinessEntityID] = [inserted].[BusinessEntityID],
                [PhoneNumber] = [inserted].[PhoneNumber],
                [ModifiedDate] = GETDATE()
            FROM [inserted], [deleted]
                [ModifiedDate] = GETDATE()
            FROM [inserted], [deleted]
            WHERE [Person].[PersonPhone].[BusinessEntityID] = [deleted].[BusinessEntityID]
            AND [Person].[PersonPhone].[PhoneNumber] = [deleted].[PhoneNumber]
        END
    END
    ELSE
    BEGIN
        DELETE FROM [Person].[PersonPhone]
        WHERE [BusinessEntityID] IN (SELECT [BusinessEntityID] FROM [deleted])
        AND [PhoneNumber] IN (SELECT [PhoneNumber] FROM [deleted])

        DELETE FROM [Person].[PhoneNumberType]
        WHERE [PhoneNumberTypeID] IN (SELECT [PhoneNumberTypeID] FROM [deleted])
    END
END;
GO
```

Результат выполнения:



c) Вставьте новую строку в представление, указав новые данные для указав новые данные для PhoneNumberType и PersonPhone для существующего BusinessEntityID (например 1). Триггер должен добавить новые строки в таблицы Person.PhoneNumberType и Person.PersonPhone. Обновите вставленные строки через представление. Удалите строки.

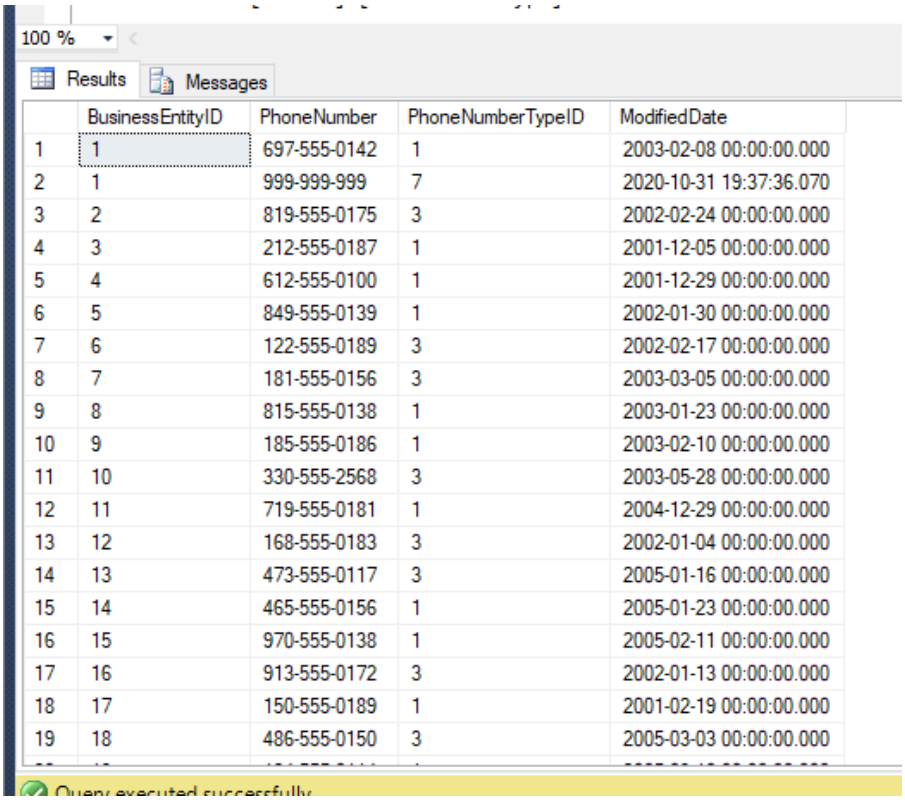
SQL запрос (вставка):

```
/*c)
Вставьте новую строку в представление, указав новые данные для PhoneNumberType и
PersonPhone для существующего BusinessEntityID (например 1). Триггер должен добавить новые строки в таблицы
Person.PhoneNumberType и Person.PersonPhone. Обновите вставленные строки через представление. Удалите строки.
*/
INSERT INTO [Person].[PhoneNumberTypeAndPhoneView] (
    [BusinessEntityID],
    [PhoneNumber],
    [Name])
VALUES(1, '999-999-999', 'NewType');

SELECT * FROM [Person].[PersonPhone]
SELECT * FROM [Person].[PhoneNumberType]
```

Результаты выполнения:

Person.PersonPhone:



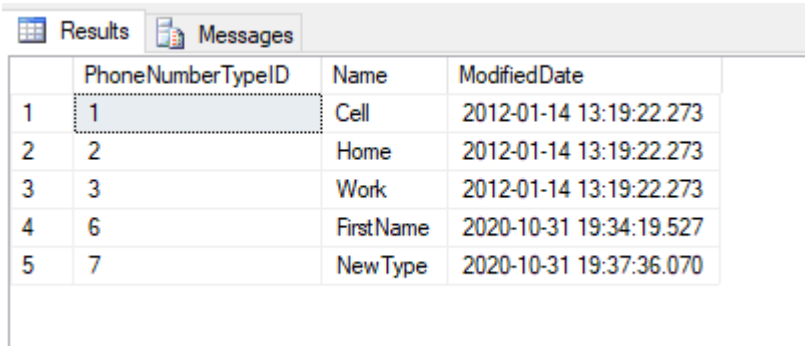
100 %

Results | Messages

	BusinessEntityID	PhoneNumber	PhoneNumberTypeID	ModifiedDate
1	1	697-555-0142	1	2003-02-08 00:00:00.000
2	1	999-999-999	7	2020-10-31 19:37:36.070
3	2	819-555-0175	3	2002-02-24 00:00:00.000
4	3	212-555-0187	1	2001-12-05 00:00:00.000
5	4	612-555-0100	1	2001-12-29 00:00:00.000
6	5	849-555-0139	1	2002-01-30 00:00:00.000
7	6	122-555-0189	3	2002-02-17 00:00:00.000
8	7	181-555-0156	3	2003-03-05 00:00:00.000
9	8	815-555-0138	1	2003-01-23 00:00:00.000
10	9	185-555-0186	1	2003-02-10 00:00:00.000
11	10	330-555-2568	3	2003-05-28 00:00:00.000
12	11	719-555-0181	1	2004-12-29 00:00:00.000
13	12	168-555-0183	3	2002-01-04 00:00:00.000
14	13	473-555-0117	3	2005-01-16 00:00:00.000
15	14	465-555-0156	1	2005-01-23 00:00:00.000
16	15	970-555-0138	1	2005-02-11 00:00:00.000
17	16	913-555-0172	3	2002-01-13 00:00:00.000
18	17	150-555-0189	1	2001-02-19 00:00:00.000
19	18	486-555-0150	3	2005-03-03 00:00:00.000

Query executed successfully.

Person.PhoneNumberType:



Results | Messages

	PhoneNumberTypeID	Name	ModifiedDate
1	1	Cell	2012-01-14 13:19:22.273
2	2	Home	2012-01-14 13:19:22.273
3	3	Work	2012-01-14 13:19:22.273
4	6	FirstName	2020-10-31 19:34:19.527
5	7	NewType	2020-10-31 19:37:36.070

SQL запрос (обновление):

```
UPDATE [Person].[PhoneNumberTypeAndPhoneView] SET
    [Name] = 'NewType2',
    [PhoneNumber] = '666-666-666'
WHERE [PhoneNumber] = '999-999-999';

SELECT * FROM [Person].[PersonPhone]
SELECT * FROM [Person].[PhoneNumberType]
```

Результат выполнения запроса:

Results | Messages

	PhoneNumberTypeID	Name	ModifiedDate
1	1	Cell	2012-01-14 13:19:22.273
2	2	Home	2012-01-14 13:19:22.273
3	3	Work	2012-01-14 13:19:22.273
4	6	FirstName	2020-10-31 19:34:19.527
5	7	NewType2	2020-10-31 19:39:35.570

SQL запрос (удаление):

```
DELETE FROM [Person].[PhoneNumberTypeAndPhoneView]
WHERE [PhoneNumber] = '666-666-666';

SELECT * FROM [Person].[PersonPhone]
SELECT * FROM [Person].[PhoneNumberType]
```

Результат выполнения запроса:

Results | Messages

	PhoneNumberTypeID	Name	ModifiedDate
1	1	Cell	2012-01-14 13:19:22.273
2	2	Home	2012-01-14 13:19:22.273
3	3	Work	2012-01-14 13:19:22.273
4	6	FirstName	2020-10-31 19:34:19.527