Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

(Университет ИТМО)

Факультет **Инфокоммуникационных технологий**

Образовательная программа **11.03.02 Программирование в инфокоммуникационных системах**

ОТЧЕТ
по лабораторной работе 7
по дисциплине **«Разработка баз данных»**

Выполнил: **студент группы K33211 Швалов Д. А.**

Проверил: **ст. преподаватель Осетрова И.С.**

Санкт-Петербург

2024

Лабораторная работа №7 «Создание триггеров»

1. Цель работы

Создание триггеров.

2. Задачи, решаемые при выполнении работы

1. Создание триггера DML.
2. Создание триггера DDL.
3. Удаление триггера DDL.
4. Дополнительное задание 1.
5. Дополнительное задание 2.

3. Объект исследования

Создание триггеров в СУБД Microsoft SQL Server с помощью Microsoft SQL Server Management Studio (SSMS).

4. Исходные данные

— методические указания к лабораторной работе;
— СУБД Microsoft SQL Server;
— Microsoft SQL Server Management Studio;
— база данных ApressFinancial.

5. Выполнение работы

5.1. Первая задача

5.1.1. Создание триггера DML

С помощью запроса, показанного на рисунке 1, был создан триггер DML под названием «trg_InsTransactions», который изменяет остаток средств на счете клиента при выполнении финансовой операции (т. е. при создании транзакции). Как видно на рисунке 2, после выполнения запроса триггер был успешно создан.

С помощью запроса, изображенного на рисунке 3, был протестирован созданный триггер. Для этого для клиента с идентификатором, равный двум, была создана новая транзакция. Как видно на рисунке 4, баланс клиента уменьшился на 200.

```
CREATE TRIGGER TransactionDetails.trg_InsTransactions
    ON TransactionDetails.Transactions
    AFTER INSERT
AS
BEGIN
    UPDATE CustomerDetails.Customers
    SET ClearedBalance = ClearedBalance + (
        SELECT
            CASE
                WHEN CreditType = 0 THEN i.Amount * -1
                ELSE i.Amount
            END
        FROM INSERTED AS i
        JOIN TransactionDetails.TransactionTypes AS tt
        ON tt.TransactionTypeId = i.TransactionTypeId
        WHERE AffectCashBalance = 1
    )
    FROM CustomerDetails.Customers AS c
    JOIN INSERTED AS i
    ON i.CustomerId = c.CustomerId
END
GO
```

Рисунок 1 – Запрос для создания триггера «trg_InsTransactions»

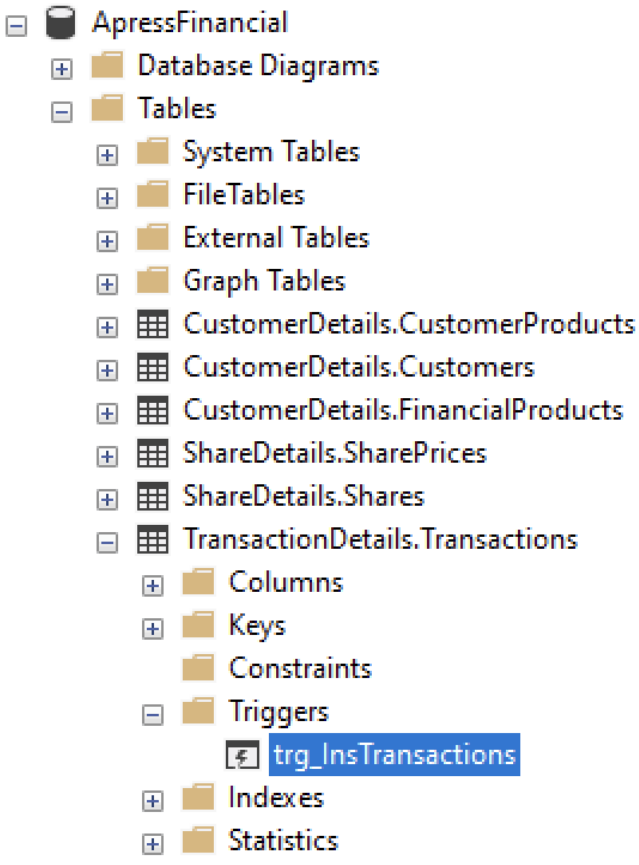

Рисунок 2 – Созданный триггер

```
SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2;

INSERT INTO TransactionDetails.Transactions (
    CustomerId,
    TransactionTypeId,
    Amount,
    RelatedProductId,
    DateEntered
) VALUES
(2, 4, 200, 1, GETDATE());

SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2
GO
```

Рисунок 3 – Запрос для тестирования созданного триггера

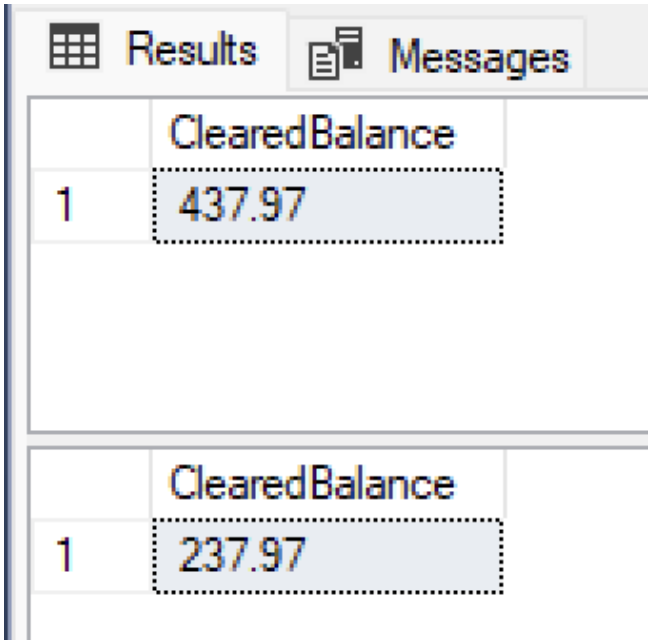

Рисунок 4 – Результат выполнения запроса

5.1.2. Изменение триггера DML

Триггер, созданный на предыдущем этапе, не будет работать в случае, если транзакция не изменяет баланс клиента. Чтобы проверить это, с помощью запроса, показанного на рисунке 5, поле «ClearedBalance» было сделано обязательным. При выполнении запроса, приведенного на рисунке 6, в котором создается транзакция, не влияющая на баланс клиента, появляется ошибка (рисунок 7).

С помощью запроса, показанного на рисунке 8, созданный ранее триггер был изменен: теперь, если подзапрос возвращает пустое значение, то вместо него возвращается значение 0. С помощью запроса, изображенного на рисунке 9, было протестировано поведение измененного триггера. Как видно на рисунке 10, теперь запрос выполняется без ошибок и транзакция создается.

```
UPDATE CustomerDetails.Customers
SET ClearedBalance = 0
WHERE ClearedBalance IS NULL
GO

IF EXISTS (
    SELECT *
    FROM sys.indexes
    WHERE name = N'IX_Customers_LastNameZip'
)
DROP INDEX IX_Customers_LastNameZip
ON CustomerDetails.Customers
GO

ALTER TABLE CustomerDetails.Customers
ALTER COLUMN ClearedBalance money NOT NULL
GO
```

Рисунок 5 – Запрос для изменения поля «ClearedBalance»

```
SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2;

INSERT INTO TransactionDetails.Transactions (
    CustomerId,
    TransactionTypeId,
    Amount,
    RelatedProductId,
    DateEntered
) VALUES
(2, 6, 200, 1, GETDATE());

SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2;
```

Рисунок 6 – Запрос для создания транзакции, не изменяющей баланс клиента

```
(1 row affected)
Msg 515, Level 16, State 2, Procedure trg_InsTransactions, Line 6 [Batch Start Line
Cannot insert the value NULL into column 'ClearedBalance', table 'ApressFinancial.C
The statement has been terminated.
```

Рисунок 7 – Сообщение об ошибке

```
ALTER TRIGGER TransactionDetails.trg_InsTransactions
ON TransactionDetails.Transactions
AFTER INSERT
AS
BEGIN
    UPDATE CustomerDetails.Customers
    SET ClearedBalance = ClearedBalance + ISNULL(
            (
                SELECT
                    CASE
                        WHEN CreditType = 0 THEN i.Amount * -1
                        ELSE i.Amount
                    END
                FROM INSERTED AS i
                JOIN TransactionDetails.TransactionTypes AS tt
                ON tt.TransactionTypeId = i.TransactionTypeId
                WHERE AffectCashBalance = 1
            ),
            0
        )
    FROM CustomerDetails.Customers AS c
    JOIN INSERTED AS i
    ON i.CustomerId = c.CustomerId
END
GO
```

Рисунок 8 – Запрос для изменения триггера

```
SELECT TransactionId, TransactionTypeId, Amount, DateEntered
FROM TransactionDetails.Transactions
WHERE CustomerId = 2
ORDER BY TransactionId DESC;

SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2;

INSERT INTO TransactionDetails.Transactions (
    CustomerId,
    TransactionTypeId,
    Amount,
    RelatedProductId,
    DateEntered
) VALUES
(2, 6, 200, 1, GETDATE());

SELECT TransactionId, TransactionTypeId, Amount, DateEntered
FROM TransactionDetails.Transactions
WHERE CustomerId = 2
ORDER BY TransactionId DESC;

SELECT ClearedBalance
FROM CustomerDetails.Customers
WHERE CustomerId = 2
GO
```

Рисунок 9 – Запрос для тестирования изменений

Results Messages

	TransactionId	TransactionTypeId	Amount	DateEntered
1	2801	4	200.00	2024-04-13 18:46:47.953
2	607	2	34215.274	2012-03-06 14:20:18.770
3	244	1	14149.7153	2012-03-22 22:04:05.440

	ClearedBalance
1	237.97

	TransactionId	TransactionTypeId	Amount	DateEntered
1	2803	6	200.00	2024-04-13 19:04:18.280
2	2801	4	200.00	2024-04-13 18:46:47.953
3	607	2	3421...	2012-03-06 14:20:18.770
4	244	1	1414...	2012-03-22 22:04:05.440

	ClearedBalance
1	237.97

Рисунок 10 – Результат выполнения запроса

5.2. Вторая задача

5.2.1. Создание триггера DDL (хранимые процедуры)

С помощью запроса, показанного на рисунке 11, был создан триггер DDL с наименованием «trg_Sprocs». Данный триггер выполняется при создании хранимой процедуры и проверяет, что триггер создается в нерабочие часы. После выполнения данного запроса новый триггер появился в интерфейсе SSMS (рисунок 12).

Чтобы протестировать созданный триггер, были выполнены два запроса. Первый, показанный на рисунке 13, был выполнен в нерабочие часы, а потому процедура была успешно создана. Второй, изображенный на рисунке 14, был выполнен уже в рабочие часы. Из-за этого процедура не была создана, вместо этого на указанную в триггере электронную почту было отправлено сообщение.

```
CREATE TRIGGER trg_Sprocs
ON DATABASE
FOR CREATE_PROCEDURE, ALTER_PROCEDURE, DROP_PROCEDURE
AS
IF DATEPART(hh, GETDATE()) > 9 AND DATEPART(hh, GETDATE()) < 17
BEGIN
    DECLARE @Message nvarchar(max)
    SELECT
        @Message =
            'Completion work during core hours. Trying to release -'
            + EVENTDATA().value
        ('(/EVENT_INSTANCE/TSQLCOmmand/CommandText)[1]', 'nvarchar(max)')
    RAISERROR (@Message, 16, 1)
    ROLLBACK
    EXEC msdb.dbo.sp_send_dbmail
        @profile_name = 'SQL Server Database Mail Profile',
        @recipients = 'exam@limtu.sp.ru',
        @body = 'A Stored procedure change',
        @subject = 'A stored procedure change has been initiated and rolled back during core hours'
END
GO
```

Рисунок 11 – Запрос для создания триггера «trg_Sprocs»

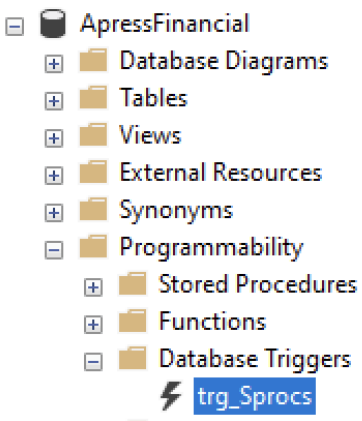

Рисунок 12 – Созданный триггер

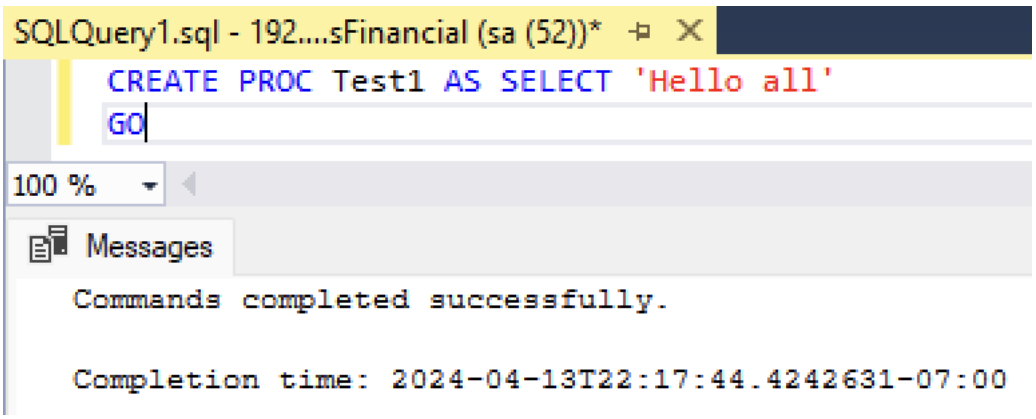

Рисунок 13 – Запрос для создания процедуры в нерабочие часы

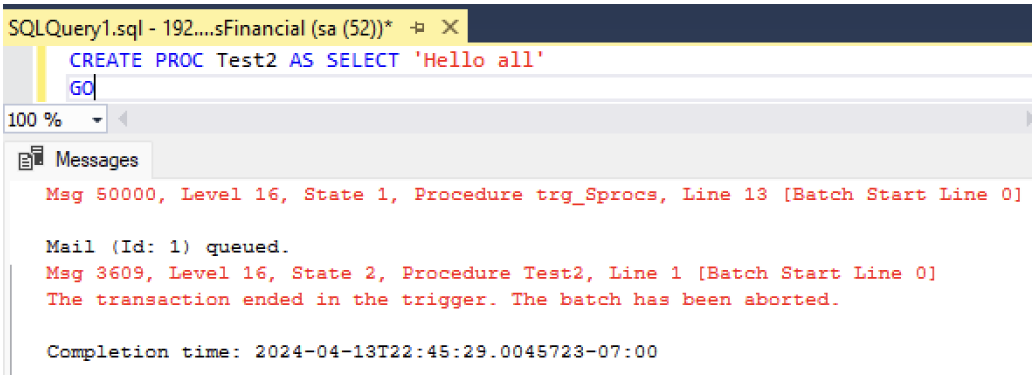

Рисунок 14 – Запрос для создания триггера в рабочие часы

5.2.2. Создание триггера DDL (любое действие)

С помощью запроса, показанного на рисунке 15, был создан триггер DLL под названием «trg_DBDump», который вызывается по любому действию, выполняемому в БД. Сам триггер выводит информацию о текущем действии. Как показано на рисунке 16, после выполнения запроса триггер был успешно создан.

С помощью запроса, показанного на рисунке 17, был протестирован созданный триггер. При выполнении данного запроса выводится сообщение с ссылкой на произошедшее событие. При нажатии левой кнопкой мыши на данную ссылку, открывается подробное описание события (рисунок 18).

```
CREATE TRIGGER trg_DBDump
ON DATABASE
FOR DDL_DATABASE_LEVEL_EVENTS
AS SELECT EVENTDATA()
GO
```

Рисунок 15 – Запрос для создания триггера «trg_DBDump»

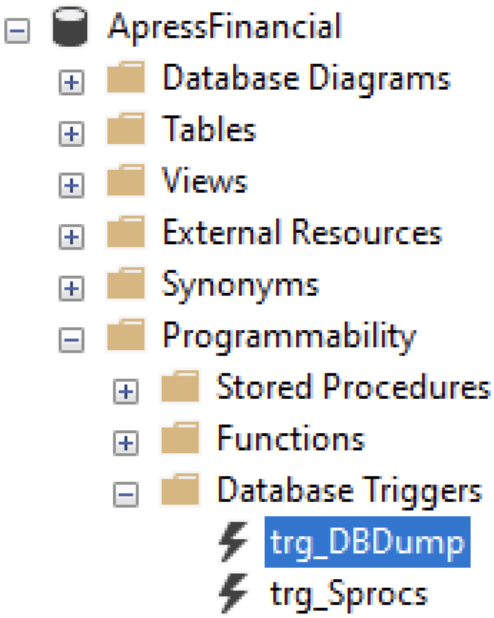

Рисунок 16 – Созданный триггер

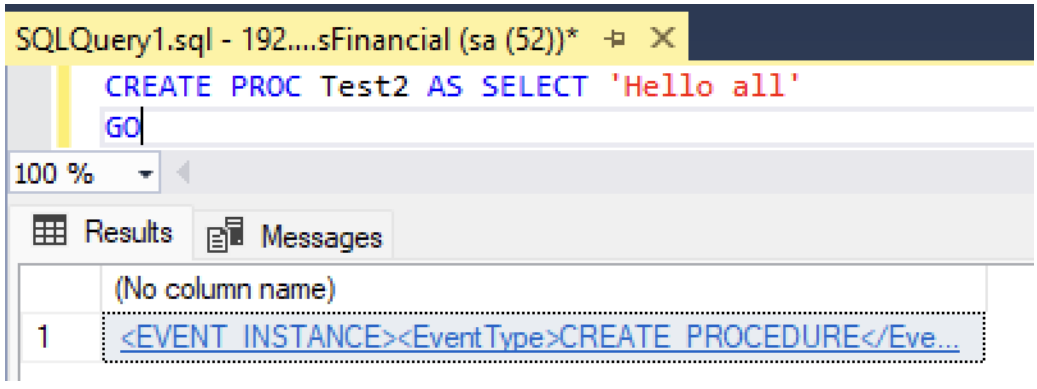

Рисунок 17 – Запрос для тестирования созданного триггера

```
<EVENT_INSTANCE>
  <EventType>CREATE_PROCEDURE</EventType>
  <PostTime>2024-04-13T19:52:26.473</PostTime>
  <SPID>52</SPID>
  <ServerName>sql</ServerName>
  <LoginName>sa</LoginName>
  <UserName>dbo</UserName>
  <DatabaseName>ApressFinancial</DatabaseName>
  <SchemaName>dbo</SchemaName>
  <ObjectName>Test2</ObjectName>
  <ObjectType>PROCEDURE</ObjectType>
  <TSQLCommand>
    <SetOptions ANSI_NULLS="ON" ANSI_NULL_DEFAULT="ON" ANSI_PADDING="ON" QUOTED_IDENTIFIER="ON" ENCRYPTED="FALSE" />
    <CommandText>CREATE PROC Test2 AS SELECT 'Hello all'
</CommandText>
  </TSQLCommand>
</EVENT_INSTANCE>
```

Рисунок 18 – Информация о событии

5.3. Третья задача

5.3.1. Удаление созданных триггеров и процедур

С помощью запросов, показанный на рисунках 19 и 20, были удалены созданные триггеры и процедуры. После выполнения данных запросов триггеры исчезли из интерфейса SSMS.

```
DROP TRIGGER trg_Sprocs ON DATABASE
GO

DROP TRIGGER trg_DBDump ON DATABASE
GO
```

Рисунок 19 – Запрос для удаления созданных триггеров

```
DROP PROC Test1
GO

DROP PROC Test2
GO
```

Рисунок 20 – Запрос для удаления созданных процедур

5.4. Четвертая задача

5.4.1. Дополнительное задание №1

С помощью запроса, показанного на рисунке 21, созданный в предыдущих заданиях триггер был изменен. Теперь триггер «trg_InsTrasnactions» работает не только при создании транзакции, но и при изменении. Однако, согласно условию данного задания, триггер пока что не изменяет данные, если произошло обновление уже существующей транзакции.

С помощью запросов, показанных на рисунках 22 и 23, было протестировано поведение измененного триггера. Как видно на рисунках, при создании транзакции триггер изменяет данные, а при обновлении — нет.

```
DROP TRIGGER TransactionDetails.trg_InsTransactions
GO

CREATE TRIGGER TransactionDetails.trg_InsTransactions
    ON TransactionDetails.Transactions
    AFTER INSERT, UPDATE
AS
BEGIN
    IF NOT EXISTS (SELECT * FROM DELETED)
        UPDATE CustomerDetails.Customers
        SET ClearedBalance = ClearedBalance + (
            SELECT
                CASE
                    WHEN CreditType = 0 THEN i.Amount * -1
                    ELSE i.Amount
                END
            FROM INSERTED AS i
            JOIN TransactionDetails.TransactionTypes AS tt
            ON tt.TransactionTypeId = i.TransactionTypeId
            WHERE AffectCashBalance = 1
        )
        FROM CustomerDetails.Customers AS c
        JOIN INSERTED AS i
        ON i.CustomerId = c.CustomerId;
END
GO
```

Рисунок 21 – Запрос для изменения триггера trg_InsTransactions

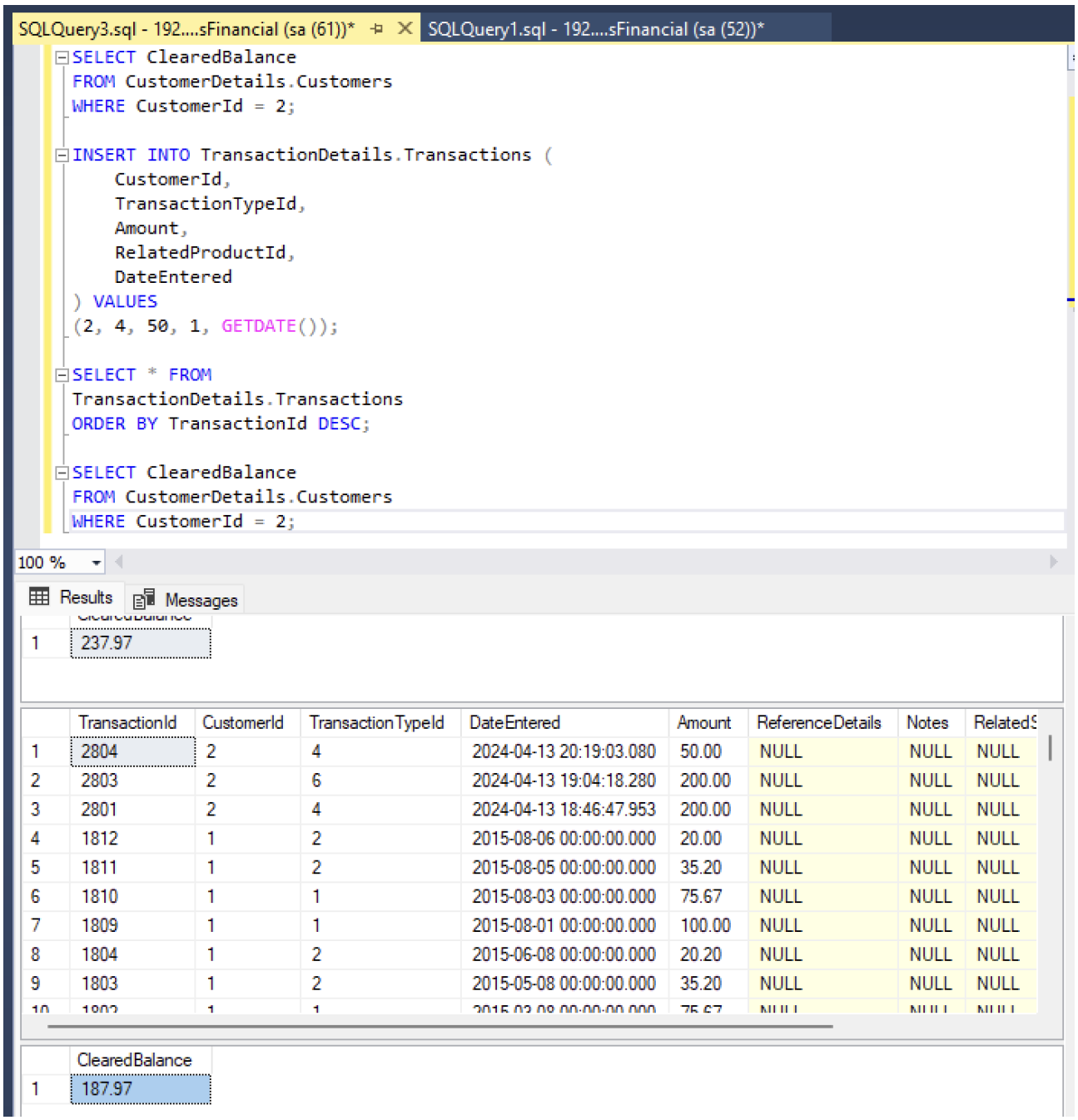

Рисунок 22 – Запрос с созданием транзакции

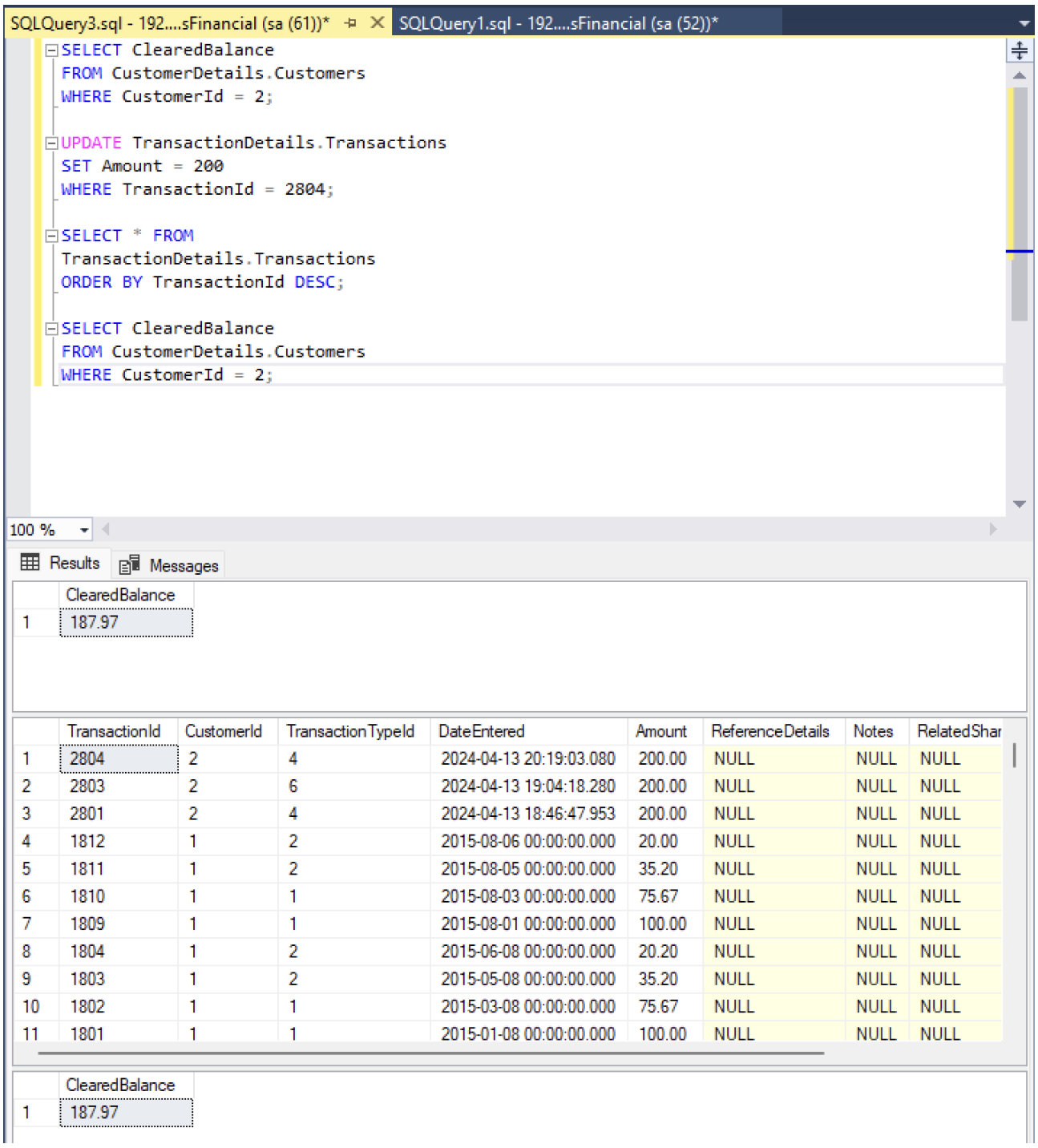

Рисунок 23 – Запрос с изменением транзакции

5.5. Пятая задача

5.5.1. Дополнительное задание №2

С помощью запроса, показанного на рисунке 24, измененный триггер был дополнен. Теперь баланс клиента меняется не только в случае, если была создана новая транзакция, но и в случае, если было изменено либо поле «Amount», либо поле «TransactionTypeId». Также были добавлены сообщения, которые выводится в зависимости от того, был ли изменен баланс клиента или нет.

Как видно на рисунке 25, при выполнении изменений одного из этих полей баланс клиента меняется, а при изменении другого поля баланс остается прежним. При этом, как изображено на рисунке 26, выводится сообщение об изменении или об отсутствии изменения.

```
DROP TRIGGER TransactionDetails.trg_InsTransactions
GO

CREATE TRIGGER TransactionDetails.trg_InsTransactions
    ON TransactionDetails.Transactions
    AFTER INSERT, UPDATE
AS
BEGIN
    IF NOT EXISTS (SELECT * FROM DELETED) OR UPDATE(Amount) OR UPDATE(TransactionTypeId)
    BEGIN
        UPDATE CustomerDetails.Customers
        SET ClearedBalance = ClearedBalance + (
            SELECT
                CASE
                    WHEN CreditType = 0 THEN i.Amount * -1
                    ELSE i.Amount
                END
            FROM INSERTED AS i
            JOIN TransactionDetails.TransactionTypes AS tt
            ON tt.TransactionTypeId = i.TransactionTypeId
            WHERE AffectCashBalance = 1
        )
        FROM CustomerDetails.Customers AS c
        JOIN INSERTED AS i
        ON i.CustomerId = c.CustomerId;
        RAISERROR(N'Изменение произошло успешно', 10, 1);
    END
    ELSE
        RAISERROR(N'Изменение не произошло', 10, 1);
END
GO
```

Рисунок 24 – Запрос для изменения триггера trg_InsTransactions

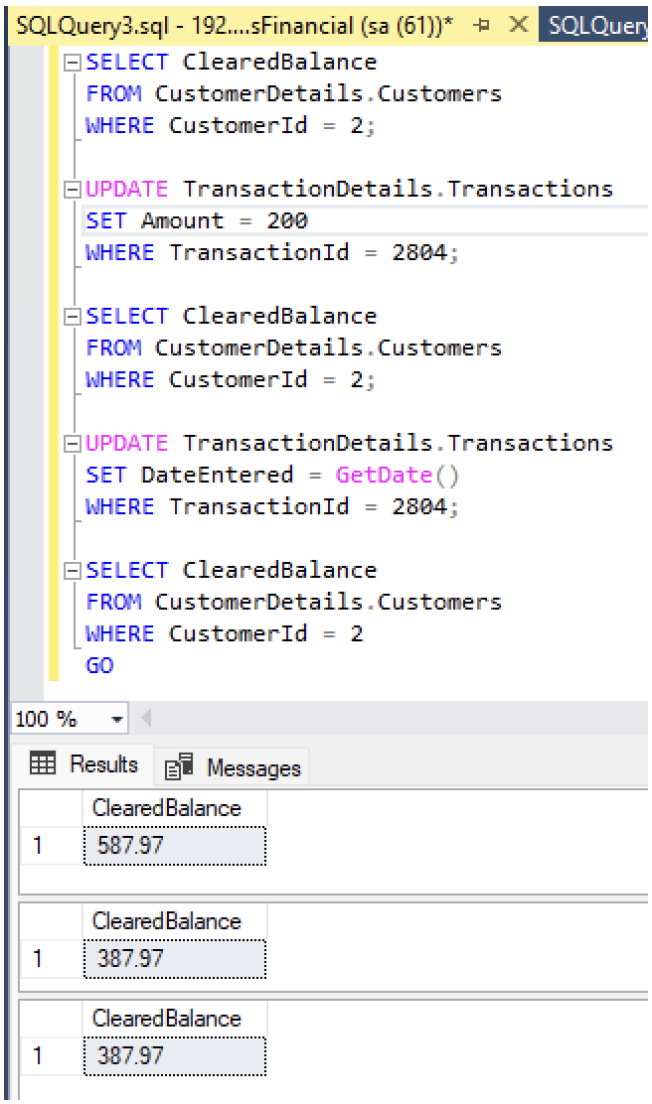

Рисунок 25 – Запрос с изменением транзакций

```
(1 row affected)

(1 row affected)
Изменение произошло успешно

(1 row affected)

(1 row affected)
Изменение не произошло

(1 row affected)

(1 row affected)
```

Рисунок 26 – Сообщения при изменении транзакций

6. Выводы и анализ результатов работы

В данной лабораторной работе изучены способы создания триггеров в SSMS. При выполнении лабораторной работы были созданы два типа триггеров: DML и DDL. При создании данных триггеров были рассмотрены различные события, для которых можно создать тот или иной триггер (создание новой строки в таблице, изменение таблиц и т. п.). Как было показано, триггеры являются очень удобным и необходимым инструментом для создания эффективных и качественных систем хранения данных. Без триггеров работа с базой данных была бы куда сложнее и менее безопасной.

Цель, поставленная в начале работы, достигнута, задачи выполнены.