**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Систем автоматизированного проектирования**

**отчет**

**по лабораторной работе №11**

**по дисциплине «Базы данных»**

**Тема: Создание триггеров**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 2308 |  | Чиков А.А. |
|  |  | Попов Н.А. |
|  |  | Бебия Р.А. |
| Преподаватель |  | Горяинов С.В. |

Санкт-Петербург

2024

**Цель работы**

Цель работы заключается в том, чтобы научиться создавать триггеры. В лабораторной работе используется база данных AdventureWorks.

**Выполнение работы**

**Упражнение 1 – создание новой таблицы**

USE [AdventureWorks]

GO

CREATE TABLE [HumanResources].[JobCandidateHistory](

[JobCandidateID][int] NOT NULL UNIQUE,

[Resume][xml] NULL,

[Rating][int] NOT NULL CONSTRAINT

[DF\_JobCandidateHistory\_Rating] Default(5),

[RejectedDate][datetime] NOT NULL,

[ContactID][int] NULL,

CONSTRAINT [FK\_JobCandidateHistory\_Contact\_ContactID] FOREIGN KEY(ContactID) REFERENCES [Person].[Contact](ContactID),

CONSTRAINT [CK\_JobCandidateHistory\_Rating] CHECK([Rating]>=0 AND [Rating]<=10)

) ON[PRIMARY]

Результат выполнения запроса представлен на Рисунке 1.

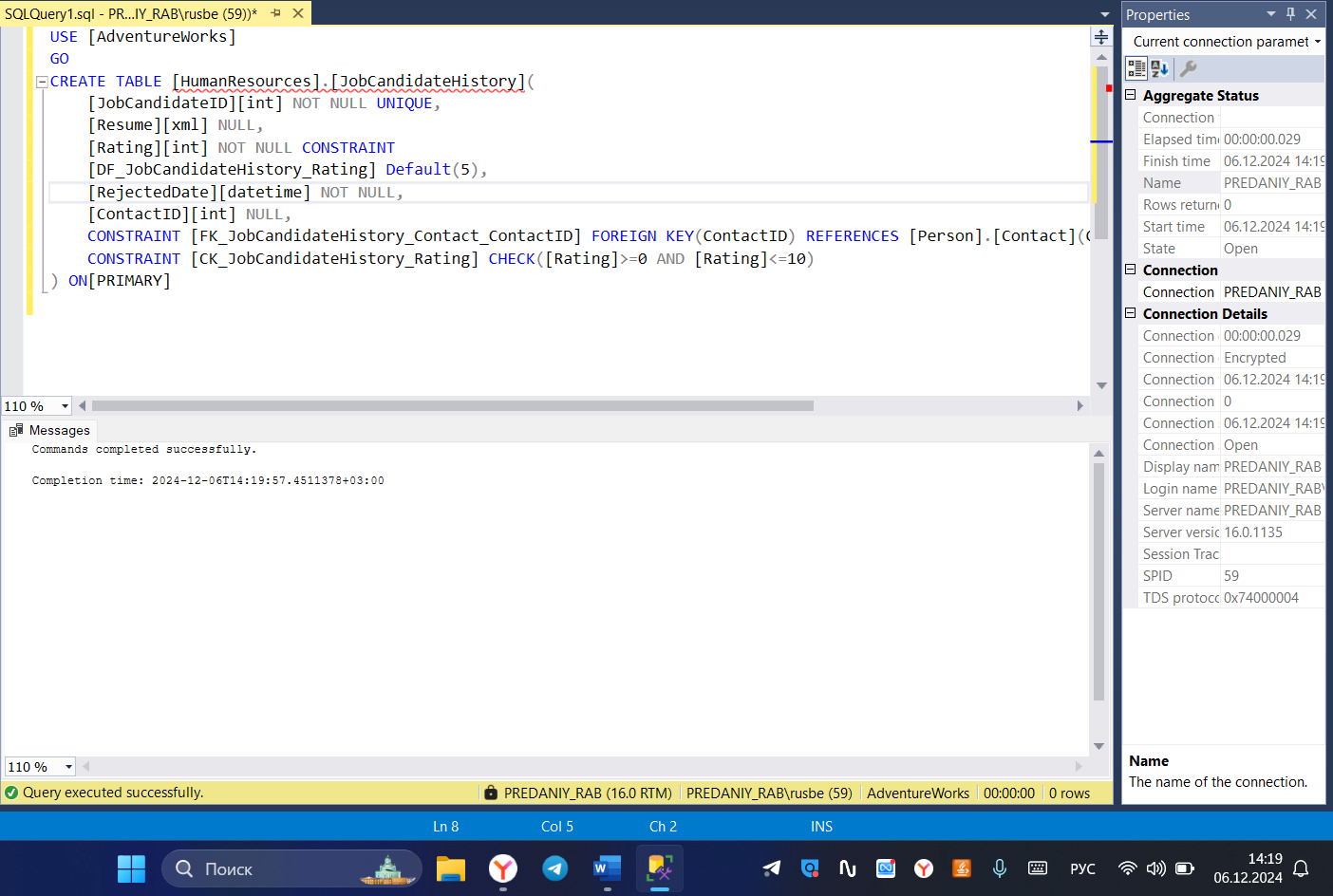


Рисунок 1 – Результат выполнения запроса

**Упражнение 2 – создание триггера для таблицы JobCandidate схемы HumanResources**

Запрос 1: Создание триггера dJobCandidate. Триггер вставляет данные в таблицу JobCandidateHistory после выполнения удаления данных из таблицы JobCandidate.

CREATE TRIGGER dJobCandidate

ON HumanResources.JobCandidate

AFTER DELETE

AS

BEGIN

INSERT INTO HumanResources.JobCandidateHistory (

JobCandidateID, Resume, RejectedDate, ContactID

)

SELECT JobCandidateID, Resume, GETDATE(), NULL

FROM DELETED;

END;

GO

Результат выполнения запроса представлен на Рисунке 2.

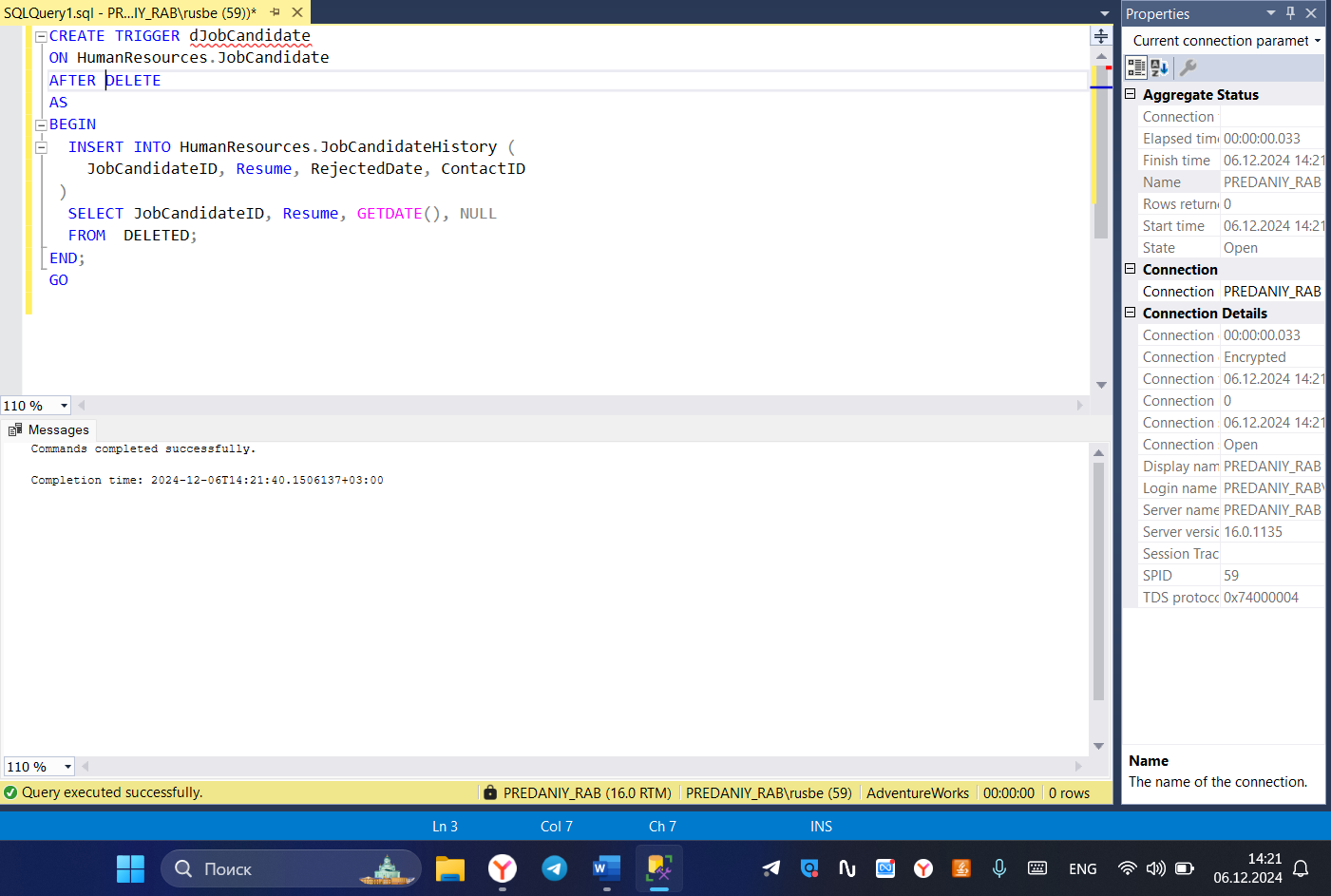


Рисунок 2 – Результат выполнения запроса

**Упражнение 3 – проверка работы триггера**

Запрос 1: Выполнение следующей команды.

USE AdventureWorks

GO

DELETE FROM HumanResources.JobCandidate

WHERE JobCandidateID = (SELECT MIN(JobCandidateID) FROM HumanResources.JobCandidate)

SELECT \* FROM [HumanResources].[JobCandidateHistory]

Результат выполнения запроса представлен на Рисунке 3.

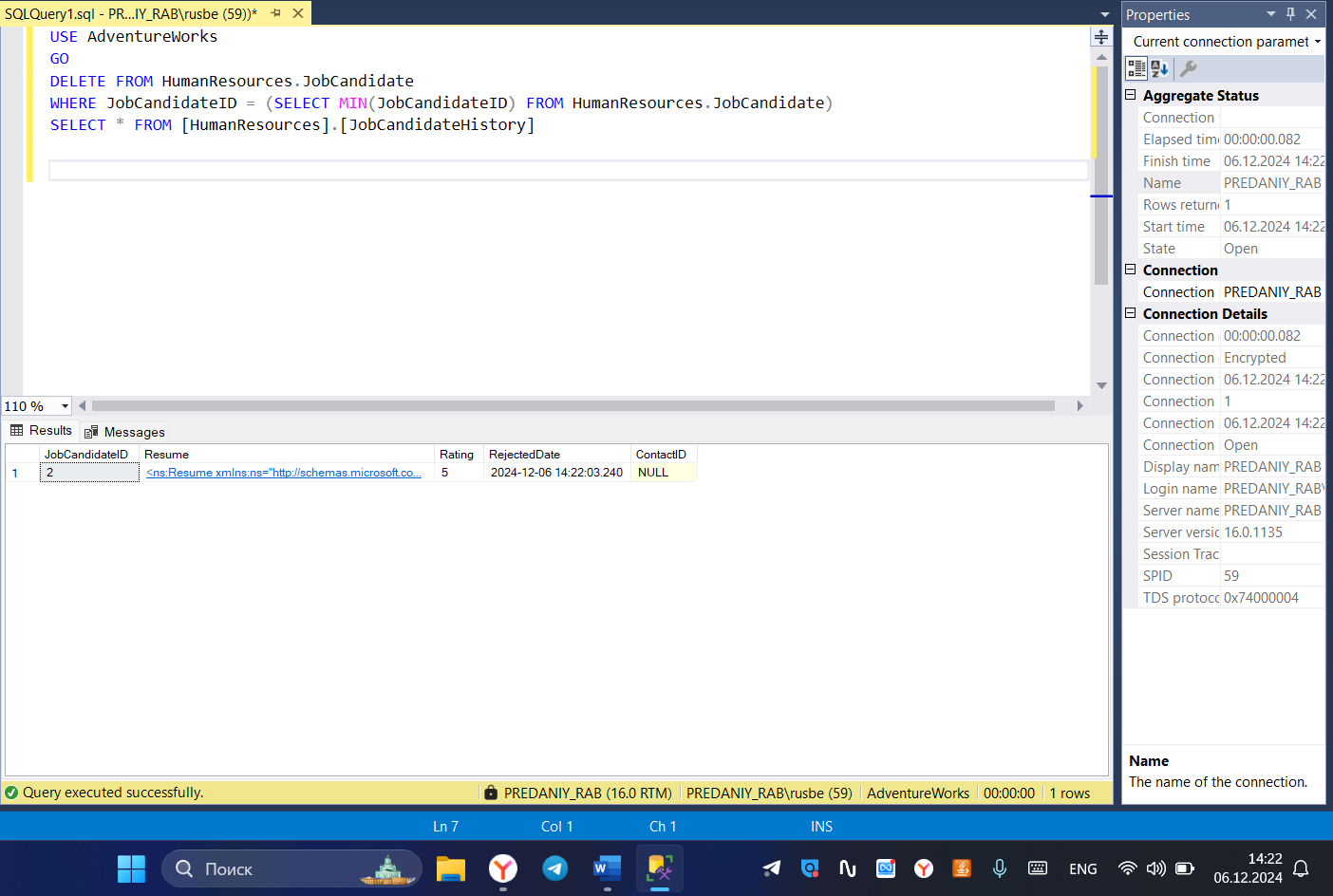


Рисунок 3 – Результат выполнения запроса

Запрос 2: Удаление данных из таблицы JobCandidateHistory.

TRUNCATE TABLE [HumanResources].[JobCandidateHistory]

Результат выполнения запроса представлен на Рисунке 4.

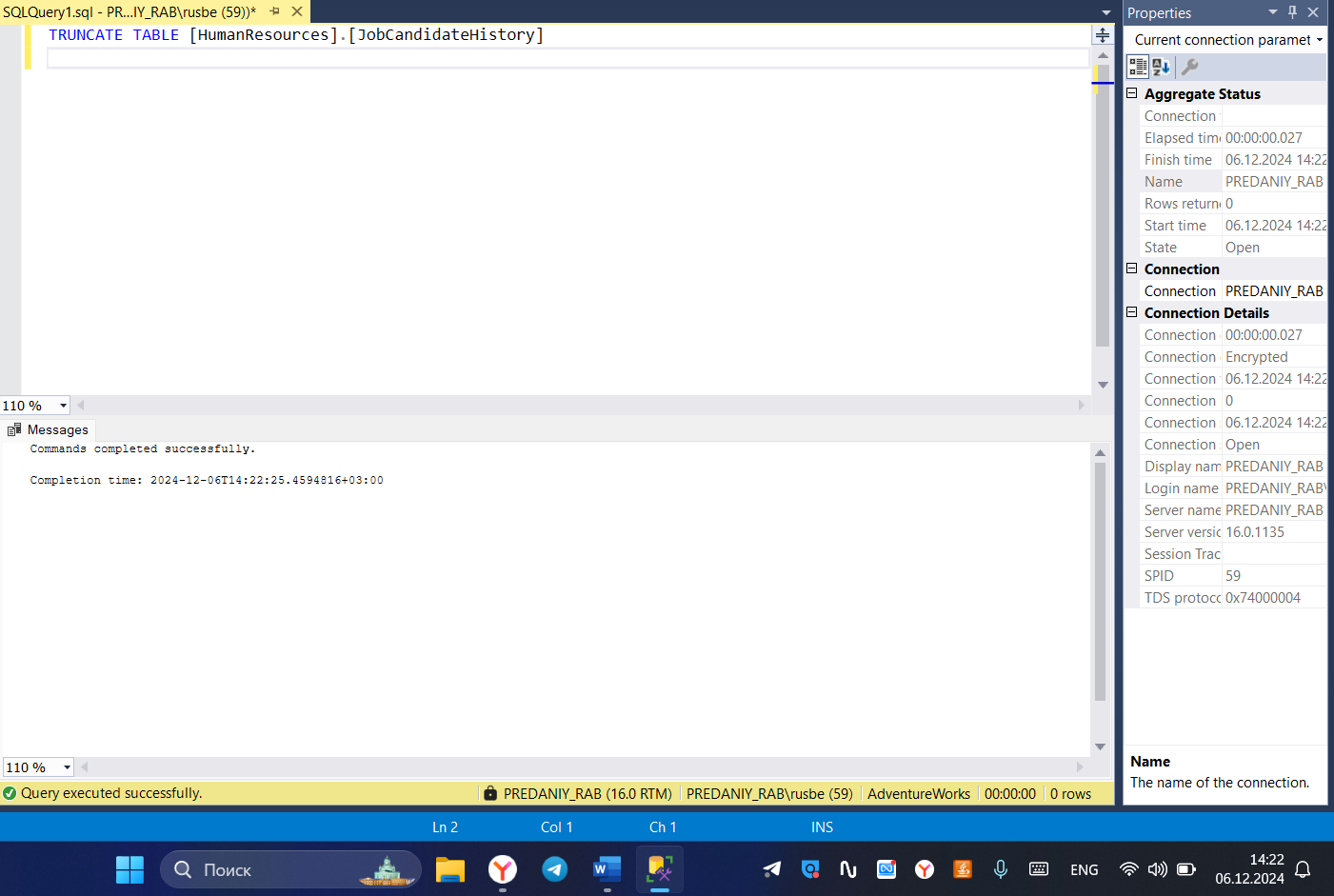


Рисунок 4 – Результат выполнения запроса

**Упражнение 4 – создание триггера на обновление и вставку**

Запрос 1: Создание триггера OrderDetailNotDiscontinued на таблицу Sales.SalesOrderDetail. Этот триггер отвергает попытки ввода заказов на товары, прием которых на склад прекращен. Информация о прекращении поставок товара находится в таблице Production.Product. Если поставки товара прекращены, то значение поля DiscontinuedDate будет иметь значение, отличное от NULL. При попытке заказать такой товар триггер выдает сообщение с помощью команды RAISERROR и откатывает транзакцию.

CREATE TRIGGER Sales.OrderDetailNotDiscontinued

ON Sales.SalesOrderDetail

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS (

SELECT 1

FROM INSERTED i

JOIN Production.Product p ON i.ProductID = p.ProductID

WHERE p.DiscontinuedDate IS NOT NULL

)

BEGIN

RAISERROR ('Ошибка, поставки данного товара прекращены.', 16, 1);

ROLLBACK TRANSACTION;

END

END;

GO

Результат выполнения запроса представлен на Рисунке 5.

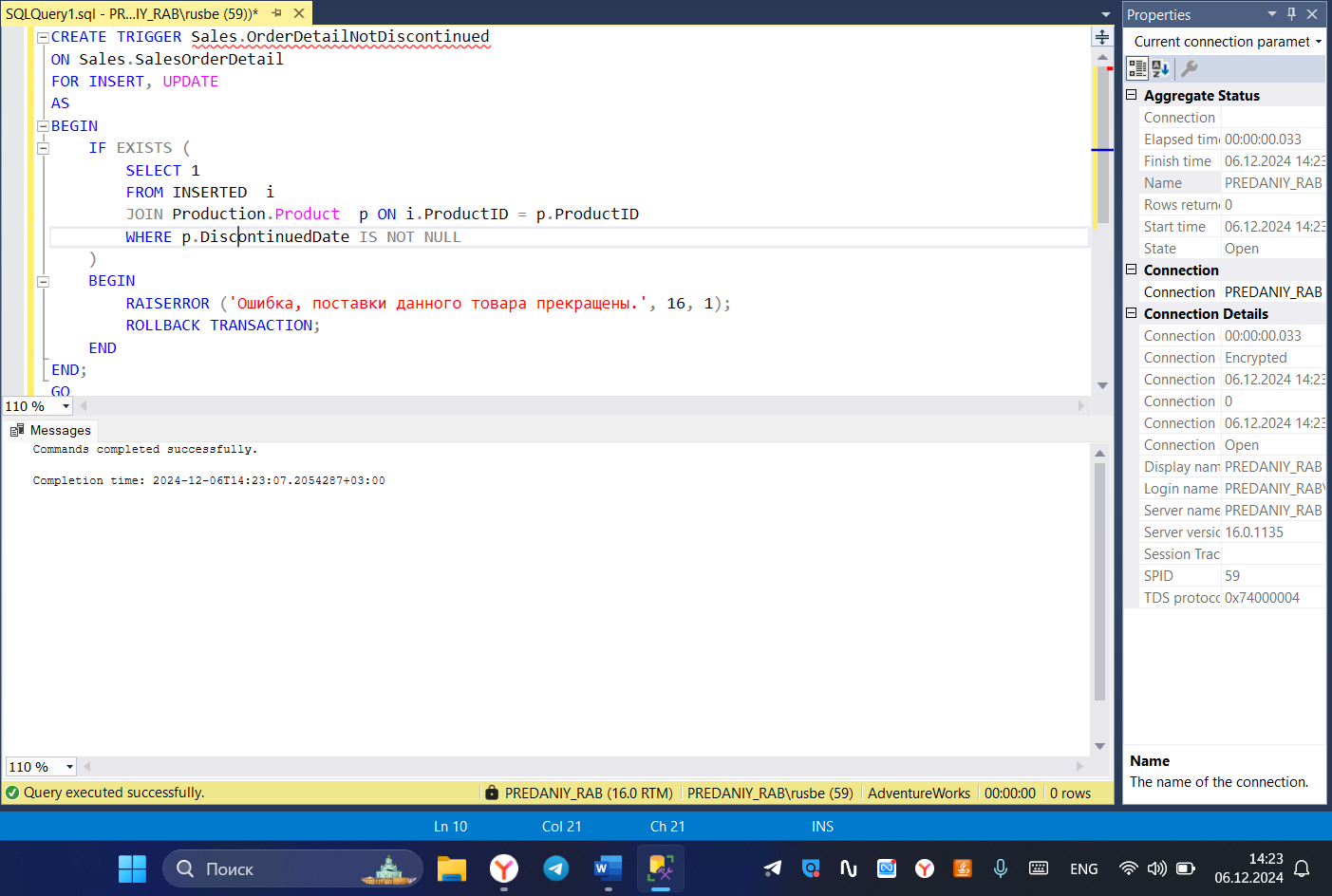


Рисунок 5 – Результат выполнения запроса

Запрос 2: Для проверки триггера в таблицу Production.Product необходимо ввести данные хотя бы об одном товаре, поставка которого прекращена. Проверим, есть ли подходящие данные в таблице Product.

USE AdventureWorks

GO

SELECT ProductID, Name FROM Production.Product

WHERE DiscontinuedDate IS NOT NULL

Результат выполнения запроса представлен на Рисунке 6.

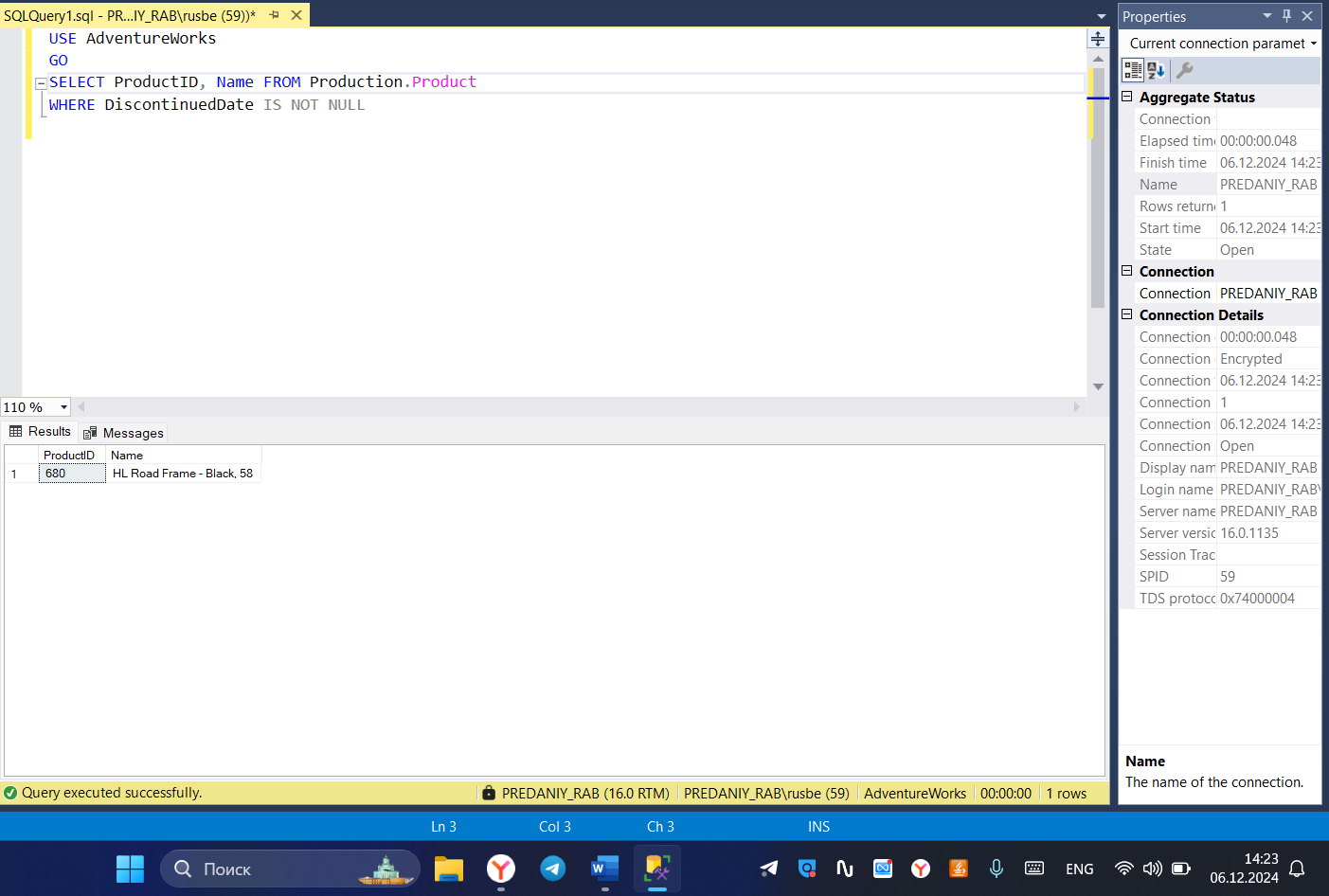


Рисунок 6 – Результат выполнения запроса

Запрос 3: Так как данные отсутствуют, введем в строку изменения.

UPDATE Production.Product

SET DiscontinuedDate = GETDATE()

WHERE ProductID = 680

Результат выполнения запроса представлен на Рисунке 7.

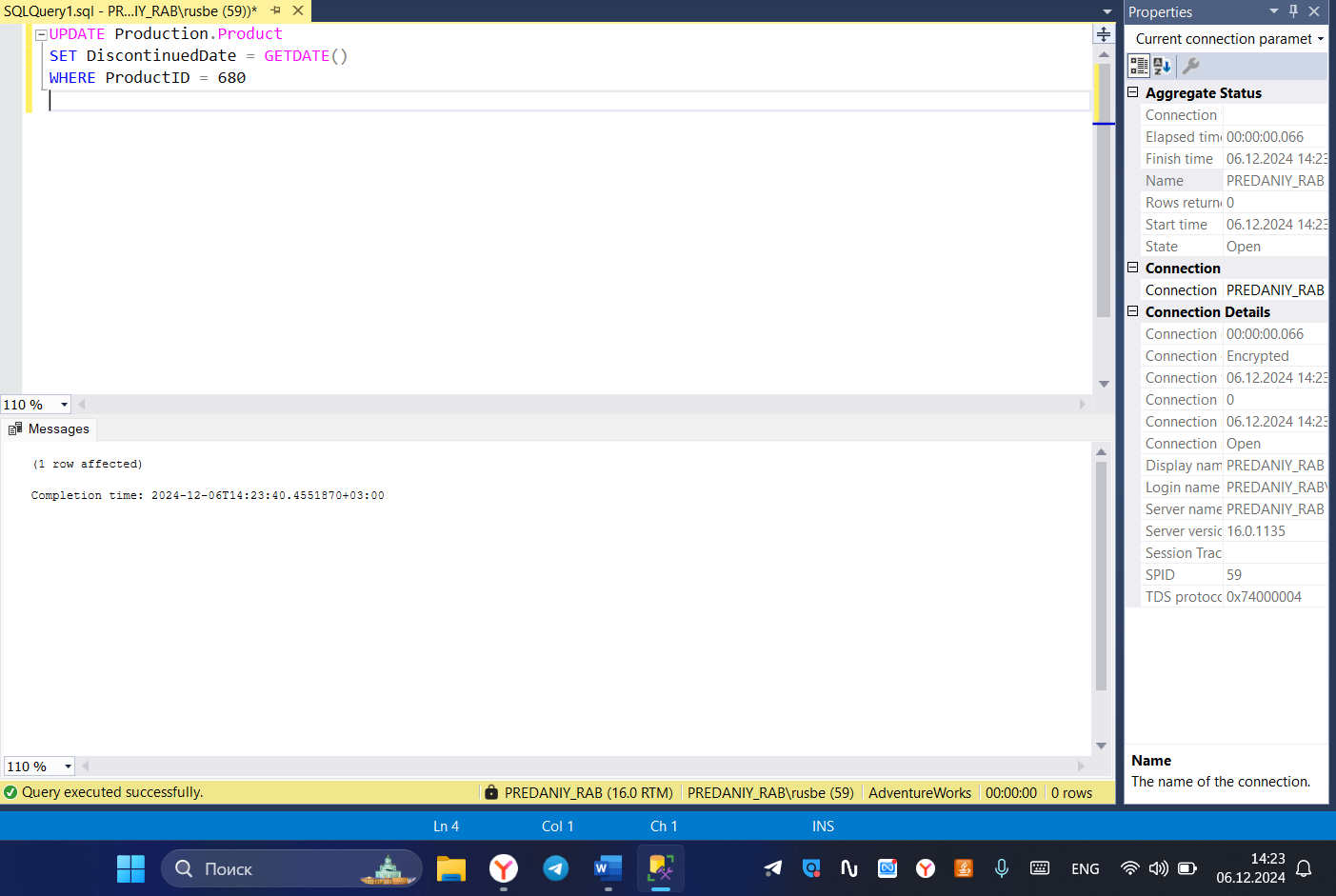


Рисунок 7 – Результат выполнения запроса

Запрос 4: Проверка работы триггера. Попытка ввода недопустимых данных должна быть отвергнута.

INSERT Sales.SalesOrderDetail

(SalesOrderID, OrderQty, ProductID, SpecialOfferID, UnitPrice, UnitPriceDiscount)

VALUES (43660, 5, 680, 1, 1431, 0)

Результат выполнения запроса представлен на Рисунке 8.

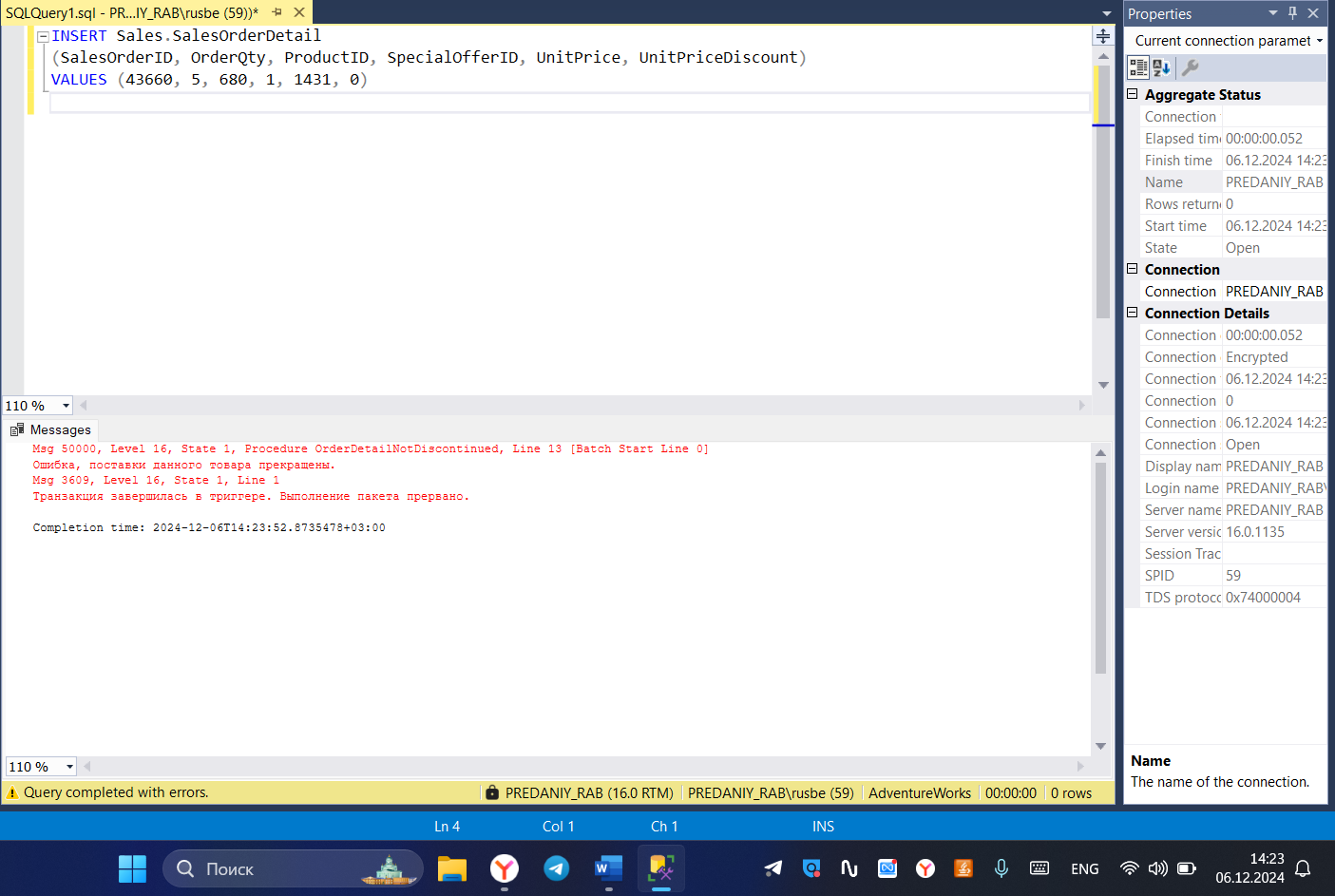


Рисунок 8 – Результат выполнения запроса

**Вывод**

В ходе лабораторной работы были изучены триггеры — объекты базы данных, которые автоматически выполняют определённые действия в ответ на изменения в таблицах или представлениях. Каждый триггер связан с конкретной таблицей и позволяет осуществлять обработку данных в моменты вставки, обновления или удаления записей.

В процессе работы использовались виртуальные таблицы INSERTED и DELETED, которые обеспечивают доступ к данным, добавленным или удалённым в ходе выполнения триггера. Это позволяет реализовывать различные проверки и обработки данных для поддержания их целостности.

Также была подробно рассмотрена команда RAISERROR, которая служит для генерации сообщений об ошибках, информируя разработчиков и пользователей о проблемах, возникших в процессе выполнения триггера. Важной частью работы с триггерами является команда ROLLBACK TRANSACTION, позволяющая откатить изменения при возникновении ошибок, тем самым предотвращая внесение недопустимых данных в базу.

Таким образом, использование триггеров в сочетании с эффективными механизмами обработки ошибок способствует обеспечению целостности и безопасности данных в системах управления базами данных.