**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Систем автоматизированного проектирования**

**отчет**

**по лабораторной работе №8**

**по дисциплине «Базы данных»**

**Тема: Создание и использование хранимых процедур**

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

Санкт-Петербург

2024

**Цель работы**

Целью работы является научиться писать и применять хранимые процедуры. В работе будет использоваться база данных AdventureWorks.

**Выполнение работы**

**Упражнение 1 – создание хранимой процедуры без параметров**

Запрос 1: Создание хранимой процедуры GetDiscounts в схеме Sales, которая извлекает следующие столбцы из таблицы Sales.Specialoffer: Description, DiscountPct, Туре, Category, StartDate, EndDate, MinQty и MaxQty, с сортировкой по параметрам MinQty и MaxQty.

USE AdventureWorks

GO

CREATE PROCEDURE Sales.GetDiscounts

AS

BEGIN

SET NOCOUNT ON

SELECT Description, DiscountPct, Type, Category, StartDate, EndDate, MinQty, MaxQty

FROM Sales.SpecialOffer

ORDER BY MinQty, MaxQty ASC

END

GO

Результат выполнения запроса представлен на Рисунке 1.

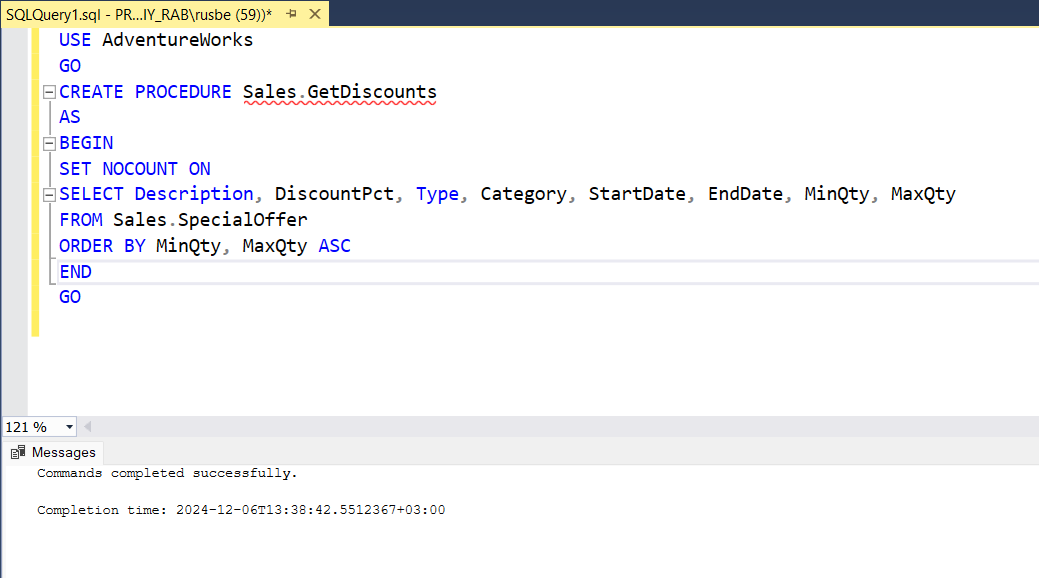


Рисунок 1 – Результат выполнения запроса

Запрос 2: Проверка хранимой процедуры.

EXEC Sales.GetDiscounts

Результат выполнения запроса представлен на Рисунке 2.

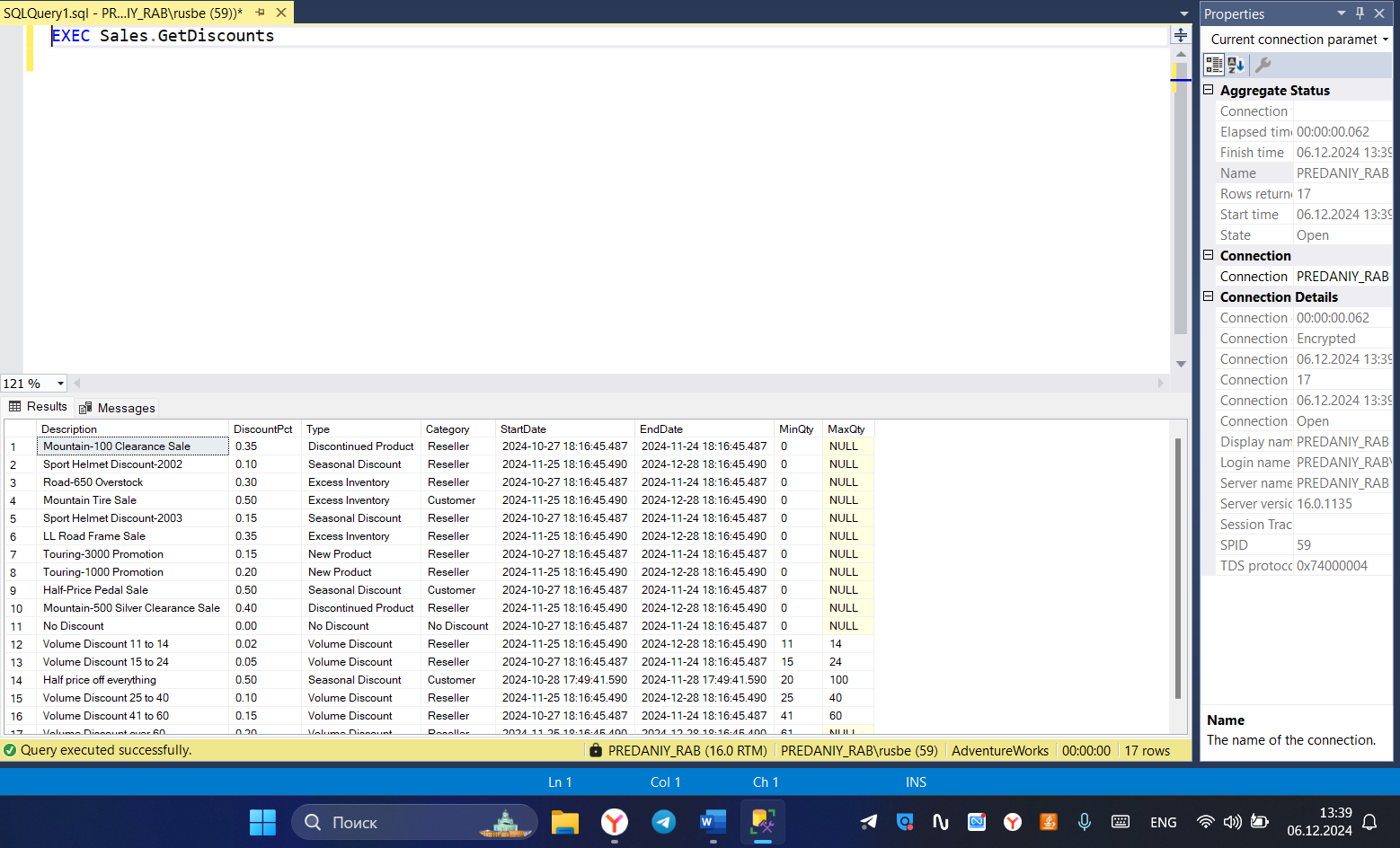


Рисунок 2 – Результат выполнения программы

**Упражнение 2 – создание хранимой процедуры с параметром**

Запрос 1: Создание хранимой процедуры GetDiscountsForCategory в схеме Sales с входным параметром @Category, который имеет тип данных nvarchar и принимает до 50 символов, с фильтрацией строк на основе @Category.

CREATE PROC Sales.GetDiscountsForCategory @Category nvarchar(50)

AS

SELECT Description, DiscountPct, Type, Category, StartDate, EndDate, MinQty, MaxQty

FROM Sales.SpecialOffer

ORDER BY Category DESC

GO

Результат выполнения запроса представлен на Рисунке 3.

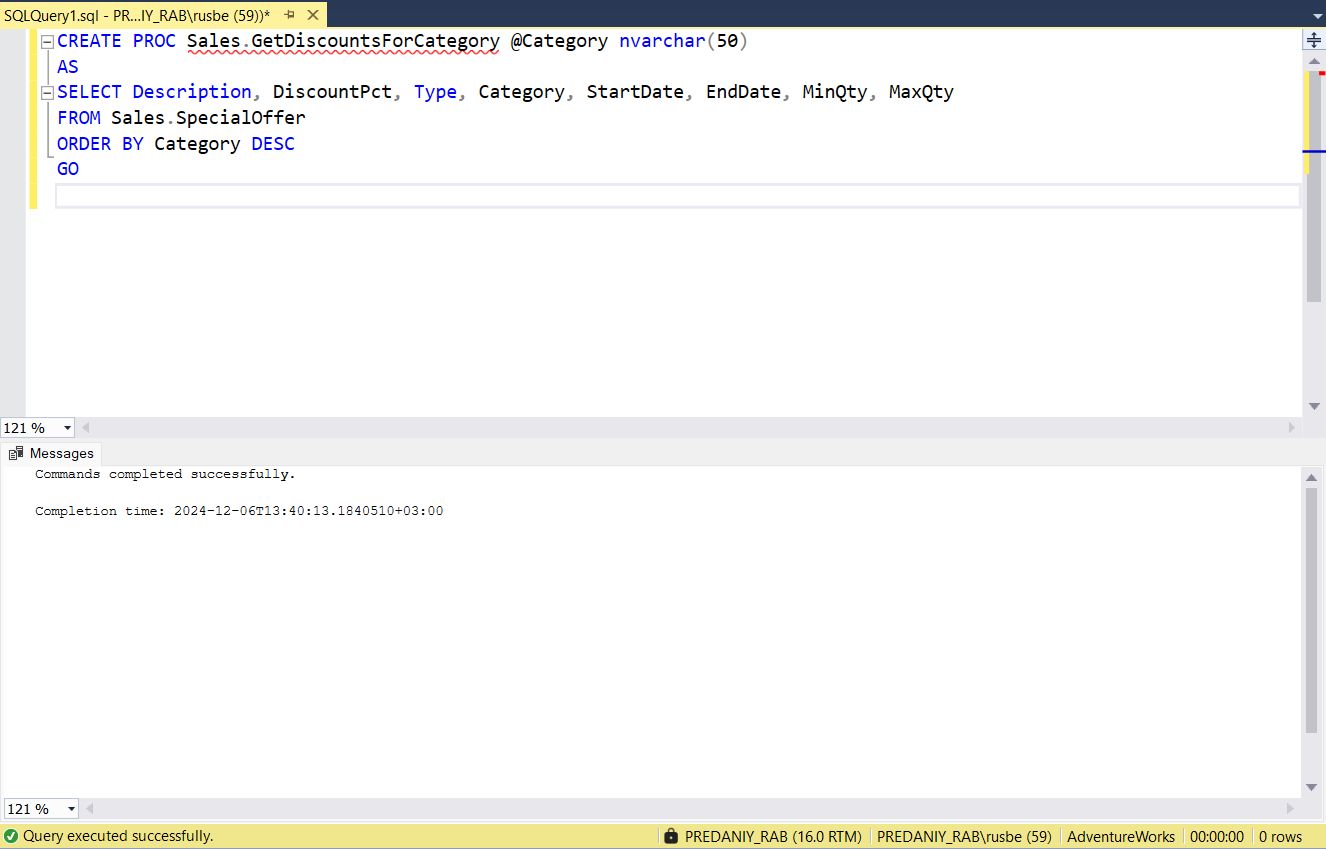


Рисунок 3 – Результат выполнения запроса

Запрос 2: Проверка хранимой процедуры.

EXEC Sales.GetDiscounts ‘Reseller’

Результат выполнения запроса представлен на Рисунке 4.

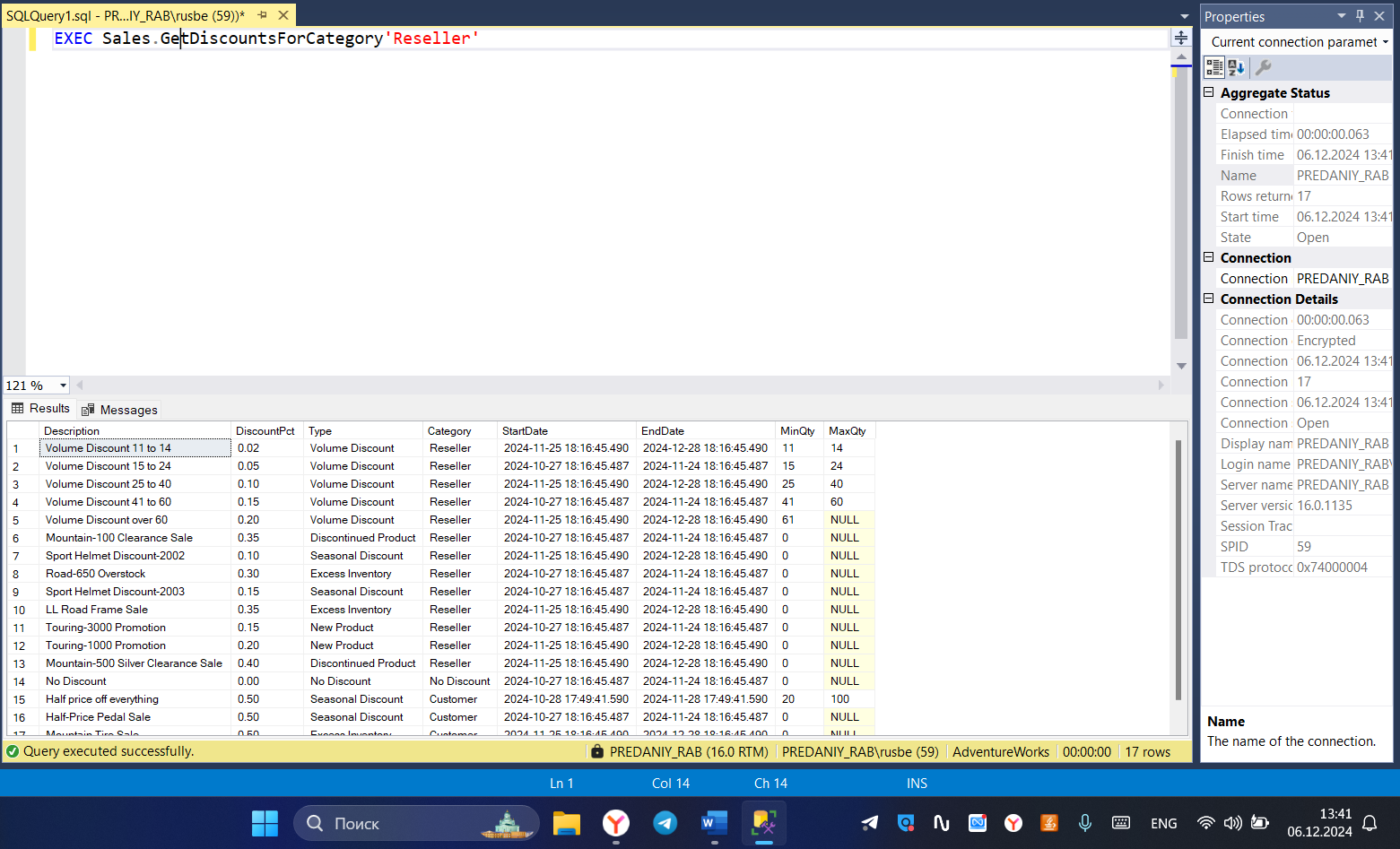


Рисунок 4 – Результат выполнения запроса

**Упражнение 3 – создание хранимой процедуры с параметрами и значениями по умолчанию**

Запрос 1: Создание хранимой процедуры GetDiscountsForCategoryAndDate в схеме Sales, имеющей кроме параметра @Categorу параметр @DateToCheck datetime, который может принимать стандартное значение NULL. Если значение NULL указано для параметра @DateToCheck, то для этого параметра задается значение текущих даты и времени с использованием функции GETDATE().

CREATE PROC Sales.GetDiscountsForCategoryAndDate

@Category AS nvarchar(50),

@DateToCheck AS datetime = NULL

AS

IF (@DateToCheck IS NULL)

BEGIN

SET @DateToCheck = GETDATE();

END

SELECT Description, DiscountPct, Type, Category, StartDate, EndDate, MinQty, MaxQty

FROM Sales.SpecialOffer

WHERE Category = @Category AND StartDate <= @DateToCheck

GO

Результат выполнения запроса представлен на Рисунке 5.

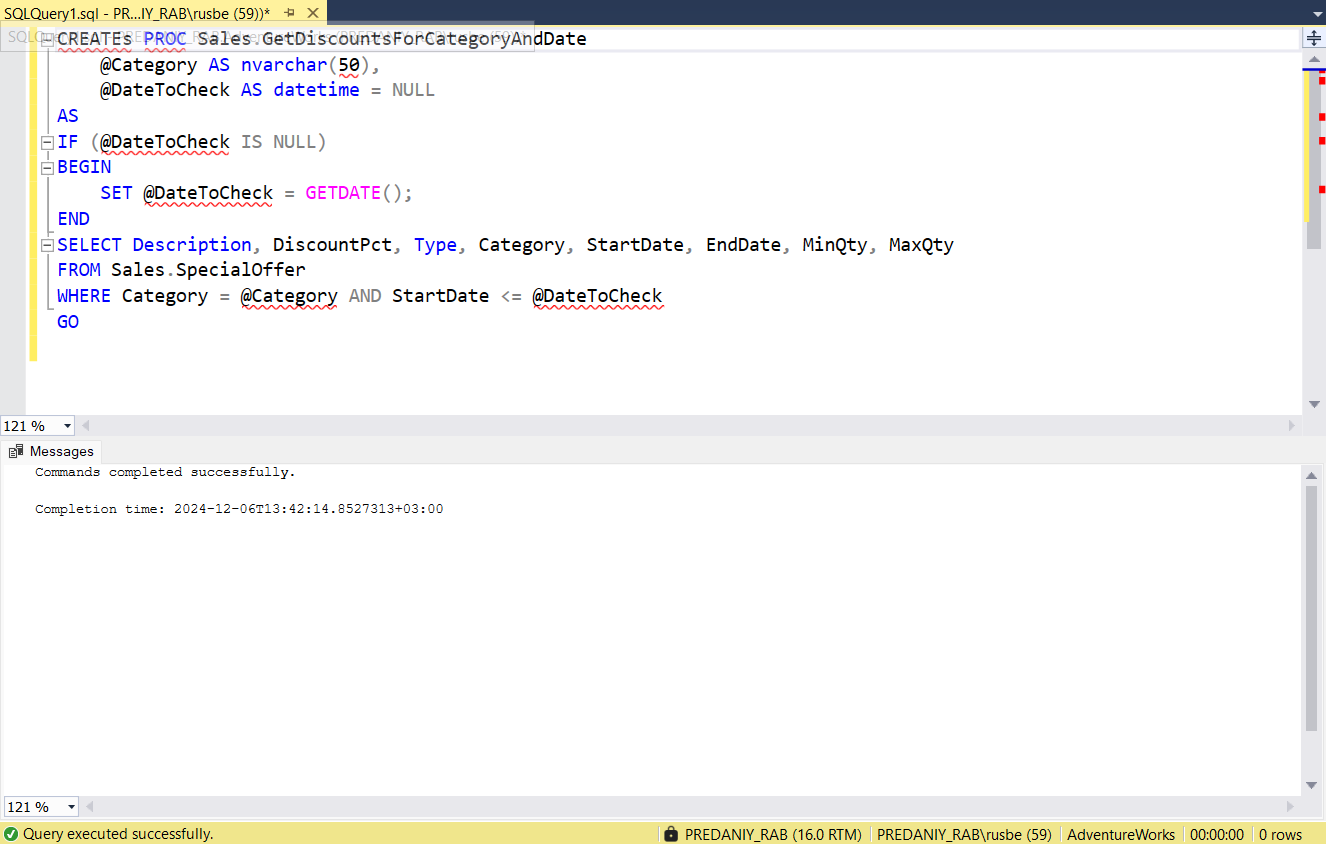


Рисунок 5 – Результат выполнения запроса

Запрос 2: Проверка хранимой процедуры.

EXEC Sales.GetDiscountsForCategoryAndDate 'Reseller'

Результат выполнения запроса представлен на Рисунке 6.

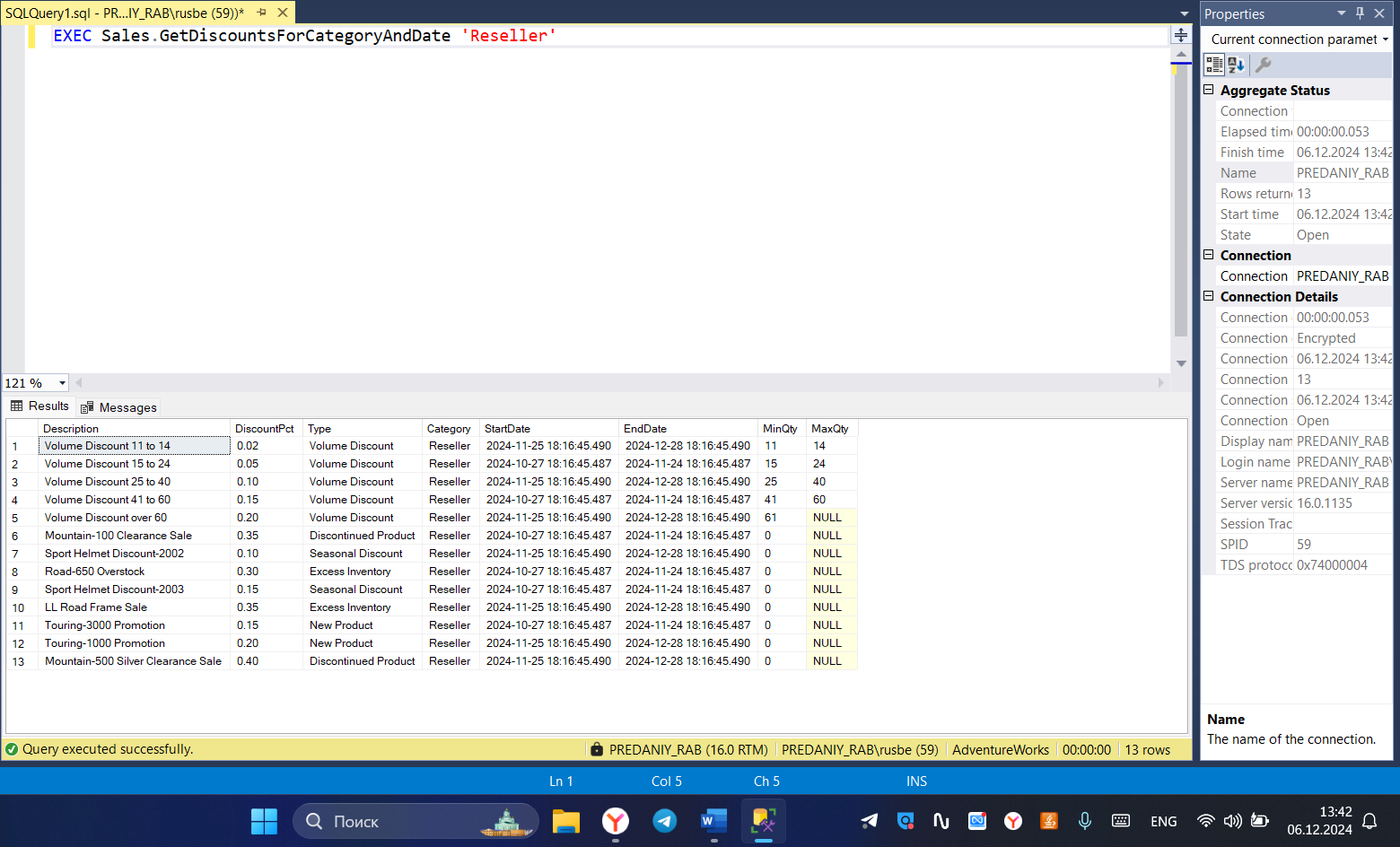


Рисунок 6 – Результат выполнения запроса

Запрос 3: Проверка варианта, когда указываются оба параметра.

DECLARE @DateToCheck datetime

SET @DateToCheck = DATEADD(MONTH, 1, GETDATE())

EXEC Sales.GetDiscountsForCategoryAndDate 'Reseller', @DateToCheck

Результат выполнения запроса представлен на Рисунке 7.

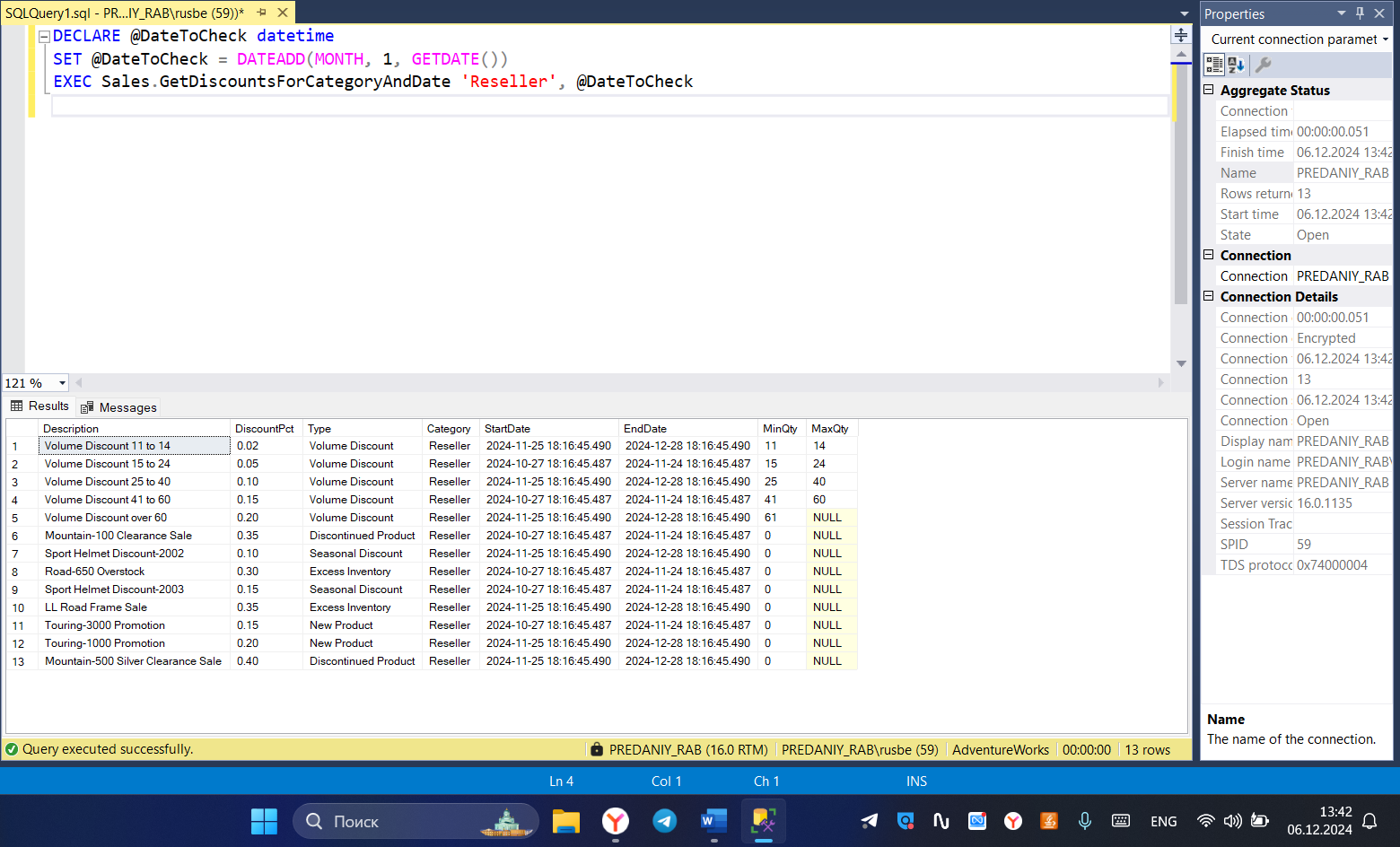


Рисунок 7 – Результат выполнения запроса

**Упражнение 4 – создание хранимой процедуры с входными и выходными параметрами**

Запрос 1: Создание хранимой процедуры AddDiscount в схеме Sales, которая вставляет новые записи в таблицу Sales.SpecialOffer. Команда INSERT защищена обработкой ошибок с использованием конструкции Try…Catch, и любые ошибки записываются в таблицу dbo.ErrorLog. Если новая вставка завершится успешно, параметр @NewProductID обновляется на значение функции SCOPE\_IDENTITY.

CREATE PROC Sales.AddDiscount

@Description AS nvarchar(255),

@DiscountPct AS smallmoney,

@Type AS nvarchar(50),

@Category AS nvarchar(50),

@StartDate AS Datetime,

@EndDate AS Datetime,

@MinQty AS int,

@MaxQty AS int,

@NewProductID AS int OUTPUT

AS

BEGIN

BEGIN TRY

INSERT INTO Sales.SpecialOffer (Description, DiscountPct, Type, Category, StartDate, EndDate, MinQty, MaxQty)

VALUES (@Description, @DiscountPct, @Type, @Category, @StartDate, @EndDate, @MinQty, @MaxQty);

SET @NewProductID = SCOPE\_IDENTITY()

RETURN 0

END TRY

BEGIN CATCH

INSERT INTO dbo.ErrorLog (ErrorMessage, ErrorLine, ErrorNumber, ErrorProcedure)

VALUES (ERROR\_MESSAGE(), ERROR\_LINE(), ERROR\_NUMBER(), ERROR\_PROCEDURE())

RETURN 1

END CATCH;

END

Результат выполнения запроса представлен на Рисунке 8.

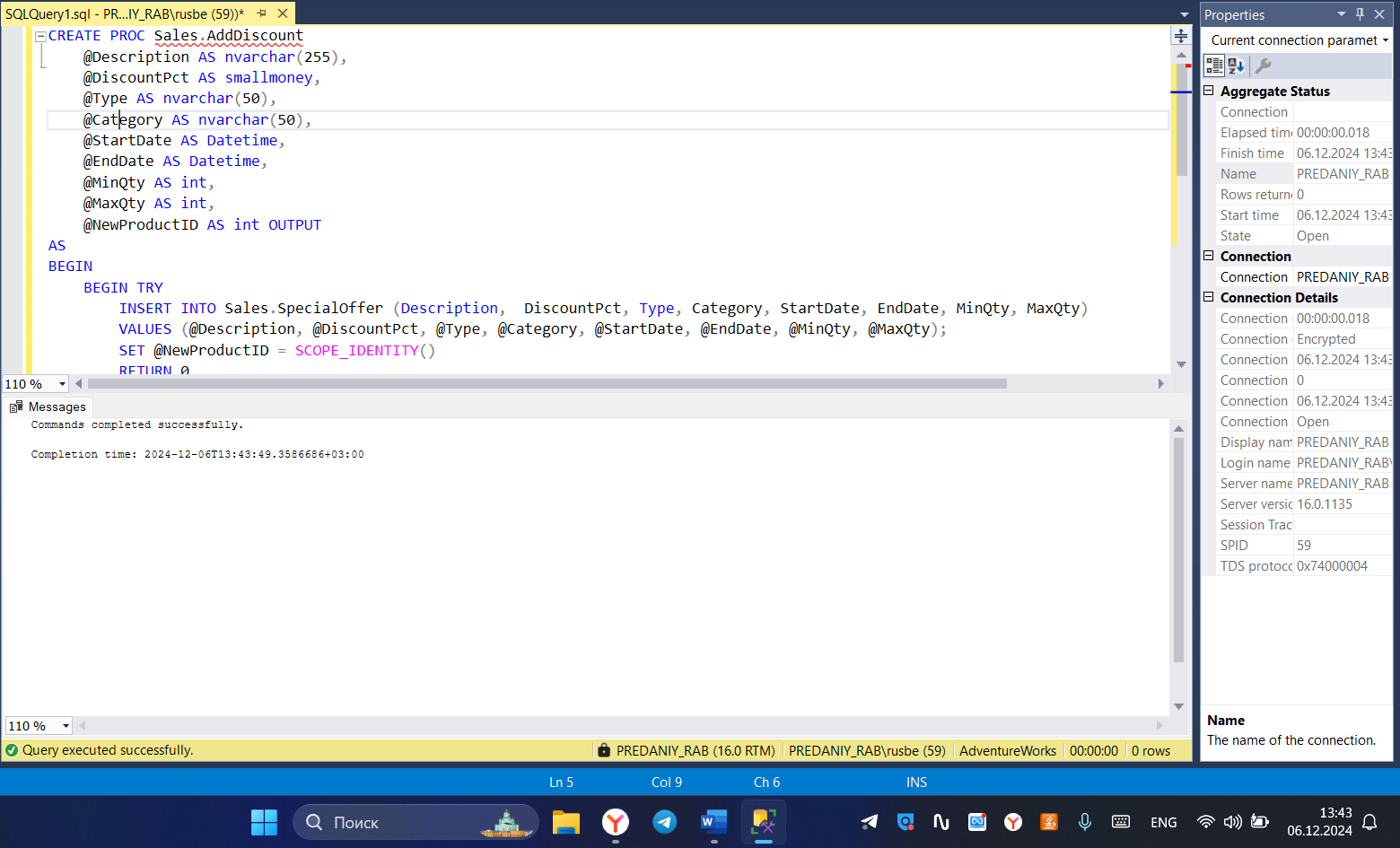


Рисунок 8 – Результат выполнения запроса

Запрос 2: Проверка хранимой процедуры.

DECLARE @StartDate datetime, @EndDate datetime

SET @StartDate = GETDATE()

SET @EndDate = DATEADD(month, 1, @StartDate)

DECLARE @NewId int

EXEC Sales.AddDiscount

'Half price off everything',

0.5,

'Seasonal Discount',

'Customer',

@StartDate,

@EndDate,

20,

100,

@NewId OUTPUT

SELECT @NewId

Результат выполнения запроса представлен на Рисунке 9.

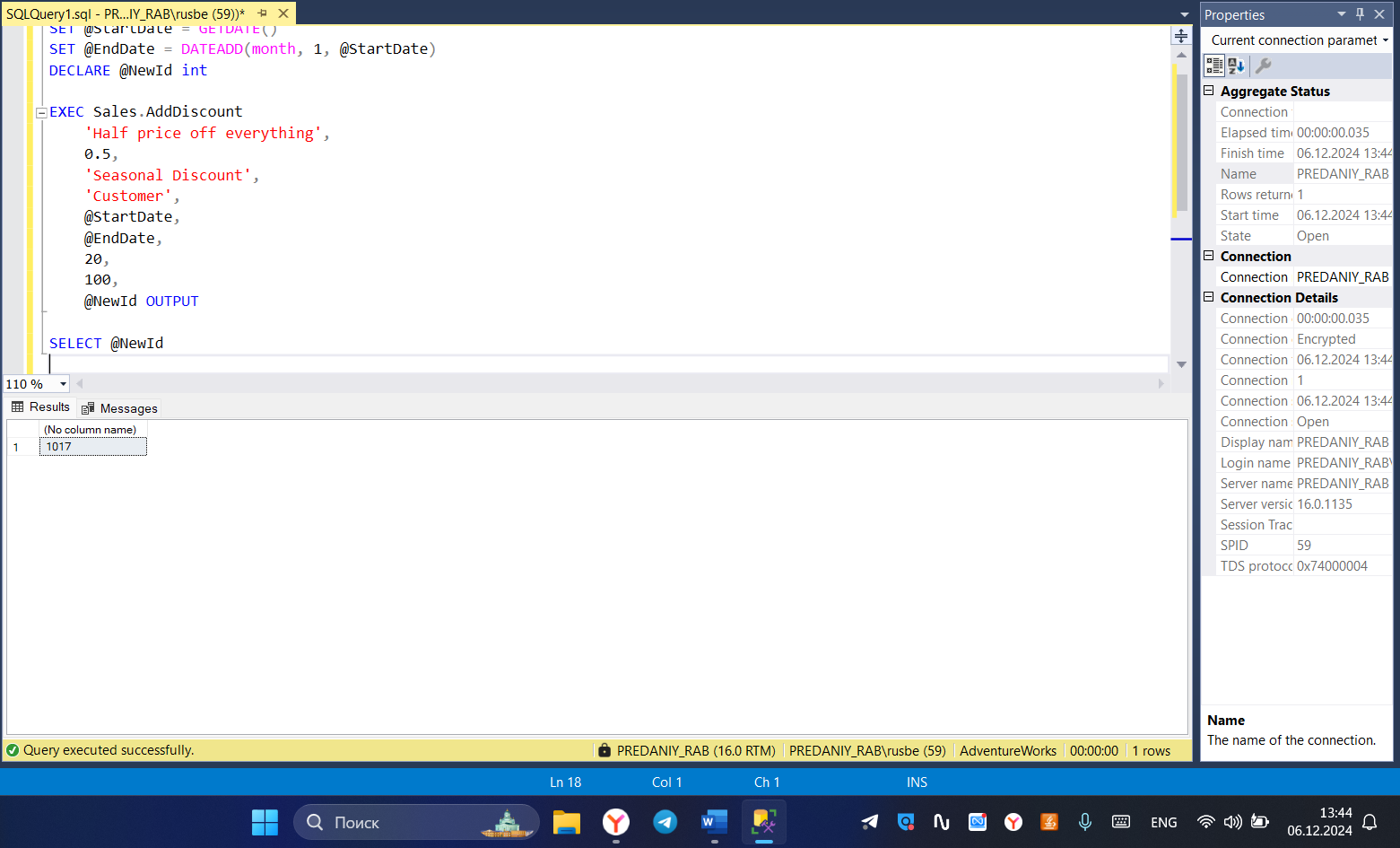


Рисунок 9 – Результат выполнения запроса

Запрос 3: Повторная проверка с использованием некорректных значений.

DECLARE @StartDate datetime, @EndDate datetime

SET @StartDate = GETDATE()

SET @EndDate = DATEADD(month, 1, @StartDate)

DECLARE @NewId int, @ReturntValue int

EXEC @ReturntValue = Sales.AddDiscount

'Half price off everything',

-0.5,

'Seasonal Discount',

'Customer',

@StartDate,

@EndDate,

0,

20,

@NewId OUTPUT

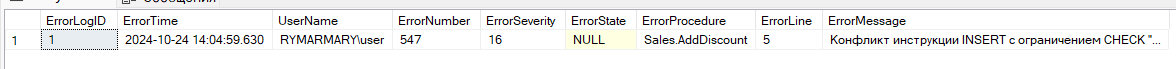
IF (@ReturntValue = 0)

SELECT @NewId

ELSE

SELECT TOP 1 \* FROM dbo.ErrorLog ORDER BY ErrorTime DESC

Результат выполнения запроса представлен на Рисунке 10.



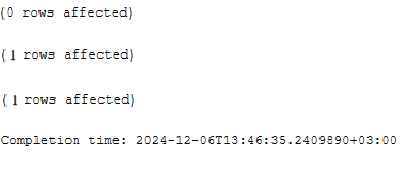


Рисунок 10 – Результат выполнения запроса

**Вывод**

В ходе выполнения лабораторной работы мы познакомились с созданием и использованием хранимых процедур на примере базы данных AdventureWorks. Были рассмотрены различные типы хранимых процедур:

1. **Процедуры без параметров** — для выполнения стандартных запросов без передачи дополнительных данных.
2. **Процедуры с параметрами** — позволяющие использовать входные параметры для фильтрации и отбора данных.
3. **Процедуры с параметрами по умолчанию** — дающие возможность задавать значения параметров по умолчанию, что упрощает использование процедуры.
4. **Процедуры с входными и выходными параметрами** — для обработки данных и возврата идентификаторов новых записей при вставке данных.

Кроме того, мы изучили механизмы обработки ошибок в хранимых процедурах с использованием конструкции TRY...CATCH. Это позволило эффективно управлять ошибками и, в случае их возникновения, сохранять информацию об ошибках в отдельной таблице.

Таким образом, данная лабораторная работа продемонстрировала, как с помощью хранимых процедур можно упростить выполнение сложных операций с данными, повысить безопасность и удобство работы с базой данных, а также улучшить производительность за счёт сокращения сетевых запросов и оптимизации обработки данных на стороне сервера.