Технология разработки и защиты баз данных

Отчет

Лабораторная работа №3

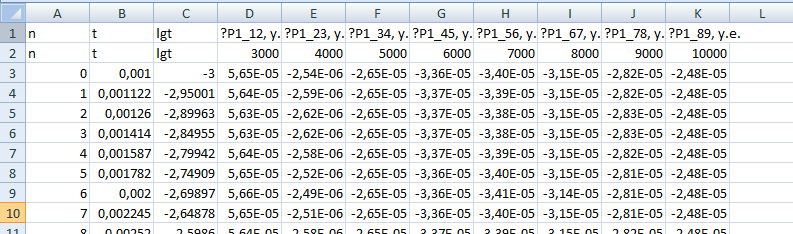
Шонхоров Кирилл Олегович 007а1

03.04.2023

1. Импортировать данные из файла текстового и эксель (Файл: 2023.xsls; Страница: “2000 250 200 D PP” Колонки: n, t, lgt, δP1\_12, у.е., δP1\_23, у.е., δP1\_34, у.е., δP1\_45, у.е., δP1\_56, у.е., δP1\_67, у.е., δP1\_78, у.е., δP1\_89, у.е.) в БД, используя мастер MS SQL Server:

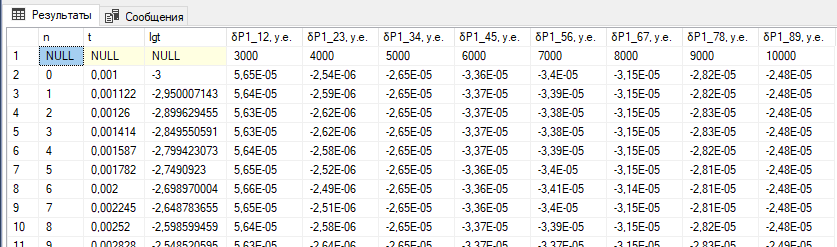
1.1 Создать копию ресурсного файла и форматировать под нужный формат БД. Сохранить как “2000 250 200 D PP.xsls”

Результат:



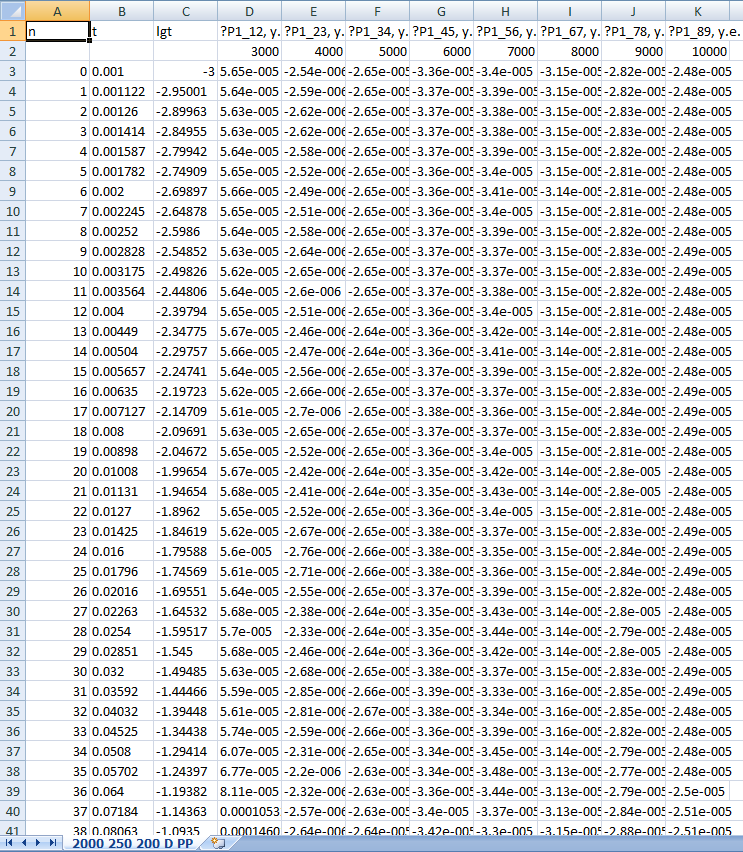
1.2 Используя мастер MS SQL Server импортировать данные в новую таблицу “lab3.2000 250 200 D PP”, выбрав источником Excel файл. Реконструировать таблицу исправив несоответствующее название столбцов.

Результат:



1. Экспортировать данные из БД в файл CSV, используя мастер MS SQL Server:

1.2 Используя мастер MS SQL Server экспортировать данные в новый CSV-файл “2000 250 200 D PP.csv”, выбрав источником SQL Client Server. Реконструировать таблицу исправив несоответствующий формат столбцов.



1. Импортировать данные из файла в БД, используя интерфейс C#:

3.1 Создать простейший интерфейс для выбора CSV-файла с последующим импортом в БД:

3.2 NoUpdateSalary

GO

CREATE TRIGGER NoUpdateSalary

ON Position

instead of UPDATE

AS

UPDATE Position

SET Salary=Salary-5000

WHERE ID\_Pos = (SELECT ID\_Pos FROM inserted)

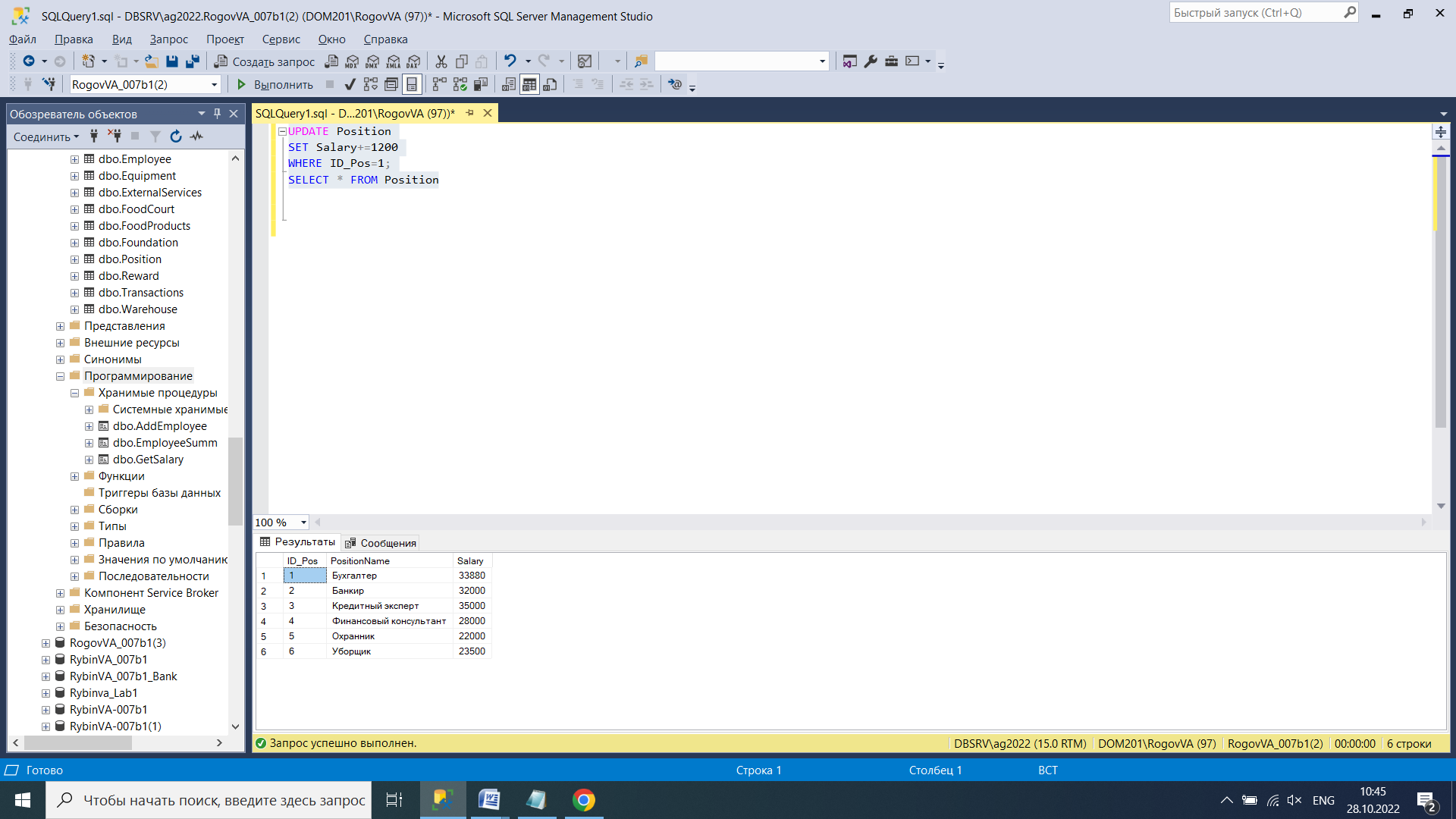
Результат:

UPDATE Position

SET Salary+=1200

WHERE ID\_Pos=1;

SELECT \* FROM Position



1. Разработать триггеры, которые выполняются после команды DELETE и вместо команды DELETE:

4.1 DeleteSalary

GO

CREATE TRIGGER DeleteSalary

ON Position

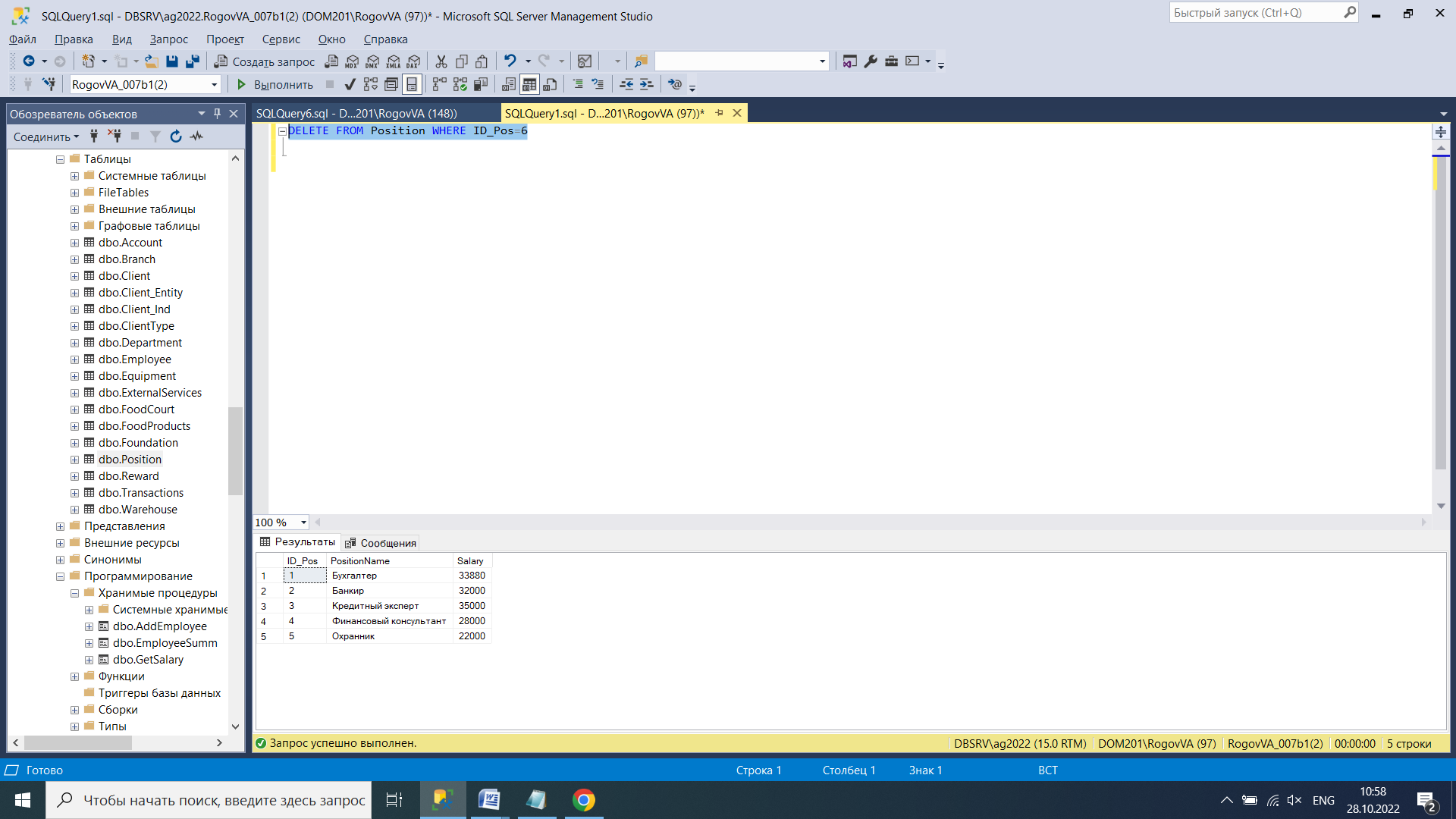
After Delete

AS

Select \* from Position

Результат:

DELETE FROM Position WHERE ID\_Pos=6



4.2 NoDeleteSalary

GO

CREATE TRIGGER NoDeleteSalary

ON Position

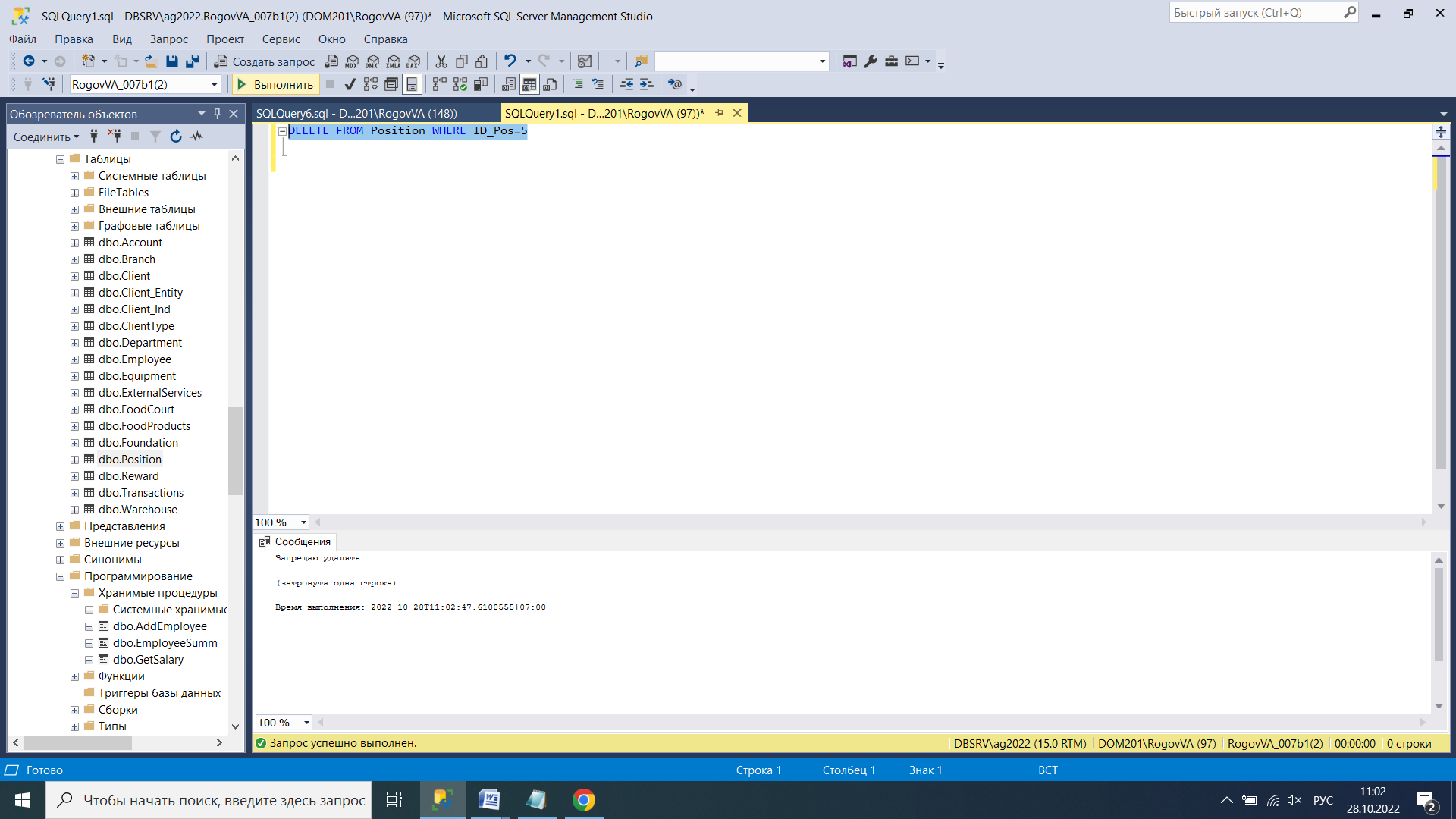
Instead of Delete

AS

Print 'Запрещаю удалять'

Результат:

DELETE FROM Position WHERE ID\_Pos=5



1. Разработать табличную функцию, работающие с данными БД – «Банк - Фонд»:

CREATE FUNCTION dbo.GetSalaryL()

RETURNS @result table

(

ID\_Employee int NOT NULL,

ID\_Pos int NOT NULL,

EmployeeName nvarchar(50) NOT NULL,

Phone nvarchar(11) NOT NULL,

Salary int NOT NULL,

PRIMARY KEY (ID\_Employee,ID\_Pos)

)

AS

BEGIN

INSERT INTO @result

(

ID\_Employee,

ID\_Pos,

EmployeeName,

Phone,

Salary

)

SELECT

Employee.ID\_Employee,

Position.ID\_Pos,

Employee.EmployeeName,

Employee.Phone,

Position.Salary

FROM Employee

INNER JOIN Position ON Position.ID\_Pos= Employee.ID\_Employee

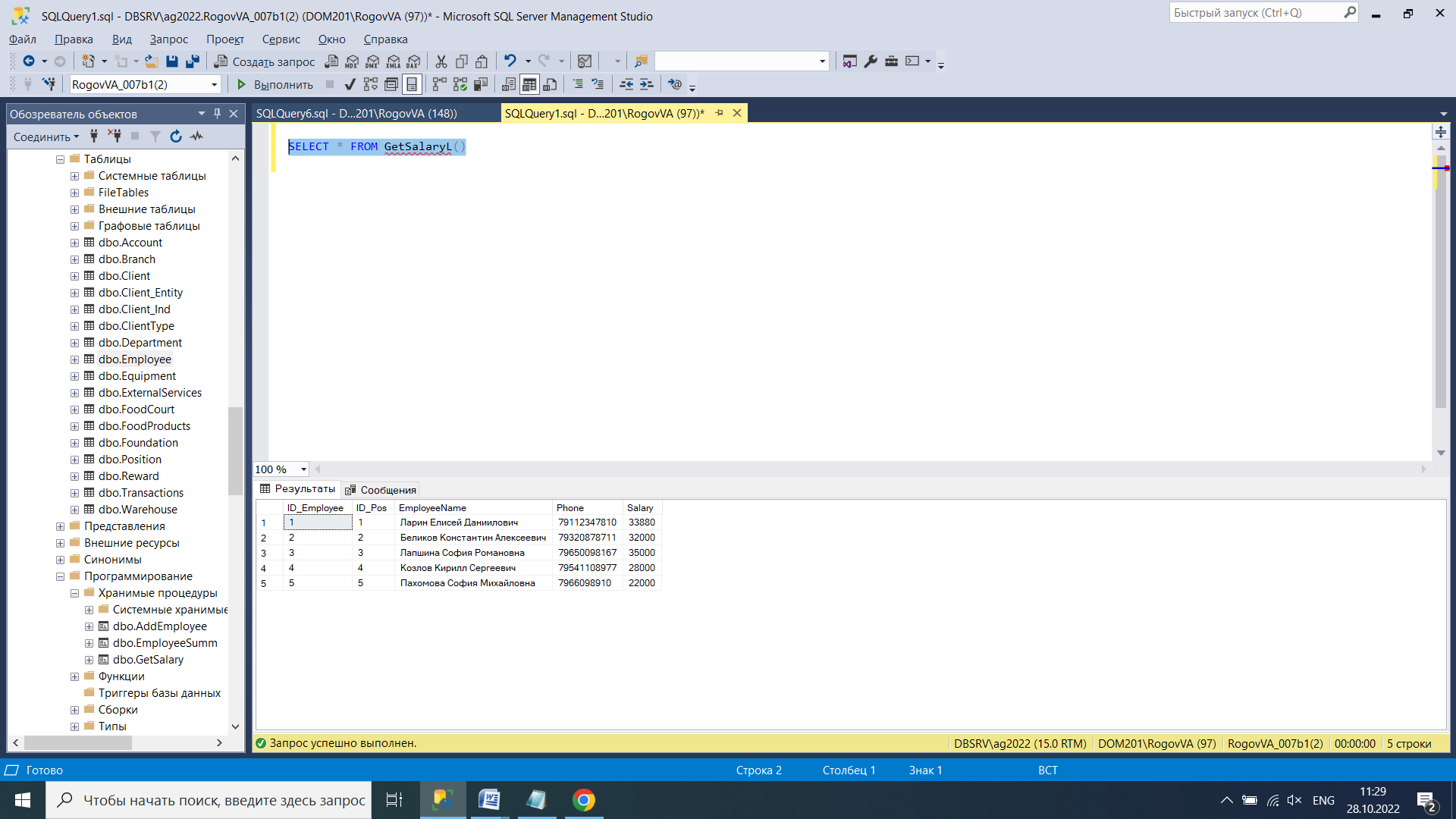
RETURN;

END;

GO

Результат:

SELECT \* FROM GetSalaryL()



6. Разработать скалярную функцию, работающие с данными БД – «Банк - Фонд»:

CREATE FUNCTION dbo.Summary(@num1 INT, @num2 INT)

RETURNS INT AS

BEGIN

DECLARE @Summary INT;

SET @Summary = ISNULL(@num1,0) + ISNULL(@num2,0);

RETURN @Summary;

END;

GO

Результат:

PRINT dbo.Summary(10,2)

