**Практическая работа №1**

**Работа с объектами БД СУБД MS SQL Server**

**с помощью системы администрирования**

**SQL Management Studio 2020**

Цель работы: изучение основных приемов работы с объектами базы данных СУБД SQL Server 2020 с помощью инструмента SQL Management Studio 2020, изучение операторов языка манипулирования данными SQL-DML.

Задачи:

1) Заполнить данными созданную таблицу *Должности* данными колонки *Должность* таблицы *Сотрудники*;

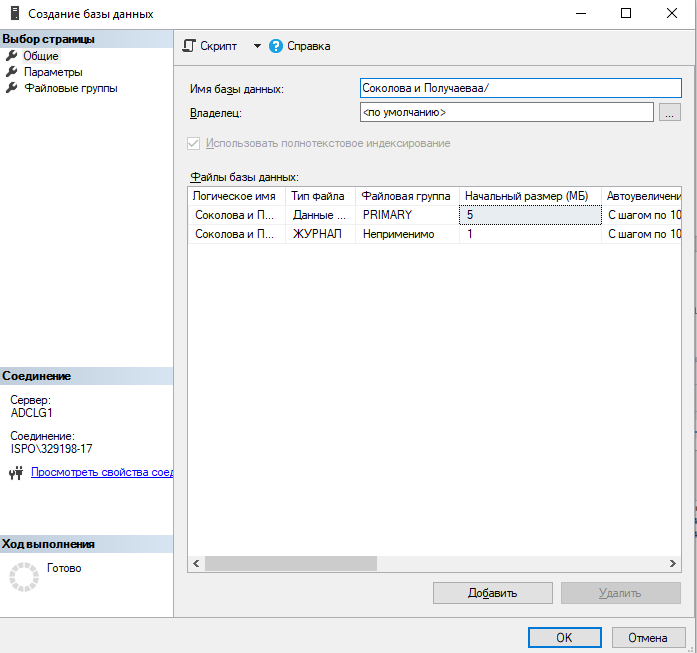
2) Заполнить данными созданную таблицу *Страны* данными соответствующих колонок таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*;

3) Заполнить данными таблицу *Области* данными соответствующих колонок таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*;

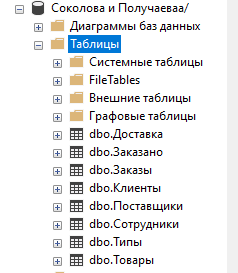
4) Заполнить данными таблицу *Регионы* данными соответствующих колонок

таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*.

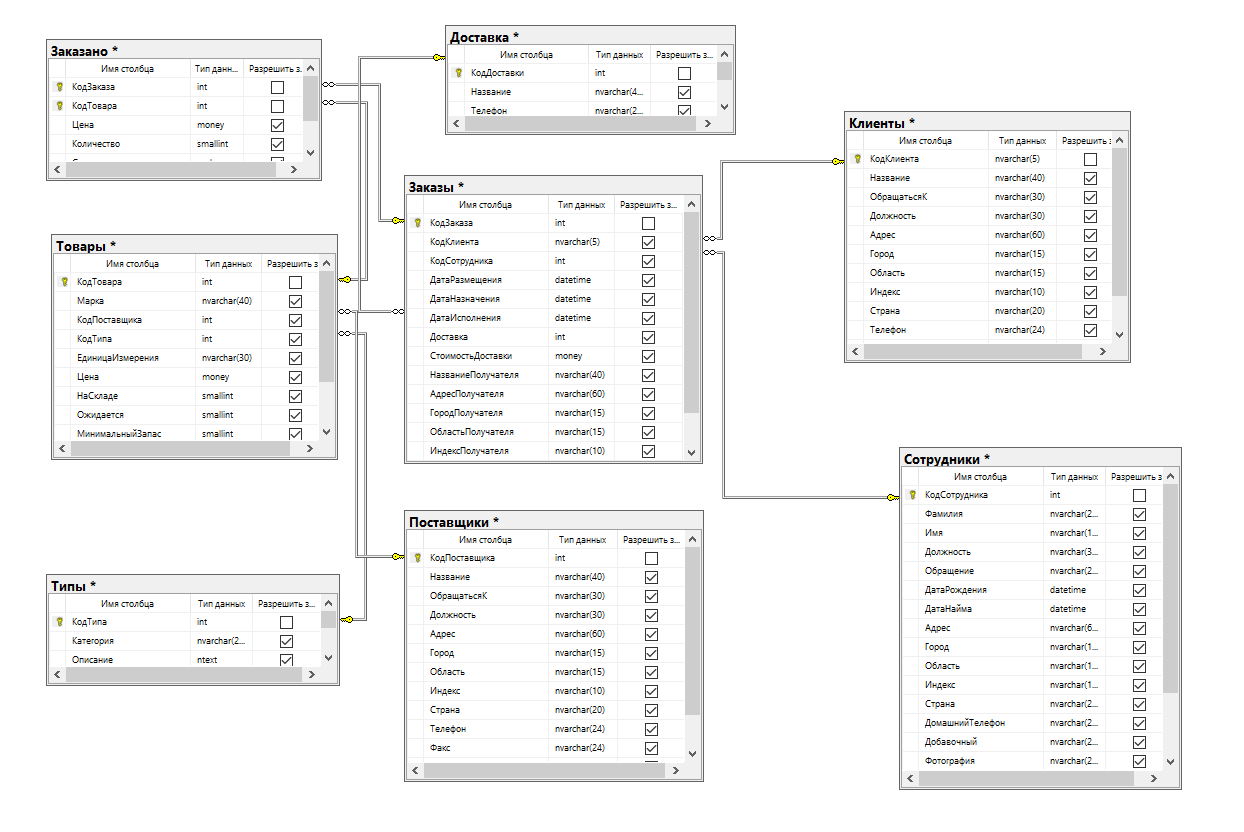
1. Мы создаем базу данных, в которой будем работать.

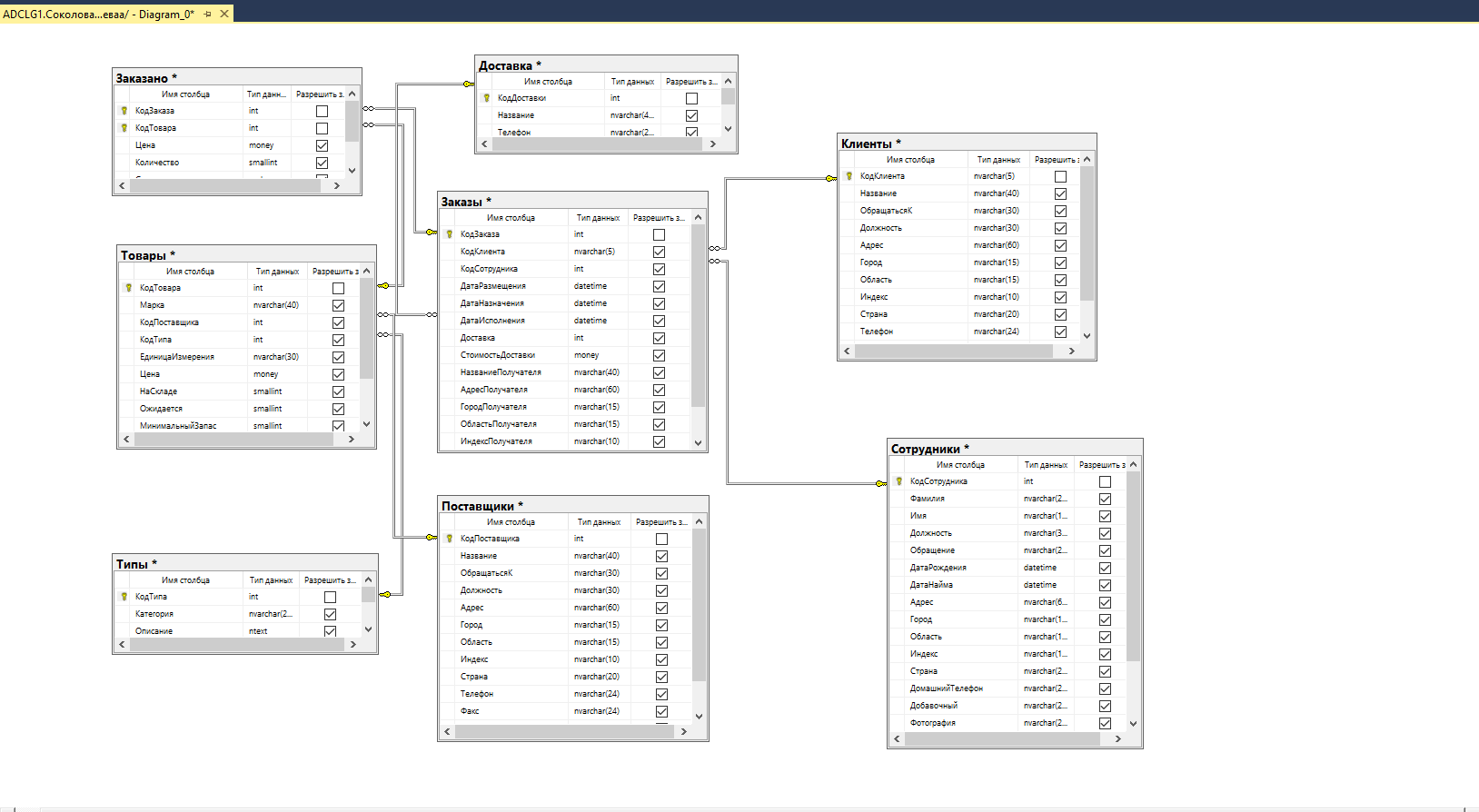


2. Затем переносим БД «Борей» СУБД Access в СУБД SQL Server с помощью вкладки.

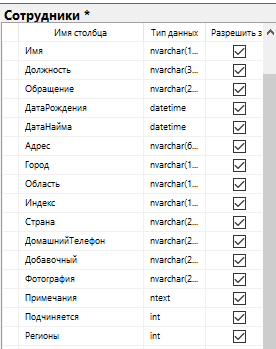


3. После удачного переноса БД «Борей» СУБД Aсcess в нашу БД СУБД SQL Server, приступаем к настройке Диаграммы (схемы) БД.

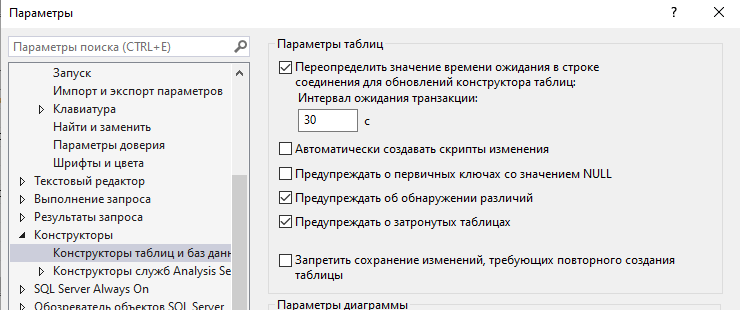




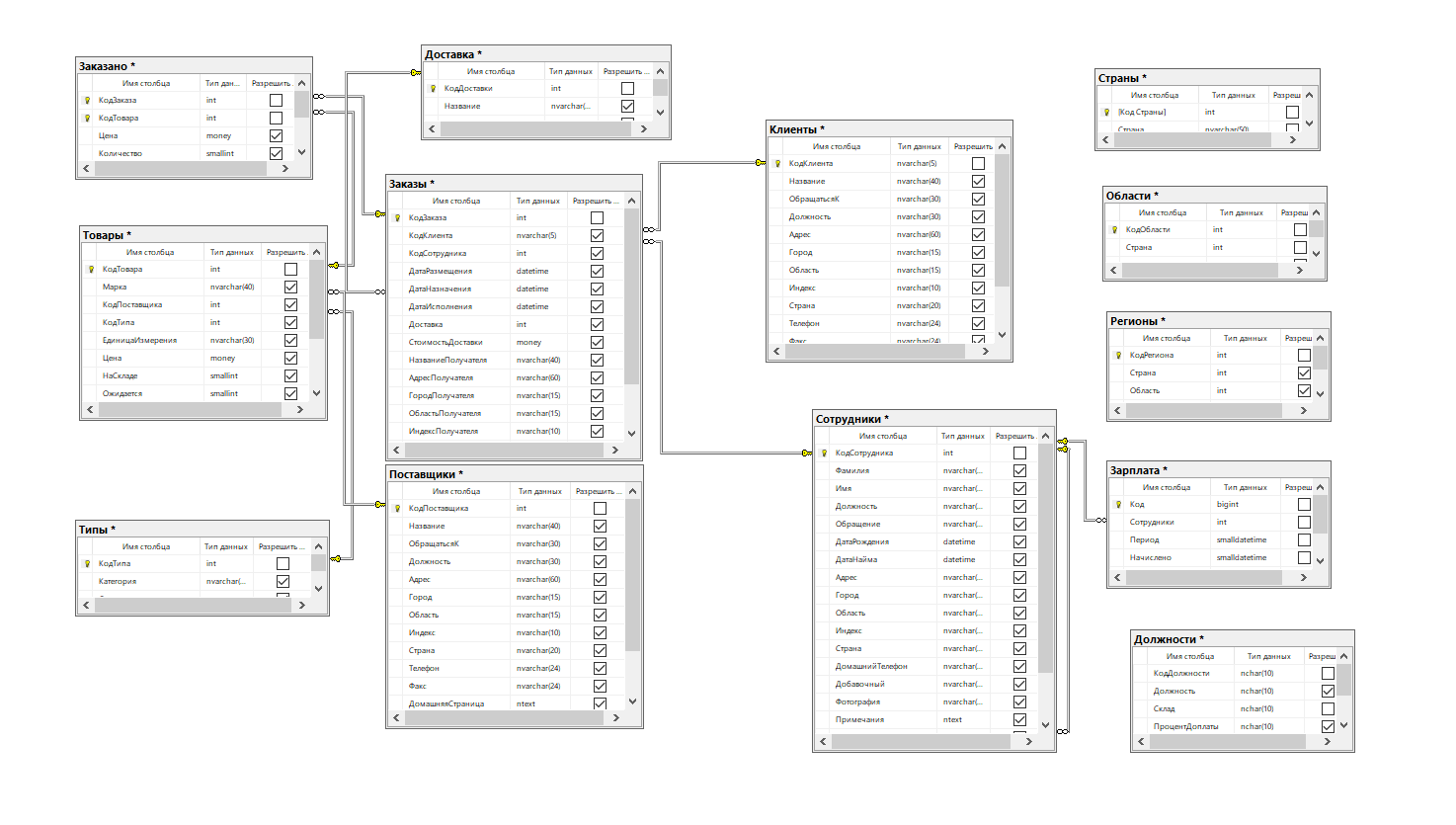
4**.** Далее добавим столбец «Регионы» в таблицы Сотрудники, Клиенты, Поставщики, Заказы.



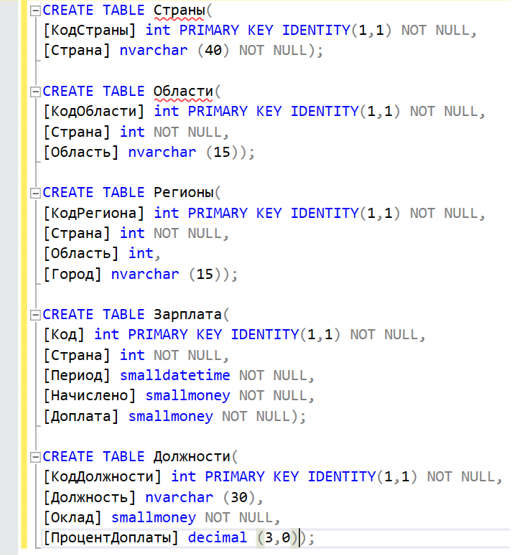
5. Запретим сохранения изменений, требующих повторного создания таблицы для удобств.



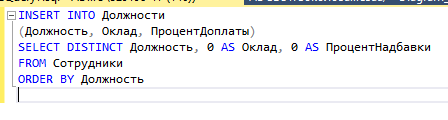
6. Создать новые таблицы Страны, Области, Регионы, Зарплата, Должности. Также на данном скрине можно заметить, что столбец «Регионы» были добавлены во все выше сказанные таблицы: Сотрудники, Клиенты, Поставщики, Заказы.

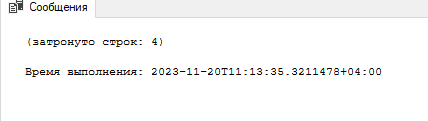


7. Первичные целочисленные ключи новых таблиц сделаем счетчиками (включить свойство Identify (1, 1)).



8**.** Заполнить данными созданную таблицу *Должности* данными колонки *Должность* таблицы *Сотрудники*.





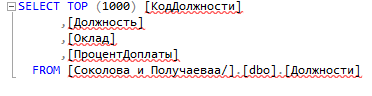
INSERT INTO Должности

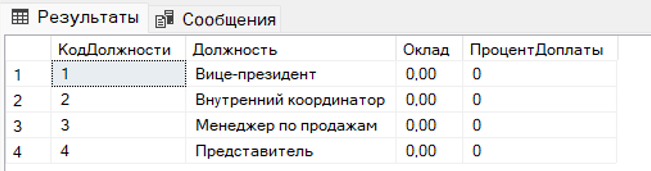
(Должность, Оклад, ПроцентДоплаты)

SELECT DISTINCT Должность, 0 AS Оклад, 0 AS ПроцентНадбавки

FROM Сотрудники

ORDER BY Должность





9.Необходимо заполнить данными созданную таблицу *Страны* данными соответствующих колонок таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*:

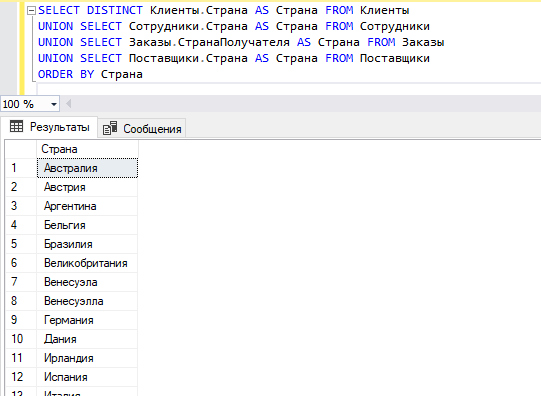
SELECT DISTINCT Клиенты.Страна AS Страна FROM Клиенты

UNION SELECT Сотрудники.Страна AS Страна FROM Сотрудники

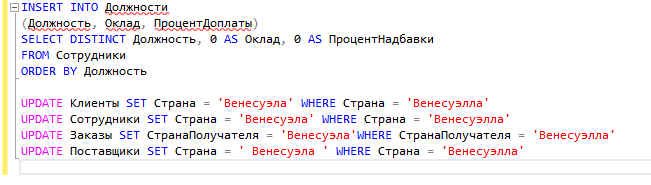
UNION SELECT Заказы.СтранаПолучателя AS Страна FROM Заказы

UNION SELECT Поставщики.Страна AS Страна FROM Поставщики

ORDER BY Страна



Проверяя результаты запроса, мы увидили что в исходных данных имеется несогласованность – страна «Венесуэла» представлена двумя названиями. Исправим неверное название запросом :



INSERT INTO Должности

(Должность, Оклад, ПроцентДоплаты)

SELECT DISTINCT Должность, 0 AS Оклад, 0 AS ПроцентНадбавки

FROM Сотрудники

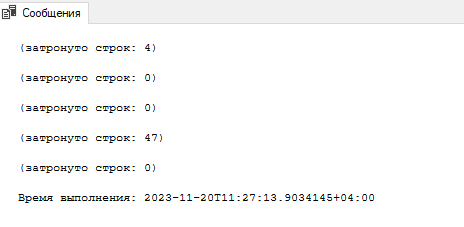
ORDER BY Должность

UPDATE Клиенты SET Страна = 'Венесуэла' WHERE Страна = 'Венесуэлла'

UPDATE Сотрудники SET Страна = 'Венесуэла' WHERE Страна = 'Венесуэлла'

UPDATE Заказы SET СтранаПолучателя = 'Венесуэла'WHERE СтранаПолучателя = 'Венесуэлла'

UPDATE Поставщики SET Страна = ' Венесуэла ' WHERE Страна = 'Венесуэлла'





10.Заполнить данными таблицу *Области* данными соответствующих колонок таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*;Для создания соответствия *Страна−Область* создать временное представление *Области−Страны* с имеющимися данными о странах и их областях

CREATE VIEW \_ОбластиСтраны AS SELECT DISTINCT Клиенты.Область AS Область, Клиенты.Страна AS Страна FROM Клиенты

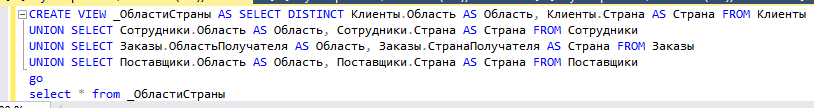
UNION SELECT Сотрудники.Область AS Область, Сотрудники.Страна AS Страна FROM Сотрудники

UNION SELECT Заказы.ОбластьПолучателя AS Область, Заказы.СтранаПолучателя AS Страна FROM Заказы

UNION SELECT Поставщики.Область AS Область, Поставщики.Страна AS Страна FROM Поставщики

go

select \* from \_ОбластиСтраны





Необходимо заменить все значения NULL в столбцах Область всех таблиц пустой строкой:

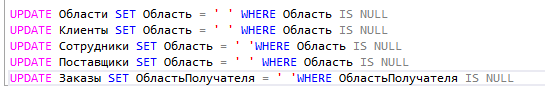
UPDATE Области SET Область = ' ' WHERE Область IS NULL

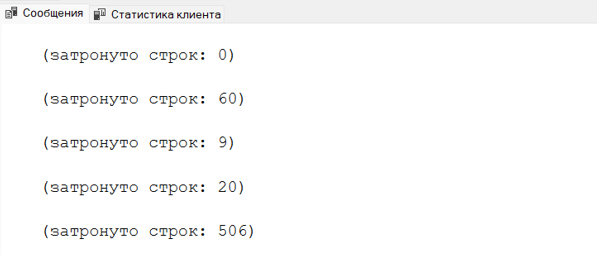
UPDATE Клиенты SET Область = ' ' WHERE Область IS NULL

UPDATE Сотрудники SET Область = ' 'WHERE Область IS NULL

UPDATE Поставщики SET Область = ' ' WHERE Область IS NULL

UPDATE Заказы SET ОбластьПолучателя = ' 'WHERE ОбластьПолучателя IS NULL



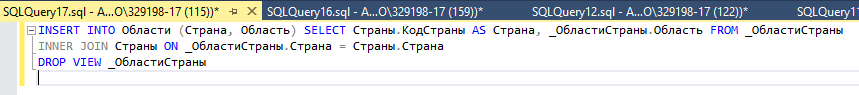


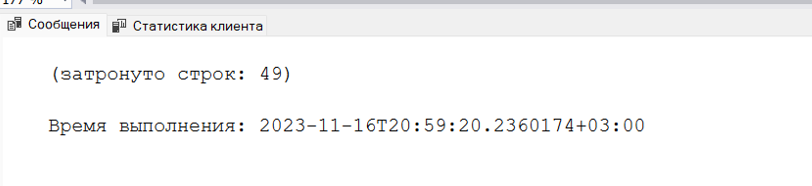
Вставим результаты запроса в таблицу Области и удалим временное представление ОбластиСтраны:

INSERT INTO Области (Страна, Область) SELECT Страны.КодСтраны AS Страна, \_ОбластиСтраны.Область FROM \_ОбластиСтраны

INNER JOIN Страны ON \_ОбластиСтраны.Страна = Страны.Страна

DROP VIEW \_ОбластиСтраны

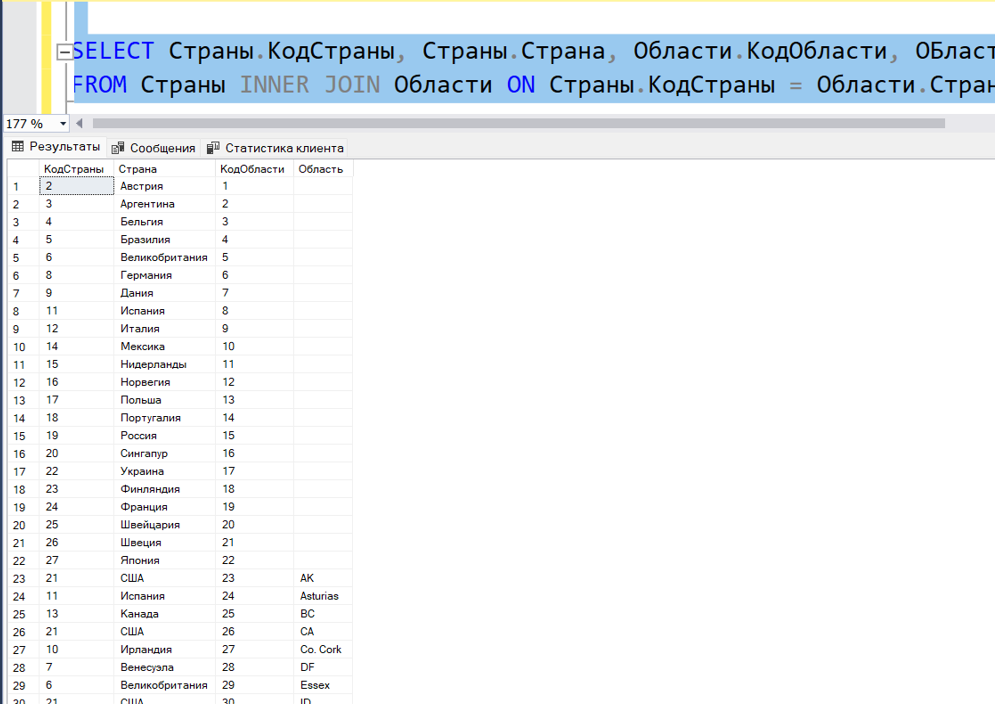




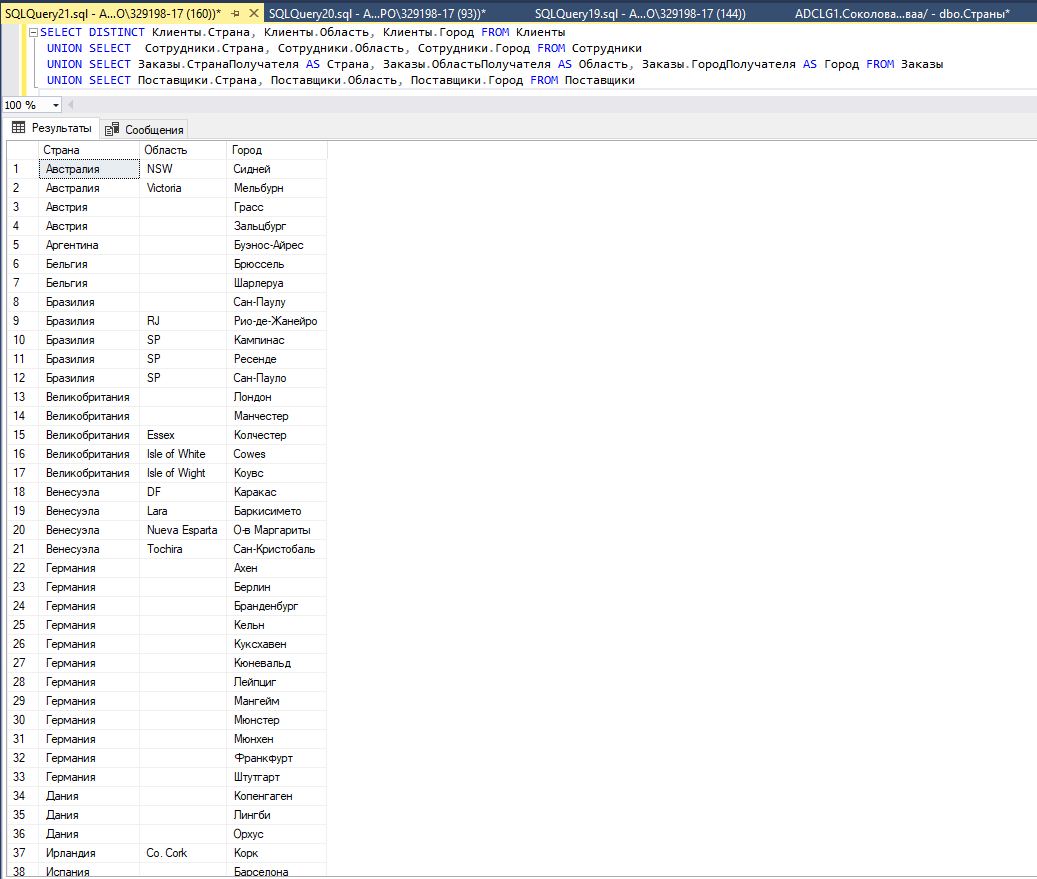
11.Заполнить данными таблицу *Регионы* данными соответствующих колонок таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы*; Определим страны, области и их коды в таблицах *Страна* и *Область.*

SELECT Страны.КодСтраны, Страны.Страна, Области.КодОбласти, ОБласти.Область

FROM Страны INNER JOIN Области ON Страны.КодСтраны = Области.Страна



12. Выберем города с соответствующими областями и странами из таблиц *Клиенты, Сотрудники, Поставщики* и *Заказы:*



На основе запросов выше создадим представление:

CREATE VIEW AS SELECT Страны.КодСтраны, Страны.Страна, Области.КодОбласти, Области.Область

FROM Страны INNER JOIN Области ON Страны.КодСтраны = Области.Страна

GO

CREATE VIEW AS SELECT DISTINCT Клиенты.Страна, Клиенты.Область, Клиенты.Город FROM Клиенты

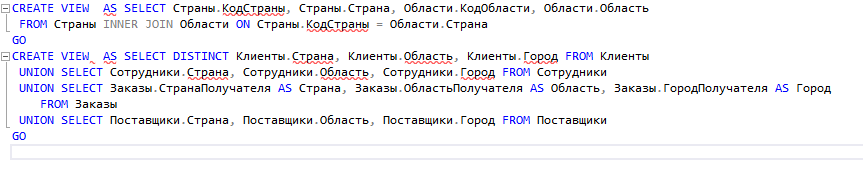
UNION SELECT Сотрудники.Страна, Сотрудники.Область, Сотрудники.Город FROM Сотрудники

UNION SELECT Заказы.СтранаПолучателя AS Страна, Заказы.ОбластьПолучателя AS Область, Заказы.ГородПолучателя AS Город

FROM Заказы

UNION SELECT Поставщики.Страна, Поставщики.Область, Поставщики.Город FROM Поставщики

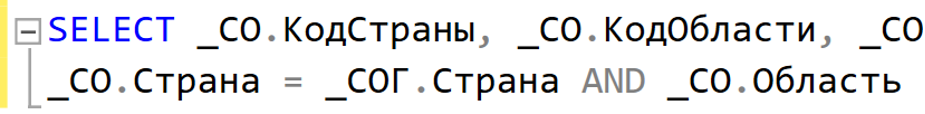
GO

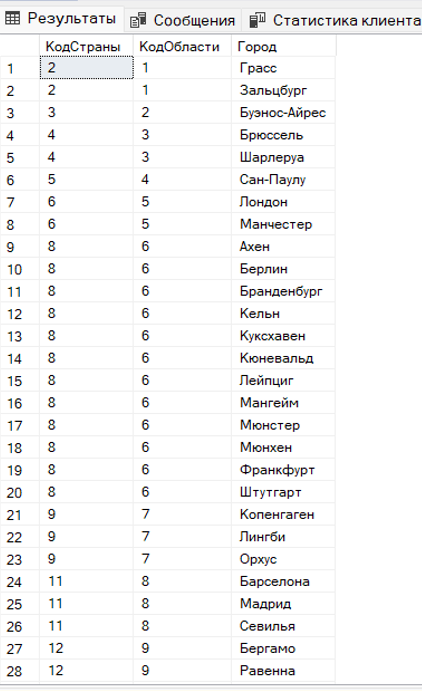


Объединить данные представлений и проверить результат.

SELECT \_СО.КодСтраны, \_СО.КодОбласти, \_СОГ.Город FROM \_СО, \_СОГ WHERE

\_СО.Страна = \_СОГ.Страна AND \_СО.Область = \_СОГ.Область



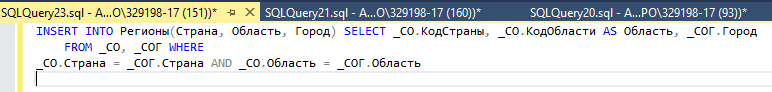


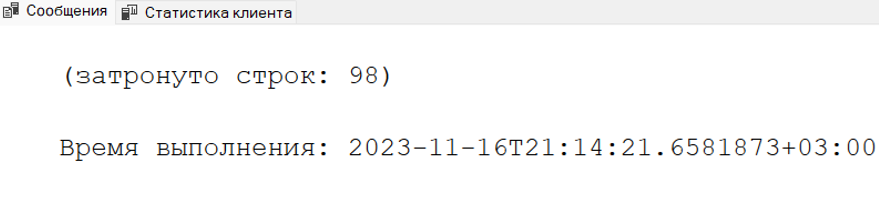
Объединенные данные обоих представлений вставить в таблицу Регионы:

INSERT INTO Регионы(Страна, Область, Город) SELECT \_СО.КодСтраны, \_СО.КодОбласти AS Область, \_СОГ.Город

FROM \_СО, \_СОГ WHERE

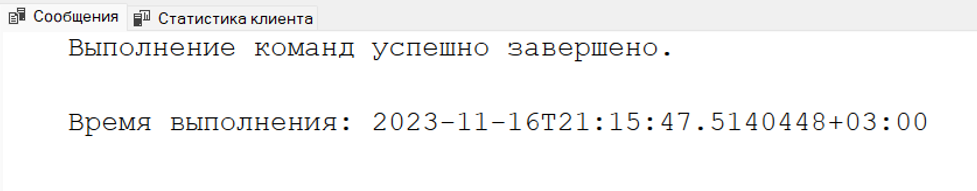
\_СО.Страна = \_СОГ.Страна AND \_СО.Область = \_СОГ.Область





Удалить временные представления:





**Практическая работа №2**

**Основы языка программирования Transact−SQL СУБД MS SQL Server**

**Цель работы:** изучение алгоритмических конструкций языка Transact-SQL

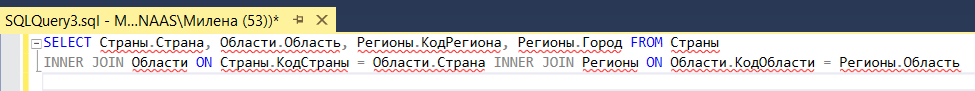
**Задачи:**

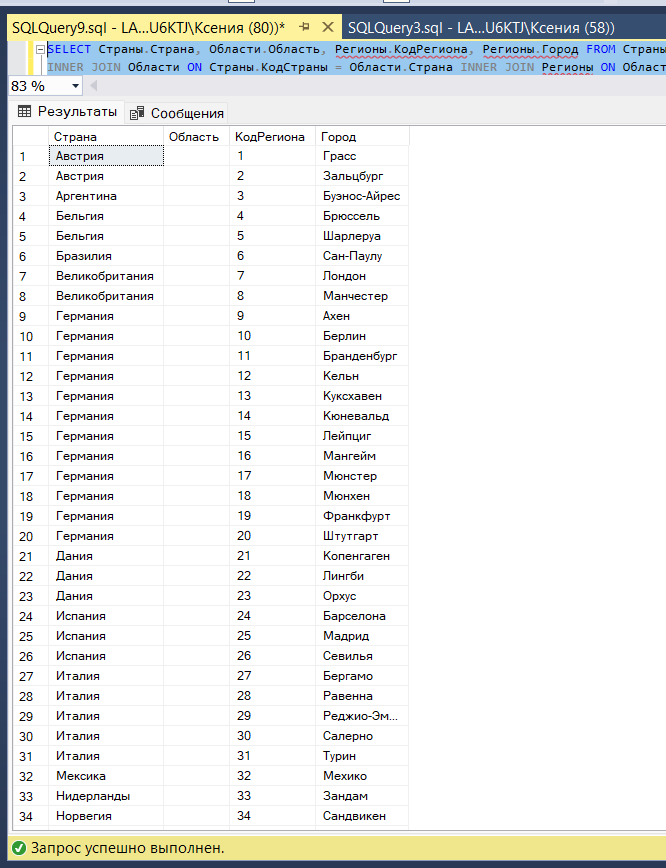
1) Устранить дублирование данных в таблицах БД, созданных в ходе выполнения лабораторной работы №1 и установить связи между таблицами;

2) Обеспечить возможность изменения окладов сотрудников с течением времени и определения актуальных значений на любой период;

3) Обеспечить условия сохранения целостности данных.

1. Заполним поле Регион таблиц Клиенты, Сотрудники, Поставщики, РегионПолучателя таблицы Заказы значениями поля КодРегиона таблица Регионы.

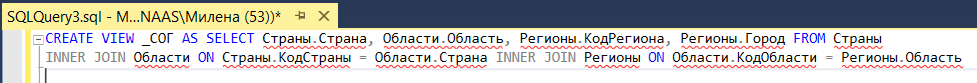


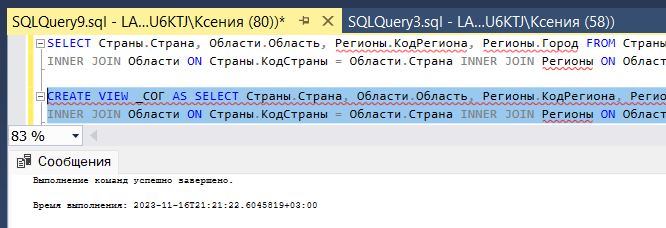


SELECT Страны.Страна, Области.Область, Регионы.КодРегиона, Регионы.Город FROM Страны

INNER JOIN Области ON Страны.КодСтраны = Области.Страна INNER JOIN Регионы ON Области.КодОбласти = Регионы.Область

На основе данного запроса составим временное представление:

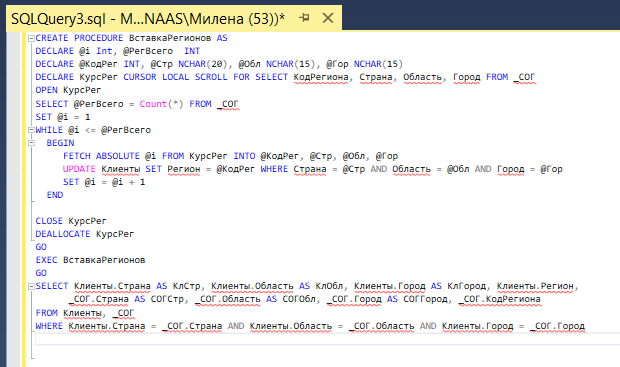


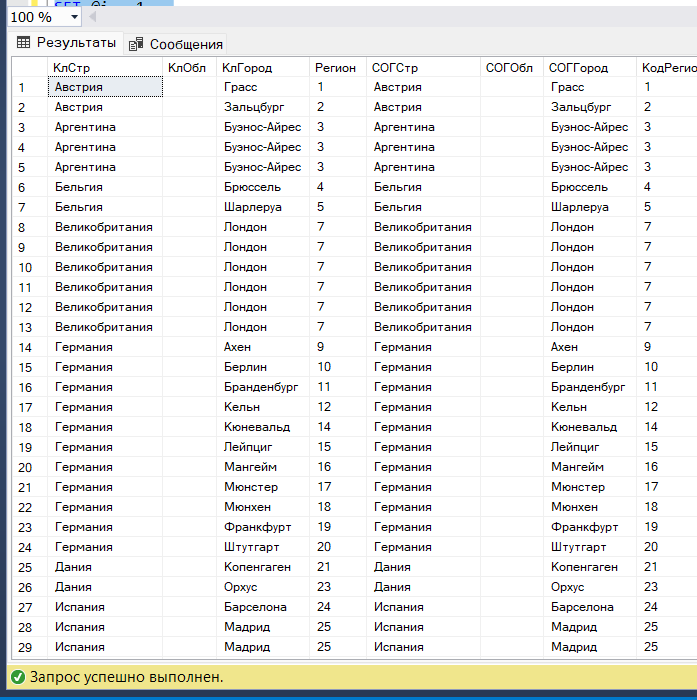


CREATE VIEW \_СОГ AS SELECT Страны.Страна, Области.Область, Регионы.КодРегиона, Регионы.Город FROM Страны

INNER JOIN Области ON Страны.КодСтраны = Области.Страна INNER JOIN Регионы ON Области.КодОбласти = Регионы.Область

Создадим хранимую процедуру Вставка-Регионов:





CREATE PROCEDURE ВставкаРегионов AS

DECLARE @i Int, @РегВсего INT

DECLARE @КодРег INT, @Стр NCHAR(20), @Обл NCHAR(15), @Гор NCHAR(15)

DECLARE КурсРег CURSOR LOCAL SCROLL FOR SELECT КодРегиона, Страна, Область, Город FROM \_СОГ

OPEN КурсРег

SELECT @РегВсего = Count(\*) FROM \_СОГ

SET @i = 1

WHILE @i <= @РегВсего

BEGIN

FETCH ABSOLUTE @i FROM КурсРег INTO @КодРег, @Стр, @Обл, @Гор

UPDATE Клиенты SET Регион = @КодРег WHERE Страна = @Стр AND Область = @Обл AND Город = @Гор

SET @i = @i + 1

END

CLOSE КурсРег

DEALLOCATE КурсРег

GO

EXEC ВставкаРегионов

GO

SELECT Клиенты.Страна AS КлСтр, Клиенты.Область AS КлОбл, Клиенты.Город AS КлГород, Клиенты.Регион,

\_СОГ.Страна AS СОГСтр, \_СОГ.Область AS СОГОбл, \_СОГ.Город AS СОГГород, \_СОГ.КодРегиона

FROM Клиенты, \_СОГ

WHERE Клиенты.Страна = \_СОГ.Страна AND Клиенты.Область = \_СОГ.Область AND Клиенты.Город = \_СОГ.Город

Заполним поле Регион таблицы Поставщики:

CREATE PROCEDURE ВставкаРегионовПост AS

DECLARE @i Int, @РегВсего INT

DECLARE @КодРег INT, @Стр NCHAR(20), @Обл NCHAR(15), @Гор NCHAR(15)

DECLARE КурсРег CURSOR LOCAL SCROLL FOR SELECT КодРегиона, Страна, Область, Город FROM \_СОГ

OPEN КурсРег

SELECT @РегВсего = Count(\*) FROM \_СОГ

SET @i = 1

WHILE @i <= @РегВсего

BEGIN

FETCH ABSOLUTE @i FROM КурсРег INTO @КодРег, @Стр, @Обл, @Гор

UPDATE Поставщики SET Регион = @КодРег WHERE Страна = @Стр AND Область = @Обл AND Город = @Гор

SET @i = @i + 1

END

CLOSE КурсРег

DEALLOCATE КурсРег

GO

EXEC ВставкаРегионовПост

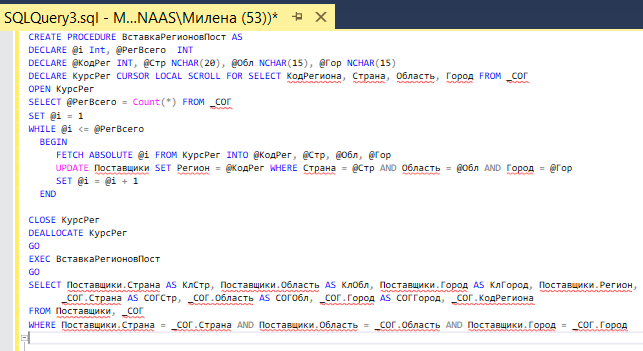
GO

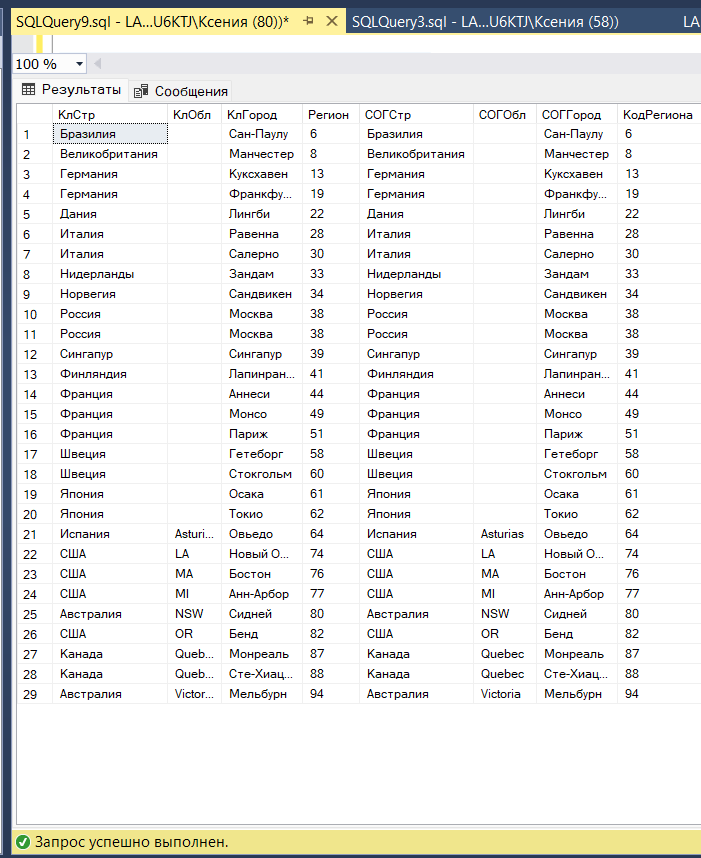
SELECT Поставщики.Страна AS КлСтр, Поставщики.Область AS КлОбл, Поставщики.Город AS КлГород, Поставщики.Регион,

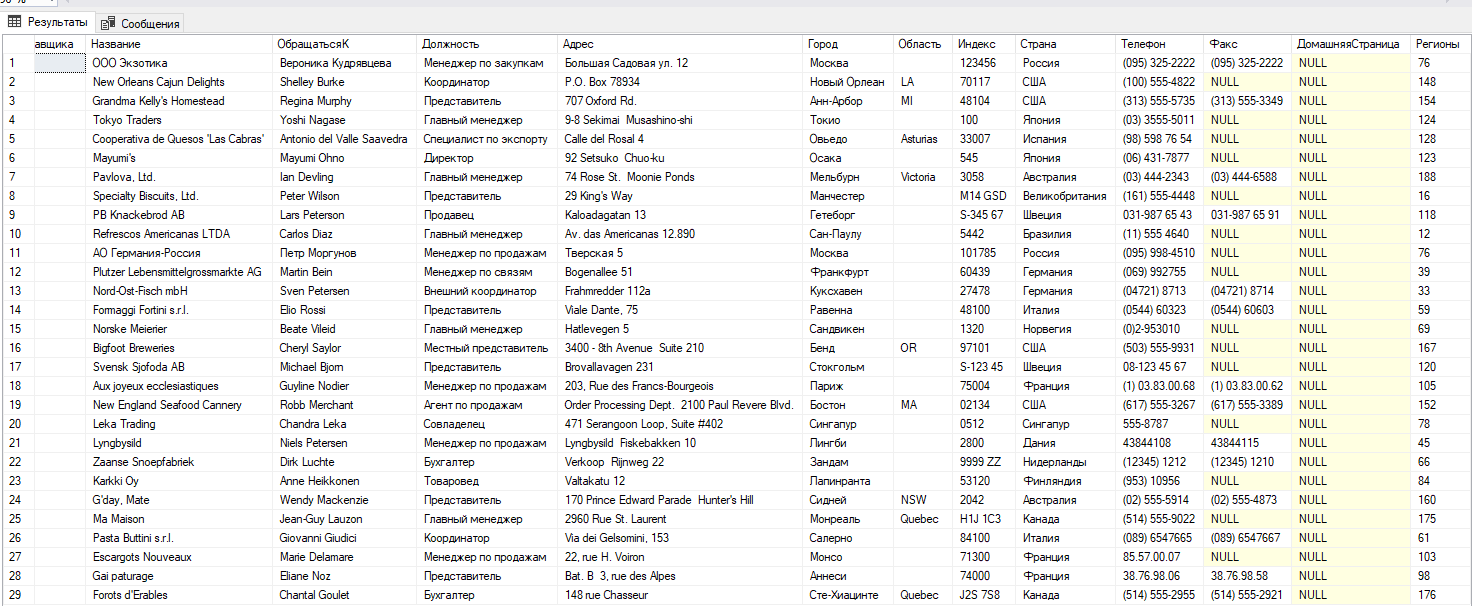
\_СОГ.Страна AS СОГСтр, \_СОГ.Область AS СОГОбл, \_СОГ.Город AS СОГГород, \_СОГ.КодРегиона

FROM Поставщики, \_СОГ

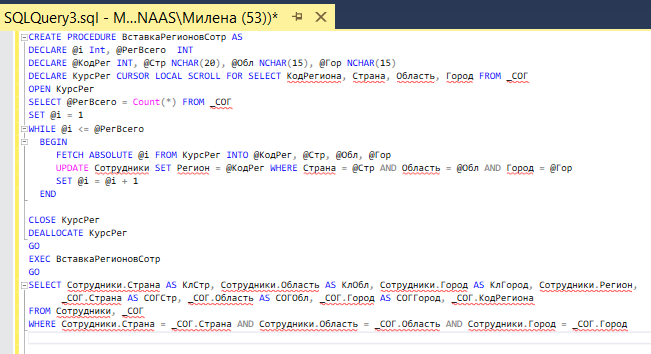
WHERE Поставщики.Страна = \_СОГ.Страна AND Поставщики.Область = \_СОГ.Область AND Поставщики.Город = \_СОГ.Город

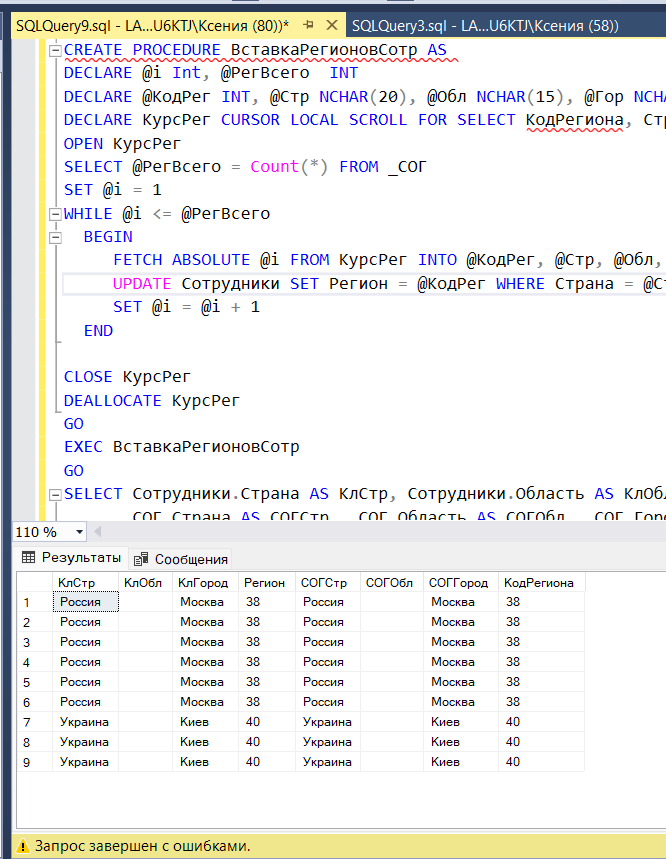






Заполним данными поле Регион таблицы Сотрудники:





CREATE PROCEDURE ВставкаРегионовСотр AS

DECLARE @i Int, @РегВсего INT

DECLARE @КодРег INT, @Стр NCHAR(20), @Обл NCHAR(15), @Гор NCHAR(15)

DECLARE КурсРег CURSOR LOCAL SCROLL FOR SELECT КодРегиона, Страна, Область, Город FROM \_СОГ

OPEN КурсРег

SELECT @РегВсего = Count(\*) FROM \_СОГ

SET @i = 1

WHILE @i <= @РегВсего

BEGIN

FETCH ABSOLUTE @i FROM КурсРег INTO @КодРег, @Стр, @Обл, @Гор

UPDATE Сотрудники SET Регион = @КодРег WHERE Страна = @Стр AND Область = @Обл AND Город = @Гор

SET @i = @i + 1

END

CLOSE КурсРег

DEALLOCATE КурсРег

GO

EXEC ВставкаРегионовСотр

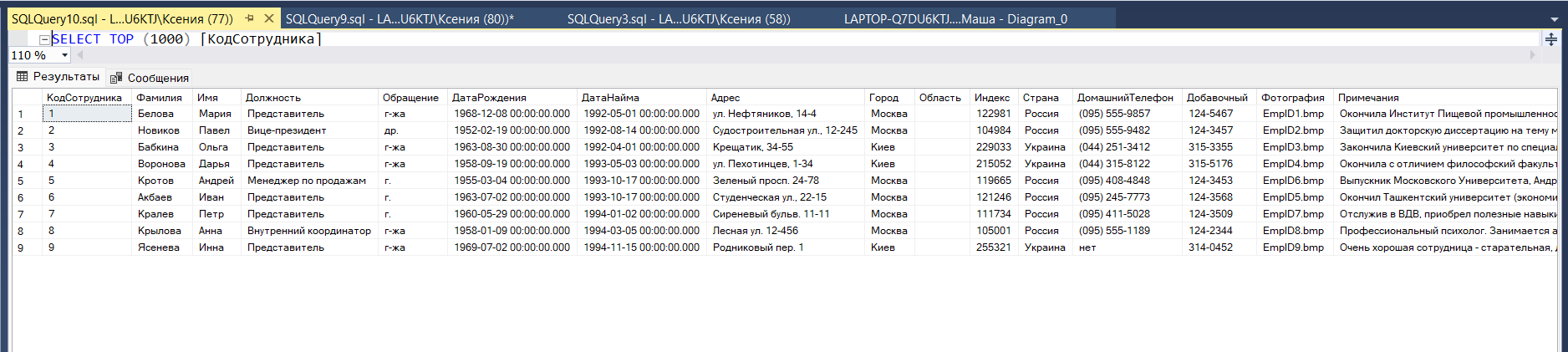
GO

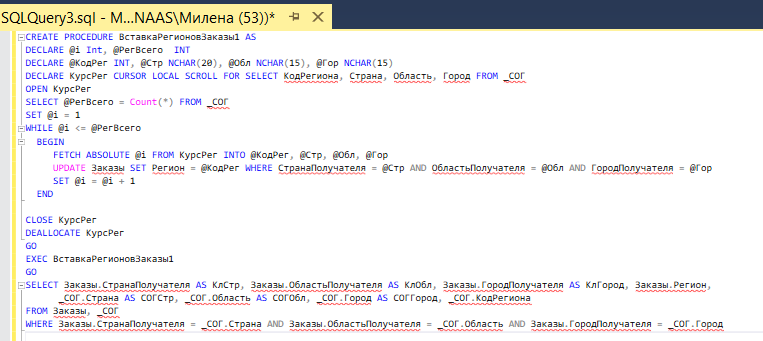
SELECT Сотрудники.Страна AS КлСтр, Сотрудники.Область AS КлОбл, Сотрудники.Город AS КлГород, Сотрудники.Регион,

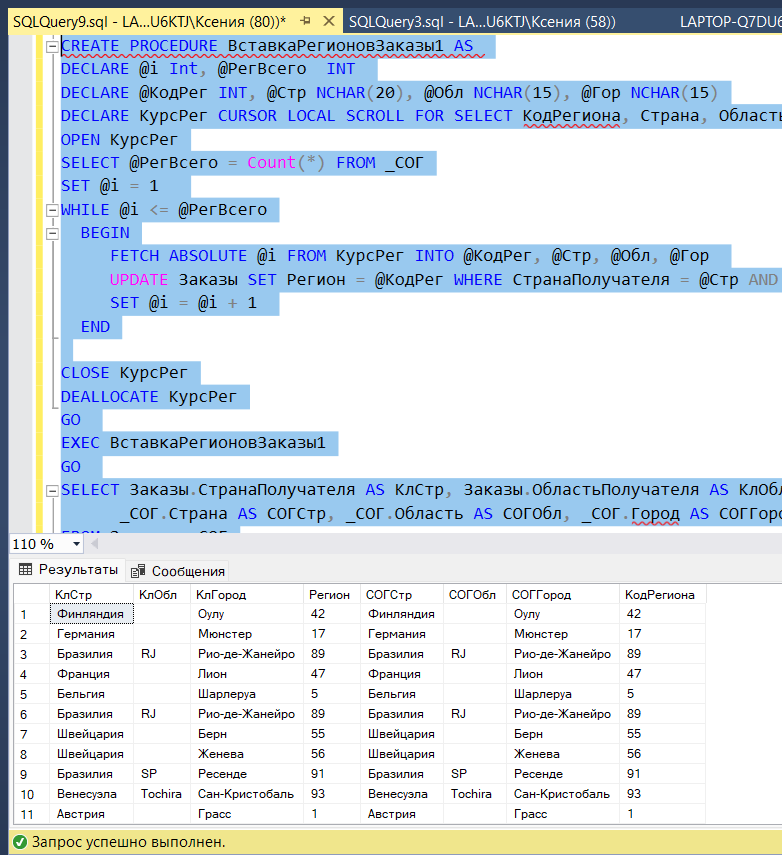
\_СОГ.Страна AS СОГСтр, \_СОГ.Область AS СОГОбл, \_СОГ.Город AS СОГГород, \_СОГ.КодРегиона

FROM Сотрудники, \_СОГ

WHERE Сотрудники.Страна = \_СОГ.Страна AND Сотрудники.Область = \_СОГ.Область AND Сотрудники.Город = \_СОГ.Город







CREATE PROCEDURE ВставкаРегионовЗаказы1 AS

DECLARE @i Int, @РегВсего INT

DECLARE @КодРег INT, @Стр NCHAR(20), @Обл NCHAR(15), @Гор NCHAR(15)

DECLARE КурсРег CURSOR LOCAL SCROLL FOR SELECT КодРегиона, Страна, Область, Город FROM \_СОГ

OPEN КурсРег

SELECT @РегВсего = Count(\*) FROM \_СОГ

SET @i = 1

WHILE @i <= @РегВсего

BEGIN

FETCH ABSOLUTE @i FROM КурсРег INTO @КодРег, @Стр, @Обл, @Гор

UPDATE Заказы SET Регион = @КодРег WHERE СтранаПолучателя = @Стр AND ОбластьПолучателя = @Обл AND ГородПолучателя = @Гор

SET @i = @i + 1

END

CLOSE КурсРег

DEALLOCATE КурсРег

GO

EXEC ВставкаРегионовЗаказы1

GO

SELECT Заказы.СтранаПолучателя AS КлСтр, Заказы.ОбластьПолучателя AS КлОбл, Заказы.ГородПолучателя AS КлГород, Заказы.Регион,

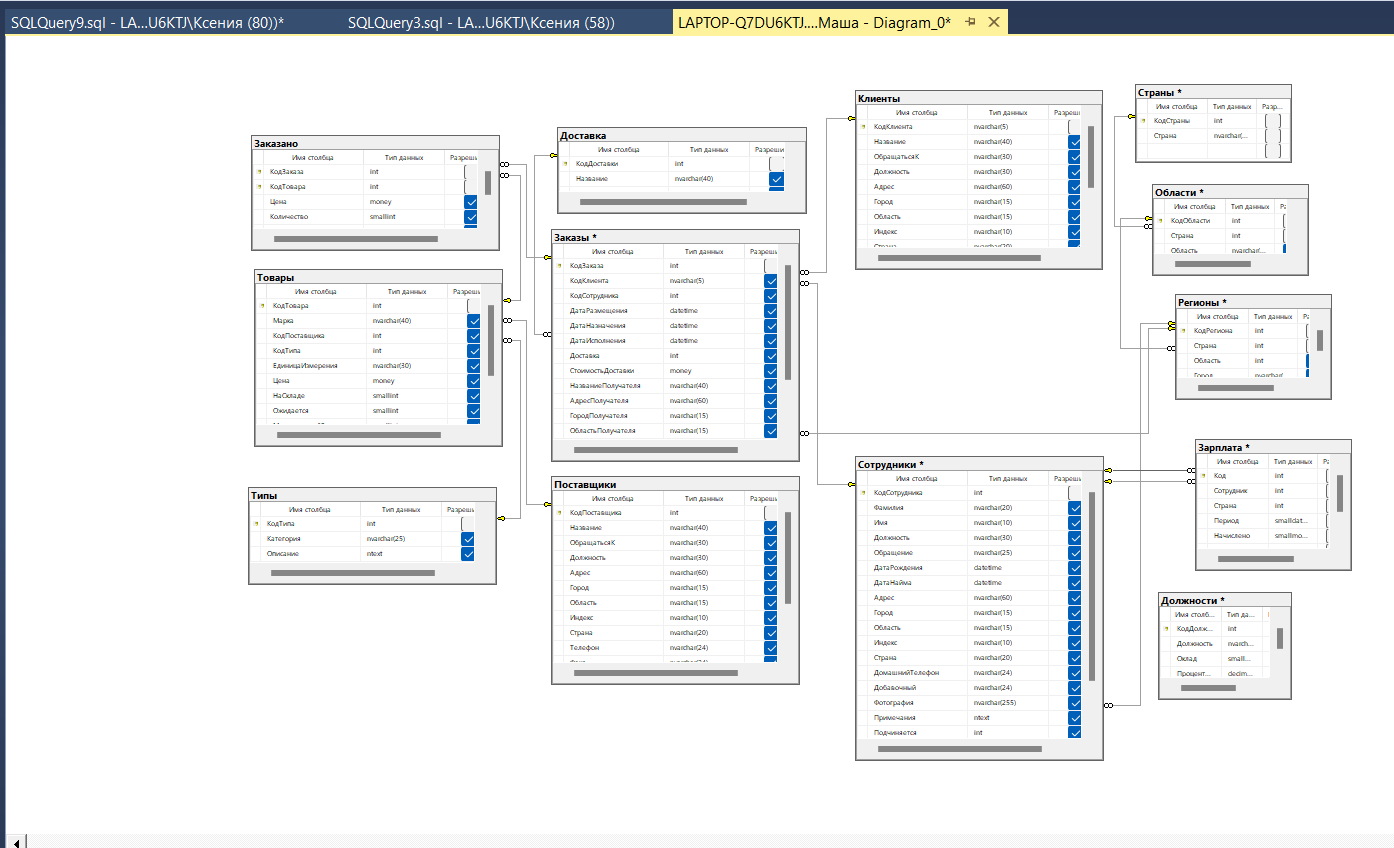
\_СОГ.Страна AS СОГСтр, \_СОГ.Область AS СОГОбл, \_СОГ.Город AS СОГГород, \_СОГ.КодРегиона

FROM Заказы, \_СОГ

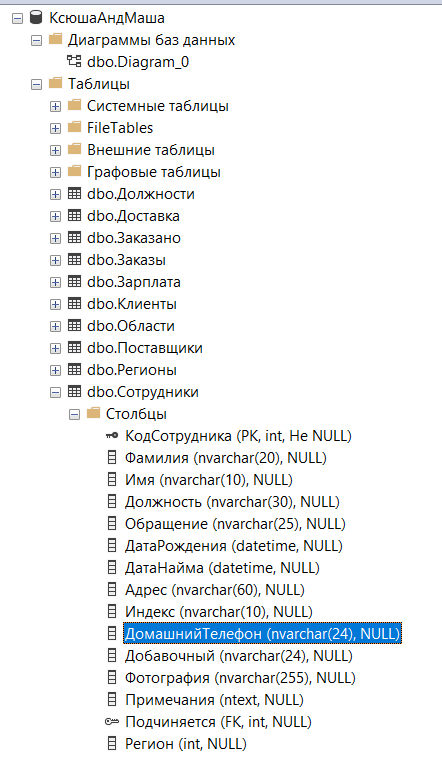
WHERE Заказы.СтранаПолучателя = \_СОГ.Страна AND Заказы.ОбластьПолучателя = \_СОГ.Область AND Заказы.ГородПолучателя = \_СОГ.Город

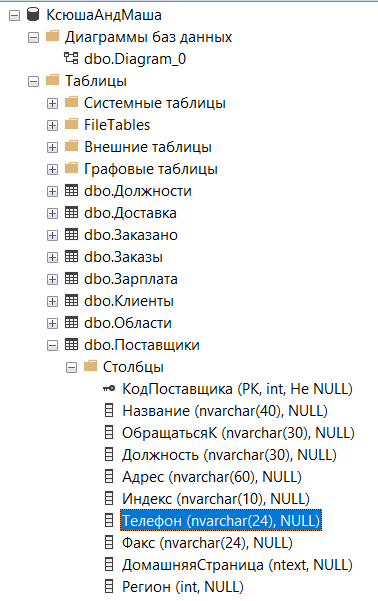


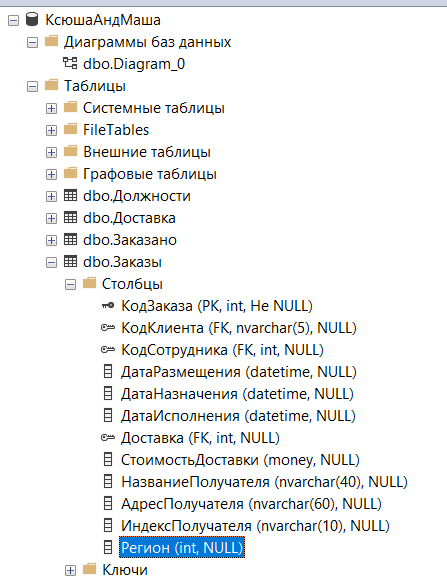
Установим связи между таблицами:

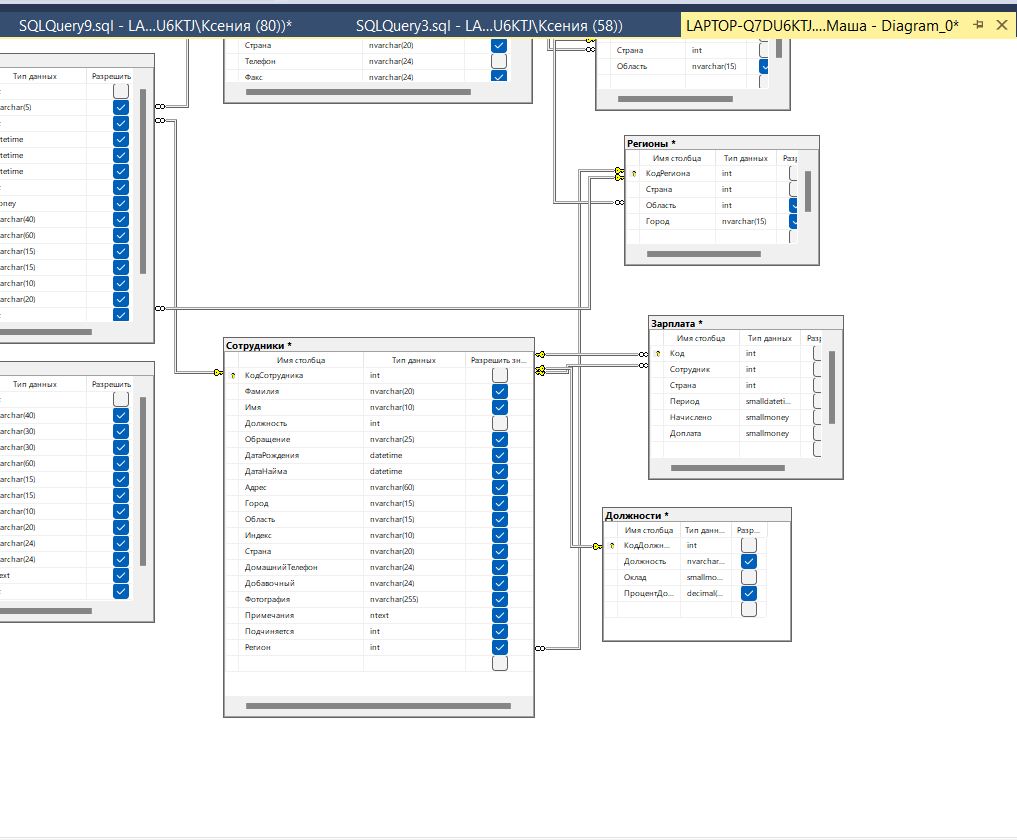


Удалим столбцы с данными о странах, областях и городах из таблиц Клиенты, Сотрудники, Поставщики и Заказы.

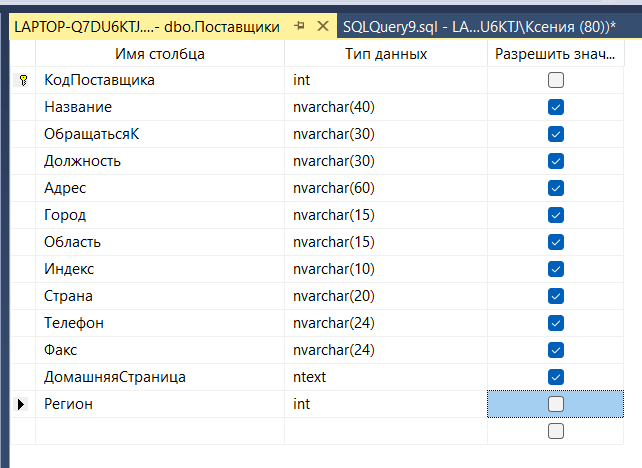


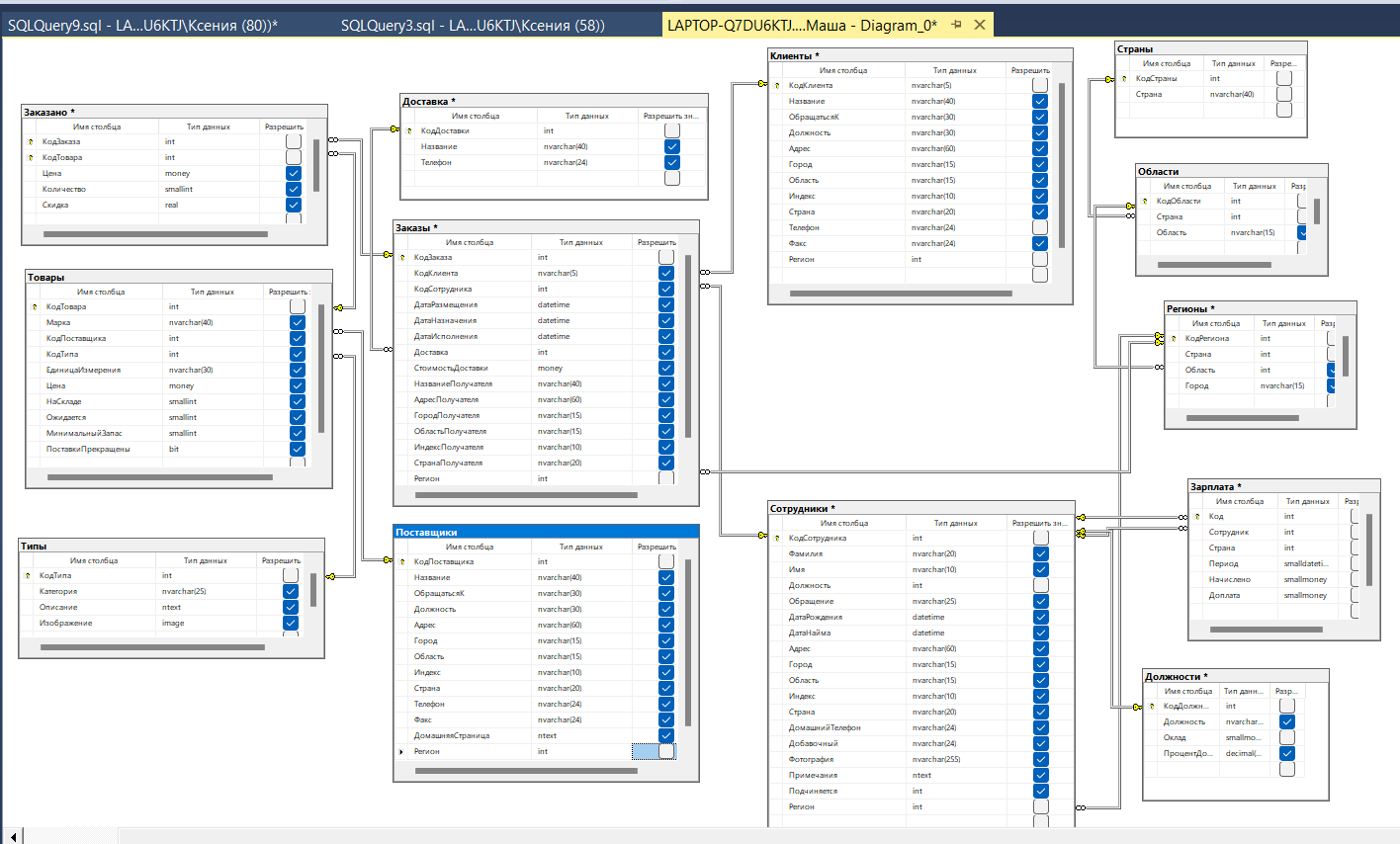




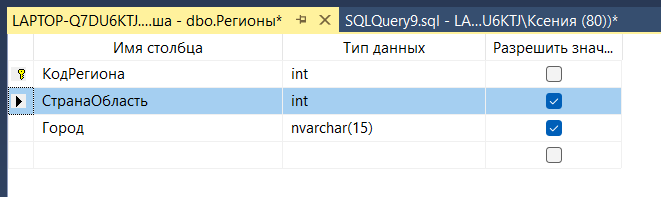
Установим связи между таблицами *Должности* и *Сотрудники*:

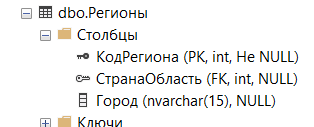
Отключаем возможность иметь значение NUL в столбцах *Регионы* таблиц *Сотрудники*, *Клиенты*, *Поставщики*, *Заказы.* В скрине ниже, показано, что во всех вышеперечисленных таблицах у столбца «РЕГИОН» отключена возможность иметь значение NULL.



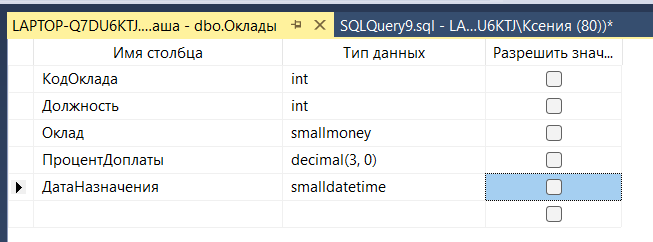


Из таблицы Регионы удаляется столбец Страна и переименовывается столбец Область в СтранаОбласть:

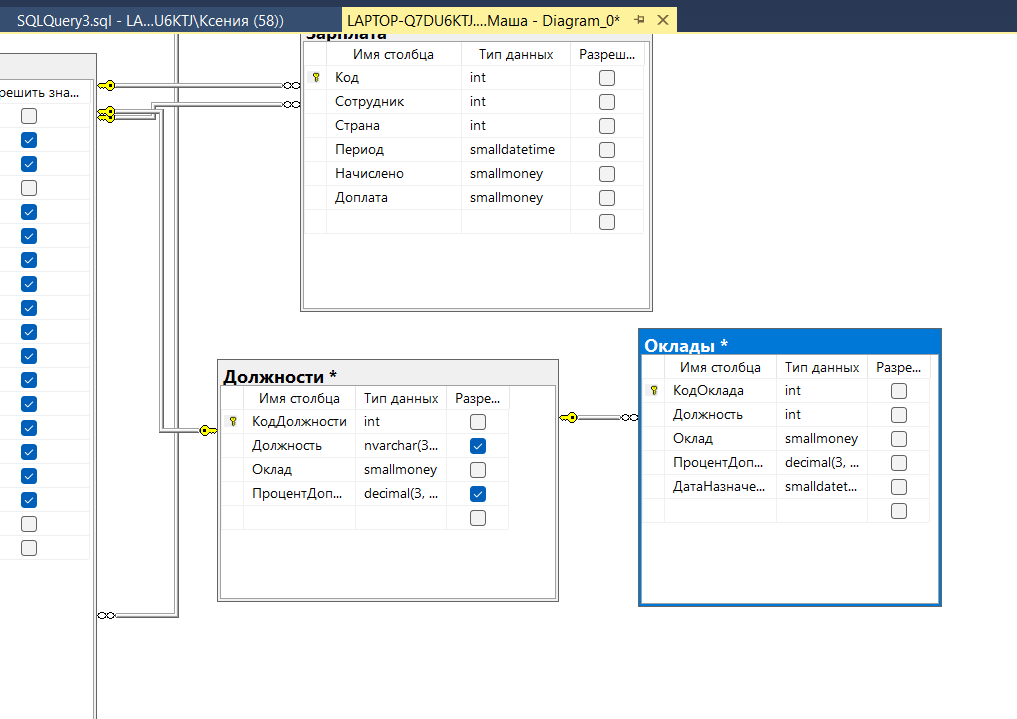




1. Создадим таблицу Оклады с полями КодОклада, Должность, Склад, ПроцентДоплаты, ДатаНазначения.

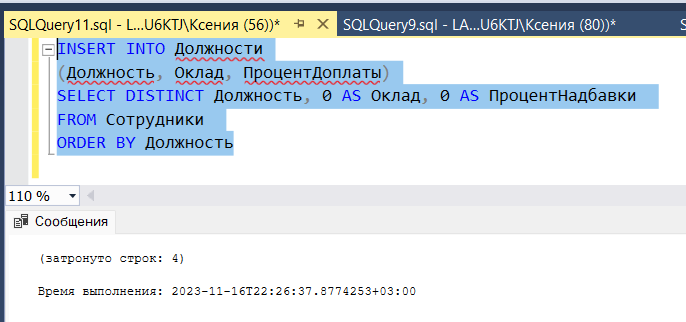


Свяжем таблицы Оклады и Должности:

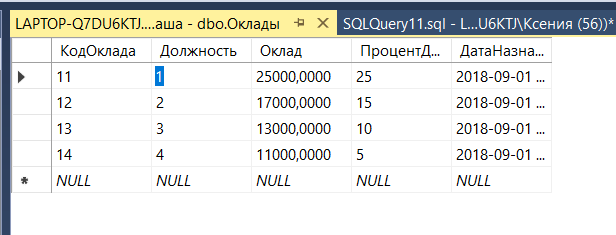
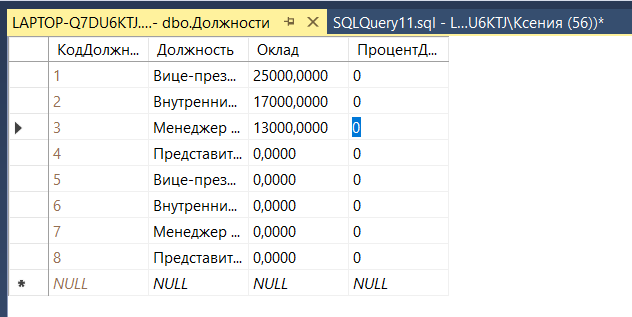


Для заполнения данными таблицы Должности следует выбрать без повторений данные колонки Должность таблицы Сотрудники и вставить их в таблицу Должности. Запрос на выборку проще всего сформировать с помощью конструктора Запросов (выполните команду Выполнить-запрос и на панели инструментов появиться вкладка Запрос). В запросе дополнительно созданы колонки Оклад и ПроцентНадбавки с постоянными значениями равными 0.

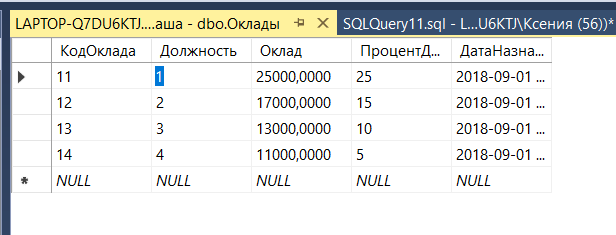
1) Для добавления результатов запроса в таблицу Должности следует изменить в контекстном меню тип запроса на «Вставить результаты…» выбрать в качестве приемника данных таблицу Должности и указать соответствие колонок результатов запроса и колонок таблицы-приемника (установить соответствие между значениями столбцов Column и Append конструктора).



Запустить запрос на выполнение. В случае правильного оформления запроса и успешного выполнения должно быть выведено системное сообщение вида «4 row(s)) affected» (4 строки обработаны). Проверить результат, открыв таблицу Должности. Заполнить поля Оклад и ПроцентДоплаты.



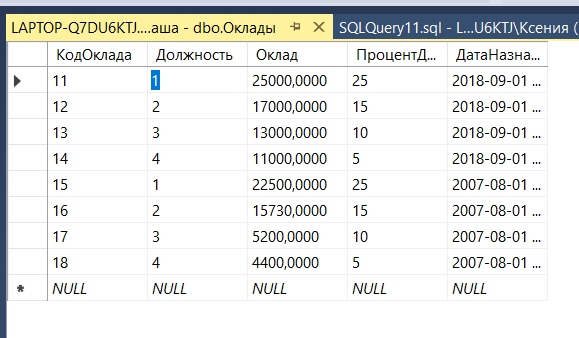
Заполним таблицу Оклады соответствующими данными таблицы Должности при постоянном значении поля ДатаНазначения ’01.09.2018’.



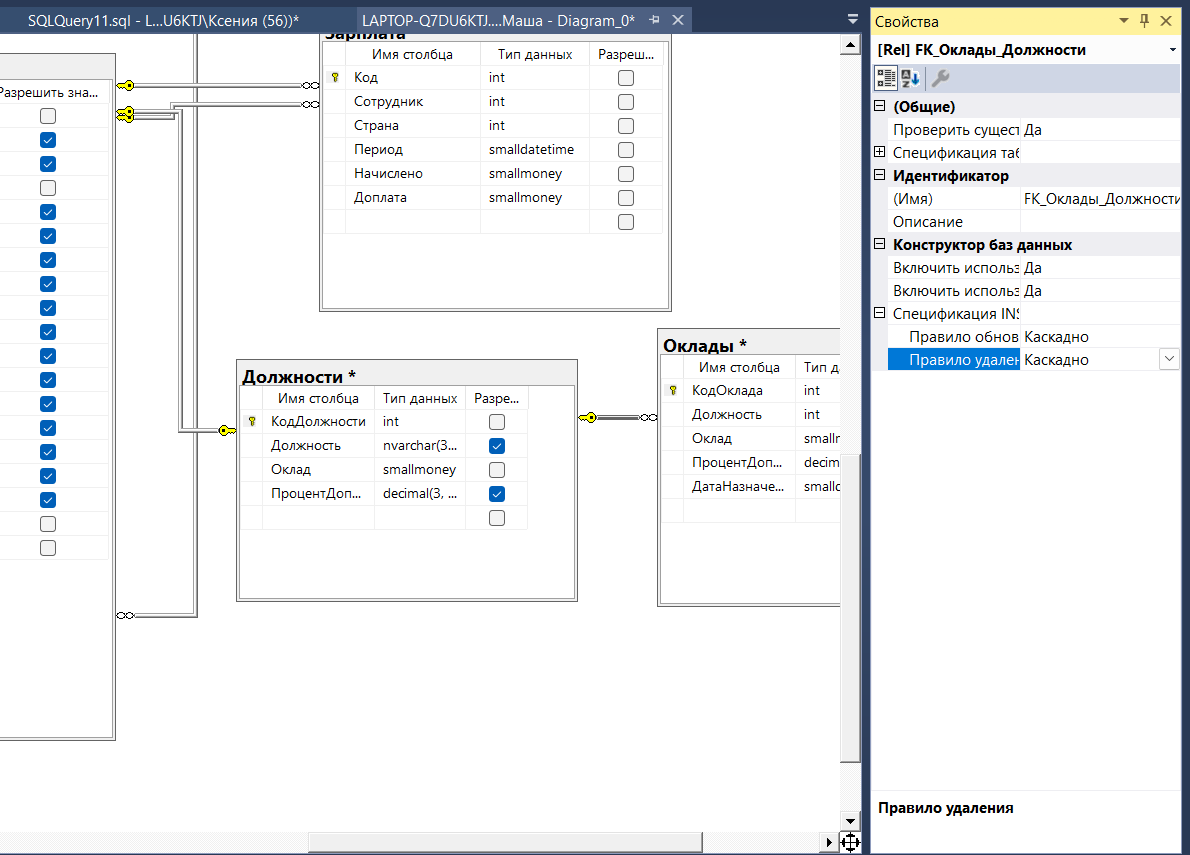
Insert Into Оклады (Должность, Оклад, ПроцентДоплаты, ДатаНазначения)

SELECT Должность, Оклад\*0.9, ПроцентДоплаты, '01.08.2007' FROM Оклады

Where ДатаНазначения = '01.01.2018'



В свойствах связей на диаграмме базы данных установим триггеры удаления и обновления в значение «Каскадное».



**ПРАКТИЧЕСКАЯ РАБОТА №3**

**СРЕДСТВА ПРОГРАММИРОВАНИЯ БИЗНЕС-ЛОГИКИ**

**СЕРВЕРА БАЗЫ ДАННЫХ**

**Цель работы:** изучение средств программирования бизнес-логики сервера базы данных

**Задачи:**Разработать серверное приложение базы данных, обеспечивающее возможности:

**1)** контроля наличия начисления для любого сотрудника за любой период

**2)** определения суммы реализации товаров любым сотрудником за любой отчетный период;

**3)** определения актуальных значений должностного оклада и процента доплаты для любого сотрудника в любой период;

**4)** проведения регулярного начисления заработной платы, как для отдельного сотрудника, так и для всех сотрудников предприятия;

1. Создадим скалярную функцию, получающую в качестве параметра любую дату и возвращающую дату, которая представляет собой первое число того же месяца и года.

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE FUNCTION ПервыйДеньМесяца (@Дата SmallDateTime)

RETURNS SmallDateTime

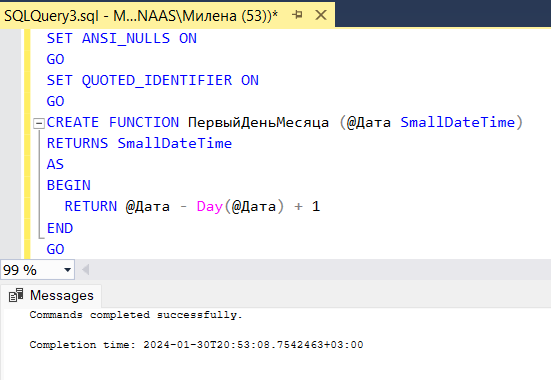
AS

BEGIN

RETURN @Дата - Day(@Дата) + 1

END

GO



Создадим скалярную функцию, возвращающую логическую истину, если запись есть, и ложь – если запись отсутствует.

CREATE FUNCTION СуществованиеНачисленияСотрЗаМесяц (@Дата SmallDateTime, @Сотр Int)

RETURNS Bit

AS

BEGIN

DECLARE @ДаНет Bit

SET @Дата = dbo.ПервыйДеньМесяца(@Дата)

IF EXISTS (SELECT \* FROM Зарплата WHERE Сотрудник = @Сотр AND Период = @Дата)

SET @ДаНет = 1

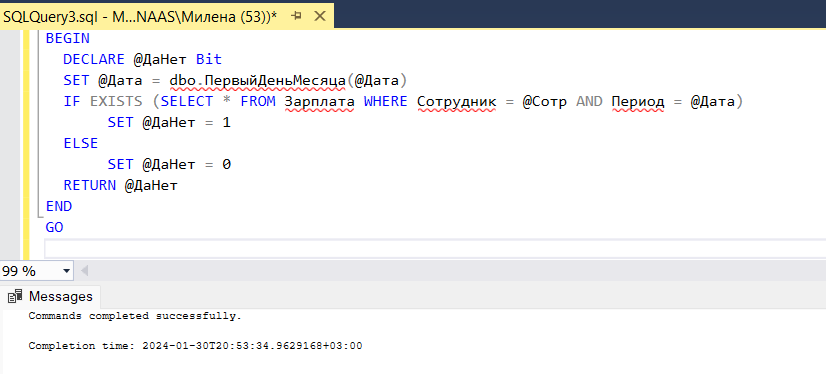
ELSE

SET @ДаНет = 0

RETURN @ДаНет

END

GO



Создадим скалярную функцию для определения суммы реализованных товаров.

CREATE FUNCTION СуммаЗаказовСотрудникаЗаОтчетныйПериод (@Дата SmallDateTime, @Сотр Int)

RETURNS MONEY

AS

BEGIN

DECLARE @Сумма Money

DECLARE @ТекущийПериод SmallDateTime, @НачОтчПериода SmallDateTime, @КонОтчПериода SmallDateTime

SET @ТекущийПериод = dbo.ПервыйДеньМесяца(@Дата)

SET @НачОтчПериода = DateAdd(Month, -1, @ТекущийПериод)

SET @КонОтчПериода = @ТекущийПериод -1

SELECT @Сумма = SUM( (Заказано.Цена\*Заказано.Количество)\*(1-Заказано.Скидка))

FROM Заказано INNER JOIN Заказы ON Заказано.КодЗаказа = Заказы.КодЗаказа

INNER JOIN Сотрудники ON Заказы.КодСотрудника = Сотрудники.КодСотрудника

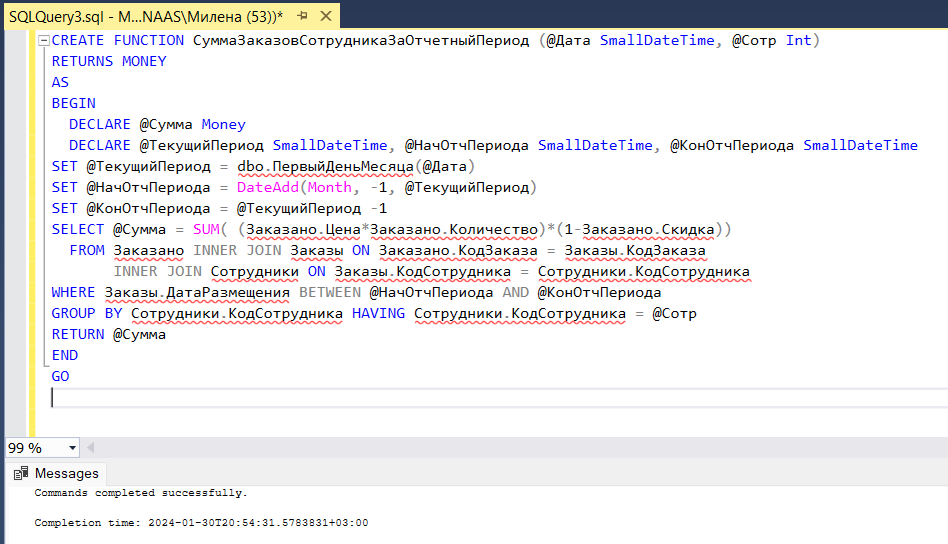
WHERE Заказы.ДатаРазмещения BETWEEN @НачОтчПериода AND @КонОтчПериода

GROUP BY Сотрудники.КодСотрудника HAVING Сотрудники.КодСотрудника = @Сотр

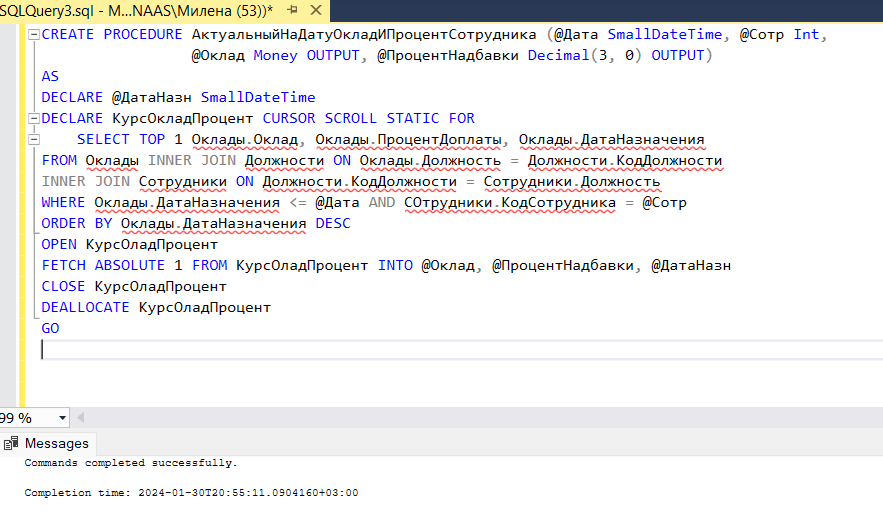
RETURN @Сумма

END

GO



Создадим процедуру для определения актуальных значений должностного оклада и процента доплаты.



CREATE PROCEDURE АктуальныйНаДатуОкладИПроцентСотрудника (@Дата SmallDateTime, @Сотр Int,

@Оклад Money OUTPUT, @ПроцентНадбавки Decimal(3, 0) OUTPUT)

AS

DECLARE @ДатаНазн SmallDateTime

DECLARE КурсОкладПроцент CURSOR SCROLL STATIC FOR

SELECT TOP 1 Оклады.Оклад, Оклады.ПроцентДоплаты, Оклады.ДатаНазначения

FROM Оклады INNER JOIN Должности ON Оклады.Должность = Должности.КодДолжности

INNER JOIN Сотрудники ON Должности.КодДолжности = Сотрудники.Должность

WHERE Оклады.ДатаНазначения <= @Дата AND СОтрудники.КодСотрудника = @Сотр

ORDER BY Оклады.ДатаНазначения DESC

OPEN КурсОладПроцент

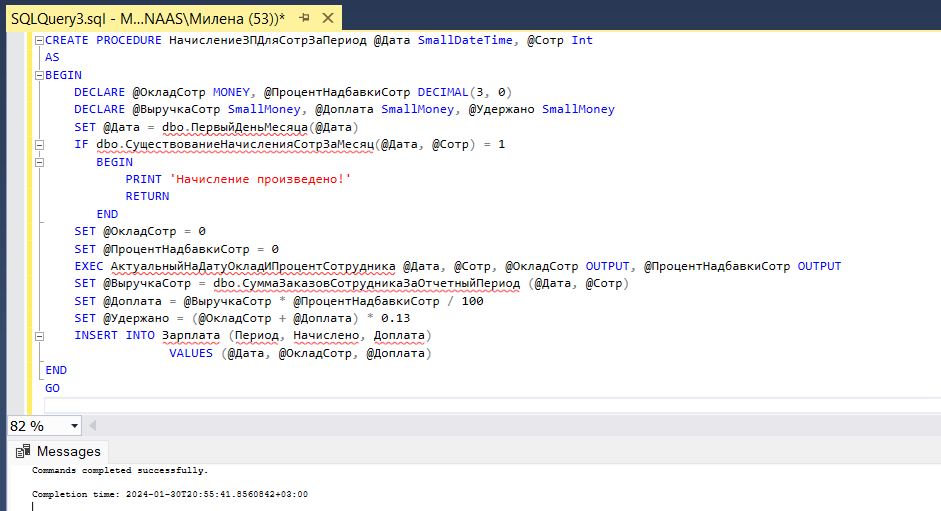
FETCH ABSOLUTE 1 FROM КурсОладПроцент INTO @Оклад, @ПроцентНадбавки, @ДатаНазн

CLOSE КурсОладПроцент

DEALLOCATE КурсОладПроцент

GO

Создадим хранимую процедуру для выполнения функции начисления зарплаты сотрудника НачисленияЗПДляСотрЗаПериод.



CREATE PROCEDURE НачислениеЗПДляСотрЗаПериод @Дата SmallDateTime, @Сотр Int

AS

BEGIN

DECLARE @ОкладСотр MONEY, @ПроцентНадбавкиСотр DECIMAL(3, 0)

DECLARE @ВыручкаСотр SmallMoney, @Доплата SmallMoney, @Удержано SmallMoney

SET @Дата = dbo.ПервыйДеньМесяца(@Дата)

IF dbo.СуществованиеНачисленияСотрЗаМесяц(@Дата, @Сотр) = 1

BEGIN

PRINT 'Начисление произведено!'

RETURN

END

SET @ОкладСотр = 0

SET @ПроцентНадбавкиСотр = 0

EXEC АктуальныйНаДатуОкладИПроцентСотрудника @Дата, @Сотр, @ОкладСотр OUTPUT, @ПроцентНадбавкиСотр OUTPUT

SET @ВыручкаСотр = dbo.СуммаЗаказовСотрудникаЗаОтчетныйПериод (@Дата, @Сотр)

SET @Доплата = @ВыручкаСотр \* @ПроцентНадбавкиСотр / 100

SET @Удержано = (@ОкладСотр + @Доплата) \* 0.13

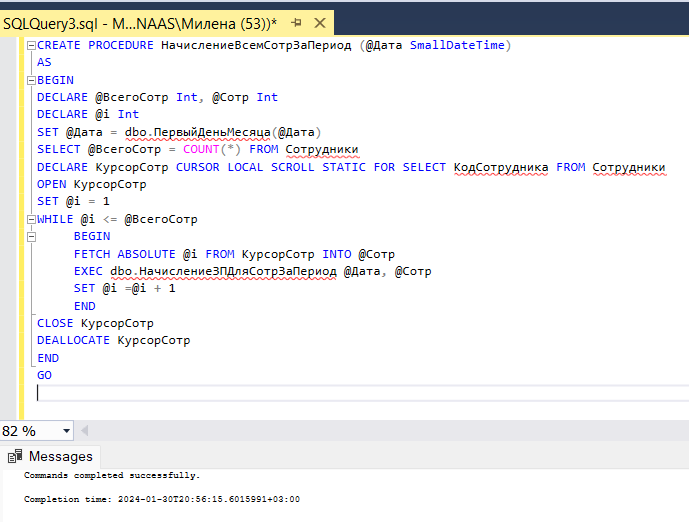
INSERT INTO Зарплата (Период, Начислено, Доплата)

VALUES (@Дата, @ОкладСотр, @Доплата)

END

GO

Создадим процедуру для проведения массового начисления всем сотрудникам.



CREATE PROCEDURE НачислениеВсемСотрЗаПериод (@Дата SmallDateTime)

AS

BEGIN

DECLARE @ВсегоСотр Int, @Сотр Int

DECLARE @i Int

SET @Дата = dbo.ПервыйДеньМесяца(@Дата)

SELECT @ВсегоСотр = COUNT(\*) FROM Сотрудники

DECLARE КурсорСотр CURSOR LOCAL SCROLL STATIC FOR SELECT КодСотрудника FROM Сотрудники

OPEN КурсорСотр

SET @i = 1

WHILE @i <= @ВсегоСотр

BEGIN

FETCH ABSOLUTE @i FROM КурсорСотр INTO @Сотр

EXEC dbo.НачислениеЗПДляСотрЗаПериод @Дата, @Сотр

SET @i =@i + 1

END

CLOSE КурсорСотр

DEALLOCATE КурсорСотр

END

GO