Государственное автономное профессиональное образовательное учреждение Свердловской области
«Уральский государственный колледж имени И.И. Ползунова»

Раздел № 1
«Установка и настройка SQL Server»

Лабораторная работа № 1.3
«Создание базы данных в среде разработки»

Цель: создать базу данных и таблицы в СУБД MS SQL Server, используя запросы. Заполнить таблицы данными, а также изучить и научиться использовать основные команды Transact-SQL.

База данных (database) – это организованная совокупность совместно используемых логически связанных данных и описаний этих данных, относящаяся к определенной предметной области, предназначенная для удовлетворения информационных потребностей организации.

Transact-SQL (T-SQL) — процедурное расширение языка SQL, созданное компанией Microsoft (для Microsoft SQL Server) и Sybase (для Sybase ASE).

Рассмотрим несколько команд T-SQL, а также их синтаксис.

Команда «CREATE DATABASE» используется для создания базы данных. Синтаксис команды: «CREATE DATABASE ИМЯ_БАЗЫ ДАННЫХ».

Далее для создания таблиц нашей базы данных будем многократно использовать инструкцию «CREATE TABLE». Синтаксис инструкции представлен на рисунке 1.

```
CREATE TABLE ИМЯ_ТАБЛИЦЫ (имя_первого_столбца тип данных,

...,
имя_последнего_столбца тип данных,
первичный ключ,
ограничения (не обязательно))
```

Рисунок 1 – Синтаксис конструкции «CREATE TABLE»

Типы данных, используемых в MS SQL Server, представлены на рисунке 2.

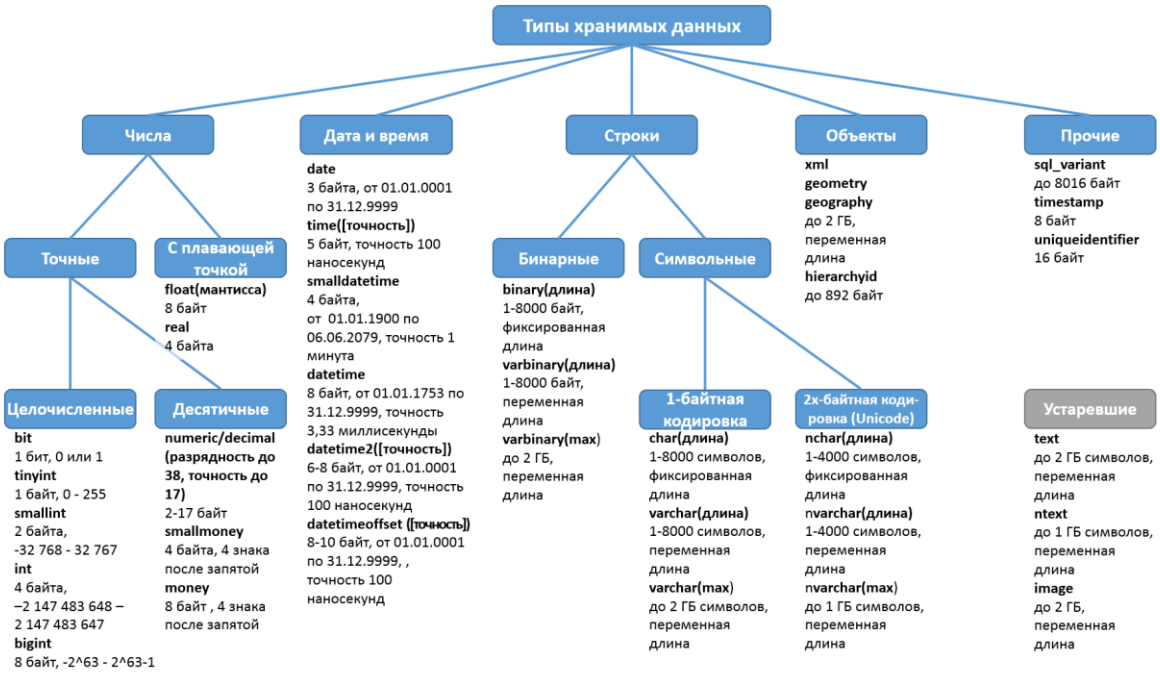


Рисунок 2 - Типы хранимых данных

В соответствии со стандартом SQL для таблицы можно задать одно или несколько ограничений, описанные на рисунке 3.

Ключ	Описание
NOT NULL	Запрет на вставку в столбец неопределенного значения NULL
UNIQUE	Значение столбца должно быть уникальным
PRIMARY KEY	Признак первичного ключа. Значение поля должно быть уникальным, оно не может содержать NULL, в таблице это ограничение может использоваться только один раз
CHECK	Ограничение-проверка на допустимое значение. В скобках за оператором CHECK указывается предикат, проверяющий допустимость значения
FOREIGN KEY и REFERENCES	Оба ограничения весьма похожи, единственное различие между ними в том, что FOREIGN KEY – ограничение внешнего ключа для таблицы, а REFERENCES – ссылка на столбец со значениями подстановки. В случае установки ограничения для таблицы сразу за FOREIGN KEY в скобках необходимо указать перечень столбцов, относящихся к ключу. В остальном синтаксис одинаков. Ограничения внешнего ключа требуют, чтобы все значения, присутствующие во внешнем ключе, соответствовали значениям родительского ключа (обеспечение ссылочной целостности)

Рисунок 3 – Ограничения столбцов таблицы

Расширенный синтаксис позволяет в момент описания связи настроить правила ссылочной целостности, которые определят поведение дочерней таблицы при изменении (ON UPDATE) или удалении (ON DELETE) связанных данных в родительской. В момент изменения значения первичного ключа и в момент удаления значения первичного ключа у родительской таблицы сервер осуществляет контроль ссылочной целостности данных. Если при создании внешнего ключа в инструкции SQL мы никак не конкретизировали поведение механизма контроля целостности, то они по умолчанию устанавливаются в режим «RESTRICT», остальные доступные варианты работы триггеров отражены на рисунке 4.

Действие	Описание
CASCADE	Изменение значения первичного ключа приводит к автоматическому изменению соответствующих значений внешнего ключа
SET NULL	При изменении значения или удалении первичного ключа все значения в связанных с внешним ключом колонках устанавливаются в NULL. В результате в дочерней таблице появляются «брошенные» строки, утерявшие связь с соответствующей записью из главной таблицы
RESTRICT	Режим по умолчанию. Внешний ключ воспринимает только значения первичного ключа или NULL. Попытки поместить в него какие-либо другие значения будут отвергаться
NO ACTION	То же самое, что и RESTRICT

Рисунок 4 - Поведение дочерней таблицы при изменении/удалении значения PK

Единожды созданная таблица не является статичным объектом, в SQL предусмотрена возможность модификации структуры уже существующих таблиц. Для этой цели реализована инструкция «ALTER TABLE». Синтаксис команды представлен на рисунке 5. Параметры инструкции представлены на рисунке 6.

```
ALTER TABLE имя_таблицы
{ADD[COLUMN] определение столбца}
| {ALTER [COLUMN] имя изменяющегося столбца}
| {DROP [COLUMN] имя_столбца RESTRICT | CASCADE}
| {ADD определение ограничения для таблицы}
| {DROP CONSTRAINT имя_ограничения RESTRICT | CASCADE}
```

Рисунок 5 - Синтаксис инструкции ALTER TABLE

Оператор	Описание
ADD [COLUMN]	Добавляет в таблицу новый столбец. Новый столбец определяется так же, как в операторе CREATE TABLE
ALTER [COLUMN]	Используется для создания или отмены значения по умолчанию для столбца
DROP [COLUMN]	Удаляет из таблицы столбец. При использовании параметра RESTRICT перед удалением столбца СУБД проверит наличие ссылок на него из других таблиц и представлений. Если таковые имеются, то столбец не будет удален. Наоборот, при использовании параметра CASCADE вместе со столбцом будут удалены все объекты, ссылающиеся на него
ADD	Позволяет добавить к таблице новое ограничение
DROP CONSTRAINT	Удаляет уже существующие ограничения. Если в предложении задан параметр RESTRICT, то в этот момент столбец не должен использоваться как родительский ключ для внешнего ключа другой таблицы. Если передается параметр CASCADE, то внешние ключи, имеющие ссылки или ограничения FOREIGN KEY, уничтожаются

Рисунок 6 – Параметры инструкции ALTER TABLE

Назначение инструкции «INSERT» – вставка в таблицу одной или нескольких строк. Практически все диалекты SQL на все 100 % поддерживают синтаксическую конструкцию, рекомендованную стандартом. На рисунке 7 представлен синтаксис данной инструкции.

```
INSERT INTO {имя таблицы | имя представления}
{имя столбца[,...]}
{DEFAULT VALUES | VALUES (Значение1[,...]) | инструкция SELECT}
```

Рисунок 7 – Синтаксис инструкции INSERT

Инструкции «SELECT» являются едва ли не самым важным элементом языка SQL, ведь именно для выбора данных изначально и создавался язык запросов (в переводе с англ. «select» означает «выбрать»). Исходным материалом для операций выборки выступают одна или несколько таблиц и представлений. Синтаксис данной инструкции представлен на рисунке 8.

```
SELECT [DISTINCT|ALL]
   { имя поля [AS псевдоним] [,...]
    |функция_агрегирования [AS псевдоним] [,...]
    |выражение для вычисления значения [AS псевдоним] [,...]
    |спецификатор.* }
FROM
  {имя_таблицы [AS псевдоним] [,...]
  |имя_представления [AS псевдоним][,...]}
[WHERE условия отбора]
  [GROUP BY имя поля [,...]] [HAVING условие]
  [ORDER BY имя поля [ASC | DESC] [,...]]
```

Рисунок 8 – Синтаксис инструкции «SELECT»

ПРАКТИЧЕСКАЯ ЧАСТЬ

1. Для создания базы данных откроем MS SQL Server Management Studio, затем в панели инструментов выберем «Создать запрос», как показано на рисунке 9.

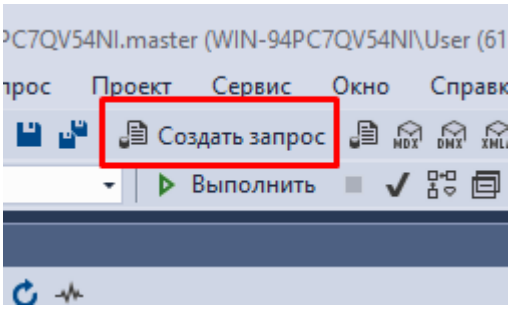


Рисунок 9 – Кнопка «Создать запрос»

2. Создадим базу данных agency. В запрос вносим следующее содержимое: «CREATE DATABASE agency», как показано на рисунке 10. После этого нажимаем «Выполнить».

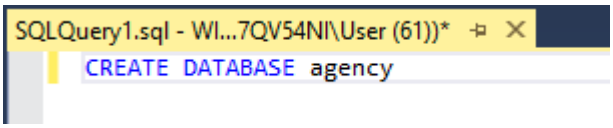


Рисунок 10 – Содержимое запроса для создания БД

3. Об успешном выполнении запроса будет свидетельствовать сообщение, изображенное на рисунке 11.

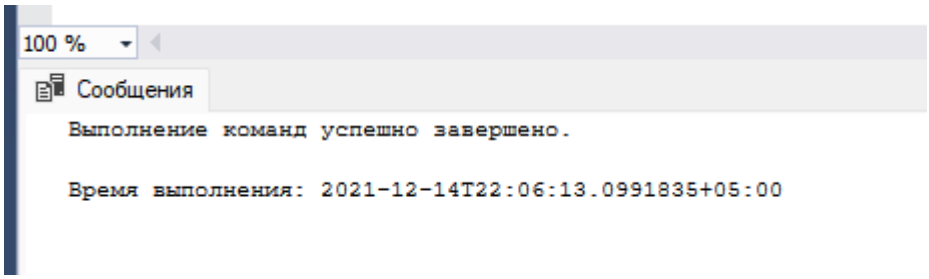


Рисунок 11 – Успешное выполнение запроса

4. После этого можно увидеть, что в обозревателе объектов отобразилась база данных «agency». Нажатием на эту БД правой кнопкой мыши выведем список параметров и выберем «Создать запрос», как показано на рисунке 12.

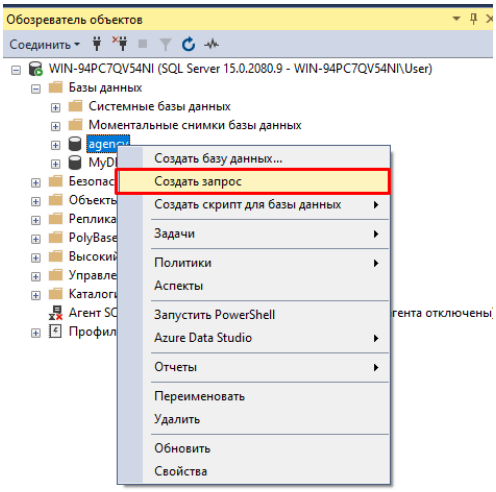


Рисунок 12 – Создание запроса

5. Рассмотрим процесс создания таблиц в выбранной базе данных. На рисунке представлена схема, созданной базы данных «agency». Внимательно изучите схему, изображенную на рисунке 13.

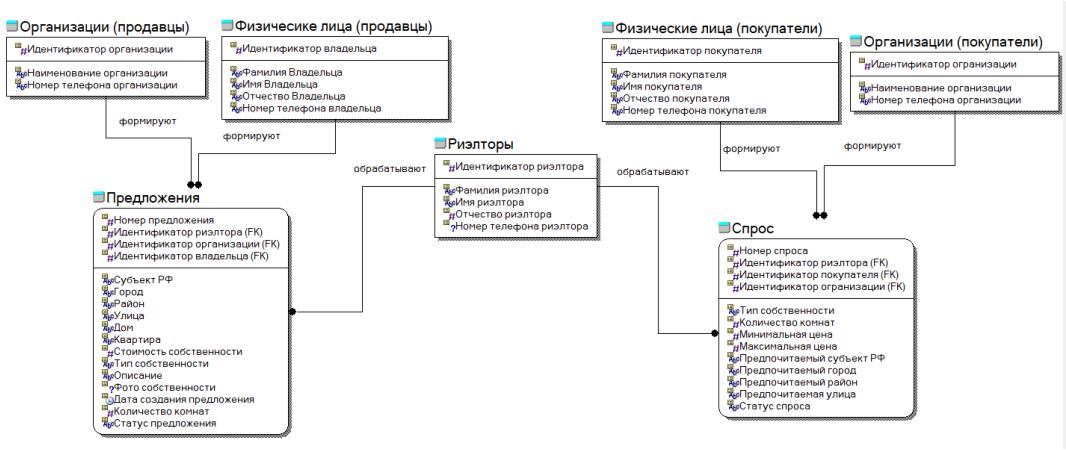


Рисунок 13 – Схема базы данных

6. Используя теорию в начале лабораторной работы, создадим 4 таблицы из этой схемы (Организации, Владельцы, Предложения, Риэлторы).

SQL-запрос представлен на рисунке 14. Записи, находящиеся между символами /* */ являются комментариями (их писать необязательно).

```
CREATE TABLE owner_fiz   /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_owner_f INT PRIMARY KEY IDENTITY NOT NULL,   /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
surname VARCHAR(40) NOT NULL,   /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
name_owner VARCHAR(40) NOT NULL, /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
patronymic VARCHAR(40), /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
num_phone VARCHAR(12) NOT NULL UNIQUE /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)


CREATE TABLE owner_company  /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_owner_c INT PRIMARY KEY IDENTITY NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
name_company VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
num_phone VARCHAR(12) NOT NULL UNIQUE  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)


CREATE TABLE realtor  /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_realt INT PRIMARY KEY IDENTITY NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
surname VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
name_realtor VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
patronymic VARCHAR(40),  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
num_phone VARCHAR(12) NOT NULL UNIQUE  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)


CREATE TABLE offer  /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_offer INT PRIMARY KEY IDENTITY NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
subject_rf  VARCHAR(40) NOT NULL, /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
city VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
district VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
street VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
house VARCHAR(10) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
apartment VARCHAR(10),  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
price INT NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
type_own VARCHAR(20) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
description_offer VARCHAR,  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
photo IMAGE,  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
date_offer DATE NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
num_room SMALLINT,  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
status_offer varchar(20) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
id_realt INT NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
id_owner_c INT,  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
id_owner_f INT,  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
FOREIGN KEY (id_owner_f) REFERENCES owner_fiz (id_owner_f),  /*СОЗДАНИЕ ВНЕШНЕГО КЛЮЧА*/
FOREIGN KEY (id_owner_c) REFERENCES owner_company (id_owner_c),  /*СОЗДАНИЕ ВНЕШНЕГО КЛЮЧА*/
FOREIGN KEY (id_realt) REFERENCES realtor (id_realt),  /*СОЗДАНИЕ ВНЕШНЕГО КЛЮЧА*/
CHECK (status_offer = 'Актуально' or status_offer = 'Неактуально')  /*УСЛОВИЕ ЗАПОЛНЕНИЯ СТОЛБЦА*/
)
```

Рисунок 14 – SQL-запрос для создания таблиц в базе данных

7. После написания запроса, нажимаем «Выполнить». Результат удачного выполнения запроса показан на рисунке 15.

```
CREATE TABLE owner_fiz   /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_owner_f INT PRIMARY KEY IDENTITY NOT NULL,   /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
surname VARCHAR(40) NOT NULL,   /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
name_owner VARCHAR(40) NOT NULL, /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
patronymic VARCHAR(40), /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
num_phone VARCHAR(12) NOT NULL UNIQUE /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)


CREATE TABLE owner_company  /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_owner_c INT PRIMARY KEY IDENTITY NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
name_company VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
num_phone VARCHAR(12) NOT NULL UNIQUE  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)


CREATE TABLE realtor  /*СОЗДАНИЕ ТАБЛИЦЫ*/
(
id_realt INT PRIMARY KEY IDENTITY NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: ПЕРВИЧНЫЙ КЛЮЧ, ИДЕНТИФИКАТОР, БЕЗ ЗНАЧЕНИЯ NULL*/
surname VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
name_realtor VARCHAR(40) NOT NULL,  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL*/
patronymic VARCHAR(40),  /*СОЗДАНИЕ СТОЛБЦА БЕЗ ПАРАМЕТРОВ*/
num_phone VARCHAR(12) NOT NULL UNIQUE  /*СОЗДАНИЕ СТОЛБЦА С ПАРАМЕТРАМИ: БЕЗ ЗНАЧЕНИЯ NULL, УНИКАЛЬНОЕ ЗНАЧЕНИЕ*/
)
```

100 %

Сообщения

```
Выполнение команд успешно завершено.

Время выполнения: 2021-12-31T15:16:40.3841418+05:00
```

Рисунок 15 – Выполнение запроса

8. После выполнения запроса, в обозревателе должны появиться 4 таблицы, как показано на рисунке 16.

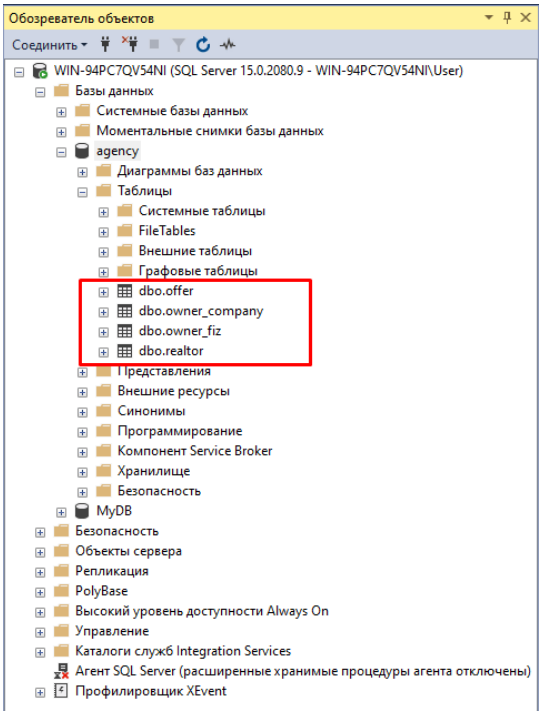


Рисунок 16 – Созданные таблицы

САМОСТОЯТЕЛЬНОЕ ЗАДАНИЕ

В качестве самостоятельного задания, необходимо создать оставшиеся 3 таблицы из схемы (см. пункт 5). При создании таблиц используйте теорию в начале лабораторной работы и пример создания первых 4 таблиц (см. пункт 6). Обратите внимание на то, каким столбцам необходимо присвоить внешний ключ. В таблице demand (Спрос) необходимо создать ограничение: столбец status_demand (Статус спроса) должен иметь значения либо «Актуально», либо «Неактуально».

Таблица «buyer_fiz» (Покупатели) состоит из следующих столбцов и параметров:

- «id_buyer_f» (Тип данных int; первичный ключ; идентификатор; без значения NULL),
- «surname» (Тип данных varchar(40); без значения NULL);
- «name_buyer» (Тип данных varchar(40); без значения NULL);
- «patronymic» (Тип данных varchar(40));
- «num_phone» (Тип данных varchar(12); без значения NULL; уникальное значение).

Таблица «buyer_company» (Организации) состоит из следующих столбцов и параметров:

- «id_buyer_c» (Тип данных int; первичный ключ; идентификатор; без значения NULL);
- «name_company» (Тип данных varchar(40); без значения NULL);
- «num_phone» (Тип данных varchar(12); без значения NULL; уникальное значение).

Таблица «demand» (Спрос) состоит из следующих столбцов и параметров:

– «id_demand» (Тип данных int; первичный ключ; идентификатор; без значения NULL),
– «type_own» (Тип данных varchar(20); без значения NULL);
– «num_room» (Тип данных smallint);
– «min_price» (Тип данных int; без значения NULL);
– «max_price» (Тип данных int; без значения NULL);
– «subject_rf» (Тип данных varchar(40); без значения NULL);
– «city» (Тип данных varchar(40); без значения NULL);
– «district» (Тип данных varchar(40); без значения NULL);
– «street» (Тип данных varchar(40); без значения NULL);
– «status_demand» (Тип данных varchar(20); без значения NULL);
– «id_realt» (Тип данных int; без значения NULL);
– «id_buyer_c» (Тип данных int);
– «id_buyer_f» (Тип данных int).

После выполнения самостоятельного задания, в обозревателе должно появиться еще 3 таблицы.

9. Заполним данными наши таблицы. Для этого будем использовать как запросы, так и графический интерфейс MS SQL Server. Заполним таблицы «Организации», «Владельцы», «Риэлторы», используя запросы. Для проверки выполнения запроса, выведем все три таблицы, с помощью инструкции выборки данных «SELECT». Выполнение запроса продемонстрировано на рисунке 17.

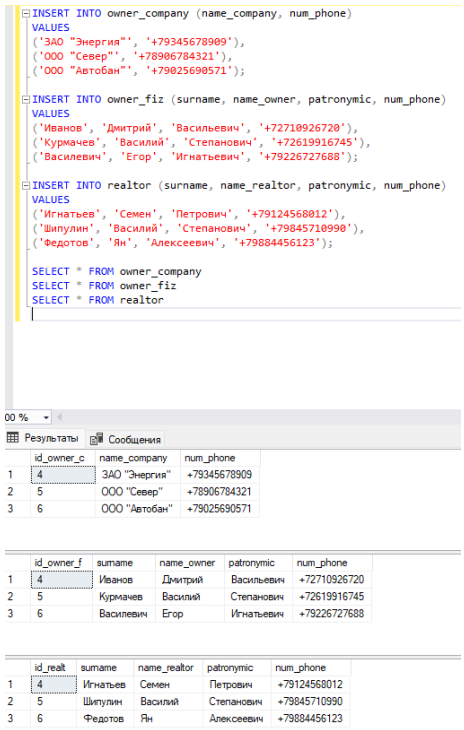
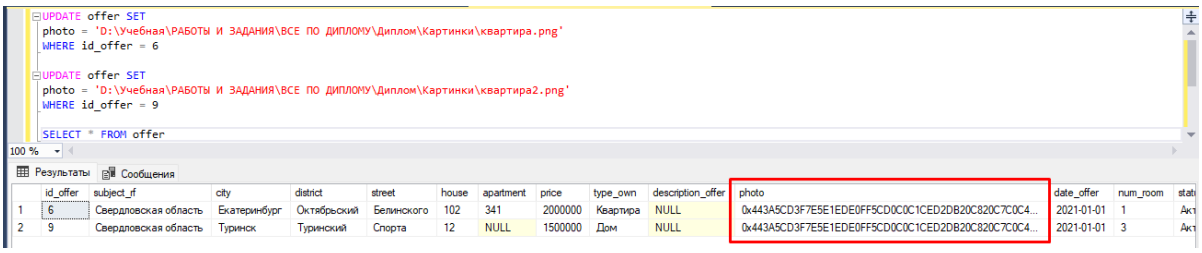


Рисунок 17 – Выполнение запроса для добавления данных

10. Заполним таблицу «Предложения», используя графический интерфейс. Для этого выводим список параметров таблицы Предложения и выбираем пункт «Изменить первые 200 строк». После этого просто вносим значения в строки (столбец description_offer (Описание) заполнять не нужно). Столбец «photo» необходимо заполнить, с помощью запроса, его мы выполним в пункте 12.

11. Для того, чтобы добавить фото, необходимо использовать конструкцию «UPDATE». В значении необходимо указать путь до картинки.



	id_offer	subject_fl	city	district	street	house	apartment	price	type_own	description_offer	photo	date_offer	num_room	stati
1	6	Свердловская область	Екатеринбург	Октябрьский	Белинского	102	341	2000000	Квартира	NULL	0x443A5CD3F7E5E1EDE0FF5CD0C0C1CED2DB20C820C7C0C4...	2021-01-01	1	Акт
2	9	Свердловская область	Туринск	Туринский	Спорта	12	NULL	1500000	Дом	NULL	0x443A5CD3F7E5E1EDE0FF5CD0C0C1CED2DB20C820C7C0C4...	2021-01-01	3	Акт

Рисунок 18 – Добавление картинки в таблицу

12. Открываем вкладку «Сервис», затем выбираем «Конструкторы» и убираем галочку с пункта «Запретить сохранение изменений, требующих повторно создания таблицы», как показано на рисунке 19. Это требуется для того, чтобы можно было изменить тип данных у столбца Описание (description_offer) в таблице Предложения (offer).

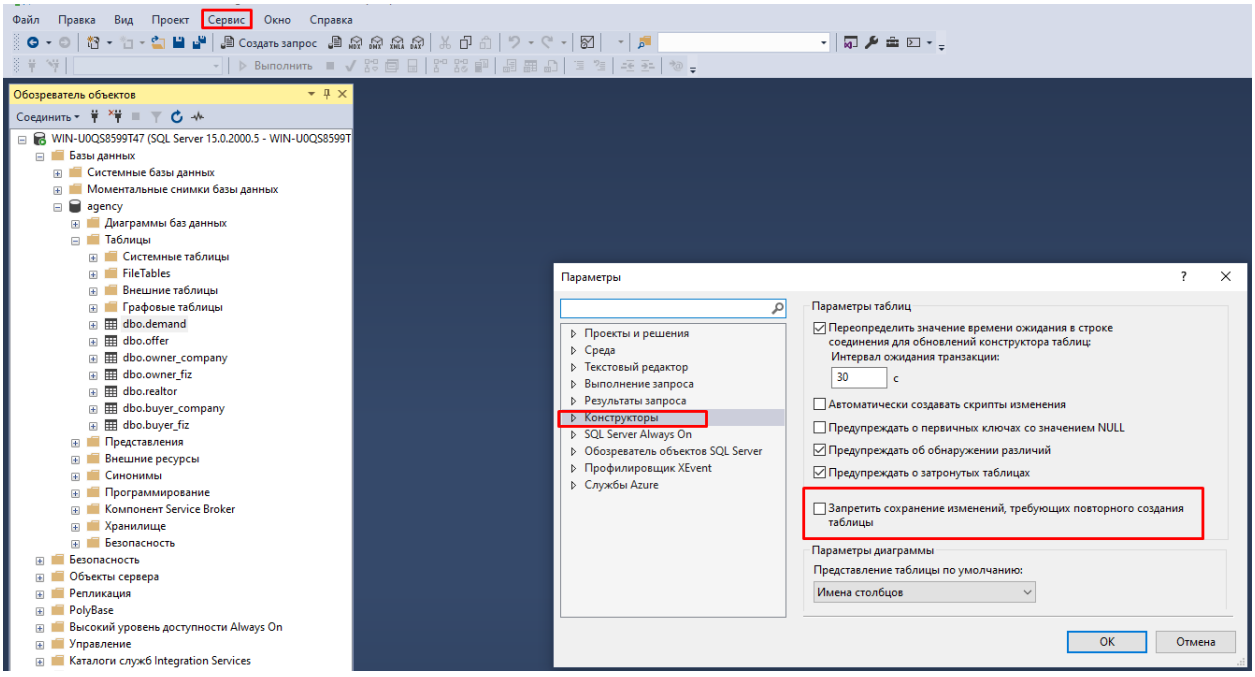


Рисунок 19 – Разрешение сохранений изменений, требующих повторного создания таблицы

13. Для того, чтобы изменить тип данных у столбца «Описание» (в данной базе данных он имеет имя «description_offer»), необходимо в обозревателе открыть параметры таблицы «Предложения» (offer), затем выбрать пункт «Проект». После этого в необходимом столбце (в нашем случаем «description_offer») изменить тип данных. Выберем varchar(200), как

показано на рисунке 20. Сохраняем и закрываем окно «Проект», как показано на рисунке 21.

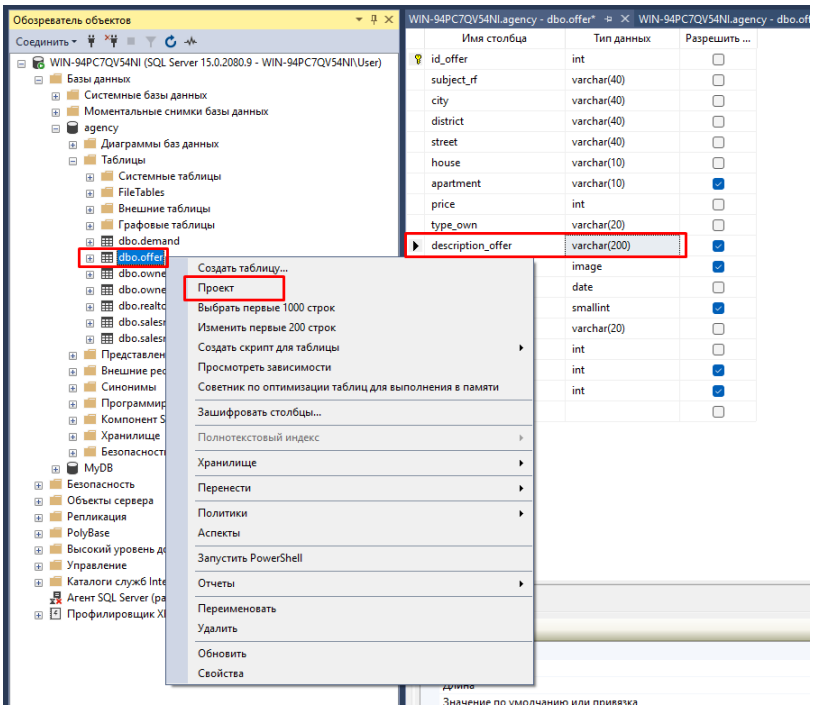


Рисунок 20 – Изменение типа данных

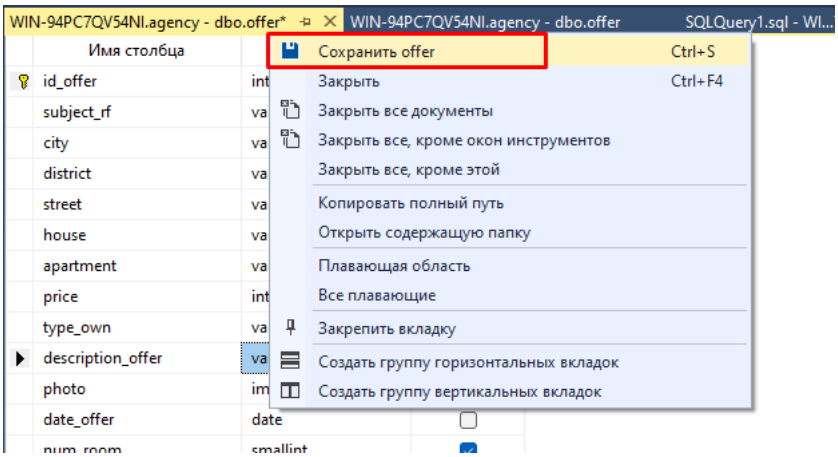


Рисунок 21 – Сохранение проекта

14. Теперь необходимо заполнить столбец Описание в таблице «Предложения» (offer). Способ заполнения выберите самостоятельно.

САМОСТОЯТЕЛЬНОЕ ЗАДАНИЕ

В качестве самостоятельного задания необходимо: заполнить оставшиеся таблицы. Две любые таблицы заполнить, с помощью запросов, а оставшуюся – с помощью графического интерфейса.