**Практическая работа №1.1**

**Тема**

Создание базы данных.

**Цель работы**

По заданной схеме данных разработать базу данных, создать и сохранить скрипт с БД.

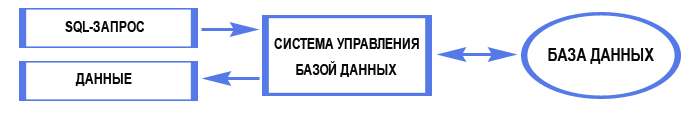
**Теоретический материал**

*База данных* - набор сведений, хранящихся некоторым упорядоченным способом.

*Система управления базами данных* — это совокупность языковых и программных средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять, обеспечивает безопасность данных и т.д.

*SQL* - язык структурированных запросов, основной задачей которого является предоставление простого способа считывания и записи информации в базу данных.

Простейшая схема работы с базой данных выглядит примерно так:



В реляционной базе данных все данные можно представить в виде простых таблиц. Таблицы в реляционных базах данных обладают рядом свойств. Основными являются следующие:

* В таблице не может быть двух одинаковых строк.
* Столбцы располагаются в определенном порядке, который создается при создании таблицы. В таблице может не быть ни одной строки, но обязательно должен быть хотя бы один столбец.
* У каждого столбца есть уникальное имя (в пределах таблицы), и все значения в одном столбце имеют один тип (число, текст, дата...).
* На пересечении каждого столбца и строки может находиться только атомарное значение (одно значение, не состоящее из группы значений). Таблицы, удовлетворяющие этому условию, называют нормализованными.

*Первичный ключ* (сокращенно РК - primary key) - столбец, значения которого во всех строках различны.

В реляционных базах данных некоторая информация из одних таблиц присутствует в других, т.е. между ними имеются связи.

Всего существует 3 типа связей:

* [Один к одному](https://office-menu.ru/uroki-sql/41-tipy-svyazej-v-relyatsionnykh-bazakh-dannykh#onetoone);
* [Один ко многим](https://office-menu.ru/uroki-sql/41-tipy-svyazej-v-relyatsionnykh-bazakh-dannykh#onetomany);
* [Многие ко многим](https://office-menu.ru/uroki-sql/41-tipy-svyazej-v-relyatsionnykh-bazakh-dannykh#manytomany).

Ход работы:

В ходе работы была открыта программа SQL Server в котором на панели задач открыли вкладку сервер и убрали галочку с запретить изменения сохранения.

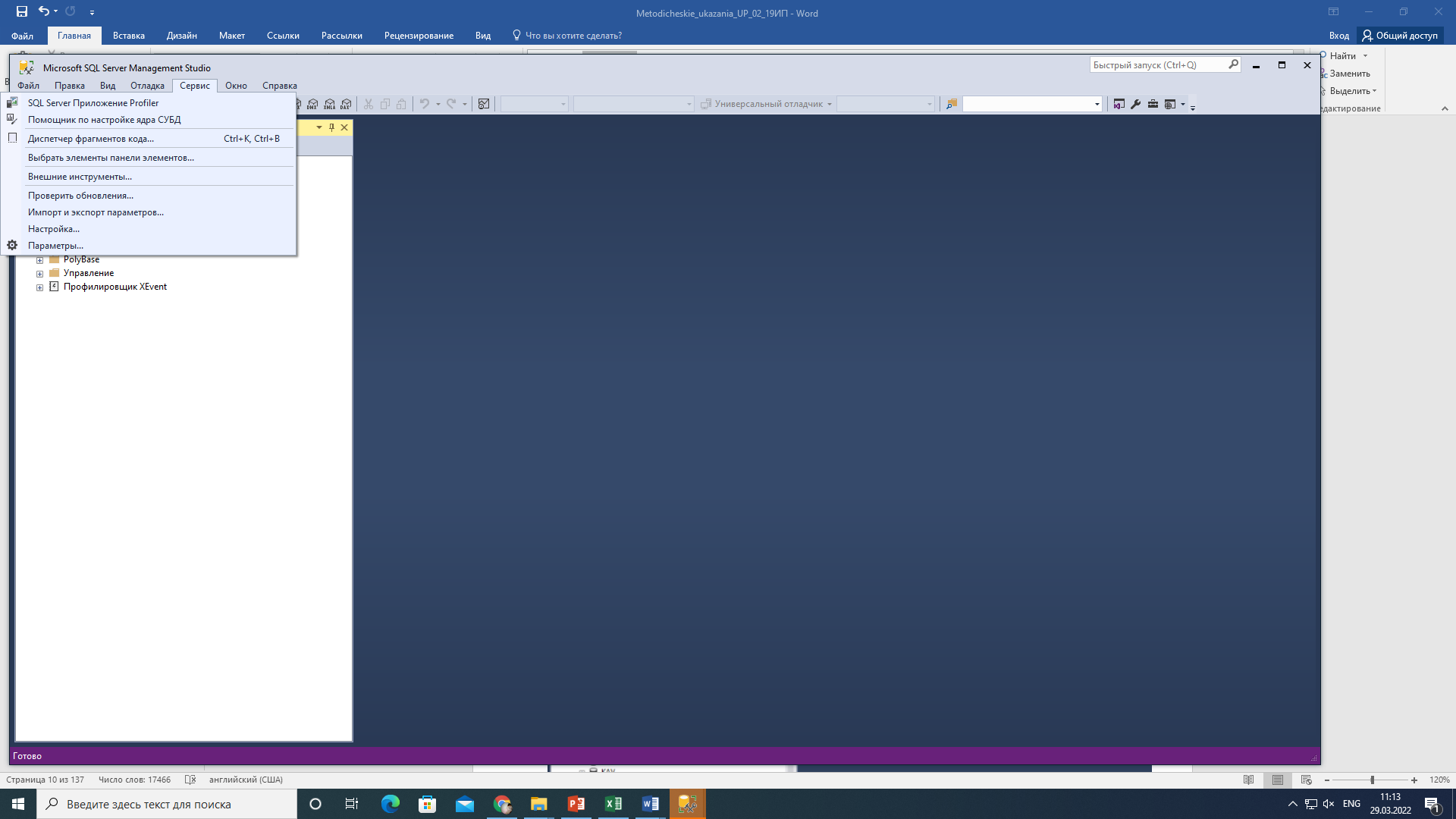


Рисунок 1. Главное меню.

Выберите в главном меню пункт «Параметры» (рисунок 1-2).

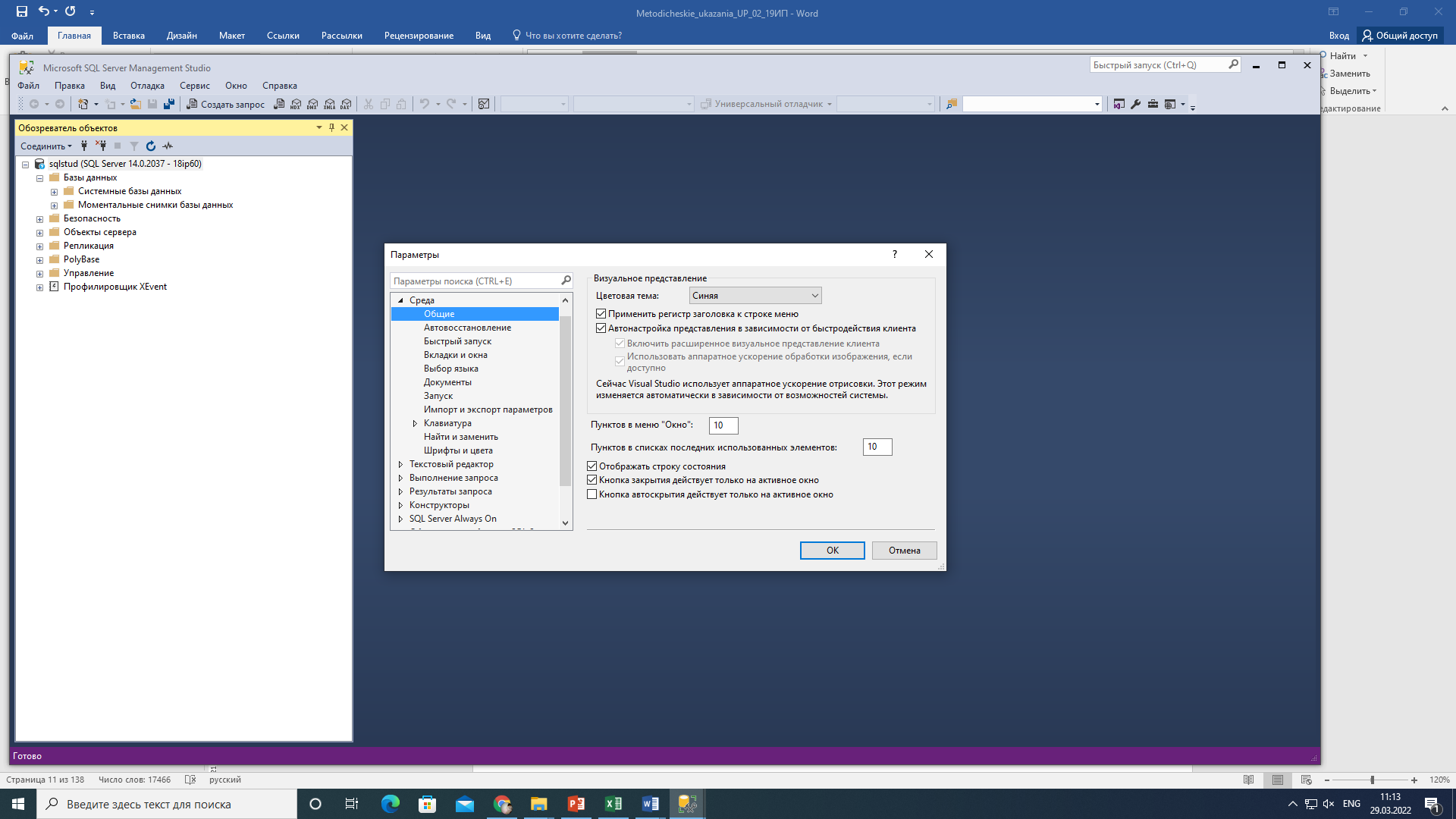


Рисунок 2. Окно «Параметры»

Уберите галочку у пункта «Запретить сохранение изменений, требующих повторного создания таблиц» (рисунок 3-4).

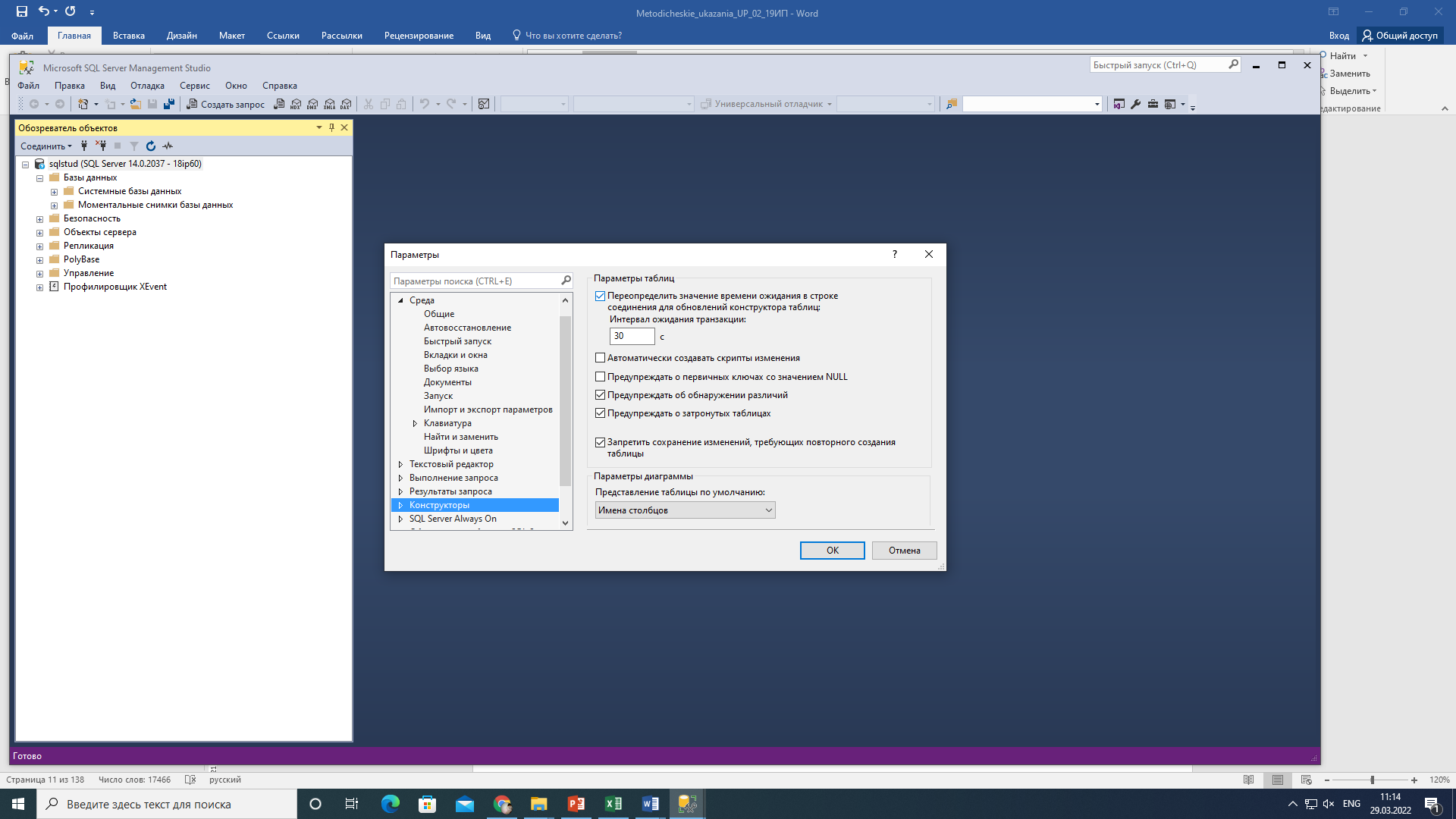


Рисунок 3 Окно «Параметры»

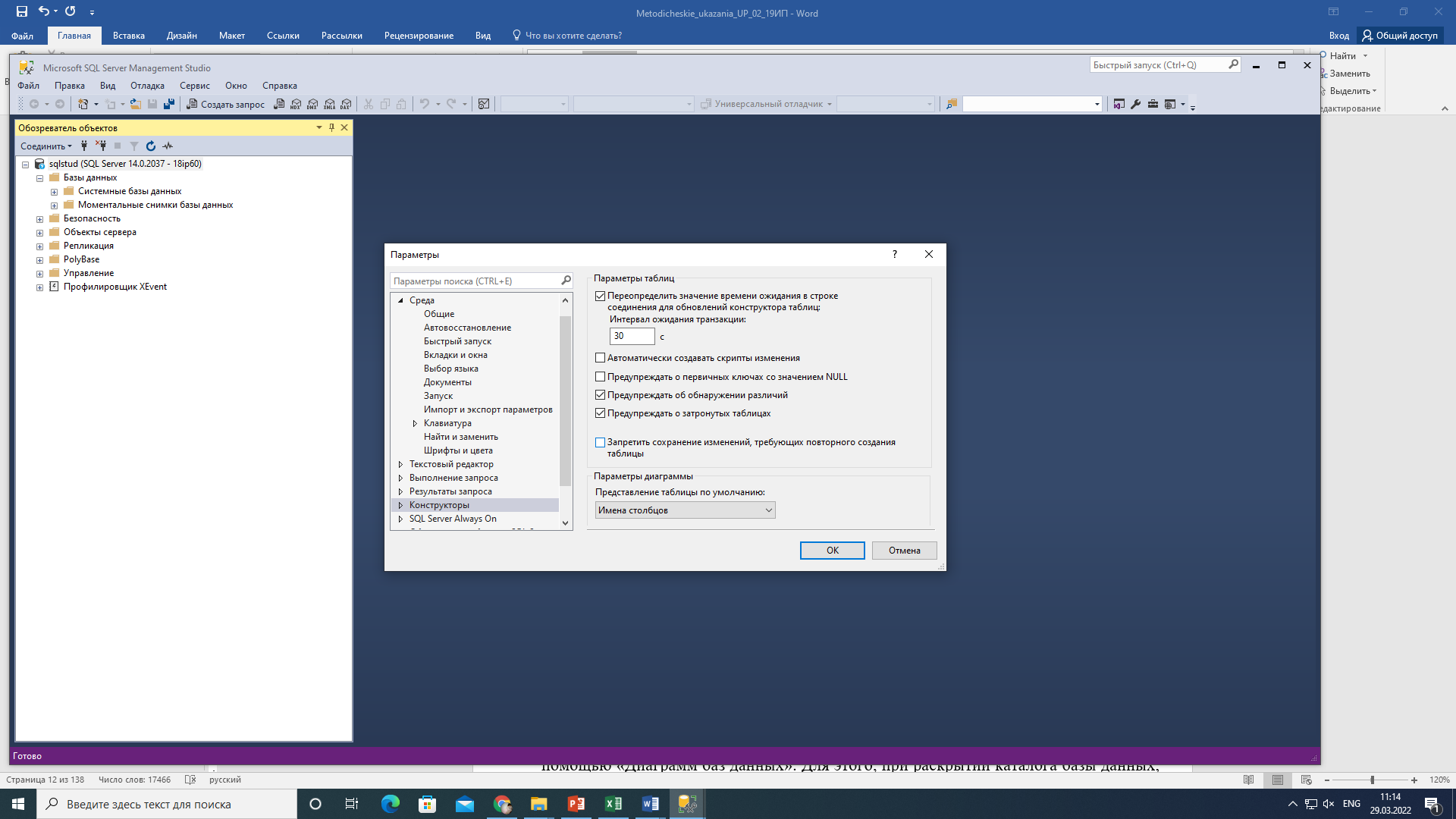


Рисунок 4. Окно «Параметры»

**Создание схемы данных.**

Схема базы данных используется для описания архитектуры базы данных, и она помогает обеспечить следующее:

* Единообразное форматирование записей данных.
* Все записи имеют уникальный первичный ключ.
* Никакие важные данные не пропущены.

Наиболее быстрым способ для создания базы данных является создание ее с помощью «Диаграмм баз данных». Для этого, при раскрытии каталога базы данных, необходимо кликнуть правой кнопкой мыши по папке «Диаграммы базы данных» и в отрывшемся меню выбрать «Создать диаграмму базы данных»

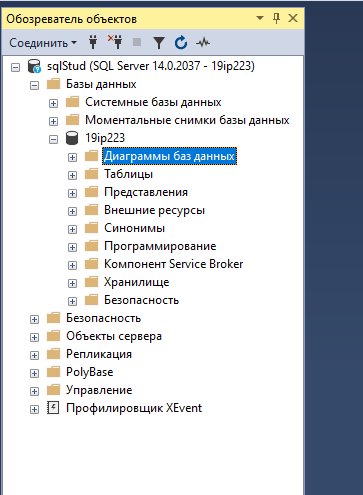


Рисунок 5. Выбор пункта «Создание диаграммы базы данных»

После того как была создана диаграмма, в неё были включены таблицы по области библиотека, так же были заданы ключи и соединения между таблицами.

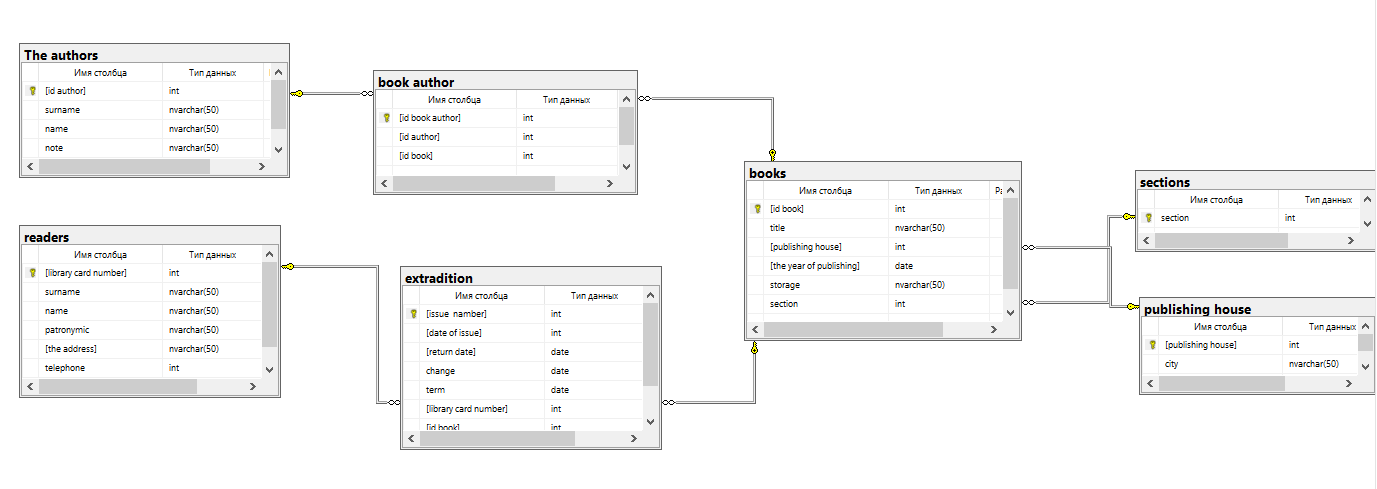


Рисунок 6 таблицы базы данных

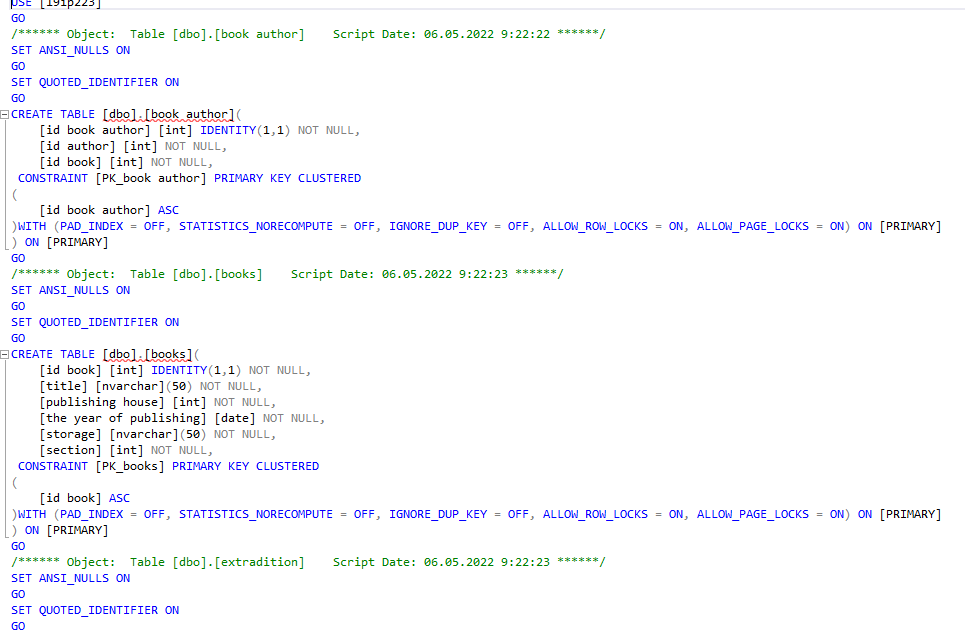


Рисунок 7 скрипт

**Контрольные вопросы**

1. Что такое «база данных»?
2. Что такое первичный ключ (Primary key)?
3. Что такое внешний ключ (Foreign key)?
4. Какие связи существуют в реляционных базах данных?
5. Что подразумевается под целостностью данных?
6. Что такое нормализация и каковы ее преимущества?

**Практическая работа №1.2**

**Тема**

Подключение базы данных.

**Цель работы**

По заданной схеме данных разработать базу данных, создать и сохранить скрипт с БД, реализовать программное приложение с возможностью авторизации в системе.

**Теоретический материал**

Windows Presentation Foundation (WPF) — аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

Преимущества WPF:

* Использование традиционных языков .NET-платформы - C# и VB.NET для создания логики приложения;
* Возможность декларативного определения графического интерфейса с помощью специального языка разметки XAML, основанном на xml и представляющем альтернативу программному созданию графики и элементов управления, а также возможность комбинировать XAML и C#/VB.NET;
* Независимость от разрешения экрана: поскольку в WPF все элементы измеряются в независимых от устройства единицах, приложения на WPF легко масштабируются под разные экраны с разным разрешением;
* Новые возможности, которых сложно было достичь в WinForms, например, создание трехмерных моделей, привязка данных, использование таких элементов, как стили, шаблоны, темы и др.;
* Хорошее взаимодействие с WinForms, благодаря чему, например, в приложениях WPF можно использовать традиционные элементы управления из WinForms;
* Богатые возможности по созданию различных приложений: это и мультимедиа, и двухмерная и трехмерная графика, и богатый набор встроенных элементов управления, а также возможность самим создавать новые элементы, создание анимаций, привязка данных, стили, шаблоны, темы и многое другое;
* Аппаратное ускорение графики - вне зависимости от того, работаете ли вы с 2D или 3D, графикой или текстом, все компоненты приложения транслируются в объекты, понятные Direct3D, и затем визуализируются с помощью процессора на видеокарте, что повышает производительность, делает графику более плавной;
* Создание приложений под множество ОС семейства Windows - от Windows XP до Windows 10.

Также стоит учитывать, что по сравнению с приложениями на Windows Forms объем программ на WPF и потребление ими памяти в процессе работы в среднем несколько выше. Но это с лихвой компенсируется более широкими графическими возможностями и повышенной производительностью при отрисовке графики.

**Ход работы:**

В ходе работы была запущена программа Visual Studio в которой создали проект WPF

**Подключение базы данных.**

Откроем среду разработки Visual Studio и создадим новый проект «WPF». В качестве названия проекта укажите наименование вашей предметной области + «IS» (рисунок 7). Нажмите кнопку «Ок».

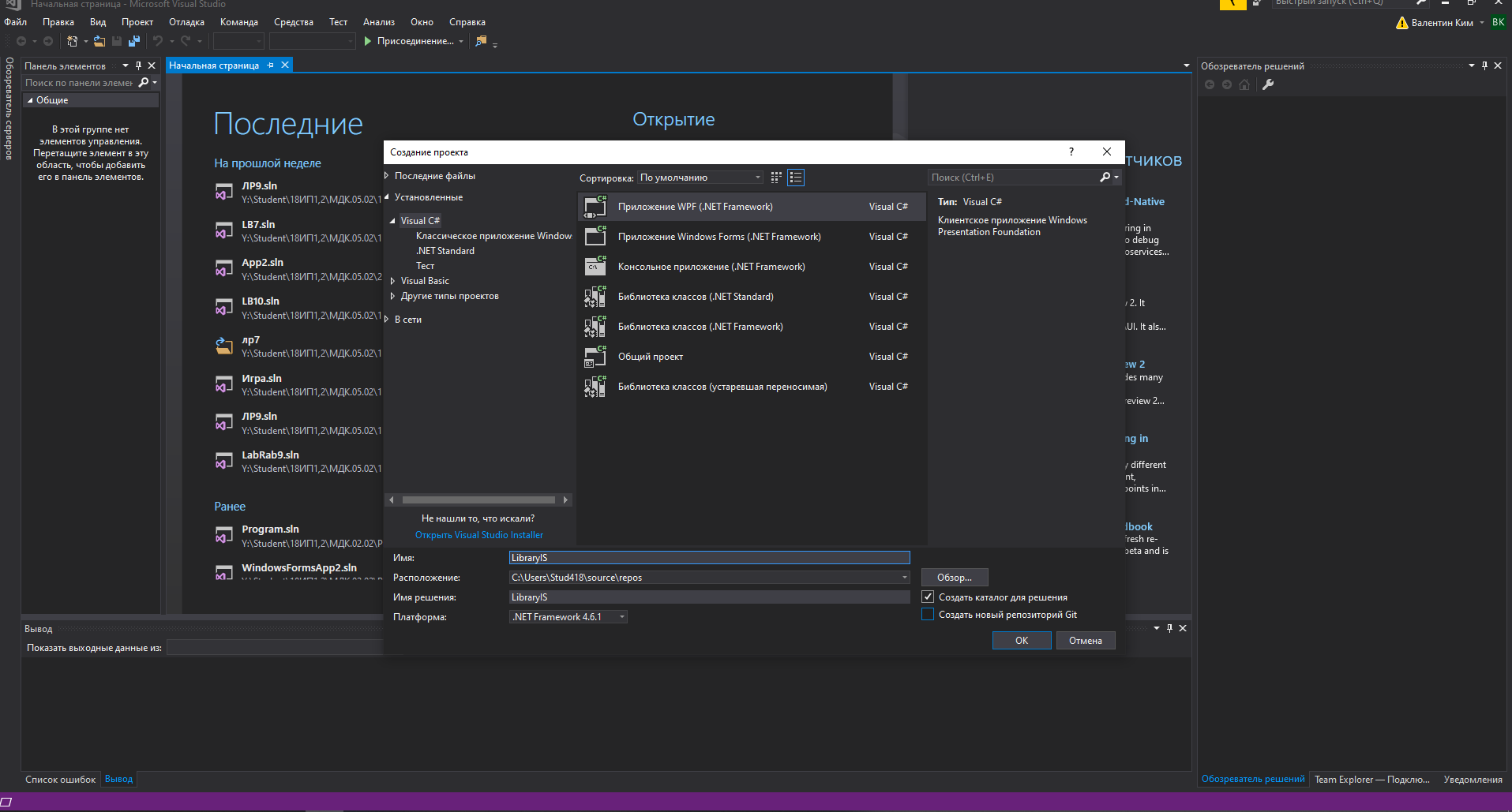


Рисунок 7. Окно «Создание проекта»

Следующим шагом будет подключение базы данных к проекту. Для этого на панели проекта нажмем правой кнопкой мыши по нашему проекту и выберем пункт «Добавить» -> «Создать новый элемент» (рисунок 8).

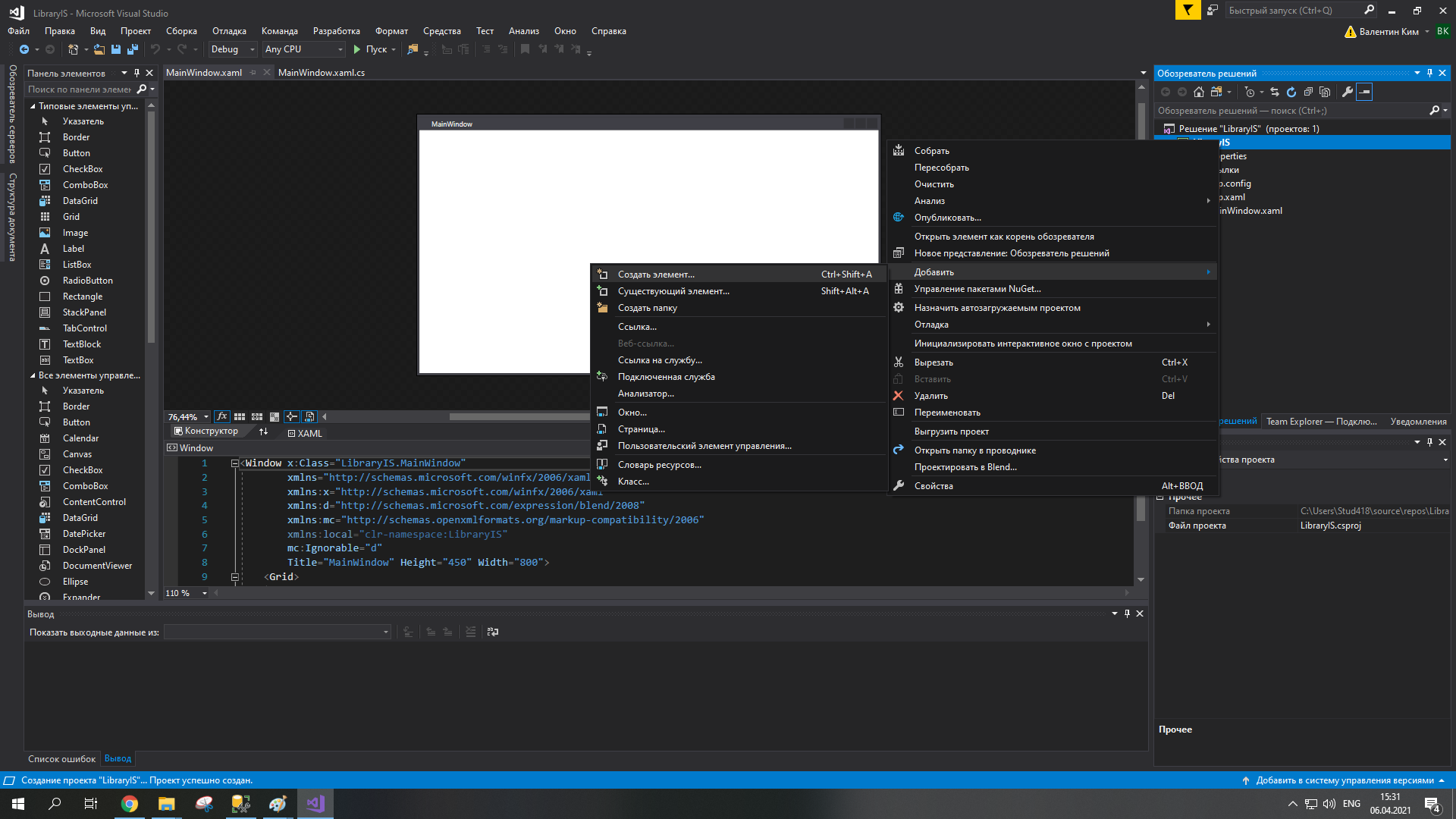


Рисунок 8. Создание нового элемента

Из списка элементов выберем элемент «Модель ADO.NET EDM». Зададим данному элементу название. (рисунок 9).

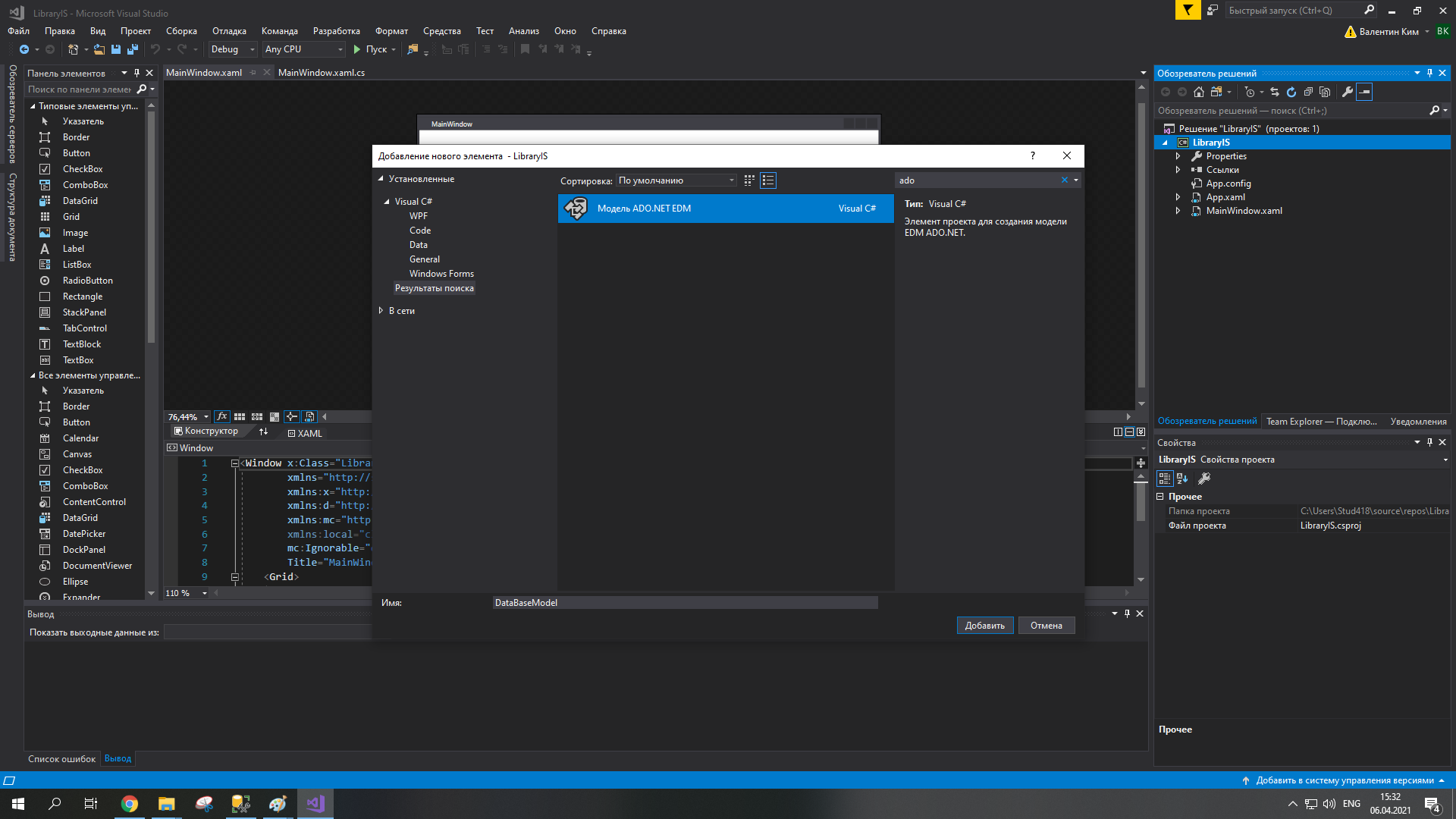


Рисунок 9. Создание нового элемента «Модель ADO.NET EDM».

Откроется мастер создания моделей EDM. Выбираем «Конструктор EF из базы данных» и жмем кнопку «Далее» (рисунок 10).

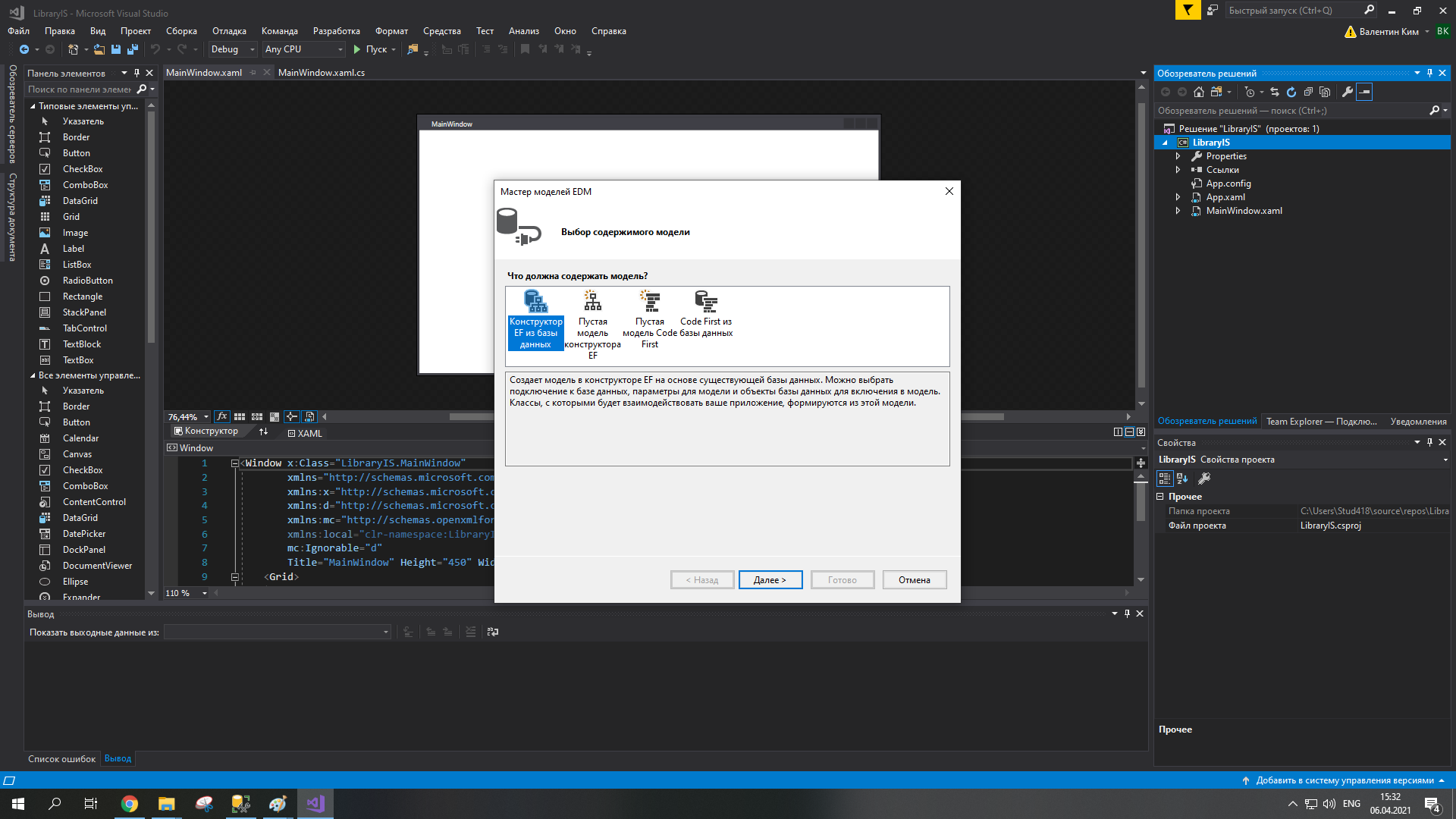


Рисунок 10. Мастер моделей EDM, шаг 1

В следующем окне нажимаем на кнопку «Создать соединение» (рисунок 11).

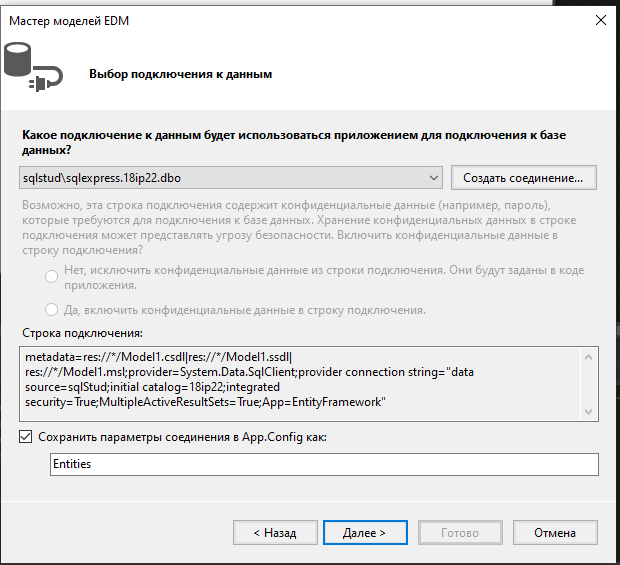


Рисунок 11. Мастер моделей EDM, шаг 2.

Задали соединение где были прописаны данные для подключения. После чего было произведено соединение где произошла выгрузка таблиц в среду Visual Studio (рисунок 12)

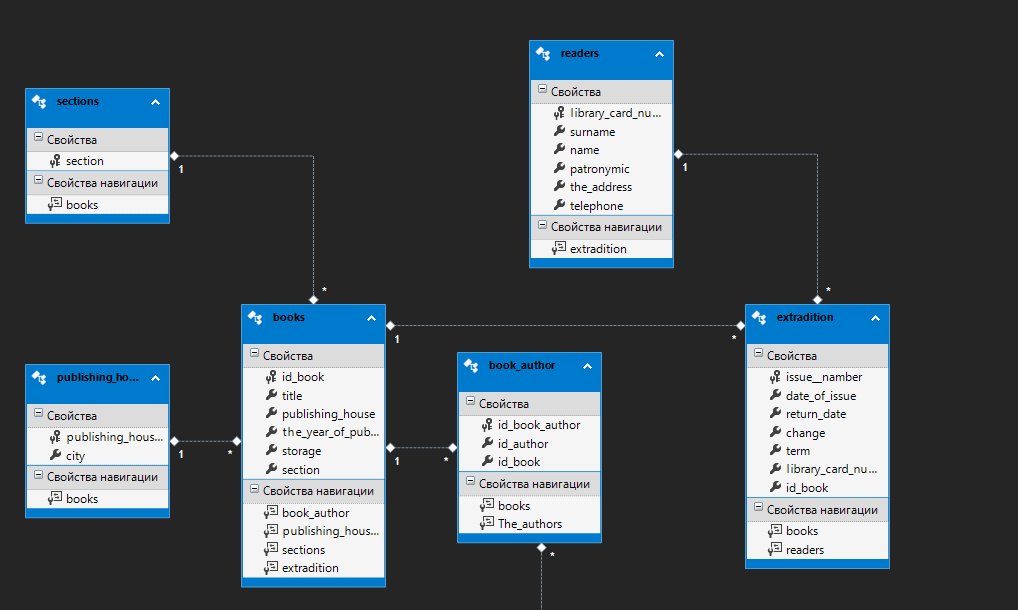


Рисунок 12 база данных

Теперь нужно реализовать возможность работать с базой данных непосредственно из кода.

Для этого был создан новый класс, например, с названием DataBase (рисунок 13).

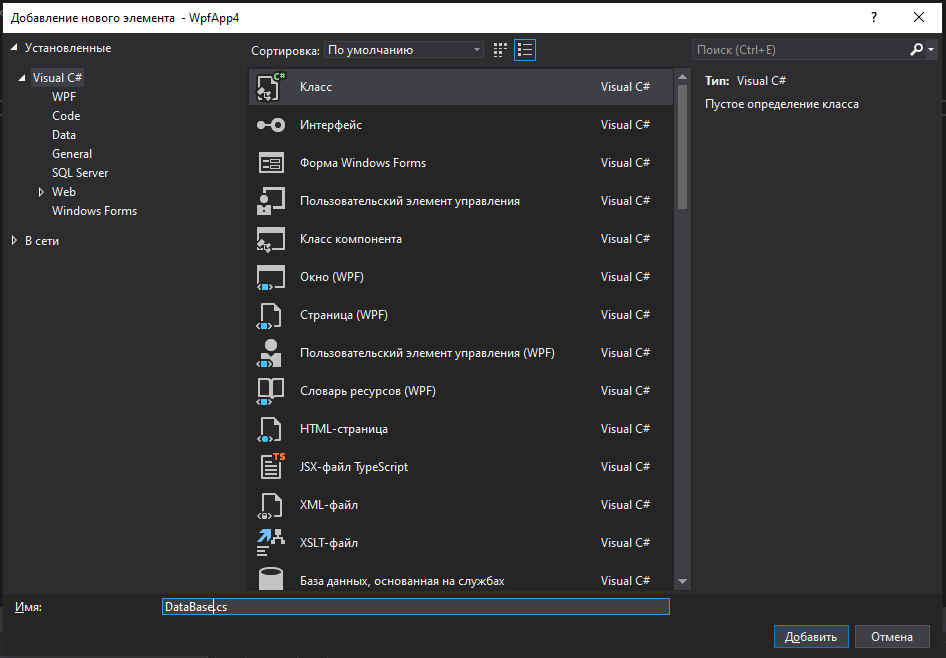
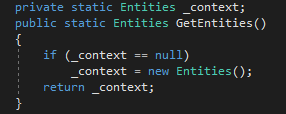


Рисунок 13. Добавление нового элемента.

После того как был создан класс, был прописан публичный статистический метод, который будет возвращать данный объект.



Получим следующий код (рисунок 14).

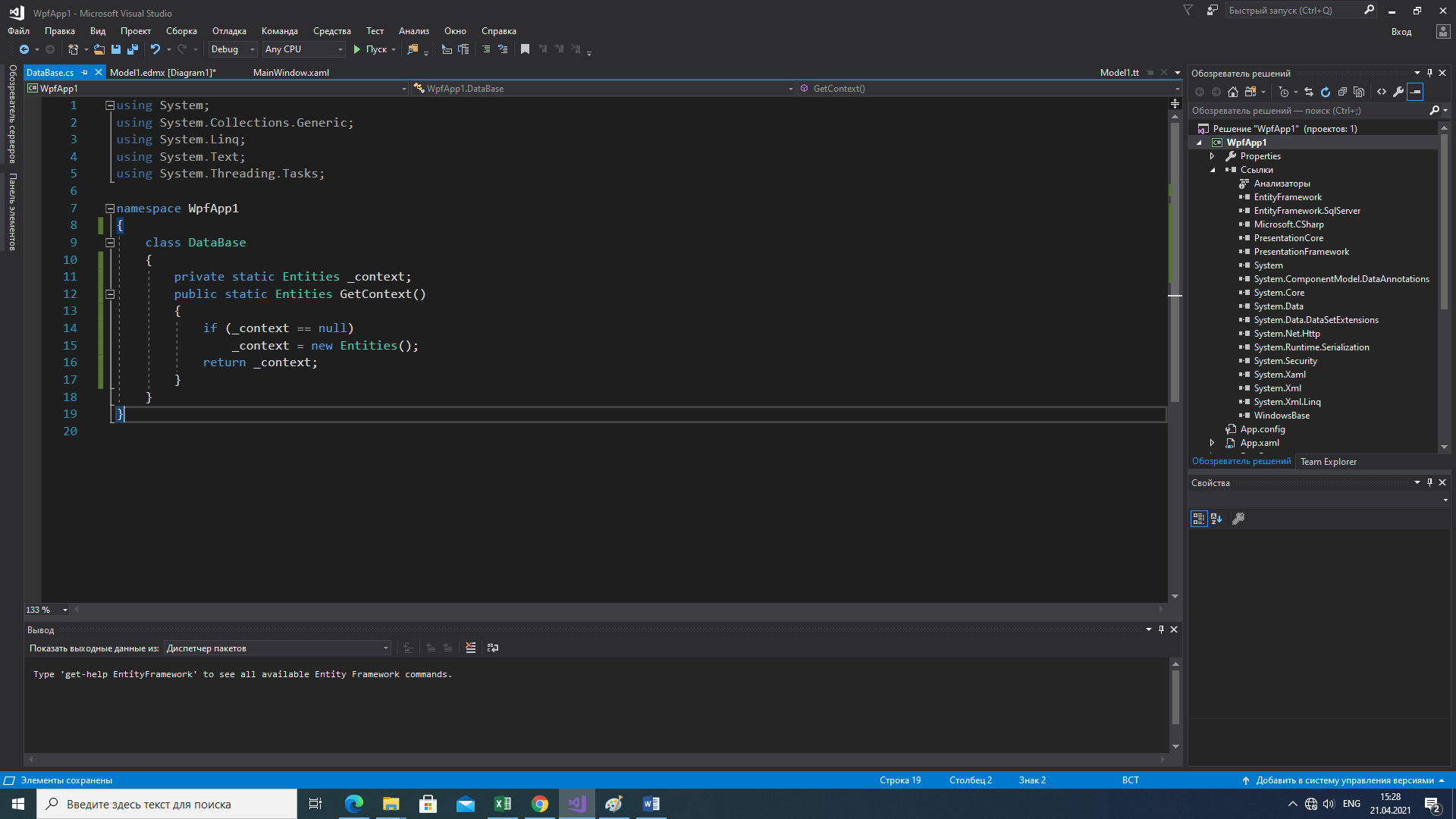


Рисунок 15. Публичный статичный метод

**Контрольные вопросы**

1. Как создать подключение к базе данных?

Для того чтобы создать подключение базы данных нужно в обозревателе решений создать новый элемент «Модель ADO.NET EDM». Задать название, после чего настроить соединение где мы указываем сервер, пароль. После чего выбираем таблицы для выгрузки и нажимаем готово .

**Практическая работа №1.3**

**Тема**

Создание руководства по стилю, логотипа, иконки. Размещение файлов в репозитории.

**Цель работы**

Разместить скрипт с базой данных, руководство по стилю, логотип и иконку в удаленный репозиторий проекта.

**Теоретический материал**

Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.

Git — одна из распределенных систем контроля версий.

GitHub — один из сервисов для использования системы контроля версий Git.

Git — одна из систем контроля версий.

Git предназначена, в основном, для работы распределенной команды разработчиков.

То есть разработчики могут находиться в разных концах света и работать над одним проектом.

Система Git очень экономична и не требует рассылки большого количества файлов. Отслеживаются и пересылаются изменения в файлах и ссылки на эти изменения. То есть основная рассылка — это рассылка разницы в ваших редактированиях.

Отсылаются только различия в папках и файлах. В любой момент времени вы можете возвратиться к тому или иному состоянию системы. Многие компании уделяют внимание хорошей и быстрой коммуникации между сотрудниками. В этом отношении, система контроля версий предоставляет большие возможности. Всю мощь и гибкость системы управления версиями вы сможете ощутить после изучения некоторого теоретического материала и применения на практике.

Основные обозначения:

repository — некоторое хранилище файлов, ссылок на изменения в файлах

commit — отслеживание изменений, сохраняет разницу в изменениях

working directory — рабочий каталог на вашем компьютере

staging area — область подготовленных файлов или рабочая область

branch — ветка, состоит из набора коммитов, обычно ссылается на последний коммит

merge — слияние, слияние веток в одну

pull — втянуть, взять проект с сервера, получить изменения из удаленного репозитория

push — вытолкнуть, отправить изменения на сервер