ASP.NET

Методические указания по созданию БД SQL в среде Visual Studio

Екатеринбург

2021 г

Оглавление

Основные сведения

Веб технология ASP.NET (Active Server Pages) это платформа для создания интерактивных веб приложений, от простого сайта визитки до огромных порталов в полном соответствии с веб стандартами. Она поддерживает работу с несколькими языками программирования, входящими в сборку фреймворка: Basic NET, C#, J# и ряд прочих.

Для создания веб-приложений на платформе ASP.NET MVC необходима среда разработки - Visual Studio.

Microsoft Visual Studio Community - бесплатная, полнофункциональная и расширяемая интегрированная среда разработки для создания современных приложений для Windows, Android и iOS, а также веб-приложений и облачных служб. ПО Visual Studio Community создана для индивидуальных разработчиков, проектов с открытым кодом, научных исследований, образования и небольших групп специалистов. Программирование на C#, Visual Basic, F#, C++, HTML, JavaScript, Python и других языках.

Visual Studio Code – бесплатный редактор кода, построен на открытом исходном коде, поддерживает множество языков программирования, C#, Razor, PHP, HTML, XML, CSS, JavaScript, Sass, Python, Perl, F#, C++ др. Отладка кода прямо из редактора, с помощью точек останова, стеков вызовов и интерактивной консоли. Visual Studio Code - расширяемый и настраиваемый. Для увеличение возможностей редактора устанавливаются расширения для добавления новых языков, тем, отладчиков и для подключения к дополнительным сервисам. Расширения выполняются в отдельных процессах, поэтому они не замедляют работу вашего редактора. Visual Studio Code работает в Windows, Linux, Mac.

Microsoft Visual Studio 2019 Professional - коммерческая среда разработки. Для индивидуальных разработчиков и предприятий. Имеет расширенные возможности отладки, диагностики, тестирования и кроссплатформенной разработки.

Используемый в среде Microsoft Visual C# язык программирования

C# – объектно-ориентированный язык программирования. Разработан в 1998-2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft.NET Framework и впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников – языков C++, Java, Delphi, Модула и Smalltalk – C#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# не поддерживает множественное наследование классов (в отличие от C++).

Установка Microsoft Visual Studio

Скачайте файл начального загрузчика Visual Studio https://www.visualstudio.com/ru/vs/community/.

Далее запустите файл начального загрузчика, чтобы установить Visual Studio Installer. Новый установщик имеет меньший размер и включает все необходимое для установки и настройки Visual Studio 2017.

Примите условия лицензии и заявление о конфиденциальности корпорации Майкрософт и нажмите кнопку Продолжить.

Когда завершится установка программы установки, Вы можете с ее помощью выбрать нужные наборы функций (рабочих нагрузок).

Указания к выполнению лабораторных работ

Лабораторные работы выполняются индивидуально. При этом каждая очередная Лабораторная работа является продолжением выполненной ранее и поэтому они должны обязательно выполняться последовательно.

Выполнив лабораторные работы, обучающийся оформляет отчет. (использовать MS Word). По каждой отдельной работе в отчете должен быть отдельный раздел.

Отчет должен содержать: название лабораторной работы; цель работы, условия задания; результаты выполнения, скриншоты проделанной работы. Цели формулируются обучающимися самостоятельно исходя из задания к лабораторной работе.

При оформлении отчета соблюдать следующие требования:

1. Отчет должен содержать титульный лист.
2. Поля страницы – обычные.
3. Цвет шрифта – черный.
4. Основной тип шрифта - Times New Roman, размер 14.
5. Межстрочный интервал – полуторный.
6. Выравнивание абзаца – по ширине.
7. Отступ первой строки – 1,25.
8. Нумерация страниц – сквозная на весь отчет, внизу, по центру. На титульной странице номер не отображается (но учитывается).
9. Разделы должны иметь заголовки и быть пронумерованы.
10. На все иллюстрации должна быть ссылка в тексте (…представлен на рис. 1 …..). Подрисуночный текст выравнен по центру.
11. Оглавление в отчете должно быть автоматическим, выполненное средствами MS Word.
12. Повторяющие пробелы для форматирования текста не допускаются.

Что такое веб-формы?

Веб-форма является одной из моделей 3 программирования для создания ASP.NET веб-сайтов и веб-приложений.

Две другие модели программирования веб-страницы и MVC (Model, View, Controller).

Web Forms является самой старой моделью программирования ASP.NET, с управляемыми событиями, веб-страницы написаны в виде комбинации HTML, серверные элементов управления и серверного кода.

Веб-формы составляются и выполняются на сервере, который генерирует HTML, который отображает веб-страницу.

Web Forms приходит с сотнями различных веб-элементы управления и веб-компонентов для создания пользовательских веб-сайтов с доступом к данным.

ASP.NET Web Forms позволяет даже новичку быстро создать простой сайт.

Элементы управления на форме называется объектами. Каждый объект обладает своим набором свойств, событий и методов.

Свойства объекта – это его характеристики (высота, ширина и т.д.).

События объекта – это события операционных систем или события, инициируемые пользователем, на которые может реагировать объект (нажатие кнопки).

Методы объекта – действия, которые можно производить с объектом в ходе выполнения программ.

В БД все объекты форм делятся на два класса:

Объекты управления – объекты, осуществляющие управление БД (Например, Кнопка или Выпадающий список).

Объекты для отображения информации – элементы, отображающие содержимое таблиц, запросов или фильтров, позволяющие добавлять изменять и удалять информацию, и проводить ее анализ.

Все формы в клиентском приложении делятся на три группы:

1. Формы для работы с данными – формы, содержащие как объекты управления, так и объекты просмотра данных. Такие формы предназначены для отображения, изменения, удаления и анализа данных.

2. Кнопочные формы – формы, содержащие только объекты управления, предназначаются для открытия всех других форм. Кнопочная форма, которая появляется первой после запуска программы, называется, главной кнопочной формой.

3. Информационные и служебные формы – формы, содержащие только элементы управления, предназначены для отображения служебной информации (справки), несвязанной с таблицами, запросами и фильтрами, либо для выполнения служебных операций, не связанных с данными (Например, форма с калькулятором).

Методы формы

Методы применяются для исполнения тех или иных действий. Методы, являющиеся членами класса, выполняют действия, составляющие функциональность данного класса. Любая форма инкапсулирует базовый набор функций, унаследованный от класса `System.Windows.Forms.Form`, куда входят методы, управляющие отображением формы и доступом к ней в пользовательском окружении. Вот некоторые методы:

`Show()` – загружает экземпляр класса формы в память, отображает его на экране и передает ему фокус ввода, при этом свойство `Visible` автоматически устанавливается в `true`. Если экземпляр формы уже загружен, но пока не виден (например, если его свойство `Visible` установлено в `false`), вызов метода `Show()` даст тот же результат, что и установка свойства `Visible` в `true`).

`ShowDialog()` – выполняет те же действия, что и `Show()`, но делает окно формы модальным. Это означает, что другим формам приложения не удастся получить фокус, пока не закрыта форма, показанная при помощи

метода ShowDialog(). Сделав окно формы модальным, вы заставите пользователя выполнить некоторые действия на этой форме, и только после этого он сможет продолжить работу с приложением.

Activate() – активирует форму, и она получает фокус ввода. Если вызвать этот метод в окне приложения, у которого в текущий момент нет активных форм, окно этого приложения на панели задач начнет мигать. В любом случае активировать разрешено только видимую форму.

Hide() – делает форму невидимой. Форма остается в памяти, но остается невидимой, пока не будет вызван метод Show() или свойство Visible этой формы не будет установлено в true. Метод Hide() устанавливает свойство Visible в false.

Close() – закрывает форму. Этот метод закрывает все удерживаемые формой ресурсы. После вызова метода Close() сделать форму видимой, вызвав метод Show(), не удастся, поскольку ресурсы формы уже освобождены. Вызов Close() на стартовой форме приложения завершает приложение.

Для вызова любого из этих методов необходима ссылка на форму, т. е в памяти должен быть заранее созданный экземпляр формы. При запуске приложения автоматически создает экземпляр стартовой формы в дополнение к тем экземплярам форм, которые создаются в код

Существует два вида дизайна форм:

1. Ленточные формы - формы, выводящие информацию по одной записи.

2. Табличные формы – формы, выводящие информацию в виде таблицы.

Объекты связи используются только в клиентском интерфейсе. В web-интерфейса функции объекта связи выполняет сервер.

Лабораторная работа 1

1. Запустите программу. Выберите - Создать проект. В окне Выбор проекта выберите Приложение Windows Forms (Visual C#), присвойте проекту имя – Progect3_Ivanov_21 (рис. 1).

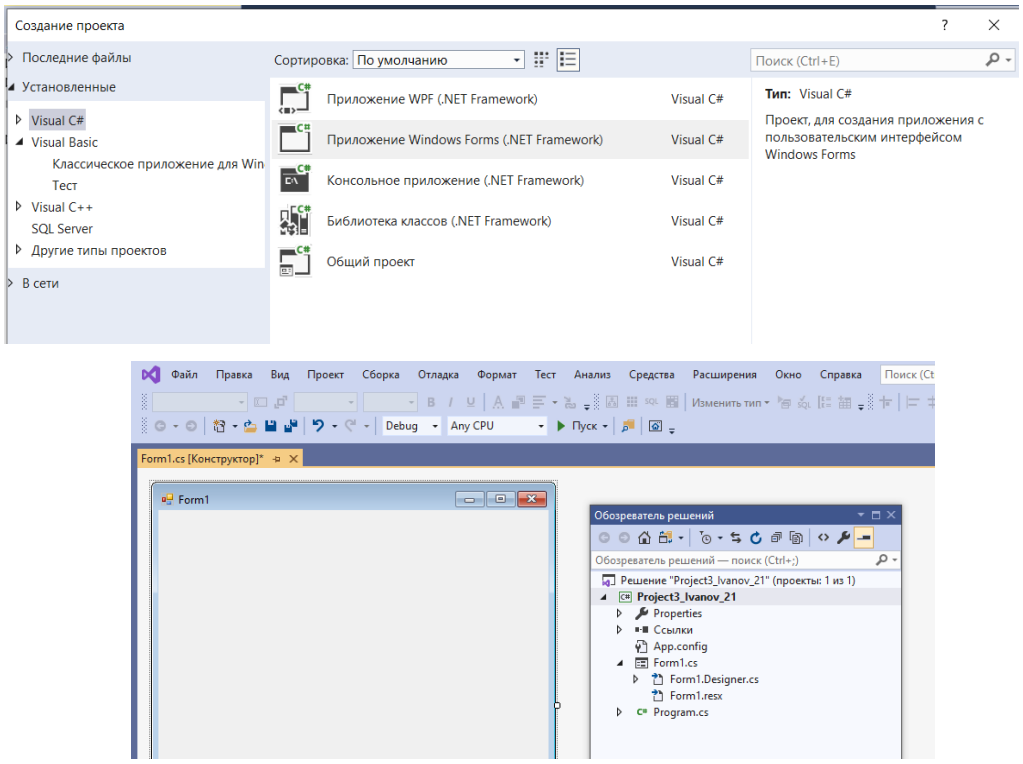


Рис. 1. Создание приложения Windows Forms

2. В контекстном меню имени проекта (Обозреватель решений) выберите Добавить Создать элемент, рис. 2.

3. В списке шаблонов элементов выберите база данных, основаная на службах, присвойте имя MyBD, рис. 3.

4. БД появится в обозревателе решений, рис. 4.

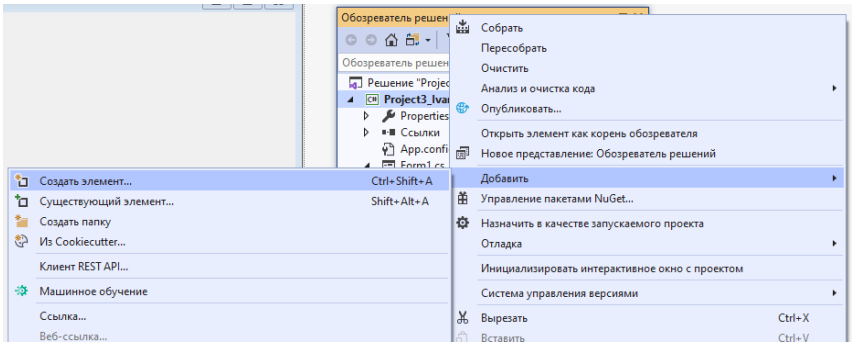


Рис. 2. Добавление элемента в проект

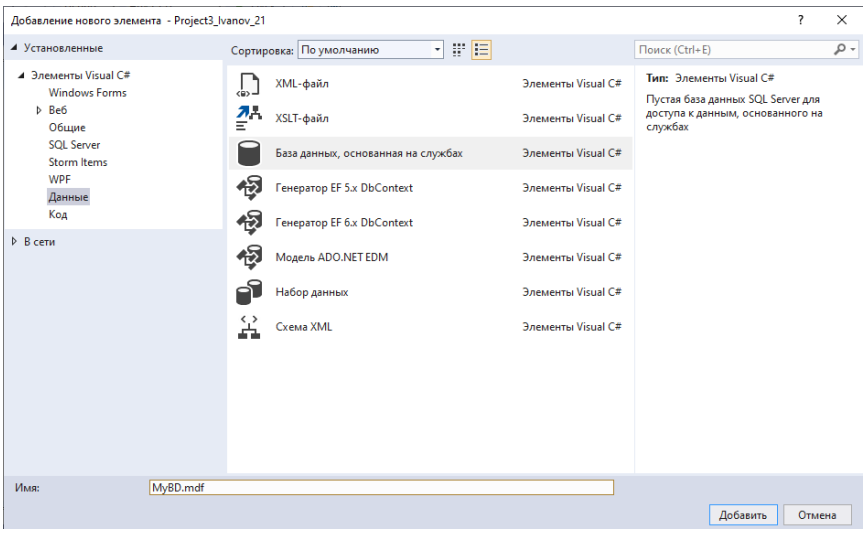


Рис. 3. Создание БД

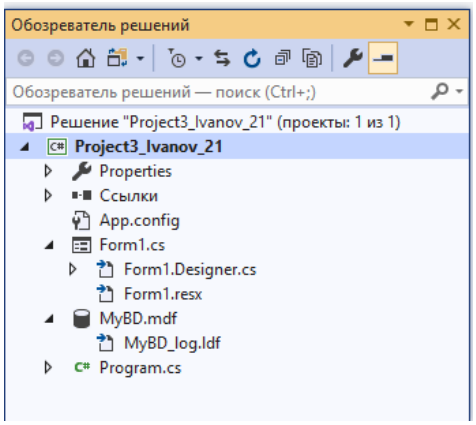


Рис. 4. Отображение БД в обозревателе решений

Добавление источника данных

5. В окне Источники данных (Вид-Другие окна), рис. 5, выберите Добавить новый источник данных, рис. 6.

6. Выбрать На странице Выбор модели базы данных нажмите кнопку Далее , чтобы принять значение по умолчанию (набор данных), рис. 7.

7. На странице сохранить строку подключения в файле конфигурации приложения нажмите кнопку Далее, рис. 8.

8. На странице Выбор объектов базы данных появится сообщение о том, что база данных не содержит объектов. Нажмите кнопку Готово, рис.9.

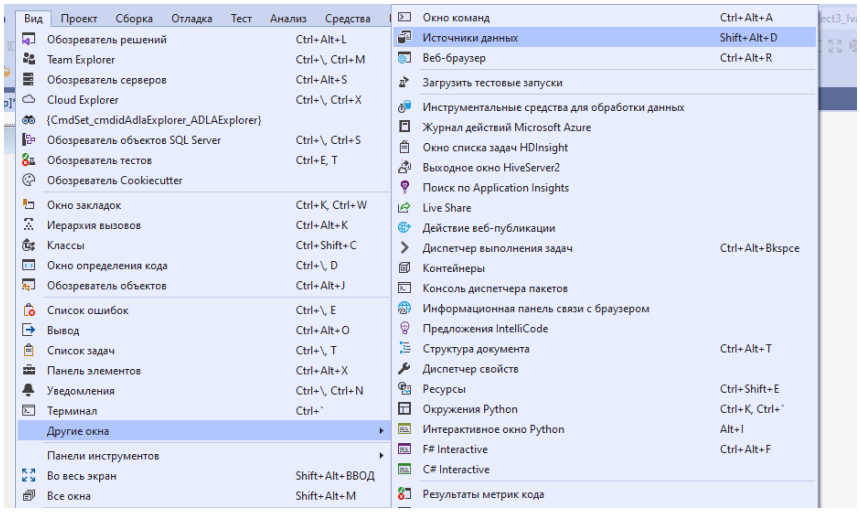


Рис. 5. Выбор команды Источники данных

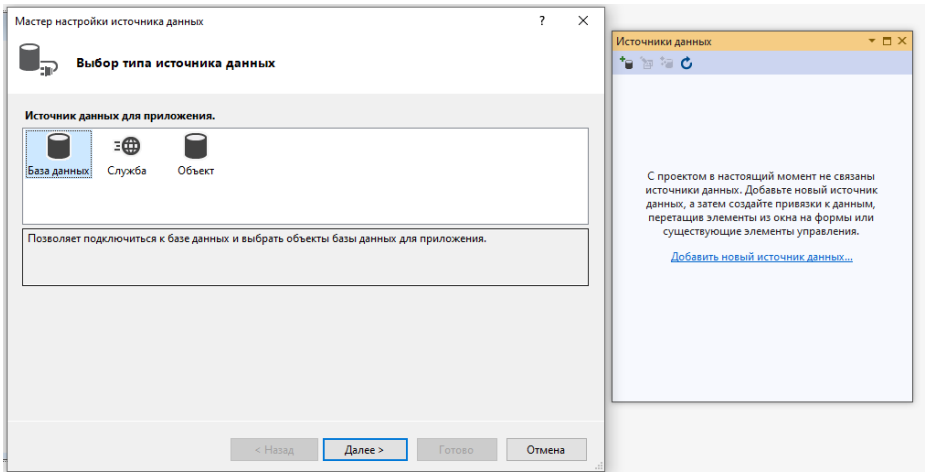


Рис. 6. Добавление нового источника данных

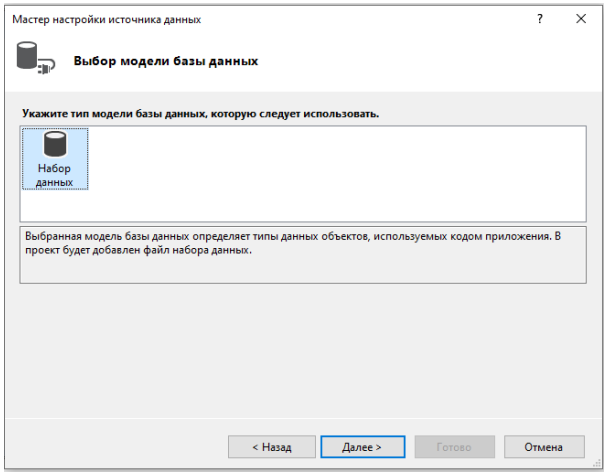


Рис. 7. Выбор модели БД

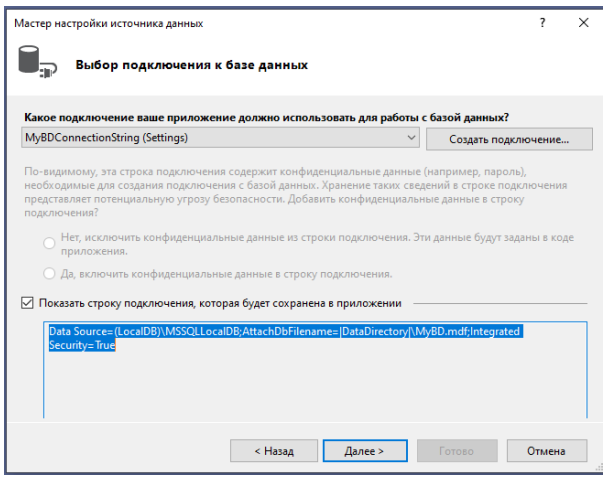


Рис. 8. Выбор подключения

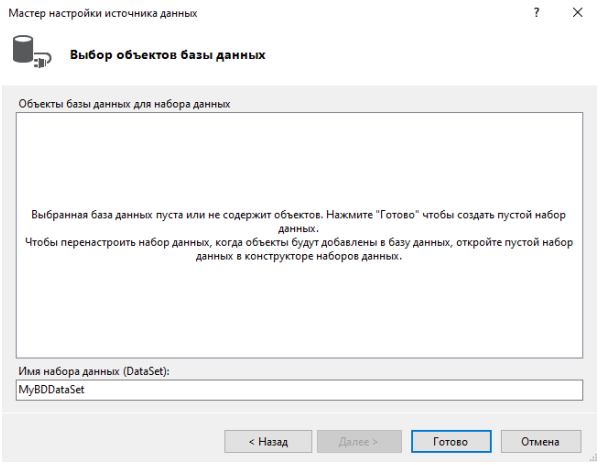


Рис. 9. Создание пустого набора данных

Лабораторная работа 2

Создание таблиц и ключей с помощью конструктор таблиц

1. В Обозреватель сервера (Вид – Обозреватель серверов) разверните узел подключения к данным, а затем узел MyBD. mdf.

2. Если вы не можете развернуть узел подключения к данным или подключение MyBD.mdf отсутствует в списке, нажмите кнопку подключиться к базе данных на панели инструментов Обозреватель сервера. В диалоговом окне Добавление соединения убедитесь, что в поле источник данныхвыбран Microsoft SQL Server файл базы данных, а затем найдите и выберите файл MyBD. mdf. Завершите добавление подключения, нажав кнопку ОК.

3. Щелкните правой кнопкой мыши таблицы и выберите команду Добавить новую таблицу, рис.10.

4. Будет открыт Конструктор таблиц, отобразится сетка с одной строкой по умолчанию, которая представляет один столбец в создаваемой таблице. Путем добавления строк в сетку будут добавлены столбцы в таблицу.

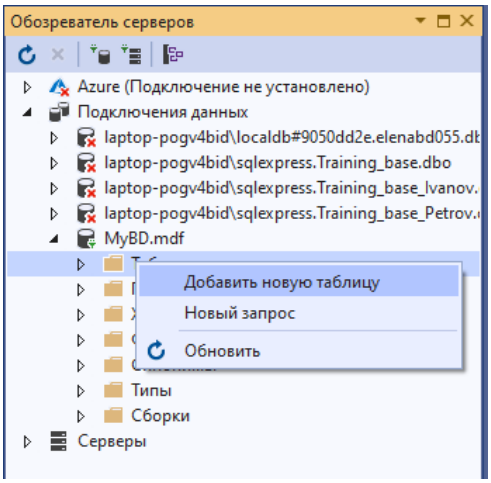


Рис. 10. Добавление новой таблицы

Конструктор таблицы имеет следующие столбцы:

— `Column Name` (Имя столбца) — имя столбца должно всегда начинаться с буквы и не должно содержать различных специальных символов и знаков препинания. Если имя поля содержит пробелы, то оно автоматически заключается в квадратные скобки.

— `Data Type` (Тип данных) — тип данных поля.

— `Allow Nulls` (Разрешить значения `Null`) — допуск значения `Null`. Если эта опция поля включена, то в случае незаполнения поля в него будет автоматически подставлено значение `Null`. То есть поле необязательно для заполнения.

Внизу автоматически пишется SQL код.

В таблице определения полей задайте значения столбцов `Column Name` («Имя столбца»), `Date Type` (Тип данных) и `Allow Nulls` (Разрешить значения `Null`), как показано на рис. 11.

Из рис. 11 следует, что таблица `Salespeople` имеет четыре поля:

— `SNUM` — уникальный номер (ключевое поле), приписанный каждому продавцу (номер служащего), числовой, целое. Чтобы сделать поле

ключевым, выделите поле, а затем на панели инструментов нажмите кнопку с изображением ключа (или в контекстном меню поля). В таблице определения полей, рядом с полем SNUM появиться изображение ключа, говорящее о том, что поле ключевое;

— SNAME — имя продавца, текстовое поле, предназначенное для хранения строк, имеющих длину не более 10 символов;

— CITY — Место расположения продавца, текстовое поле, предназначенное для хранения строк, имеющих длину не более 10 символов.

— COMM – вознаграждение (комиссионные) продавца, числовое поле с плавающей точкой (decimal (6,2) означает, 6 - максимальное общее число хранимых десятичных разрядов, включая символы слева и справа от десятичной запятой, 2 - число хранимых десятичных разрядов справа от десятичной запятой).

Сохраните таблицу под именем Salespeople, для этого в первой строке (CREATE TABLE [dbo].[имя таблицы]) укажите Salespeople. В левом верхнем углу конструктора таблиц. **Нажмите Обновить!**

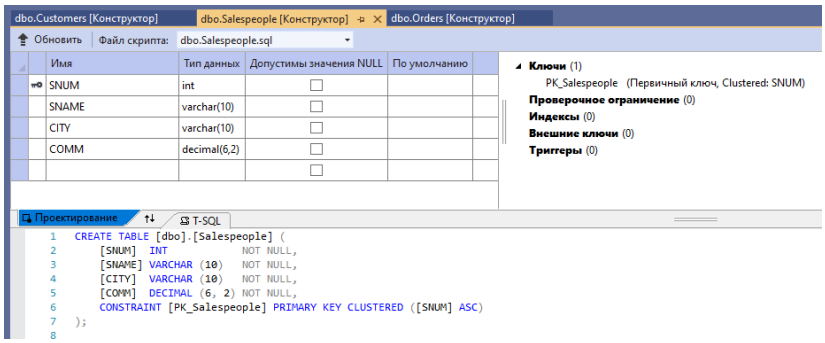


Рис. 11. Создание таблицы Salespeople

Так же обновите папку Таблицы в обозревателе серверов.

5. Создайте вторую таблицу- Customers (Покупатели) с помощью конструктора.

6. Добавьте в нее поля: (в соответствии с рис. 12).

— CNUM – уникальный номер, присвоенный покупателю;

— CNAME – имя покупателя;

— CITY – место расположения покупателя;

— RATING – цифровой код, определяющий уровень предпочтения данного покупателя. Чем больше, тем больше предпочтение.

SNUM – номер продавца, назначенного данному покупателю (из таблицы Salespeople).

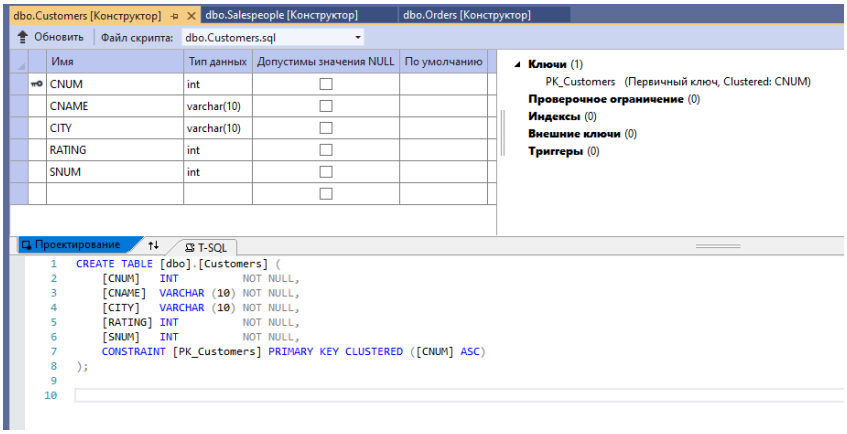


Рис. 12. Создание таблицы Customers

7. Создайте таблицу Orders (Заказы), рис. 13. Включите в нее следующие поля:

— ONUM – уникальный номер, присвоенный данной покупке;

— AMT – количество (сумма заказа);

— ODATE – дата заказа (покупки);

— CNUM – номер покупателя, сделавшего покупку (из таблицы (Customers);

— SNUM – номер продавца, обслуживающего покупателя (из таблицы Salespeople).

Поле ONUM (уникальный номер, присвоенный данной покупке) является первичным ключом, в данном примере будет числовым **счётчиком** (добавить для поля [ONUM] INT IDENTITY (1, 1)). То есть данное поле должно автоматически заполняться числовыми значениями Эти настройки

показывают, что значение поля `ONUM` у первой записи в таблице будет равным 1, у второй — 2, у третьей — 3 и т.д. Более того, оно должно быть **ключевым**.

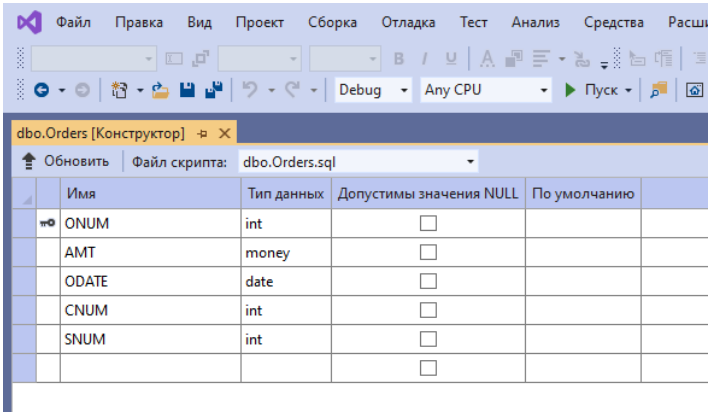


Рис.13. Таблица `Orders`

Лабораторная работа 3

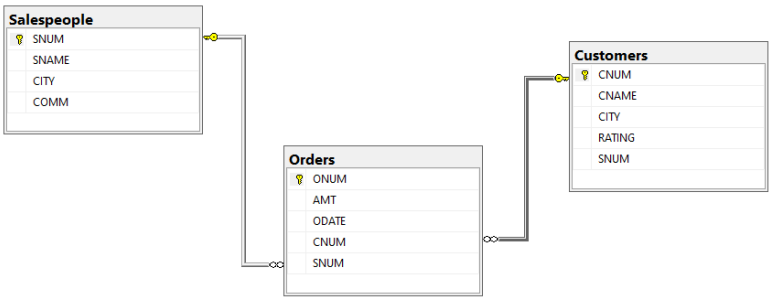
Обеспечение целостности данных

Целостность базы данных (database integrity) — соответствие имеющейся в базе данных информации её внутренней логике, структуре и всем явно заданным правилам. Каждое правило, налагающее некоторое ограничение на возможное состояние базы данных, называется **ограничением целостности** (integrity constraint).

Целостность базы данных - свойство базы данных, означающее, что БД содержит полную и непротиворечивую информацию, необходимую для корректного функционирования приложений.

Внешний ключ – это столбец (или комбинация столбцов), совпадающий с первичным ключом некоторой таблицы. Если значение внешнего ключа соответствует значению первичного ключа, становится понятно, что между объектами базы данных представленными совпадающими строками, существует логическое взаимоотношение. Одним из главных ограничений отношения является ссылочная целостность, которая определяет, что каждое значение внешнего ключа, не являющееся NULL-значением, должно ссылаться (`REFERENCES`) на некоторое существующее значение первичного ключа.

Для рассматриваемого примера необходимо осуществить создание связей согласно схеме на рис. 14.



а

Рис. 14. Схема БД

Создание внешних ключей

1. В контекстной области в правой части сетки конструктор таблиц для таблицы Orders щелкните правой кнопкой мыши внешние ключи и выберите Добавить новый внешний ключ, рис. 15.

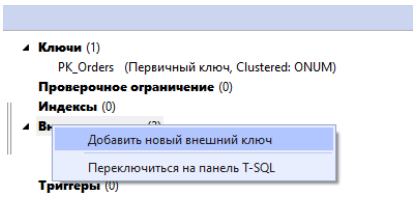


Рис. 15. Добавление внешнего ключа

2. В появившемся текстовом поле замените текст **ToTable** на Salespeople.

3. На панели T-SQL обновите последнюю строку, чтобы она соответствовала следующему примеру (рис.16):

```
8    CONSTRAINT [FK_Orders_Salespeople] FOREIGN KEY ([SNUM]) REFERENCES [dbo].[Salespeople] ([SNUM]),
```

Рис. 16. Определение внешнего ключа

4. Добавьте еще один внешний ключ для связи со второй таблицей.

5. Должно получиться, как на рис. 17.

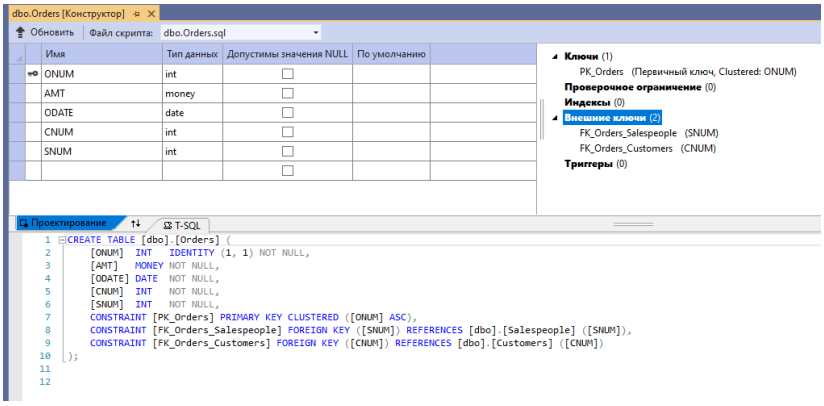


Рис. 17. Создание таблицы Orders

Заполнения таблиц начальными данными

Для начала заполните таблицу `Salespeople`. Для заполнения этой таблицы в обозревателе объектов в контекстном меню таблицы выбрать Показать таблицу данных, рис. 18.

1. Заполните таблицу `Salespeople`, как показано на рис.19.
2. Заполним таблицу `Customers` (рис.20).
3. Заполним таблицу `Orders` (рис.21). Так как поле `ONUM` является первичным полем связи и **ключевым числовым счётчиком**, то оно заполняется автоматически (заполнять его не нужно).

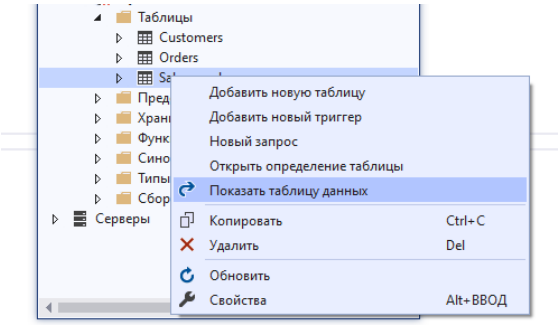


Рис. 18. Отобразить таблицу с данными

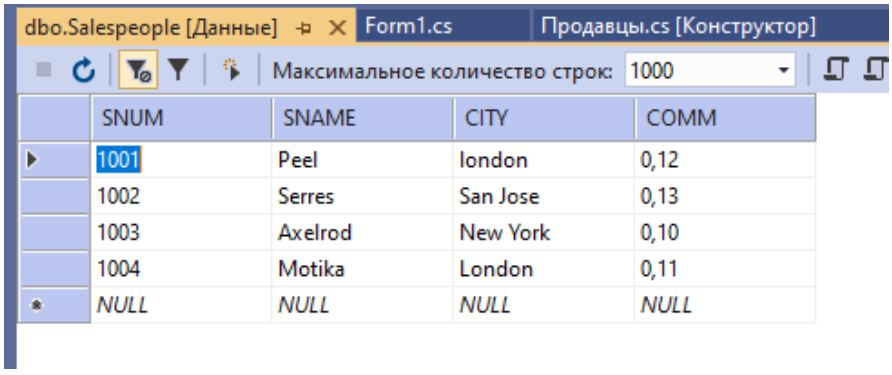
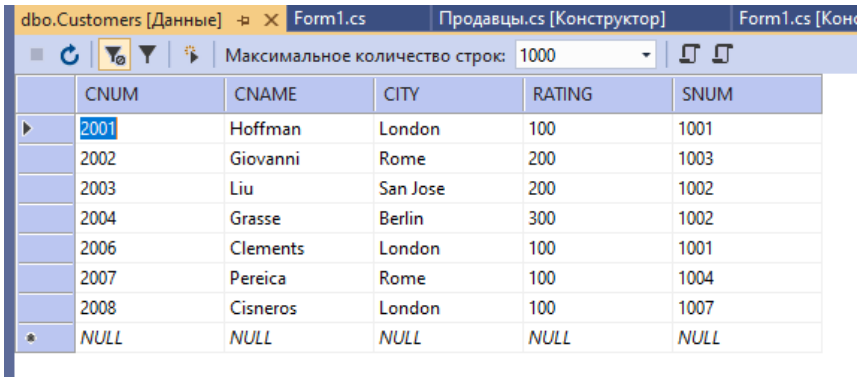
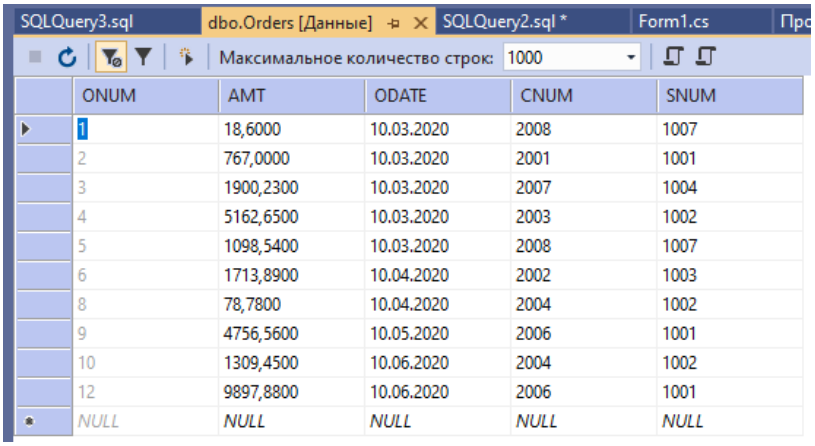


Рис. 19. Заполнение таблицы `Salespeople`



CNUM	CNAME	CITY	RATING	SNUM
2001	Hoffman	London	100	1001
2002	Giovanni	Rome	200	1003
2003	Liu	San Jose	200	1002
2004	Grasse	Berlin	300	1002
2006	Clements	London	100	1001
2007	Pereica	Rome	100	1004
2008	Cisneros	London	100	1007
NULL	NULL	NULL	NULL	NULL

Рис. 20. Заполнение таблицы `Customers`



ONUM	AMT	ODATE	CNUM	SNUM
1	18,6000	10.03.2020	2008	1007
2	767,0000	10.03.2020	2001	1001
3	1900,2300	10.03.2020	2007	1004
4	5162,6500	10.03.2020	2003	1002
5	1098,5400	10.03.2020	2008	1007
6	1713,8900	10.04.2020	2002	1003
8	78,7800	10.04.2020	2004	1002
9	4756,5600	10.05.2020	2006	1001
10	1309,4500	10.06.2020	2004	1002
12	9897,8800	10.06.2020	2006	1001
NULL	NULL	NULL	NULL	NULL

Рис. 21. Заполнение таблицы `Orders`