



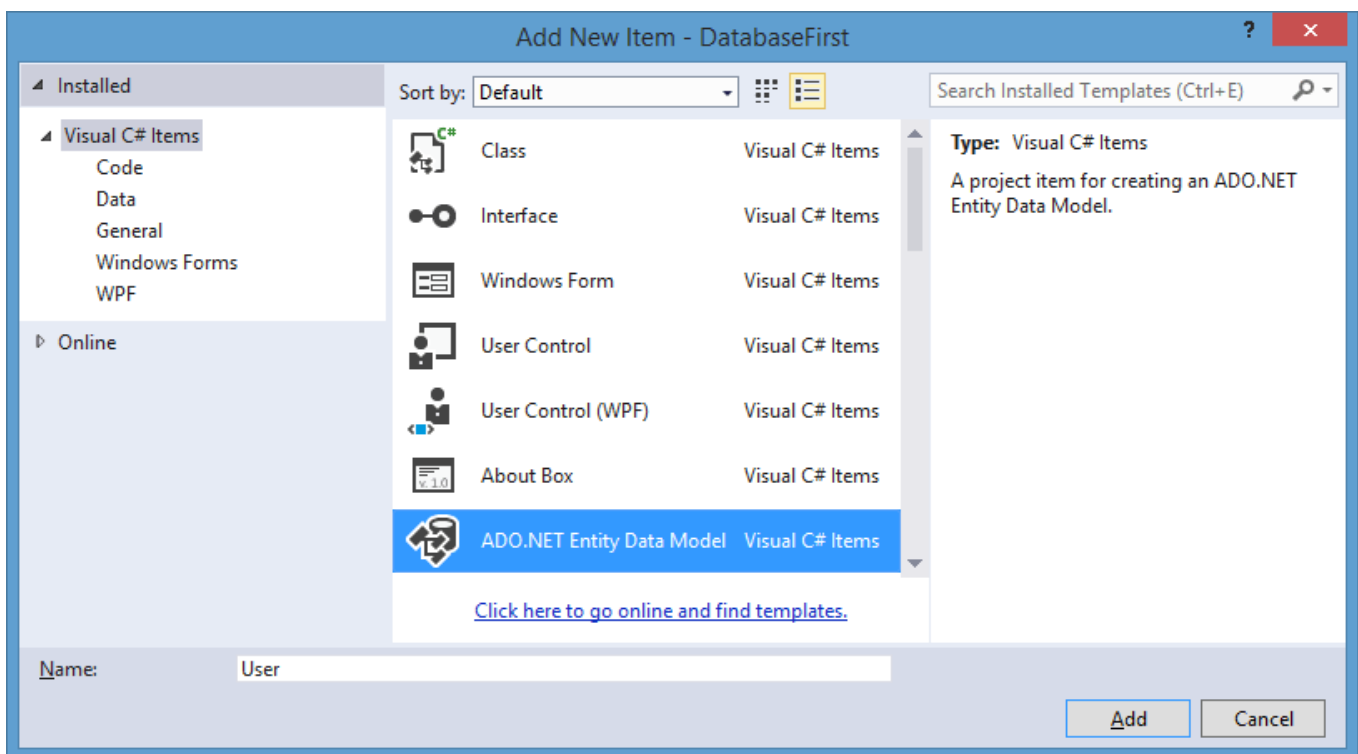
Model First

Последнее обновление: 31.10.2015

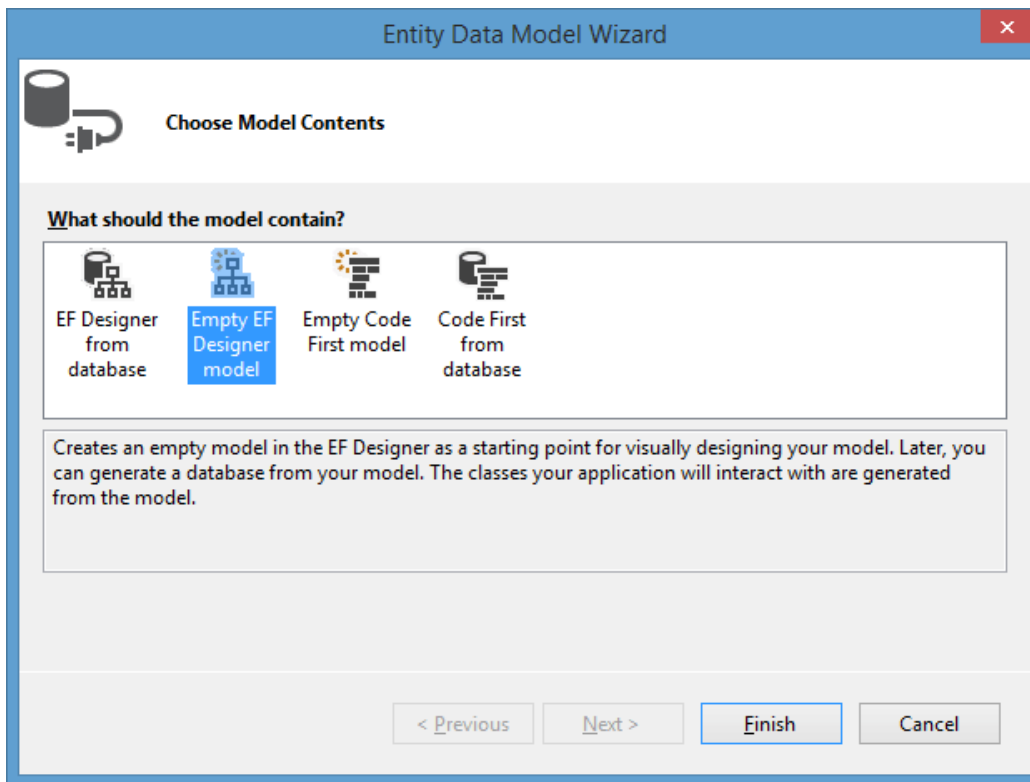


Model First представляет еще один подход к работе с Entity Framework. Суть данного подхода состоит в том, что сначала делается модель, а потом по ней создается база данных.

Итак, создадим новый проект по типу Console Application. И затем добавим в проект новый элемент. Нажмем правой кнопкой мыши на проект в окне Solution Explorer и в появившемся списке выберем **Add -> New Item**. И затем в окне добавления нового элемента выберем **ADO.NET Entity Data Model**:



Поскольку модель будет описывать человека, то назовем ее User. Нажмем ОК и нам откроется мастер создания модели. Если у нас Visual Studio с пакетами обновления SP2, SP3, то мастер создания модели выглядит следующим образом:



Окно нам предлагает четыре варианта создания модели, из которых нам надо выбрать **Empty EF Designer Model**.

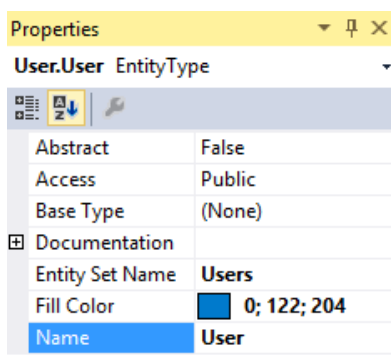
Нажмем кнопку Finish, и перед нами откроется пустое окно создания модели.

The Entity Data Model Designer lets you visualize and design your Entity Data Model.

Create new entities in the model by dragging items from the [Toolbox](#).

Add existing entities and relationships to this diagram by dragging them from the [Model Browser](#).

Перетащим на это поле с панели Toolbox (Панель Инструментов) в левой части элемент Entity. Теперь у нас на поле создания модели имеется небольшая схема будущей модели, в которой сейчас по умолчанию указано лишь одно поле - Id. Во-первых, переименуем сущность. По умолчанию она называется Entity1. Выделим схему и перейдем к окну свойств в правом нижнем углу:



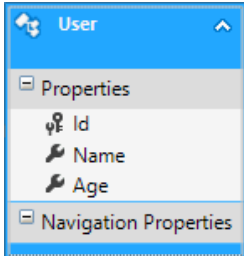
Name

The name of the entity.

Здесь изменим значение свойства Name на User. Это у нас будет имя сущности. И также изменим значение свойства Entity Set Name на Users. Это у нас будет название набора объектов User.

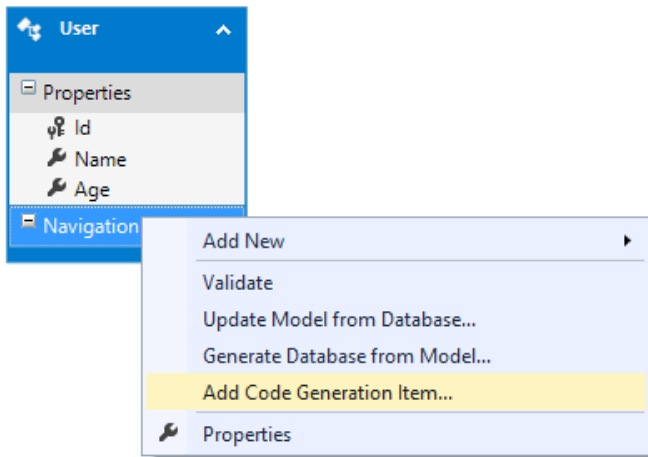
Далее создадим несколько свойств. Сущность у нас будет простая и будет содержать всего два свойства для имени и возраста. Итак, выделим схему сущности и нажмем на правую кнопку мыши. В выпадающем списке выберем **Add New -> Scalar Property**. После этого будет добавлено новое свойство. Scalar Property подразумевают свойства на основе простейших типов int, float, string и т.д. Добавим два свойства - Name и Age. По умолчанию все добавляемые свойства имеют тип string. Однако мы можем изменить тип в окне свойств.

Таким образом, у нас должна получиться следующая схема сущности:

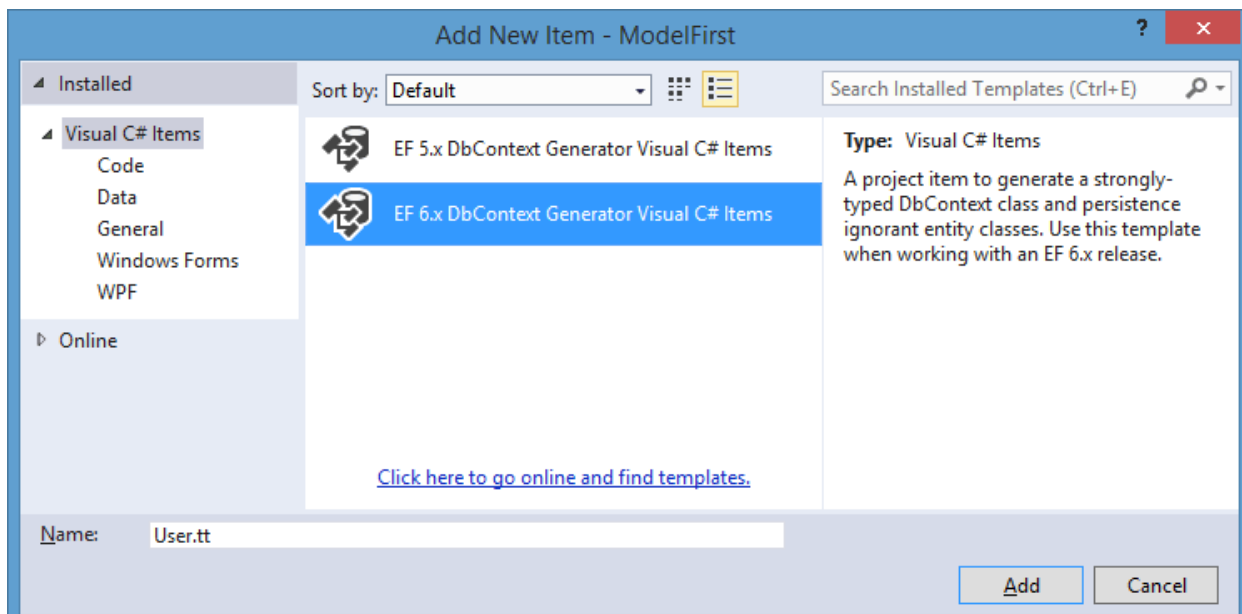


После создания диаграммы модели перестроим проект с помощью опции Rebuild.

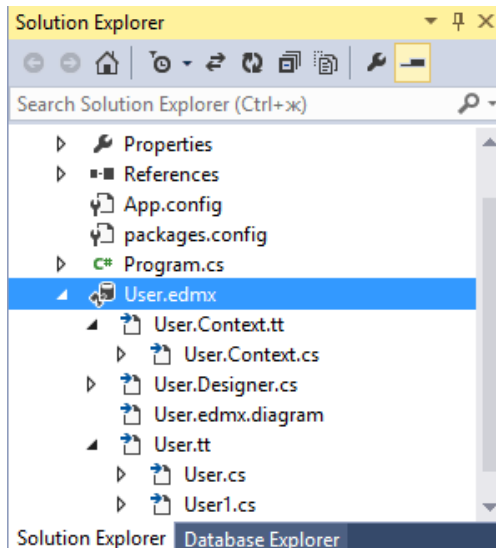
Теперь по модели мы можем сгенерировать код и базу данных. Вначале сгенерируем код модели. Для этого нажмем на диаграмму модели правой кнопкой мыши и выберем пункт **Add Code Generation Item**:



Далее нам будет предложено выбрать версию EF. Выберем шестую версию:



После этого в структуре проекта мы можем увидеть узел *User.tt*, который в качестве подузла будет содержать класс модели в файле *User.cs*:



Также здесь мы можем найти файл контекста данных *User.Context.cs*, который выглядит в моем случае следующим образом:

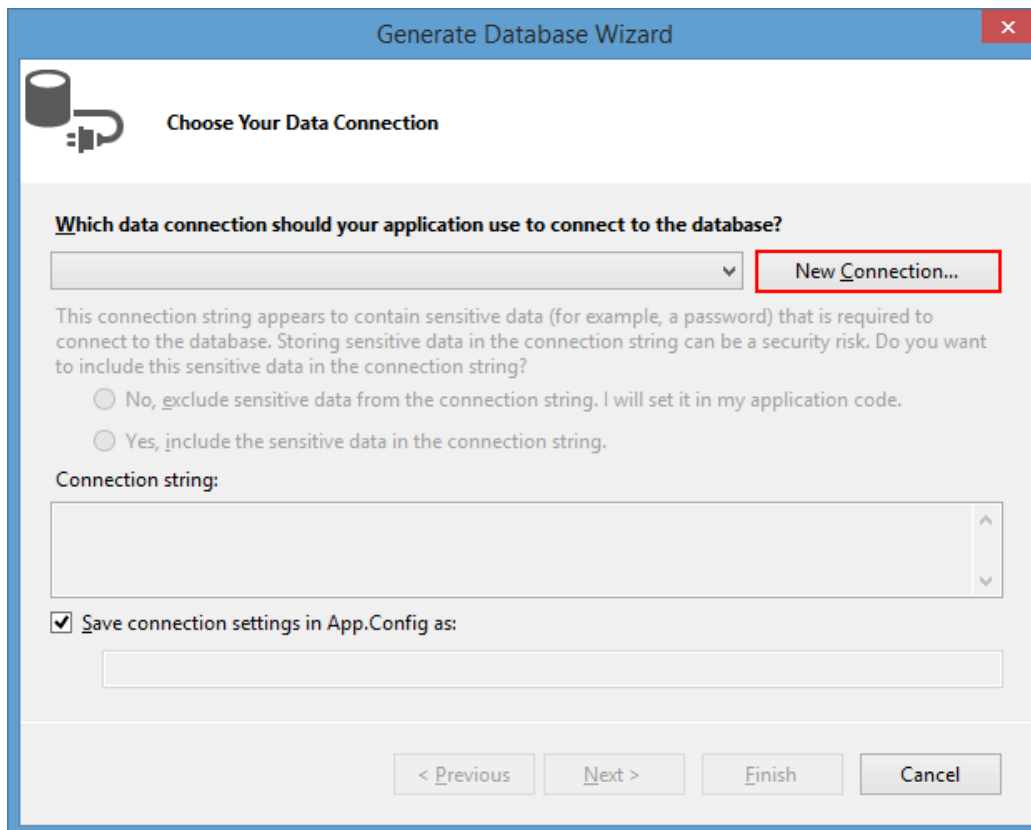
```
namespace ModelFirst
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Infrastructure;

    public partial class UserContainer : DbContext
    {
        public UserContainer()
            : base("name=UserContainer")
        {
        }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            throw new UnintentionalCodeFirstException();
        }

        public virtual DbSet<User> Users { get; set; }
    }
}
```

Теперь сгенерируем базу данных по нашей модели. Итак, нажмем на диаграмму модели правой кнопкой мыши и в выпадающем списке выберем **Generate Database from Model** (Сгенерировать базу данных по модели). Перед нами откроется мастер создания подключения.



The screenshot shows the 'Generate Database Wizard' dialog box with the title 'Generate Database Wizard' and a close button. The main heading is 'Choose Your Data Connection'. Below this, there is a question: 'Which data connection should your application use to connect to the database?'. A dropdown menu is shown with a red box around the 'New Connection...' option. Below the dropdown, there is a warning message: 'This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?'. There are two radio buttons: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below this, there is a text box labeled 'Connection string:'. At the bottom, there is a checkbox labeled 'Save connection settings in App.Config as:' with an empty text box next to it. At the very bottom, there are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

Нажмем на кнопку **New Connection** (Новое подключение). Далее открывается новое окно, где будет предложено настроить подключение и создать базу данных:

Connection Properties

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
(localdb)\v11.0 Refresh

Log on to the server

☒ Use Windows Authentication

☐ Use SQL Server Authentication

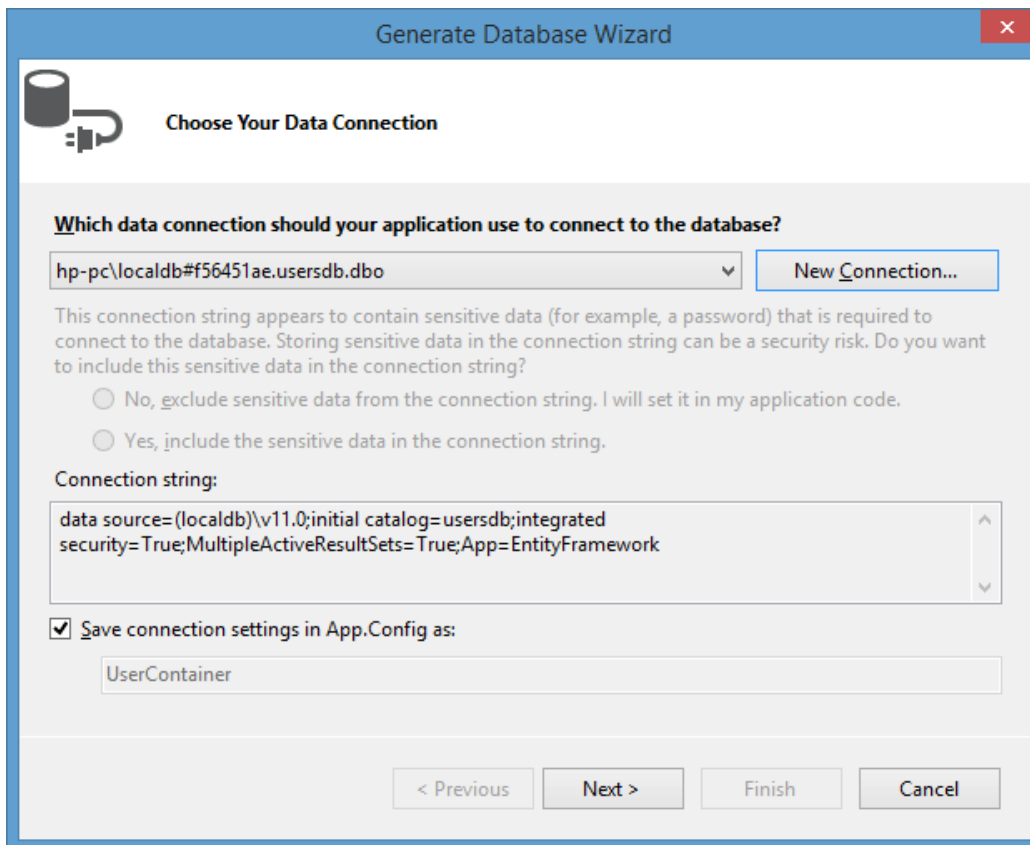
User name: Password:

☐ Save my password

Advanced...

Test Connection OK Cancel

Здесь нам надо ввести имя сервера. А также название бд. В качестве имени базы данных введем usersdb, а в качестве сервера: **(localdb)\v11.0**. Нажмем OK и затем Visual Studio установит в качестве подключения модели только что созданную базу данных:



Generate Database Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

hp-pc\localdb#f56451ae.usersdb.dbo New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Connection string:

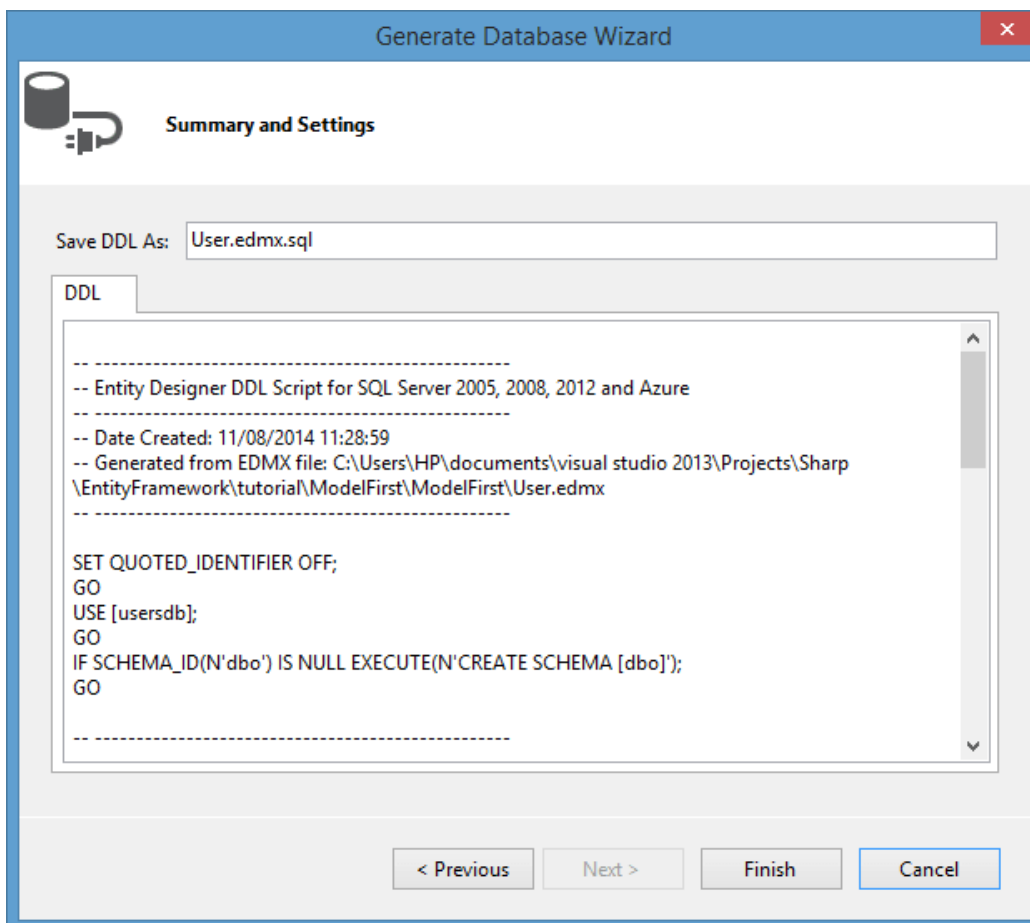
data source=(localdb)\v11.0;initial catalog=usersdb;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework

☒ Save connection settings in App.Config as:

UserContainer

< Previous Next > Finish Cancel

После этого будет сгенерирован скрипт базы данных:



Generate Database Wizard

Summary and Settings

Save DDL As: User.edmx.sql

DDL

```
-- Entity Designer DDL Script for SQL Server 2005, 2008, 2012 and Azure
-- Date Created: 11/08/2014 11:28:59
-- Generated from EDMX file: C:\Users\HP\documents\visual studio 2013\Projects\Sharp
\EntityFramework\tutorial\ModelFirst\ModelFirst\User.edmx

SET QUOTED_IDENTIFIER OFF;
GO
USE [usersdb];
GO
IF SCHEMA_ID(N'dbo') IS NULL EXECUTE(N'CREATE SCHEMA [dbo]');
GO
```

< Previous Next > Finish Cancel

Нажмем Finish (Готово). У нас автоматически откроется в Visual Studio файл скрипта User.edmx.sql. И в завершении нам надо будет запустить этот скрипт. Для этого нажмем в верхнем левом углу на зеленую кнопку Execute (Выполнить):

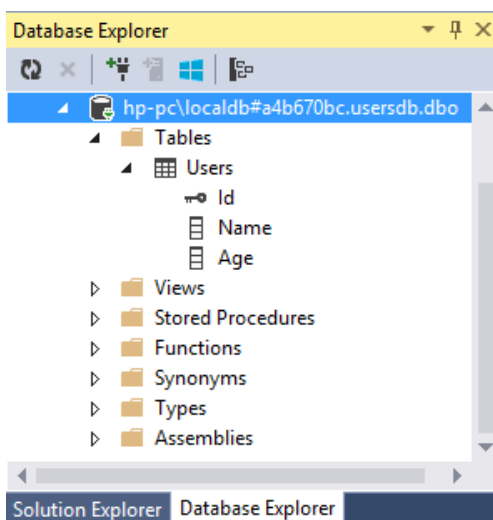
```

-- Entity Designer DDL Script for S
-- Date Created: 11/08/2014 11:28:5
-- Generated from EDMX file: C:\Use

SET QUOTED_IDENTIFIER OFF;
GO
USE [usersdb];
GO
IF SCHEMA_ID(N'dbo') IS NULL EXECUT
GO

```

После этого в нижнем окне Visual Studio нам сообщит об успешном (или неуспешном) создании базы данных. Если мы откроем окно Database Explorer (его можно открыть, выбрав в меню View->Other Windows), то мы увидим нашу базу данных. А раскрыв узел, также увидим, что она содержит всю ту схему, которую мы определили в модели:



Это все была работа по созданию модели и базы данных. И в конце определим минимальный код для работы с базой данных:

```

static void Main(string[] args)
{
    using(UserContainer db = new UserContainer())
    {
        // добавление элементов
        db.Users.Add(new User {Name="Tom", Age=45});
        db.Users.Add(new User {Name="John", Age=22});
        db.SaveChanges();
        // получение элементов
        var users = db.Users;
        foreach (User u in users)
            Console.WriteLine("{0}.{1} - {2}", u.Id, u.Name, u.Age);
    }
    Console.Read();
}

```