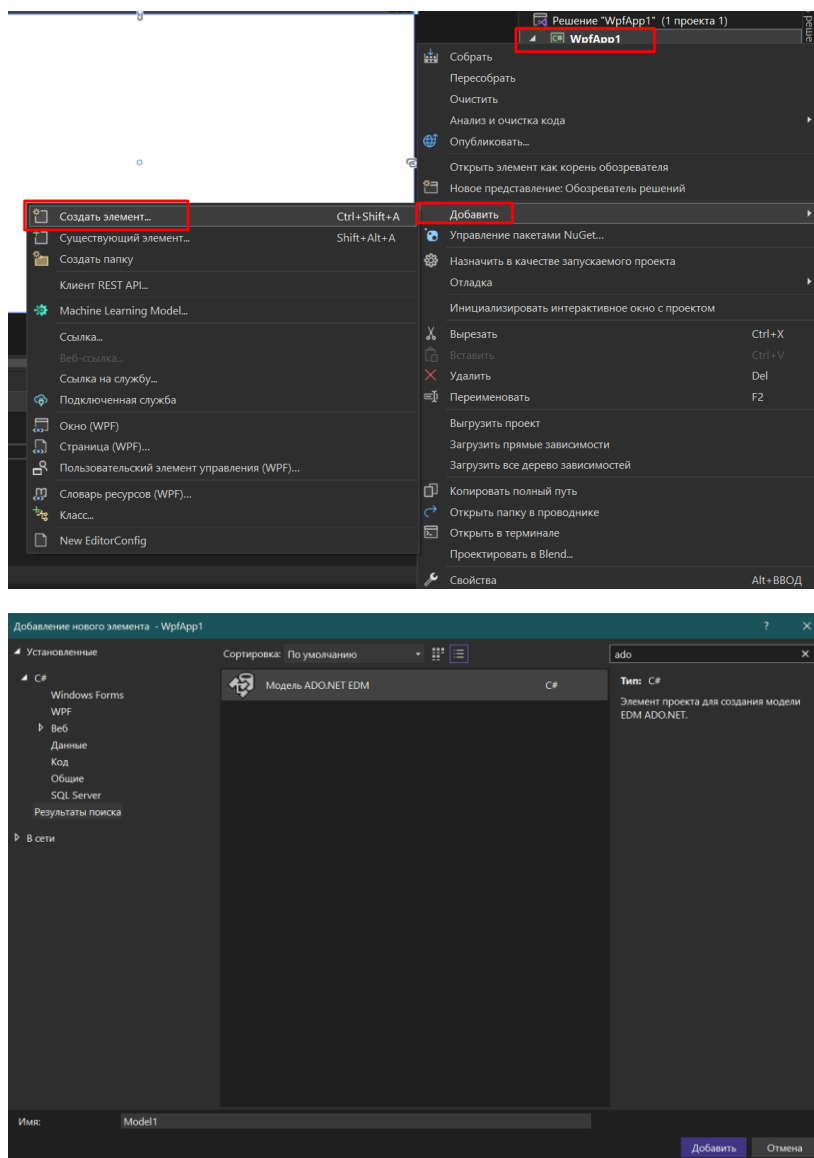


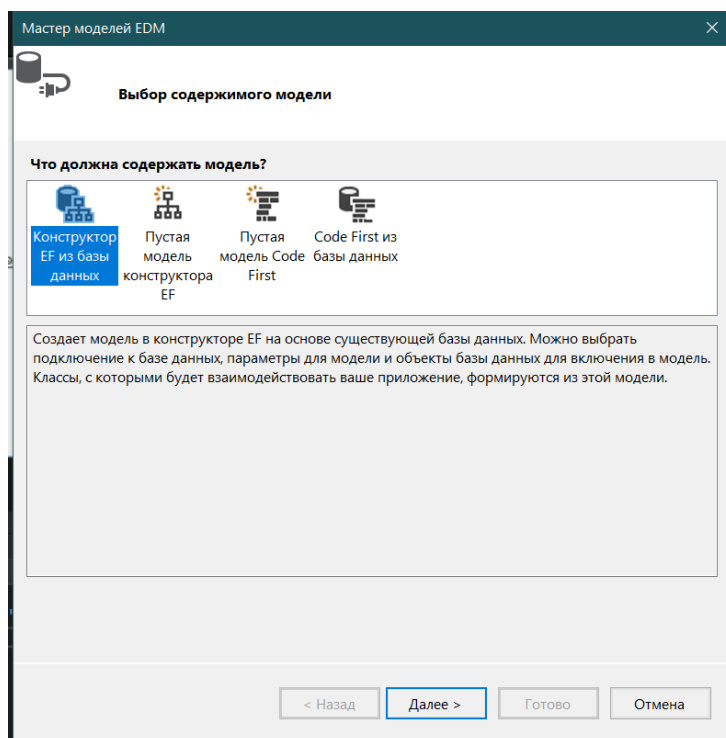
Работа с данными через модель ADO.NET

Создание ADO.NET	2
Создание списка через ListView	9
Авторизация (хеширование в отдельном файле)	13
Фильтрация данных	14
Отображение изменений в БД.....	15
Удаление списка данных	16
Работа с конкретным элементом данных.....	17

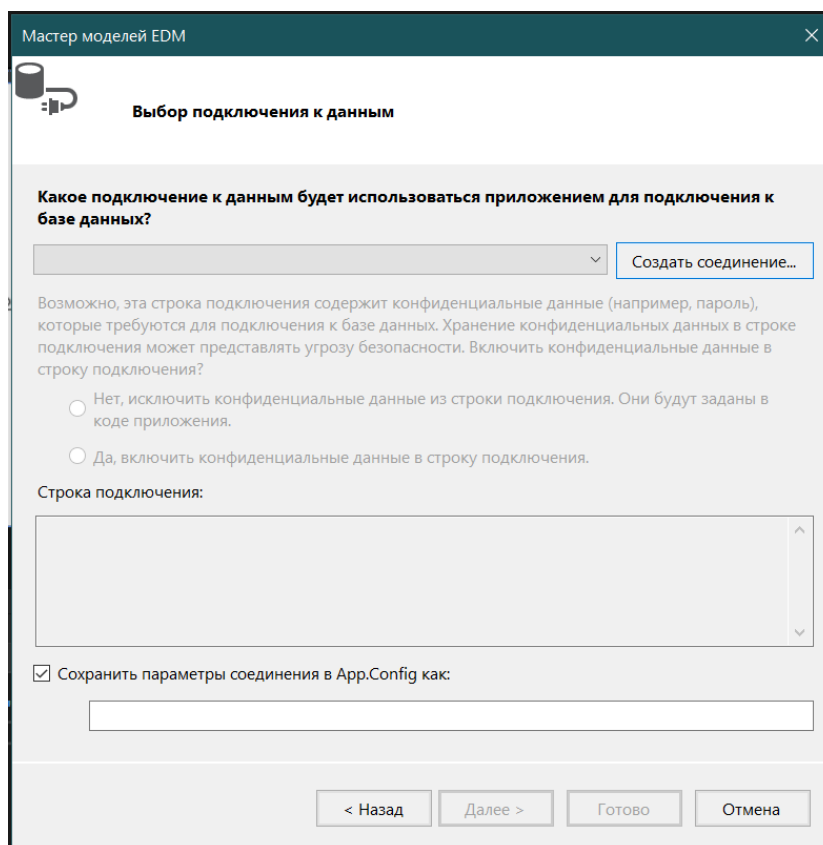
Создание ADO.NET

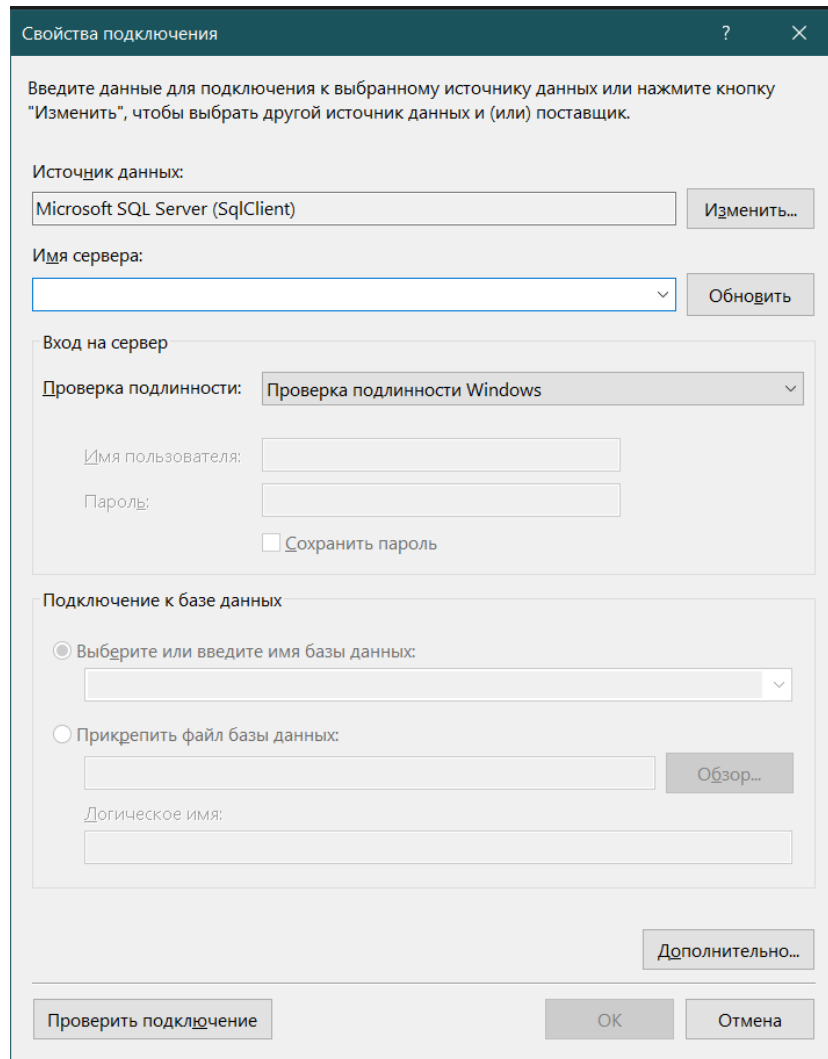
Для работы с БД необходимо добавить Модель ADO.NET





Указываем данные для подключения к Базе Данных





Свойства подключения

Введите данные для подключения к выбранному источнику данных или нажмите кнопку "Изменить", чтобы выбрать другой источник данных и (или) поставщик.

Источник данных:
Microsoft SQL Server (SqlClient) Изменить...

Имя сервера:
Обновить

Вход на сервер

Проверка подлинности: Проверка подлинности Windows

Имя пользователя:

Пароль:

☐ Сохранить пароль

Подключение к базе данных

☒ Выберите или введите имя базы данных:

☐ Прикрепить файл базы данных:
 Обзор...

Логическое имя:

Дополнительно...

Проверить подключение ОК Отмена

Указываем параметры создаваемой модели ADO.NET

Мастер моделей EDM

Выбор подключения к данным

Какое подключение к данным будет использоваться приложением для подключения к базе данных?

desktop-nheuae9\squlexpress.up_11_02.dbo Создать соединение...

Возможно, эта строка подключения содержит конфиденциальные данные (например, пароль), которые требуются для подключения к базе данных. Хранение конфиденциальных данных в строке подключения может представлять угрозу безопасности. Включить конфиденциальные данные в строку подключения?

☐ Нет, исключить конфиденциальные данные из строки подключения. Они будут заданы в коде приложения.

☐ Да, включить конфиденциальные данные в строку подключения.

Строка подключения:

```
metadata=res://*/Model1.csdl|res://*/Model1.ssdl|
res://*/Model1.msl;provider=System.Data.SqlClient;provider connection string="data
source=DESKTOP-NHEUA9\SQLEXPRESS;initial catalog=up_11_02;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"
```

☒ Сохранить параметры соединения в App.Config как:

up_11_02Entities

< Назад Далее > Готово Отмена

Мастер моделей EDM

Выберите параметры и объекты базы данных

Какие объекты базы данных нужно включить в модель?

☒ Таблицы

☐ Представления

☐ Хранимые процедуры и функции

☒ Формировать имена объектов во множественном или единственном числе

☒ Включить столбцы внешних ключей в модель

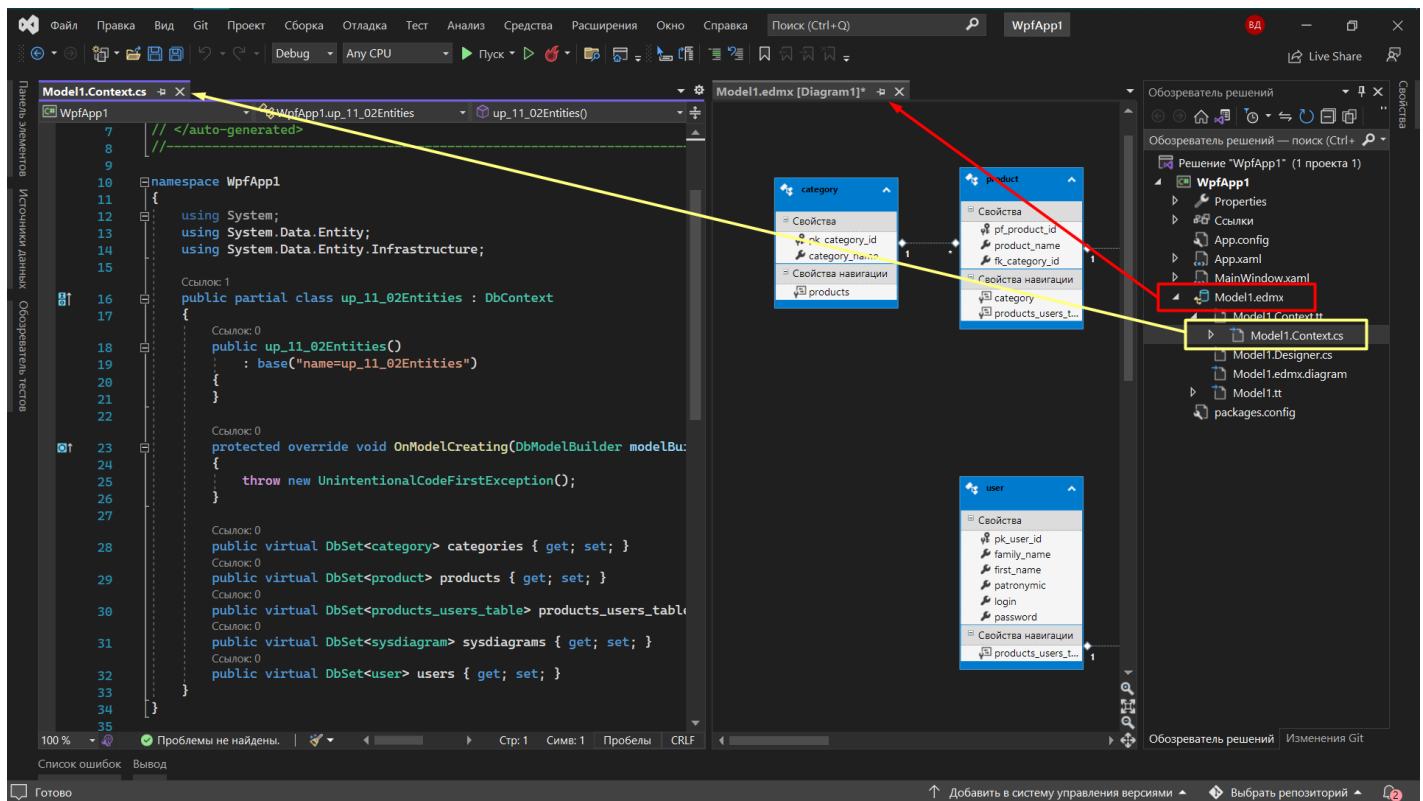
☒ Импортировать выбранные хранимые процедуры и функции в модель сущностей

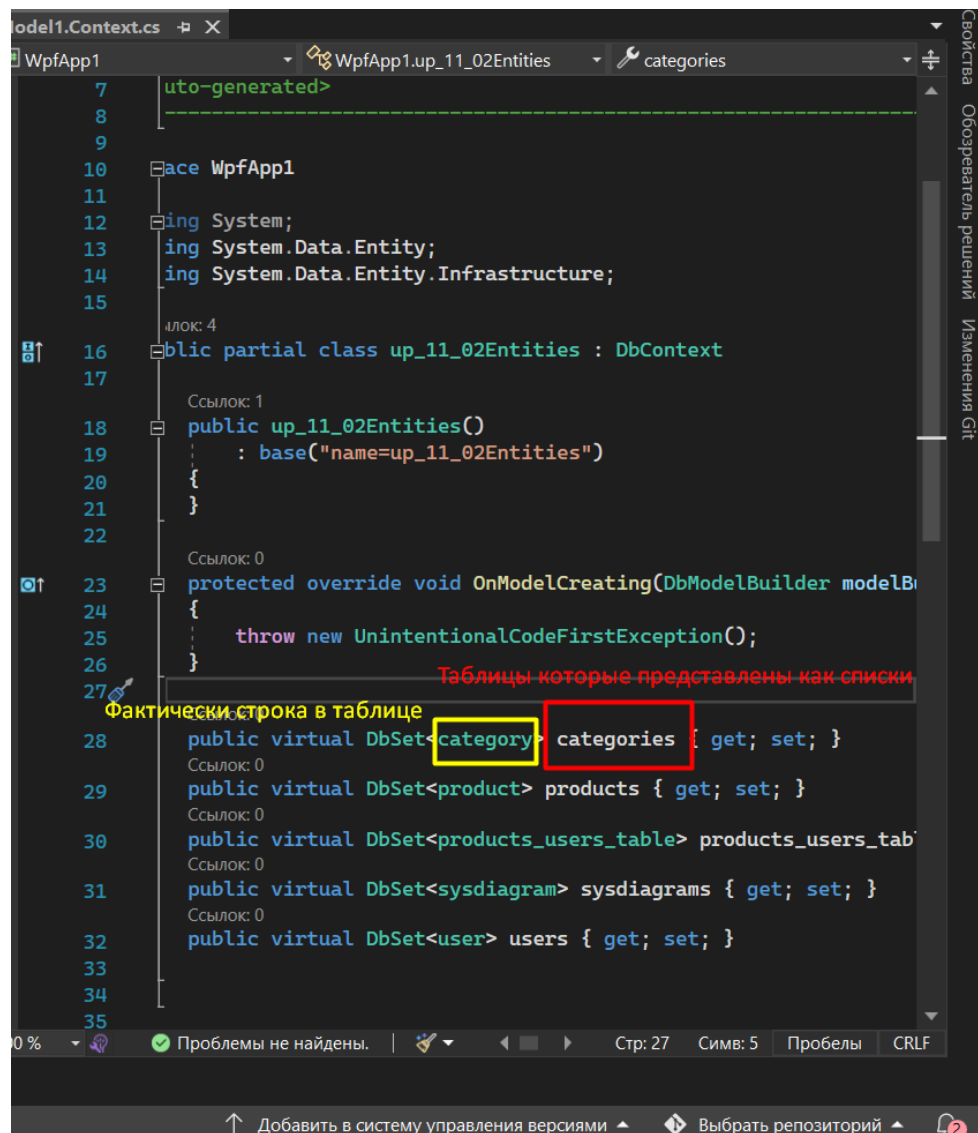
Пространство имен модели:

up_11_02Model

< Назад Далее > Готово Отмена

Созданная модель ADO.NET



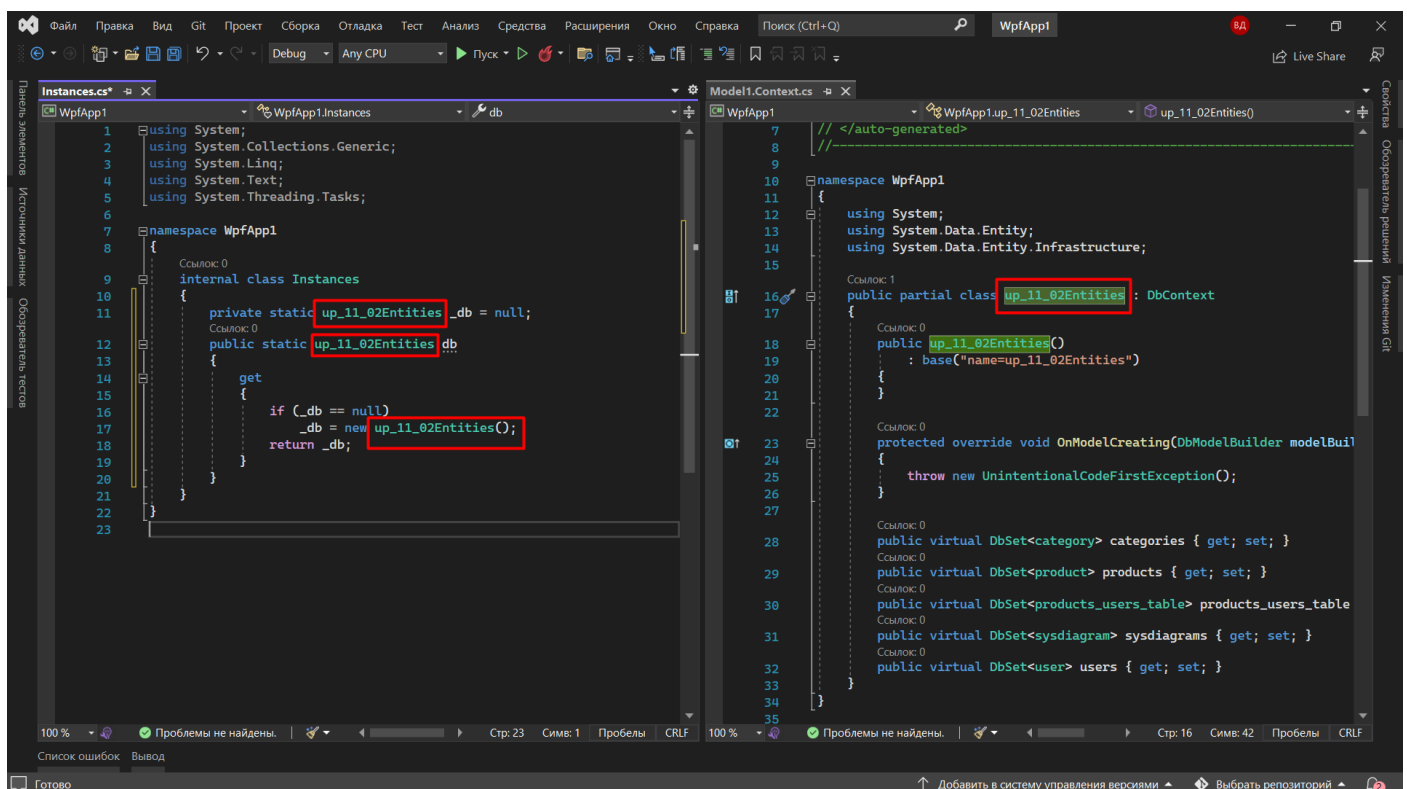
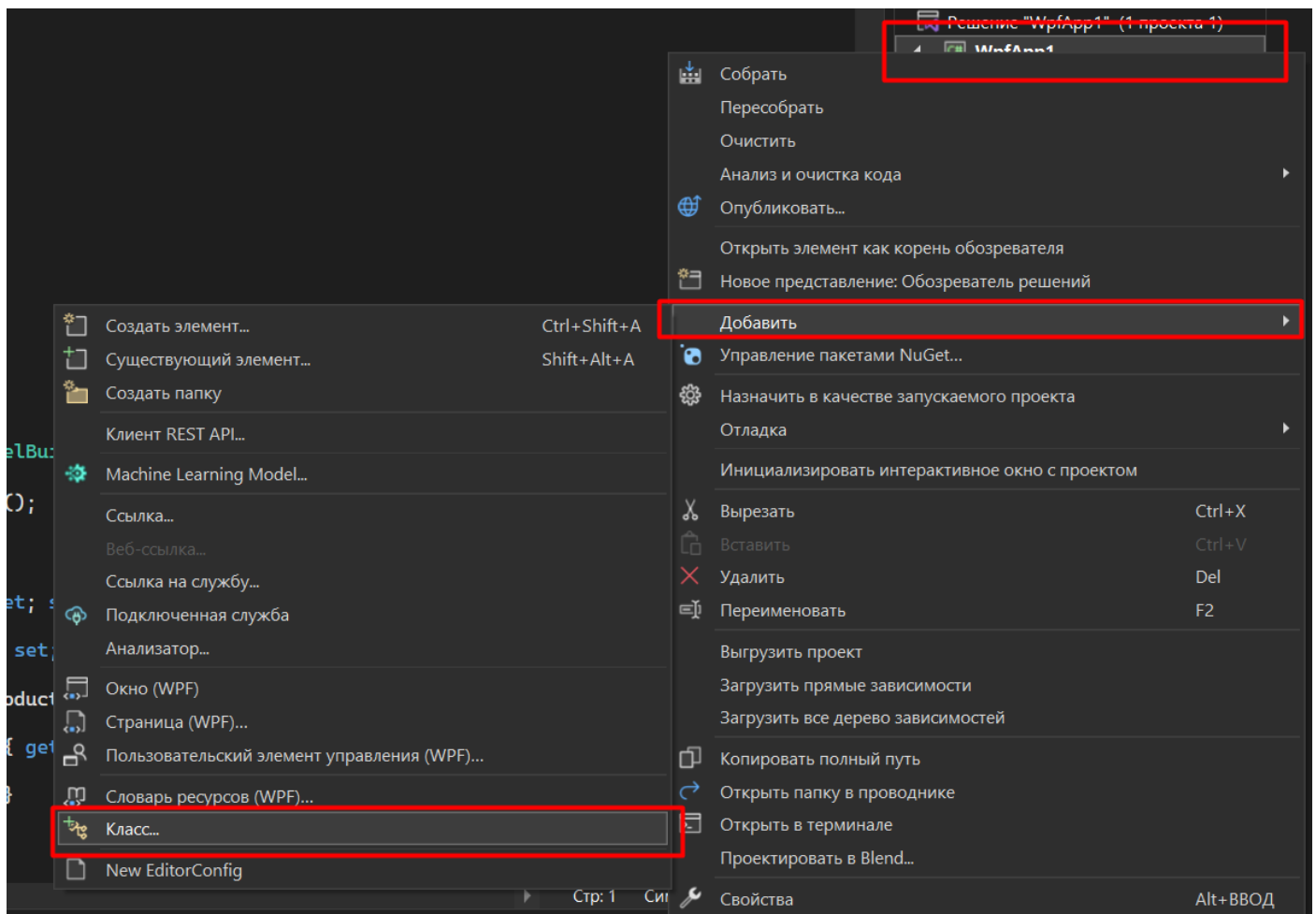


```
7  <!--auto-generated-->
8
9
10 namespace WpfApp1
11 {
12     using System;
13     using System.Data.Entity;
14     using System.Data.Entity.Infrastructure;
15
16     [Block: 4]
17     public partial class up_11_02Entities : DbContext
18     {
19         [Link: 1]
20         public up_11_02Entities()
21             : base("name=up_11_02Entities")
22         {
23         }
24
25         [Link: 0]
26         protected override void OnModelCreating(DbModelBuilder modelBuilder)
27         {
28             throw new UnintentionalCodeFirstException();
29         }
30
31         [Link: 0]
32         public virtual DbSet<category> categories { get; set; }
33
34         [Link: 0]
35         public virtual DbSet<product> products { get; set; }
36
37         [Link: 0]
38         public virtual DbSet<products_users_table> products_users_table { get; set; }
39
40         [Link: 0]
41         public virtual DbSet<sysdiagram> sysdiagrams { get; set; }
42
43         [Link: 0]
44         public virtual DbSet<user> users { get; set; }
45     }
46 }
```

Фактически строка в таблице

Таблицы которые представлены как списки

Для подключения с Базе Данных всегда используется синглтон (конструкция, которая НЕ позволяет создавать множество одинаковых объектов). Синглтон выносится в отдельный класс.



Теперь Базу данных необходимо вызывать как **Instances.db**.

Создание списка через ListView

```
<ListView>
  <ListView.ItemTemplate>
    <DataTemplate>
      <Grid>
        <TextBlock Text="{Binding }"/>
      </Grid>
    </DataTemplate>
  </ListView.ItemTemplate>
</ListView>
```

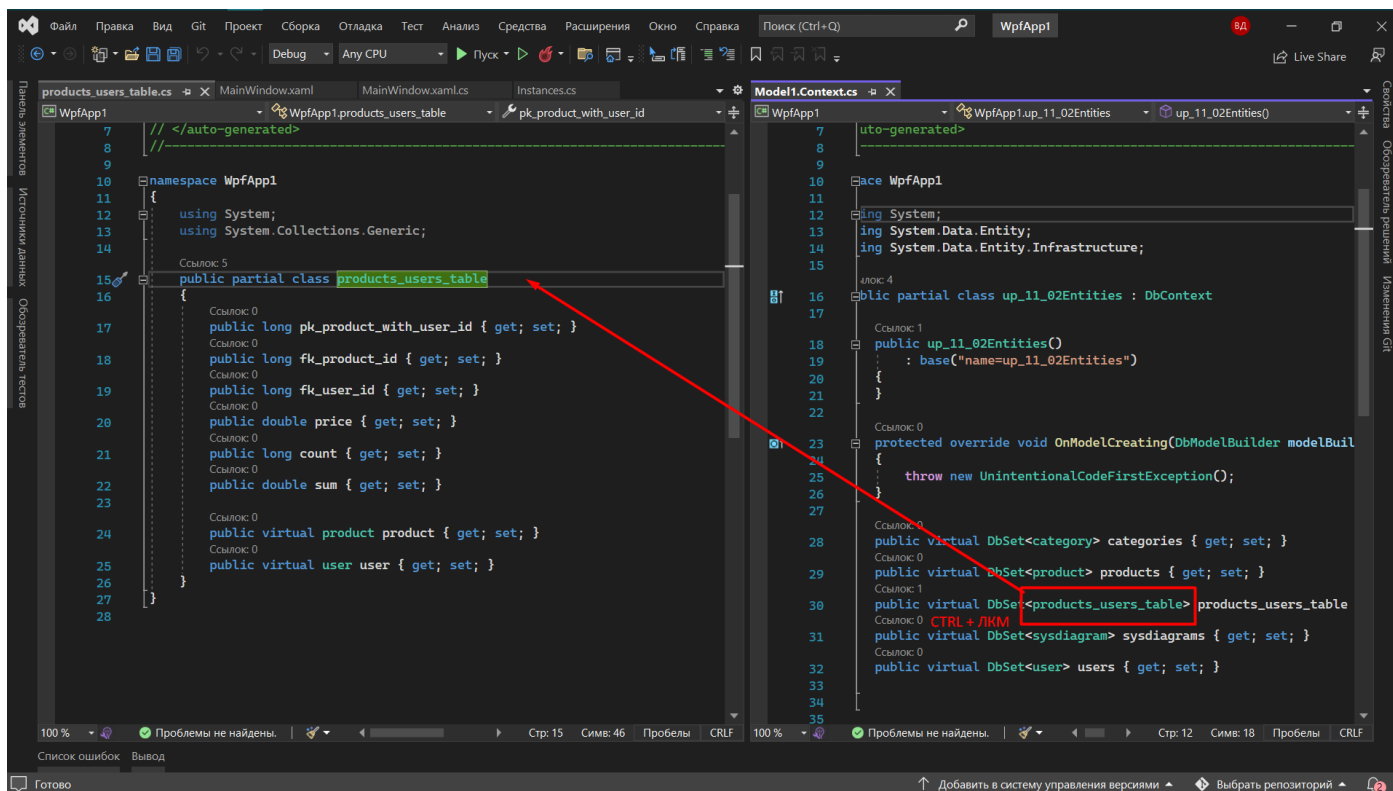
Внутри DataTemplate может быть любой удобный для вас контейнер (Grid, StackPanel и т.д.). Макет не будет отображаться, необходимо создавать макет «вслепую». Для LISTVIEW обязательно необходимо добавить **x:Name** по которому можно будет найти данный элемент.

```
Ссылка: 2
public partial class MainWindow : Window
{
    Ссылка: 0
    public MainWindow()
    {
        InitializeComponent();

        //ItemSource - указать источник данных для отрисовки дочерних
        //                элементов из DataTemplatea
        //Instances.db - БД
        //products_users_table - Таблица
        //OrderBy - отсортировать все записи по параметру pk_product_with_user_id
        //Skip - пропустить N записей
        //Take - получить следующие N записей
        //ToList - преобразовать данный набор данных в список

        listView.ItemsSource =
            Instances.db.products_users_table
                .OrderBy(q => q.pk_product_with_user_id)
                .Skip(0)
                .Take(50)
                .ToList();
    }
}
```

Необходимо определиться какие данные куда производить Binding



```
9
10 namespace WpfApp1 Класс строки в таблице
11 {
12     using System;
13     using System.Collections.Generic;
14
15     Ссылки: 5
16     public partial class products_users_table
17     { Все столбцы таблицы указаны как свойства (get/set)
18
19         Ссылки: 0
20         public long pk_product_with_user_id { get; set; }
21         Ссылки: 0
22         public long fk_product_id { get; set; }
23         Ссылки: 0
24         public long fk_user_id { get; set; }
25         Ссылки: 0
26         public double price { get; set; }
27         Ссылки: 0
28         public long count { get; set; }
29         Ссылки: 0
30         public double sum { get; set; }
31
32         Ссылки: 0
33         public virtual product product { get; set; }
34         Ссылки: 0
35         public virtual user user { get; set; }
36     }
37
38     Ссылки на связанные строки из внешних таблиц.
39
40     Т.к. таблицы PRODUCTS_WITH_USERS и PRODUCTS связаны 1 ко МНОГИМ
(каждой записи PROD_WTH_USERS соответствует только 1 PRODUCT),
то здесь сразу можно вызвать связанную запись. Без явного использования
INNER JOIN.
```

100 % Проблемы не найдены. Стр: 14 Симв: 5 Пробелы CRLF

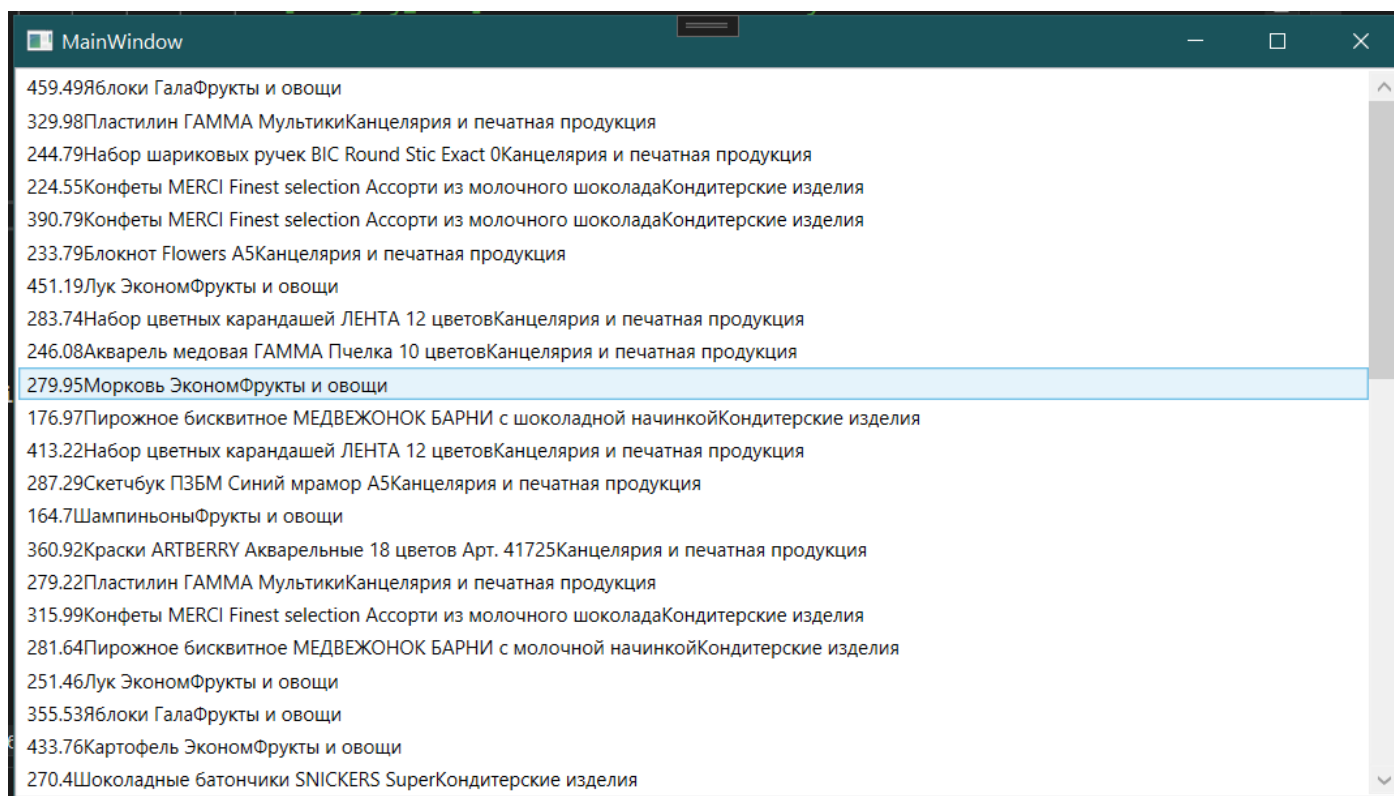
```
<ListView Name="listView">
  <ListView.ItemTemplate>
    <DataTemplate>
      <Grid>
        <!--
          Т.к. в ItemSource находятся данные из таблицы
          products_users_table, то с данным классом и необходимо
          работать.

          В нем есть свойство [price], которое мы можем сразу указать
          для Binding.

          [product_name] мы не можем явно указать.
          Нам необходимо "перейти" в связанную строку таблицы
          products и уже в ней есть свойство [product_name].

          [category_name] мы также не можем явно указать.
          Нам необходимо "перейти" в связанную строку таблицы
          products, из нее перейти в таблицу categories,
          и уже в ней есть свойство [category_name].
        -->
        <TextBlock Text="{Binding price}"/>
        <TextBlock Text="{Binding product.product_name}"/>
        <TextBlock Text="{Binding product.category.category_name}"/>
      </Grid>
    </DataTemplate>
  </ListView.ItemTemplate>
</ListView>
```

Ровно 50 записей, каждая из которых отображает информацию о Цене, Продукте, Категории.



Авторизация (хеширование в отдельном файле)

```
//FirstOrDefault - получить первый элемент который удовлетворет условию,  
// если такого элемента нет, вернуть Null  
  
var user = Instances.db.users  
    .FirstOrDefault(q => q.login.Contains("") && q.password.Contains(""));  
  
if (user != null)  
{  
    //Такой пользователь существует  
}  
else  
{  
    //Такого пользователя не существует  
}
```

Фильтрация данных

//Where – получить все элементы которые удовлетворет условию

```
var data = Instances.db.products_users_table
    .Where(q => q.price > 100)
    .OrderBy(w => w.pk_product_with_user_id)
    .Skip(0)
    .Take(50)
    .ToList();
```

Отображение изменений в БД

```
//Перезагрузить все данные из базы данных
```

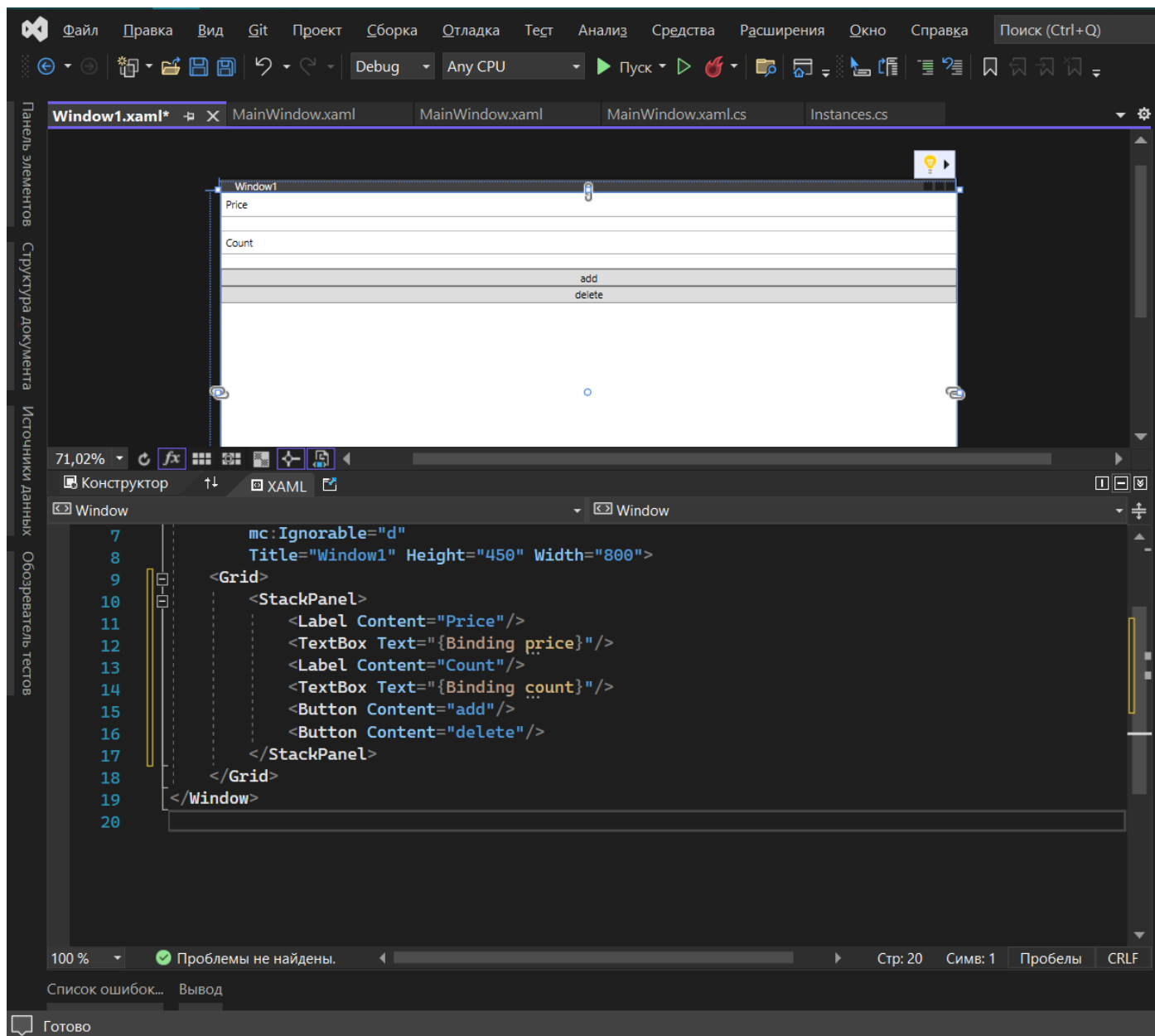
```
Instances.db.ChangeTracker.Entries().ToList().ForEach(q => q.Reload());
```

Удаление списка данных

```
//Cast – преобразовать данные в указанный формат  
//RemoveRange – удалить набор данных  
//SaveChanges – сохранить все внесенные изменения  
  
var removed_data = listView.SelectedItems.Cast<products_users_table>().ToList();  
Instances.db.products_users_table.RemoveRange(removed_data);  
Instances.db.SaveChanges();
```


Работа с конкретным элементом данных

Создана форма для редактирования цены и количества.



Ссылка: 0

```
public Window1(products_users_table currentItem)
{
    InitializeComponent();

    //Если передан null, то данная форма используется для добавления новой записи
    //и необходимо создать новую строку в Базе данных

    //Если передан объект (строка), то данная форма используется для редактирования

    if (currentItem == null)
        currentItem = new products_users_table();

    //Установить связь между currentItem и формой.
    //Чтобы корректно работал Binding. Все данные которые будут вноситься в TextBox
    //будут автоматически переноситься в currentItem

    this.DataContext = currentItem;
}
```

Ссылка: 0

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    //Получить обратно закрепленную строку таблицы данные которой мы редактировали
    var item = this.DataContext as products_users_table;

    //Если новая строка – добавит значение,
    //Если строка старая – изменит необходимые значения
    Instances.db.products_users_table.AddOrUpdate(item);
    Instances.db.SaveChanges();
}
```

Ссылка: 0

```
private void Button_Click_1(object sender, RoutedEventArgs e)
{
    var item = this.DataContext as products_users_table;

    //Удалить текущий элемент
    Instances.db.products_users_table.Remove(item);
    Instances.db.SaveChanges();
}
```