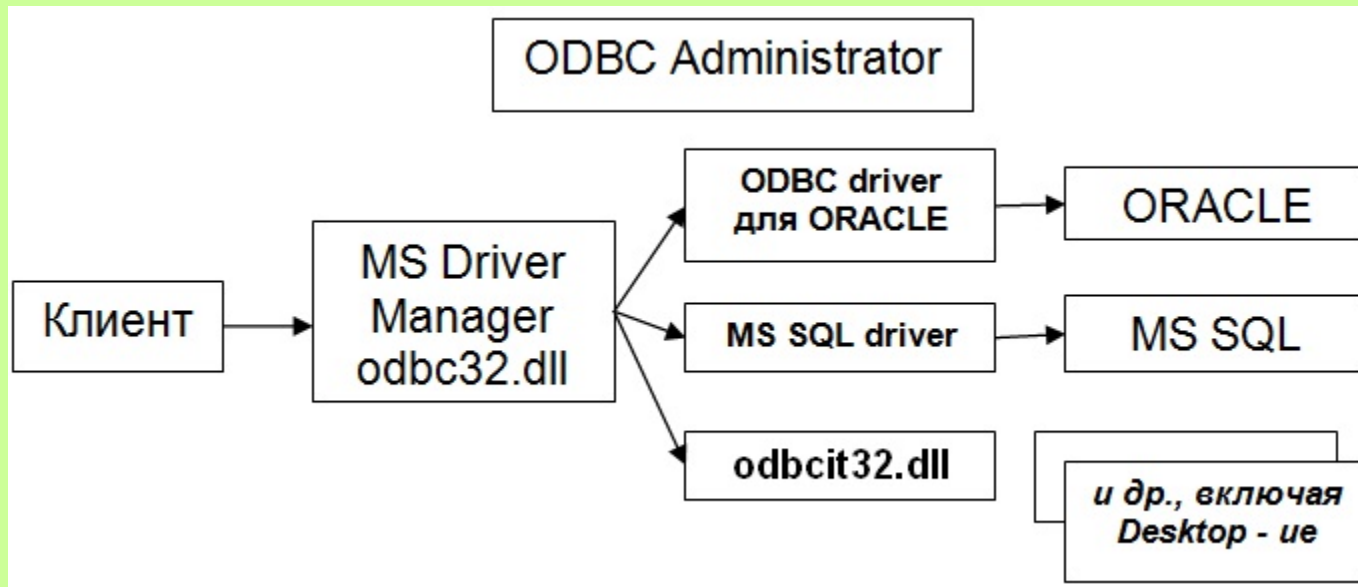


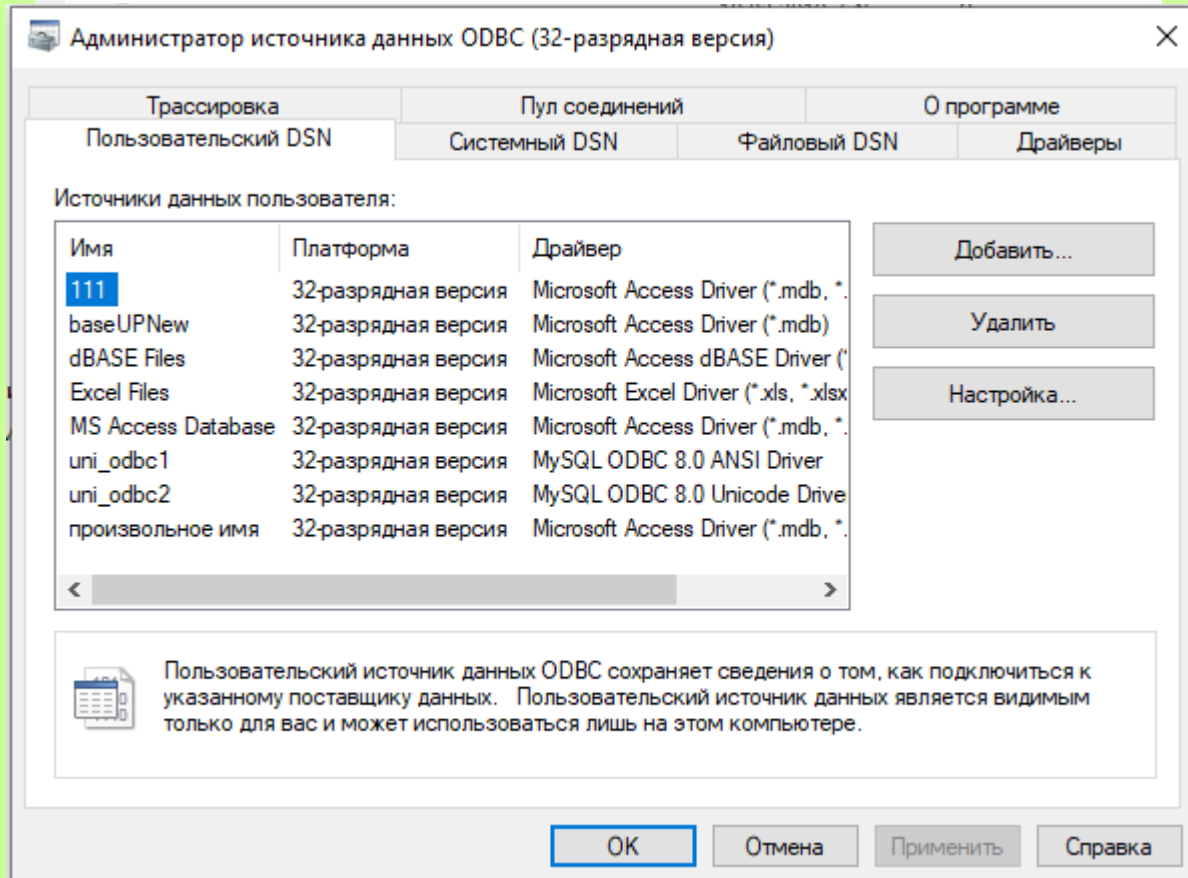
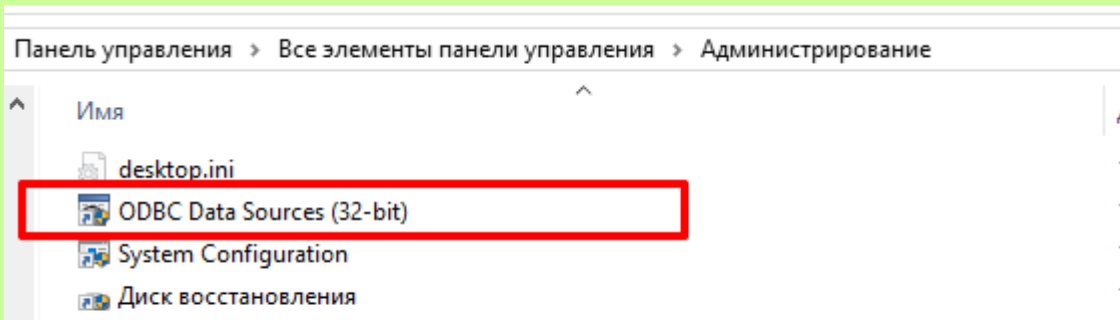
Практическое применение БД в программных системах

ODBC

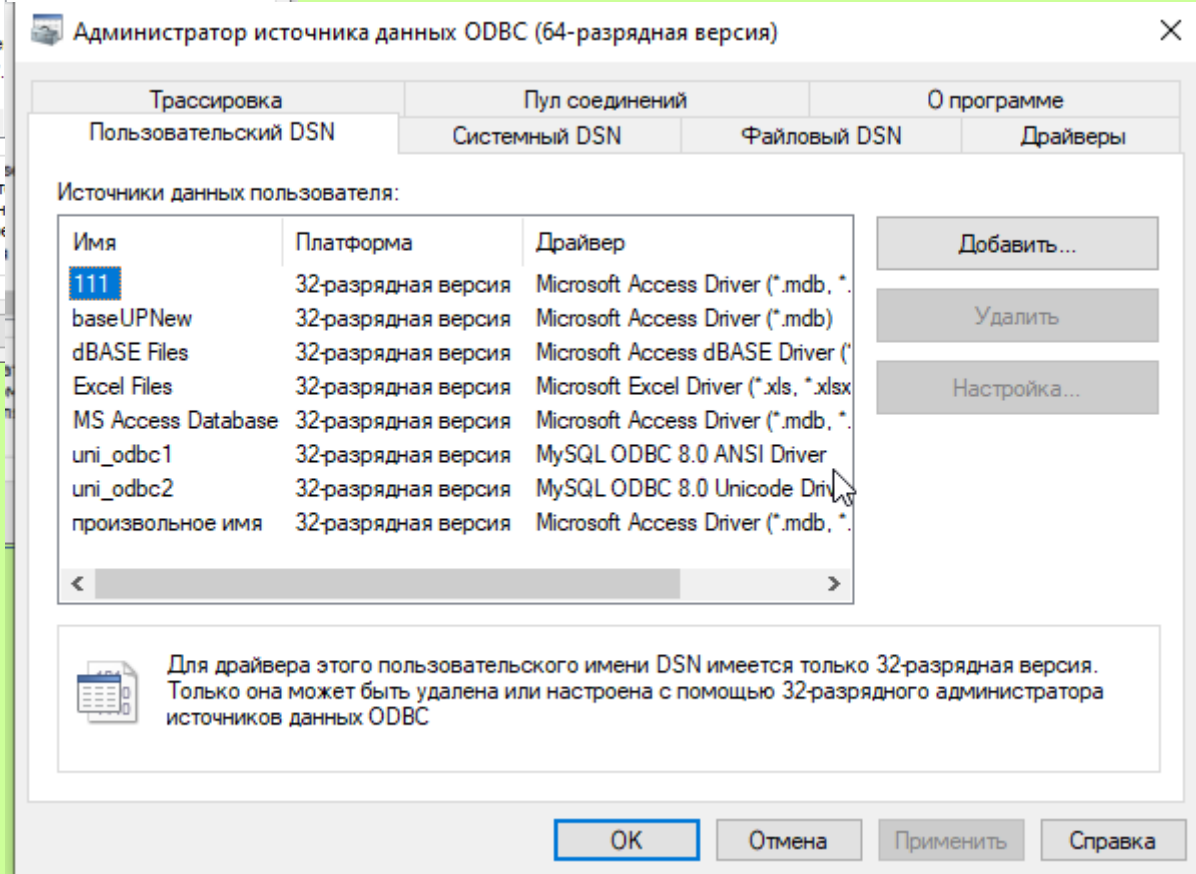
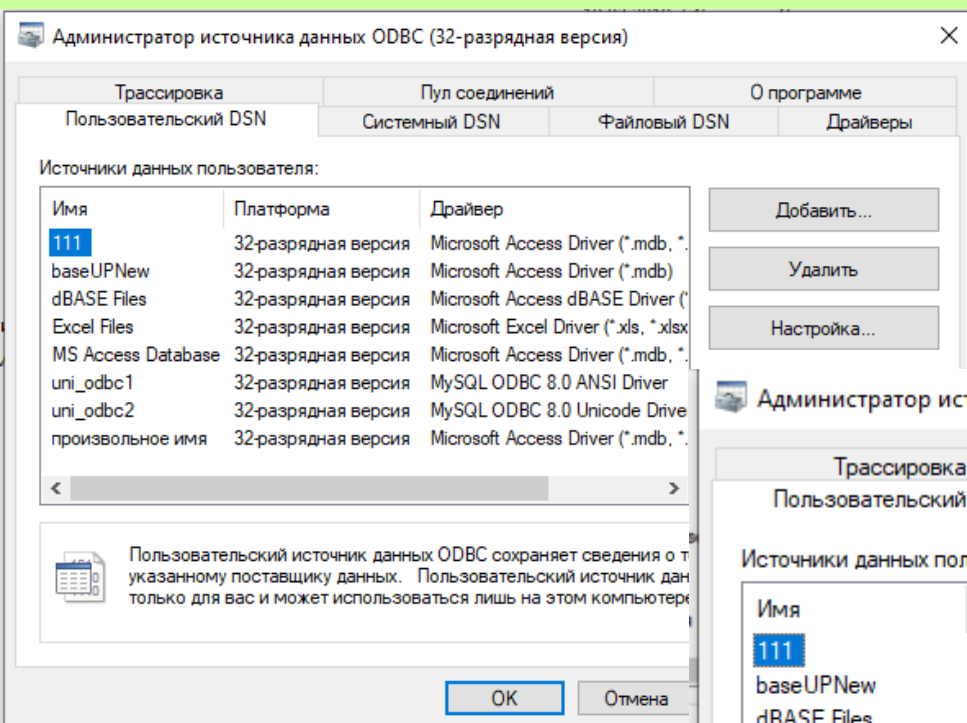
- ODBC (англ. Open Database Connectivity) — это программный интерфейс (API) доступа к базам данных, разработанный фирмой Microsoft, в сотрудничестве с Simba Technologies на основе спецификаций Call Level Interface (CLI)



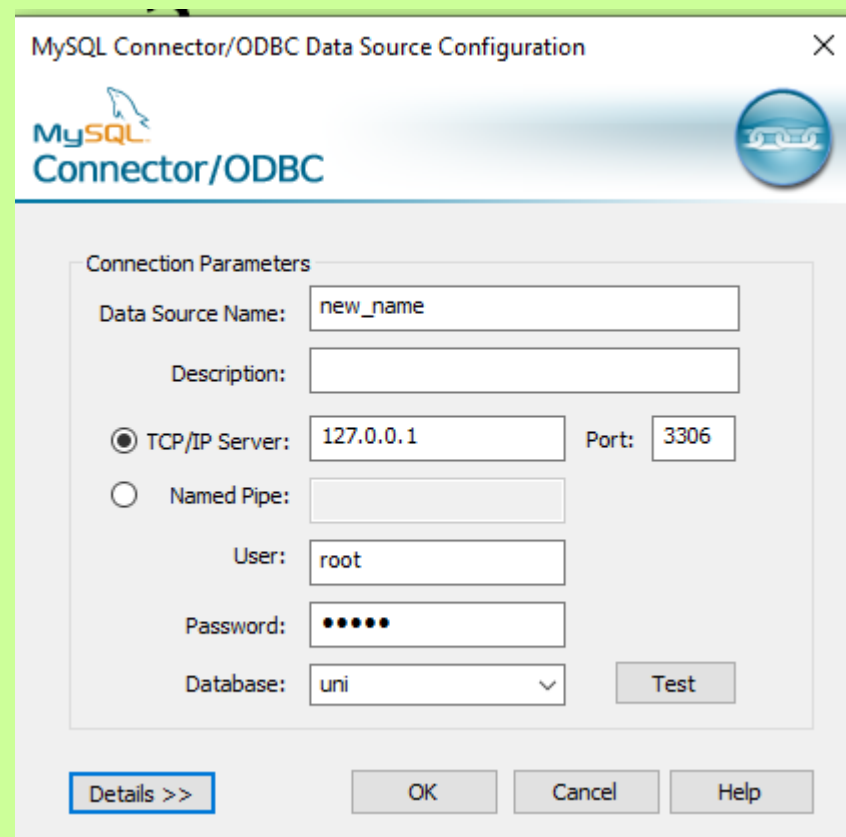
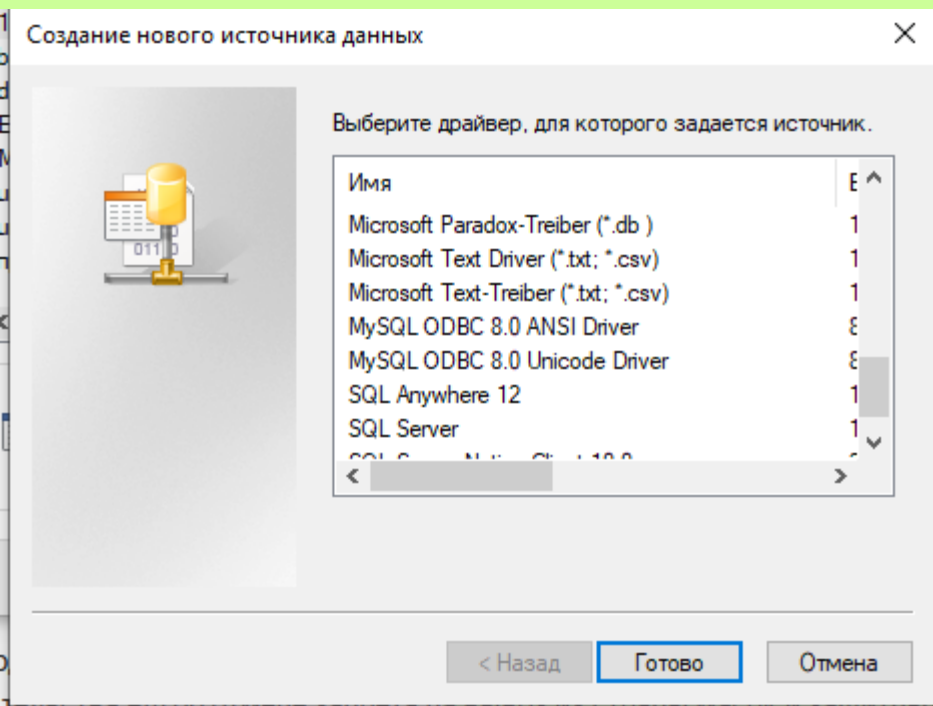
ODBC



ODBC



Создание ODBC



ORM

- **ORM (Object-Relational Mapping)** – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как проприетарные, так и свободные реализации этой технологии.
- Библиотеки ORM существуют для самых разных языков программирования. В общих чертах, технология ORM позволяет проектировать работу с данными в терминах классов, а не таблиц данных. Она позволяет преобразовывать классы в данные, пригодные для хранения в базе данных, причем схему преобразования определяет сам разработчик. Кроме того, ORM предоставляет простой API-интерфейс для CRUD-операций над данными. Благодаря технологии ORM нет необходимости писать SQL-код для взаимодействия с локальной базой данных

Entity Framework

- **Entity Framework** — это инструмент, упрощающий сопоставление объектов в программном обеспечении с таблицами и столбцами реляционной базы данных.
Entity Framework (EF) — это ORM-фреймворк с открытым исходным кодом для ADO.NET, который является частью .NET Framework.

Hibernate

- **JPA** — это технология, обеспечивающая объектно-реляционное отображение простых JAVA объектов и предоставляющая API для сохранения, получения и управления такими объектами.
- **Hibernate** — самая популярная реализация спецификации JPA (), предназначенная для решения задач объектно-реляционного отображения (ORM). Распространяется свободно на условиях GNU Lesser General Public License.

ORM (Object-Relational Mapping)

ORM

- **ORM (Object-Relational Mapping)** – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как проприетарные, так и свободные реализации этой технологии.
- Библиотеки ORM существуют для самых разных языков программирования. В общих чертах, технология ORM позволяет проектировать работу с данными в терминах классов, а не таблиц данных. Она позволяет преобразовывать классы в данные, пригодные для хранения в базе данных, причем схему преобразования определяет сам разработчик. Кроме того, ORM предоставляет простой API-интерфейс для CRUD-операций над данными. Благодаря технологии ORM нет необходимости писать SQL-код для взаимодействия с локальной базой данных

Entity Framework

- **Entity Framework** — это инструмент, упрощающий сопоставление объектов в программном обеспечении с таблицами и столбцами реляционной базы данных. **Entity Framework (EF)** — это ORM-фреймворк с открытым исходным кодом для ADO.NET, который является частью .NET Framework.

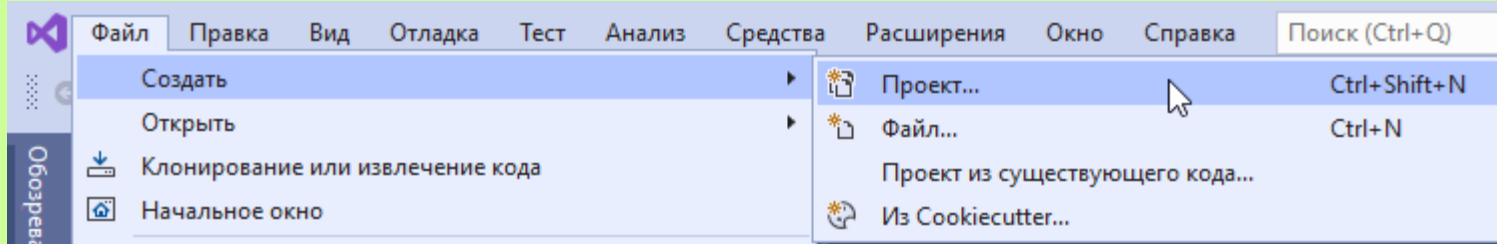
ORM в .NET

- **Entity Framework (EF)** — это ORM-фреймворк с открытым исходным кодом для ADO.NET, который является частью .NET Framework.
- **NHibernate** — ORM-решение для платформы Microsoft .NET, портированное с Java. Это бесплатная библиотека с открытым кодом, распространяется под лицензией GNU Lesser General Public License.

nHibernate по шагам

- MySQL 8 ,установлен первым
- Visual Studio 2019 Community

nHibernate Шаг1



Настроить новый проект

Приложение Windows Forms (.NET Framework)

C#

Windows


Рабочий с

Имя проекта

hibernate_mysql_try

Расположение

... \ПБД\

Имя решения 

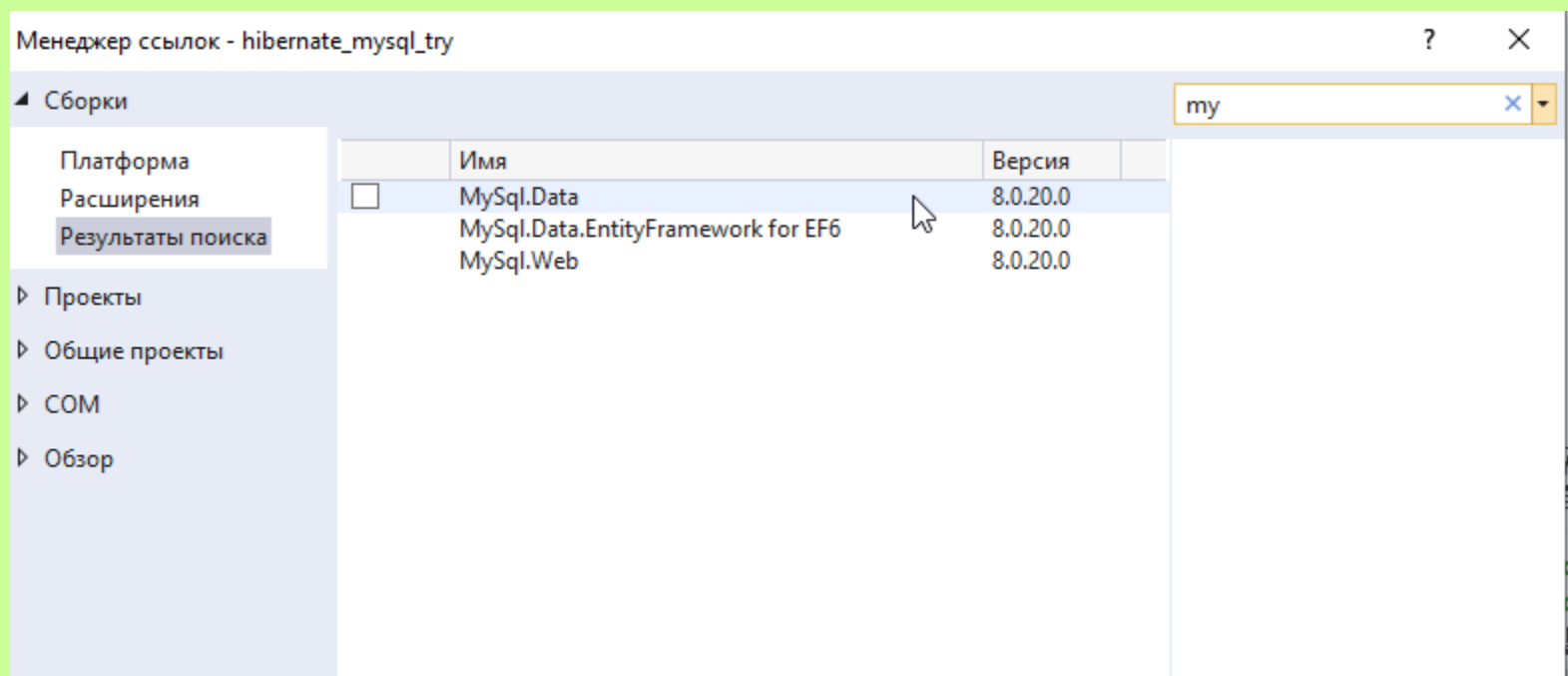
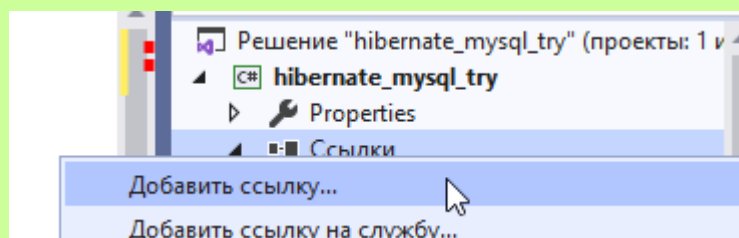
hibernate_mysql_try

☐ Поместить решение и проект в одном каталоге

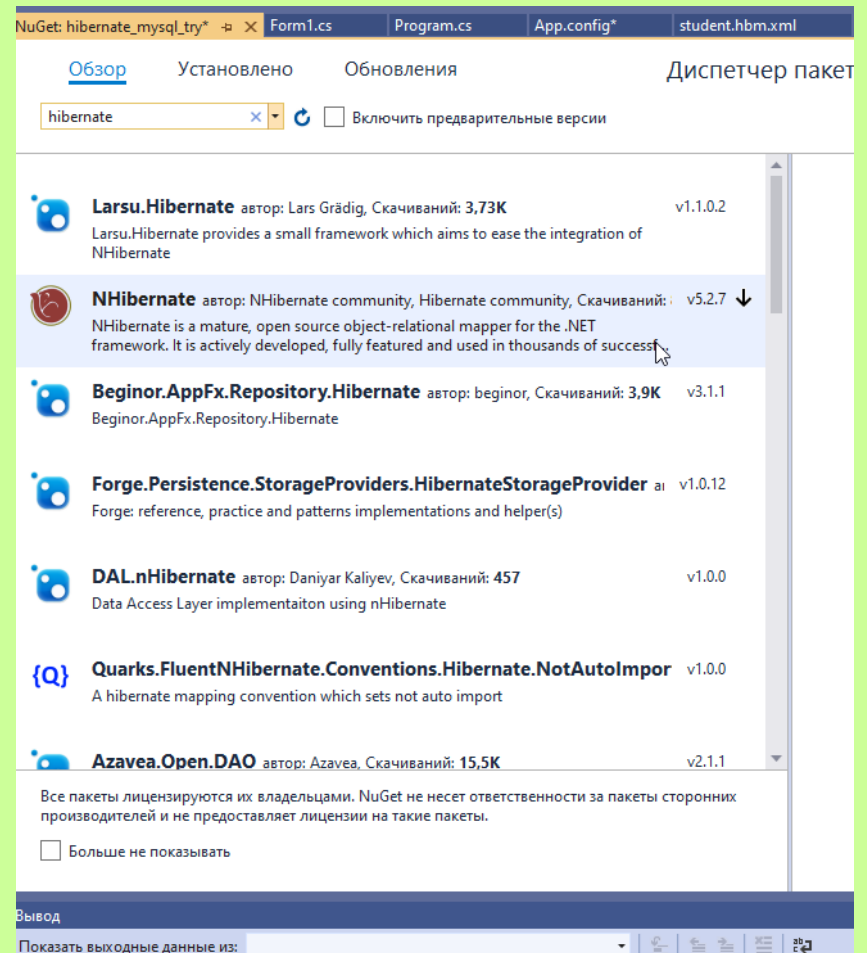
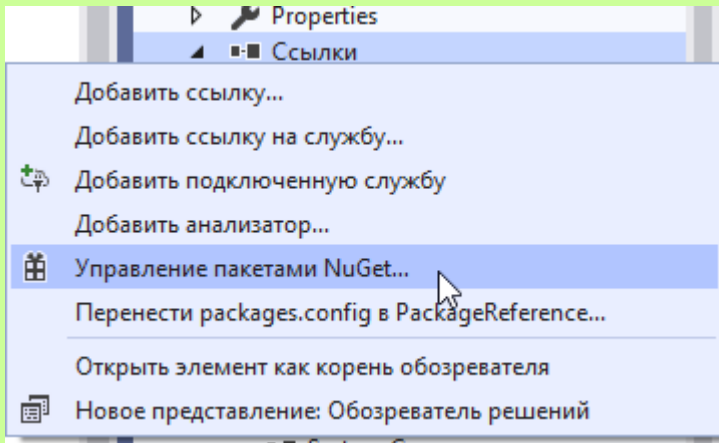
Платформа

.NET Framework 4.7.2

nHibernate Шаг1



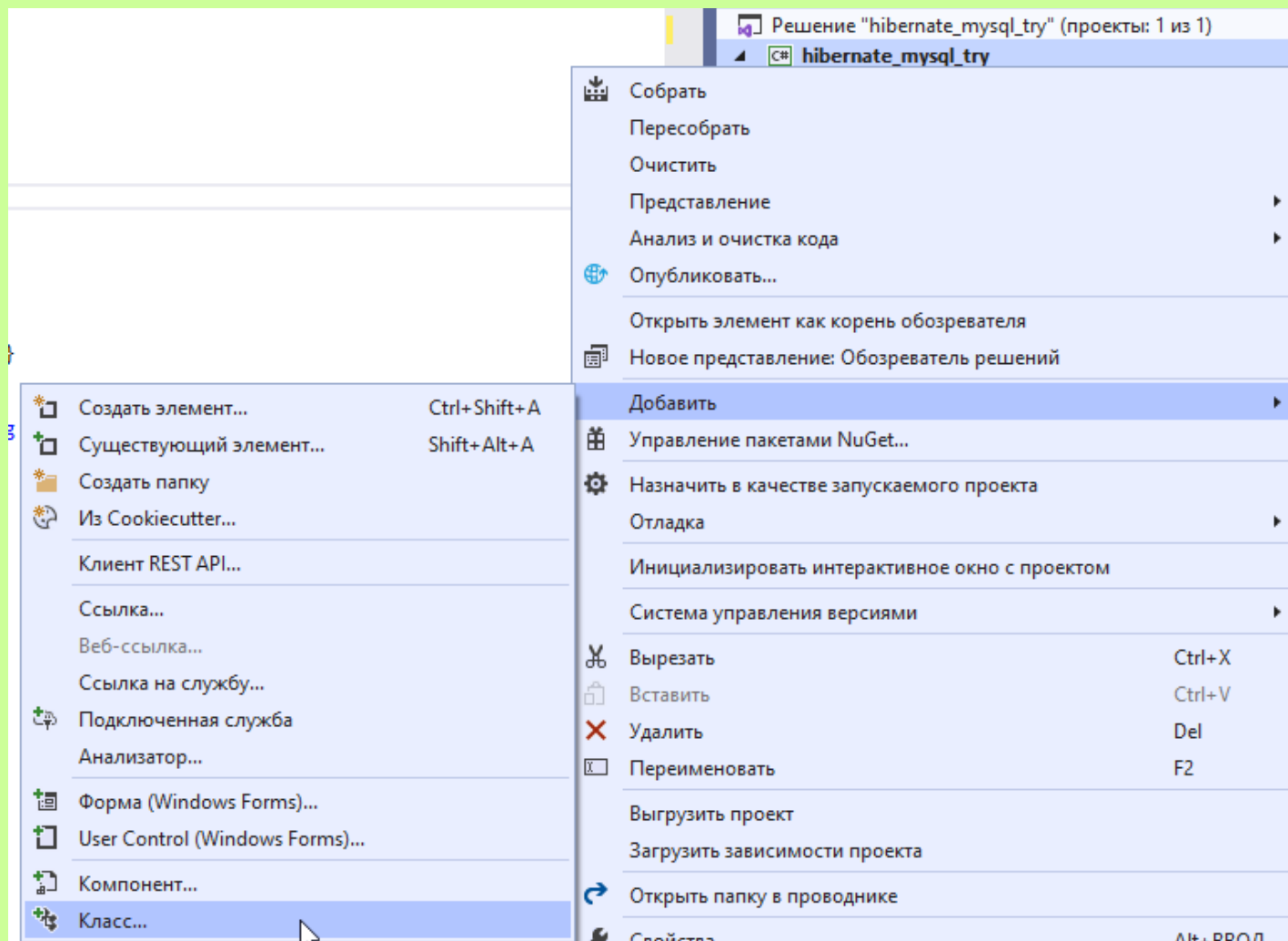
nHibernate Шаг1



nHibernate War1

```
hibernate_mysql_try hibernate_mysql_try.Form1
1 using MySql.Data.MySqlClient;
2 using NHibernate;
3 using NHibernate.Cfg;
```

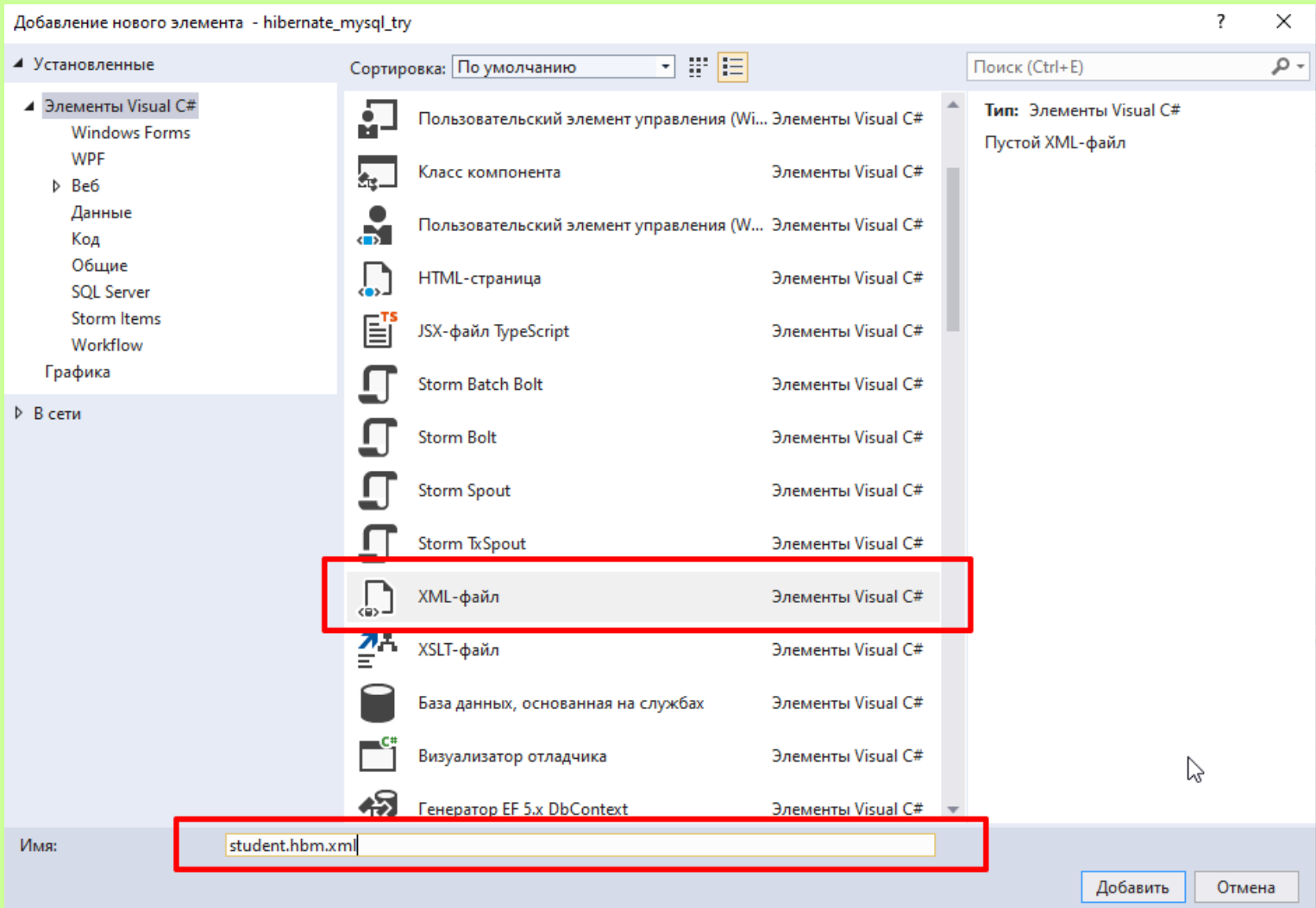
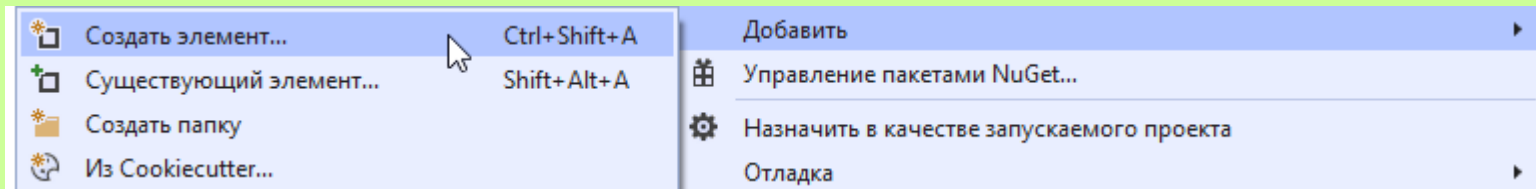
nHibernate Шаг2



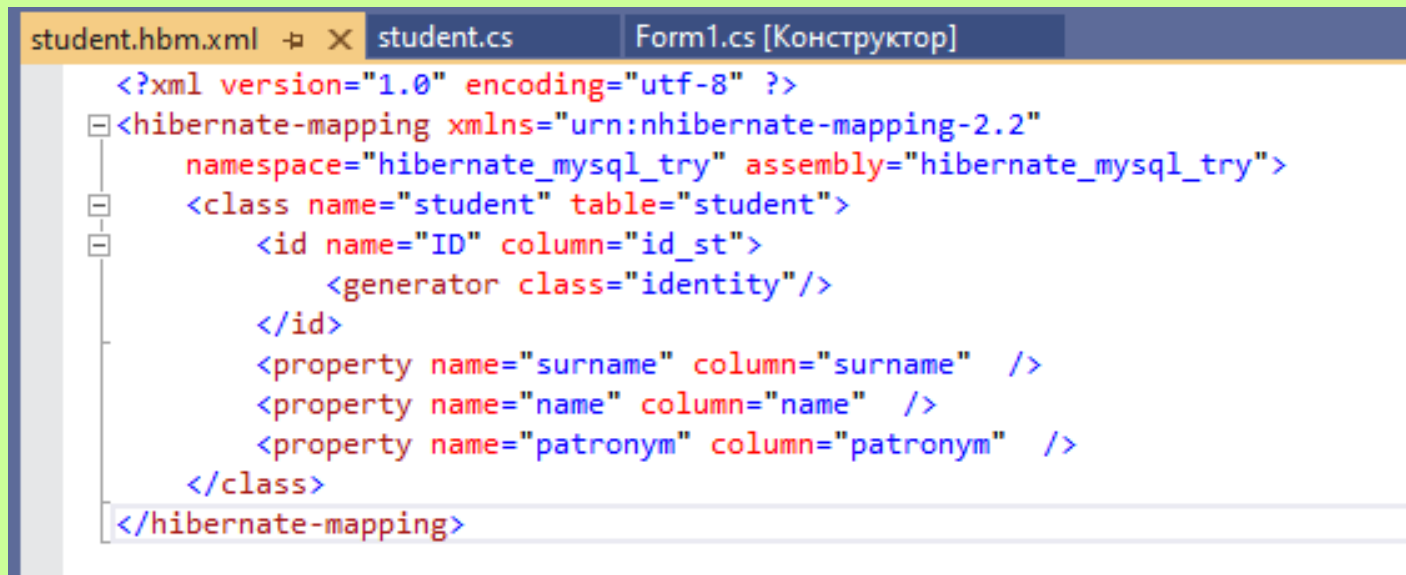
nHibernate War2

```
hibernate_mysql_try  hibernate_mysql_try.student  ID
1  using Microsoft.SqlServer.Server;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace hibernate_mysql_try
9  {
10     ссылка: 1
11     class student
12     {
13         private int id;
14         public string surname;
15         public string name;
16         public string patronym;
17
18         ссылка: 1
19         public int ID { get => id; set => id = value; }
20
21         Ссылок: 0
22         public student(int id_, string surname_, string name_, string patronym_)
23         {
24             ID = id_;
25             surname = surname_;
26             patronym = patronym_;
27         }
28     }
29 }
```

nHibernate Шаг3



nHibernate Шаг3



```
student.hbm.xml  X student.cs  Form1.cs [Конструктор]

<?xml version="1.0" encoding="utf-8" ?>
<hibernate-mapping xmlns="urn:nhibernate-mapping-2.2"
  namespace="hibernate_mysql_try" assembly="hibernate_mysql_try">
  <class name="student" table="student">
    <id name="ID" column="id_st">
      <generator class="identity"/>
    </id>
    <property name="surname" column="surname" />
    <property name="name" column="name" />
    <property name="patronym" column="patronym" />
  </class>
</hibernate-mapping>
```

App.config было

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
</configuration>
```

Form1.cs* Program.cs App.config student.hbm.xml student.cs Form1.cs [Конструкто

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="hibernate-configuration"
      type="NHibernate.Cfg.ConfigurationSectionHandler, NHibernate" />
  </configSections>

  <hibernate-configuration xmlns="urn:hibernate-configuration-5.2.7">
    <session-factory>
      <property name="connection.provider">
        NHibernate.Connection.DriverConnectionProvider
      </property>
      <property name="dialect">NHibernate.Dialect.MySQLDialect</property>
      <property name="query.substitutions">hqlFunction=SQLFUNC</property>
      <property name="connection.driver_class">
        NHibernate.Driver.MySqlDataDriver
      </property>
      <property name="connection.connection_string">
        server=127.0.0.1;port=3306;Uid=root;password=MySQL;database=uni
      </property>
      <property name="show_sql">true</property>
      <mapping assembly="hibernate_mysql_try" />
    </session-factory>
  </hibernate-configuration>
</configuration>
```

Строка соединения

- Сайт с примерами
<https://www.connectionstrings.com/mysql/>
- [MySQL Connector/Net](#)
- **Standard**
- Server=myServerAddress;Database=myDataBase;Uid=myUsername;Pwd=myPassword;
- **Specifying TCP port**
- Server=myServerAddress;Port=3306;Database=myDataBase;Uid=myUsername;Pwd=myPassword;
- **Multiple servers**
- Server=serverAddress1, serverAddress2, serverAddress3;Database=myDataBase;Uid=myUsername;Pwd=myPassword;

App.config стало



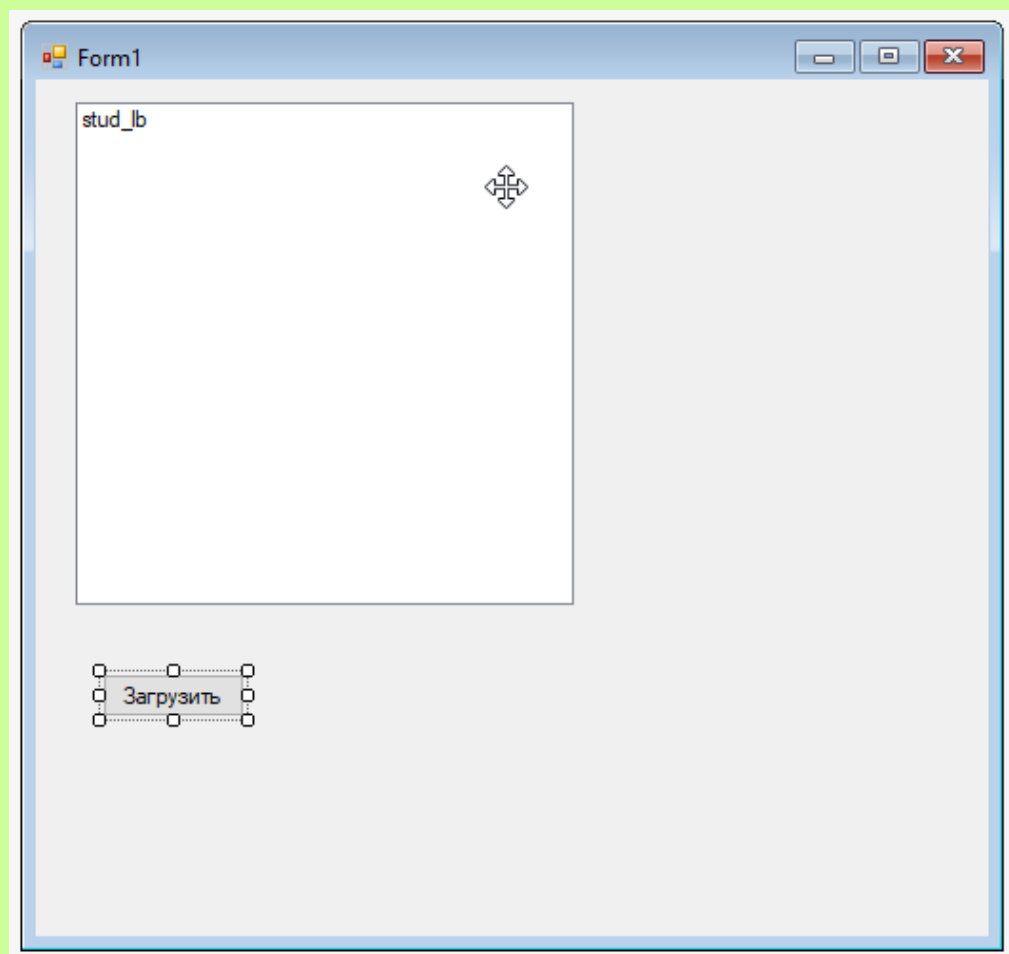
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <configSections>
    <section name="hibernate-configuration"
      type="NHibernate.Cfg.ConfigurationSectionHandler, NHibernate" />
  </configSections>

  <hibernate-configuration xmlns="urn:hibernate-configuration-5.2.7">
    <session-factory>
      <property name="connection.provider">
        NHibernate.Connection.DriverConnectionProvider
      </property>
      <property name="dialect">NHibernate.Dialect.MySQLDialect</property>
      <property name="query.substitutions">hqlFunction=SQLFUNC</property>
      <property name="connection.driver_class">
        NHibernate.Driver.MySqlDataDriver
      </property>
      <property name="connection.connection_string">
        server=127.0.0.1;port=3306;Uid=root;password=MySQL;database=uni
      </property>
      <property name="show_sql">true</property>
      <mapping assembly="hibernate_mysql_try" />
    </session-factory>
  </hibernate-configuration>

</configuration>
```


nHibernate Шаг 4



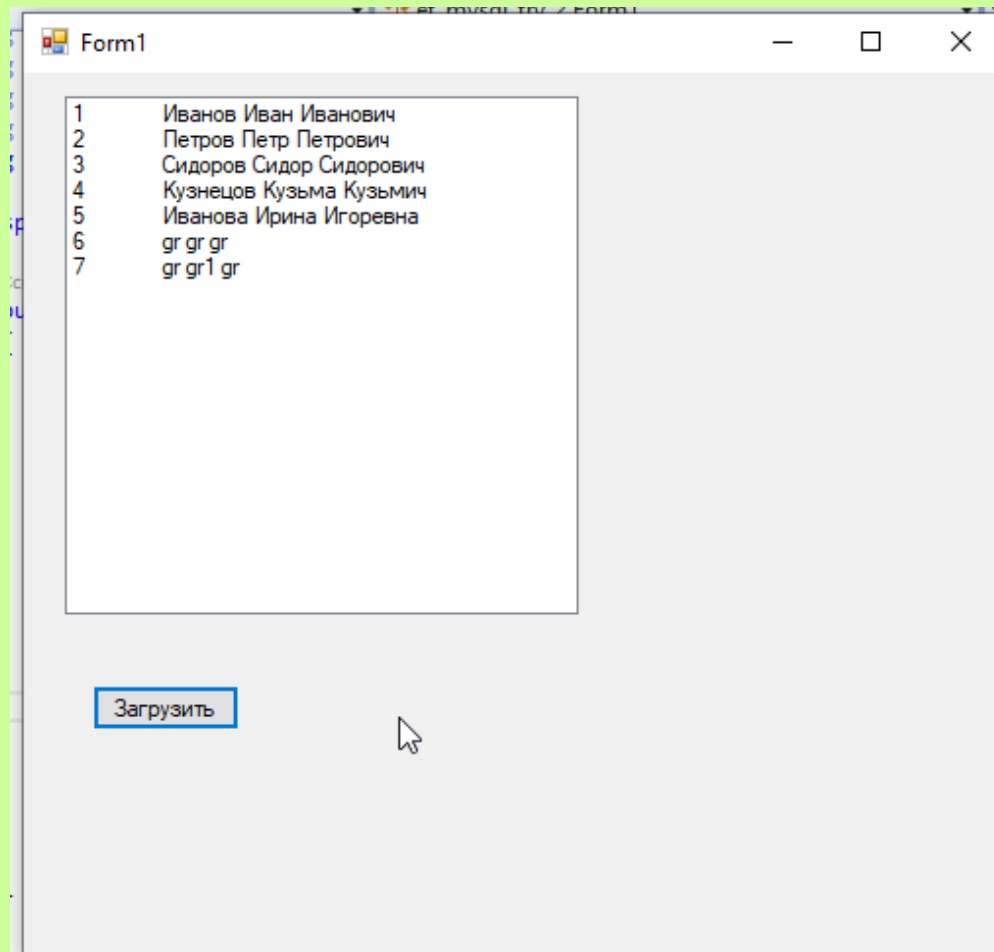
nHibernate War 4

ссылка: 1

```
private void Load_b_Click(object sender, EventArgs e)
{
    stud_lb.Items.Clear();
    sessionFactory = new Configuration().Configure().BuildSessionFactory();
    session = sessionFactory.OpenSession();
    using (var students_transaction = session.BeginTransaction())
    {
        var students = session.Query<student>().ToList();

        foreach (var st in students)
        {
            stud_lb.Items.Add(st.ID.ToString()+"\t"+st.surname+' '+st.name+' '+st.patronym);
        }
    }
}
```

nHibernate Шаг 4



Способы взаимодействия Entity Framework с БД

- **Database first:** Entity Framework создает набор классов, которые отражают модель конкретной базы данных
- **Model first:** сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
- **Code first:** разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы

Entity Framework Database first

по шагам

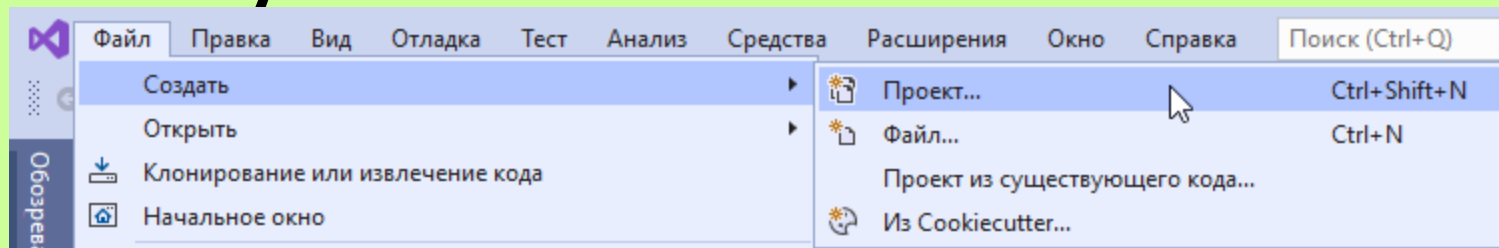
- MySQL 8 ,установлен первым
- Visual Studio 2019 Community

Data Provider ADO.NET

- SQL Server Connection
- OLEDB Connection
- ODBC Connection

Entity Framework Шаг1

В
свойствах
проекта



Настроить новый проект


Приложение Windows Forms (.NET Framework) C# Windows Рабочий стол

Имя проекта

EF_mysql_try

Расположение

\\localhost\ЛПБД\

Имя решения 

EF_mysql_try

☐ Поместить решение и проект в одном каталоге

Платформа

.NET Framework 4.7.2

Добавление нового элемента - EF_mysql_try

Установленные

Элементы Visual C#

- Windows Forms
- WPF
- Веб
- Данные
- Код
- Общие
- SQL Server
- Storm Items
- Workflow
- Графика

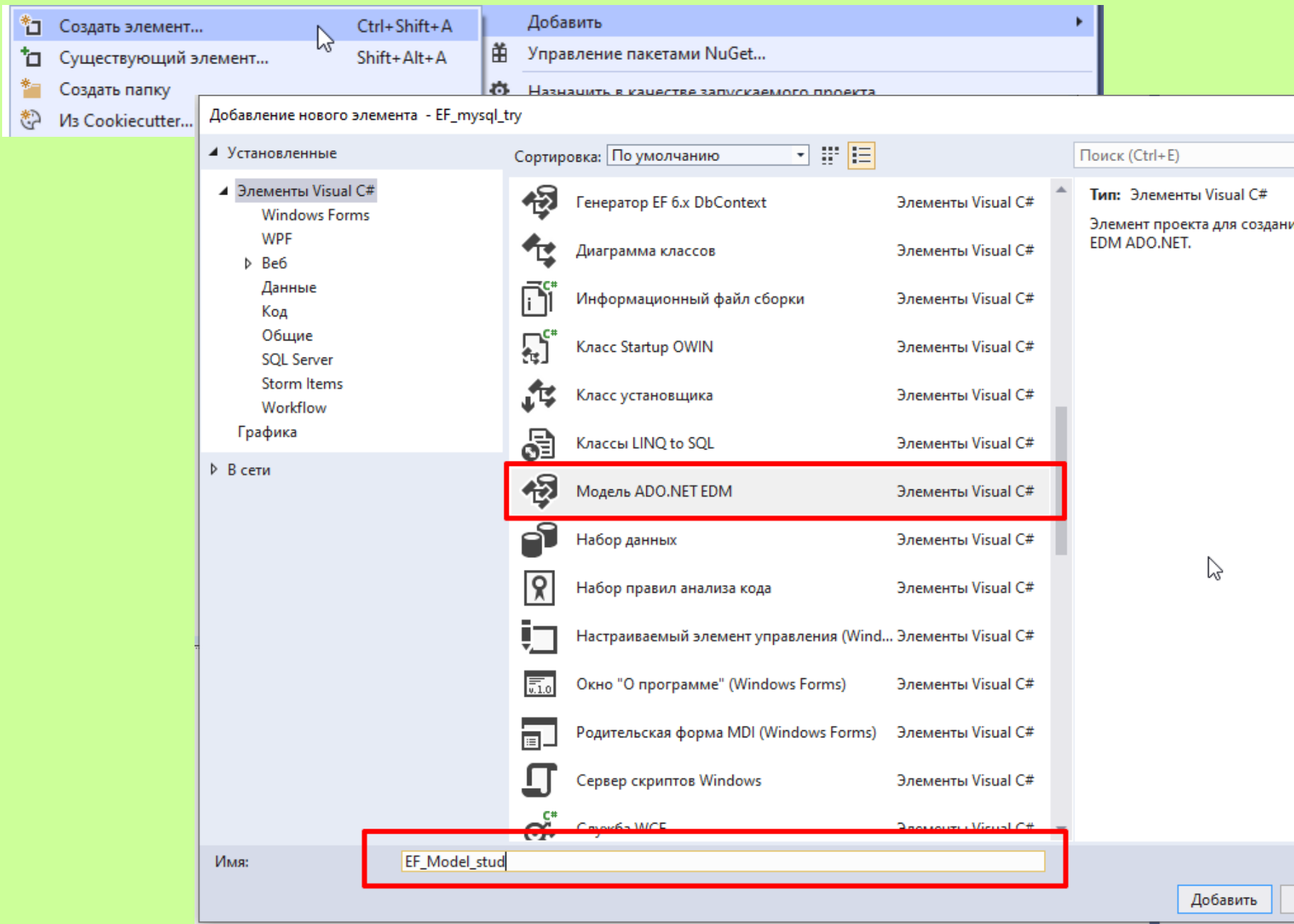
В сети

Сортировка: По умолчанию

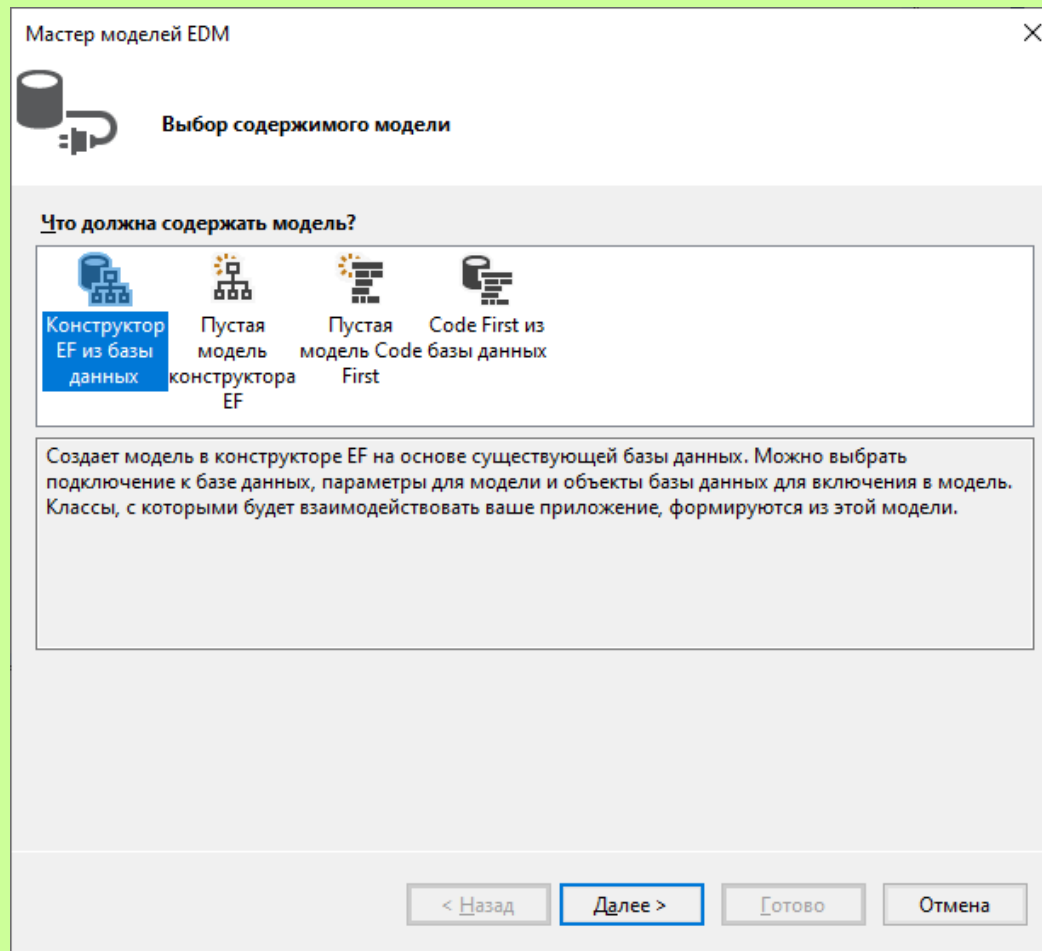
Иконка	Название элемента	Тип
	Генератор EF 6.x DbContext	Элементы Visual C#
	Диаграмма классов	Элементы Visual C#
	Информационный файл сборки	Элементы Visual C#
	Класс Startup OWIN	Элементы Visual C#
	Класс установщика	Элементы Visual C#
	Классы LINQ to SQL	Элементы Visual C#
	Модель ADO.NET EDM	Элементы Visual C#
	Набор данных	Элементы Visual C#
	Набор правил анализа кода	Элементы Visual C#
	Настраиваемый элемент управления (Windows Forms)	Элементы Visual C#
	Окно "О программе" (Windows Forms)	Элементы Visual C#
	Родительская форма MDI (Windows Forms)	Элементы Visual C#
	Сервер скриптов Windows	Элементы Visual C#
	Служба WCF	Элементы Visual C#

Имя: EF_Model_stud

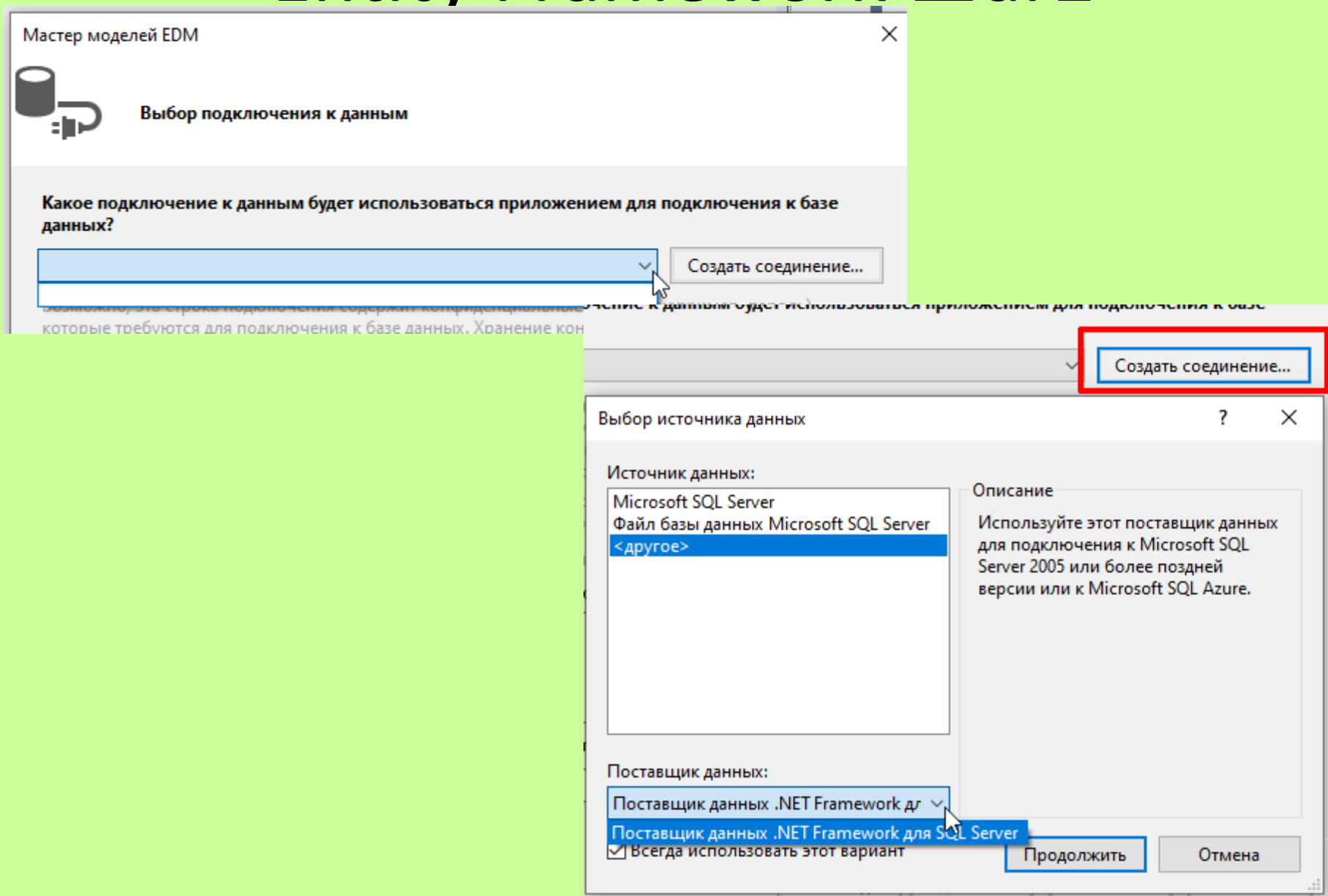
Добавить



Entity Framework Шаг2

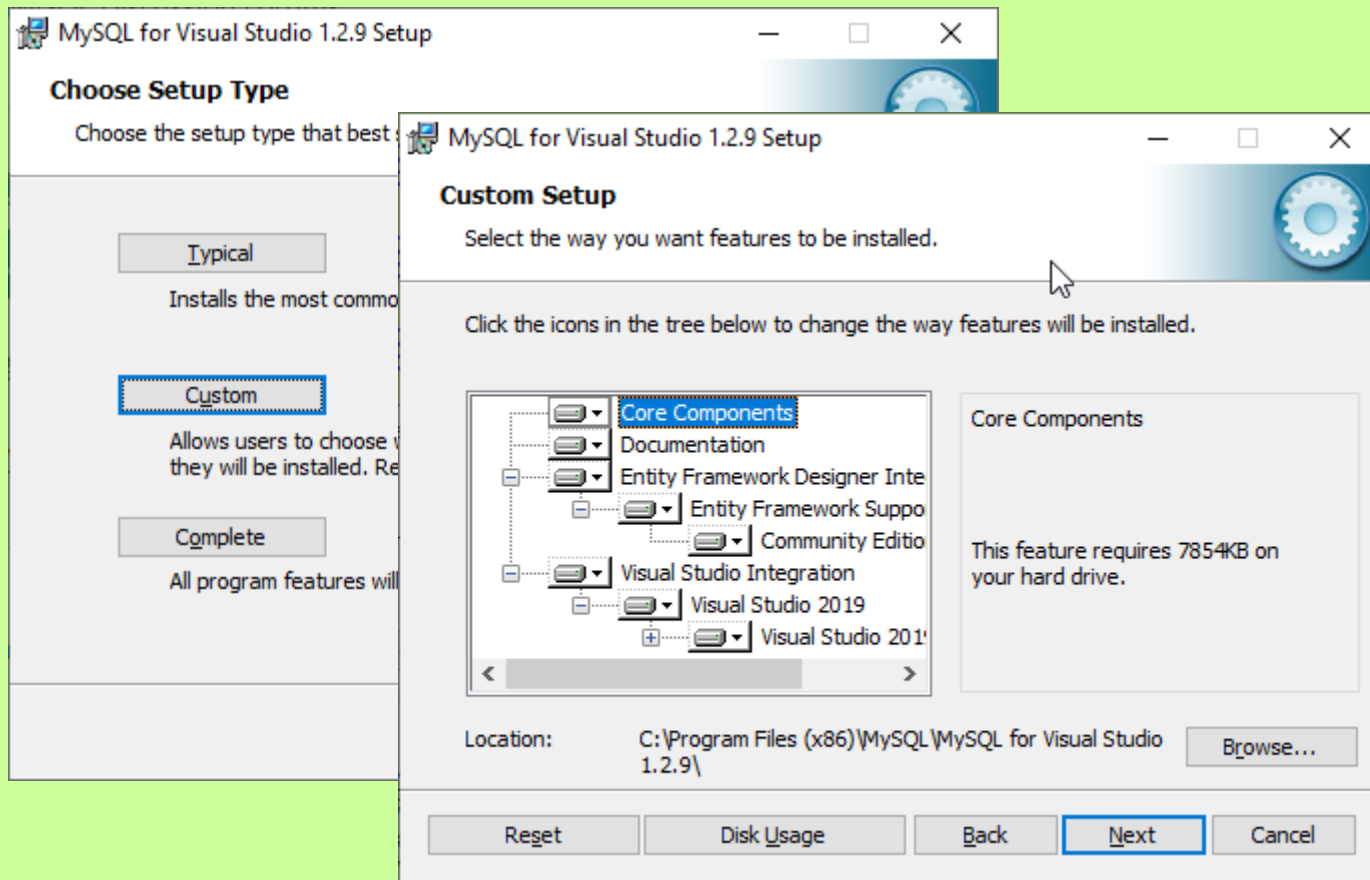


Entity Framework Шаг2

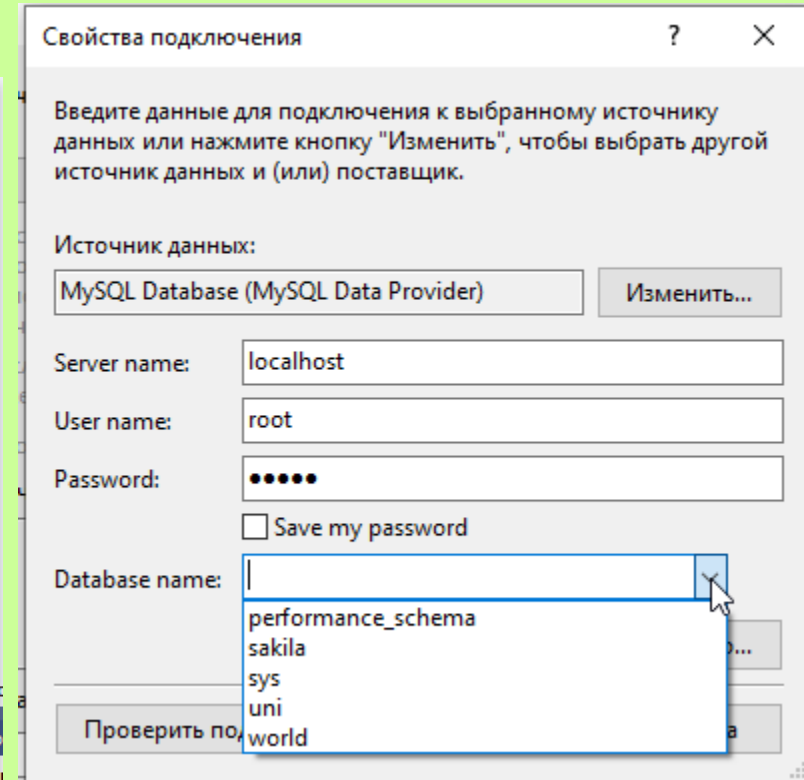
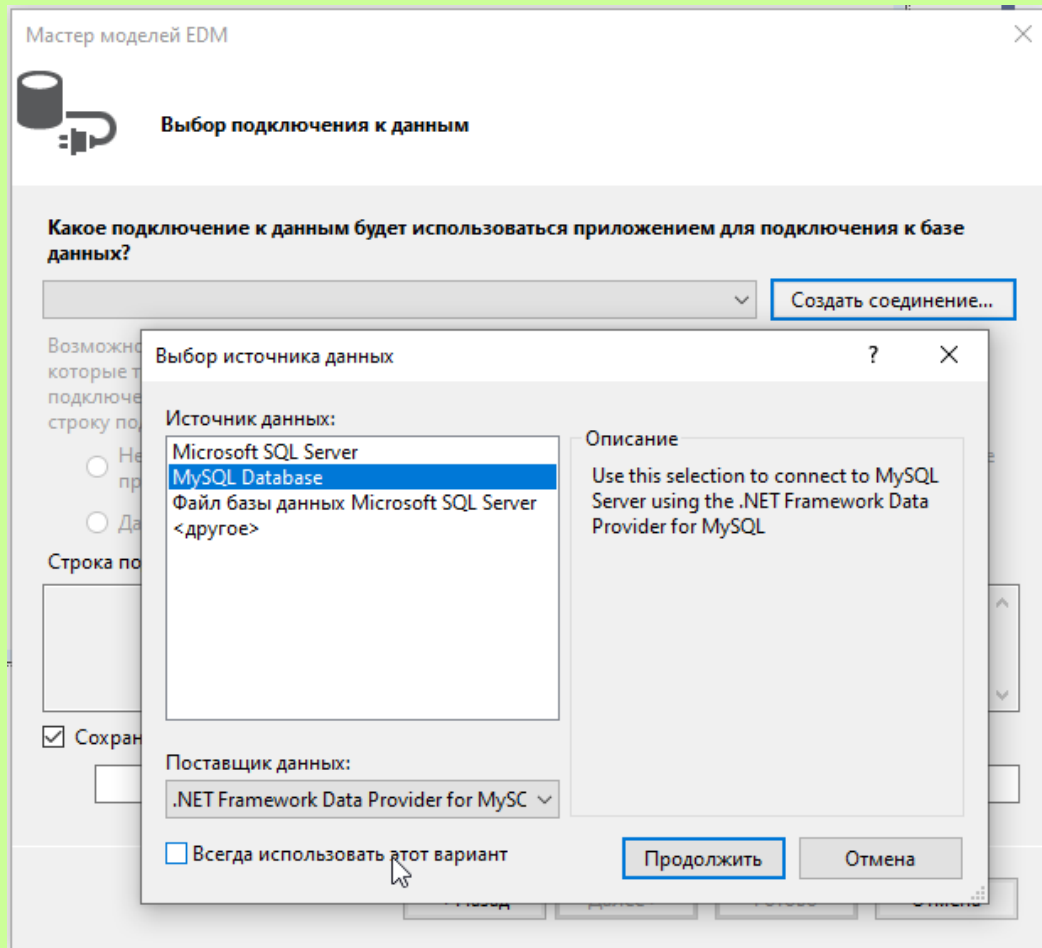


MySQL Connectors

- <https://dev.mysql.com/downloads/>
- <https://dev.mysql.com/downloads/connector/net/>
- <https://dev.mysql.com/downloads/windows/visualstudio/>




Entity Framework Шаг2



Entity Framework Шаг2

Мастер моделей EDM

 Выбор подключения к данным

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

localhost(uni) 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. Ссылка будет содержать маску.

☒ Да, включить конфиденциальные данные в строку подключения.

Строка подключения:


```
metadata=res://*/ef_Model_stud.csd|res://*/ef_Model_stud.ssd|res://*/ef_Model_stud.msl;provider=MySQL.Data.MySqlClient;provider connection string="server=localhost;user id=root;database=uni"
```

☒ Сохранить параметры соединения в App.Config как:

uniEntities

< Назад Далее > Go

Мастер моделей EDM

 Выберите версию

Какую версию Entity Framework вы хотите использовать?

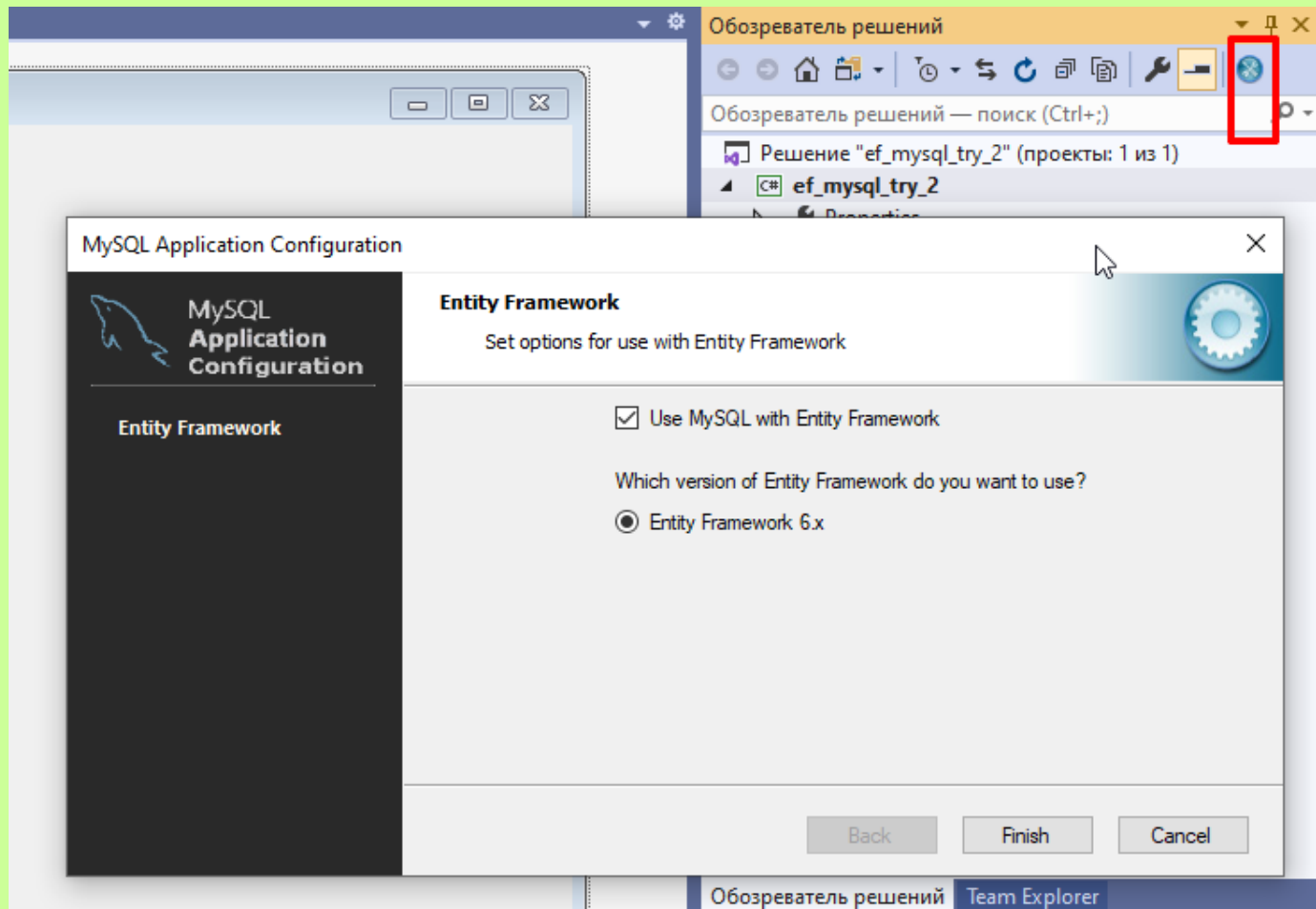
☐ Entity Framework 6.x

☒ Entity Framework 5.0

i Не удалось найти поставщик базы данных Entity Framework, совместимый с последней версией Entity Framework, для вашего подключения к данным. Если вы уже установили совместимый поставщик, то перед выполнением этого действия обязательно перестройте свой проект. В противном случае закройте мастер, установите совместимый поставщик, перестройте проект и только затем выполняйте это действие.

[Получить дополнительные сведения об этом](#)

Entity Framework Шаг2



Entity Framework Шаг2

Мастер моделей EDM

Выберите параметры и объекты базы данных

Какие объекты базы данных нужно включить в модель?

- ☒ Таблицы
 - ☒ uni
 - ☐ cas_activities
 - ☐ st_group
 - ☒ student
 - ☐ student_activity
 - ☐ trigger_test
 - ☐ Представления
 - ☐ Хранимые процедуры и функции

☒ Формировать имена объектов во множественном или единственном числе

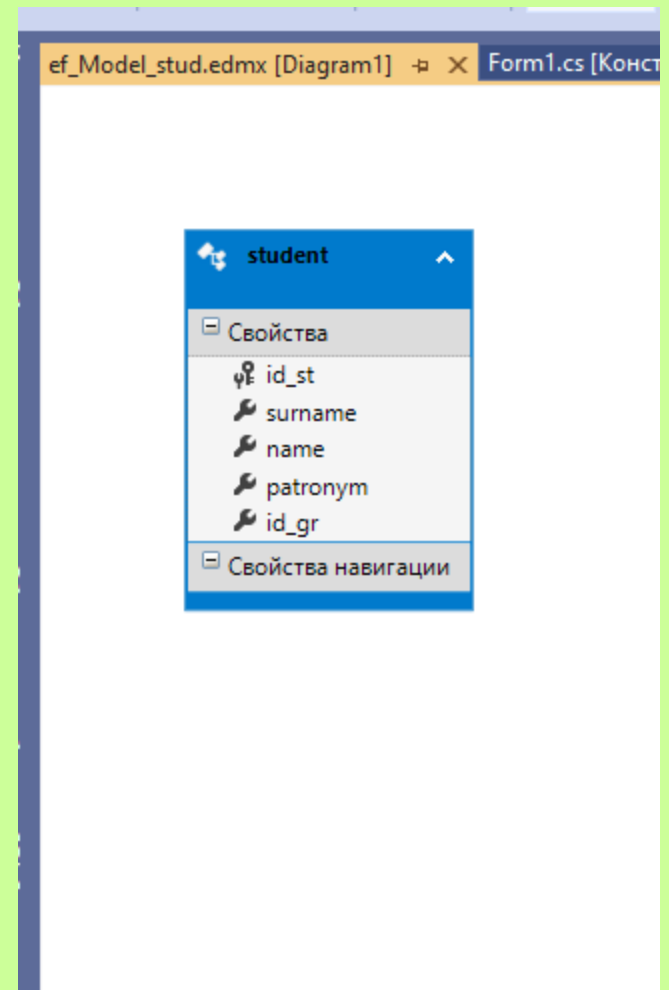
☒ Включить столбцы внешних ключей в модель

☐ Импортировать выбранные хранимые процедуры и функции в модель сущностей

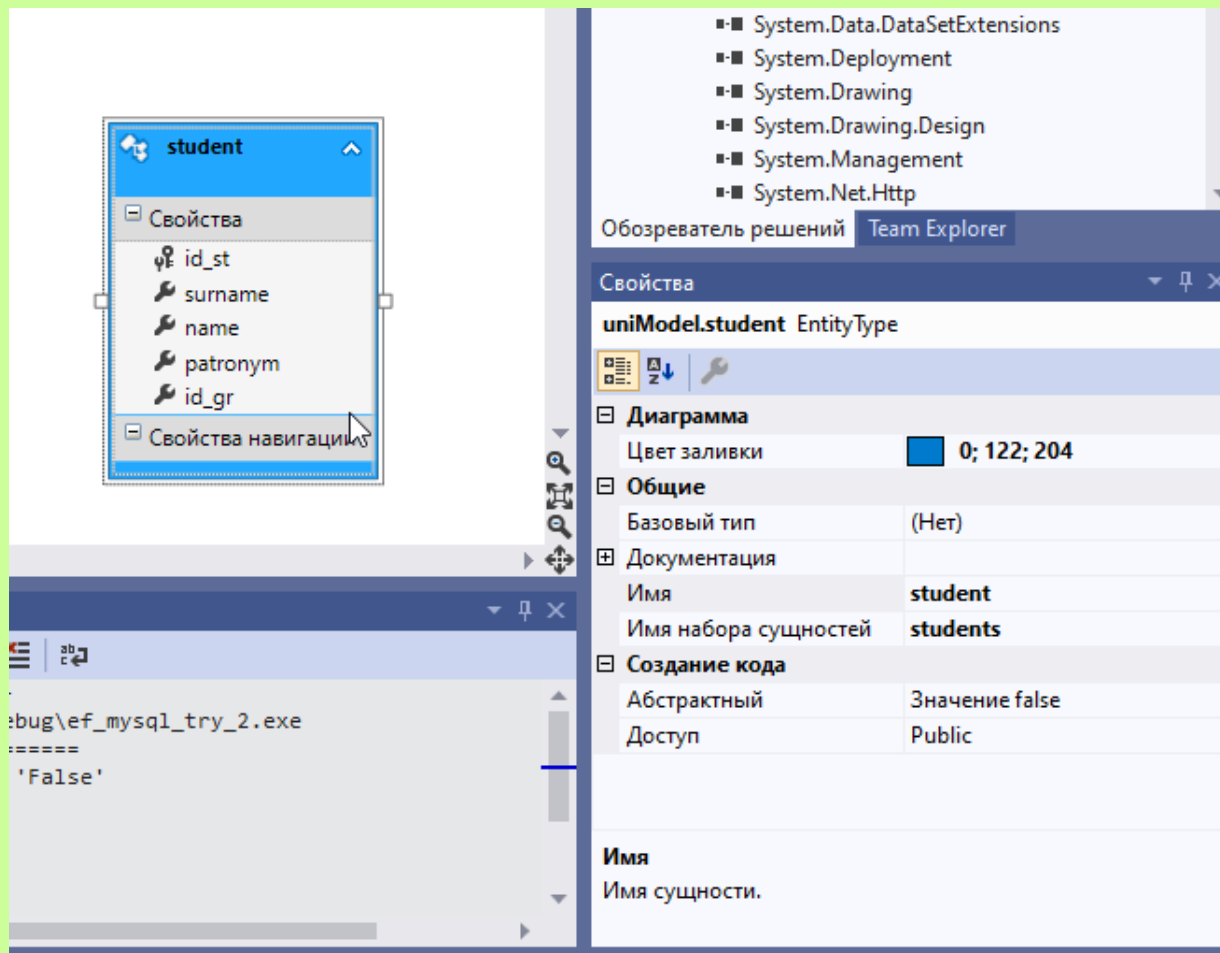
Пространство имен модели:

uniModel

< Назад Далее > Готово Отмена



Entity Framework Шаг 3



Entity Framework Шаг 3

The screenshot displays the Visual Studio IDE with two main windows. The left window, titled 'ef_mysql_try_2', shows the code for the 'student' entity in the 'ef_mysql_try_2' namespace. The code is auto-generated and includes comments in Russian. The right window, titled 'Обозреватель решений' (Solution Explorer), shows the project structure for 'ef_mysql_try_2'.

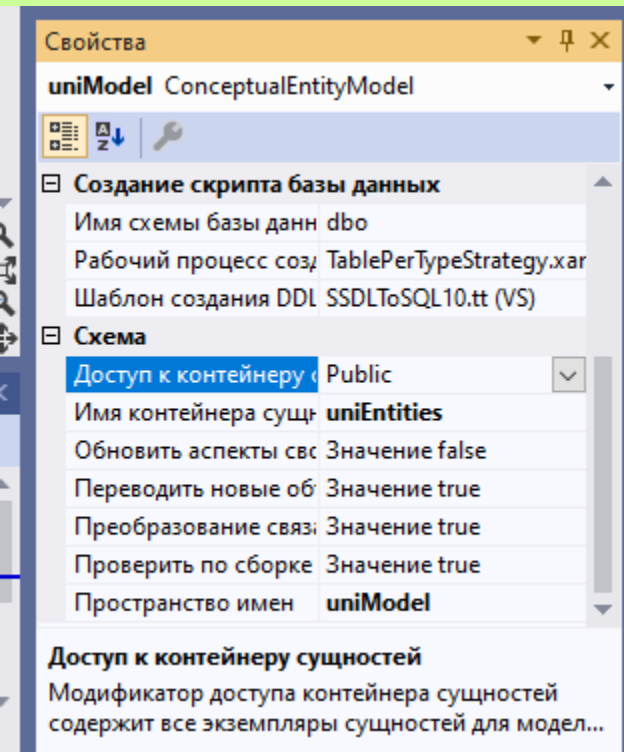
Code in ef_mysql_try_2.student:

```
1 //-----  
2 // <auto-generated>  
3 //     Этот код создан по шаблону.  
4 //  
5 //     Изменения, вносимые в этот файл вручную, могут  
6 //     Изменения, вносимые в этот файл вручную, будут  
7 // </auto-generated>  
8 //-----  
9  
10 namespace ef_mysql_try_2  
11 {  
12     using System;  
13     using System.Collections.Generic;  
14  
15     ссылка: 1  
16     public partial class student  
17     {  
18         ссылка: 0  
19         public int id_st { get; set; }  
20         ссылка: 0  
21         public string surname { get; set; }  
22         ссылка: 0  
23         public string name { get; set; }  
24         ссылка: 0  
25         public string patronym { get; set; }  
26         ссылка: 0  
27         public Nullable<int> id_gr { get; set; }  
28     }  
29 }
```

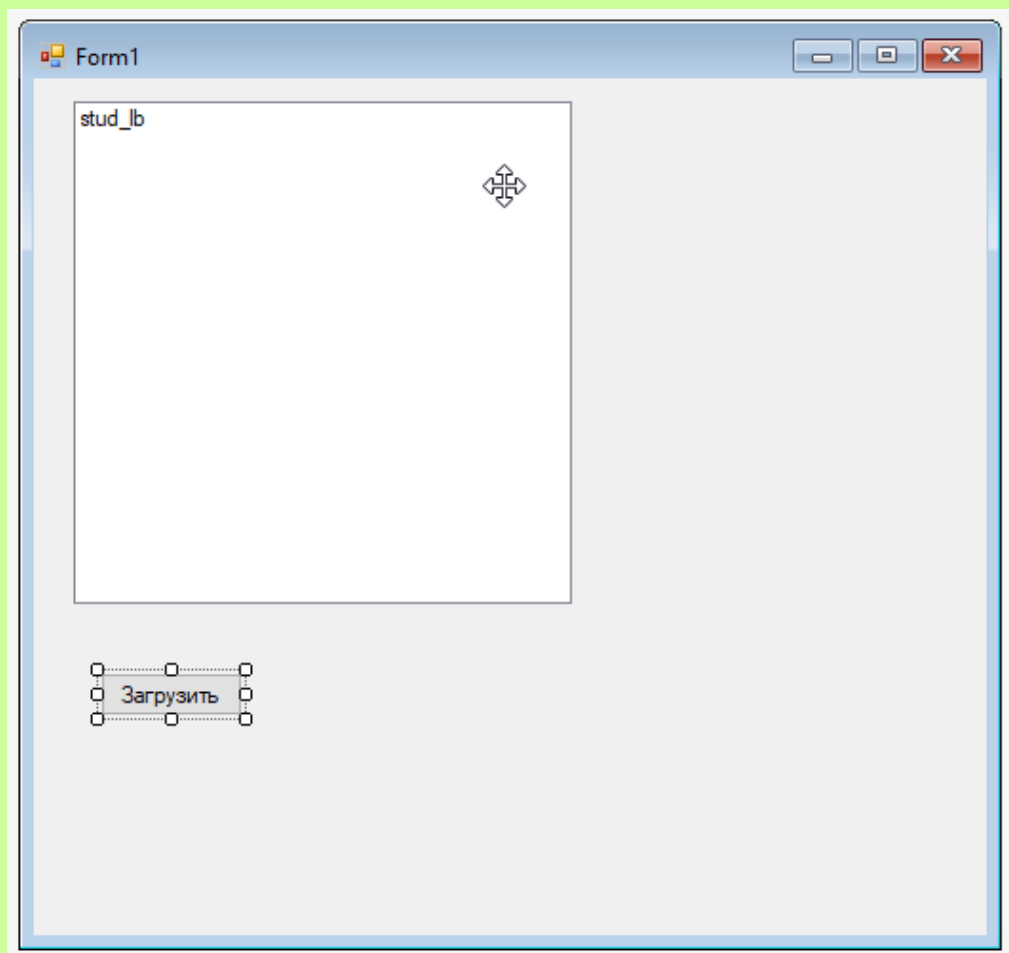
Solution Explorer Structure:

- Решение "ef_mysql_try_2" (проекты: 1 из 1)
 - ef_mysql_try_2
 - Properties
 - Ссылки
 - App.config
 - ef_Model_stud.edmx
 - ef_Model_stud.Context.tt
 - ef_Model_stud.Designer.cs
 - ef_Model_stud.edmx.diagram
 - ef_Model_stud.tt
 - ef_Model_stud.cs
 - student.cs
 - Form1.cs
 - packages.config
 - Program.cs

Entity Framework Шаг 3



Entity Framework Шаг 4



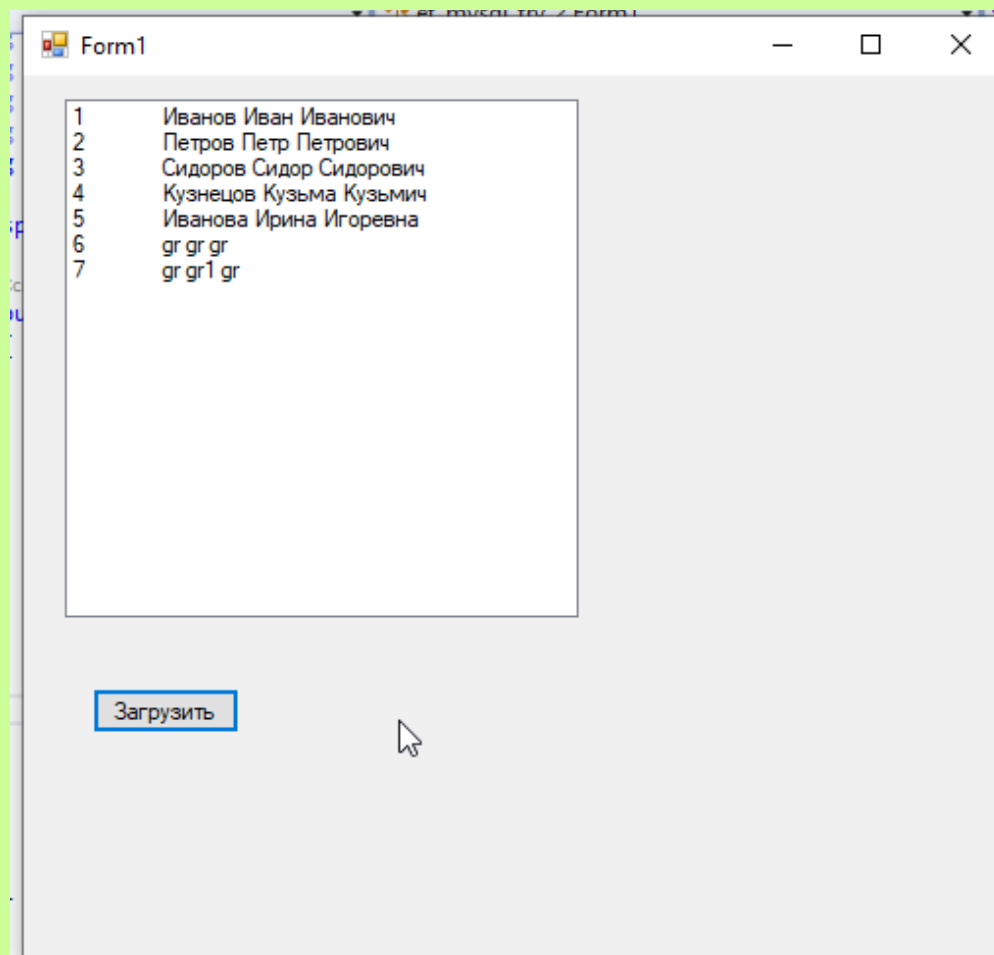
Entity Framework War 4

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ef_mysql_try_2
{
    Ссылка: 3
    public partial class Form1 : Form
    {
        ссылка: 1
        public Form1()
        {
            InitializeComponent();
        }

        ссылка: 1
        private void Load_button_Click(object sender, EventArgs e)
        {
            stud_lb.Items.Clear();
            using (uniEntities db = new uniEntities())
            {
                var students = db.students;
                foreach (student s in students)
                {
                    stud_lb.Items.Add(s.id_st.ToString() + '\t' + s.surname + ' ' + s.name + ' ' + s.patronym);
                }
            }
        }
    }
}
```

Entity Framework Шаг



ADO.NET

- ADO.NET (ActiveX Data Object для **.NET**) — это набор классов, предоставляющих службы доступа к данным программистам, которые используют платформу .NET Framework. ADO.NET имеет богатый набор компонентов для создания распределенных приложений, совместно использующих данные. Это неотъемлемая часть платформы .NET Framework, которая предоставляет доступ к реляционным данным, XML-данным и данным приложений.

Архитектура ADO.NET



Подключенные классы

Object	SQL Server	OLE DB	ODBC
Connection	SqlConnection	OleDbConnection	OdbcConnection
Command	SqlCommand	OleDbCommand	OdbcCommand
Data reader	SqlDataReader	OleDbDataReader	OdbcDataReader
Data adapter	SqlDataAdapter	OleDbDataAdapter	OdbcDataAdapter

ADO.Net по шагам

- MySQL 8 ,установлен первым
- Visual Studio 2019 Community
- MySQL для Visual Studio 1.2.9

ADO.Net шаг 1

Настроить новый проект

Приложение Windows Forms (.NET Framework)

C#

Windows


Рабочий стол

Имя проекта

ado_net_try

Расположение

F:\E\Надежда\Универ\ПБД\

Имя решения 

ado_net_try

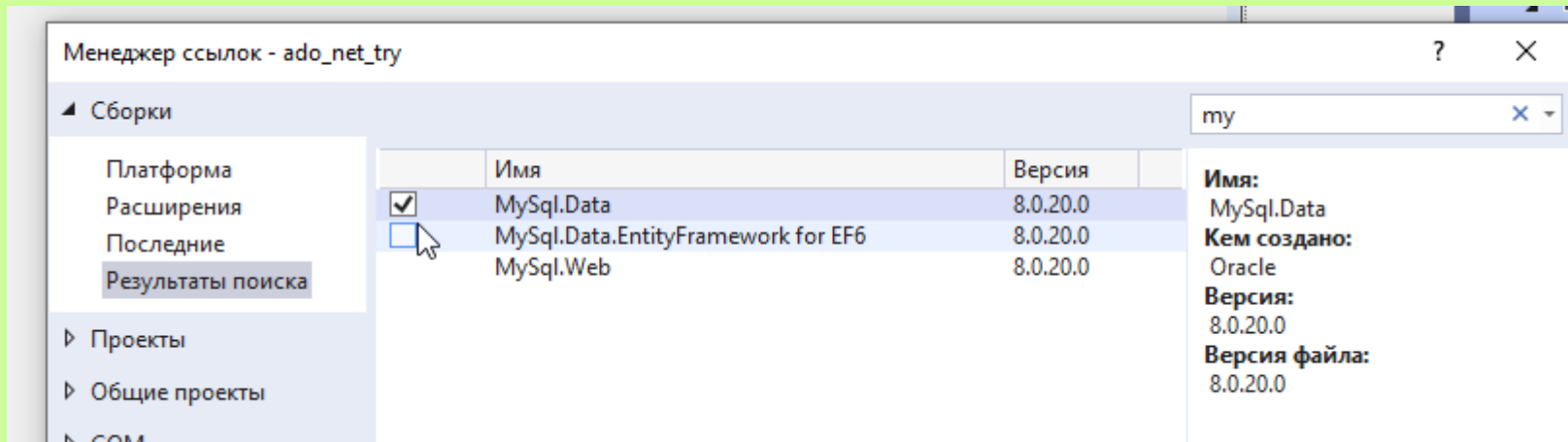
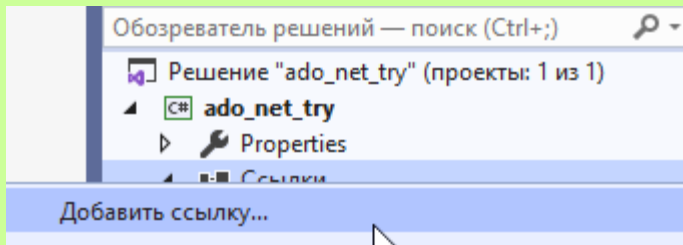
☐

Поместить решение и проект в одном каталоге

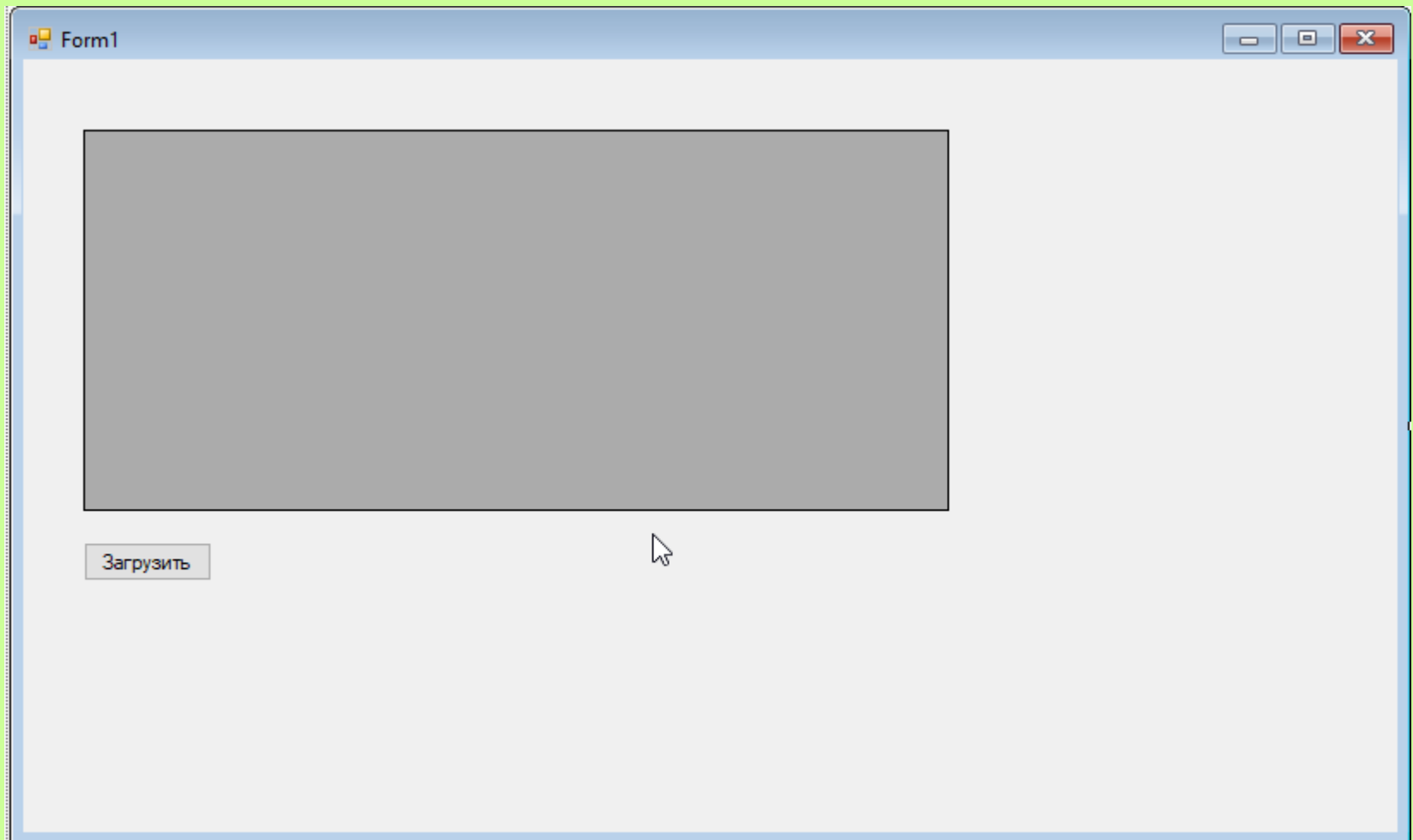
Платформа

.NET Framework 4.7.2

ADO.Net шаг 1



ADO.Net шаг 2



ADO.Net шаг 2

```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10 :
11 using MySql.Data.MySqlClient;
12
```

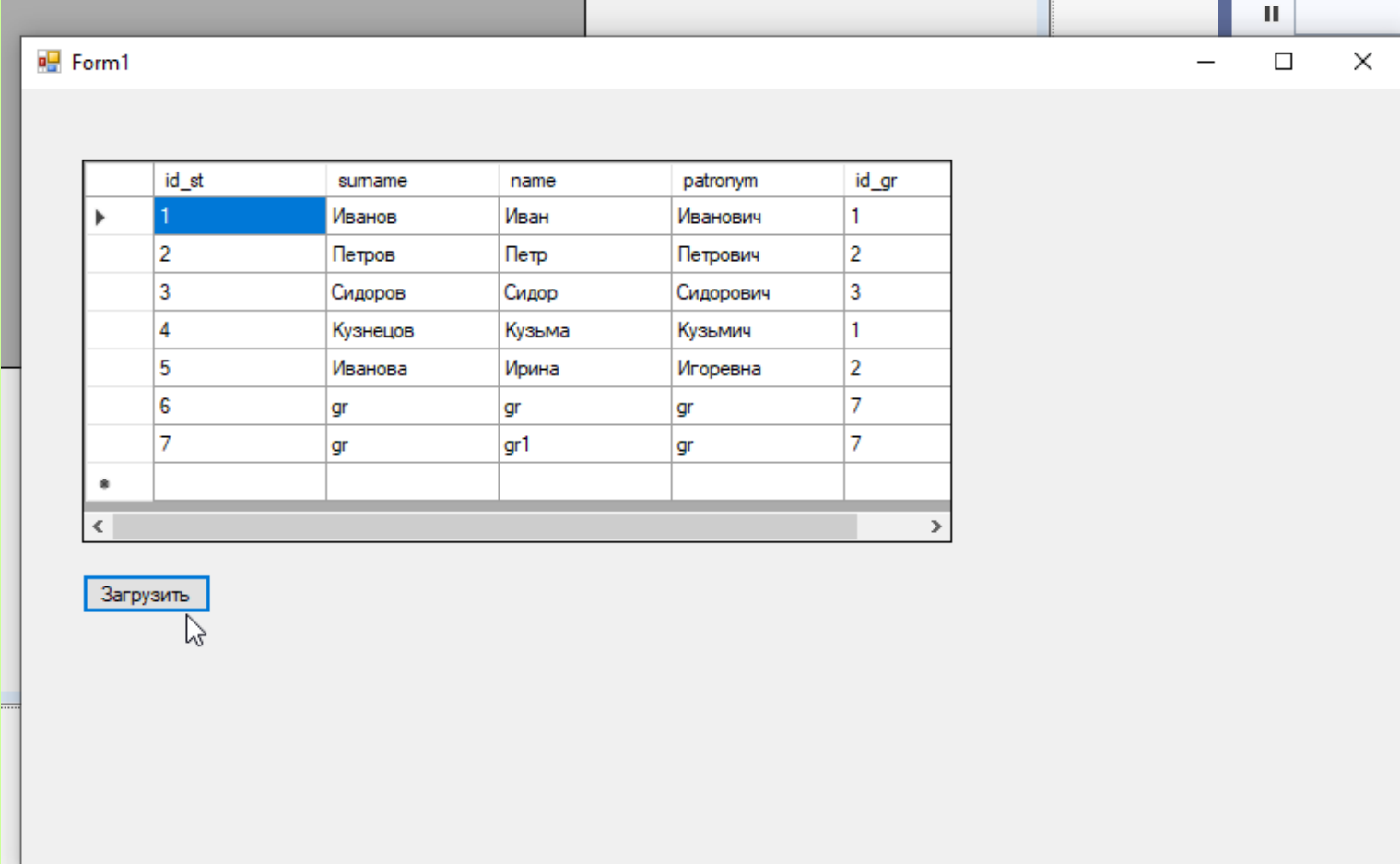
ADO.Net шаг 2

```
namespace ado_net_try
{
    ссылка: 3
    public partial class Form1 : Form
    {
        ссылка: 1
        public Form1()
        {
            InitializeComponent();
        }

        ссылка: 1
        private void Load_b_Click(object sender, EventArgs e)
        {
            MySqlConnectionStringBuilder mysqlCSB;
            mysqlCSB = new MySqlConnectionStringBuilder();
            mysqlCSB.Server = "127.0.0.1";
            mysqlCSB.Database = "uni";
            mysqlCSB.UserID = "root"; mysqlCSB.Password = "MySQL";
            DataTable dt = new DataTable();

            string queryString = @"SELECT * from student";
            using (MySqlConnection con = new MySqlConnection())
            {
                con.ConnectionString = mysqlCSB.ConnectionString;
                MySqlCommand com = new MySqlCommand(queryString, con);
                try
                {
                    con.Open();
                    using (MySqlDataReader dr = com.ExecuteReader())
                    {
                        if (dr.HasRows) { dt.Load(dr); }
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                con.Close();
                dataGridView1.DataSource = dt;
            }
        }
    }
}
```

ADO.Net итог



The screenshot shows a Windows application window titled "Form1". Inside the window, there is a data grid with 7 rows and 6 columns. The first row is selected. Below the grid, there is a button labeled "Загрузить" (Load) with a mouse cursor pointing at it.

	id_st	surname	name	patronym	id_gr
▶	1	Иванов	Иван	Иванович	1
	2	Петров	Петр	Петрович	2
	3	Сидоров	Сидор	Сидорович	3
	4	Кузнецов	Кузьма	Кузьмич	1
	5	Иванова	Ирина	Игоревна	2
	6	gr	gr	gr	7
	7	gr	gr1	gr	7
*					

Загрузить

ADO.Net с параметром

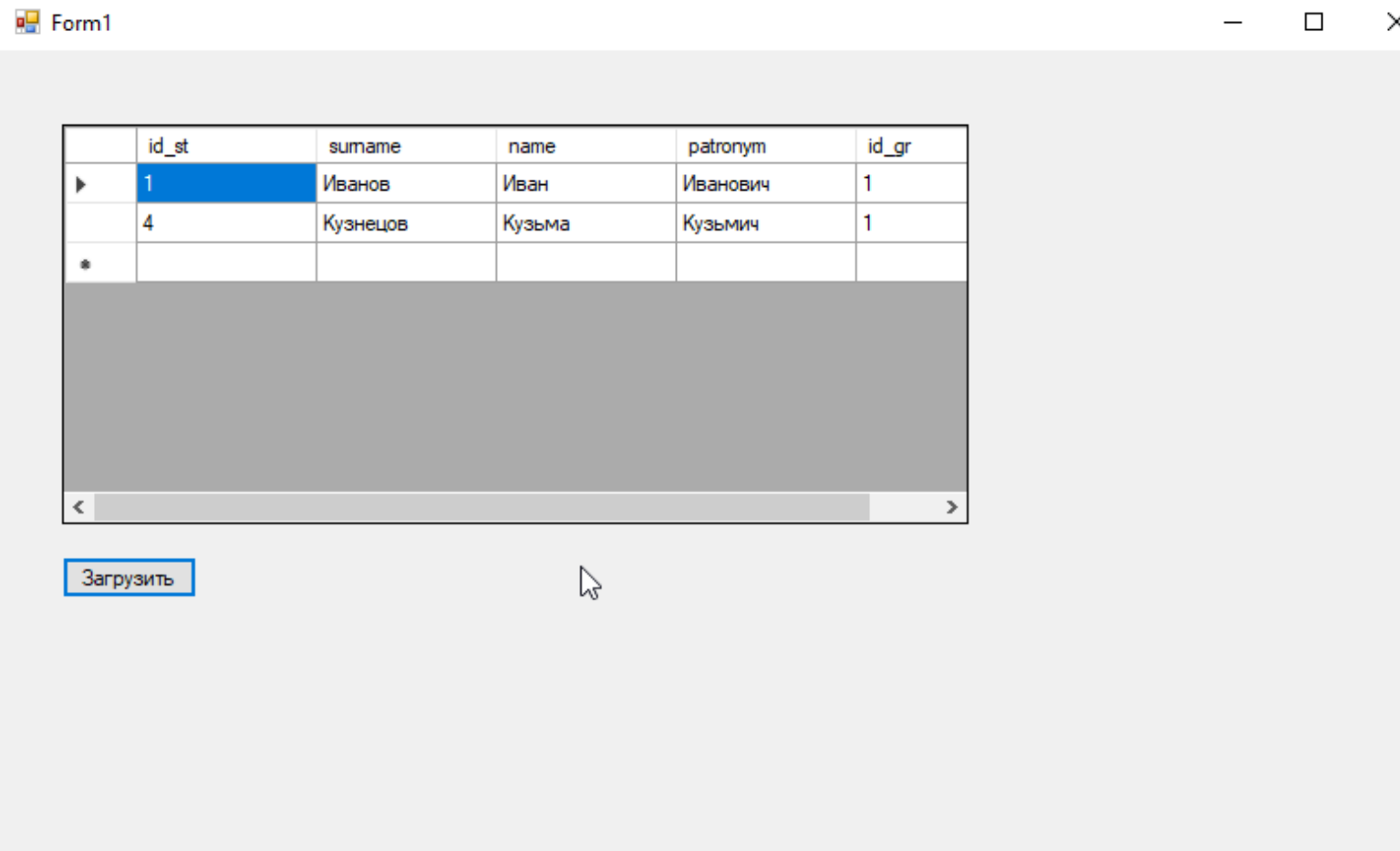
```
private void Load_b_Click(object sender, EventArgs e)
{
    MySqlConnectionStringBuilder mysqlCSB;
    mysqlCSB = new MySqlConnectionStringBuilder();
    mysqlCSB.Server = "127.0.0.1";
    mysqlCSB.Database = "uni";
    mysqlCSB.UserID = "root"; mysqlCSB.Password = "MySQL";
    DataTable dt = new DataTable();

    string queryString = @"SELECT * from student where id_gr=@id_g";
    using (MySqlConnection con = new MySqlConnection())
    {
        con.ConnectionString = mysqlCSB.ConnectionString;
        MySqlCommand com = new MySqlCommand(queryString, con);

        com.Parameters.AddWithValue("id_g", 1);

        try
        {
            con.Open();
            using (MySqlDataReader dr = com.ExecuteReader())
            {
                if (dr.HasRows) { dt.Load(dr); }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        con.Close();
        dataGridView1.DataSource = dt;
    }
}
```

ADO.Net с параметром



	id_st	sumame	name	patronym	id_gr
▶	1	Иванов	Иван	Иванович	1
	4	Кузнецов	Кузьма	Кузьмич	1
*					

Below the table is a large gray rectangular area and a horizontal scrollbar.

At the bottom left is a button labeled "Загрузить".

Подходы к визуализации

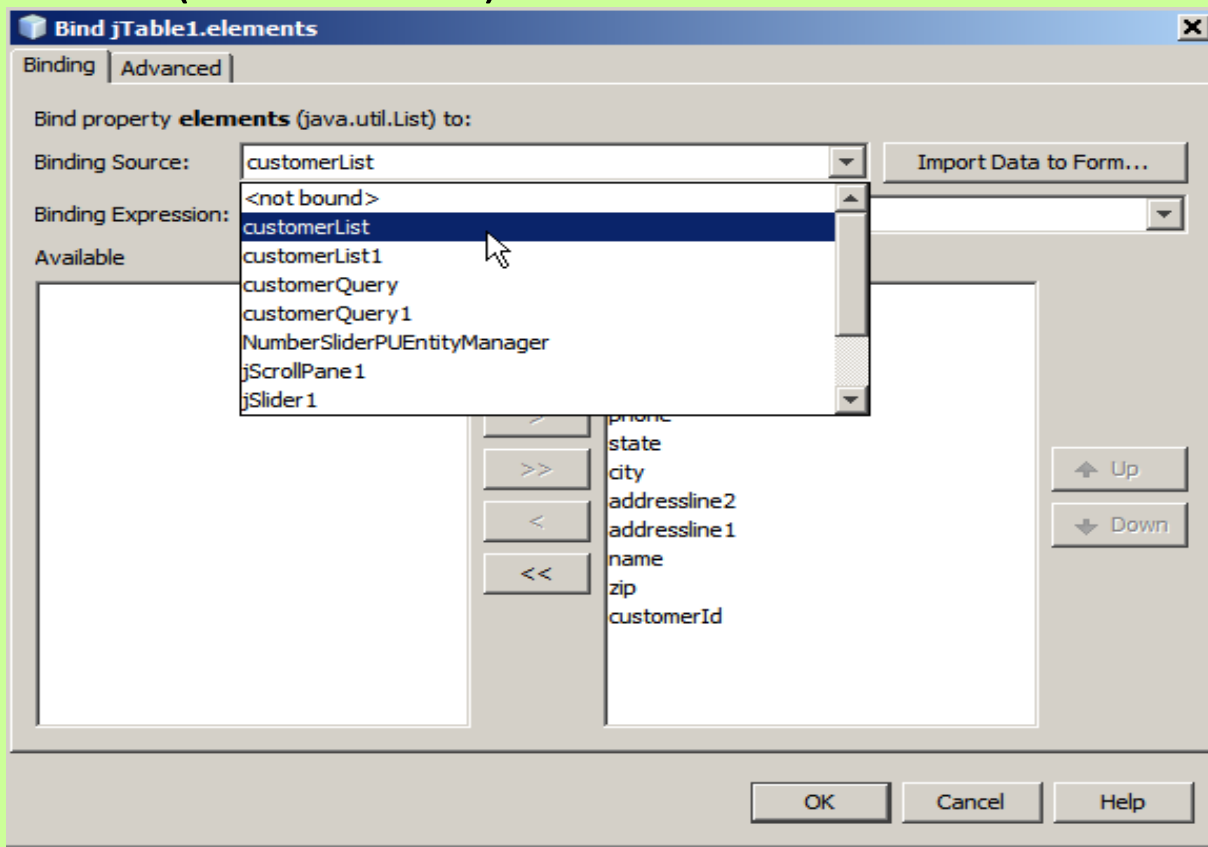
- «Интерактивный» интерфейс
- Разделение слоев и сохранение по сигналу

Способы отображения данных из базы в приложении

- Ручная отрисовка в задуманном программистом виде
- В стандартных компонентах/элементах интерфейса
- В Data Aware(DB Aware) компонентах

DB Aware в Java

- https://netbeans.org/kb/docs/java/gui-binding_ru.html
- Чтобы привязать таблицу базы данных к существующему компоненту JTable, выполните следующие действия.
- Щелкните правой кнопкой мыши компонент в конструкторе графических интерфейсов и выберите Bind ("Привязка") > elements ("элементы").



Создать проект

Последние шаблоны

Установленные шаблоны

Сортировать по: По умолчанию

Установленные шаблоны: поиск

Тип: Visual C#

Проект, для создания приложения с пользовательским интерфейсом Windows Forms

Приложение Windows Forms	Visual C#
Приложение WPF	Visual C#
Консольное приложение	Visual C#
Библиотека классов	Visual C#
Приложение обозревателя WPF	Visual C#
Библиотека настраиваемых элементов управления WPF	Visual C#
Библиотека пользовательских элементов управления...	Visual C#
Пустой проект	Visual C#
Служба Windows	Visual C#
Библиотека элементов управления Windows Forms	Visual C#

Имя: myApp1

Расположение: D:\E\Надежда\Универ\ПБД\

Имя решения: myApp1

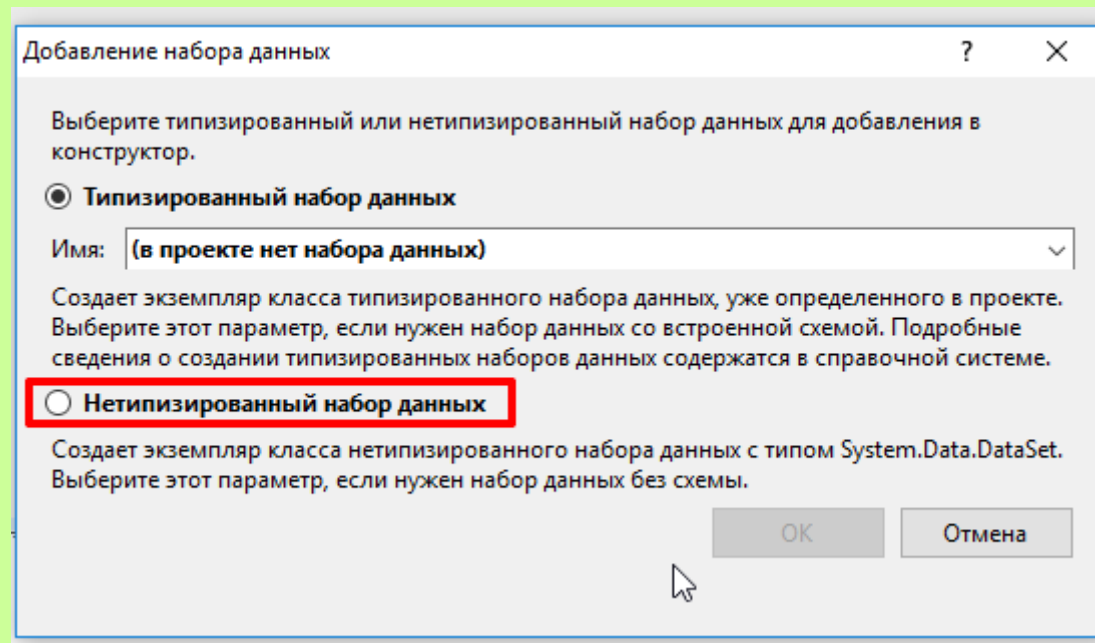
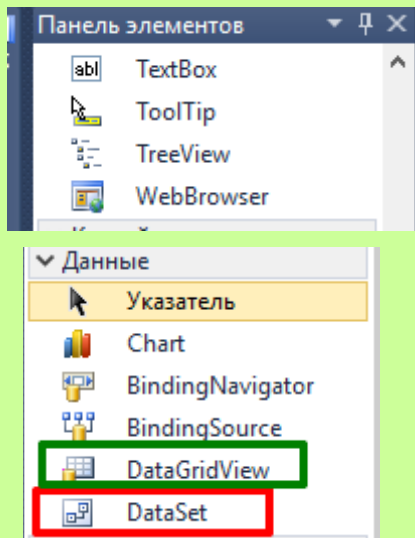
Обзор...

☒ Создать каталог для решения

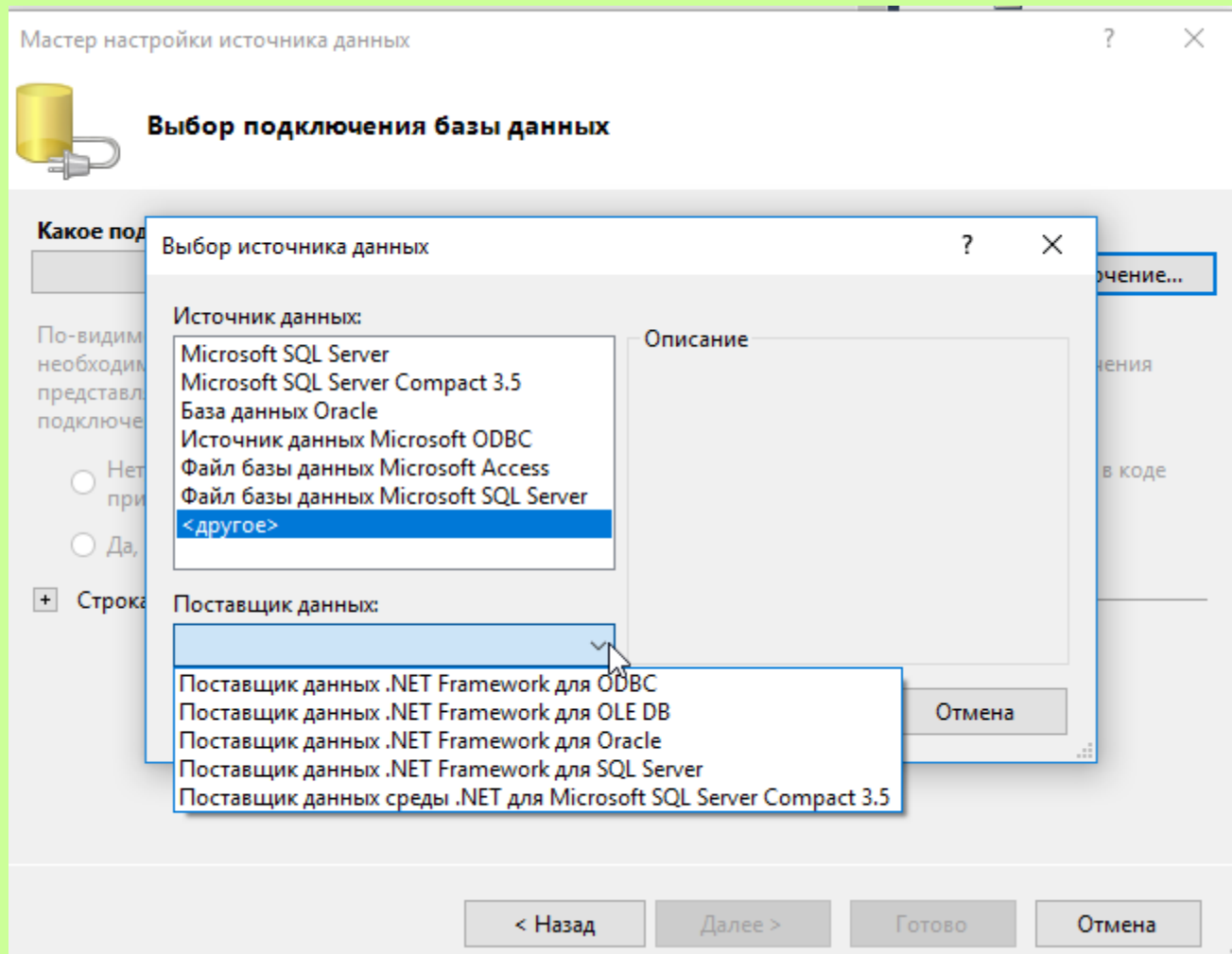
☐ Добавить в систему управления версиями

OK Отмена

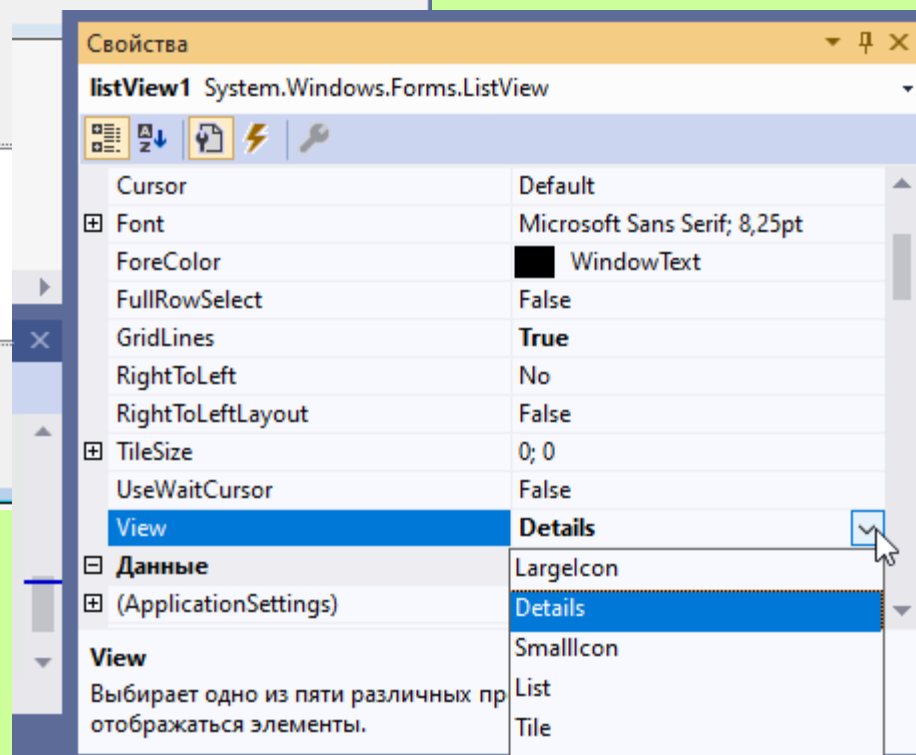
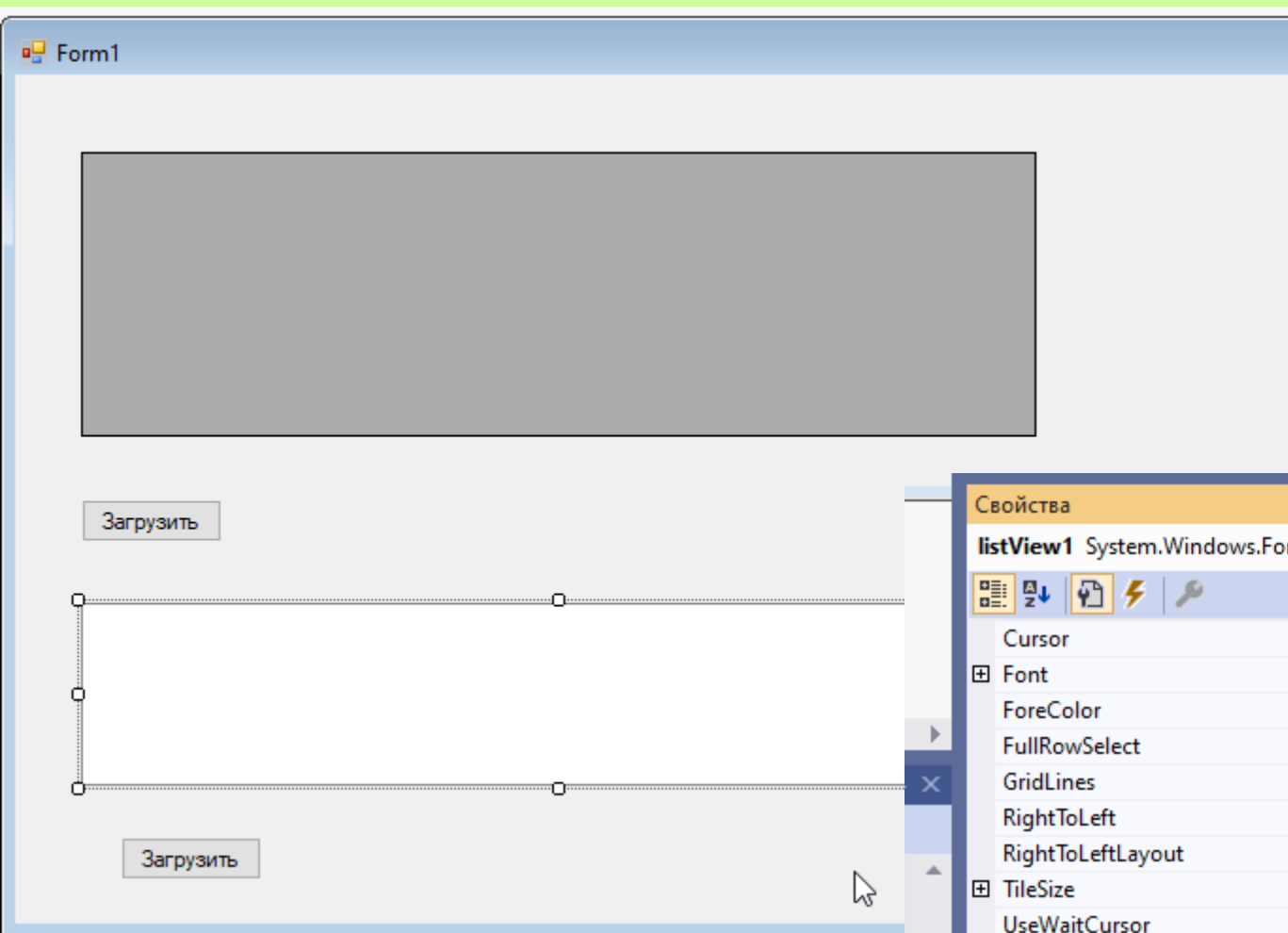
Data Aware C#



Источники данных в VS



DataGridView ListView



ListView отображение

```
private void Load_b2_Click(object sender, EventArgs e)
{
    MySqlConnectionStringBuilder mysqlCSB;
    mysqlCSB = new MySqlConnectionStringBuilder();
    mysqlCSB.Server = "127.0.0.1";
    mysqlCSB.Database = "uni";
    mysqlCSB.UserID = "root"; mysqlCSB.Password = "MySQL";

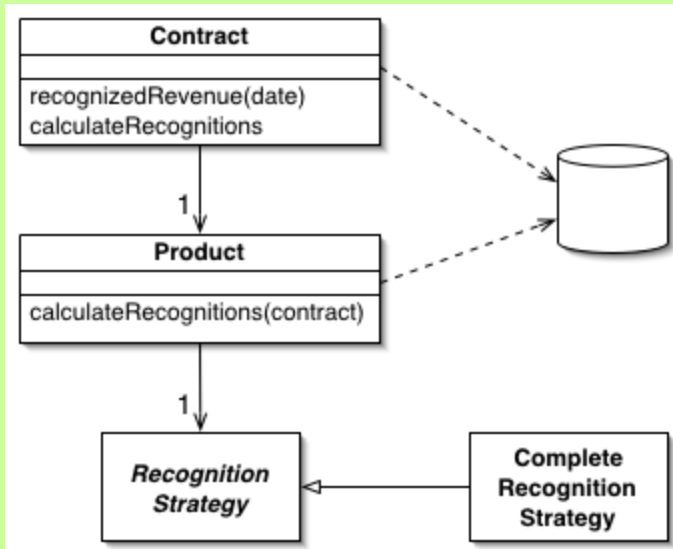
    string queryString = @"SELECT * from student where id_gr=@id_g";
    using (MySqlConnection con = new MySqlConnection())
    {
        con.ConnectionString = mysqlCSB.ConnectionString;
        MySqlCommand com = new MySqlCommand(queryString, con);
        com.Parameters.AddWithValue("id_g", 1);

        try
        {
            con.Open();
            using (MySqlDataReader dr = com.ExecuteReader())
            {
                listView1.Columns.Clear();
                if (dr.HasRows) {
                    for (int i=0;i<dr.FieldCount;i++)
                    {
                        listView1.Columns.Add(dr.GetName(i));
                    }
                }
                while (dr.Read())
                {
                    ListViewItem item = listView1.Items.Add(dr[0].ToString());
                    for (int i = 1; i < dr.FieldCount; i++)
                    {
                        item.SubItems.Add(dr[i].ToString());
                    }
                }
            }
        }
        catch (Exception ex) {MessageBox.Show(ex.Message);}
        con.Close();
    }
}
```


Паттерны проектирования при работе с БД

- Архитектура (моделирование поведения)
- Предметная область
- Связь с БД

Модель предметной области (Domain Model)



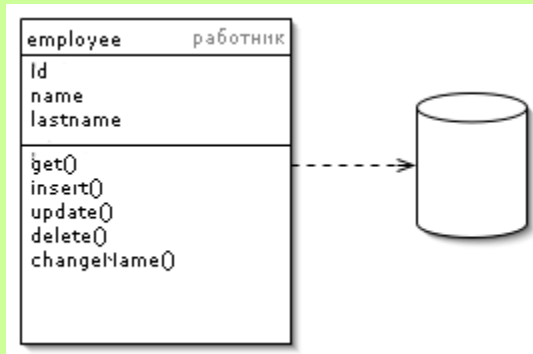
- Типовое решение **модель предметной области предусматривает создание сети**
- взаимосвязанных объектов, каждый из которых представляет некую осмысленную сущность— либо такую крупную, как промышленная корпорация, либо настолько мелкую, как строка формы заказа

Реализация модели предметной области означает пополнение приложения целым слоем объектов, описывающих различные стороны определенной области бизнеса. Одни объекты призваны имитировать элементы данных, которыми оперируют в этой области, а другие должны формализовать те или иные бизнес-правила. Функции тесно сочетаются с данными, которыми они манипулируют.

СВЯЗЬ С БД

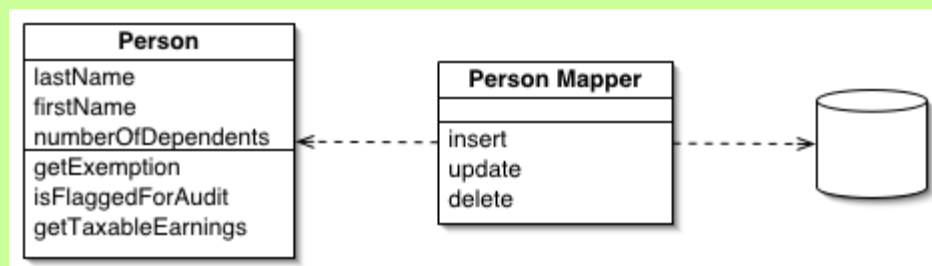
- *Active Record*
- *Data Mapper*
- *Table Data Gateway*

Active Record



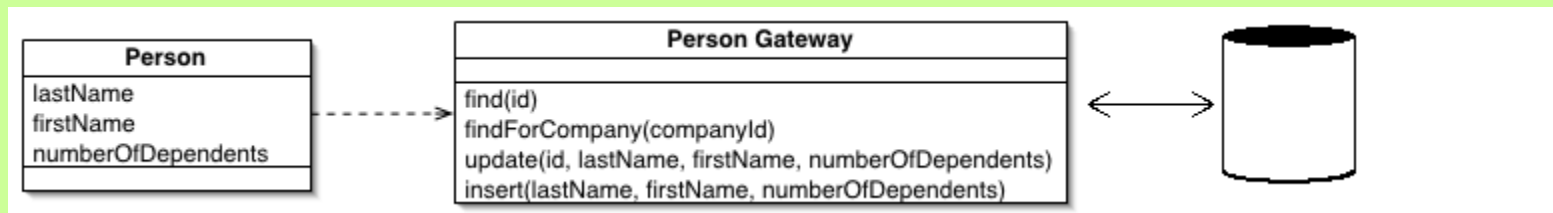
- Один объект управляет и данными, и поведением. Большинство этих данных постоянны и их надо хранить в БД. Этот паттерн использует наиболее очевидный подход - хранение логики доступа к данным в объекте сущности.
- Объект является "обёрткой" одной строки из БД или представления, включает в себя доступ к БД и логику обращения с данными.
- Пример: объект "Работник" содержит данные об одном работнике и методы: добавить, обновить или удалить. Помимо прочего, отдельным методом вынесена *смена имени*.

Data Mapper



- Data Mapper — это программная прослойка, разделяющая объект и БД. Его обязанность — пересылать данные между ними и изолировать их друг от друга. При использовании Data Mapper'а объекты не нуждаются в знании о существовании БД. Они не нуждаются в SQL-коде, и (естественно) в информации о структуре БД.

Table Data Gateway



- Объект выступает в качестве шлюза между данными в приложении и в БД. Один объект работает сразу со всеми записями в таблице.
- Объект шлюза к таблице содержит все запросы SQL для доступа к отдельной таблице или представлению (view): выборка, обновление, вставка, удаление (CRUD). Остальной код, для взаимодействия с БД, обращается к методам объекта шлюза.
- Пример: объект шлюза `PersonGateway` содержит методы для доступа к таблице `person` в БД. Методы содержат SQL-код для выборки, вставки, обновления и удаления. Объект может содержать специальную выборку, например *поиск по компании*.

. Как сохранить связи между объектами в базе данных?

- У меня есть ссылка на один объект — *отображение внешних ключей (Foreign Key Mapping, 258)*.
- У меня есть ссылка на коллекцию объектов — *отображение внешних ключей (Foreign Key Mapping, 258)*.
- У меня есть отношение типа "многие ко многим" — *отображение с помощью таблицы ассоциаций (Association Table Mapping, 269)*.
- У меня есть коллекция объектов, которые используются только в контексте другого объекта — *отображение зависимых объектов (Dependent Mapping, 283)*.
- У меня есть поле, в котором хранится объект-значение (Value Object, 500) — *внедренное значение (Embedded Value, 288)*.
- У меня есть сложная сеть объектов, которые не используются другими частями базы данных — *сериализованный крупный объект (SerializedLOB, 292)*.

LINQ - язык интегрированных запросов

- [LINQ to DataSet и SQL](#) LINQ to DataSet — название, данное API-интерфейсу LINQ, который предназначен для работы с DataSet. У многих разработчиков есть масса кода, полагающегося на DataSet. Те, кто не хотят отставать от новых веяний, но и не готовы переписывать свой код, благодаря этому интерфейсу могут воспользоваться всей мощью LINQ.
- LINQ to SQL — наименование, присвоенное API-интерфейсу IQueryable<T>, который позволяет запросам LINQ работать с базой данных Microsoft SQL Server. Чтобы воспользоваться преимуществами LINQ to SQL в проект понадобится добавить ссылку на сборку System.Data.Linq.dll, а также директиву using System.Data.Linq.
- **LINQ to Entities**
- [LINQ](#) --- LINQ to Entities
- API-интерфейс LINQ to Entities, полностью интегрированного в Entity Framework.
- **LINQ to MySQL**
- LINQ интерфейс MySQL