Лабораторная работа № 31

Тема: Разработка приложения по технологии Entity Framework.

Программное обеспечение: MS Visual Studio 2015

Цель: научиться создавать приложение Database First Entity Framework.

Время на выполнение: 2 часа.

Entity Framework упрощает разработку клиентского приложения, восполняя пробел между отношениями БД и объектами языка программирования. Возможность генерации любого из трех уровней клиент-серверного приложения на основе другого позволяет говорить о EF как о CASE-инструменте. Автоматизация разработки необходимых для взаимодействия с БД структур данных – сущностных классов – позволяет не только сэкономить усилия разработчика, но и упростить его задачу в области реализации GUI как визуального отображения данных.

Database First: Entity Framework создает набор классов, которые отражают модель конкретной существующей базы данных.

Рассмотрение реализации клиентского приложения, изложенное в данной лабораторной работе, подразумевает, что на стороне сервера все уже реализовано, так как процесс разворачивания EF в случае подхода Database First требует, прежде всего, подключения к базе данных.

Задание: Разработать клиентское приложение для работы с базой данных «Приемная комиссия» с помощью Database First Entity Framework.

Порядок выполнения:

1. Создайте новый проект Windows Forms.
2. Добавьте пакет NuGet, для этого следует перейти в пункт меню «Сервис \ Диспетчер пакетов NuGet \ Управление пакетами NuGet для решения». Затем установить Entity Framework (см. Рисунок 1).

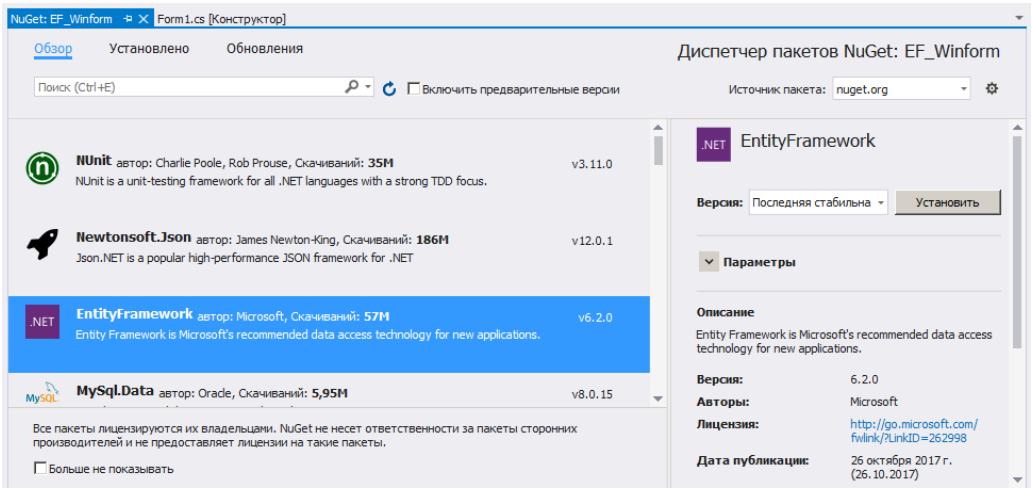


Рисунок 1 - Установка пакетов NuGet

3. После успешной установки необходимо собрать решение, нажав

Ctrl+Shift+B. После чего можно приступать к созданию модели EDM.

Создание EDM-модели

4. После добавления EF к проекту необходимо подключиться к существующей базе данных. Для этого нужно перейти в пункт меню «Проект \ Добавить компонент…» и в разделе «Данные» выбрать пункт «Модель ADO.NET EDM» (см. Рисунок 2).

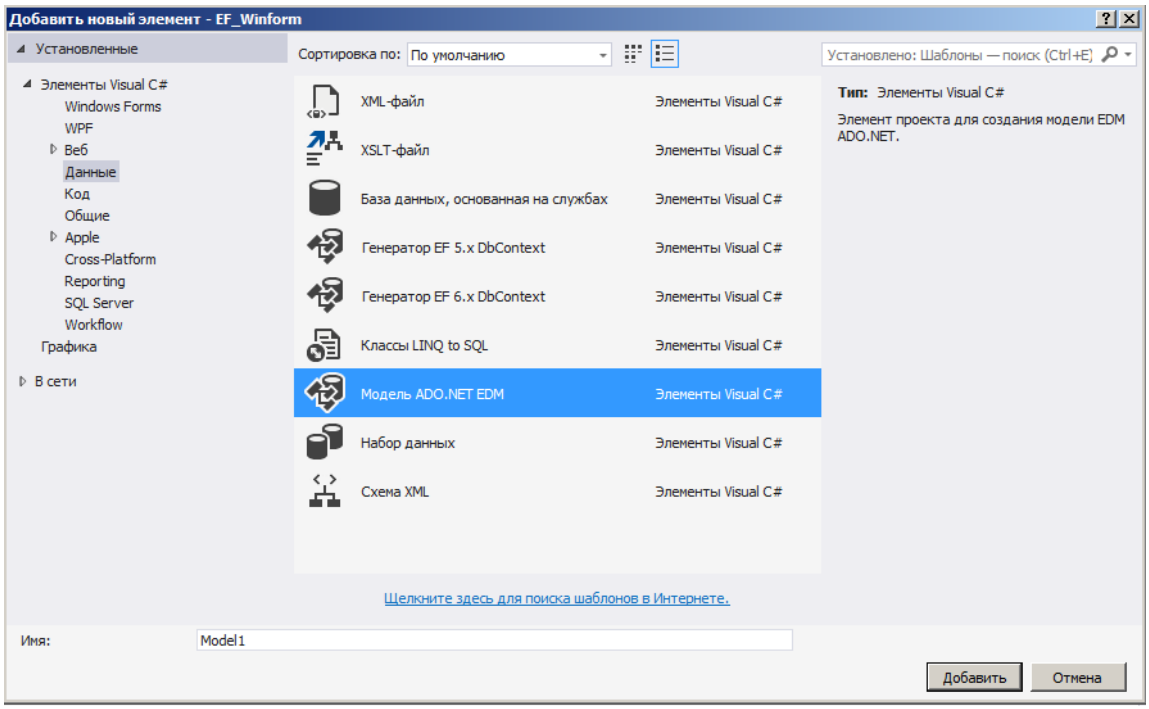


Рисунок 2 - Добавление EDM в проект

5. Так как уже есть существующая БД, то в следующем окне следует выбрать пункт «Создать из базы данных» (см. Рисунок 3):

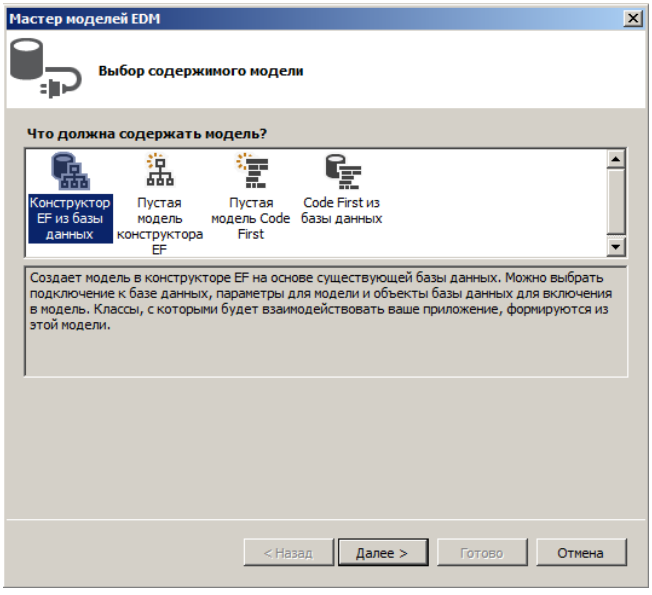


Рисунок 3 - Генерация модели из существующей БД

6. Далее при выборе подключения необходимо нажать кнопку «Создать

подключение» и выбрать поставщик данных для MS SQL Server (см. Рисунок 4).

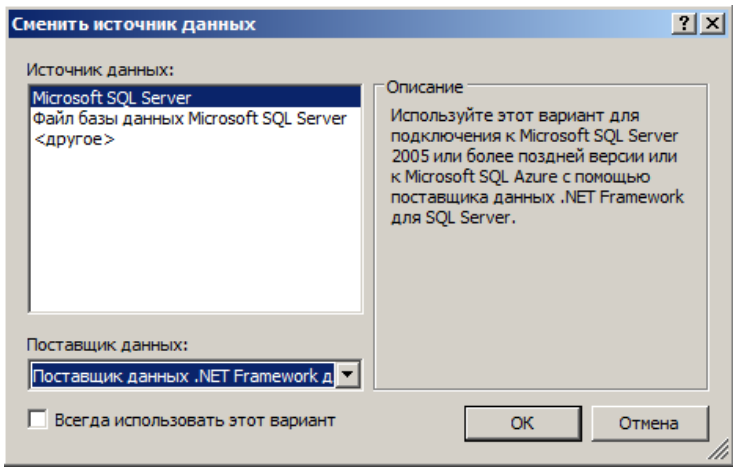


Рисунок 4 - Задание поставщика данных

7. В свойствах подключения укажите учебный сервер и Вашу базу данных (см. Рисунок 5).

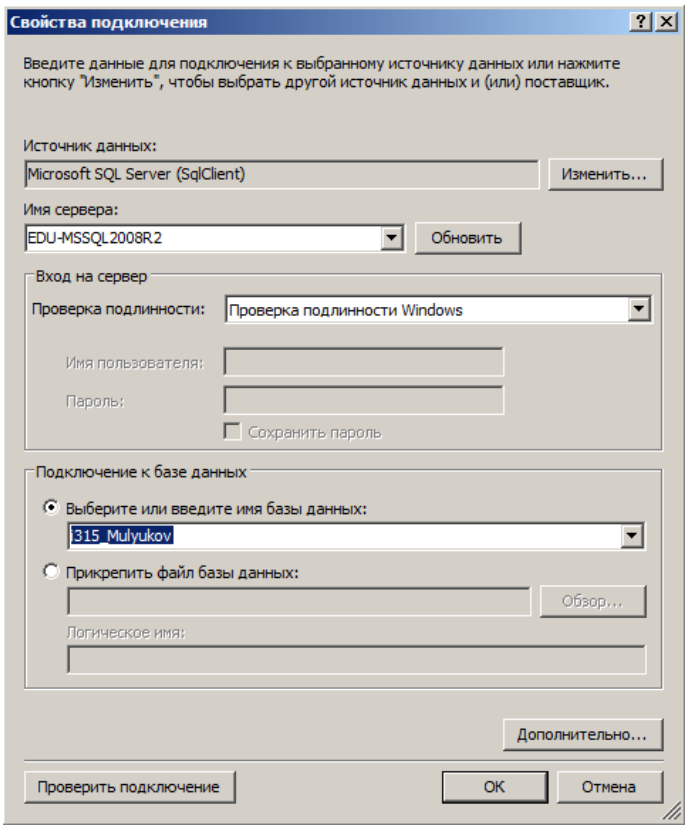


Рисунок 5 - Параметры подключения к БД

8. При возникновении выбора версии, укажите Entity Framework 6.x.
9. Включите в модель все таблицы базы данных «Приемная комиссия», схема модели БД представлена на рисунке 6.

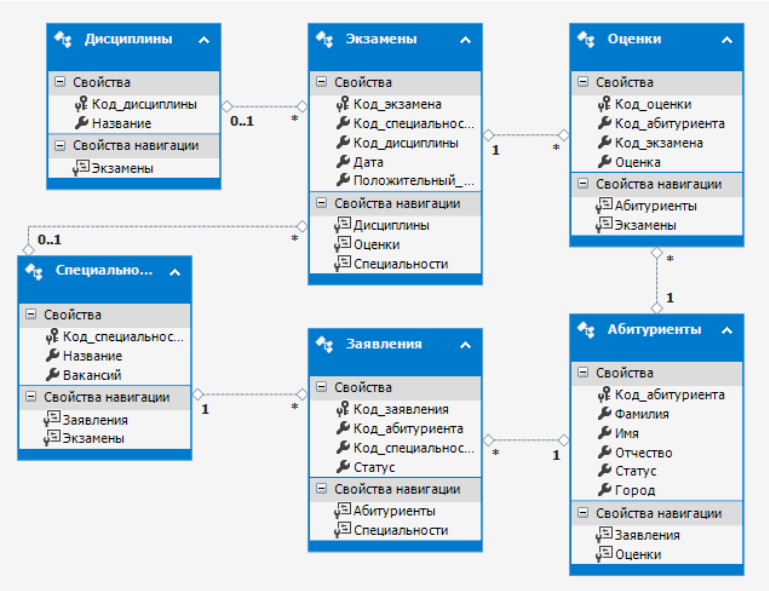


Рисунок 6 – Схема модели БД

Реализация работы со списком абитуриентов

10. Перейдите в конструктор формы и разместите на форме следующие элементы (см. Рисунок 7):
    - DataGridView dataGridView1 для отображения таблицы – списка абитуриентов;
    - TextBox textBox1 для ввода слова поиска;
    - ComboBox comboBox1 для выбора поля (параметра) поиска;
    - Button button1 – кнопку поиска;
    - Button button2 – кнопку редактирования выделенной записи;
    - Button button3 – кнопку добавления новой записи;
    - Label label1 – отображаемую в том случае, если список пуст.

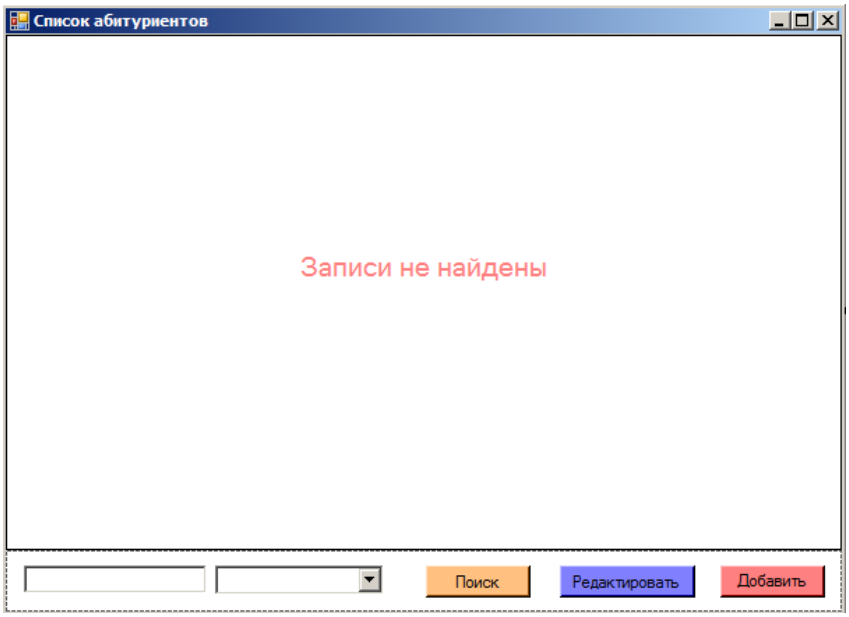
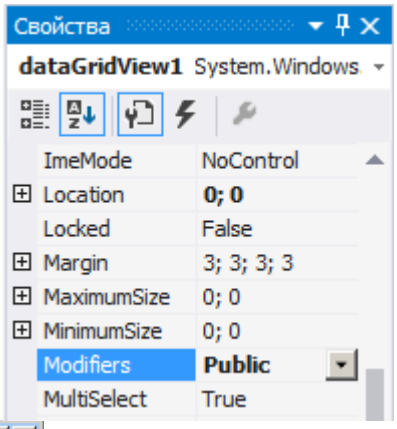


Рисунок 7 – Вид основного рабочего окна в редакторе форм

11. Настроив внешний вид элементов согласно общему дизайну приложения с помощью Конструктора форм, сделаем *label1* и *dataGridView1* доступными из других форм, указав модификатор *Public* (поле *Modifiers* в панели свойств элемента), а *comboBox1* заполним следующими строками (поле *Items* раздела «Данные» в панели свойств элемента):



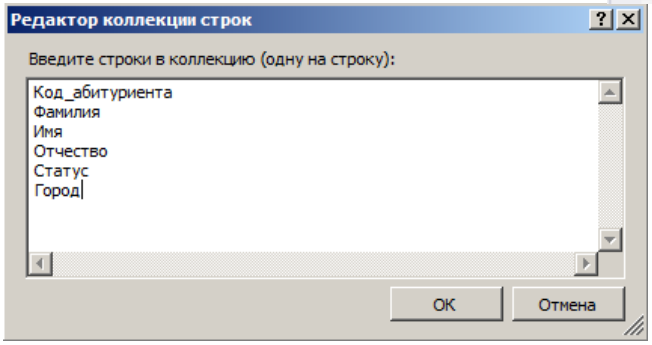


Рисунок 8 - Коллекция строк comboBox

12. Определим новый объект типа DbContext (хоть теперь он и называется pr116_IvanovaEntities) и список из объектов, соответствующий таблице Абитуриенты в базе данных. Из объекта pr116_IvanovaEntities «вытащим» данные в список и свяжем его с dataGridView1 следующим образом:

```
public partial class Form1 : Form
{
    public List<Абитуриенты> sheetabiturients;
    public pr116_IvanovaEntities db;

    ссылка: 1
    public Form1()
    {
        InitializeComponent();
        db = new pr116_IvanovaEntities();
        sheetabiturients = db.Абитуриенты
        .OrderBy(o => o.Код_абитуриента).ToList();
        dataGridView1.DataSource = sheetabiturients;
        dataGridView1.ReadOnly = true;
        if (dataGridView1.RowCount == 0) label1.Visible =
        true;
        else label1.Visible = false;
    }
```

Данный код обеспечивает получение и обработку (а именно, сортировку таблицы по коду абитуриента) данных, первоначальную установку отображения компонентов. Уже на данном этапе используются LINQ-запросы и лямбда-выражения.

13. Прежде чем собрать приложение и посмотреть результат, стоит обратить внимание на определение сущностного класса Абитуриенты, которому будет соответствовать каждая строка в dataGridView1:

```
public partial class Абитуриенты
{
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:Dc
    ссылок: 0
    public Абитуриенты()
    {
        this.Заявления = new HashSet<Заявления>();
        this.Оценки = new HashSet<Оценки>();
    }

    ссылок: 0
    public int Код_абитуриента { get; set; }
    ссылок: 0
    public string Фамилия { get; set; }
    ссылок: 0
    public string Имя { get; set; }
    ссылок: 0
    public string Отчество { get; set; }
    ссылок: 0
    public string Статус { get; set; }
    ссылок: 0
    public string Город { get; set; }

    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:Cc
    ссылка: 1
    public virtual ICollection<Заявления> Заявления { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:Cc
    ссылка: 1
    public virtual ICollection<Оценки> Оценки { get; set; }
}
```

14. Теперь внесем изменения со стороны GUI:
    - поскольку столбцы Заявления и Оценки ни разработчику, ни пользователю абсолютно не нужны, то проблема решается просто – не отображать их;
    - изменим название и размер колонки для удобство пользования и информативности визуализации данных.

Исправим это следующим кодом:

```
dataGridView1.Columns[0].HeaderText = "№";
dataGridView1.Columns[0].Width = 30;
dataGridView1.Columns[6].Visible = false;
dataGridView1.Columns[7].Visible = false;
```

15. Запустите приложение и просмотрите результат.