# ЛАБОРАТОРНАЯ РАБОТА №7

## Работа с базой данных в подключенном стиле

## 1 Цель работы

Освоить возможности работы с базой данных на основе классов ADO.Net в подключенном стиле.

## 2 Постановка задачи

Для заданной в индивидуальном задании предметной области спроектировать таблицу БД, в которой есть 4-5 полей. Таблица должна содержать допускающие присваивание значение поля NULL.

Доступ к БД реализовать на основе шаблона проектирования Active Record Программа должна иметь возможность отображать все записи таблицы, отображать записи, удовлетворяющие заданному критерию поиска, выполнять вставку и удаление записей.

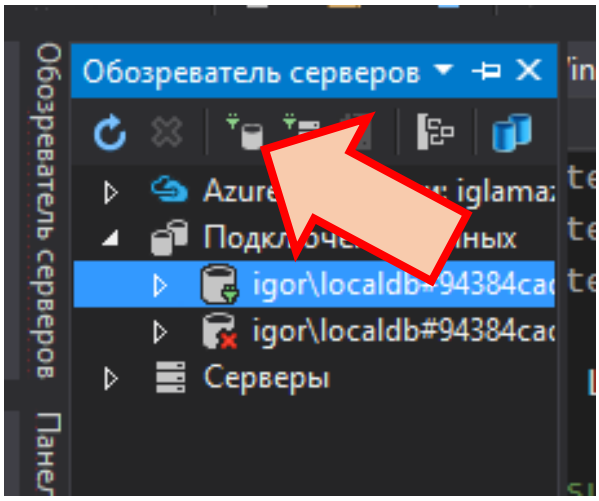
## 3 Индивидуальные задания

Варианты предметной области для проектирования БД:

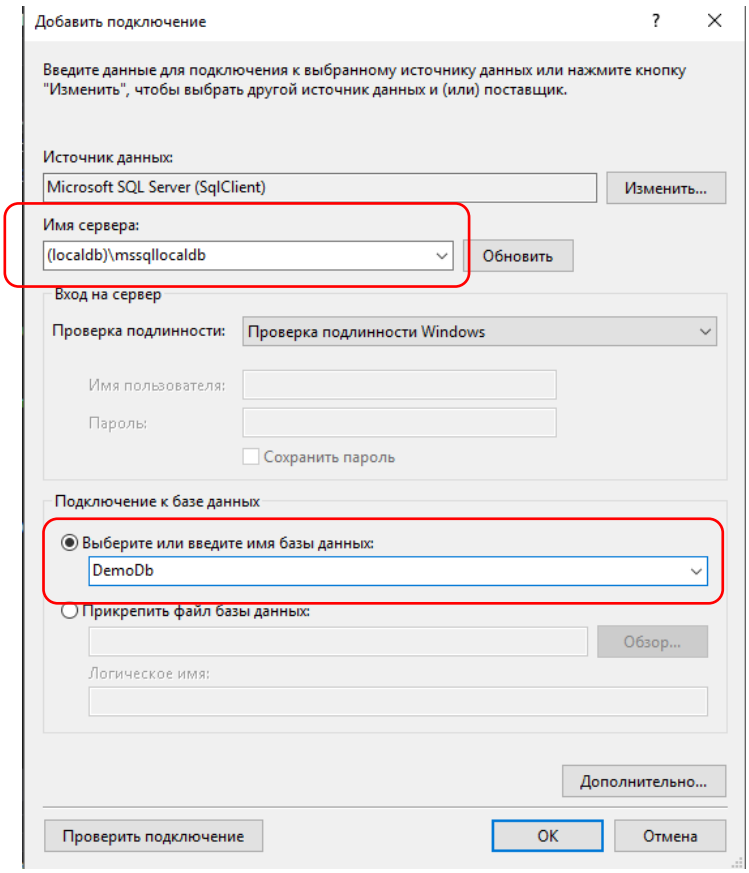
1. Автотранспорт
2. Жилищно-коммунальная сфера
3. Здравоохранение
4. Бытовое обслуживание населения
5. Образование
6. Муниципальное управление
7. Железнодорожный транспорт
8. Авиаперевозки
9. Компьютерная техника
10. Энергетика

## 4 Пример программы

В VisualStudio откройте окно обозревателя серверов и добавьте подключение к базе данных:



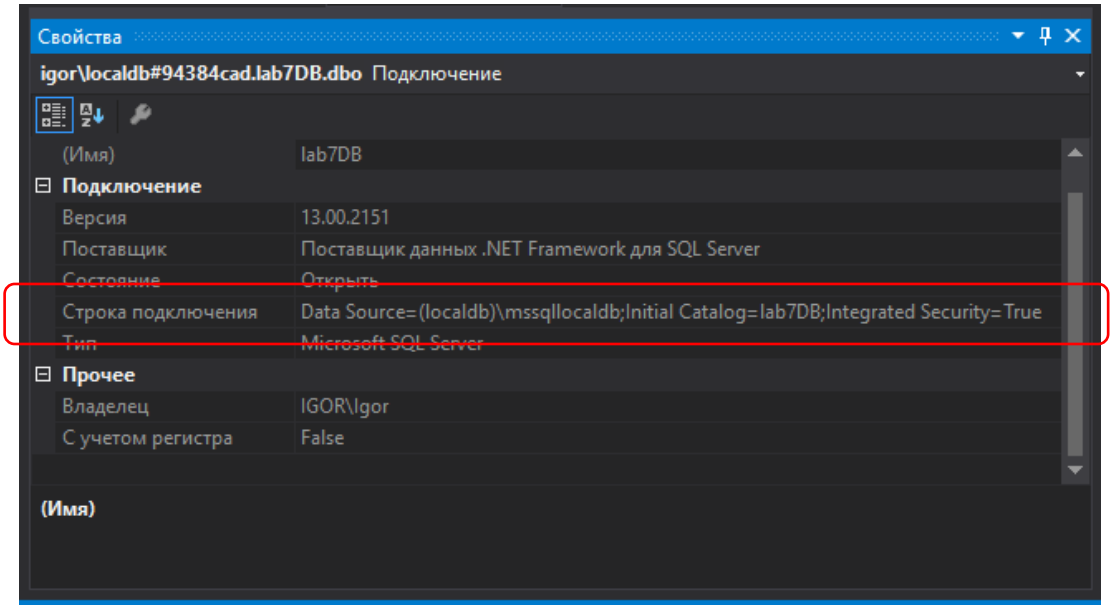

В открывшемся окне укажите имя сервера и имя базы данных:




Сервер **(localdb)\mssqllocaldb** можно использовать в Visual Studio 2015. Для (localdb) не нужно устанавливать на компьютер локальную версию

SqlServer. Если использовать локальную версию SqlServer, имя сервера может выглядеть, например, как **“.\SQLEXPRESS”**.

Созданная база данных появляется в окне обозревателя серверов. В окне свойств базы данных можно увидеть строку подключения:

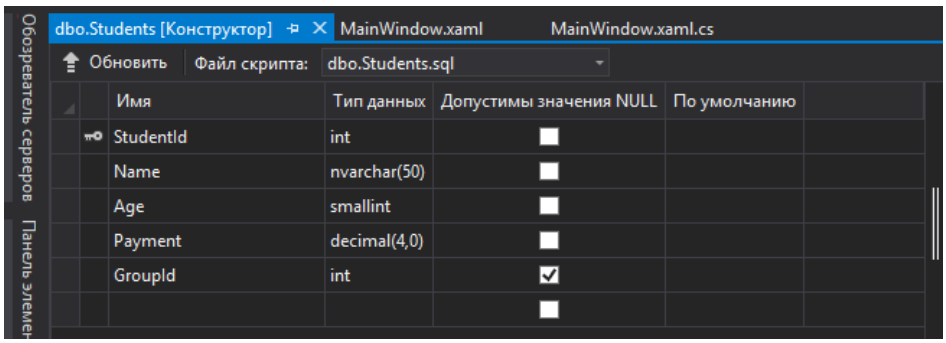



Откройте файл конфигурации (app.config) и внутри раздела configuration добавьте раздел, описывающий строку подключения:

```
<connectionStrings>
    <add name="DemoConnection"
       connectionString="Data Source=(localdb)\mssqllocaldb;
           Initial Catalog=lab7DB; Integrated Security=True"
         providerName="System.Data.SqlClient"/>
</connectionStrings>
```

(строка подключения должна быть записана одной строкой, без символов перевода строки). В дальнейшем строку подключения можно будет извлечь из файла конфигурации в любом месте программы. Для этого в обозревателе решения необходимо в ссылки добавить библиотеку System.Configuration, а в коде воспользоваться статическим свойством **ConfigurationManager.ConnectionStrings**.

Добавьте в созданную базу данных таблицу согласно индивидуальному заданию. Например:




4.1 Пример кода класса для взаимодействия с базой данных:

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data.SqlClient;

namespace Lab7
{
    public class Student
    {
        public int StudentId { get; set; }
        public String Name { get; set; }
        public int Age { get; set; }
        public decimal Payment { get; set; }
        public int GroupId { get; set; }

        static SqlConnection connection;

        public Student()
        {
            // Получение строки подключения из файла конфигурации
            var connString = ConfigurationManager
                .ConnectionStrings["DemoConnection"]
                .ConnectionString;
            // Создание объекта подключения
            connection = new SqlConnection(connString);
        }
        static Student()
        {
            // Получение строки подключения из файла конфигурации
            var connString = ConfigurationManager
                .ConnectionStrings["DemoConnection"]
                .ConnectionString;
            // Создание объекта подключения
            connection = new SqlConnection(connString);
        }
        /// <summary>
        /// Переопределение метода ToString()
        /// </summary>
```

```
        /// <returns></returns>
        public override string ToString()
        {
            return String.Format("id={0} - name: {1} - age: {2} -
payment: {3} - group No: {4}",
                StudentId, Name, Age, Payment, GroupId);
        }

        /// <summary>
        /// Получение списка всех студентов
        /// </summary>
        /// <returns>IEnumerable<Student></returns>
        public static IEnumerable<Student> GetAllStudents()
        {
            var commandString = "SELECT StudentId, Name, Age, Payment,
GroupId FROM Students";
            SqlCommand getAllCommand = new SqlCommand(commandString,
connection);
            connection.Open();
            var reader = getAllCommand.ExecuteReader();
            if(reader.HasRows)
            {
                while(reader.Read())
                {
                    var studentId = reader.GetInt32(0);
                    var name = reader.GetString(1);
                    var age = reader.GetInt16(2);
                    var payment = reader.GetDecimal(3);
                    var groupId = reader.GetSqlInt32(4);

                    var student = new Student
                    {
                        StudentId = studentId,
                        Name = name,
                        Age = age,
                        Payment = payment,
                        GroupId = groupId.IsNull == true
                                        ? 0
                                        : groupId.Value
                    };
                    yield return student;
                }
            };
            connection.Close();
        }

        /// <summary>
        /// Добавление новой записи в базу данных
        /// </summary>
        public void Insert()
        {
```

```
            var commandString = "INSERT INTO Students (Name, Age,
Payment, GroupId)"
                              + "VALUES (@name, @age, @payment, @grId)";
            SqlCommand insertCommand = new SqlCommand(commandString,
connection);

            insertCommand.Parameters.AddRange(new SqlParameter[] {
                    new SqlParameter("name", Name),
                    new SqlParameter("age", Age),
                    new SqlParameter("payment", Payment),
                    new SqlParameter("grId", GroupId),
                    });

            connection.Open();
            insertCommand.ExecuteNonQuery();
            connection.Close();
        }

        /// <summary>
        /// Получение записи с заданным id
        /// </summary>
        /// <param name="id">Значение StudentId для поиска</param>
        /// <returns>объект класса Student</returns>
        public static Student GetStudent(int id)
        {
            foreach(var student in GetAllStudents())
            {
                if (student.StudentId == id)
                    return student;
            }
            return null;
        }
        /// <summary>
        /// Изменение текущей записи в базе данных
        /// </summary>
        public void Update()
        {
            var commandString = "UPDATE Students SET Name=@name,
Age=@age, Payment=@payment, GroupId=@grId WHERE(StudentId=@id)";
            SqlCommand updateCommand = new SqlCommand(commandString,
connection);

            updateCommand.Parameters.AddRange(new SqlParameter[] {
                    new SqlParameter("name", Name),
                    new SqlParameter("age", Age),
                    new SqlParameter("payment", Payment),
                    new SqlParameter("grId", GroupId),
                    new SqlParameter("Id", StudentId),
                    });

            connection.Open();
```

```
            updateCommand.ExecuteNonQuery();
            connection.Close();
        }
        /// <summary>
        /// Удаление записи из базы данных
        /// </summary>
        /// <param name="id">Значение StudentId для удаляемой
записи</param>
        public static void Delete(int id)
        {
            var commandString = "DELETE FROM Students
WHERE(StudentId=@id)";
            SqlCommand deleteCommand = new SqlCommand(commandString,
connection);
            deleteCommand.Parameters.AddWithValue("id", id);
            connection.Open();
            deleteCommand.ExecuteNonQuery();
            connection.Close();
        }


    }
}
```

4.2 Пример разметки окна:

```
<Window.Resources>
        <Style TargetType="Button">
            <Setter Property="Button.Margin" Value="5"/>
        </Style>
</Window.Resources>
<Grid>
     <Grid.ColumnDefinitions>
          <ColumnDefinition Width="120"/>
          <ColumnDefinition/>
     </Grid.ColumnDefinitions>
     <StackPanel>
          <Button x:Name="btnFill" Click="btnFill_Click">
               Заполнить список
          </Button>
          <Button x:Name="btnAdd" Click="btnAdd_Click">
               Добавить
          </Button>
          <Button x:Name="btnEdit" Click="btnEdit_Click">
               Изменить
          </Button>
          <Button x:Name="btnRemove" Click="btnRemove_Click">
               Удалить
          </Button>
     </StackPanel>
```

```
        <ListBox x:Name="lBox" Grid.Column="1" ItemsSource="{Binding}">
        </ListBox>
</Grid>
```

4.3 Пример кода окна:

```
using System.Collections.ObjectModel;
using System.Windows;

namespace Lab7
{
    /// <summary>
    /// Логика взаимодействия для MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        /// <summary>
        /// Коллекция для привязки к списку
        /// </summary>
        ObservableCollection<Student> Students;

        public MainWindow()
        {
            Students = new ObservableCollection<Student>();
            InitializeComponent();
            lBox.DataContext = Students;
        }
        /// <summary>
        /// Заполнение коллекции данными
        /// </summary>
        void FillData()
        {
            Students.Clear();
            foreach (var item in Student.GetAllStudents())
            {
                Students.Add(item);
            }
        }

        private void btnFill_Click(object sender, RoutedEventArgs e)
        {
            FillData();
        }

        private void btnAdd_Click(object sender, RoutedEventArgs e)
        {
            var student = new Student()
            {
                Name = "Новый студент",
                Age = 22,
```

```
                Payment = 300,
                GroupId = 6
            };
            student.Insert();
            FillData();
        }

        private void btnEdit_Click(object sender, RoutedEventArgs e)
        {
            var student = (Student)lBox.SelectedItem;
            student.Name = "Измененное имя";
            student.Update();
            FillData();
        }

        private void btnRemove_Click(object sender, RoutedEventArgs e)
        {
            var id = ((Student)lBox.SelectedItem).StudentId;
            Student.Delete(id);
            FillData();
        }
    }
}
```

4.4 Пример выполнения программы: