**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.

1. **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.

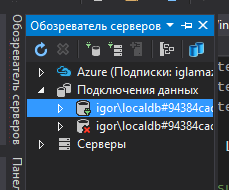
1. **Индивидуальные задания**

Варианты предметной области для проектирования БД:

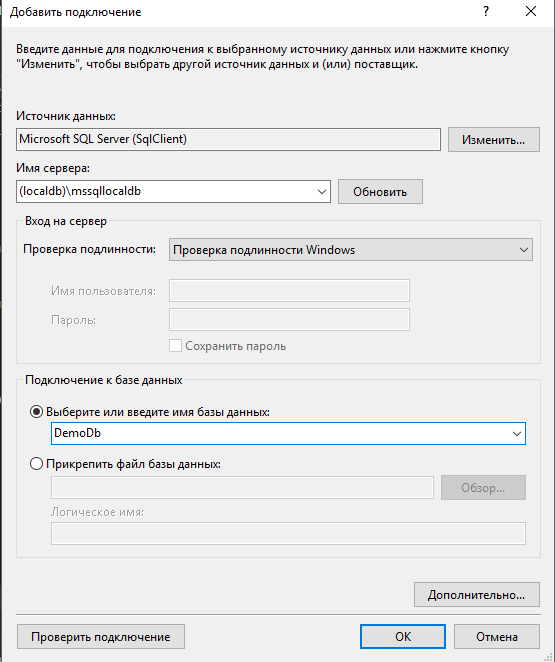
* 1. Автотранспорт
  2. Жилищно-коммунальная сфера
  3. Здравоохранение
  4. Бытовое обслуживание населения
  5. Образование
  6. Муниципальное управление
  7. Железнодорожный транспорт
  8. Авиаперевозки
  9. Компьютерная техника 10.Энергетика

1. **Пример программы**

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



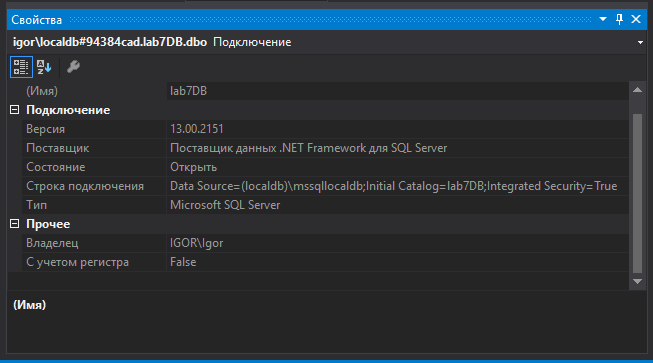
В открывшемся окне укажите имя сервера и имя базы данных:



Сервер **(localdb)\mssqllocaldb** можно использовать в Visual Studio 2022. 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"

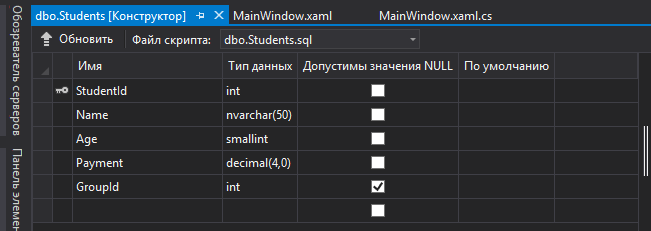
connectionString="Data Source=(localdb)\mssqllocaldb; Initial Catalog=lab7DB; Integrated Security=True"

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:



# Project 1. Использование строки запроса и SqlDataReader

* 1. Пример кода класса для взаимодействия с базой данных:

using System.Collections.Generic;

using System.Configuration;

using System.Data.SqlClient;

namespace BD

{

internal class Person

{

static SqlConnection connection;

public int Id { get; set; }

public string Name { get; set; }

public decimal Sum { get; set; }

static Person()

{

var connstring = ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString;

connection = new SqlConnection(connstring);

}

private static void openConnection()

{

// из-за метода getAllPersons() подключение может оставаться открытым

if (connection.State == System.Data.ConnectionState.Open)

connection.Close();

connection.Open();

}

public static IEnumerable<Person> getAllPersons()

{

openConnection();

var SQLstr = "SELECT \* FROM Person";

SqlCommand getAllCommand = new SqlCommand(SQLstr,connection);

var reader = getAllCommand.ExecuteReader();

if(reader.HasRows)

{

while(reader.Read())

{

var id = reader.GetInt32(0);

var name = reader.GetString(1);

var sum = reader.GetDecimal(2);

var person = new Person() { Id = id, Name = name, Sum = sum};

yield return person;

}

}

connection.Close();

}

public void Insert()

{

openConnection();

var SQLstr = "INSERT INTO Person (Name, Sum) VALUES (@name, @sum)";

SqlCommand insertCommand = new SqlCommand(SQLstr,connection);

insertCommand.Parameters.AddRange(new SqlParameter[]

{

new SqlParameter("name", Name),

new SqlParameter("sum", Sum)

});

insertCommand.ExecuteNonQuery();

connection.Close();

}

public static void Delete(int id)

{

openConnection();

var commandString = "DELETE FROM Person WHERE(Id=@id)";

SqlCommand deleteCommand = new SqlCommand(commandString, connection);

deleteCommand.Parameters.AddWithValue("id", id);

deleteCommand.ExecuteNonQuery();

connection.Close();

}

public static Person Find(string name)

{

foreach (var person in getAllPersons())

{

if (person.Name == name)

return person;

}

return null;

}

public void Update()

{

openConnection();

var commandString = "UPDATE Person SET Name=@name, Sum=@sum WHERE(Id=@id)";

SqlCommand updateCommand = new SqlCommand(commandString, connection);

updateCommand.Parameters.AddRange(new SqlParameter[] {

new SqlParameter("name", Name),

new SqlParameter("sum", Sum),

new SqlParameter("id", Id),});

updateCommand.ExecuteNonQuery();

connection.Close();

}

public override string ToString()

{

return $"№ {Id} Имя: {Name}, Сумма: {Sum:0.000}";

}

}

}

* 1. Пример разметки окна:

<Window x:Class="BD.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BD"

mc:Ignorable="d"

Title="Работа с БД" Height="450" Width="800">

<DockPanel LastChildFill="True" Background="Bisque">

<StackPanel Background="LightGreen" Width="150">

<Button x:Name="buttonView" Height ="50" Margin="10" Click="buttonView\_Click">Просмотр</Button>

<Button x:Name="buttonInsert" Height ="50" Margin="10" Click="buttonInsert\_Click">Вставить</Button>

<Button x:Name="buttonFind" Height ="50" Margin="10" Click="buttonFind\_Click">Найти</Button>

<Button x:Name="buttonChange" Height ="50" Margin="10" Click="buttonChange\_Click">Обновить</Button>

<Button x:Name="buttonDelete" Height ="50" Margin="10" Click="buttonDelete\_Click">Удалить</Button>

</StackPanel>

<StackPanel x:Name="stackpanelPerson" DockPanel.Dock="Top" Background="LightSkyBlue">

<StackPanel Orientation="Horizontal">

<Label Width="200" Margin="10">Имя</Label>

<TextBox x:Name="textBoxName" Width="200" Margin="10" Text="{Binding Path=Name}"></TextBox>

</StackPanel>

<StackPanel Orientation="Horizontal">

<Label Width="200" Margin="10">Сумма</Label>

<TextBox x:Name="textBoxSum" Width="200" Margin="10" Text="{Binding Path=Sum}"></TextBox>

</StackPanel>

</StackPanel>

<ListBox x:Name="listBox" ItemsSource="{Binding}" Margin="10" Background="Beige"></ListBox>

</DockPanel>

</Window>

* 1. Пример кода окна:

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace BD

{

public partial class MainWindow : Window

{

Person person;

ObservableCollection<Person> collection = new ObservableCollection<Person>();

public MainWindow()

{

InitializeComponent();

person = new Person();

stackpanelPerson.DataContext = person;

listBox.DataContext = collection;

FillData();

}

private void buttonView\_Click(object sender, RoutedEventArgs e)

{

// MessageBox.Show(person.ToString());

FillData();

}

private void FillData()

{

collection.Clear();

foreach (var p in Person.getAllPersons())

{

collection.Add(p);

}

}

private void buttonInsert\_Click(object sender, RoutedEventArgs e)

{

person.Insert();

FillData();

}

private void buttonFind\_Click(object sender, RoutedEventArgs e)

{

person = Person.Find(textBoxName.Text);

if (person == null)

{

MessageBox.Show("Нет такой записи!");

person = new Person();

}

else

MessageBox.Show(person.ToString());

}

private void buttonChange\_Click(object sender, RoutedEventArgs e)

{

if(listBox.SelectedIndex == -1)

{

MessageBox.Show("Нет выбранной записи!");

return;

}

person.Id = ((Person)listBox.SelectedItem).Id;

if (textBoxName.Text.Length > 0)

person.Name = textBoxName.Text;

else

person.Name = ((Person)listBox.SelectedItem).Name;

decimal d = Convert.ToDecimal(textBoxSum.Text);

if (d != 0)

person.Sum = d;

else

person.Sum = ((Person)listBox.SelectedItem).Sum;

person.Update();

FillData();

}

private void buttonDelete\_Click(object sender, RoutedEventArgs e)

{

if (listBox.SelectedIndex == -1)

{

MessageBox.Show("Нет выбранной записи!");

return;

}

var id = ((Person)listBox.SelectedItem).Id;

Person.Delete(id);

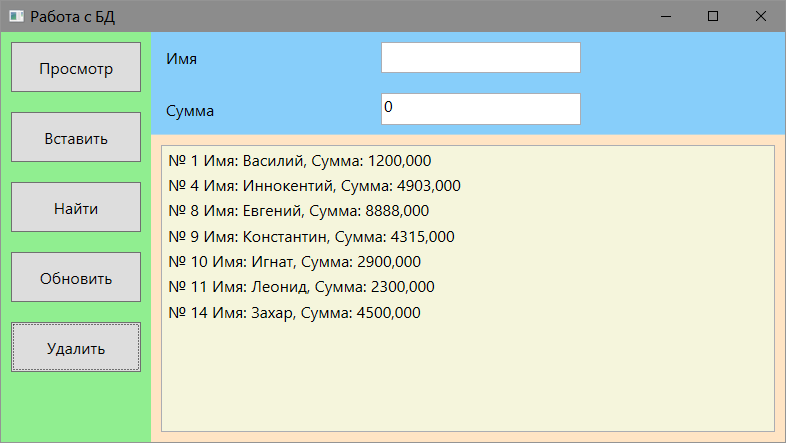
FillData();

}

}

}

* 1. Пример выполнения программы:



# Project 2. Работа с SqlDataAdapter и DataTable

* 1. Пример кода класса для взаимодействия с базой данных:

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

namespace BD\_Adapter

{

public class Person

{

static string connectionString;

static SqlConnection connection;

static SqlDataAdapter adapter;

static DataTable personTable = new DataTable();

public int ID { get; set; }

public string Name { get; set; }

public int Age { get; set; }

public string Company { get; set; }

public decimal Salary { get; set; }

public static void newConnection()

{

connectionString = ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString;

connection = new SqlConnection(connectionString);

connection.Open();

}

public static DataTable ViewAll()

{

newConnection();

string sql = "SELECT \* FROM Person";

adapter = new SqlDataAdapter(sql, connection);

adapter.Fill(personTable);

connection.Close();

return personTable;

}

public static void Update()

{

newConnection();

// commandBuilder позволяет автоматически сгенерировать нужные SQL-выражения

SqlCommandBuilder commandBuilder = new SqlCommandBuilder(adapter);

adapter.Update(personTable);

connection.Close();

}

public string Find()

{

// для каждого варианта поиска требуется отдельный SELECT

newConnection();

DataTable personTable1 = new DataTable();

string str = "";

if (Name != null)

{

SqlCommand command = new SqlCommand("SELECT \* FROM Person WHERE (Name=@name)", connection);

command.Parameters.AddWithValue("name", Name);

adapter = new SqlDataAdapter(command);

adapter.Fill(personTable1);

foreach (DataRow row in personTable1.Rows)

{

// получаем все ячейки строки

var cells = row.ItemArray;

foreach (object cell in cells)

str += $"\t{cell}";

str += "\n";

}

return str;

}

else if (Company != null)

{

SqlCommand command = new SqlCommand("SELECT \* FROM Person WHERE (Company=@company)", connection);

command.Parameters.AddWithValue("company", Company);

adapter = new SqlDataAdapter(command);

adapter.Fill(personTable1);

foreach (DataRow row in personTable1.Rows)

{

// получаем все ячейки строки

var cells = row.ItemArray;

foreach (object cell in cells)

str += $"\t{cell}";

str += "\n";

}

return str;

}

return null;

}

public override string ToString()

{

return $"{ID} {Name}, {Age} лет. Сотрудник {Company}, оклад {Salary}";

}

}

}

* 1. Пример разметки окна:

<Window x:Class="BD\_Adapter.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BD\_Adapter"

mc:Ignorable="d"

Title="BD + Adapter" Height="450" Width="800" >

<Grid Background="LightSkyBlue">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="150"></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0" Background="LightSteelBlue">

<Button Height ="50" Margin="10" Click="ButtonView\_Click">Просмотр</Button>

<Button Height ="50" Margin="10" Click="ButtonUpdate\_Click">Обновить</Button>

<Button Height ="50" Margin="10" Click="ButtonFind\_Click">Найти</Button>

</StackPanel>

<DataGrid AutoGenerateColumns="False" x:Name="personGrid" Grid.Column="1" Margin="25">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="ID" Width="50"/>

<DataGridTextColumn Binding="{Binding Name}" Header="Name" Width="130"/>

<DataGridTextColumn Binding="{Binding Age}" Header="Age" Width="130"/>

<DataGridTextColumn Binding="{Binding Company}" Header="Company" Width="130"/>

<DataGridTextColumn Binding="{Binding Salary}" Header="Salary" Width="130"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* 1. Пример кода окна:

using System;

using System.Collections.Generic;

using System.Data;

using System.Windows;

namespace BD\_Adapter

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

DataTable personTable = new DataTable();

public MainWindow()

{

InitializeComponent();

Fill();

}

private void Fill()

{

personTable.Rows.Clear();

personTable = Person.ViewAll();

personGrid.ItemsSource = personTable.DefaultView;

}

private void ButtonView\_Click(object sender, RoutedEventArgs e)

{

Fill();

}

private void ButtonFind\_Click(object sender, RoutedEventArgs e)

{

Person person = new Person();

WindowPerson windowPerson = new WindowPerson(person);

if (windowPerson.ShowDialog() == false)

return;

MessageBox.Show(person.Find());

}

private void ButtonUpdate\_Click(object sender, RoutedEventArgs e)

{

Person.Update();

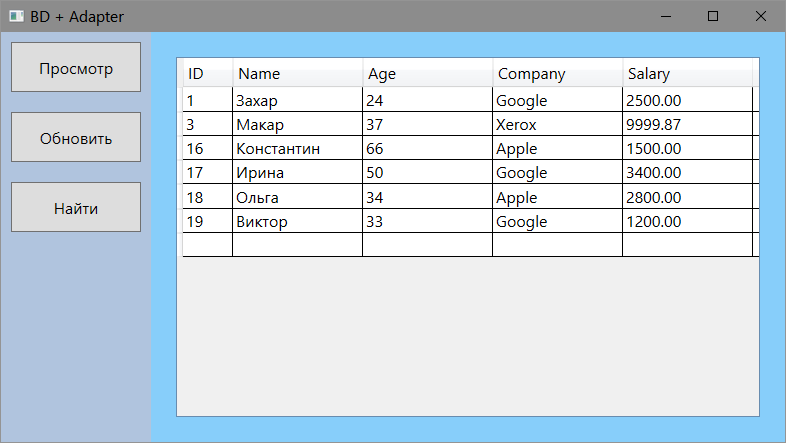
Fill();

}

}

}

* 1. Пример выполнения программы:



* 1. Разметка вспомогательного окна для поиска:

<Window x:Class="BD\_Adapter.WindowPerson"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BD\_Adapter"

mc:Ignorable="d"

Title="WindowPerson" Height="400" Width="450" FontSize="18">

<Grid x:Name="grid" Background="LightSalmon">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition Width="2\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Label Content="ID" Margin="15"/>

<Label Content="Name" Margin="15" Grid.Row="1"/>

<Label Content="Age" Margin="15" Grid.Row="2"/>

<Label Content="Company" Margin="15" Grid.Row="3"/>

<Label Content="Salary" Margin="15" Grid.Row="4"/>

<TextBox x:Name="textBoxID" Margin="15" Grid.Row="0" Grid.Column="1" Text="{Binding Path=ID}"/>

<TextBox x:Name="textBoxName" Margin="15" Grid.Row="1" Grid.Column="1" Text="{Binding Path=Name}"/>

<TextBox x:Name="textBoxAge" Margin="15" Grid.Row="2" Grid.Column="1" Text="{Binding Path=Age}"/>

<TextBox x:Name="textBoxCompany" Margin="15" Grid.Row="3" Grid.Column="1" Text="{Binding Path=Company}"/>

<TextBox x:Name="textBoxSalary" Margin="15" Grid.Row="4" Grid.Column="1" Text="{Binding Path=Salary}"/>

<Button x:Name="buttonCancel" Content="Cancel" Margin="15" Grid.Row="5" Click="buttonCancel\_Click" />

<Button x:Name="buttonOk" Content="OK" Margin="15" Grid.Row="5" Grid.Column="1" Click="buttonOk\_Click"/>

</Grid>

</Window>

* 1. Код вспомогательного окна

using System.Windows;

namespace BD\_Adapter

{

public partial class WindowPerson : Window

{

public WindowPerson(Person person)

{

InitializeComponent();

grid.DataContext = person;

}

private void buttonOk\_Click(object sender, RoutedEventArgs e)

{

this.DialogResult = true;

}

private void buttonCancel\_Click(object sender, RoutedEventArgs e)

{

this.DialogResult = false;

}

}

}

* 1. Вид вспомогательного окна

