МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И ПРОГРАММИРОВАНИЯ

КАФЕДРА компьютерных технологий и программной инженерии

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старший преподаватель |  |  |  | Н.В. Путилова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №8 |
| «Проектирование взаимодействия базы данных и приложения» |
| по курсу: Проектирование баз данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4932 |  | 28.09.21 |  | А.И.Белов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

1. Задание

По аналогии с примерами, приведенными в п. 1 и 2 реализовать запросы а) .. в), указанные в варианте задания. Все запросы должны не содержать вложенных запросов или агрегатных функций. (Используйте псевдонимы)

Вариант: 2. Садоводство: участки, владельцы с учетом совместной собственности, линии/номер участка, площадь стоимость постройки, тип построек ,взносы в фонд садоводства

а. номера участков владельцев с отчеством, заканчивающимся на «ич»

б. участки, на которых зарегистрировано более 1 типа постройки

в. тип взносов, которые пока никто не оплатил

г. Владелец (владельцы) участка максимальной площади

д. Владельцы участков с максимальным числом типов построек

е. Владельцы, оплатившие все типы взносов на букву о

ж. Участки, на которых нет бань, но есть туалеты

1.  Физическая модель БД
2. Код приложения

Классы:

Bulding.cs

public class Building

{

public virtual int ID { get; set; }

public virtual string TypeName { get { return Type.Name; } }

public virtual TypeBuilding Type { get; set; }

public virtual double Size { get; set; }

public virtual double Price { get; set; }

}

TypeBuilding.cs

public class TypeBuilding

{

public virtual int ID { get; set; }

public virtual string Name { get; set; }

}

Маппинг:

BuldingMapping.cs

public class BuildingMapping : ClassMap<Building>

{

public BuildingMapping()

{

Table("Building");

Id(x => x.ID).Column("ID");

Map(x => x.Size).Column("Size");

Map(x => x.Price).Column("Price");

References<TypeBuilding>(x => x.Type, "ID\_TypeBuilding").Cascade.AllDeleteOrphan();

}

}

TypeBuildingMapping.cs

public class TypeBuildingMapping : ClassMap<TypeBuilding>

{

public TypeBuildingMapping()

{

Table("TypeBuilding");

Id(x => x.ID).Column("ID");

Map(x => x.Name).Column("Name");

}

}

Форма:

public partial class Form1 : Form

{

string stringConnection = "Data Source=DESKTOP-FC16M2F;Initial Catalog=GardeningDB;User ID=sa;Password=1";

public Form1()

{

InitializeComponent();

}

public void RefreshButtonClick(object sender, System.EventArgs e)

{

dgv.DataSource = null;

switch (toolStripComboBox1.SelectedItem.ToString())

{

case "ADO.NET: Подключение источника данных": LoadData1(); break;

case "ADO.NET: Чтение результата запроса": LoadData2(); break;

case "Nhibernate: Маппинг таблицы": LoadData3(); break;

}

}

public SqlConnection GetSql()

{

return new SqlConnection(stringConnection);

}

public ISessionFactory GetNhibernate()

{

var cfg = Fluently.Configure().Database(MsSqlConfiguration.MsSql2012.ConnectionString(stringConnection)).Mappings(m => m.FluentMappings.AddFromAssembly(Assembly.GetExecutingAssembly())).BuildConfiguration();

var factory = cfg.BuildSessionFactory();

return factory;

}

/// <summary>

/// ADO.NET: Подключение источника данных

/// </summary>

public void LoadData1()

{

SqlConnection connection = GetSql();

connection.Open();

SqlCommand command = new SqlCommand("SELECT Building.Size, Building.Price, TypeBuilding.Name FROM Building JOIN TypeBuilding ON Building.ID\_TypeBuilding=TypeBuilding.ID", connection);

SqlDataReader dataReader = command.ExecuteReader();

if(dataReader.HasRows) {

DataTable dt = new DataTable();

dt.Load(dataReader);

dgv.DataSource = dt;

};

}

/// <summary>

/// ADO.NET: Чтение результата запроса

/// </summary>

public void LoadData2()

{

SqlConnection connection = GetSql();

connection.Open();

SqlCommand command = new SqlCommand("SELECT \* FROM Building JOIN TypeBuilding ON Building.ID\_TypeBuilding=TypeBuilding.ID", connection);

SqlDataReader dataReader = command.ExecuteReader();

if(dataReader.HasRows)

{

List<Building> list = new List<Building>();

while (dataReader.Read())

{

list.Add(new Building()

{

Price = Convert.ToDouble(dataReader["Price"]),

Size = Convert.ToDouble(dataReader["Size"]),

Type = new TypeBuilding()

{

Name = dataReader["Name"].ToString()

}

});

}

dgv.DataSource = list;

dgv.Columns["ID"].Visible = false;

dgv.Columns["Type"].Visible = false;

}

}

/// <summary>

/// Nhibernate: Маппинг таблицы

/// </summary>

public void LoadData3()

{

ISession session = GetNhibernate().OpenSession();

var list = session.Query<Building>().ToList();

dgv.DataSource = list;

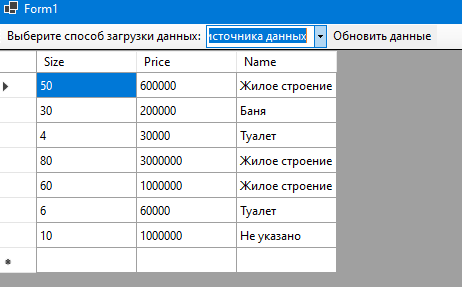
dgv.Columns["ID"].Visible = false;

dgv.Columns["Type"].Visible = false;

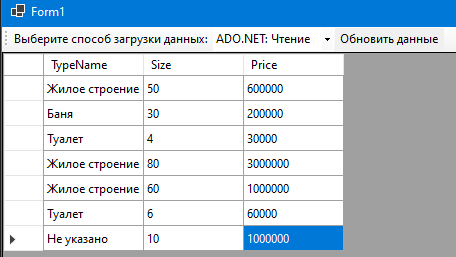
}

}

1. Скриншоты
2. ADO.NET: Подключение источника данных



1. ADO.NET: Чтение результата запроса



1. Nhibernate: Маппинг таблицы

