**Министерсво образования и нАукИ украины**

**Одесский национальный МОРСКОЙ университет**

Кафедра «Техническая кибернетика»

### Курсовая работа

По дисциплине: «Организация баз данных»

Тема: «Информационная система учебного отдела ВУЗа. Расписание практических занятий.»

Тема 2. Задание 1.

Выполнил:

студент 3 к. 2 гр. КСФ

Шипков В.И.

Руководитель:

Ст. преподаватель

Рублев И.С.

# 

# Одесса 2018

ОГЛАВЛЕНИЕ

1. [ЗАДАЧА 3](#_Toc482690009)

2. [ПОСТРОЕНИЕ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ МОДЕЛИ БД 5](#_Toc482690010)

3. [СОЗДАНИЕ БАЗЫ ДАННЫХ В MySQL Server 7](#_Toc482690011)

4. [РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ 8](#_Toc482690012)

[ВЫВОД 10](#_Toc482690013)

[СПИСОК ЛИТЕРАТУРЫ 11](#_Toc482690014)

[ПРИЛОЖЕНИЕ 12](#_Toc482690015)

1. **ЗАДАЧА**
2. **Построить логическую и физическую модели БД:**

Логическая модель должна учитывать максимально возможное количество требований предметной области. При необходимости можно вводить «суррогатные ключи» в проектируемые сущности. В этом случае должны быть определены соответствующие им альтернативные ключи.

* При разработке логической модели должны быть правильно определены основные сущности, между ними установлены связи, отвечающие указанным ниже требованиям на добавление, изменение и удаление данных.
* Те требования предметной области, которые не удается выполнить при построении модели, должны быть учтены при создании клиентского приложения и ограничений ссылочной целостности БД.

Физическая модель должна быть построена для СУБД MySQL, причем схема модели БД должны удовлетворять условиям не меньше чем НБКФ.

**ОПИСАНИЕ ЗАДАЧИ:**

**Расписание занятий** составляется на один семестр.

Система должна предусматривать учет:

* действующего аудиторного фонда и его занятости лекциями;
* всех учебных групп факультетов ВУЗа с учетом количества студентов в них;
* преподаваемых дисциплин ( практических занятий );
* преподавателей, ведущих занятия и кафедр, на которых они работают;

**Аудитория** может быть занята для занятий с учебной группой по определенным датам (дням недели с понедельника по пятницу) с учебной недели №\_\_ по учебную неделю №\_\_, в определенное время (с 1 пары по 5 пару), если количество мест в ней не менее чем количество студентов в группе. Расписание пар: 8:30 - 10:00; 10:10 - 11:30; 11:40 - 13:00; 13:10 - 14:30; 14:40 - 16:00. Аудитория может быть учебной лабораторией или компьютерным классом, т.е в ней проводятся занятия не по всем предметам.

Учебные недели 1 семестра - с 1 по 16 неделю, начиная с 1 сентября;

2 семестра - с 1 по 20 неделю, начиная с 20 января.

Каждая **аудитория ВУЗа**, характеризуется следующими данными:

**Аудитория:**

* Номер аудитории (3х-значный номер, 1 цифра - этаж корпуса, другие две - номер на этаже);
* Корпус (главный, административный, лабораторный);
* Количество мест в аудитории;

**Практические занятия** проводятся для студентов определенной учебной группы, заданного курса и факультета. Практические занятия проводятся по дисциплине, запланированной факультетом. Название дисциплины может быть одинаковым для разных факультетов.

**Занятие:**

Название дисциплины

Количество занятий по дисциплине;

Учебная группа;

Специальность студентов;

Факультет, где обучаются студенты;

Кафедра, на которой работает преподаватели;

Преподаватели, ведущие занятия (1-2 человека.)

Занятия проводятся 1 или 2 преподавателями одной кафедры. Если в группе менее 16 студентов, то преподаватель один, если более - два.

**Преподаватели:**

№ удостоверения отдела кадров;

ФИО преподавателя;

Кафедра; Факультет, к которому относится кафедра;

Должность; Звание; Преподавательский стаж;

**Расписание занятий** - определяет дату, время, аудиторию, учебную группу и предмет проводимого занятия.

При этом должны выполняться следующие условия:

В конкретные день недели и пару аудитория может быть свободна либо занята под определенное занятие, причем в одну и ту же аудиторию на данные день и пару нельзя планировать более одного занятия. Количество занятий не превышает количества занятий, предусмотренного программой.

Количество студентов на занятии не должно превышать вместимости аудитории.

На определенные (день, пару) для данной учебной группы нельзя планировать более 1 занятия.

Занятия по дисциплине могут проводиться 1-2 раза в неделю, возможно в различных аудиториях. В течение семестра в разные учебные недели количество занятий может быть различно.

Изменения в расписании занятий (добавление, изменение, удаление) производить без удаления информации из других таблиц.

Удаление записи о занятии должно автоматически освобождать занятую аудиторию.

Изменения в должности, званиях и стаже преподавателя, а также при его замене не должна нарушаться целостность данных.

Предусмотреть быстрый поиск свободных аудиторий для заданных дня недели, пары и учебного корпуса.

Предусмотреть быстрый поиск по кафедрам и ФИО преподавателя (выписка для преподавателя ).

Пример данных в БД:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Дата** | **27.01.03** | **10.02.03** | **13.02.03** | **10.02.03** | **13.02.03** |
| День недели | Понедельник | Понедельник | Четверг | Понедельник | Четверг |
| № пары | 3 | 4 | 3 | 2 | 2 |
| Начало пары | 11:40 | 13:10 | 11:40 | 10:10 | 10:10 |
| Аудитория | 216 | 407 | 104 | 301 | 304 |
| Корпус | Учебн | Учебн | Учебн | Учебн | Учебн |
| Факультет | КС | МП | МП | КС | КС |
| Семестр | 6 | 1 | 1 | 6 | 6 |
| Группа | 3 КС (16 чел) | 1 МП (28 чел) | 1 МП (28 чел) | 3 КС (16 чел) | 1, 2, 3, 4 КС |
| Специальность | Инф технол | Механ порт | Механ порт | Инф техн | Инф техн |
| Дисциплина | Базы данных | Физика | Выч техника | Теория множ | Теория множ |
| ФИО преподавателя | Ивлев В.С. | Петров А.И. | Ивлев В.С. | Ефимов А.В. | Ефимов А.В. |
|  |  | Васильев Н.А. | Гарин К.С. |  |  |
| № удостоверения ОК | 1234567 | 2345678 | 1234567 | 3456780 | 3456780 |
| Кафедра | ТК | Физики и хим | ТК | Математ | Математ |
| Факультет | СМ | СМ | СМ | КС | КС |
| Должность | Ст преп | Доцент | Ст преп | Доцент | Доцент |
| Звание |  | Доцент |  | Доцент | Доцент |
| Стаж | 4 | 13 | 4 | 8 | 8 |

1. **На основании физической модели создать соответствующие таблицы БД «Praktika1A»**

Для всех таблиц создать необходимые триггера и одну хранимую процедуру (аудитория практического занятия).

1. **В Visual Studio создать программу-клиент для «Praktika1A.**

Программа должна иметь следующие формы:

**Главная форма**- для формирования **Расписания занятий**. Позволяет закреплять для **Практического занятия** проводимого для **Учебной группы** определенную **Аудиторию** на конкретные **День недели (дату), Пару**.

Позволяет проводить фильтрацию / отбор данных по факультету, названию предмета, ФИО преподавателя.

**Вспомогательные формы:**

Форма учета **Практических занятий** и определения для нее **Группы студентов** и **Преподавателей.**

**Форма - отчет** Расписание практических занятий для учебных групп факультета по дням недели:

|  |  |  |
| --- | --- | --- |
| Кораблестроительный факультет 1 семестр 03г | | |
| Курс 3 специальность КС, гр 3-4 | | |
|  | Понедельник | Вторник | Среда | Четверг | Пятница |
| 1 пара |  |  |  |  |  |
| 2 пара | Компьютерная графика  Серов А.С. 216 учб | Теория автоматов  Ефимов А.В. 103 учб |  |  |  |
| 3 пара | Базы данных  Ивлев В.С. 216 учб |  |  |  | Численные методы и моделирование  Арин К.В. 304 учб |
| 4 пара |  |  |  | Теория множеств  Ефимов А.В.301 учб |  |

1. **Использовать транзакции** (блокирование таблиц) для многопользовательской работы с БД.

**Результатом работы являются:**

Файл и глава в отчете о созданных логической и физической моделях (сущности, их свойства, ключи; связи, их тип, ссылочная целостность)

Файлы базы данных, заполненные информацией. Источник информации - расписание лекций текущего семестра ОНМУ. Описание и текст созданных триггеров, индексов и хранимых процедур БД.

Файлы программы-клиента. Распечатка текста программ и примеров - отчетов разработанных форм.

1. **ПОСТРОЕНИЕ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ МОДЕЛИ БД**

На основании представленной задачи, мной были выделены следующие сущности и их атрибуты:

**Сущность** – это класс однотипных объектов, информация о которых должна быть учтена в модели.

**Атрибуты сущности** – это характеристика или свойство сущности, имеющая имя, уникальное для данной сущности.

При построении своей логической модели, я выделил следующие сущности: audience, day\_week, degree, department, discipline, faculty, form\_training, number\_lesson, post, specialty, sschedule, study\_group, teachers, type\_occupation.

Атрибутами сущности, то есть особыми полями, которые отображают основную логику связей, так называемые первичные ключи. Помогают осуществлять точные операции над базами и помогают пользователю избежать проблем в связи заполнением и дальнейшей эксплуатацией созданной базы.

**Связи** – это некоторая ассоциация между двумя сущностями. Одна сущность может быть связана с другой или сама с собой. Связи позволяют по экземпляру одной сущности находить соответствующие экземпляры другой, связанные с нею.

В разработанных сущностях, присутствуют следующие атрибуты:

**audience:**

number\_aud – номер аудитории (первичный ключ);

quantity\_seats – количество мест;

housing – корпус.

**day\_week:**

id\_day – уникальный идентификатор дня (первичный ключ);

name\_day – название дня недели.

**degree:**

id\_degree – уникальный идентификатор звания преподавателя (первичный ключ);

name\_degree – звание преподавателя.

**department:**

id\_department – уникальный идентификатор кафедры (первичный ключ);

name\_department – название кафедры.

**discipline:**

id\_discipline – уникальный идентификатор дисциплины (первичный ключ);

name\_discipline – название дисциплины.

**faculty:**

id\_faculty – уникальный идентификатор факультета (первичный ключ);

name\_faculty – название факультета.

**form\_training:**

id\_form – уникальный идентификатор формы обучения (первичный ключ);

name\_form– название формы обучения.

**number\_lesson:**

id\_num – уникальный идентификатор занятия (первичный ключ);

number – номер пары;

week\_start – начальная неделя;

week\_end – конечная неделя.

**post:**

id\_post – уникальный идентификатор поста (первичный ключ);

name\_post– название поста.

**specialty:**

id\_specialty – уникальный идентификатор специальности (первичный ключ);

name\_specialty– название специальности;

id\_faculty - уникальный идентификатор факультета;

id\_discipline - уникальный идентификатор дисциплины.

**sschedule:**

id\_schedule – уникальный идентификатор пункта в расписании (первичный ключ);

id\_group – уникальный идентификатор учебной группы;

number\_aud - номер аудитории;

id\_day - уникальный идентификатор дня;

id\_type - уникальный идентификатор типа занятия;

id\_num - уникальный идентификатор занятия;

card\_num - номер карты преподавателя.

**study\_group:**

id\_group – уникальный идентификатор учебной группы (первичный ключ);

name\_group – название учебной группы;

number\_students – количество студентов в группе;

id\_specialty - уникальный идентификатор специальности;

id\_form - уникальный идентификатор формы обучения;

date\_creation – дата создания учебной группы.

**teachers:**

card\_num – номер карты преподавателя (первичный ключ);

experience – стаж работы;

FIO\_teacher – фамилия, имя и отчество преподавателя;

id\_department - уникальный идентификатор кафедры

id\_post - уникальный идентификатор поста;

id\_degree - уникальный идентификатор звания преподавателя.

**type\_occupation:**

id\_type – уникальный идентификатор типа занятия (первичный ключ);

name\_type – название типа занятия.

В общем виде вся логическая структура составленной базы данных, имеет следующий вид (рис.2.1):

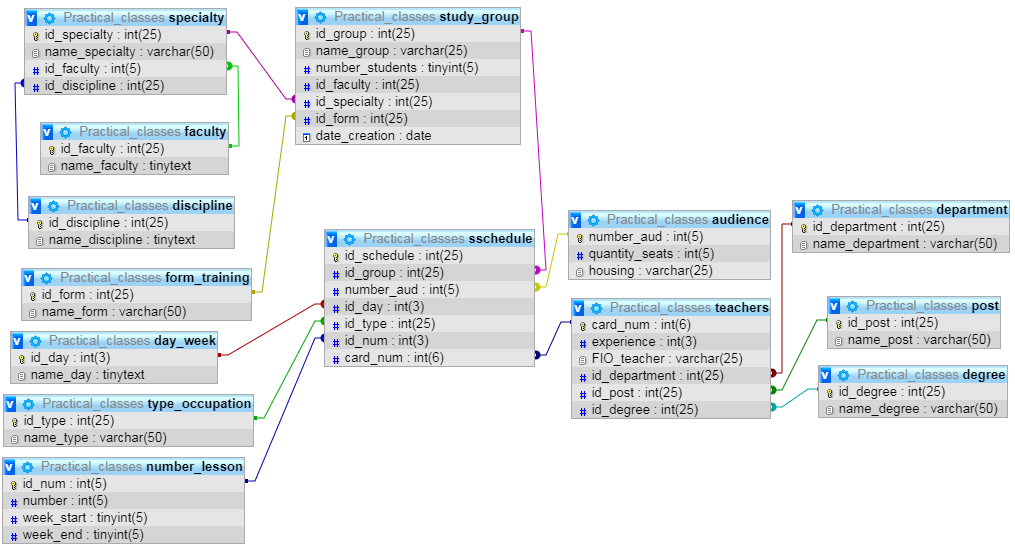


Рисунок 2.1 – Модель базы данных расписания

1. **СОЗДАНИЕ БАЗЫ ДАННЫХ В MySQL Server**

Создание базы данных велось в **MySQL Server,** которая в дальнейшем была подключена к проекту на Visual Studio и использована с языком программирования C#, для создания полноценного клиентского приложения. Пример одной из таблиц приведен на рис.3.1.

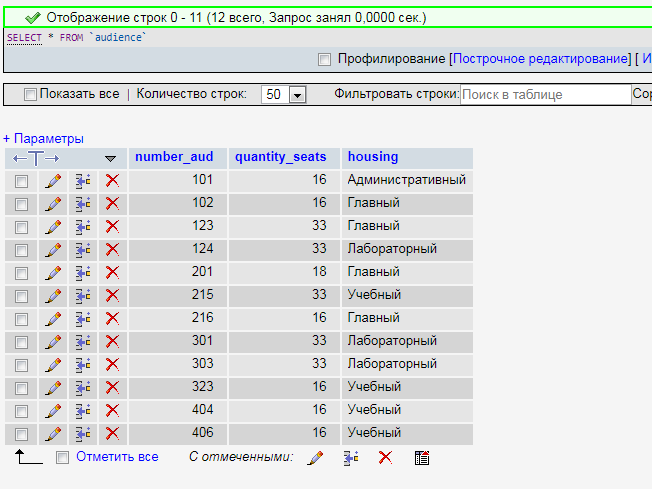


Рисунок 3.1 – Отображение таблицы аудиторий

Структура созданной базы в интерфейсе **phpMyAdmin** выглядит следующим образом, как показано на рис.3.2, для упрощения взаимодейстия и обработки информации базы данных.

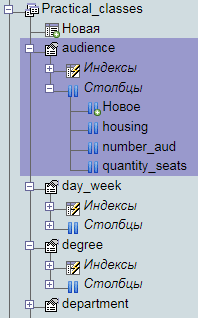
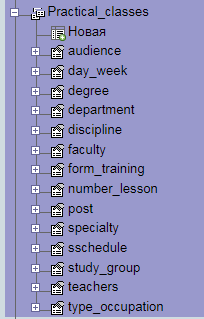


Рисунок 3.2. – Структура организация данных в базе данных

В ходе выполнения первичной настройки базы данных необходимо будет ввести данные, их типы, предельный размер, а так же указать ключи и индексы для создания связей между таблицами.

1. **РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ**

Клиентское приложение создавалось в среде разработки Visual Studio 2015 на языке программирования C#. Данное клиентское приложение разработано с помощью технологии **Windows Forms** и имеет 1 основную форму. В главном окне приложения располагаются вкладки: общее расписание, аудитории, преподаватели, занятия.

За соединение с базой данных отвечает библиотека **MySql.Data**, предназначенный для подключения к базе данных. Транзакцией управляет строка **public string**, предназначенный для явного управления транзакцией. Кроме того, в каждой форме имеется целый ряд компонентов **dataGridView** для взаимодействия с таблицами в базе данных. Исходный код всех модулей представлен ниже в приложении.

При переходе по первой вкладке (общее расписание) отображается таблица с полным расписанием ВУЗа и кнопка перезагрузки приложения. По переходу на вторую вкладку (аудитории) пользователю выводится таблица со всеми аудиториями и отсортированными по занятости, а так же предоставляется возможность удалить, изменить или добавить аудиторию, как показано на рис.4.1.

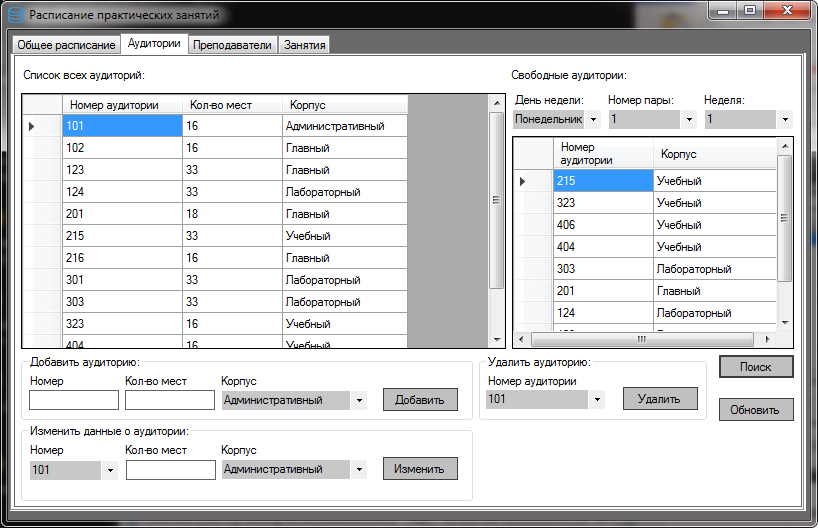


Рисунок 4.1. – Переход по вкладке «Аудитории»

По переходу на следующую вкладку (преподаватели) пользователю выводится таблица со всеми преподавателями ВУЗа с возможностью быстрого поиска по ФИО и кафедре, а так же удаление, изменение или добавление преподавателей, как показано на рис.4.2.

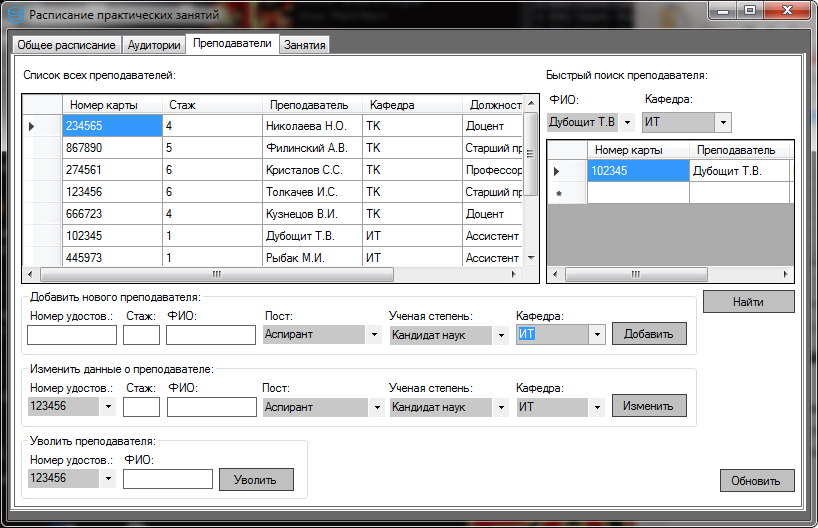


Рисунок 4.2. – Переход по вкладке «Преподаватели»

При переходе по последней вкладке (занятия) отображается таблица с всеми существующими занятиями ВУЗа и возможностью добавления нового занятия, как отображено на рис.4.3.

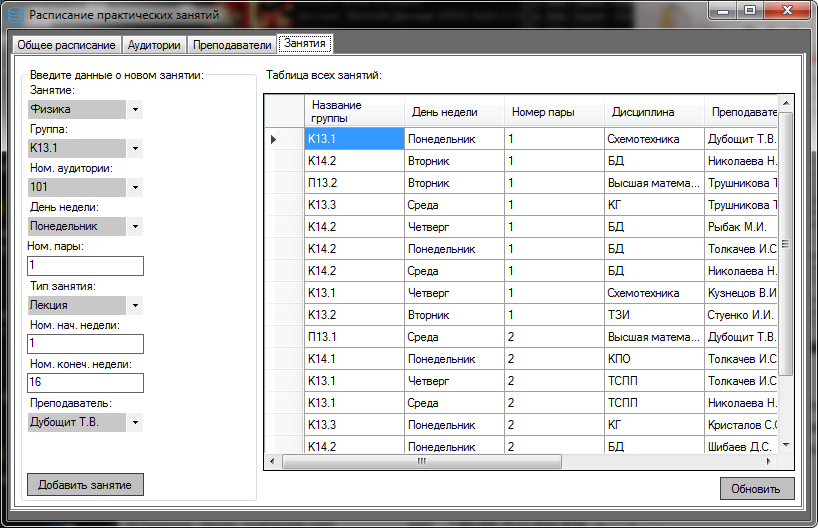


Рисунок 4.3. – Переход по вкладке «Занятия»

**ВЫВОД**

В курсовом проекте мною было проведено проектирование и создание базы данных, а также клиентского приложения в среде разработки Visual Studio 2012 в соответствии с заданием проекта.

При этом в процессе работы была создана база данных предназначенная для формирования и хранения данных по книгам, хранящимся в библиотеке. Кроме того, база данных позволяет хранить информацию количестве книг, их расположению на полке, а количество страниц в книге.

Созданное к базе данных клиентское приложение позволяет быстро находить нужные книги, а также вносить изменения в их данные.

**СПИСОК ЛИТЕРАТУРЫ**

1. Нейгел К. и др. C# 5.0 и платформа .NET 4.5 для профессионалов = Professional C# 5.0 and .NET 4.5. — М.: «Диалектика», 2013. — 1440 с.
2. Хейлсберг А., Торгерсен М., Вилтамут С., Голд П.. [Язык программирования C#. Классика Computers Science. 4-е издание](http://www.piter.com/book.phtml?978545900283) = C# Programming Language, 4th Ed. — СПб.: [«Питер»](https://www.wikiwand.com/ru/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)), 2012. — 784 с.
3. Стиллмен Э., Грин Дж.. [Изучаем C#. 2-е издание](http://www.piter.com/book.phtml?978544610105) = Head First C#, 2ed. — СПб.: [«Питер»](https://www.wikiwand.com/ru/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)), 2012. — 704 с.
4. Троелсен Э. [Язык программирования C# 5.0 и платформа .NET 4.5, 6-е издание](http://www.williamspublishing.com/Books/978-5-8459-1814-7.html) = Pro C# 5.0 and the .NET 4.5 Framework, 6th edition. — М.: «Вильямс», 2013. — 1312 с.

**ПРИЛОЖЕНИЕ**

**Код формы «Form1» (Библиотека)**

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;

using System.Data.SqlClient;

namespace KP

{

public partial class Form1 : Form

{

SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\lenovo\OneDrive\С\KP\KP\4Kurs.mdf;Integrated Security=True;Connect Timeout=30;");

string constring = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\lenovo\OneDrive\С\KP\KP\4Kurs.mdf;Integrated Security=True;Connect Timeout=30;";

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"INSERT INTO table1 VALUES ('" + textBox1.Text + "','" + textBox2.Text + "','" + textBox3.Text + "','" + textBox5.Text + "','" + textBox6.Text + "')";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

disp\_data();

MessageBox.Show("Записано в таблицу!");

}

//SqlCommand cmd = new SqlCommand(@"INSERT INTO [Table]

//(ItemID,ItemName) VALUES

//(@ItemIDss,@Namess)");

//(Id\_пользователя, Клиент, Название\_книги)

//private void FillSource()

// {

// }

public void disp\_data()

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa = new SqlDataAdapter(@"SELECT \* FROM table1", sqlCon);

DataTable dtbl = new DataTable();

sqlDa.Fill(dtbl);

dataGridView1.DataSource = dtbl;

}

//SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\lenovo\OneDrive\С\KP\KP\4Kurs.mdf;Integrated Security=True;Connect Timeout=30");

//con.Open();

//SqlCommand cmd = con.CreateCommand();

//cmd.CommandType = CommandType.Text;

//cmd.CommandText = @"select \* from table1";

//cmd.ExecuteNonQuery();

//DataTable dt = new DataTable();

//SqlDataAdapter da = new SqlDataAdapter(cmd);

//da.Fill(dt);

//con.Close();

//dataGridView1.DataSource = cmd;

}

private void button2\_Click(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa = new SqlDataAdapter(@"SELECT \* FROM table1", sqlCon);

DataTable dtbl = new DataTable();

sqlDa.Fill(dtbl);

dataGridView1.DataSource = dtbl;

}

}

private void Form1\_Load(object sender, EventArgs e)

{

disp\_data();

}

private void button3\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"DELETE FROM table1 WHERE Id\_пользователя='"+textBox1.Text+"'";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

disp\_data();

MessageBox.Show("Удаление выполнено!");

}

private void button4\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"UPDATE table1 set Название\_книги='"+textBox4.Text+"' where Название\_книги='" + textBox3.Text + "'";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

disp\_data();

MessageBox.Show("Обновление выполнено!");

}

private void button5\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

f2.Show();

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void tabPage1\_Click(object sender, EventArgs e)

{

}

public void disp\_data2()

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa1 = new SqlDataAdapter(@"SELECT \* FROM Table2", sqlCon);

DataTable dtbl1 = new DataTable();

sqlDa1.Fill(dtbl1);

dataGridView2.DataSource = dtbl1;

}

}

private void button10\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"INSERT INTO Table2 VALUES ('" + textBox12.Text + "','" + textBox11.Text + "','" + textBox15.Text + "','" + textBox8.Text + "','" + textBox7.Text + "','" + textBox13.Text + "','" + textBox14.Text + "','" + textBox10.Text + "')";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

textBox7.Text = "";

textBox8.Text = "";

textBox10.Text = "";

textBox11.Text = "";

textBox12.Text = "";

textBox13.Text = "";

textBox14.Text = "";

textBox15.Text = "";

disp\_data2();

MessageBox.Show("Записано в таблицу!");

}

private void button6\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

f2.Show();

}

private void button7\_Click(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa = new SqlDataAdapter(@"SELECT \* FROM Table2", sqlCon);

DataTable dtbl = new DataTable();

sqlDa.Fill(dtbl);

dataGridView2.DataSource = dtbl;

}

}

private void button8\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"DELETE FROM Table2 WHERE Шифр='" + textBox12.Text + "'";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

disp\_data2();

MessageBox.Show("Удаление выполнено!");

}

private void button9\_Click(object sender, EventArgs e)

{

con.Open();

SqlCommand cmd = con.CreateCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = @"UPDATE Table2 set Колво\_экземпляров='" + textBox10.Text + "' where Колво\_экземпляров='" + textBox9.Text + "'";

cmd.ExecuteNonQuery();

//cmd.ExecuteNonQueryAsync();

con.Close();

disp\_data2();

MessageBox.Show("Обновление выполнено!");

}

//private void textBox5\_TextChanged(object sender, EventArgs e)

//{

// te("Данная программа предоставляет возможность управления клиентами библиотеки, возможность уделние и добавления, а тажке исправления и поиска информации в локальной базе данных.");

//}

}

}

**Код формы «Form2» (Поисковая форма)**

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;

using System.Data.SqlClient;

namespace KP

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\lenovo\OneDrive\С\KP\KP\4Kurs.mdf;Integrated Security=True;Connect Timeout=30");

string constring = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\lenovo\OneDrive\С\KP\KP\4Kurs.mdf;Integrated Security=True;Connect Timeout=30";

private void Form2\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa = new SqlDataAdapter(@"SELECT \* FROM table1 where Название\_книги='"+textBox1.Text+"'", sqlCon);

DataTable dtbl = new DataTable();

sqlDa.Fill(dtbl);

dataGridView1.DataSource = dtbl;

}

}

private void button2\_Click(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection(constring))

{

sqlCon.Open();

SqlDataAdapter sqlDa1 = new SqlDataAdapter(@"SELECT \* FROM Table2 where Шифр='" + textBox2.Text + "'", sqlCon);

DataTable dtbl1 = new DataTable();

sqlDa1.Fill(dtbl1);

dataGridView2.DataSource = dtbl1;

}

}

}

}