**Министерсво образования и нАукИ украины**

**Одесский национальный МОРСКОЙ университет**

# КораблеСтроительный Факультет

**Кафедра технической кибернетики**

### Курсовая работа

### по базам данных

**«Библиотека. Библиотекарь отдела коплектации»**

**Тема 8. Задание 1.**

**Выполнил:**

**студент КСФ**

**3 курс 2 группа**

**Савичев М.Г.**

**Руководитель:**

**Ст. преподаватель**

**Рублев И.С.**

# Одесса 2017

ОГЛАВЛЕНИЕ

[ЗАДАЧА 3](#_Toc482690009)

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

[СОЗДАНИЕ БАЗЫ ДАННЫХ В SQL Server 7](#_Toc482690011)

[РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ 8](#_Toc482690012)

[ВЫВОД 10](#_Toc482690013)

[СПИСОК ЛИТЕРАТУРЫ 11](#_Toc482690014)

[ПРИЛОЖЕИЕ 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 требованиям на добавление, изменение и удаление данных.
* Те требования предметной области, которые не удается выполнить при построении логической модели БД, должны быть учтены путем ограничений ссылочной целостности таблиц БД, с помощью триггеров или хранимых процедур, либо при создании клиентского приложения.

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

**ОПИСАНИЕ ЗАДАЧИ:**

Пусть требуется разработать информационную систему для автоматизации учета получения и выдачи книг в библиотеке. Система должна предусматривать режимы ведения системного каталога, отражающего перечень областей знаний, по которым имеются книги в библиотеке. Внутри библиотеки области знаний в систематическом каталоге могут иметь уникальный внутренний номер и полное наименование. Каждая книга может содержать сведения из нескольких областей знаний. Каждая книга в библиотеке может присутствовать в нескольких экземплярах.

Каждая книга, хранящаяся в библиотеке, характеризуется следующими параметрами:

* уникальный шифр;
* название;
* фамилии авторов (могут отсутствовать);
* место издания (город);
* издательство;
* год издания;
* количество страниц;
* стоимость книги;
* количество экземпляров книги в библиотеке.

Книги могут иметь одинаковые названия, но они различаются по своему уникальному шифру (ISBN).

В библиотеке ведется картотека читателей.

На каждого читателя в картотеку заносятся следующие сведения:

* фамилия, имя, отчество;
* домашний адрес;
* телефон (будем считать, что у нас два телефона — рабочий и домашний);
* дата рождения.

Каждому читателю присваивается уникальный номер читательского билета.

Каждый читатель может одновременно держать на руках не более 5 книг. Читатель не должен одновременно держать более одного экземпляра книги одного названия.

Каждая книга в библиотеке может присутствовать в нескольких экземплярах.

Каждый экземпляр имеет следующие характеристики:

* уникальный инвентарный номер;
* шифр книги, который совпадает с уникальным шифром из описания книг;
* место размещения в библиотеке.

В случае выдачи экземпляра книги читателю в библиотеке хранится специальный вкладыш, в котором должны быть записаны следующие сведения:

* номер билета читателя, который взял книгу;
* дата выдачи книги;
* дата возврата.

Предусмотреть следующие ограничения на информацию в системе:

1. Книга может не иметь ни одного автора.

2. В библиотеке должны быть записаны читатели не моложе 17 лет.

3. В библиотеке присутствуют книги, изданные начиная с 1960 по текущий год.

4. Каждый читатель может держать на руках не более 5 книг.

5. Каждый читатель при регистрации в библиотеке должен дать телефон для связи: он может быть рабочим или домашним.

6. Каждая область знаний может содержать ссылки на множество книг, но каж­дая книга может относиться к различным областям знаний.

При работе с системой библиотекарь отдела комплектации должен решать следующие задачи:

1. Принимать новые книги и регистрировать их в библиотеке.
2. Относить книги к одной или к нескольким областям знаний.
3. Проводить инвентаризацию книг, то есть назначение новых инвентарных номеров вновь принятым книгам, и, помещая их на полки библиотеки, запоминать место размещения каждого экземпляра.
4. Проводить дополнительную инвентаризацию, если поступило несколько экземпляров книги, которая уже есть в библиотеке, при этом информация о книге в предметный каталог не вносится, а каждому новому экземпляру присваивается новый инвентарный номер и для него определяется место на полке библиотеки.
5. **На основании физической модели создать соответствующие таблицы БД «Bibl2A».**

Для всех таблиц создать необходимые триггера и одну хранимую процедуры (генерация экзаменационной ведомости).

1. **В Visual Studio создать программу-клиент для «Exam1A.**

Программа должна иметь следующие формы:

**Главная форма** – инвентаризации книг, добавление новых экземпляров книг.

Позволяет проводить фильтрацию/отбор данных по ISBN, по Названию, по Автору.

**Вспомогательные формы**:

Ввода и редактирования данных о Названиях и ISBN изданий книг

**Форма - отчет**

Список экземпляров книг заданного Названия

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. Описание и текст созданных триггеров, индексов и хранимых процедур БД.

Файлы программы-клиента. Требование динамической регистрации алиаса БД в BDE при запуске программы. 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: description, books, ticket, readers.

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. В качестве примера возьмем таблицу «books», у которой есть 4 поля, одно из которых является первичным ключом – ninv.

**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. Например, связи могут выражаться следующими фразами – экзаменов много, а экзаменационная ведомость одна, то есть перспектива связи **один ко многим**.

В сущностях, созданных мною присутствуют следующие атрибуты:

**Description:**

Crypt- уникальный шифр;

Name-название;

Author-фамилии авторов;

City-место издания;

Publisher-издательство;

Date-год издания;

sheets – количество страниц;

price – стоимость книги;

amount – количество экземпляров книги;

**Books:**

ninv – Уникальный инвентарный номер;

crypt – Уникальный шифр;

location – место размещения в библиотеке;

tags – Области знаний;

**ticket:**

id – уникальный номер читательского билета;

ninv – Уникальный инвентарный номер;

issuedate – Дата выдачи книги;

returndate – Дата возврата;

**Readers:**

id – уникальный номер читательского билета;

fio – Фамилия, имя, отчество;

adress – Домашний адрес;

number - Телефон;

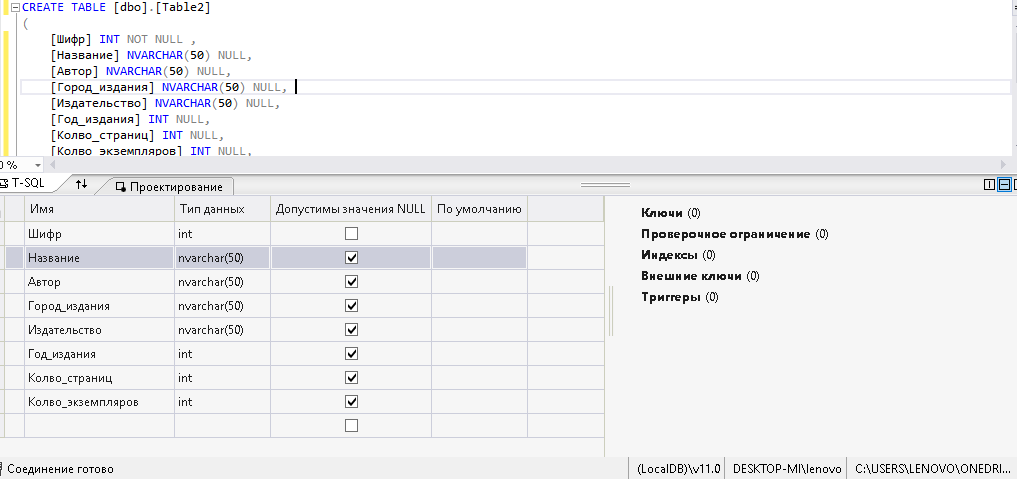
date – Дата рождения;

В общем виде вся логическая структура моей базы данных выглядит так:

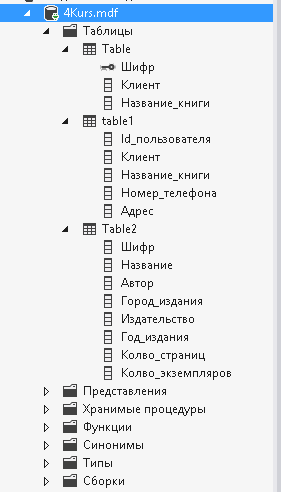
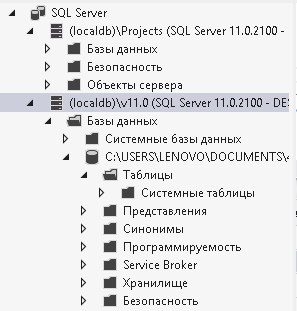


1. **СОЗДАНИЕ БАЗЫ ДАННЫХ В SQL Server**

Создание базы данных изначально велось в **SQL Server,** который встроен в Framework C#, что облегчает работу с данными на всех уровнях разработки.



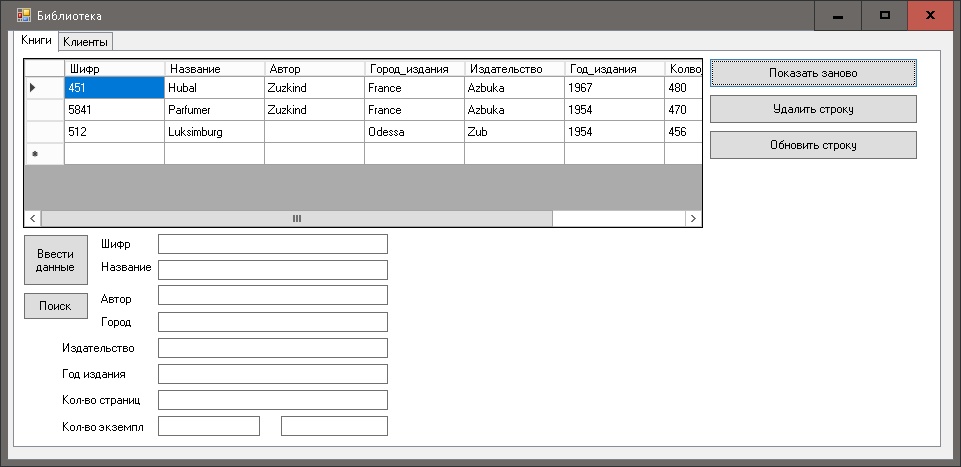
Созданная база в интерфейсе С# выглядит следующим образом для упрощения взаимодейстия и обработки информации базы данных.

В ходе выполнения первичной настройки базы данных необходимо будет ввести данные, их типы и предельный размер.

1. **РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ**

Клиентское приложение создавалось мною в среде разработки Visual Studio 2012 на языке программирования C#. Данное клиентское приложение разработано с помощью технологии **MySqlClient** и имеет 2 формы: Библиотека, внутри которой разбито на два окна две базы данных, а также форма поиска, необходимая для параллельного поиска информации о Клиентах и Книгах.

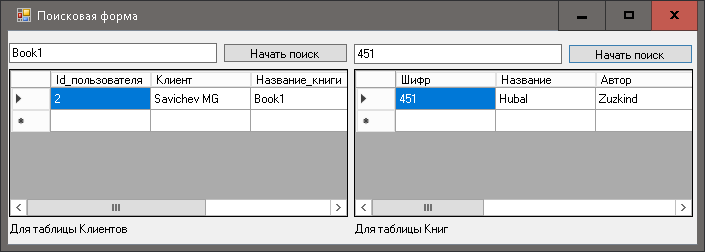
Главной формой является форма, где реализовано инвентаризация книг, добавление новых экземпляров книг. Позволяет проводить фильтрацию/отбор данных по ISBN, по Названию, по Автору.

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.

Не визуальный компонент **GetSet** **в C#** представляет собой источник данных, который обеспечивает связь между набором данных и компонентами отображения и редактирования данных. Текст всех модулей представлен ниже в приложении.

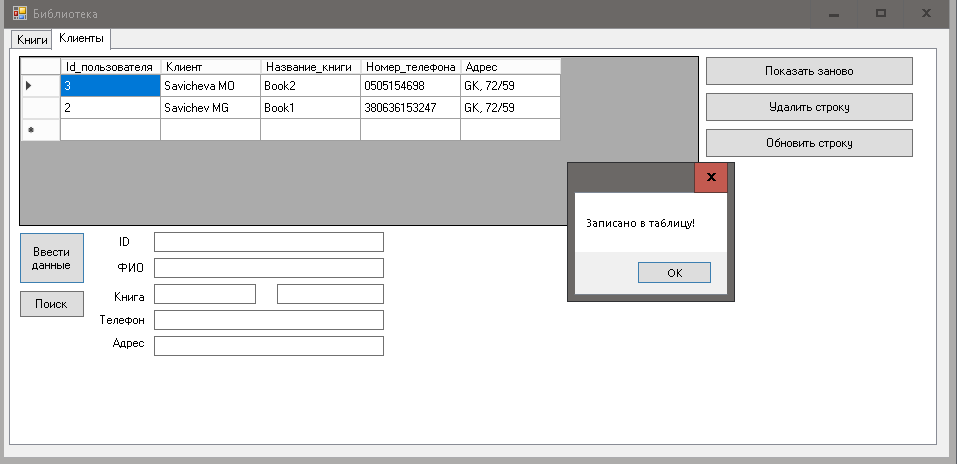
При загрузке приложения сразу появляется главная форма - форма «Библиотека».

На ней расположена таблица, в которой отображаются книги, которые есть в библиотеке, на втором табе можно увидеть Клиентов библиотеки, а также возможность добавить новые экземпляры книг в библиотеку, доступ к параллельному поиску пользователей и книг.

**Форма «Поисковая форма» выводит список книг, или пользователей, основываясь на шифре книги или названию присвоенной пользователю книги из библиотеки.**

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.

На втором табе можно увидеть список клиентов библиотеки, а также произвести поиск и добавление новой информации в таблицу.



1. **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.

1. **СПИСОК ЛИТЕРАТУРЫ**
2. 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.
3. 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.
4. 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.
5. 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.
6. **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;

}

}

}

}