МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ГЛАВНОЕ УПРАВЛЕНИЕ ПО ОБРАЗОВАНИЮ

МОГИЛЕВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**БАЗЫ ДАННЫХ И**

**СИСТЕМЫ УПРАВЛЕНИЯ**

**БАЗАМИ ДАННЫХ**

ДОМАШНЯЯ КОНТРОЛЬНАЯ РАБОТА №2

Выполнил Учащийся группы ПО-455

И. А. Дудкин

Шифр 16

2023

**12 Опишите операторы присваивания, процесс объявления переменных SQL.**

Локальная переменная Transact-SQL представляет собой объект, содержащий одно значение определенного типа. Переменные обычно используются в пакетах и скриптах:

* + в качестве счетчика цикла;
  + для хранения значения, которое необходимо проверить инструкцией управления потоком;
  + для хранения значения, возвращенного функцией или хранимой процедурой.

Имена некоторых системных функций Transact-SQL начинаются с двух символов @ (@@). Хотя в предыдущих версиях сервера SQL Server @@функции называются глобальными переменными, @@функции не являются переменными и используются иначе. Функции, помеченные символами @@, являются системными, и синтаксис их использования соответствует правилам вызова функций.

* В представлении нельзя использовать переменные.
* Откат транзакции не влияет на изменения переменных.

Следующий скрипт создает небольшую тестовую таблицу из 26 строк. Переменная используется в скрипте в качестве:

* счетчика цикла для управления количеством вставляемых строк;
* значения, вставляемого в столбец целочисленного типа;
* аргумента функции, формирующей строку, которая вставляется в столбец символьного типа:

-- Create the table.

CREATE TABLE TestTable (cola INT, colb CHAR(3));

GO

SET NOCOUNT ON;

GO

-- Declare the variable to be used.

DECLARE @MyCounter INT;

-- Initialize the variable.

SET @MyCounter = 0;

-- Test the variable to see if the loop is finished.

WHILE (@MyCounter < 26)

BEGIN;

-- Insert a row into the table.

INSERT INTO TestTable VALUES

-- Use the variable to provide the integer value

-- for cola. Also use it to generate a unique letter

-- for each row. Use the ASCII function to get the

-- integer value of 'a'. Add @MyCounter. Use CHAR to

-- convert the sum back to the character @MyCounter

-- characters after 'a'.

(@MyCounter,

CHAR( ( @MyCounter + ASCII('a') ) )

);

-- Increment the variable to count this iteration

-- of the loop.

SET @MyCounter = @MyCounter + 1;

END;

GO

SET NOCOUNT OFF;

GO

-- View the data.

SELECT cola, colb

FROM TestTable;

GO

DROP TABLE TestTable;

GO

Инструкция DECLARE инициализирует переменную Transact-SQL следующим образом:

* Назначение имени. Первым символом имени должен быть одиночный символ @.
* Назначение длины и типа данных, определяемого системой или пользователем. Для числовых переменных задаются также точность и масштаб. Для переменных типа XML может быть дополнительно задана коллекция схем.
* Присваивает созданной переменной значение NULL.

Например, следующая инструкция DECLARE создает локальную переменную @mycounter типа int.

SQL

DECLARE @MyCounter INT;

Инструкция DECLARE позволяет объявить несколько переменных одинакового или разного типов через запятую.

Например, следующая инструкция DECLARE создает три локальные переменные с именем @LastName, @FirstName и @StateProvince, присваивая каждой из них значение NULL:

DECLARE @LastName NVARCHAR(30), @FirstName NVARCHAR(20), @StateProvince NCHAR(2);

Областью видимости переменной называют диапазон инструкций Transact-SQL, которые могут к ней обращаться. Областью видимости переменной являются все инструкции между ее объявлением и концом пакета или хранимой процедуры, где она объявлена.

При объявлении переменной присваивается значение NULL. Чтобы изменить значение переменной, применяется инструкция SET. Этот способ присвоения значений переменным является предпочтительным. Кроме того, переменной можно присвоить значение, указав ее в списке выбора инструкции SELECT.

Чтобы присвоить значение переменной при помощи инструкции SET, необходимо указать ее имя и присваиваемое значение. Этот способ присвоения значений переменным является предпочтительным. Например, следующий пакет объявляет две переменные, присваивает им значения и использует их в предложении WHERE инструкции SELECT:

USE AdventureWorks2022;

GO

-- Declare two variables.

DECLARE @FirstNameVariable NVARCHAR(50),

@PostalCodeVariable NVARCHAR(15);

-- Set their values.

SET @FirstNameVariable = N'Amy';

SET @PostalCodeVariable = N'BA5 3HX';

-- Use them in the WHERE clause of a SELECT statement.

SELECT LastName, FirstName, JobTitle, City, StateProvinceName, CountryRegionName

FROM HumanResources.vEmployee

WHERE FirstName = @FirstNameVariable

OR PostalCode = @PostalCodeVariable;

GO

Переменной можно присвоить значение, указав ее в списке выбора. Если список выбора ссылается на переменную, то ей должно быть присвоено скалярное значение, или инструкция SELECT должна возвращать только одну строку. Пример:

USE AdventureWorks2022;

GO

DECLARE @EmpIDVariable INT;

SELECT @EmpIDVariable = MAX(EmployeeID)

FROM HumanResources.Employee;

GO

**49 Опишите синтаксические конструкции и переменные РНР.**

PHP является языком с динамической типизацией. Это значит, что тип данных переменной выводится во время выполнения, и в отличии от ряда других языков программирования в PHP не надо указывать перед переменной тип данных.

В PHP есть десять базовых типов данных:

bool (логический тип)

int (целые числа)

float (дробные числа)

string (строки)

array (массивы)

object (объекты)

callable (функции)

mixed (любой тип)

resource (ресурсы)

null (отсутствие значения)

Из этих типов данных первые четыре являются скалярными: bool, int, float, string. Поэтому вначале именно их и рассмотрим.

int (целочисленный тип)

Представляет целое число со знаком.

$num = -100;

echo $num;

Здесь переменная $num представляет целочисленный тип, так как ей присваивается целочисленное значение.

Кроме десятичных целых чисел PHP обладает возможностью использовать также двоичные, восьмеричные и шестнадцатеричные числа. Шаблоны чисел для других систем:

шестнадцатеричные : 0[xX][0-9a-fA-F]

восьмеричные : 0[0-7]

двоичные : 0b[01]

<?php

// Все числа в десятичной системе имеют значение 28

$num\_10 = 28; // десятичное число

$num\_2 = 0b11100; // двоичное число (28 в десятичной системе)

$num\_8 = 034; // восьмеричное число (28 в десятичной)

$num\_16 = 0x1C; // шестнадцатиричное число (28 в десятичной)

echo "num\_10 = $num\_10 <br>";

echo "num\_2 = $num\_2 <br>";

echo "num\_8 = $num\_8 <br>";

echo "num\_16 = $num\_16";

?>

Переменная типа int занимает в памяти 32 бита, то есть может принимаь значения от -2 147 483 648 до 2 147 483 647. Если переменная получает числовое значение вне этого диапазона, то она трактуется как переменная типа float

Тип float (числа с плавающей точкой)

Размер числа с плавающей точкой зависит от платформы. Максимально возможное значение, как правило, составляет 1.8\*10308 с точностью около 14 десятичных цифр. Например:

<?php

$a1 = 1.5;

$a2 = 1.3e4; // 1.3 \* 10^4

$a3 = 6E-8; // 0.00000006

?>

Тип bool (логический тип)

Переменные логического типа могут принимать два значения: true и false или иначе говоря истина и ложь. Чаще всего логические значения используются в условных конструкциях:

<?php

$foo = true;

$boo = false;

?>

Тип string (строки)

Для работы с текстом можно применять строки. Строки бывают двух типов: в двойных кавычках и одинарных. От типа кавычек зависит обработка строк интерпретатором. Так, переменные в двойных кавычках заменяются значениями, а переменные в одинарных кавычках остаются неизменными.

<?php

$a=10;

$b=5;

$result = "$a+$b <br>";

echo $result;

$result = '$a+$b';

echo $result;

?>

Специальное значение null

Значение null указывает, переменная не имеет значения. Использование данного значения полезно в тех случаях, когда мы хотим указать, что переменная не имеет значения. Например, если мы просто определим переменную без ее инициализации, и затем попробуем ее использовать, то нам интерпретатор выдаст сообщение, что переменная не установлена:

<?php

$a = null;

echo "a = $a";

?>

**88 «Учебное заведение»**

1) Для создания информационной системы спроектируйте базу данных в Microsoft SQL Server, содержащую не менее трех таблиц. Установите отношения между таблицами.

2) Заполните таблицы данными (не менее 10 записей).

3) На языке SQL напишите запросы:

- на вывод некоторых полей из двух таблиц;

- на вывод данных по условию, представляющему выражение:

* 1. типа сравнения;
  2. с логическим оператором «И»;
  3. с логическим оператором «ИЛИ»;

- с вычислениями над полями БД;

- параметрический.

4) Разработать форму на языке C#, содержащую все созданные объекты БД.

5) Реализовать добавление, удаление изменение данных в таблице, а также поиск и фильтрацию данных.

**Решение**

Создадим форму и разместим на ней соответствующие элементы управления: поля ввода данных и кнопки для добавления, редактирования и удаления записи, а так же кнопки для выполнения заданий. Внешний вид формы представлен на рисунке 1.

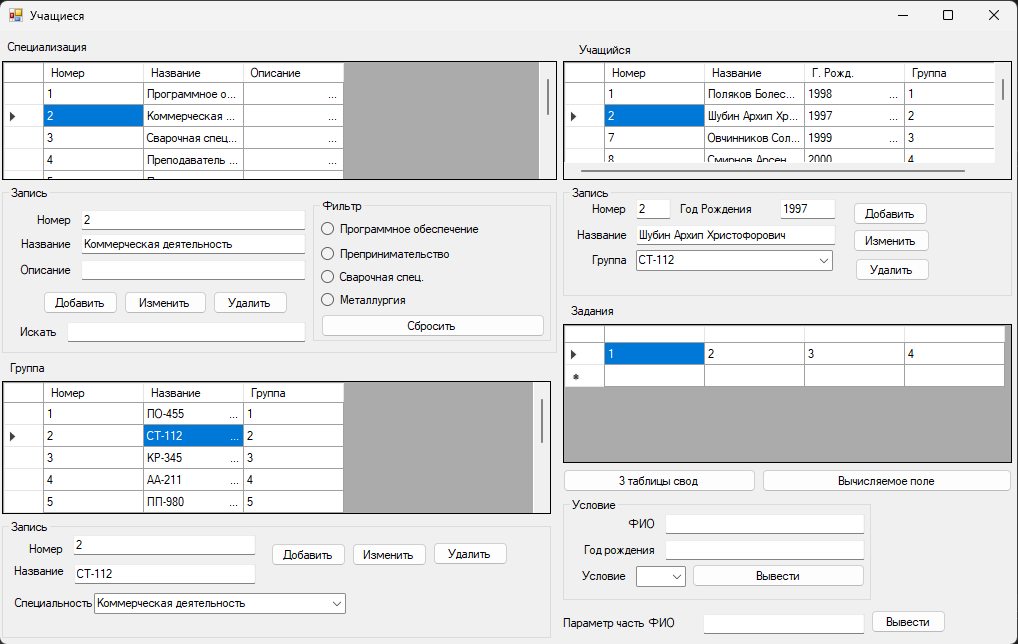


Рисунок 1 Внешний вид основной формы

Создадим вторую форму для добавления специализаций, и разместим на ней элементы управления (рисунок 2)

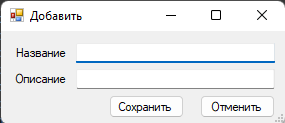


Рисунок 2 Внешний вид формы добавления специализации

Заполним таблицы данными, у нас это будут таблицы: Специализация, Студент и Группа.

На рисунке 3 представлен внешний вид приложения в котором выполнено задание вывода данных из трех таблиц таблиц.

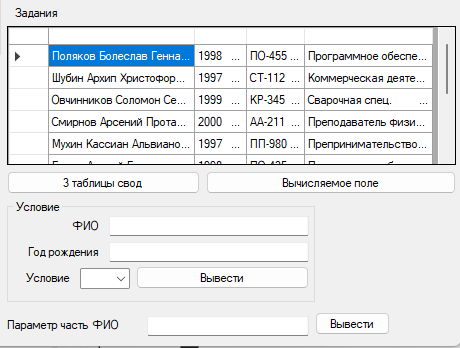


Рисунок 3 Приложение вкладка выполнения заданий

На рисунке 4 представлен результат выполнения действий над полями таблицы Студент, выведено ФИО студента, год рождения, год рождения минус 1000 и поясняющая надпись.

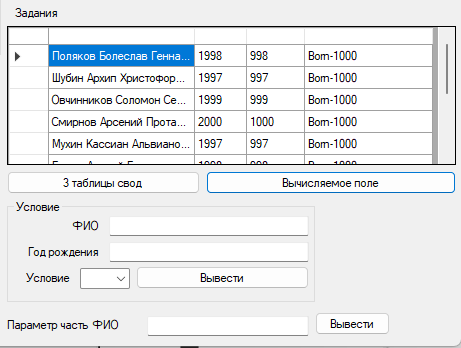


Рисунок 4 Результат выполнения задания вычисляемого поля

На рисунке 5 представлен вид приложения с выполненным заданием выбора студента по ФИО, либо по году его рождения.

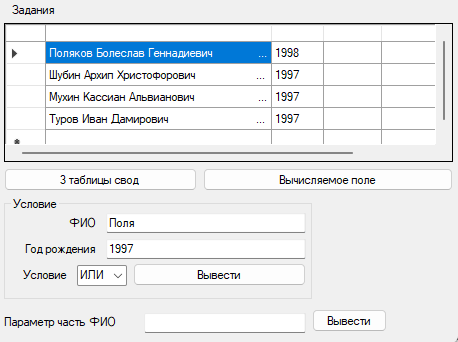


Рисунок 5 Результат выполнения задания Условие

На рисунке 6 представлен вывод задания с параметром, в качестве параметра часть ФИО студента.

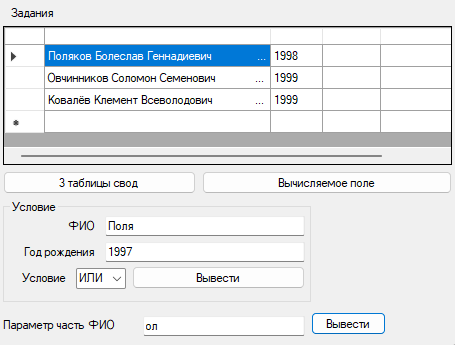
****

Рисунок 6 Результат выполнения задания Параметр

На рисунке 7 представлен результат выполнения задания Фильтр по специальности.

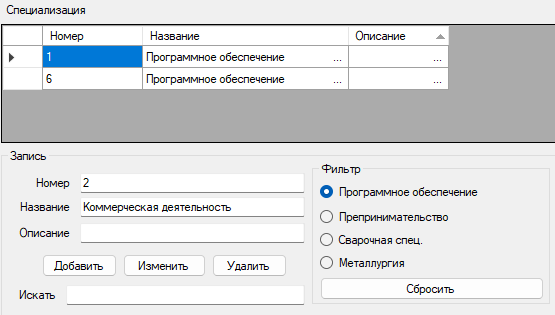
****

Рисунок 7результат выполнения задания Фильтр

На рисунке 8 представлен результат выполнения задания Поиск

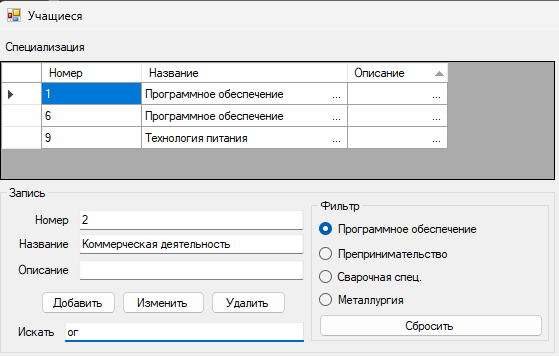


Рисунок 8результат выполнения задания Поиск

**Код программы**

**Главная Форма**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Fakultet

{

public partial class Form1 : Form

{

DataBase database = new DataBase();

int selectedRow;

public Form1()

{

InitializeComponent();

CreateColumns();

refreshDG(dataGridView1);

refreshDG2(dataGridView2);

refreshDG3(dataGridView3);

refreshDG4(dataGridView4,$"select TOP (1) '1' as Col1, '2' as col2, '3' as col3, '4' as col4 from [SPEC]");

}

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

{

CreateColumns();

refreshDG(dataGridView1);

refreshDG2(dataGridView2);

refreshDG3(dataGridView3);

//refreshDG4(dataGridView4,$"select '1' as Col1, '2' as col2, '3' as col3, '4' as col4 from [SPEC]");

}

private void CreateColumns()

{

dataGridView1.Columns.Add("ID", "Номер");

dataGridView1.Columns.Add("Name", "Название");

dataGridView1.Columns.Add("Descr", "Описание");

//dataGridView1.Columns.Add("IsNew", String.Empty);

dataGridView2.Columns.Add("ID", "Номер");

dataGridView2.Columns.Add("Name", "Название");

dataGridView2.Columns.Add("Born", "Г. Рожд.");

dataGridView2.Columns.Add("IDGroup", "Группа");

//dataGridView2.Columns.Add("IsNew", String.Empty);

dataGridView3.Columns.Add("ID", "Номер");

dataGridView3.Columns.Add("Name", "Название");

dataGridView3.Columns.Add("IDSPEC", "Группа");

//dataGridView3.Columns.Add("IsNew", String.Empty);

dataGridView4.Columns.Add("Col1", String.Empty);

dataGridView4.Columns.Add("Col2", String.Empty);

dataGridView4.Columns.Add("Col3", String.Empty);

dataGridView4.Columns.Add("Col4", String.Empty);

}

private void ReadSinglRow(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetInt32(0), record.GetString(1), record.GetString(2));

}

private void ReadSinglRow2(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetInt32(0), record.GetString(1), record.GetString(2), record.GetInt32(3));

}

private void ReadSinglRow3(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetInt32(0), record.GetString(1), record.GetInt32(2));

}

private void ReadSinglRow4(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetString(0), record.GetString(1), record.GetString(2), record.GetString(3));

}

private void refreshDG(DataGridView dgw)

{

dgw.Rows.Clear();

string querystr = $"Select \* from [Spec]";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dgw, reader);

}

reader.Close();

}

private void refreshDG2(DataGridView dgw)

{

dgw.Rows.Clear();

string querystr = $"Select \* from [Student]";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow2(dgw, reader);

}

reader.Close();

}

private void refreshDG3(DataGridView dgw)

{

dgw.Rows.Clear();

string querystr = $"Select \* from [Group]";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow3(dgw, reader);

}

reader.Close();

}

private void refreshDG4(DataGridView dgw, string s)

{

dgw.Rows.Clear();

string querystr = s;

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow4(dgw, reader);

}

reader.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

/\*

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

selectedRow = e.RowIndex;

if (e.RowIndex >= 0)

{

DataGridViewRow row = dataGridView1.Rows[selectedRow];

textBox1.Text = row.Cells[0].Value.ToString();

textBox2.Text = row.Cells[1].Value.ToString();

textBox3.Text = row.Cells[2].Value.ToString();

}

}

\*/

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

{

Form2 f\_add = new Form2();

f\_add.ShowDialog();

refreshDG(dataGridView1);

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

DataGridView dwg = dataGridView1;

dwg.Rows.Clear();

string querystr = $" select \* from [Spec] Where concat(Name,Descr) like '%" + textBox4.Text + "%'";

SqlCommand com = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dwg, reader);

}

reader.Close();

}

private void radioButton1\_Click(object sender, EventArgs e)

{

DataGridView dwg = dataGridView1;

dwg.Rows.Clear();

string querystr = $" select \* from [Spec] Where Name = 'Программное обеспечение'";

SqlCommand com = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dwg, reader);

}

reader.Close();

}

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

{

refreshDG(dataGridView1);

}

private void button3\_Click\_1(object sender, EventArgs e)

{

Form2 f\_add = new Form2();

f\_add.ShowDialog();

refreshDG(dataGridView1);

}

private void dataGridView1\_CellClick\_1(object sender, DataGridViewCellEventArgs e)

{

selectedRow = e.RowIndex;

if (e.RowIndex >= 0)

{

DataGridViewRow row = dataGridView1.Rows[selectedRow];

textBox1.Text = row.Cells[0].Value.ToString();

textBox2.Text = row.Cells[1].Value.ToString();

textBox3.Text = row.Cells[2].Value.ToString();

}

}

private void radioButton2\_Click(object sender, EventArgs e)

{

DataGridView dwg = dataGridView1;

dwg.Rows.Clear();

string querystr = $" select \* from [Spec] Where Name = 'Препринимательство'";

SqlCommand com = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dwg, reader);

}

reader.Close();

}

private void radioButton4\_Click(object sender, EventArgs e)

{

DataGridView dwg = dataGridView1;

dwg.Rows.Clear();

string querystr = $" select \* from [Spec] Where Name = 'Металлургия'";

SqlCommand com = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dwg, reader);

}

reader.Close();

}

private void radioButton3\_Click(object sender, EventArgs e)

{

DataGridView dwg = dataGridView1;

dwg.Rows.Clear();

string querystr = $" select \* from [Spec] Where Name = 'Сварочная спец.'";

SqlCommand com = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

ReadSinglRow(dwg, reader);

}

reader.Close();

}

private void button2\_Click\_1(object sender, EventArgs e)

{

var addQuery = $"delete from [Spec] where ID={textBox1.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG(dataGridView1);

}

private void button1\_Click\_1(object sender, EventArgs e)

{

var addQuery = $"update [Spec] set Name='{textBox2.Text}' , Descr = '{textBox3.Text}' where ID = {textBox1.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG(dataGridView1);

}

private void Form1\_Load\_1(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "fAKULTETDataSet.SPEC". При необходимости она может быть перемещена или удалена.

this.sPECTableAdapter.Fill(this.fAKULTETDataSet.SPEC);

// TODO: данная строка кода позволяет загрузить данные в таблицу "fAKULTETDataSet.GROUP". При необходимости она может быть перемещена или удалена.

this.gROUPTableAdapter.Fill(this.fAKULTETDataSet.GROUP);

// TODO: данная строка кода позволяет загрузить данные в таблицу "databaseDataSet1.Group". При необходимости она может быть перемещена или удалена.

//this.groupTableAdapter.Fill(this.databaseDataSet1.Group);

// TODO: данная строка кода позволяет загрузить данные в таблицу "databaseDataSet1.SPEC". При необходимости она может быть перемещена или удалена.

//this.SPECTableAdapter.Fill(this.databaseDataSet1.SPEC);

}

private void dataGridView2\_CellClick(object sender, DataGridViewCellEventArgs e)

{

selectedRow = e.RowIndex;

if (e.RowIndex >= 0)

{

DataGridViewRow row = dataGridView2.Rows[selectedRow];

textBox5.Text = row.Cells[0].Value.ToString();

textBox6.Text = row.Cells[1].Value.ToString();

textBox7.Text = row.Cells[2].Value.ToString();

comboBox1.SelectedIndex = Int32.Parse(row.Cells[3].Value.ToString())-1;

}

}

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

{

var addQuery = $"insert into [Student] (Name,Born, IdGroup) values ('{textBox6.Text}' , '{textBox7.Text}',{comboBox1.SelectedIndex+1})";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG2(dataGridView2);

}

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

{

var addQuery = $"delete from [Student] where ID={textBox5.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG2(dataGridView2);

}

private void dataGridView3\_CellClick(object sender, DataGridViewCellEventArgs e)

{

selectedRow = e.RowIndex;

if (e.RowIndex >= 0)

{

DataGridViewRow row = dataGridView3.Rows[selectedRow];

textBox8.Text = row.Cells[0].Value.ToString();

textBox9.Text = row.Cells[1].Value.ToString();

comboBox2.SelectedIndex = Int32.Parse(row.Cells[2].Value.ToString())-1;

}

}

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

{

var addQuery = $"insert into [Gruop] (Name,IdSpec) values ('{textBox9.Text}' ,{comboBox2.SelectedIndex + 1})";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG3(dataGridView3);

}

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

{

var addQuery = $"delete from [Group] where ID={textBox8.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG3(dataGridView3);

}

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

{

var addQuery = $"update [Student] set Name='{textBox6.Text}' , Born = '{textBox7.Text}', IdGroup = '{comboBox1.SelectedIndex+1}' where ID = {textBox5.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG2(dataGridView2);

}

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

{

var addQuery = $"update [Group] set Name='{textBox9.Text}' , IdSpec = '{comboBox2.SelectedIndex+1}' where ID = {textBox8.Text}";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

refreshDG3(dataGridView3);

}

private void button11\_Click(object sender, EventArgs e)

{

dataGridView4.Rows.Clear();

string querystr = $"select s.fio as c1, s.born as c2, g.name as c3, d.name as c4 from [SPEC] d, [Student] s, [Group] g where g.idSpec=d.id and s.idGroup=g.id";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow4(dataGridView4, reader);

}

reader.Close();

}

private void button12\_Click(object sender, EventArgs e)

{

dataGridView4.Rows.Clear();

string querystr = $"select fio as col1, CAST(born AS varchar(5)) as col2, CAST(born-1000 AS varchar(5)) as col3, 'Born-1000' as col4 from [Student] s";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow4(dataGridView4, reader);

}

reader.Close();

}

private void button13\_Click(object sender, EventArgs e)

{// условие

dataGridView4.Rows.Clear();

string s = "OR";

if (comboBox3.SelectedIndex == 0) {

s = "AND"; }

string n = '%'+textBox10.Text+'%';

string l = textBox11.Text;

string querystr = $"select fio as c1, CAST(born AS varchar(5)) as c2, '' as c3, '' as c4 from [Student] s where fio like '{n}' {s} born='{l}'";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow4(dataGridView4, reader);

}

reader.Close();

}

private void button14\_Click(object sender, EventArgs e)

{

dataGridView4.Rows.Clear();

string n = '%'+textBox12.Text+'%';

string querystr = $"select fio as c1, CAST(born AS varchar(5)) as c2, '' as c3, '' as c4 from [Student] s where fio like'{n}'";

SqlCommand command = new SqlCommand(querystr, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSinglRow4(dataGridView4, reader);

}

reader.Close();

}

class DataBase

{

SqlConnection sqlConnection = new SqlConnection(@"Data Source=localhost\SQLEXPRESS;Initial Catalog=FAKULTET;Integrated Security=True");

public void openConnection()

{

if (sqlConnection.State == System.Data.ConnectionState.Closed)

{

sqlConnection.Open();

}

}

public void closeConnection()

{

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

{

sqlConnection.Close();

}

}

public SqlConnection getConnection()

{

return sqlConnection;

}

}

}

**Форма 2**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Fakultet

{

public partial class Form2 : Form

{

DataBase dataBase = new DataBase();

public Form2()

{

InitializeComponent();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

dataBase.openConnection();

var addQuery = $"insert into [Spec] (Name,Descr) values ('{textBox1.Text}' , '{textBox2.Text}')";

var command = new SqlCommand(addQuery, dataBase.getConnection());

command.ExecuteNonQuery();

dataBase.closeConnection();

this.Close();

}

private void button2\_Click\_1(object sender, EventArgs e)

{

this.Close();

}

}

}

**Список использованных источников**

1. Дейт, К.Дж. Введение в системы баз данных/ К.Дж.Дейт. – М.: Вильямс, 2018
2. Дюбуа, Поль. MySQI / Поль Дюбуа. – 3-е изд. – М.: Издательский дом "Вильямс", 2007. - 816 с.: ил.
3. Колисниченко, Д.Н. Профессиональное программирование на PНP / Д.Н. Колисниченко. – Санкт-Петербург: БХВ-Петербург, 2007. –416 с.
4. Роб, П. Системы баз данных: проектирование, реализация и управление / П.Роб, К.Коронел. – 5-е изд., перераб. и доп.: пер. с англ. – Санкт-Петербург: БХВ-Петербург, 2004. - 1040 с.: ил
5. Хернандес, М.Дж. SQL-запросы для простых смертных: практическое руководство по манипулированию данными в SQL / М.Дж.Хернандес, Дж.Л.Вьескас. – Москва: Лори, 2000. - 473 с.