Лабораторная работа №3

Выполнила:

Группа:

Требования

1. Выполняемый вариант;
2. Задание;
3. Измененная процедура второго запроса;
4. Созданные формы;
5. Скриншоты результатов тестирования.

Выполняемый вариант

Вариант №2: «Успеваемость студентов»:

Имеются группы (шифр, год поступления, номер), студенты (номер зачетки, ФИО, дата рождения, пол) и предметы (наименование, ФИО преподавателя). Студенты входят в состав определенной группы, предмет изучается группами студентов в определенном количестве часов и заканчивается определенным видом аттестации (зачетом или экзаменом). Каждый студент получает оценку или признак зачета /незачета по каждому предмету.

Выходные документы:

* для указанной группы выдать по каждому предмету список студентов с их оценками по экзамену и со средним баллом группы, упорядоченный по фамилиям студентов и по убыванию среднего балла каждого предмета;
* ведомости по зачетам и экзаменам по каждому предмету каждой группы, упорядоченные по ФИО студентов с подсчетом количества должников.

Задание

1. Создание клиентского приложения с главной формой, содержащей меню, из которого вызываются все остальные формы;
2. Создание форм для отображения и всех видов корректировки (добавление, удаление, замена) каждой таблицы с выдачей сообщений об исключениях, возникающих в хранимых процедурах на сервере;
3. Создание форм для вывода двух выходных документов в форме листа Excel.

Измененная процедура второго запроса

ALTER PROCEDURE [dbo].[Procedure\_2]

AS

SELECT Groups.Code\_of\_group, Subjects.Name\_of\_subject, Students.Full\_name, Student\_has\_Subjects.Result\_of\_certification,

SUM(CASE WHEN LOWER(Student\_has\_Subjects.Result\_of\_certification)='not credit' OR Student\_has\_Subjects.Result\_of\_certification='0' OR Student\_has\_Subjects.Result\_of\_certification='1' OR Student\_has\_Subjects.Result\_of\_certification='2' THEN 1 ELSE 0 END)

OVER(PARTITION BY Groups.Code\_of\_group,Subjects.Name\_of\_subject) AS TOTAL\_BAD\_MARKS\_FOR\_GROUP

FROM Students, Groups, Student\_has\_Subjects,Subjects

WHERE Students.Groups\_Cipher\_of\_group=Groups.Cipher\_of\_group AND Students.Code\_of\_record\_book=Student\_has\_Subjects.Students\_Code\_of\_record\_book AND

Subjects.Name\_of\_subject IN(SELECT Name\_of\_subject FROM Subjects WHERE Subjects.Cipher\_of\_subject=Student\_has\_Subjects.Subjects\_Cipher\_of\_subject)

ORDER BY Groups.Code\_of\_group ASC, Subjects.Name\_of\_subject ASC, Students.Full\_name ASC;

RETURN 0

Созданные формы

Меню

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated with medium confidence

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

{

Groups sendTo = new Groups();

sendTo.ShowDialog();

}

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

{

Subjects sendTo = new Subjects();

sendTo.ShowDialog();

}

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

{

Students sendTo = new Students();

sendTo.ShowDialog();

}

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

{

Groups\_has\_Subj sendTo = new Groups\_has\_Subj();

sendTo.ShowDialog();

}

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

{

Student\_has\_Subj sendTo = new Student\_has\_Subj();

sendTo.ShowDialog();

}

private void запрос1ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Task1 sendTo = new Task1();

sendTo.ShowDialog();

}

private void запрос2ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Task2 sendTo = new Task2();

sendTo.ShowDialog();

}

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

{

this.Close();

}

Форма групп

Graphical user interface

Description automatically generated with low confidence

public Groups()

{

InitializeComponent();

DisplayData();

}

//Загрузка данных

private void DisplayData()

{

sqlConnection1.Open();

DataTable dt = new DataTable();

sqlDataAdapter1.Fill(dt);

dataGridView1.DataSource = dt;

sqlConnection1.Close();

}

//Очищаем поля ввода

private void ClearTextBoxes()

{

textBox1.Text = ""; textBox2.Text = ""; textBox3.Text = "";

}

//Обновить данные после закрытия

private void Groups\_Deactivate(object sender, EventArgs e)

{

sqlDataAdapter1.Update(this.lab2\_checkDataSet.Groups);

//this.groupsTableAdapter.Fill(this.lab2\_checkDataSet.Groups);

}

//Кнопка добавить

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

{

if (textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "")

{

try

{

sqlConnection1.Open();

sqlInsertCommand1.Parameters["@Code\_of\_group"].Value = textBox1.Text;

sqlInsertCommand1.Parameters["@Number\_of\_group"].Value = Convert.ToInt32(textBox2.Text);

sqlInsertCommand1.Parameters["@Date\_of\_foundation"].Value = Convert.ToInt32(textBox3.Text);

sqlInsertCommand1.ExecuteNonQuery();

ClearTextBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("Запись уже имеется!");

}

finally

{

sqlConnection1.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Недостаточно данных!");

}

}

//Кнопка найти

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

{

dataGridView1.ClearSelection();

if (textBox1.Text != "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

if (dataGridView1.Rows[i].Cells[1].Value != null)

{

if (dataGridView1.Rows[i].Cells[1].Value.ToString().Equals(textBox1.Text))

{

dataGridView1.CurrentCell = dataGridView1.Rows[i].Cells[1];

dataGridView1.Rows[i].Selected = true;

return;

}

}

}

MessageBox.Show("Данной записи не существует!");

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка изменить

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

{

if (textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "")

{

try

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlUpdateCommand1.Parameters["@Cipher\_of\_group"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlUpdateCommand1.Parameters["@Code\_of\_group"].Value = textBox1.Text;

sqlUpdateCommand1.Parameters["@Number\_of\_group"].Value = Convert.ToInt32(textBox2.Text);

sqlUpdateCommand1.Parameters["@Date\_of\_foundation"].Value = Convert.ToInt32(textBox3.Text);

sqlUpdateCommand1.ExecuteNonQuery();

ClearTextBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

finally

{

sqlConnection1.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка удалить

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

{

ClearTextBoxes();

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlDeleteCommand1.Parameters["@Cipher\_of\_group"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlDeleteCommand1.ExecuteNonQuery();

sqlConnection1.Close();

DisplayData();

}

Форма студентов

Graphical user interface

Description automatically generated

public Students()

{

InitializeComponent();

DisplayData();

}

//Загрузка данных

private void DisplayData()

{

sqlConnection1.Open();

DataTable dt = new DataTable();

sqlDataAdapter1.Fill(dt);

dataGridView1.DataSource = dt;

sqlConnection1.Close();

dataGridView1.Columns[0].HeaderText = "Код зачетки";

dataGridView1.Columns[1].HeaderText = "ФИО студента";

dataGridView1.Columns[1].Width = 200;

dataGridView1.Columns[2].HeaderText = "Код группы";

dataGridView1.Columns[3].HeaderText = "Пол";

dataGridView1.Columns[4].HeaderText = "Дата рождения";

}

//Очищаем поля ввода

private void ClearTextBoxes()

{

textBox1.Text = ""; textBox2.Text = ""; textBox3.Text = ""; comboBox1.SelectedIndex = -1; comboBox2.SelectedIndex=-1;

}

//Загрузка формы

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

{

sqlConnection2.Open();

DataTable dg = new DataTable();

sqlDataAdapter2.Fill(dg);

comboBox1.DataSource = dg;

sqlConnection2.Close();

comboBox2.Items.Add("M");

comboBox2.Items.Add("W");

ClearTextBoxes();

}

//Обновить данные после закрытия

private void Students\_Deactivate(object sender, EventArgs e)

{

sqlDataAdapter1.Update(this.lab2\_checkDataSet.Subjects);

//this.groupsTableAdapter.Fill(this.lab2\_checkDataSet.Subjects);

}

//Кнопка добавить

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

{

if (textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "" && comboBox2.Text!="")

{

try

{

sqlConnection2.Open();

sqlInsertCommand1.Parameters["@Code\_of\_record\_book"].Value = Convert.ToInt32(textBox1.Text);

sqlInsertCommand1.Parameters["@Groups\_Cipher\_of\_group"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlInsertCommand1.Parameters["@Full\_name"].Value = textBox2.Text;

sqlInsertCommand1.Parameters["@Gender"].Value = comboBox2.Text;

sqlInsertCommand1.Parameters["@Date\_of\_birth"].Value = DateTime.Parse(textBox3.Text);

sqlInsertCommand1.ExecuteNonQuery();

ClearTextBoxes();

} catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("Запись уже имеется!");

}

finally

{

sqlConnection2.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Недостаточно данных!");

}

}

//Кнопка найти

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

{

dataGridView1.ClearSelection();

if (textBox1.Text != "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

if (dataGridView1.Rows[i].Cells[0].Value != null)

{

if (dataGridView1.Rows[i].Cells[0].Value.ToString().Equals(textBox1.Text))

{

dataGridView1.CurrentCell = dataGridView1.Rows[i].Cells[0];

dataGridView1.Rows[i].Selected = true;

return;

}

}

}

MessageBox.Show("Данной записи не существует!");

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка изменить

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

{

if(textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "" && comboBox2.Text!="")

{

try

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection2.Open();

sqlUpdateCommand1.Parameters["@Code\_of\_record\_book"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlUpdateCommand1.Parameters["@Groups\_Cipher\_of\_group"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlUpdateCommand1.Parameters["@Gender"].Value = comboBox2.Text;

sqlUpdateCommand1.Parameters["@Full\_name"].Value = textBox2.Text;

sqlUpdateCommand1.Parameters["@Date\_of\_birth"].Value = DateTime.Parse(textBox3.Text);

sqlUpdateCommand1.ExecuteNonQuery();

ClearTextBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

finally

{

sqlConnection2.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка удалить

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

{

ClearTextBoxes();

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection2.Open();

sqlDeleteCommand1.Parameters["@Code\_of\_record\_book"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlDeleteCommand1.ExecuteNonQuery();

sqlConnection2.Close();

DisplayData();

}

Форма предметов

Graphical user interface

Description automatically generated

public Subjects()

{

InitializeComponent();

DisplayData();

}

//Загрузка данных

private void DisplayData()

{

sqlConnection1.Open();

DataTable dt = new DataTable();

sqlDataAdapter1.Fill(dt);

dataGridView1.DataSource = dt;

sqlConnection1.Close();

}

//Очищаем поля ввода

private void ClearTextBoxes()

{

textBox1.Text = ""; textBox2.Text = "";

}

//Обновить данные после закрытия

private void Subjects\_Deactivate(object sender, EventArgs e)

{

sqlDataAdapter1.Update(this.lab2\_checkDataSet.Subjects);

}

//Кнопка добавить

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

{

if (textBox1.Text != "" && textBox2.Text != "")

{

try

{

sqlConnection1.Open();

sqlInsertCommand1.Parameters["@Name\_of\_subject"].Value = textBox1.Text;

sqlInsertCommand1.Parameters["@Name\_of\_professor"].Value = textBox2.Text;

sqlInsertCommand1.ExecuteNonQuery();

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("Запись уже имеется!");

}

finally

{

sqlConnection1.Close();

ClearTextBoxes();

DisplayData();

}

}

else

{

MessageBox.Show("Недостаточно данных!");

}

}

//Кнопка найти

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

{

dataGridView1.ClearSelection();

if (textBox1.Text != "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

if (dataGridView1.Rows[i].Cells[1].Value != null)

{

if (dataGridView1.Rows[i].Cells[1].Value.ToString().Equals(textBox1.Text))

{

dataGridView1.CurrentCell = dataGridView1.Rows[i].Cells[1];

dataGridView1.Rows[i].Selected = true;

return;

}

}

}

MessageBox.Show("Данной записи не существует!");

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка изменить

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

{

if (textBox1.Text != "" && textBox2.Text != "")

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlUpdateCommand1.Parameters["@Cipher\_of\_subject"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlUpdateCommand1.Parameters["@Name\_of\_subject"].Value = textBox1.Text;

sqlUpdateCommand1.Parameters["@Name\_of\_professor"].Value = textBox2.Text;

sqlUpdateCommand1.ExecuteNonQuery();

sqlConnection1.Close();

ClearTextBoxes();

DisplayData();

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка удалить

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

{

ClearTextBoxes();

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlDeleteCommand1.Parameters["@Cipher\_of\_subject"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlDeleteCommand1.ExecuteNonQuery();

sqlConnection1.Close();

DisplayData();

}

Форма предметов групп

Graphical user interface

Description automatically generated

public Groups\_has\_Subj()

{

InitializeComponent();

DisplayData();

}

//Загрузка данных

private void DisplayData()

{

sqlConnection2.Open();

DataTable dt = new DataTable();

sqlDataAdapter1.Fill(dt);

dataGridView1.DataSource = dt;

sqlConnection2.Close();

dataGridView1.Columns[0].Visible = false;

dataGridView1.Columns[1].HeaderText = "Код группы";

dataGridView1.Columns[2].HeaderText = "Название предмета";

dataGridView1.Columns[2].Width = 200;

dataGridView1.Columns[3].HeaderText = "Количество часов";

dataGridView1.Columns[3].Width = 150;

dataGridView1.Columns[4].HeaderText = "Тип аттестации";

dataGridView1.Columns[4].Width = 150;

}

//Очищаем поля ввода

private void ClearTextBoxes()

{

textBox1.Text = ""; comboBox1.SelectedIndex = -1; comboBox2.SelectedIndex = -1; comboBox3.SelectedIndex = -1;

}

//Загрузка формы

private void Groups\_has\_Subj\_Load(object sender, EventArgs e)

{

sqlConnection1.Open();

DataTable dg = new DataTable(), dg1 = new DataTable();

sqlDataAdapter2.Fill(dg);

comboBox1.DataSource = dg;

sqlDataAdapter3.Fill(dg1);

comboBox2.DataSource = dg1;

comboBox3.Items.Add("credit");

comboBox3.Items.Add("exam");

sqlConnection1.Close();

ClearTextBoxes();

}

//Кнопка добавить

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

{

if (textBox1.Text != "" && comboBox3.Text!="")

{

try

{

sqlConnection2.Open();

sqlInsertCommand1.Parameters["@Groups\_Cipher\_of\_group"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlInsertCommand1.Parameters["@Subjects\_Cipher\_of\_subject"].Value = Convert.ToInt32(comboBox2.SelectedValue);

sqlInsertCommand1.Parameters["@Number\_of\_study\_hours"].Value = Convert.ToInt32(textBox1.Text);

sqlInsertCommand1.Parameters["@Type\_of\_certification"].Value = comboBox3.Text;

sqlInsertCommand1.ExecuteNonQuery();

ClearTextBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("Запись уже имеется!");

}

finally

{

sqlConnection2.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Недостаточно данных!");

}

}

//Кнопка найти

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

{

dataGridView1.ClearSelection();

if (comboBox1.Text != "" && comboBox2.Text!= "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

if (dataGridView1.Rows[i].Cells[1].Value != null)

{

//Console.WriteLine(dataGridView1.Rows[i].Cells[2].Value.ToString().Equals(comboBox2.SelectedValue.ToString()));

//Console.WriteLine(" Table: "+ dataGridView1.Rows[i].Cells[1].Value.ToString()+" "+ " Table: " + dataGridView1.Rows[i].Cells[2].Value.ToString());

if (dataGridView1.Rows[i].Cells[1].Value.Equals(comboBox1.Text) && dataGridView1.Rows[i].Cells[2].Value.Equals(comboBox2.Text))

{

dataGridView1.CurrentCell = dataGridView1.Rows[i].Cells[1];

dataGridView1.Rows[i].Selected = true;

return;

}

}

}

MessageBox.Show("Данной записи не существует!");

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка изменить

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

{

if (textBox1.Text != "" && comboBox3.Text != "")

{

try

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection2.Open();

sqlUpdateCommand1.Parameters["@Self\_table\_id"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlUpdateCommand1.Parameters["@Groups\_Cipher\_of\_group"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlUpdateCommand1.Parameters["@Subjects\_Cipher\_of\_subject"].Value = Convert.ToInt32(comboBox2.SelectedValue);

sqlUpdateCommand1.Parameters["@Number\_of\_study\_hours"].Value = Convert.ToInt32(textBox1.Text);

sqlUpdateCommand1.Parameters["@Type\_of\_certification"].Value = comboBox3.Text;

sqlUpdateCommand1.ExecuteNonQuery();

ClearTextBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

finally

{

sqlConnection2.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка удалить

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

{

ClearTextBoxes();

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection2.Open();

sqlDeleteCommand1.Parameters["@Self\_table\_id"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlDeleteCommand1.ExecuteNonQuery();

sqlConnection2.Close();

DisplayData();

}

Форма предметов студентов

В данной программе во избежание выбрасывания исключения пользоветлю реализовано динамическое формирование второго ComboBox, то есть сперва пользователь выбирает Код зачетки, после чего приложение само подгрузит возможные предметы на основе имеющихся у данной группы (таблица Groups\_has\_Subject), а затем возможные варианты аттестации (либо оценки, если Type\_of\_certification == exam, либо зачет/незачет, если Type\_of\_certification == credit).

Graphical user interface, application

Description automatically generated

SortedDictionary<int,string> MapOfSubjects;

int GroupValue = 0;

int SubjectKey = 0;

public Student\_has\_Subj()

{

InitializeComponent();

DisplayData();

}

//Загрузка данных

private void DisplayData()

{

sqlConnection1.Open();

DataTable dt = new DataTable();

sqlDataAdapter1.Fill(dt);

dataGridView1.DataSource = dt;

sqlConnection1.Close();

dataGridView1.Columns[0].Visible = false;

dataGridView1.Columns[1].HeaderText = "Код зачётки";

dataGridView1.Columns[1].Width = 150;

dataGridView1.Columns[2].HeaderText = "Предмет";

dataGridView1.Columns[2].Width = 200;

dataGridView1.Columns[3].HeaderText = "Результат аттестации";

}

//Загрузка формы

private void Student\_has\_Subj\_Load(object sender, EventArgs e)

{

sqlConnection1.Open();

DataTable dg = new DataTable();

sqlDataAdapter2.Fill(dg);

comboBox1.DataSource = dg;

sqlConnection1.Close();

ClearComboBoxes();

}

//Очищаем поля ввода

private void ClearComboBoxes()

{

comboBox2.SelectedIndex=-1;comboBox3.SelectedIndex=-1; comboBox1.SelectedIndex = -1;

}

//Формируем варианты предметов на основе comboBox1.SelectedValue

private void comboBox1\_SelectedValueChanged(object sender, EventArgs e)

{

sqlConnection2.Open();

DataTable dtGHS = new DataTable(), dtSt = new DataTable(), dtSub = new DataTable();

sqlDataAdapter4.Fill(dtGHS);

sqlDataAdapter3.Fill(dtSub);

sqlDataAdapter2.Fill(dtSt);

sqlConnection2.Close();

MapOfSubjects = new SortedDictionary<int,string>();

comboBox2.Items.Clear();

comboBox3.Items.Clear();

for (int i = 0; i < dtSt.Rows.Count; i++)

{

if (Convert.ToInt32(dtSt.Rows[i][0]) == Convert.ToInt32(comboBox1.SelectedValue))

{

GroupValue = Convert.ToInt32(dtSt.Rows[i][1]);

break;

}

}

for (int i = 0; i < dtGHS.Rows.Count; i++)

{

if (Convert.ToInt32(dtGHS.Rows[i][1]) == GroupValue)

{

MapOfSubjects.Add(Convert.ToInt32(dtGHS.Rows[i][2]),"");

}

}

for (int i = 0; i < dtSub.Rows.Count; i++)

{

if (MapOfSubjects.ContainsKey(Convert.ToInt32(dtSub.Rows[i][0])))

{

MapOfSubjects[Convert.ToInt32(dtSub.Rows[i][0])] = dtSub.Rows[i][1].ToString();

comboBox2.Items.Add(dtSub.Rows[i][1].ToString());

}

}

}

//Формируем варианты результатов аттестации на основе comboBox2.SelectedValue

private void comboBox2\_SelectedValueChanged(object sender, EventArgs e)

{

sqlConnection2.Open();

DataTable dtGHS = new DataTable(), dtSt = new DataTable();

sqlDataAdapter4.Fill(dtGHS);

sqlDataAdapter2.Fill(dtSt);

sqlConnection2.Close();

comboBox3.Items.Clear();

SubjectKey = MapOfSubjects.FirstOrDefault(x => x.Value == comboBox2.Text).Key;

//Console.WriteLine(myKey);

for (int i = 0; i < dtGHS.Rows.Count; i++)

{

//Console.WriteLine(Convert.ToInt32(dtGHS.Rows[i][1]) + " " + GroupValue);

//Console.WriteLine(Convert.ToInt32(dtGHS.Rows[i][2]) + " " + SubjectKey);

if (Convert.ToInt32(dtGHS.Rows[i][1]) == GroupValue && Convert.ToInt32(dtGHS.Rows[i][2]) == SubjectKey)

{

if (dtGHS.Rows[i][4].ToString().Trim().Equals("credit"))

{

comboBox3.Items.Add("credit");

comboBox3.Items.Add("not credit");

}

else

{

comboBox3.Items.Add("5");

comboBox3.Items.Add("4");

comboBox3.Items.Add("3");

comboBox3.Items.Add("2");

comboBox3.Items.Add("1");

}

break;

}

}

}

//Кнопка добавить

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

{

if (comboBox2.Text != "" && comboBox3.Text != "")

{

try

{

sqlConnection1.Open();

sqlInsertCommand1.Parameters["@Students\_Code\_of\_record\_book"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlInsertCommand1.Parameters["@Subjects\_Cipher\_of\_subject"].Value = SubjectKey;

sqlInsertCommand1.Parameters["@Result\_of\_certification"].Value = comboBox3.Text;

sqlInsertCommand1.ExecuteNonQuery();

ClearComboBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("Запись уже имеется!");

}

finally

{

sqlConnection1.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Недостаточно данных!");

}

}

//Кнопка найти

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

{

dataGridView1.ClearSelection();

if (comboBox1.Text != "" && comboBox2.Text != "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

if (dataGridView1.Rows[i].Cells[1].Value != null)

{

//Console.WriteLine(dataGridView1.Rows[i].Cells[2].Value.ToString().Equals(comboBox2.SelectedValue.ToString()));

//Console.WriteLine(" Table: "+ dataGridView1.Rows[i].Cells[1].Value.ToString()+" "+ " Table: " + dataGridView1.Rows[i].Cells[2].Value.ToString());

if (Convert.ToInt32(dataGridView1.Rows[i].Cells[1].Value) == Convert.ToInt32(comboBox1.Text) && dataGridView1.Rows[i].Cells[2].Value.Equals(comboBox2.Text))

{

dataGridView1.CurrentCell = dataGridView1.Rows[i].Cells[1];

dataGridView1.Rows[i].Selected = true;

return;

}

}

}

MessageBox.Show("Данной записи не существует!");

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка изменить

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

{

if(comboBox1.Text != "" && comboBox2.Text != "" && comboBox3.Text!="")

{

try

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlUpdateCommand1.Parameters["@Self\_table\_id"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlUpdateCommand1.Parameters["@Code\_of\_record\_book"].Value = Convert.ToInt32(comboBox1.SelectedValue);

sqlUpdateCommand1.Parameters["@Subjects\_Cipher\_of\_subject"].Value = SubjectKey;

sqlUpdateCommand1.Parameters["@Result\_of\_certification"].Value = comboBox3.Text;

sqlUpdateCommand1.ExecuteNonQuery();

ClearComboBoxes();

}

catch (FormatException)

{

MessageBox.Show("Ошибка приведения типов!");

}

finally

{

sqlConnection1.Close();

DisplayData();

}

}

else

{

MessageBox.Show("Поля не заполнены!");

}

}

//Кнопка удалить

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

{

int rowIndex = dataGridView1.CurrentCell.RowIndex;

sqlConnection1.Open();

sqlDeleteCommand1.Parameters["@Self\_table\_id"].Value = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells[0].Value.ToString());

sqlDeleteCommand1.ExecuteNonQuery();

ClearComboBoxes();

sqlConnection1.Close();

DisplayData();

}

Запрос №1

Graphical user interface, text

Description automatically generated

public Task1()

{

InitializeComponent();

}

private void Task1\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "lab2\_checkDataSet.Groups". При необходимости она может быть перемещена или удалена.

this.groupsTableAdapter.Fill(this.lab2\_checkDataSet.Groups);

// TODO: данная строка кода позволяет загрузить данные в таблицу "lab2\_checkDataSet.Student\_has\_Subjects". При необходимости она может быть перемещена или удалена.

this.student\_has\_SubjectsTableAdapter.Fill(this.lab2\_checkDataSet.Student\_has\_Subjects);

}

//Получаем данные

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

{

sqlConnection2.Open();

System.Data.DataTable dt = new System.Data.DataTable();

int param = Convert.ToInt32(comboBox1.SelectedValue);

SqlCommand cmd = new SqlCommand();

cmd = new SqlCommand("dbo.Procedure\_1", sqlConnection2);

cmd.Parameters.Add(new SqlParameter("@Cipher\_of\_group", param));

cmd.CommandType = CommandType.StoredProcedure;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter();

sqlDataAdapter.SelectCommand = cmd;

sqlDataAdapter.Fill(dt);

dataGridView1.DataSource = dt;

dataGridView1.Columns[0].HeaderText = "Код группы";

dataGridView1.Columns[1].HeaderText = "Предмет";

dataGridView1.Columns[2].HeaderText = "ФИО студента";

dataGridView1.Columns[3].HeaderText = "Результат";

dataGridView1.Columns[4].HeaderText = "Средний результат";

sqlConnection2.Close();

}

//Экспорт данных

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

{

Excel.Application excelApp = new Excel.Application();

excelApp.Application.Workbooks.Add(Type.Missing);

for (int i = 1; i < dataGridView1.Columns.Count + 1; i++)

{

excelApp.Cells[1, i] = dataGridView1.Columns[i - 1].HeaderText;

}

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 0; j < dataGridView1.Columns.Count; j++)

{

if (j > dataGridView1.Columns.Count-2)

excelApp.Cells[i + 2, j + 1] = Convert.ToDouble(dataGridView1.Rows[i].Cells[j].Value);

else

excelApp.Cells[i + 2, j + 1] = dataGridView1.Rows[i].Cells[j].Value.ToString();

}

}

excelApp.Columns.AutoFit();

excelApp.Visible = true;

}

Запрос №2

Graphical user interface, text, application

Description automatically generated

public Task2()

{

InitializeComponent();

}

//Получаем данные

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

{

sqlConnection1.Open();

DataTable dt = new DataTable();

SqlCommand cmd = new SqlCommand();

cmd = new SqlCommand("dbo.Procedure\_2", sqlConnection1);

cmd.CommandType = CommandType.StoredProcedure;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter();

sqlDataAdapter.SelectCommand = cmd;

sqlDataAdapter.Fill(dt);

dataGridView1.DataSource = dt;

dataGridView1.Columns[0].HeaderText = "Код группы";

dataGridView1.Columns[1].HeaderText = "Предмет";

dataGridView1.Columns[2].HeaderText = "ФИО студента";

dataGridView1.Columns[3].HeaderText = "Результат";

dataGridView1.Columns[4].HeaderText = "Количество неатестованных";

sqlConnection1.Close();

}

//Экспорт данных

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

{

Excel.Application excelApp = new Excel.Application();

excelApp.Application.Workbooks.Add(Type.Missing);

for (int i = 1; i < dataGridView1.Columns.Count + 1; i++)

{

excelApp.Cells[1, i] = dataGridView1.Columns[i - 1].HeaderText;

}

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 0; j < dataGridView1.Columns.Count; j++)

{

if (j > dataGridView1.Columns.Count - 2)

excelApp.Cells[i + 2, j + 1] = Convert.ToInt32(dataGridView1.Rows[i].Cells[j].Value);

else

excelApp.Cells[i + 2, j + 1] = dataGridView1.Rows[i].Cells[j].Value.ToString();

}

}

excelApp.Columns.AutoFit();

excelApp.Visible = true;

}

Скриншоты результатов тестирования

Форма групп

**Кнопка добавить:**

|  |  |
| --- | --- |
| Graphical user interface  Description automatically generated | |
| Успешное добавление: | |
| Table  Description automatically generated | |
| Если запись уже существует: | |
|  | |
| Если поля не заполнены: | Если тип не совпадает: |
|  |  |

**Кнопка найти:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если ключ отсутствует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка изменить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если тип не совпадает: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка удалить:**

|  |
| --- |
| Graphical user interface  Description automatically generated |
| Успешное выполнение: |
|  |

Форма студентов

**Кнопка добавить:**

|  |  |
| --- | --- |
|  | |
| Успешное добавление: | |
|  | |
| Если запись уже существует: | |
|  | |
| Если поля не заполнены: | Если тип не совпадает: |
|  |  |

**Кнопка найти:**

|  |
| --- |
|  |
| Успешное выполнение (поиск происходит только по первичному ключу): |
|  |
| Если ключ отсутствует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка изменить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если тип не совпадает: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка удалить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |

Форма предметов

**Кнопка добавить:**

|  |
| --- |
|  |
| Успешное добавление: |
|  |
| Если запись уже существует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка найти:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если ключ отсутствует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка изменить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка удалить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |

Форма предметов групп

**Кнопка добавить:**

|  |  |
| --- | --- |
|  | |
| Успешное добавление: | |
|  | |
| Если запись уже существует: | |
|  | |
| Если поля не заполнены: | Если тип не совпадает: |
|  |  |

**Кнопка найти:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если ключ отсутствует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка изменить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если тип не совпадает: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка удалить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |

Форма предметов групп

**Кнопка добавить:**

|  |
| --- |
|  |
| Успешное добавление: |
|  |
| Если запись уже существует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка найти:**

|  |
| --- |
| Graphical user interface  Description automatically generated |
| Успешное выполнение |
| Graphical user interface, table  Description automatically generated |
| Если ключ отсутствует: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка изменить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |
| Если поля не заполнены: |
|  |

**Кнопка удалить:**

|  |
| --- |
|  |
| Успешное выполнение: |
|  |

Запрос №1

|  |
| --- |
| Входные данные: |
| Graphical user interface  Description automatically generated |
| Выполненный запрос: |
|  |
| Экспорт в excel: |
|  |

Запрос №2

|  |
| --- |
| Входные данные: |
|  |
| Выполненный запрос: |
|  |
| Экспорт в excel: |
|  |