МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра вычислительной техники



**РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА**

**ПО ДИСЦИПЛИНЕ «ИНФОРМАЦИОННЫЕ СИСТЕМЫ»**

Факультет: АВТ Преподаватель: Бычков М.И.

Группа: АВТ-819

Студент:

Ванин К.Е.

Вариант: 8

Новосибирск, 2021 г.

Оглавление

[Цель работы 3](#_Toc41705251)

[Задание 3](#_Toc41705252)

[Ход работы 3](#_Toc41705253)

[Выводы 20](#_Toc41705254)

[Список источников 21](#_Toc41705255)

[Листинг 22](#_Toc41705256)

# Цель работы

Разработать информационную систему согласно варианту задания.

# Задание

Разработать информационную систему, которая автоматизирует деятельность центра обучения фирмы. Сотрудники фирмы должны раз в год прослушать набор курсов, предназначенных для их отдела.

Система должна:

1. содержать сведения о сотрудниках фирмы (имя, фамилия, номер должности, номер отдела, дата приема на работу);

2. сведения о курсах, которые преподаются в центре: (название, номер, длительность, предназначение (для какого отдела), краткое содержание);

3. сведения о прохождении обучения (номер сотрудника, номер курса, дата прохождения);

4. выдавать справки о сотрудниках, проходящих обучение в данном месяце, годе;

5. выдавать справки о сотрудниках, которые должны проходить обучение (срок больше года);

6. выдавать справки о сотрудниках отделов с курсами, которые они прослушали;

7. иметь средства ввода и обновления всех данных.

# Ход работы

Создадим базу данных с именем «training\_center».

CREATE TABLE training\_center;

Создадим таблицы courses (курсы), departments (отделы – таблица-справочник), employees (сотрудники), positions (должности – таблица-справочник), training\_info (сведения о прохождении обучения).

Структура таблиц следующая:

* courses (id, название, длительность в часах, id отдела, описание курса)
* departments (id, название)
* employees (id, имя, фамилия, id должности, id отдела, дата приема на работу)
* positions (id, название)
* training\_info (id, id сотрудника, id курса, дата прохождения)

CREATE TABLE courses (

id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL UNIQUE,

duration\_hours INT NOT NULL,

department\_id BIGINT UNSIGNED NOT NULL,

description VARCHAR(255) DEFAULT NULL,

FOREIGN KEY (department\_id) REFERENCES departments (id) ON DELETE CASCADE

);

CREATE TABLE departments (

id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE employees (

id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL,

surname VARCHAR(50) NOT NULL,

position\_id BIGINT UNSIGNED NOT NULL,

department\_id BIGINT UNSIGNED NOT NULL,

employment\_date DATE NOT NULL,

FOREIGN KEY (position\_id) REFERENCES positions (id) ON DELETE CASCADE,

FOREIGN KEY (department\_id) REFERENCES departments (id) ON DELETE CASCADE

);

CREATE TABLE positions (

id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL UNIQUE,

);

CREATE TABLE training\_info (

id SERIAL PRIMARY KEY,

employee\_id BIGINT UNSIGNED NOT NULL,

course\_id BIGINT UNSIGNED NOT NULL,

date DATE NOT NULL,

FOREIGN KEY (employee\_id) REFERENCES employees (id) ON DELETE CASCADE,

FOREIGN KEY (course\_id) REFERENCES courses (id) ON DELETE CASCADE

);

После создания таблицы были заполнены произвольными данными. Содержимое таблиц представлено на рис. 1-5.

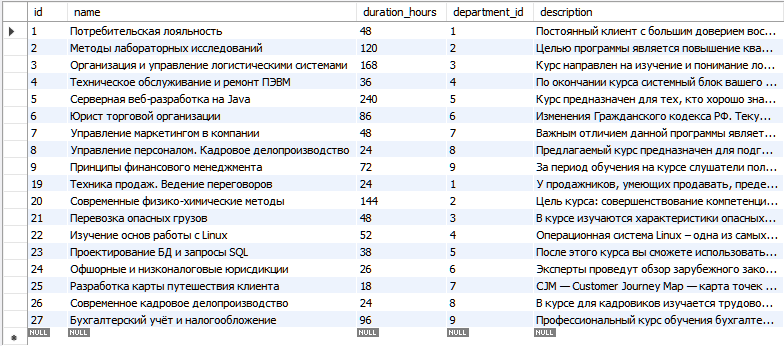


Рис. 1 – Содержимое таблицы courses.

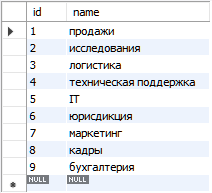


Рис. 2 – Содержимое таблицы departments.

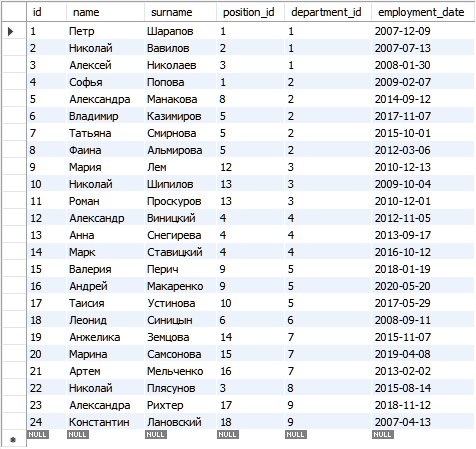


Рис. 3 – Содержимое таблицы employees.

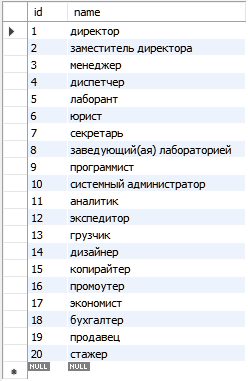


Рис. 4 – Содержимое таблицы positions.

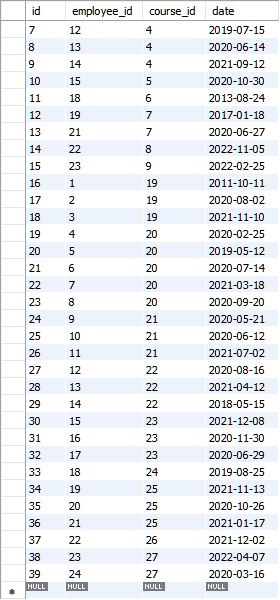


Рис. 5 – Содержимое таблицы training\_info.

Создадим процедуры, которые будут делать выборку соединением таблиц оператором INNER JOIN согласно заданию:

* all\_certificates (делает выборку всех данных в БД)
* certificates\_after\_year (делает выборку курсов, которые будут пройдены через год и позже)
* certificates\_by\_department (делает выборку курсов, которые были пройдены сотрудниками из заданного пользователем отдела)
* certificates\_coming\_month (делает выборку курсов, которые будут пройдены сотрудниками в течение месяца)
* certificates\_coming\_year (делает выборку курсов, которые будут пройдены сотрудниками в течение года)

DELIMITER $$

CREATE PROCEDURE all\_certificates()

BEGIN

SELECT employees.id, employees.name, employees.surname, positions.name AS position, departments.name AS department, employment\_date,

courses.name AS course, duration\_hours, training\_info.date AS course\_date, description AS course\_description

FROM training\_info

INNER JOIN employees

ON training\_info.employee\_id = employees.id

INNER JOIN courses

ON training\_info.course\_id = courses.id

INNER JOIN positions

ON position\_id = positions.id

INNER JOIN departments

ON employees.department\_id = departments.id

ORDER BY id;

END $$

CREATE PROCEDURE certificates\_after\_year()

BEGIN

SELECT employees.id, employees.name, employees.surname, positions.name AS position, departments.name AS department, employment\_date,

courses.name AS course, duration\_hours, training\_info.date AS course\_date, description AS course\_description

FROM training\_info

INNER JOIN employees

ON training\_info.employee\_id = employees.id

INNER JOIN courses

ON training\_info.course\_id = courses.id

INNER JOIN positions

ON position\_id = positions.id

INNER JOIN departments

ON employees.department\_id = departments.id

WHERE training\_info.date >= now() + INTERVAL 1 YEAR

ORDER BY id;

END $$

CREATE PROCEDURE certificates\_by\_department(dept VARCHAR(50))

BEGIN

SELECT employees.id, employees.name, employees.surname, positions.name AS position, departments.name AS department, employment\_date,

courses.name AS course, duration\_hours, training\_info.date AS course\_date, description AS course\_description

FROM training\_info

INNER JOIN employees

ON training\_info.employee\_id = employees.id

INNER JOIN courses

ON training\_info.course\_id = courses.id

INNER JOIN positions

ON position\_id = positions.id

INNER JOIN departments

ON employees.department\_id = departments.id

WHERE departments.name = dept AND training\_info.date < now()

ORDER BY id;

END $$

CREATE PROCEDURE certificates\_coming\_month()

BEGIN

SELECT employees.id, employees.name, employees.surname, positions.name AS position, departments.name AS department, employment\_date,

courses.name AS course, duration\_hours, training\_info.date AS course\_date, description AS course\_description

FROM training\_info

INNER JOIN employees

ON training\_info.employee\_id = employees.id

INNER JOIN courses

ON training\_info.course\_id = courses.id

INNER JOIN positions

ON position\_id = positions.id

INNER JOIN departments

ON employees.department\_id = departments.id

WHERE training\_info.date >= now() AND training\_info.date < now() + INTERVAL 1 MONTH

ORDER BY id;

END $$

CREATE PROCEDURE certificates\_coming\_year()

BEGIN

SELECT employees.id, employees.name, employees.surname, positions.name AS position, departments.name AS department, employment\_date,

courses.name AS course, duration\_hours, training\_info.date AS course\_date, description AS course\_description

FROM training\_info

INNER JOIN employees

ON training\_info.employee\_id = employees.id

INNER JOIN courses

ON training\_info.course\_id = courses.id

INNER JOIN positions

ON position\_id = positions.id

INNER JOIN departments

ON employees.department\_id = departments.id

WHERE training\_info.date >= now() AND training\_info.date < now() + INTERVAL 1 YEAR

ORDER BY id;

END $$

Создадим приложение Windows Forms.

Подключим драйвер MySQL.

Создадим класс DatabaseInfo. В этом классе будет инициализироваться информация, необходимая для открытия соединения с базой данных «training\_center»: хост, порт, имя БД, имя пользователя, пароль. В классе описан метод GetDBInfo(), возвращающий строку, необходимую для открытия соединения.

Создадим класс DatabaseConnection. В этом классе будут описаны методы открытия соединения с БД (Connect()), закрытия соединения с БД (Close()), выборки таблиц (Select(string sql)) и выполнения запросов с операциями манипулирования данными (Query(string sql)).

Установим пакет NuGet «DocX», который представляет собой библиотеку для создания и редактирования .docx-файлов.

Создадим класс Certificate. В этом классе будет описан конструктор с параметрами «имя, фамилия, должность, отдел, курс, длительность курса, дата прохождения курса». Внутри конструктора будет происходить создание документа со справкой для сотрудника. Файл будет сохранен в той же директории, что и исполняемый файл приложения под именем ИмяСотрудника\_ФамилияСотрудника\_Отдел\_ДатаПрохожденияКурса.docx.

Создадим новую форму (рис. 6). Поместим на нее объекты DataGridView, контейнер Panel, два ComboBox, RichTextBox, три кнопки.

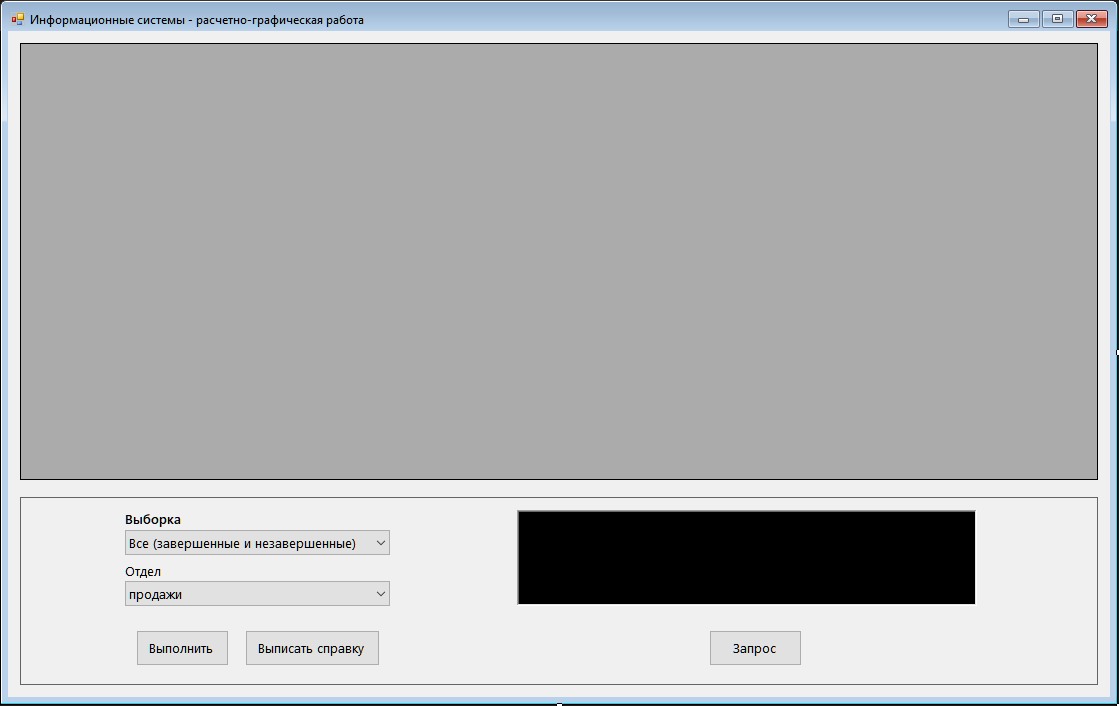


Рис. 6 – Созданная форма.

В верхнем ComboBox доступны следующие опции:

1. Все (завершенные и незавершенные)
2. В ближайший месяц
3. В ближайший год
4. Спустя год и позже
5. Завершенные (по отделу)

В нижнем ComboBox опции соответствуют выборке из таблицы departments (см. рис. 2).

После запуска приложения в верхнем ComboBox будет по умолчанию выбрано значение 1, а в нижнем – «продажи». При этом нижний ComboBox будет недоступен для взаимодействия и помечен темно-серым цветом (рис. 7). При выборе в верхнем ComboBox значения 5 становится доступным для взаимодействия нижний ComboBox.

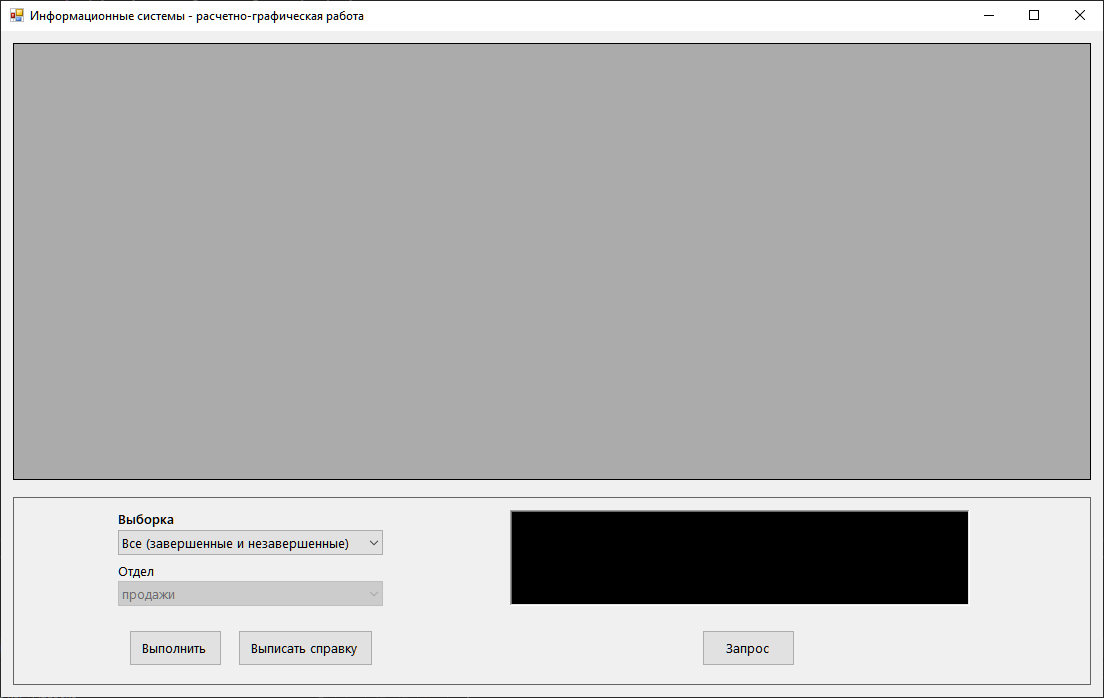


Рис. 7 – В верхнем ComboBox не выбрана опция «Завершенные (по отделу)».

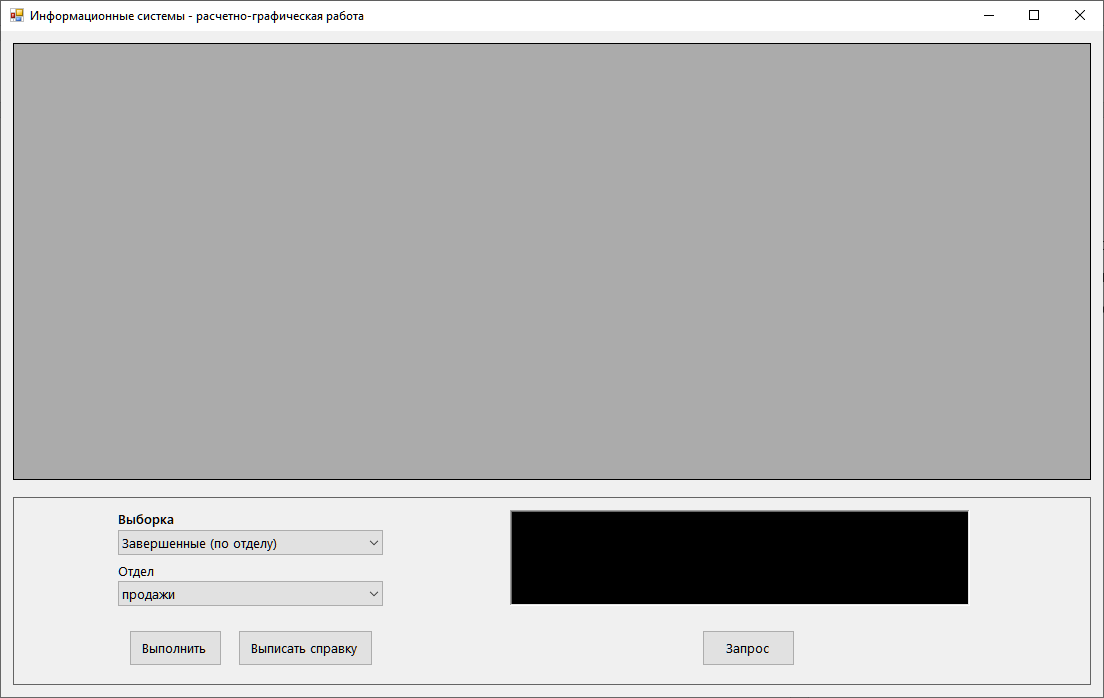


Рис. 8 – В верхнем ComboBox выбрана опция «Завершенные (по отделу)».

При нажатии на кнопку «Выполнить» будет вызван метод Select(string sql), в который будет передана строка, в которой будет вызываться процедура в зависимости от выбора пользователя в верхнем ComboBox:

* 1 – «CALL all\_certificates»
* 2 – «CALL certificates\_coming\_month»
* 3 – «CALL certificates\_coming\_year»
* 4 – «CALL certificates\_after\_year»
* 5 – «CALL certificates\_by\_department('аргумент')», где аргумент – это текстовое значение опции, которую выбрал пользователь в нижнем ComboBox.

Результатом выполнения метода станет вывод таблицы в DataGridView (рис. 9).

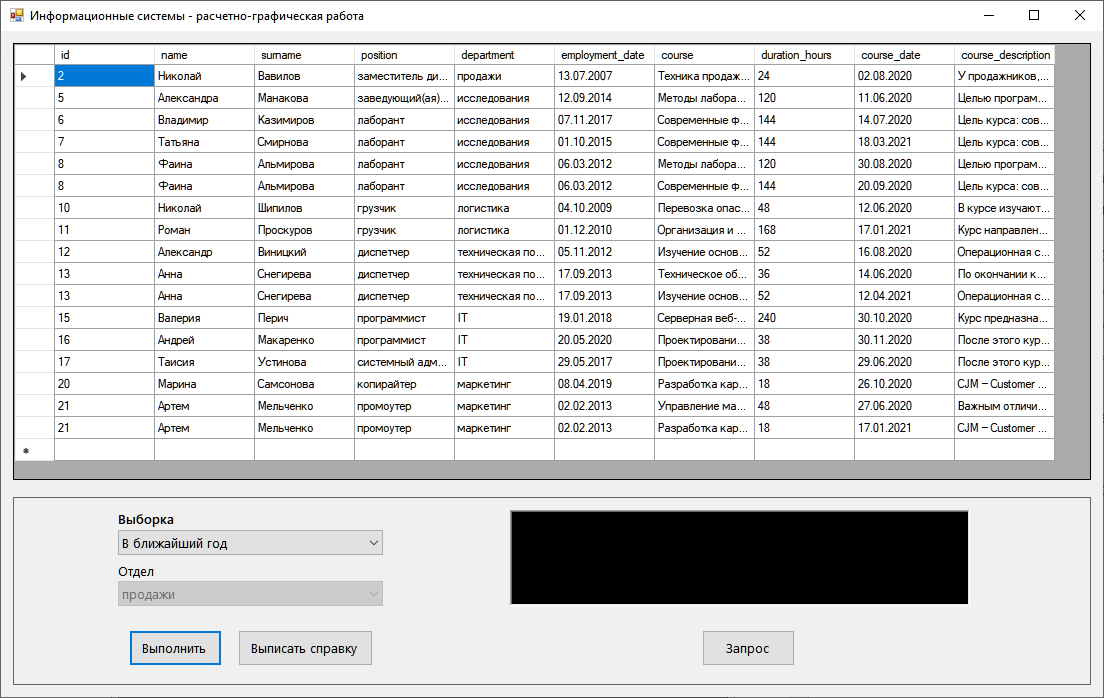


Рис. 9 – Пример результирующей выборки.

После того, как была произведена выборка, можно выписать справку в формате .docx. Для этого нужно с помощью DataGridView выбрать строку с необходимым сотрудником, нажав на кнопку, отмеченную на рис. 10 красным цветом.

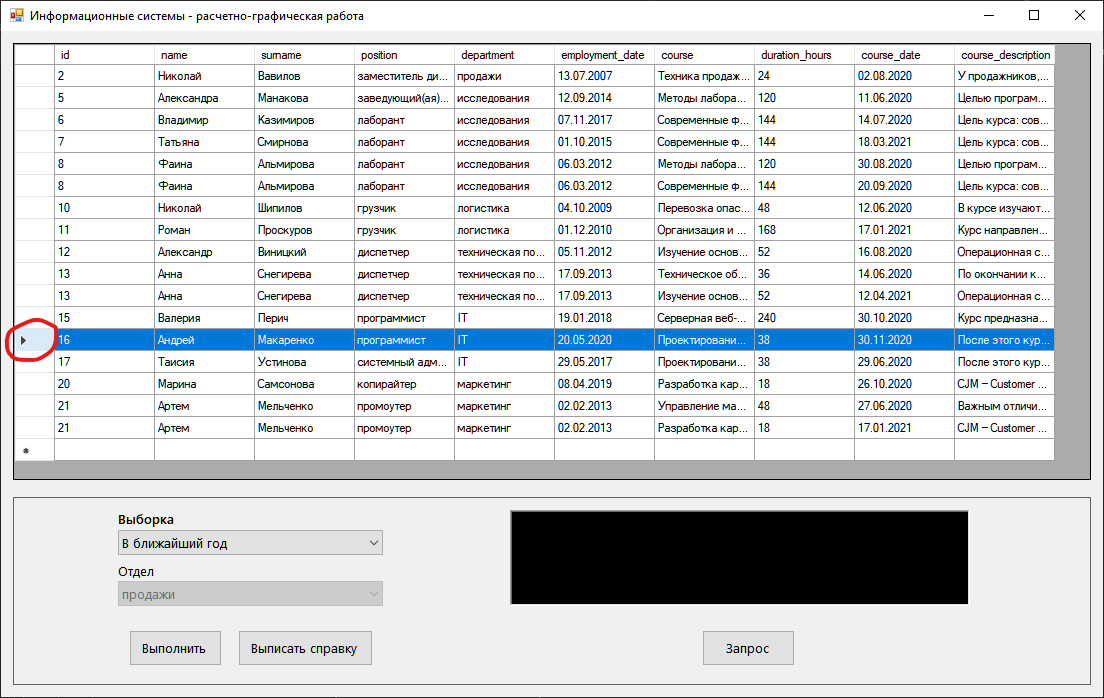


Рис. 10 – Выбор строки с данными и сотруднике.

После выбора строки для создания справки следует нажать на кнопку «Выписать справку». Будет создан экземпляр класса Certificate, в который будут переданы соответствующие данные о сотруднике из выбранной строки.

После этого, если процесс «WINWORD.EXE» доступен, через Microsoft Word будет автоматически открыт созданный документ со справкой (рис. 11).

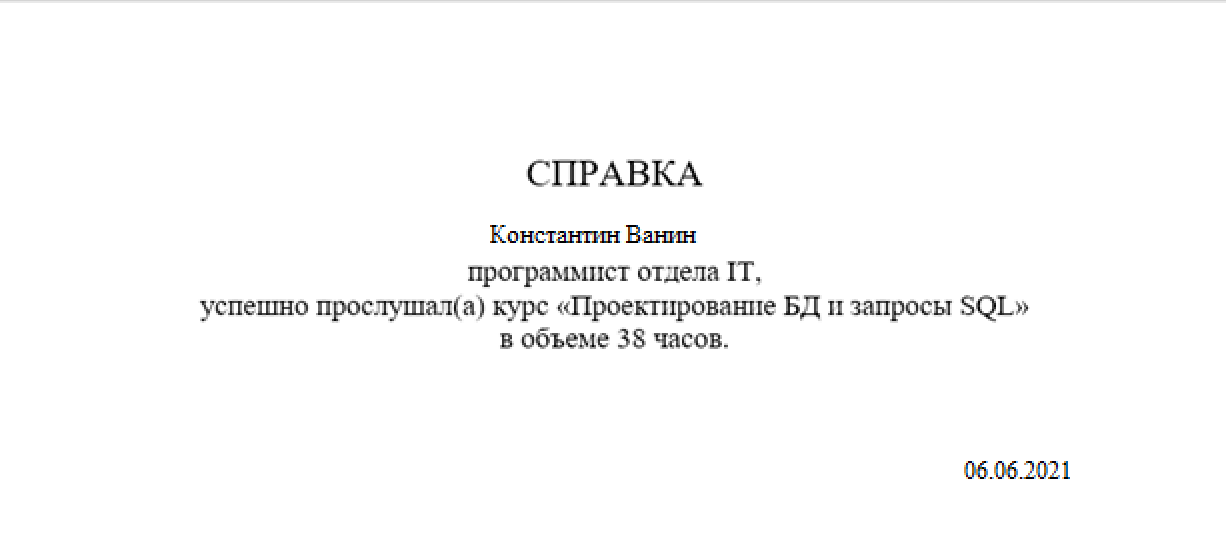


Рис. 11 – Созданная справка.

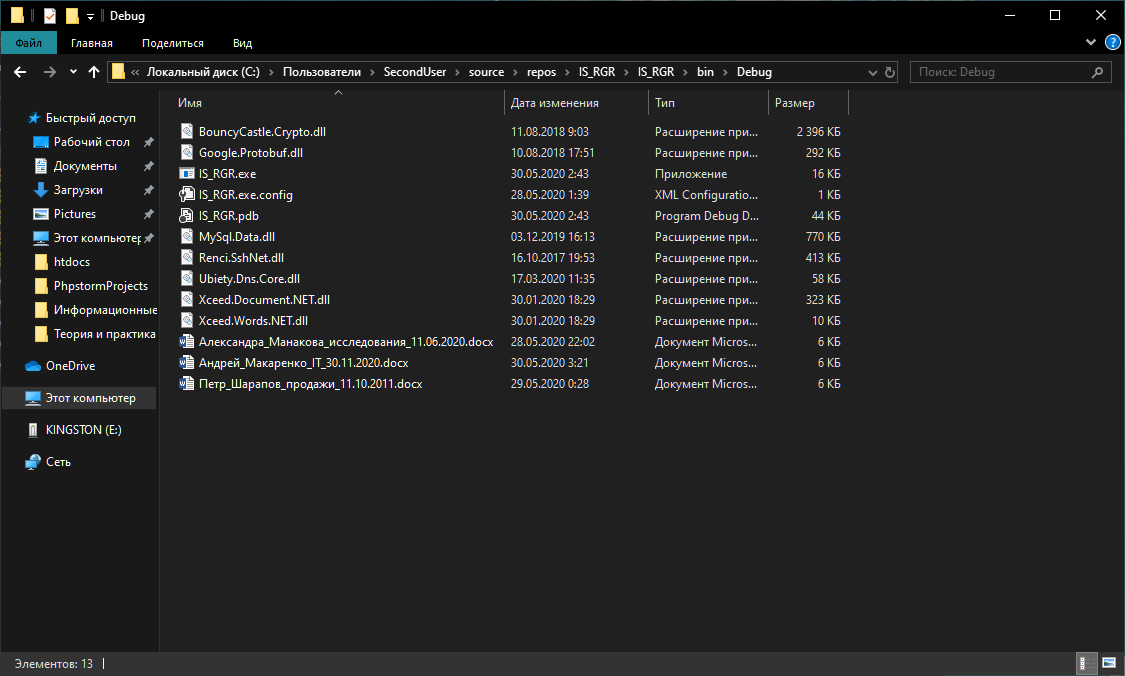


Рис. 12 – Имена и директория сохраненных файлов.

Для выполнения операций манипулирования данными необходимо знать структуру таблиц, так как приложение производит лишь те выборки, которые описаны в процедурах, ориентированных на рядового пользователя.

Если же пользователь имеет эту информацию, он может ввести SQL-запрос в RichTextBox, стилизованный под окно консоли, и нажать кнопку «Запрос». В RichTextBox будет выведен тип выполненного запроса и количество строк, измененных с его помощью (рис. 13-15).

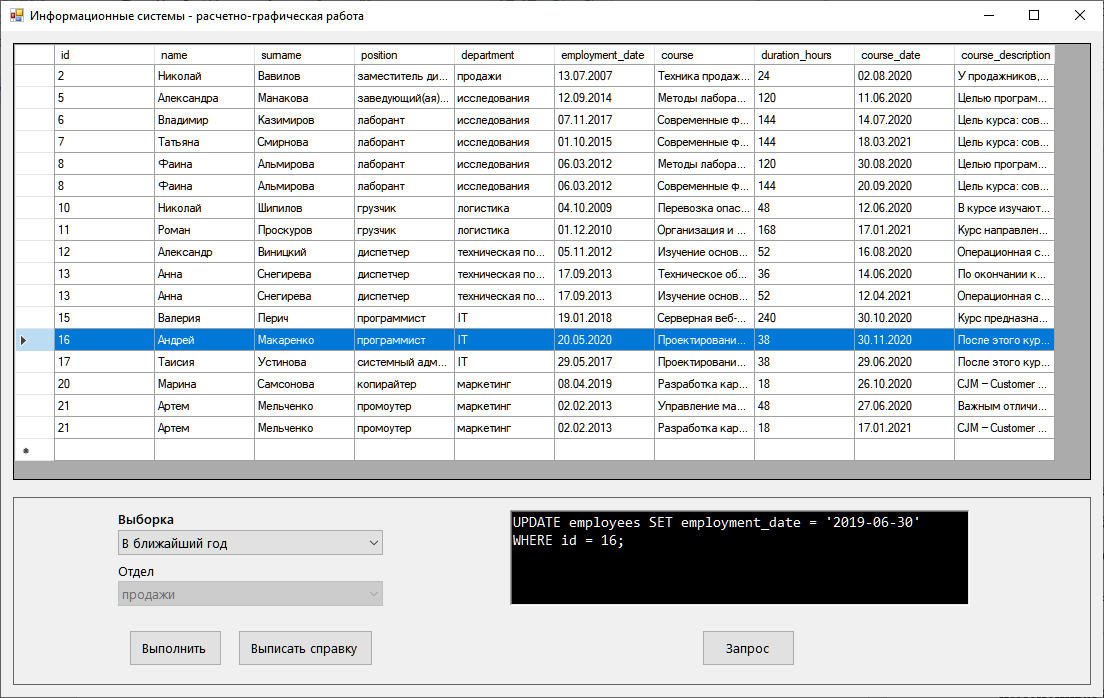


Рис. 13 – SQL-запрос.

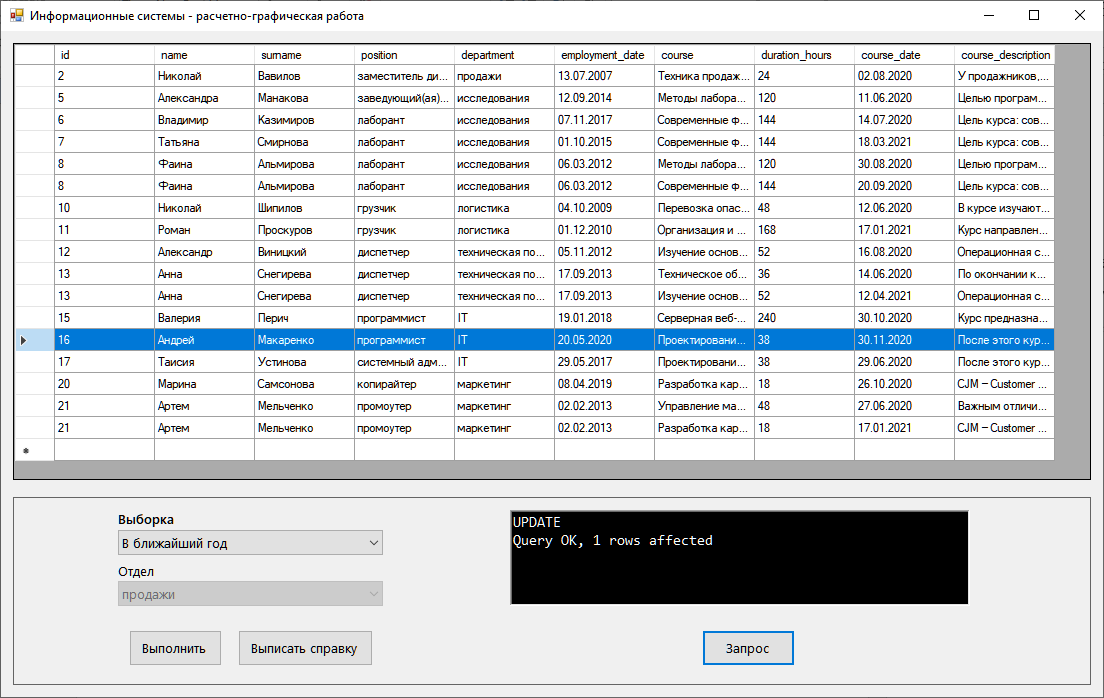


Рис. 14 – Сообщение об успешности операции.

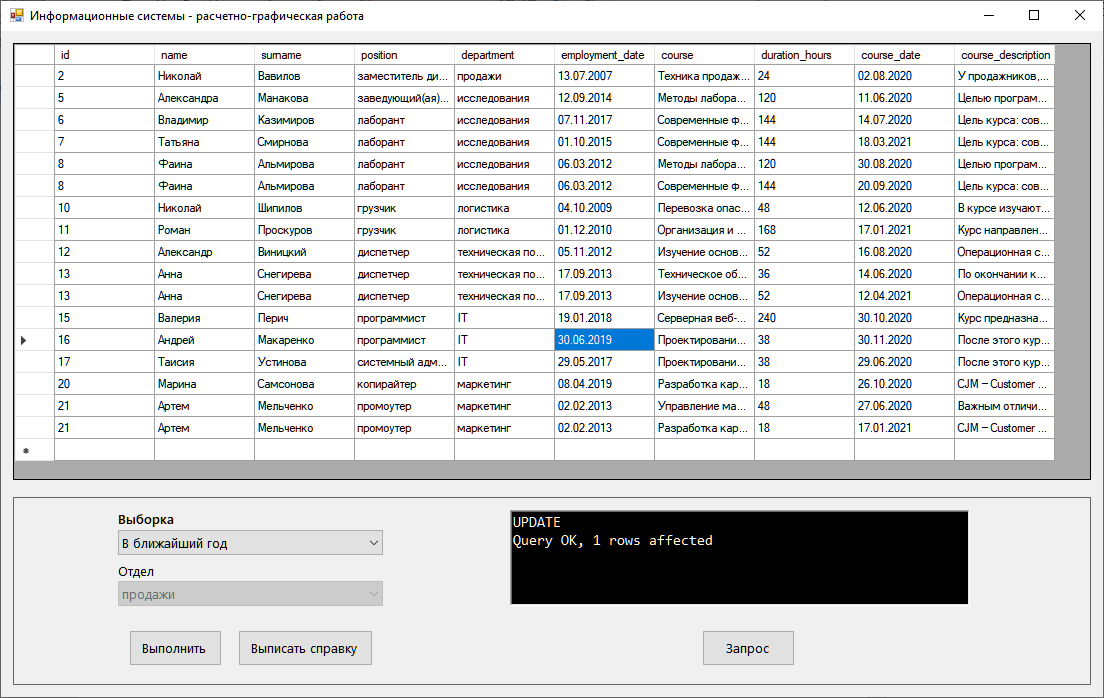


Рис. 15 – Повторная выборка (измененные данные отмечены синим цветом).

# Выводы

В ходе выполнения расчетно-графической работы была разработана информационная система, которая автоматизирует выдачу справок о пройденных сотрудниками курсах. Система позволяет производить удобочитаемую для пользователя выборку информации и создавать документы с готовыми справками. Для продвинутых пользователей также предусмотрено изменение данных в БД. Для работы информационной системы необходим .NET Framework 4.5.2.

# Список источников

1. MySQL Documentation // MySQL URL: https://dev.mysql.com/doc/ (дата обращения: 06.06.2021).
2. Документация по .NET // Microsoft Docs URL: https://docs.microsoft.com/ru-ru/dotnet/ (дата обращения: 06.06.2021).
3. Руководство по ADO.NET и работе с базами данных // METANIT.COM - Сайт о программировании URL: https://metanit.com/sharp/adonet/ (дата обращения: 06.06.2021).

Листинг

Класс «Program»

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace IS\_RGR

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Класс «Form1»

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.Diagnostics;

namespace IS\_RGR

{

public partial class Form1 : Form

{

int selectedRowCount = 0;

string sql = "CALL all\_certificates";

public Form1()

{

InitializeComponent();

}

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

{

switch(comboBoxSelect.SelectedIndex)

{

case 0:

comboBoxDepartment.Enabled = false;

sql = "CALL all\_certificates";

break;

case 1:

comboBoxDepartment.Enabled = false;

sql = "CALL certificates\_coming\_month";

break;

case 2:

comboBoxDepartment.Enabled = false;

sql = "CALL certificates\_coming\_year";

break;

case 3:

comboBoxDepartment.Enabled = false;

sql = "CALL certificates\_after\_year";

break;

case 4:

comboBoxDepartment.Enabled = true;

sql = "CALL certificates\_by\_department('" + comboBoxDepartment.SelectedItem + "')";

break;

}

Debug.WriteLine(sql);

}

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

{

sql = "CALL certificates\_by\_department('" + comboBoxDepartment.SelectedItem + "')";

Debug.WriteLine(sql);

}

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

{

dataGridView1.DataSource = DatabaseConnection.Select(sql);

}

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

{

selectedRowCount = dataGridView1.CurrentCell.RowIndex;

Debug.WriteLine(selectedRowCount);

}

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

{

DataGridViewCellCollection cells = dataGridView1.Rows[selectedRowCount].Cells;

new Certificate(

cells[1].Value.ToString(),

cells[2].Value.ToString(),

cells[3].Value.ToString(),

cells[4].Value.ToString(),

cells[6].Value.ToString(),

cells[7].Value.ToString(),

DateTime.Parse(cells[8].Value.ToString()).ToString("dd/MM/yyyy")

);

}

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

{

richTextBox1.Text = DatabaseConnection.Query(richTextBox1.Text);

}

}

}

Класс «DatabaseInfo»

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Diagnostics;

namespace IS\_RGR

{

class DatabaseInfo

{

private string host;

private int port;

private string dbname;

private string username;

private string password;

public DatabaseInfo ()

{

host = "localhost";

port = 3306;

dbname = "training\_center";

username = "root";

password = "admin";

}

public string GetDBInfo()

{

string info = "server=" + host +

"; database=" + dbname +

"; port=" + port +

"; userid=" + username +

"; password=" + password;

return info;

}

}

}

Класс «DatabaseConnection»

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

using MySql.Data.MySqlClient;

using System.Data;

namespace IS\_RGR

{

class DatabaseConnection

{

public static MySqlConnection conn;

private static MySqlCommand cmd;

public static DataTable Select(string sql)

{

DataSet ds = new DataSet();

try

{

Connect();

MySqlDataAdapter adapter = new MySqlDataAdapter(sql, conn);

adapter.Fill(ds);

Close();

}

catch (Exception e)

{

Debug.WriteLine("Error: " + e);

}

return ds.Tables[0];

}

public static string Query(string sql)

{

string queryType = sql.Split()[0].ToUpper();

int rowCount = 0;

try

{

Connect();

cmd = new MySqlCommand(sql, conn);

rowCount = cmd.ExecuteNonQuery();

Close();

}

catch (Exception e)

{

return "Error: " + e.Message;

}

return queryType + "\nQuery OK, " + rowCount + " rows affected";

}

public static void Connect ()

{

string parameters = new DatabaseInfo().GetDBInfo();

conn = new MySqlConnection(parameters);

try

{

conn.Open();

}

catch (Exception e)

{

Debug.WriteLine("Error: " + e.Message);

}

}

public static void Close ()

{

try

{

conn.Close();

}

catch (Exception e)

{

Debug.WriteLine("Error: " + e.Message);

}

}

}

}

Класс «Certificate»

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Diagnostics;

using Xceed.Words.NET;

using Xceed.Document.NET;

namespace IS\_RGR

{

class Certificate

{

public Certificate(string name, string surname, string position,

string department, string course, string duration, string date)

{

string fileName = name + "\_" + surname + "\_" + department + "\_" + date + ".docx";

DocX document = DocX.Create(fileName);

Formatting titleFormat = new Formatting();

titleFormat.FontFamily = new Font("Times New Roman");

titleFormat.Size = 18;

Formatting textFormat = new Formatting();

textFormat.FontFamily = new Font("Times New Roman");

textFormat.Size = 14;

document.InsertParagraph("СПРАВКА", false, titleFormat)

.Alignment = Alignment.center;

document.InsertParagraph("");

document.InsertParagraph(name + " " + surname + ",", false, textFormat)

.Alignment = Alignment.center;

document.InsertParagraph(position + " отдела " + department + ",", false, textFormat)

.Alignment = Alignment.center;

document.InsertParagraph("успешно прослушал(а) курс «" + course + "»", false, textFormat)

.Alignment = Alignment.center;

document.InsertParagraph("в объеме " + duration + " часов.", false, textFormat)

.Alignment = Alignment.center;

document.InsertParagraph("");

document.InsertParagraph(date, false, textFormat)

.Alignment = Alignment.right;

document.Save();

try

{

Process.Start("WINWORD.EXE", fileName);

}

catch (Exception) { }

}

}

}