УДК 004

Создание локального приложения для управления БД на языке программирования C#

Эрдман Александр Алексеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В статье рассмотрен процесс создания приложения для локального управления базами данных. Приложения написано на языке программирования C#, с использованием библиотеки WinForms и дополнительного модуля FastReport для генерации отчётов. Результатом исследования будет являться приложения для управления базами данных и описание его работы.
Ключевые слова: C#, WinForms, приложение, БД

Creating a local database management application in a programming language C#

Erdman Alexander Alekseevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The article describes the process of creating an application for local database management. The application is written in the C# programming language, using the WinForms library and an additional FastReport module for generating reports. The result of the study will be a database management application and a description of its work.
Keywords: C#, WinForms, application, database

1 Введение

1.1 Актуальность

На сегодняшний день базы данных являются чуть ли не основным источником хранения информации. Например, каждый интернет-магазин имеет свою базу данных как минимум товаров и заказов. Помимо магазинов, базы данных активно используют и другие организации, например высшие учебные заведения. С учётом большого количества информации о студентах, хранения в виде обычного Word документа становится крайне неудобным. Взять к примеру сайт любого ВУЗа. В данных условиях буду уместны только базы данных, так как они имеют огромное преимущество – базы данных способны работать по средством SQL-запросов, то есть сайт, использую определённые скрипты, может спокойно работать с базой данных. На

локальном уровне базам данных также отдают приоритет, если речь идёт о большом количестве информации. Зачастую удобно взять готовую базу данных и отредактировать её локально, не имея при этом никаких задержек при интернет-соединении и других негативных сторон удалённого редактирования БД. В данной статье будет рассмотрен способ, при котором можно локально работать с БД, в частности будет написана программа для управления БД.

1.2 Обзор исследований

Д.М. Лопатин реализовал простой интерфейс на основе библиотеки WinForms [1]. О.М. Коновалов и А.А. Фоминых разработали систему на базе библиотеки WinForms, а также привели пример реализации базы данных MSSQL [2]. Д.А. Михайличенко предложил метод организации архитектуры приложений по принципу разделения пользовательского интерфейса и модели данных [3]. Н.Ж. Ганижева провела анализ современных систем управления базами данных [4]. А.В. Неустроев описал принцип работы языка программирования C# с базой данных MSSQL server [5]. А.А. Коваль рассмотрел основные достоинства и недостатки наиболее популярных на сегодняшний день приложений для создания отчетов [6]. Т.В. Ромашкина и В.И. Шагалиев показали базовые принципы работы языка программирования C# и его использование при создании приложений для Windows [7].

1.3 Цель исследования

Целью исследования является создание приложения на языке программирования C# с использование библиотеки WinForms и модуля FastReport для локального управления базой данных.

2 Материалы и методы

Для создания приложения используется язык программирования C#, модуль FastReport для генерации отчётов и библиотека WinForms. В качестве IDE используется Microsoft Visual Studio 2022.

3 Результаты и обсуждения

Для создания приложения сначала нужно установить Visual Studio 2022. Сделать это с официального сайта. После установки IDE нужно создать новый проект и выбрать язык программирования C# и шаблон Windows Forms App (.NET Framework) (рис 1).

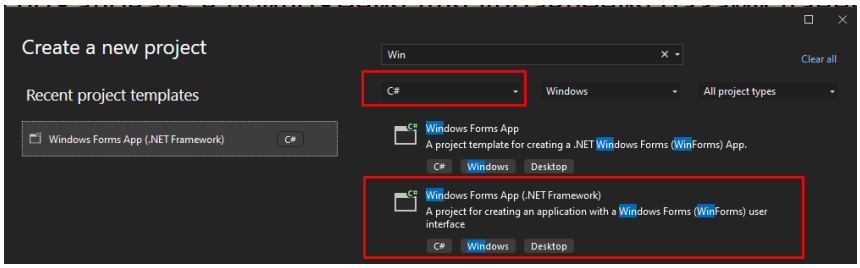


Рисунок 1. Создание проекта

После создания проекта будет пустая форма. Для начало нужно добавить элемент tabControl. Он нужен для того, чтобы наше приложение имело вкладки, на которых будет размещён функционал программы. После размещения tabControl нужно создать 4 вкладки с названиями «О программе» - здесь будет размещено краткое описание программы и её возможности; «SQL Запросы» - на данной вкладки можно будет осуществлять SQL запросы к базе данных: «Редактирование БД» - на данной вкладе будет непосредственное управление БД, а именно возможности просмотра таблиц, редактирование записей, добавления записей и кнопка отчётов; «Отчёты» - данная вкладка будет содержать превью отчёта, который можно будет сохранить в нужный пользователю файл или распечатать напрямую. После создания вкладок, можно приступить к их заполнению. В первой вкладке «О программе» разместим описание программы с помощью элемента label, в котором напишем нужный текст (рис. 2).

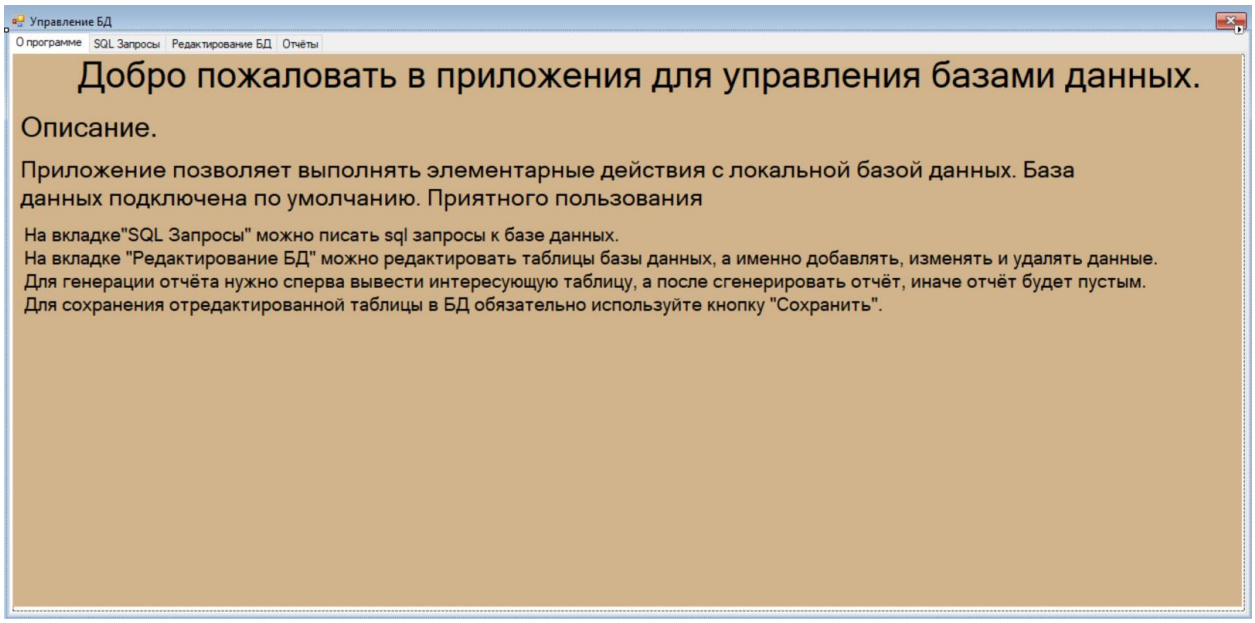


Рисунок 2. Вкладка «О программе»

Далее приступим к оформлению вкладки «SQL Запросы». Для её работоспособности нужна база данных. Для создания базы данных нужно щёлкнуть правой кнопке по «Проводнику» нашего проекта и выбрать «Добавить новый объект», после чего выбрать из предложенного списка базу данных (рис. 3). Тематика базы данных будет журнал посещения лекций студентами. Название базы данных – «Jurnal_Lekcii». После того, как база данных была создана и названа, нужно создать в ней таблицы. Таблиц будет 3: «Студенты», «Пропуски» и «Посещение». Таблицы будут создаваться посредством SQL запроса, так как это надёжнее и удобнее, в отличие от визуального редактора. Визуальный редактор создаём таблицы посредствам внутренних скриптов, которые могут иногда не срабатывать. Это проблема присуща используемой IDE. Чтобы выполнить SQL запрос необходимо в обозреватели серверов выбрать правой кнопкой мыши базу данных и нажать

на «Новый запрос», после чего откроется вкладка, где можно прописывать SQL запросы (рис. 4).

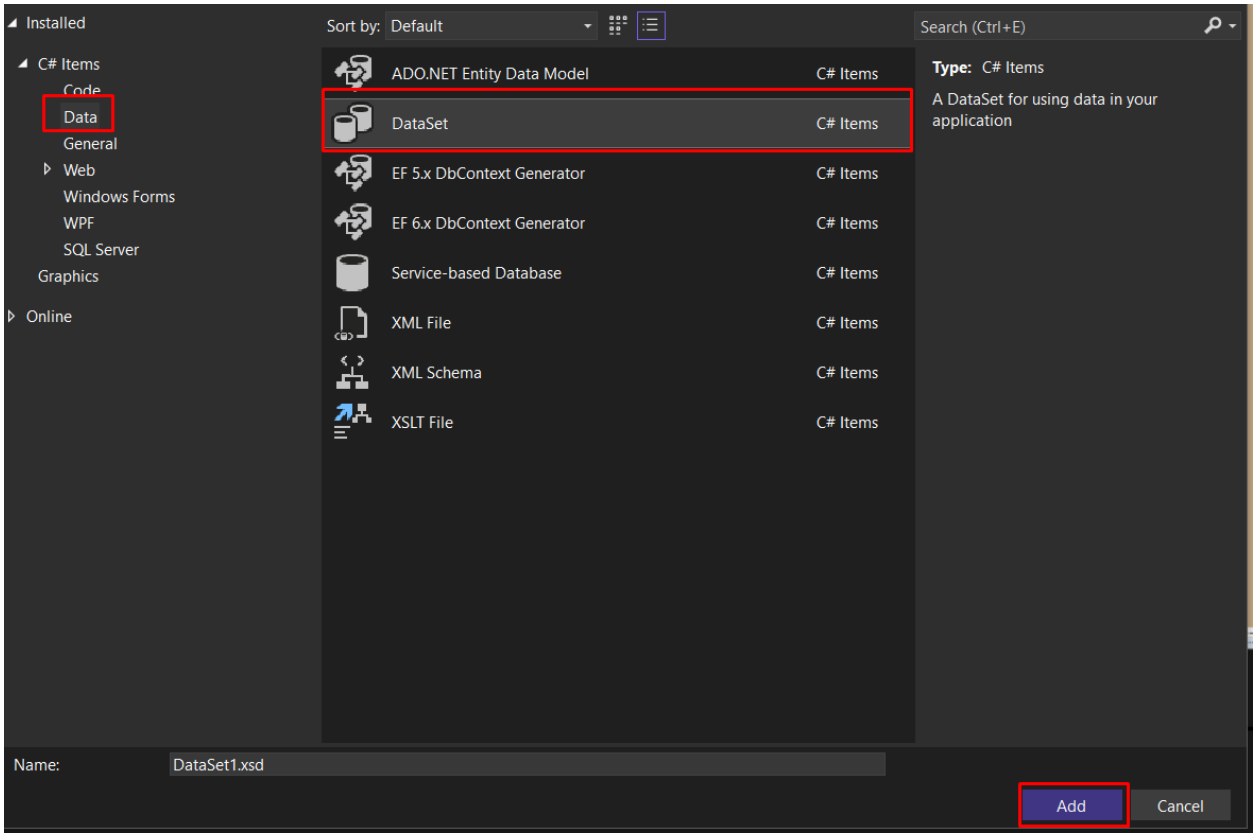


Рисунок 3. Создание базы данных

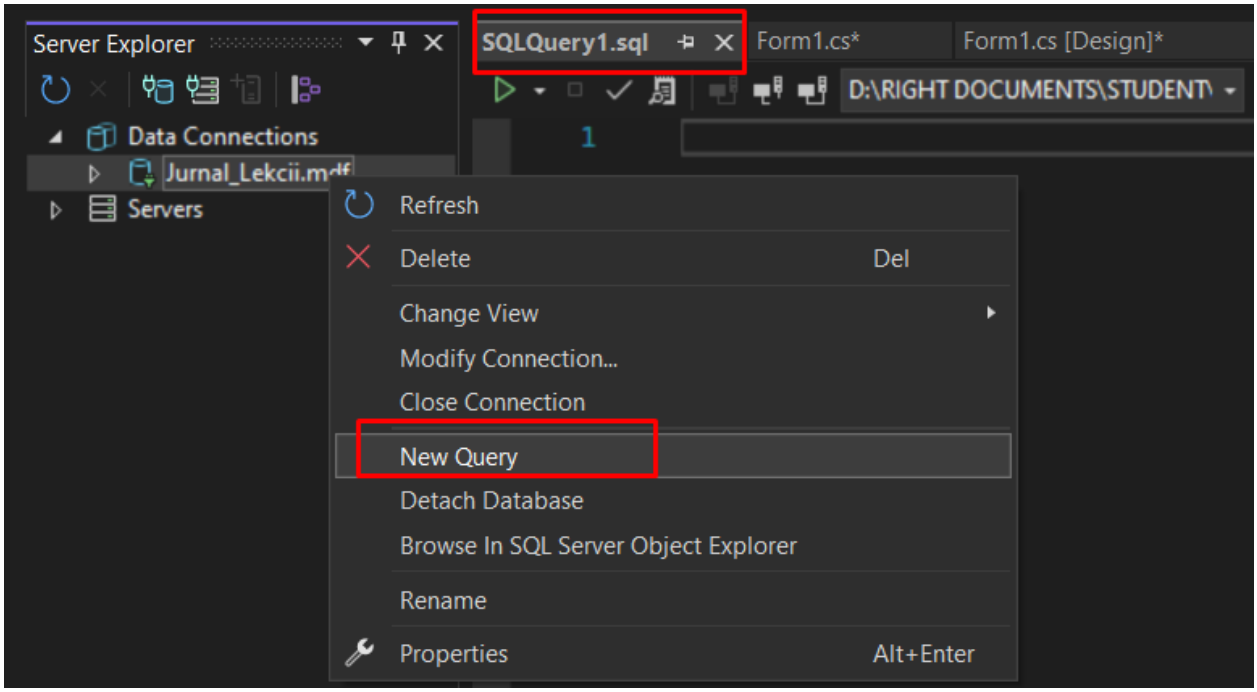


Рисунок 4. Создание нового SQL запроса

SQL запрос на создание таблиц «Студенты», «Посещение», «Пропуски» (рис. 5,6,7). Каждый SQL запрос необходимо отдельно выполнить.

```
CREATE TABLE [dbo].[Students] (
    [Id]           INT           IDENTITY (1, 1) NOT NULL,
    [Name]         NVARCHAR (50) NOT NULL,
    [Email]        NVARCHAR (50) NULL,
    [PhoneNumber]  NVARCHAR (50) NOT NULL,
    [DataBirthday] DATE          NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

Рисунок 5. SQL запрос на создание таблицы «Студенты»

```
CREATE TABLE [dbo].[Pocechenie_Pricyt] (
    [Id]                 INT           IDENTITY (1, 1) NOT NULL,
    [dateOfLecii]        DATE          NOT NULL,
    [nameOfLecii]        NVARCHAR (50) NOT NULL,
    [TimeOfLecii]        NVARCHAR (50) NOT NULL,
    [Pricytcvoval_Name] NVARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

Рисунок 6. SQL запрос на создание таблицы «Посещение»

```
CREATE TABLE [dbo].[Pocechenie_Otcytctvoval] (
    [Id]                 INT           IDENTITY (1, 1) NOT NULL,
    [dateOfLecii]        DATE          NOT NULL,
    [nameOfLecii]        NVARCHAR (50) NOT NULL,
    [TimeOfLecii]        NVARCHAR (50) NOT NULL,
    [Oncytctvoval_Name] NVARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

Рисунок 7. SQL запрос на создание таблицы «Пропуски»

База данных и таблицы готовы. Нужно заполнить таблицы данными. Данные будут случайно сгенерированные. Это облегчит заполнение базы данных. Случайно сгенерированные данные не отразятся на дальнейшем работе приложения. После окончательной настройки базы данных, нужно её подключить в приложение. Для подключения базы данных нужна строка подключения, которую можно получить в свойствах базы данных. После получения строки подключения, нужно занести строку подключения в файл App.config (рис. 8). Далее в главной форме нужно объявить строку подключения следующим образом: «string connectionString = ConfigurationManager.ConnectionStrings["Jurnal"].ConnectionString;»

После подключения базы данных в приложение можно переходить к оформлению вкладки «SQL запросы». На данной вкладки нужно разместить

элементы: dataGridView – данный элемент позволит выводить данные таблиц; textbox – в данном элементе будут прописываться непосредственно SQL запросы; два элемента Button – две кнопки для отправки SQL запроса и очистки поля. Разместим данные элементы следующим образом (рис. 9)

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
    </startup>
    <connectionStrings>
        <add name='Jurnal' connectionString='Data Source=(LocalDB)\MSSQLLocalDB;At
    </connectionStrings>
</configuration>
```

Рисунок 8. Строка подключения в App.config

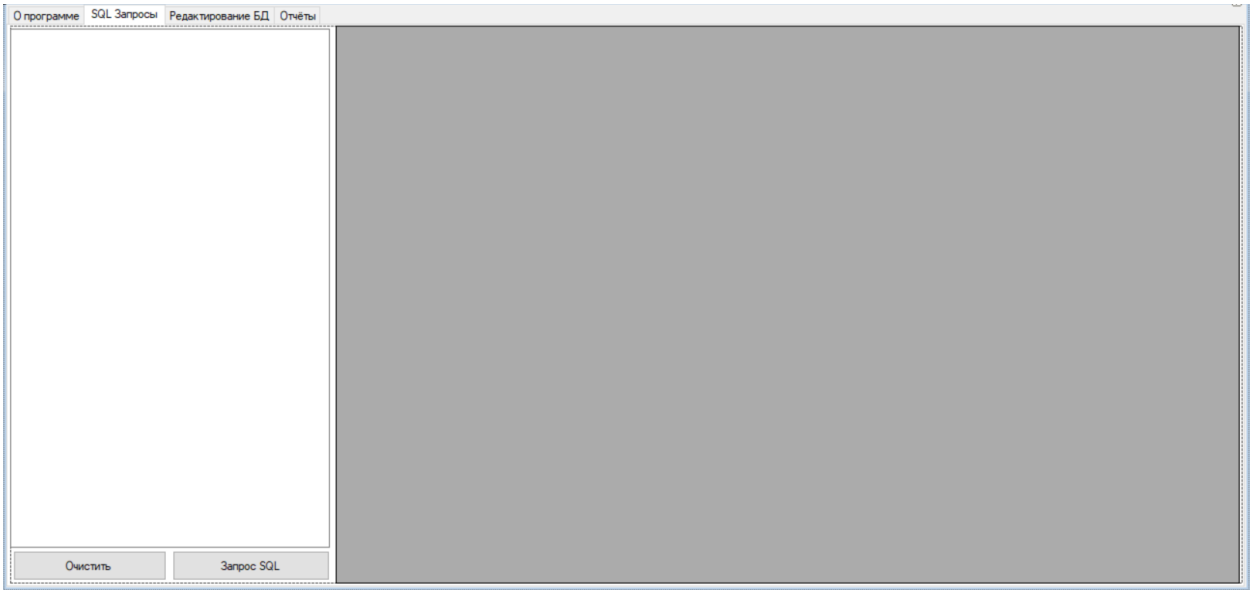


Рисунок 9. Размещение элементов на вкладке «SQL Запросы»

Элементы размещены, теперь нужно их запрограммировать. Из элемента texBox, который имеет имя «InsertSQLZapros», нужно по нажатию на кнопку «Запрос SQL» получить написанный текст (SQL запрос), обработать и вывести результат в dataGridView. Кнопка «Очистить» по нажатию будет очищать textbox. Для реализации данных действий нужно создать два метода. Методы будут созданы в отдельном классе, который будет подключен к главной форме. Создание отдельного класса, в котором будет содержаться методы вкладки «SQL Запросы» обусловлено удобством написания кода и не загромождением кодом главной формы приложения. В дальнейшем для каждой вкладки или для каждой большой совокупности действий будет создан отдельный класс. Все классы подключаются к главной форме. Первый метод осуществляется при нажатии по кнопке «Запрос SQL», второй метод при нажатии «Очистить» (рис. 10).

```
public partial class Form1 : Form
{

    //Кнопка запуска SQL команды
    1 reference
    private void SQLZapros_Click(object sender, EventArgs e)
    {
        SqlDataAdapter dataAdapter = new SqlDataAdapter(
            InsertSQLZapros.Text, connectionString);

        DataSet dataSet = new DataSet();

        dataAdapter.Fill(dataSet);

        dataGridView1.DataSource = dataSet.Tables[0];

    }

    //Кнопка очистки поля ввода SQL запроса
    1 reference
    private void ClearSQL_Click(object sender, EventArgs e)
    {
        InsertSQLZapros.Clear();
    }

}
```

Рисунок 10. Методы нажатий на кнопки «Запрос SQL» и «Очистить»

После реализации функционала на вкладке «SQL Запросы», можно перейти к вкладке «Редактирование БД». Данная вкладка будет самой объёмной в плане функционала, так как тут будут размешены инструменты для управления всеми тремя таблицами базы данных. Для реализации инструментов, необходимо добавить элементы: dataGridView – в данном элементе будут размещаться данные таблиц; button – будут иметь свои методы при нажатии на выполнение конкретных действий; texBox – данные элементы нужны для возможности добавления новых записей в базу данных; dateTimePicker – является вспомогательным элементом для удобного выбора даты; label – элементы, выполняющие роль подписей и создающие дизайн приложения. После размещения и названия всех элементов, вкладка будет иметь следующий вид (рис. 11). Для дальнейшего программирования вкладки, необходимо объявить строки sql1, sql2, sql3, которые будут иметь SQL запросы на вывод данных из каждой таблицы. Это нужно в дальнейшем, для показа данных. Также нужно объявить DataSet ds1, ds2, ds3. Данные датасеты будут заполняться данными из соответствующих таблиц (рис. 12).

Запрограммируем элементы для таблицы «Студенты». Для этого нужно создать новый класс. Далее создадим метод нажатия на кнопку «Добавить» в таблицу «Студенты», который будет добавлять данные из соответствующих texBox элементов и выводить данные в dataGridView, причём данные в dataGridView будут отсортированы таким образом, что на первом месте будет добавленная новая запись (рис. 13). Для элемента dateTimePicker нужен отдельный метод, который позволит при нажатии на нужную дату перенести её в соответствующий textBox (рис. 14). Кнопка для показа таблицы «Студенты» имеет следующий метод (рис. 15).

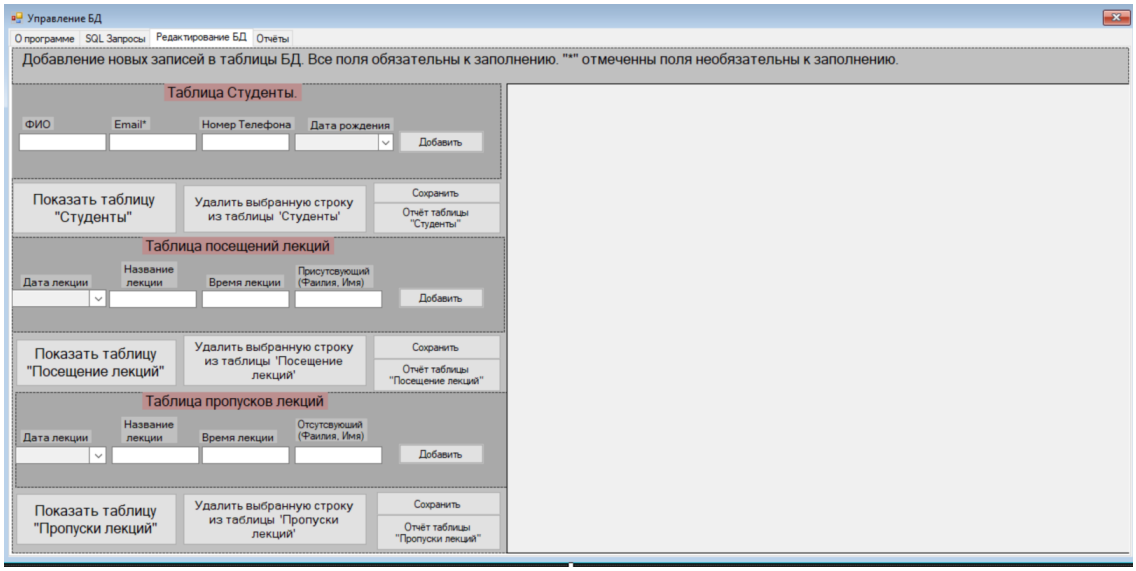


Рисунок 11. Внешний вид вкладки «Редактирование БД» после размещения всех элементов

```
public partial class Form1 : Form
{
    DataSet ds1;
    DataSet ds2;
    DataSet ds3;
    SqlDataAdapter adapter;
    SqlCommandBuilder commandBuilder;
    //объявление строки подключения
    string connectionString = ConfigurationManager.ConnectionStrings["Jurnal"].ConnectionString;
    //Команды показа таблиц
    string sql1 = "SELECT * FROM Students";
    string sql2 = "SELECT * FROM Pocechenie_Pricyt";
    string sql3 = "SELECT * FROM Pocechenie_Otcytctvoval";
```

Рисунок 12. Объявление строк в файле Form1.cs

```
private void button1_Click(object sender, EventArgs e)
{

    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        connection.Open();
        adapter = new SqlDataAdapter(sql1, connection);

        ds1 = new DataSet();
        SqlCommand command = new SqlCommand($"INSERT INTO [Students] (Name, Email, PhoneNumber, DataBirthday) VALUES (@Name, @Email, @PhoneNumber, @DataBirthday)", connection);

        if (textBox1.Text == "" || textBox1.Text == " " || textBox3.Text == "" || textBox3.Text == " " || textBox4.Text == "" || textBox4.Text == " ")
        {
            MessageBox.Show("Обязательное поле не заполнено!");
        }
        else
        {
            command.Parameters.AddWithValue("Name", textBox1.Text);
            command.Parameters.AddWithValue("Email", textBox2.Text);
            command.Parameters.AddWithValue("PhoneNumber", textBox3.Text);
            command.Parameters.AddWithValue("DataBirthday", textBox4.Text);
            command.ExecuteNonQuery().ToString();
            MessageBox.Show("Данные внесены в таблицу 'Студенты'");
            textBox1.Clear();
            textBox2.Clear();
            textBox3.Clear();
            textBox4.Clear();
        }

        SqlDataAdapter dataAdapter = new SqlDataAdapter(
            "SELECT * FROM Students ORDER BY ID DESC", connectionString);

        dataAdapter.Fill(ds1);
        dataGridView3.DataSource = ds1.Tables[0];
        dataGridView3.AutoResizeColumns();
        dataGridView3.Columns["Id"].ReadOnly = true;
        dataGridView3.Columns[0].HeaderText = "Id";
        dataGridView3.Columns[1].HeaderText = "Имя";
        dataGridView3.Columns[2].HeaderText = "Почта";
        dataGridView3.Columns[3].HeaderText = "Телефон";
        dataGridView3.Columns[4].HeaderText = "Дата рождения";
    }
}
```

Рисунок 13. Реализация метода кнопки «Добавить» в таблицу «Студенты»

```
//Перенос выбранной даты в textBox
1 reference
private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
  textBox4.Text = dateTimePicker1.Value.ToString("yyyy-MM-dd");
}
```

Рисунок 14. Метод переноса даты из dateTimePicker в texBox

```
//кнопка показа таблицы студенты
1 reference
private void ViewStud_Click(object sender, EventArgs e)
{
    dataGridView3.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
    dataGridView3.AllowUserToAddRows = false;

    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        connection.Open();
        adapter = new SqlDataAdapter(sql1, connection);

        ds1 = new DataSet();
        adapter.Fill(ds1);
        dataGridView3.DataSource = ds1.Tables[0];
        dataGridView3.AutoResizeColumns();
        // делаем недоступным столбец id для изменения
        dataGridView3.Columns["Id"].ReadOnly = true;
        dataGridView3.Columns[0].HeaderText = "Id";
        dataGridView3.Columns[1].HeaderText = "Имя";
        dataGridView3.Columns[2].HeaderText = "Почта";
        dataGridView3.Columns[3].HeaderText = "Телефон";
        dataGridView3.Columns[4].HeaderText = "Дата рождения";
    }


}
```

Рисунок 15. Метод по нажатию кнопки показа данных из таблицы «Студенты»

Кнопка удаления строки/строк из таблицы при нажатии должна требовать подтверждения для удаления (рис. 16). После того, как будут совершены действия над таблицей, их нужно сохранить. Для этого необходима кнопка «Сохранить». Данная кнопка также будет запрашивать подтверждение при нажатии (рис. 17). Для элементов редактирования двух других таблиц методы аналогичны, только меняются соответствующие sql(2 и 3) и ds(2 и 3), а также названия элементов textBox и заголовки dataGridView.

```
//кнопка удаления записи из dataGridView (из БД не удаляет)
1 reference
private void DeleteStud_Click(object sender, EventArgs e)
{


    DialogResult dialogResult = MessageBox.Show("Вы уверены, что хотите удалить выбранную строку/строки?", "Some Title", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        foreach (DataGridViewRow row in dataGridView3.SelectedRows)
        {
            dataGridView3.Rows.Remove(row);
        }
        MessageBox.Show("Удаление из списка выполено, сохраните обязательно изменения");
    }
    else if (dialogResult == DialogResult.No)
    {
        MessageBox.Show("Удаление отменено");
    }

}
```

Рисунок 16. Метод по нажатию кнопки удаления записи из таблицы «Студенты»

```
//кнопка сохранения изменений редактирования
1 reference
private void SaveStud_Click(object sender, EventArgs e)
{

    DialogResult dialogResult = MessageBox.Show("Вы уверены, что хотите внести изменения?", "Some Title", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            connection.Open();
            adapter = new SqlDataAdapter(sql1, connection);
            commandBuilder = new SqlCommandBuilder(adapter);
            adapter.InsertCommand = new SqlCommand("sp_CreateUser", connection);
            adapter.InsertCommand.CommandType = CommandType.StoredProcedure;
            adapter.InsertCommand.Parameters.Add(new SqlParameter("@Name", SqlDbType.NVarChar, 50, "Name"));
            adapter.InsertCommand.Parameters.Add(new SqlParameter("@Email", SqlDbType.NVarChar, 50, "Email"));
            adapter.InsertCommand.Parameters.Add(new SqlParameter("@PhoneNumber", SqlDbType.NVarChar, 50, "PhoneNumber"));
            adapter.InsertCommand.Parameters.Add(new SqlParameter("@DataBirthday", SqlDbType.Date, 10, "DataBirthday"));

            SqlParameter parameter = adapter.InsertCommand.Parameters.Add("@Id", SqlDbType.Int, 0, "Id");
            parameter.Direction = ParameterDirection.Output;

            adapter.Update(ds1);

        }
    }
    else if (dialogResult == DialogResult.No)
    {
        MessageBox.Show("Изменения НЕ внесены");
    }
}
```

Рисунок 17. Метод по нажатию кнопки сохранения изменений в таблице «Студенты»

После релализации методов для элементов управления таблицами, можно перейти к программированию отчёта. Отчёт будет содержать название редактируемой таблицы и полный список данных таблицы. Для того, чтобы работать с отчётом, нужно подклчючить соответсвуюший модуль. Для этого нужно воспользоваться инструментом NuGet и установить компонент FastReport.Compat (рис. 18). После установки компонента, нужно подключить модуль в главной форме:

using FastReport;

После подключения модуля, нужно скачать программу FastReport. В директории программы FastReport находятся файлы .dll содержащие необходимые элементы. Чтобы их добавить, нужно в панели элементов правой кнопкой мыши создать новую группу и добавить новые элементы. В

открывшемся окне необходимо нажать на кнопку "Обзор" и выбрать файлы FastReport.dll, FastReport.Web.dll, которые находятся по следующему пути: C:\Program Files\FastReports\FastReport.Net.

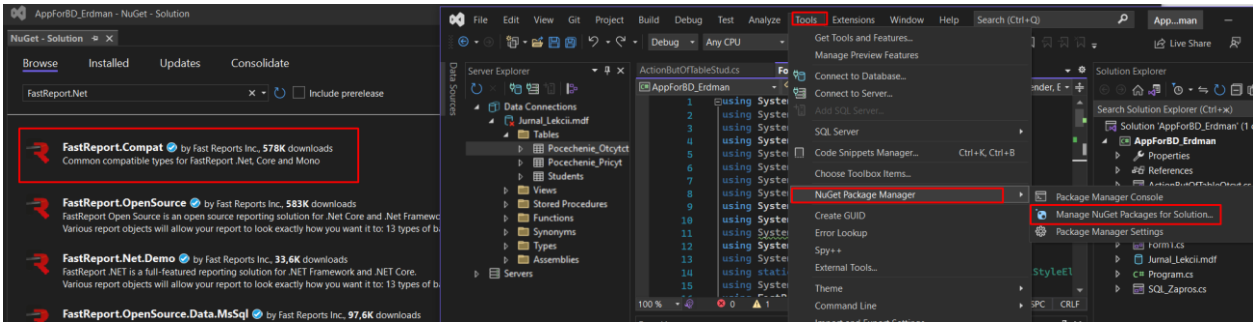


Рисунок 18.

После того, как элементы импортированы в IDE, нужно добавить два элемента: report и previewControl. Первый компонент добавляется вообщем на форму и предоставляет функционал модуля FastReport для данной формы, а элемент previewControl добавлятся в последнюю вкладку приложения «Отчёты» и отвечает за показ превью отчёта, а также предоставляет иструменты, такие как печать или сохранить отчёт в нужном формате. После проделанных дейсвтий, нужно создать три файла report.frx (report1.frx, report2.frx, report3.frx) в папке, которая находится по следующему пути: Название_Проекта\Название_Проекта\bin\Debug.

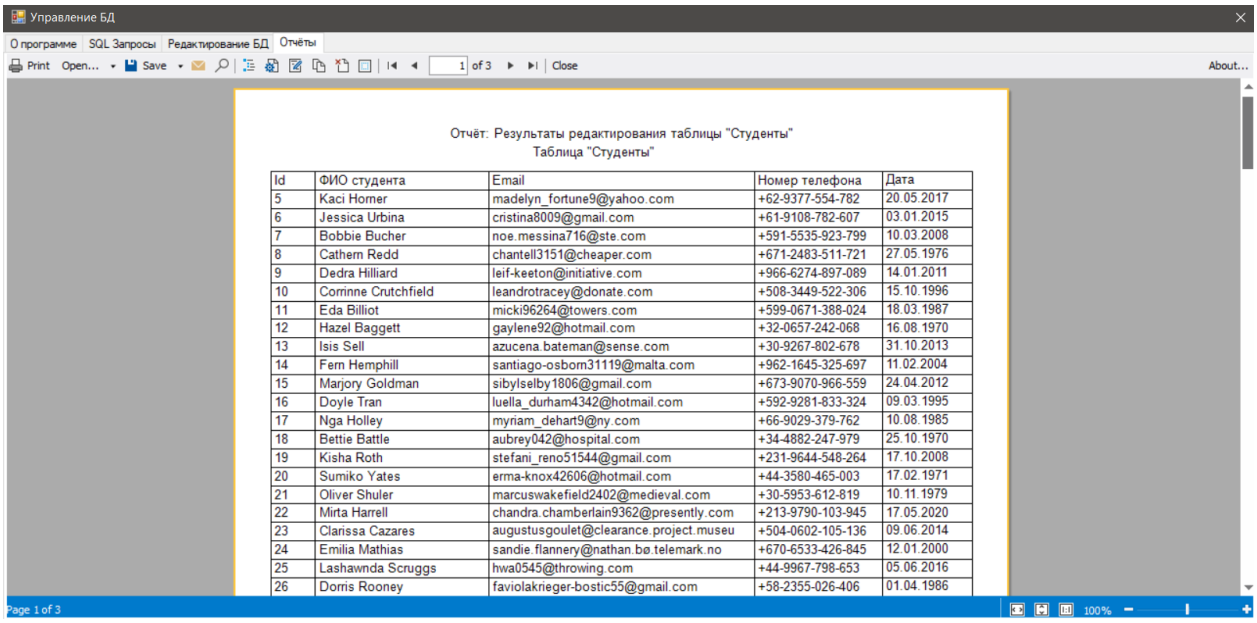
Данные файлы служат шаблонами для заполнения. Их нужно отредактировать для каждой таблицы через программу FastReport. В каждом шаблоне обязательно нужно создать table1, так как в дальнейшем в методе кнопки «Отчёт», в процессе создания отчёта данная таблица будет заполнятся данными из таблицы нашей базы данных.

Теперь нужно вернуться на вкладку «Редактирование БД» и создать метод для кнопки отчёта для таблицы «Студенты» (рис. 19).

```
private void Otchet_Click(object sender, EventArgs e)
{


    report.Load(Application.StartupPath + "\\ReportMyStudent.frx"); // загружаем файл отчета
    report.Preview = previewControl1;

    //выводим данные из datagrid
    FastReport.Table.TableBase Mytbl = (FastReport.Table.TableBase)this.report.Report.FindObject("Table1");

    Mytbl.RowCount = this.dataGridView3.RowCount; // синхронизируем кол-во строк таблиц
    Mytbl.ColumnCount = this.dataGridView3.ColumnCount; // синхронизируем кол-во колонок таблиц

    for (short RowShag = 0; RowShag != this.dataGridView3.RowCount - 0; RowShag++)
    {
        for (short ColShag = 0; ColShag != this.dataGridView3.ColumnCount; ColShag++)
        {



            Mytbl[ColShag, RowShag].Text = Convert.ToString(this.dataGridView3[ColShag, RowShag].Value); // переносим данные по ячейкам
            Mytbl[ColShag, RowShag].Border.Lines = BorderLines.All; // границы ячеек таблицы отчета

        }
    }

    report.Show();// отображаем отчет
}
```

Рисунок 19. Создание метода при нажатии кнопки «Отчёт» для таблицы «Студенты»

Методы кнопкок отчёта для оставшихся двух таблиц аналогичны. После реализации методов кнопок отчётов, можно проверить их функциональность. Для этого сначала нужно нажать на кнопку «Показать таблицу» и нажать на кнопку отчёт. Для примера можно протестировать отчёт таблицы «Студенты». Для этого нужно нажать на кнопку «Показать таблицу Студенты», далее нажать на кнопку «Отчёт», после перейти во вкладку «Отчёты» (рис. 20). Как видно отчётность работает.



Id	ФИО студента	Email	Номер телефона	Дата
5	Kaci Horner	madelyn_fortune9@yahoo.com	+62-9377-554-782	20.05.2017
6	Jessica Urbina	cristina8009@gmail.com	+61-9108-782-607	03.01.2015
7	Bobbie Bucher	noe.messina716@ste.com	+591-5535-923-799	10.03.2008
8	Cathern Redd	chantell3151@cheaper.com	+671-2483-511-721	27.05.1976
9	Dedra Hilliard	leif-keeton@initiative.com	+966-6274-897-089	14.01.2011
10	Corrinne Crutchfield	leandrotracey@donate.com	+508-3449-522-306	15.10.1996
11	Eda Billiot	micki96264@towers.com	+599-0671-388-024	18.03.1987
12	Hazel Baggett	gaylene92@hotmail.com	+32-0657-242-068	16.08.1970
13	Isis Sell	azucena.bateman@sense.com	+30-9267-802-678	31.10.2013
14	Fern Hemphill	santiago-osborn31119@malta.com	+962-1645-325-697	11.02.2004
15	Marjory Goldman	sibylselby1806@gmail.com	+673-9070-966-559	24.04.2012
16	Doyle Tran	luella_durham4342@hotmail.com	+592-9281-833-324	09.03.1995
17	Nga Holley	myriam_dehart9@ny.com	+66-9029-379-762	10.08.1985
18	Bettie Battle	aubrey042@hospital.com	+34-4882-247-979	25.10.1970
19	Kisha Roth	stefani_reno51544@gmail.com	+231-9644-548-264	17.10.2008
20	Sumiko Yates	erma-knox42606@hotmail.com	+44-3580-465-003	17.02.1971
21	Oliver Shuler	marcuswakefield2402@medieval.com	+30-5953-612-819	10.11.1979
22	Mirta Harrell	chandra.chamberlain9362@presently.com	+213-9790-103-945	17.05.2020
23	Clarissa Cazares	augustusgoulet@clearance.project.museu	+504-0602-105-136	09.06.2014
24	Emilia Mathias	sandie.flannery@nathan.bø.telemark.no	+670-6533-426-845	12.01.2000
25	Lashawnda Scruggs	hwa0545@throwing.com	+44-9967-798-653	05.06.2016
26	Dorris Rooney	faviolakrieger-bostic55@gmail.com	+58-2355-026-406	01.04.1986

Рисунок 20. Реализация отчёта таблицы «Студенты»

Таким образом на основе Windows Forms App (.NET Framework), языка программирования C# и модуля FastReport.Compat было разработано приложения для локального управления базой данных. В приложение было реализованы следующие возможности: обращение к базе данных посредствам SQL запросов; редактирование базы данных – удаление, добавление и изменение существующих записей; формирование отчётов о изменениях таблиц.

Библиографический список

1. Лопатин Д.К. Разработка простых интерфейсов с использованием библиотеки WinForms // Журнал научных публикаций аспирантов и докторантов. 2014. № 3 (93). С. 265-267.
2. Коновалов О.М., Фоминых А.А. Разработка регистратуры поликлиники на языке программирования c# с использованием WinForms и базы данных MS SQL // В сборнике: Актуальные проблемы прикладной информатики в образовании, экономике, государственном и муниципальном управлении. Материалы Международной научной конференции. Под редакцией А.Ю. Юдинцева, Г.Н. Трошкиной. Барнаул, 2022. С. 52-58.
3. Михайличенко Д.А. Ключевая ошибка проектирования программного

продукта в WinForm Microsoft Visual Studio // Труды Ростовского государственного университета путей сообщения. 2016. № 2. С. 38-40.
4. Ганижева Н.Ж. Обзор и сравнение современных систем управления базами данных // Коррекционно-педагогическое образование: электронный журнал. 2022. № 2 (32). С. 70-72.
5. Неустроев А.В. Работа с базой данных C# // Academy. 2016. № 6 (9). С. 44-46.
6. Коваль А.А. Анализ современных приложений для составления отчетов // В сборнике: Альманах научных работ молодых ученых Университета ИТМО. Материалы XLVI научной и учебно-методической конференции. 2017. С. 96-98.
7. Ромашкина Т.В., Шагалиев В.И. Создание приложений для Windows средствами языка C# // Хроники объединенного фонда электронных ресурсов Наука и образование. 2015. № 12 (79). С. 115.