Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

**ПРОГРАММНОЕ ПОДКЛЮЧЕНИЕ К СЕРВЕРУ POSTGRESQL**

Отчёт о лабораторной работе № 4

по дисциплине «Базы данных»

Выполнил: студент гр. 430-2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Швоева Д. С.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Проверил: ассистент каф. АСУ

\_\_\_\_\_\_\_\_\_\_\_\_\_Яблонский Я. В.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Томск 2022

**ОГЛАВЛЕНИЕ**

[1 ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ 2](#_Toc122462758)

[2 ОПИСАНИЕ ТРЕБОВАНИЙ К ОТЧЕТАМ 3](#_Toc122462759)

[3 ЛИСТИНГ 4](#_Toc122462760)

[4 ОПИМАНИЕ ПРОЦЕССА РЕАЛИЗАЦИИ 8](#_Toc122462761)

[ВЫВОД 10](#_Toc122462762)

# ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

Научиться создавать программы, взаимодействующие с сервером СУБД.

# ОПИСАНИЕ ТРЕБОВАНИЙ К ОТЧЕТАМ

На рисунке 2.1 показаны требования к отчетам.

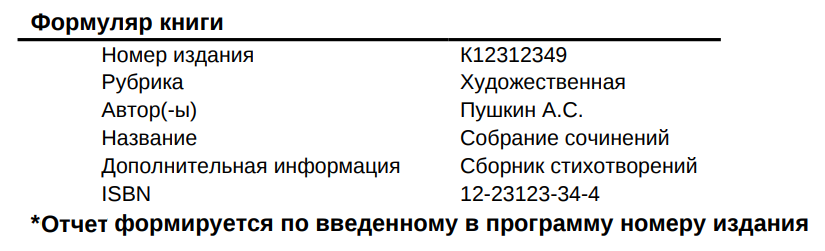


Рисунок 2.1 – Вид отчета

# ЛИСТИНГ

using System.Data;

using Npgsql;

namespace SUBSCRIPTION

{

public partial class Form1 : Form

{

private NpgsqlConnection \_connection;

private NpgsqlCommandBuilder \_command;

private NpgsqlDataAdapter \_dataAdapter;

private DataSet \_dataSet;

private bool \_newRowAdding = false;

public Form1()

{

InitializeComponent();

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

ReloadData();

}

private void LoadData()

{

try

{

\_dataAdapter = new NpgsqlDataAdapter("SELECT \*, 'Delete' AS Command FROM books",

\_connection);

\_command = new NpgsqlCommandBuilder(\_dataAdapter);

\_command.GetDeleteCommand();

\_command.GetUpdateCommand();

\_command.GetInsertCommand();

\_dataSet = new DataSet();

\_dataAdapter.Fill(\_dataSet, "books");

dataGridView1.DataSource = \_dataSet.Tables["books"];

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

{

DataGridViewLinkCell cell = new DataGridViewLinkCell();

dataGridView1[7, i] = cell;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void ReloadData()

{

try

{

\_dataSet.Tables["books"].Clear();

\_dataAdapter.Fill(\_dataSet, "books");

dataGridView1.DataSource = \_dataSet.Tables["books"];

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

{

DataGridViewLinkCell cell = new DataGridViewLinkCell();

dataGridView1[7, i] = cell;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

\_connection = new NpgsqlConnection(

"Host=localhost;" +

"Username=postgres;" +

"Password=sds61224;" +

"Database=subscription"

);

\_connection.Open();

LoadData();

}

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

{

try

{

if (e.ColumnIndex == 7)

{

string task = dataGridView1.Rows[e.RowIndex].Cells[7].Value.ToString(); //9

if (task == "Delete")

{

if (MessageBox.Show("Удалить эту строку?", "Удаление",

MessageBoxButtons.YesNo, MessageBoxIcon.Question)

== DialogResult.Yes)

{

int rowIndex = e.RowIndex;

dataGridView1.Rows.RemoveAt(rowIndex);

\_dataSet.Tables["books"].Rows[rowIndex].Delete();

\_dataAdapter.Update(\_dataSet, "books");

}

}

else if (task == "Insert")

{

int rowIndex = dataGridView1.Rows.Count - 2;

DataRow row = \_dataSet.Tables["books"].NewRow();

row[0] = dataGridView1.Rows[rowIndex].Cells[0].Value;

row[1] = dataGridView1.Rows[rowIndex].Cells[1].Value;

row[2] = dataGridView1.Rows[rowIndex].Cells[2].Value;

row[3] = dataGridView1.Rows[rowIndex].Cells[3].Value;

row[4] = dataGridView1.Rows[rowIndex].Cells[4].Value;

row[5] = dataGridView1.Rows[rowIndex].Cells[5].Value;

row[6] = dataGridView1.Rows[rowIndex].Cells[6].Value;

\_dataSet.Tables["books"].Rows.Add(row);

\_dataSet.Tables["books"].Rows.RemoveAt(\_dataSet.Tables["books"].Rows.Count - 1);

dataGridView1.Rows.RemoveAt(dataGridView1.Rows.Count - 2);

dataGridView1.Rows[e.RowIndex].Cells[7].Value = "Delete";

\_dataAdapter.Update(\_dataSet, "books");

\_newRowAdding = false;

}

else if (task == "Update")

{

int r = e.RowIndex;

\_dataSet.Tables["books"].Rows[r][0] = dataGridView1.Rows[r].Cells[0].Value;

\_dataSet.Tables["books"].Rows[r][1] = dataGridView1.Rows[r].Cells[1].Value;

\_dataSet.Tables["books"].Rows[r][2] = dataGridView1.Rows[r].Cells[2].Value;

\_dataSet.Tables["books"].Rows[r][3] = dataGridView1.Rows[r].Cells[3].Value;

\_dataSet.Tables["books"].Rows[r][4] = dataGridView1.Rows[r].Cells[4].Value;

\_dataSet.Tables["books"].Rows[r][5] = dataGridView1.Rows[r].Cells[5].Value;

\_dataSet.Tables["books"].Rows[r][6] = dataGridView1.Rows[r].Cells[6].Value;

\_dataAdapter.Update(\_dataSet, "books");

dataGridView1.Rows[e.RowIndex].Cells[7].Value = "Delete";

}

ReloadData();

}

}

catch

{

}

}

private void dataGridView1\_UserAddedRow(object sender, DataGridViewRowEventArgs e)

{

try

{

if (\_newRowAdding == false)

{

\_newRowAdding = true;

int lastRow = dataGridView1.Rows.Count - 2;

DataGridViewRow row = dataGridView1.Rows[lastRow];

DataGridViewLinkCell cell = new DataGridViewLinkCell();

dataGridView1[7, lastRow] = cell;

row.Cells["Command"].Value = "Insert";

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

try

{

if (\_newRowAdding == false)

{

int rowIndex = dataGridView1.SelectedCells[0].RowIndex;

DataGridViewRow editingRow = dataGridView1.Rows[rowIndex];

DataGridViewLinkCell cell = new DataGridViewLinkCell();

dataGridView1[7, rowIndex] = cell;

editingRow.Cells["Command"].Value = "Update";

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

\_connection.Close();

\_connection.Open();

string bookId = textBox1.Text;

NpgsqlCommand command = new NpgsqlCommand(

"SELECT books.bookid, heading.heading, books.author, books.name, books.information, books.isbn " +

"FROM books JOIN heading ON books.headingid = heading.id " +

$"WHERE bookid = '{bookId}';",

\_connection);

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

string information = string.Empty;

if (!reader.IsDBNull(4))

{

information = reader.GetString(4);

}

string report = $"Номер издания: {reader.GetString(0)}\r\n" +

$"Рубрика: {reader.GetString(1)}\r\n" +

$"Автор(-ы): {reader.GetString(2)}\r\n" +

$"Название: {reader.GetString(3)}\r\n" +

$"Дополнительная информация: {information}\r\n" +

$"ISBN: {reader.GetString(5)}\r\n";

textBox2.Text = report;

}

}

}

}

# ОПИМАНИЕ ПРОЦЕССА РЕАЛИЗАЦИИ

В программе использовалась библиотека Npgsql. Npgsql — это поставщик данных ADO.NET с открытым исходным кодом для PostgreSQL. Она позволяет программам, написанным на C#, Visual Basic, F#, получать доступ к серверу базы данных PostgreSQL. Библиотека реализована на 100% в коде C#, является бесплатной и имеет открытый исходный код. Также доступен поставщик Entity Framework Core, который предоставляет пользователям EF Core некоторые уникальные функции базы данных PostgreSQL. Так же используется библиотека System.Data.

При запуске окна (функция Form1\_Load) происходит подключение к серверу СУБД PostgreSQL. Далее в программе создаются объекты SqlComman для вставки, изменения и удаления в базе данных и запрос на вывод всей информации из таблицы в DataSet, а после в DataGridView. (функция LoadData). Обработка нажатия на toolStripButton (обновление таблицы) происходит в функции ReloadData. В ней происходит очистка таблицы DataSet и ее дальнейшее заполнение по примеру функции LoadData. После создаются функции dataGridView1\_UserAddedRow, dataGridView1\_CellValueChanged, dataGridView1\_CellContentClick для удаления, добавления и изменений элементов в таблице.

Вывод отчета в окне происходит в TextBox (функция button1\_Click). При нажатии кнопки берется номер издания из TextBox, и отправляется запрос на получение данных из таблиц «books» и «heading» о книге с данным номером. Результат записывается в NpgsqlDataReader, после чего отображается в TextBox.

Вид генерируемого программой отчета и изменяемой таблицы показан на рисунке 4.1.

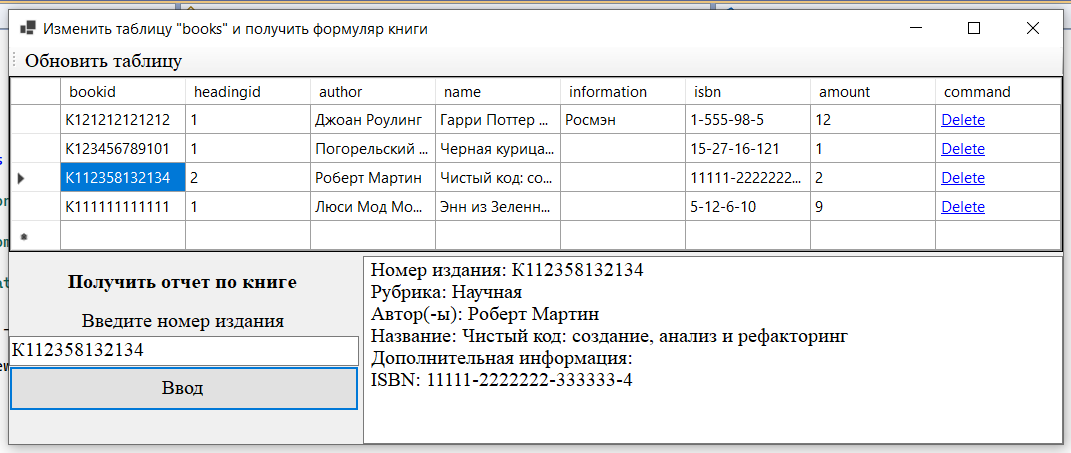


Рисунок 4.1 – Результат работы программы

# ВЫВОД

В ходе выполнения лабораторной работы был рассмотрен способ создания программы, взаимодействующие с сервером СУБД. Так же была разработана программа, в которой выполняется подключение к серверу СУБД PostgreSQL, изменение таблицы «books», формирование отчета.