Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОННИКИ
Факультет компьютерных систем и сетей
Кафедра электронных вычислительных машин

Лабораторная работа №6
Создание приложения для базы данных

Студент: В.С. Шевцов
Преподаватель: Д.В. Куприянова

МИНСК 2024

СОДЕРЖАНИЕ

1 СОЗДАНИЕ ПРИЛОЖЕНИЯ

Для создания приложения был выбран язык C# .Net Core 5.

Для работы с PostgreSQL был установлен пакет Npgsql, обеспечивающий все необходимые функции для работы с бд.

Графический интерфейс был реализован с помощью windows forms.

1.2 Подключение к базе данных

Для подключения к бд требуется указать имя базы данных, пароль, порт на которой база запущена и другую конфигурационную информацию.

Скрипт подключения к базе данных представлен на рисунке 1.1

```
public static NpgsqlConnection conn = new NpgsqlConnection("Server=127.0.0.1;User Id=postgres;Password=1;Port=5432;Database=Parking;");

Ссылок: 9
public static void ExecuteQuery(string sqlCommand, bool isEditable = false, string tableName = "")
{
    NpgsqlDataAdapter adapter = new(sqlCommand, conn);
```

Рисунок 1.1 – Скрипт подключения к базе данных

1.3 Графические окна

Для взаимодействия с базой были созданы следующие окна: окно выбора операции и ее запуск (рисунок 1.2), окно вывода результата запроса (рисунок 1.3), окно выбора исходной таблицы для просмотра (рисунок 1.4), редактирования, удаления и добавления данных (рисунок 1.5).

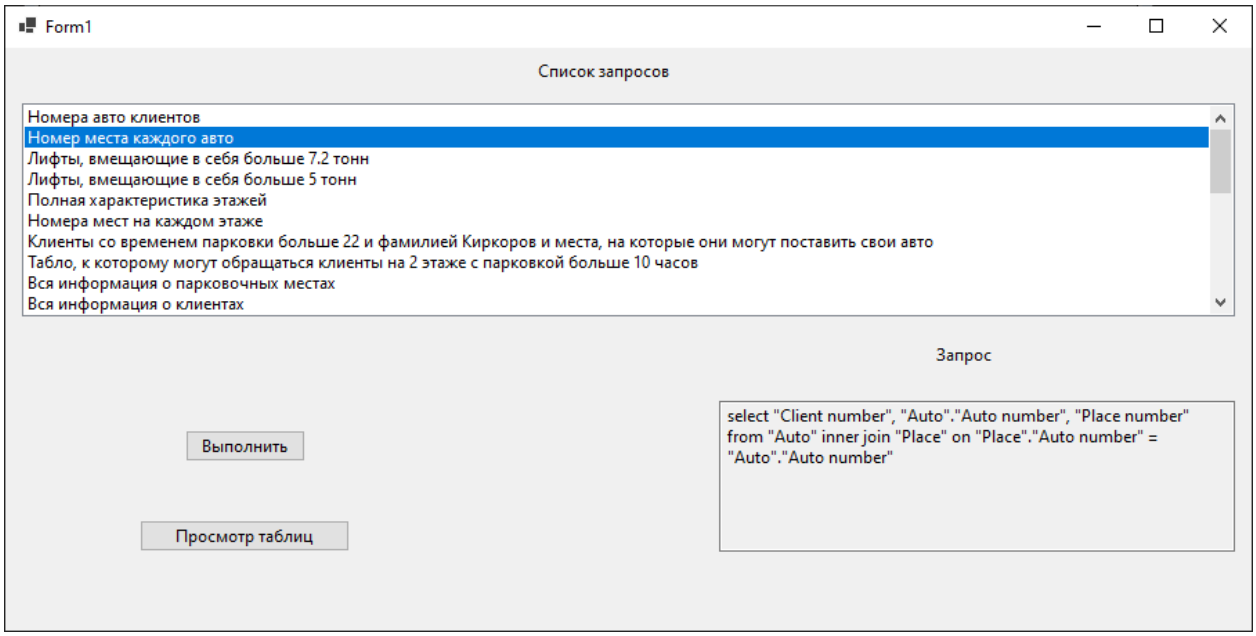

Рисунок 1.2 – окно выбора операции

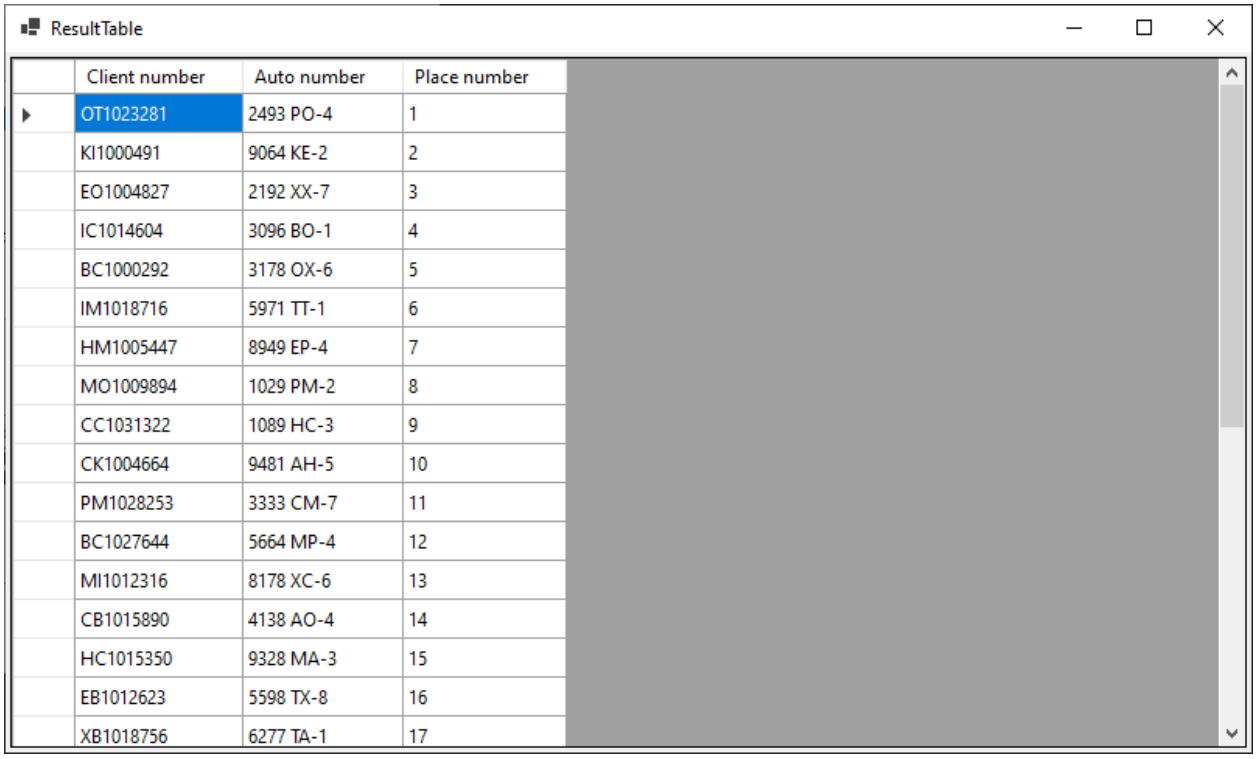

Рисунок 1.3 – окно вывода результата операции

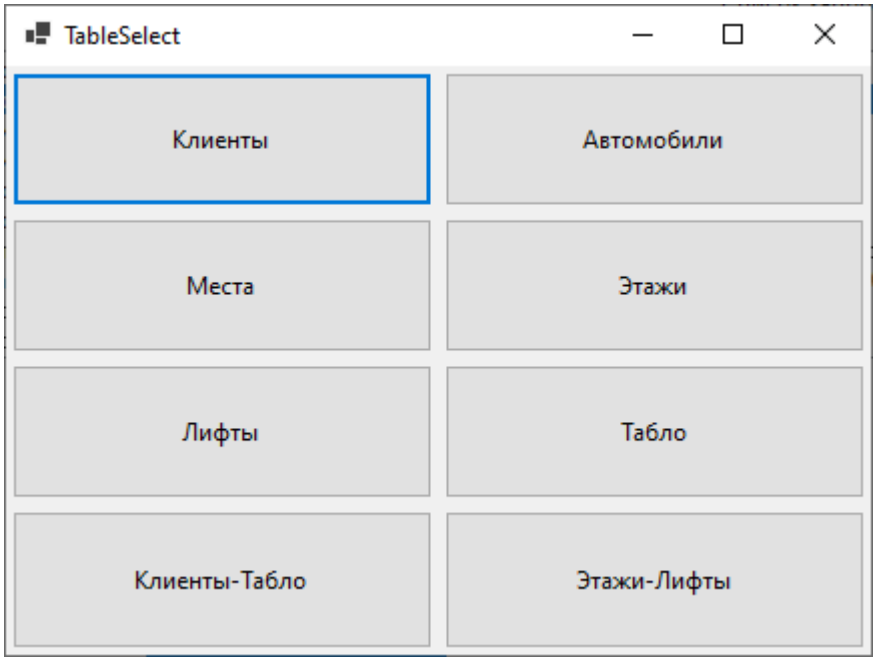

Рисунок 1.4 – Выбор таблицы для просмотра

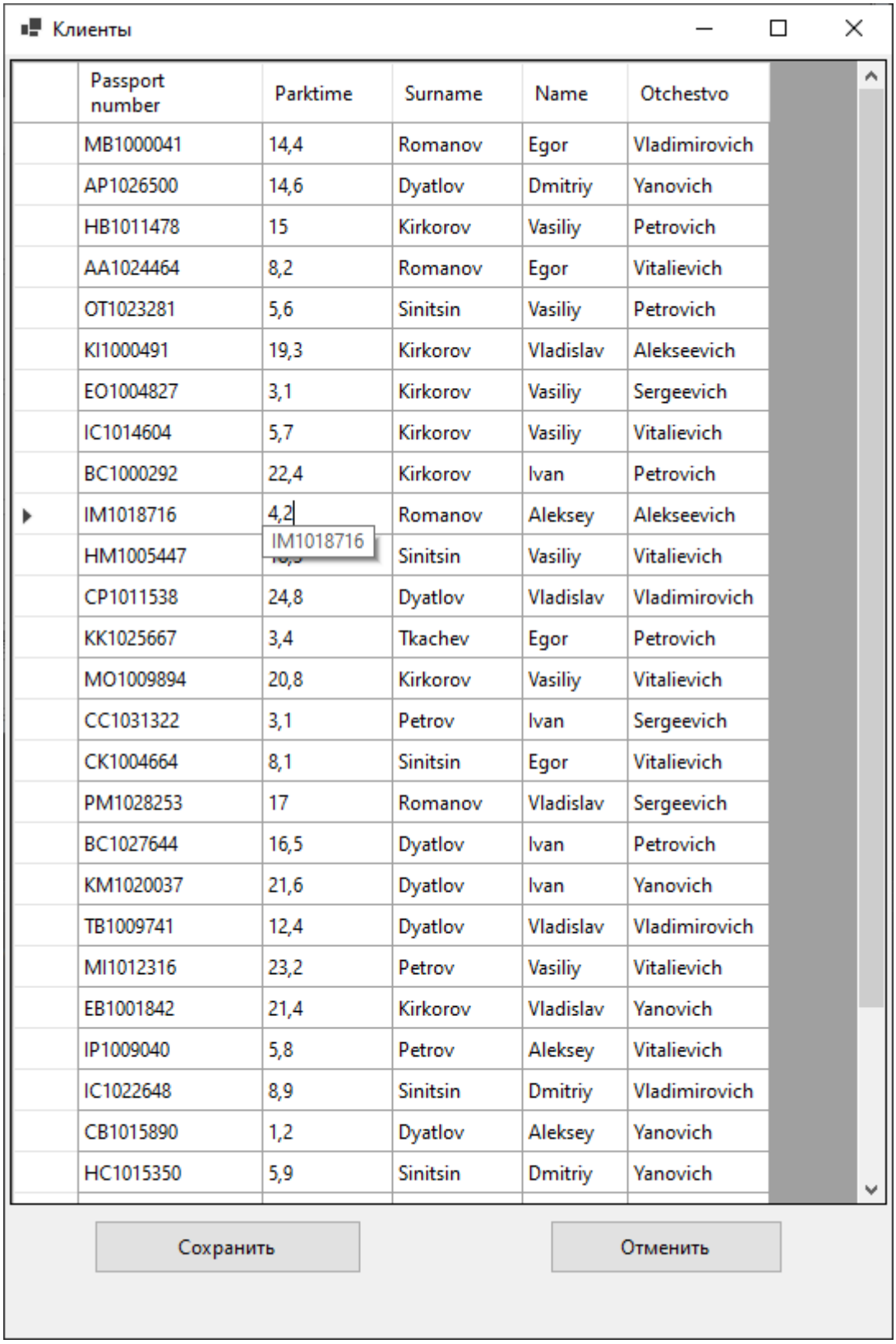

Рисунок 1.5 – окно для работы с таблицей

1.4 Извлечение данных из базы и запись изменений

Для получения данных из базы использовался класс NpgsqlDataAdapter с последующей их записью в DataSet для передачи в графическую форму.

```
public ResultTableEditable(NpgsqlDataAdapter adapter)
{
    InitializeComponent();
    ds = new DataSet();
    this.adapter = adapter;
    this.adapter.Fill(ds);
    dataGridView1.DataSource = ds.Tables[0];
}
```

Рисунок 1.6 – извлечение данных из бд

Для записи изменений в базу данных использовался метод адаптера Update и автоматический конструктор CRUD-операций.

```
ссылка: 1
private void SaveButton_Click(object sender, EventArgs e)
{
    NpgsqlCommandBuilder commandBuilder = new(adapter);
    adapter.Update(ds);
}
```

Рисунок 1.7 – сохранение внесенных изменений

В адаптер данные записываются на основе информации о подключении к бд и sql запросе.

```
public static void ExecuteQuery(string sqlCommand, bool isEditable = false, string tableName = "")
{
    NpgsqlDataAdapter adapter = new(sqlCommand, conn);
    if (isEditable)
    {
        var resultForm = new ResultTableEditable(adapter);
        resultForm.Text = tableName;
        resultForm.Show();
    }
    else
    {
        var resultForm = new ResultTable(adapter);
        resultForm.Show();
    }
}
```

Рисунок 1.8 – создание адаптера

Собственно сами запросы представлены на рисунке 1.9.

```
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    var listBox = (ListBox)sender;
    switch (listBox.SelectedIndex)
    {
        case 0:
            {
                var sql = "select \"Passport number\", \"Parktime\", \"Auto number\" from \"Client\" " +
                    "inner join \"Auto\" on \"Auto\".\"Client number\" = \"Client\".\"Passport number\"";
                textBox1.Text = sql;
                //ExecuteQuery(sql);
                break;
            }
        case 1:
            {
                var sql = "select \"Client number\", \"Auto\".\"Auto number\", \"Place number\" from \"Auto\" " +
                    "inner join \"Place\" on \"Place\".\"Auto number\" = \"Auto\".\"Auto number\"";
                textBox1.Text = sql;
                //ExecuteQuery(sql);
                break;
            }
        case 2:
            {
                var sql = "select \"Elevator number\", \"lift capacity\", \"Auto number\", \"Mass\" from \"Elevator\" " +
                    "left outer join \"Auto\" on \"Auto\".\"Mass\" <= \"Elevator\".\"lift capacity\" where \"Mass\" > 7200";
                textBox1.Text = sql;
                //ExecuteQuery(sql);
                break;
            }
        case 3:
            {
                var sql = "select \"Elevator number\", \"lift capacity\", \"Auto number\", \"Mass\" from \"Elevator\" " +
                    "right outer join \"Auto\" on \"Auto\".\"Mass\" <= \"Elevator\".\"lift capacity\" where \"Mass\" > 5000";
                textBox1.Text = sql;
                //ExecuteQuery(sql);
                break;
            }
```

Рисунок 1.9 – sql-запросы

ЗАКЛЮЧЕНИЕ

В ходе данной лабораторной работы было создано простое серверное приложение на базе .NET Core 5. Был реализован функционал SQL-запросов из лабораторных работ 4 и 5. Создано графическое приложение для работы с таблицами базы данных.

ПРИЛОЖЕНИЕ А

Листинг кода

```
namespace WinFormsApp1
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
conn.Open();
}
public static ApplicationContext db = new ApplicationContext();
public static NpgsqlConnection conn = new
NpgsqlConnection("Server=127.0.0.1;User
Id=postgres;Password=1;Port=5432;Database=Parking;");
public static void ExecuteQuery(string sqlCommand, bool
isEditable = false, string tableName = "")
{
NpgsqlDataAdapter adapter = new(sqlCommand, conn);
if (isEditable)
{
var resultForm = new ResultTableEditable(adapter);
resultForm.Text = tableName;
resultForm.Show();
}
else
{
var resultForm = new ResultTable(adapter);
resultForm.Show();
}
}
private void listBox1_SelectedIndexChanged(object sender,
EventArgs e)
{
var listBox = (ListBox)sender;
switch (listBox.SelectedIndex)
{
case 0:
{
var sql = "select \"Passport number\", \"Parktime\", \"Auto
number\" from \"Client\" " +
"inner join \"Auto\" on \"Auto\".\"Client number\" =
\"Client\".\"Passport number\"";
textBox1.Text = sql;
break;
}
case 1:
{
var sql = "select \"Client number\", \"Auto\".\"Auto number\",
\"Place number\" from \"Auto\" " +
```

```
"inner join \"Place\" on \"Place\".\"Auto number\" =
\"Auto\".\"Auto number\"";
textBox1.Text = sql;
break;
}
case 2:
{
var sql = "select \"Elevator number\", \"lift capacity\", \"Auto
number\", \"Mass\" from \"Elevator\" " +
"left outer join \"Auto\" on \"Auto\".\"Mass\" <=
\"Elevator\".\"lift capacity\" where \"Mass\" > 7200";
textBox1.Text = sql;
break;
}
case 3:
{
var sql = "select \"Elevator number\", \"lift capacity\", \"Auto
number\", \"Mass\" from \"Elevator\" " +
"right outer join \"Auto\" on \"Auto\".\"Mass\" <=
\"Elevator\".\"lift capacity\" where \"Mass\" > 5000";
textBox1.Text = sql;
break;
}
case 4:
{
var sql = "select \"Table\".\"Floor number\",
\"Floor\".\"Height\", \"Place type\", \"Place numbers\" from
\"Floor\"" +
" right outer join \"Table\" on \"Floor\".\"Floor number\" =
\"Table\".\"Floor number\"";
textBox1.Text = sql;
break;
}
case 5:
{
var sql = "select \"Table\".\"Floor number\",
\"Floor\".\"Height\", \"Place type\", \"Place numbers\" from
\"Floor\"" +
" left outer join \"Table\" on \"Floor\".\"Floor number\" =
\"Table\".\"Floor number\"";
textBox1.Text = sql;
break;
}
case 6:
{
var sql = "select * from \"Client\" cross join \"Place\" where
\"Parktime\" > 22 and \"Surname\" = 'Kirkorov'";
textBox1.Text = sql;
break;
}
case 7:
{
```

```
var sql = "select * from \"Table\" cross join \"Client\" where
\"Floor number\" = 2 and \"Parktime\" > 10";
textBox1.Text = sql;
break;
}
case 8:
{
var sql = "select * from \"Auto\" full outer join \"Place\" on
\"Auto\".\"Auto number\" = \"Place\".\"Auto number\" order by
\"Auto\".\"Auto number\" desc";
textBox1.Text = sql;
break;
}
case 9:
{
var sql = "select * from \"Client\" full outer join \"Auto\" on
\"Passport number\" = \"Client number\"";
textBox1.Text = sql;
break;
}
case 10:
{
var sql = "select \"Client\".\"Surname\", \"Name\",
\"Parktime\", \"Auto\".\"Auto number\" from \"Client\" " +
"join \"Auto\" on \"Passport number\" = \"Client number\" " +
"join \"Place\" on \"Auto\".\"Auto number\"=\"Place\".\"Auto
number\" " +
"group by \"Name\", \"Surname\", \"Parktime\", \"Auto\".\"Auto
number\" " +
"having \"Parktime\" > (select avg(\"Parktime\") from
\"Client\") " +
"order by \"Auto\".\"Auto number\"";
textBox1.Text = sql;
break;
}
case 11:
{
var sql = "select sum(\"Mass\") as \"Summary auto mass\",
min(\"Parktime\") as \"Minimum parktime\",max(\"Table\".\"Table
number\") as \"Maximum Table number\"," +
"(select count(*) from \"Client\" where \"Surname\" =
'Kirkorov') as \"Kirkorov count\"from \"Client\" " +
"join \"Auto\" on \"Passport number\" = \"Auto\".\"Client
number\" " +
"join \"Client_Table\" on \"Client_Table\".\"Client number\" =
\"Passport number\" " +
"join \"Table\" on \"Client_Table\".\"Table number\" =
\"Table\".\"Table number\"";
textBox1.Text = sql;
break;
}
case 12:
```

```
{
var sql = "with ClientAuto as (" +
"select \"Passport number\" from \"Client\"  " +
"union select \"Auto\".\"Client number\" from \"Auto\"), " +
"AutoPlace as (" +
"select \"Auto\".\"Auto number\" from \"Auto\"    " +
"union select \"Place\".\"Auto number\" from \"Place\")" +
"select \"Surname\", \"Passport number\", \"Auto number\" from
\"Client\" " +
"join \"Auto\" on \"Passport number\" = \"Client number\" " +
"where \"Passport number\" in (select * from ClientAuto) and
\"Auto number\" in (select * from AutoPlace)";
textBox1.Text = sql;
break;
}
case 13:
{
var sql = "with ClientAuto as (" +
"select \"Passport number\" from \"Client\"  " +
"union all select \"Auto\".\"Client number\" from \"Auto\"), " +
"AutoPlace as (     " +
"select \"Auto\".\"Auto number\" from \"Auto\"    " +
"union all select \"Place\".\"Auto number\" from \"Place\")" +
"select \"Surname\", \"Passport number\", \"Auto number\" from
\"Client\" " +
"join \"Auto\" on \"Passport number\" = \"Client number\" " +
"where \"Passport number\" not in (select * from ClientAuto
where \"Passport number\" = 'AA1024464') and \"Auto number\" in
(select * from AutoPlace);";
textBox1.Text = sql;
break;
}
case 14:
{
var sql = "with ClientAuto as (" +
"select \"Passport number\" from \"Client\"  " +
"except select \"Auto\".\"Client number\" from \"Auto\"), " +
"AutoPlace as (     " +
"select \"Auto\".\"Auto number\" from \"Auto\"    " +
"intersect select \"Place\".\"Auto number\" from
\"Place\")select \"Surname\", \"Passport number\", \"Auto
number\" from \"Client\" " +
"join \"Auto\" on \"Passport number\" = \"Client number\" " +
"where \"Passport number\" not in (select * from ClientAuto) and
\"Auto number\" in (select * from AutoPlace);";
textBox1.Text = sql;
break;
}
case 15:
{
var sql = "select \"Surname\", \"Name\", \"Parktime\",
\"Auto\".\"Auto number\"from \"Client\" join \"Auto\" on
```

```
\"Passport number\" = \"Client number\"join \"Place\" on
\"Auto\".\"Auto number\"=\"Place\".\"Auto number\"where exists
(select * from \"Client\" where \"Surname\" = 'Kirkorov')order
by \"Surname\" asc;";
textBox1.Text = sql;
break;
}
case 16:
{
var sql = "select \"Surname\", \"Name\", \"Parktime\",
\"Auto\".\"Auto number\"from \"Client\" join \"Auto\" on
\"Passport number\" = \"Client number\"join \"Place\" on
\"Auto\".\"Auto number\"=\"Place\".\"Auto number\"where
\"Parktime\" > any(select \"Parktime\" from \"Client\" where
\"Surname\" = 'Sinitsin')order by \"Auto\".\"Auto number\";";
textBox1.Text = sql;
break;
}
case 17:
{
var sql = "select \"Surname\", \"Name\", \"Parktime\",
\"Auto\".\"Auto number\", \"Mass\" from \"Client\" join \"Auto\"
on \"Passport number\" = \"Client number\" join \"Place\" on
\"Auto\".\"Auto number\"=\"Place\".\"Auto number\" group by
\"Surname\", \"Name\", \"Mass\", \"Parktime\", \"Auto\".\"Auto
number\" having \"Parktime\" > (select avg(\"Parktime\") from
\"Client\")  and \"Mass\" between min(\"Mass\") and
max(\"Mass\") order by \"Auto\".\"Auto number\";";
textBox1.Text = sql;
break;
}
case 18:
{
var sql = "select \"Client\".\"Surname\", \"Name\",
\"Parktime\", \"Auto\".\"Auto number\" from \"Client\" join
\"Auto\" on \"Passport number\" = \"Client number\" join
\"Place\" on \"Auto\".\"Auto number\"=\"Place\".\"Auto number\"
where \"Name\" <> 'Vasiliy' order by \"Name\" asc;";
textBox1.Text = sql;
break;
}
case 19:
{
var sql = "select \"Surname\", \"Passport number\", \"Auto
number\" from \"Client\" join \"Auto\" on \"Passport number\" =
\"Client number\" where \"Passport number\" like '%B%' order by
\"Passport number\";";
textBox1.Text = sql;
break;
}
case 20:
{
```

```
var sql = "select \"Passport number\", \"Floor\".\"Floor
number\", \"Place type\", \"Height\" from \"Client\" join
\"Client_Table\" on \"Client number\" = \"Passport number\" join
\"Table\" on \"Table\".\"Table number\" =
\"Client_Table\".\"Table number\" join \"Floor\" on
\"Table\".\"Floor number\" = \"Floor\".\"Floor number\";";
textBox1.Text = sql;
break;
}
case 21:
{
var sql = "select avg(\"Mass\") as \"Average auto mass\",
avg(\"Parktime\") as \"Average parktime\", (select count(*) from
\"Client\" where \"Surname\" = 'Kirkorov') as \"Kirkorov count\"
from \"Client\" join \"Auto\" on \"Passport number\" =
\"Auto\".\"Client number\" join \"Client_Table\" on
\"Client_Table\".\"Client number\" = \"Passport number\";";
textBox1.Text = sql;
break;
}
case 22:
{
var sql = "select * from \"Client\" join \"Client_Table\" on
\"Client number\" = \"Passport number\" join \"Table\" on
\"Table\".\"Table number\" = \"Client_Table\".\"Table number\"
join \"Floor\" on \"Table\".\"Floor number\" = \"Floor\".\"Floor
number\";";
textBox1.Text = sql;
break;
}
case 23:
{
var sql = "select \"Surname\", \"Passport number\",
\"Auto\".\"Auto number\", \"Place number\" from \"Client\" join
\"Auto\" on \"Passport number\" = \"Client number\" join
\"Place\" on \"Auto\".\"Auto number\" = \"Place\".\"Auto
number\" where \"Passport number\" not in (select \"Passport
number\" from \"Client\" where \"Passport number\" like
'%4%');";
textBox1.Text = sql;
break;
}
case 24:
{
var sql = "select \"Surname\", \"Passport number\",
\"Auto\".\"Auto number\", \"Place number\" from \"Client\" join
\"Auto\" on \"Passport number\" = \"Client number\" join
\"Place\" on \"Auto\".\"Auto number\" = \"Place\".\"Auto
number\" where \"Passport number\" in (select \"Passport
number\" from \"Client\" where \"Passport number\" like '%13%')
order by \"Surname\";";
textBox1.Text = sql;
```

```
break;
}
default: break;
}
}
private void ExecuteButton_Click(object sender, EventArgs e)
{
ExecuteQuery(textBox1.Text);
}
private void ShowTablesButton_Click(object sender, EventArgs e)
{
TableSelect form = new();
form.Show();
}
}
}

namespace WinFormsApp1
{
public partial class ResultTableEditable : Form
{
private DataSet ds;
private NpgsqlDataAdapter adapter;
public ResultTableEditable(NpgsqlDataAdapter adapter)
{
InitializeComponent();
ds = new DataSet();
this.adapter = adapter;
this.adapter.Fill(ds);
dataGridView1.DataSource = ds.Tables[0];
}
private void CanselButton_Click(object sender, EventArgs e)
{
this.Close();
}
private void SaveButton_Click(object sender, EventArgs e)
{
NpgsqlCommandBuilder commandBuilder = new(adapter);
adapter.Update(ds);
}
}
}
```