

ГУАП

ФАКУЛЬТЕТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

преподаватель

И.А. Юрьева

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

ОТЧЕТЫ О ЛАБОРАТОРНЫХ РАБОТАХ

По дисциплине: МДК.04.01 Технология разработки и защиты баз данных

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

021к

В.Д. Панков Вася
Панков

подпись, дата

инициалы, фамилия

Санкт-Петербург 2023

Содержание

1 Лабораторная работа № 1 «Взаимодействие с базами данных в MS Visual Studio»	2
1.1 Лабораторная работа № 1 «Взаимодействие с базами данных в MS Visual Studio»	2
2 Лабораторная работа № 2	29
2.1 Часть 1	29
2.2 Часть 2 (Индивидуальное задание)	41
3 Лабораторная работа № 3	87
3.1 Часть 1	87
3.2 Часть 2	93
3.3 Дополнительное задание	102

1 Лабораторная работа № 1 «Взаимодействие с базами данных в MS Visual Studio»

1.1 Лабораторная работа № 1 «Взаимодействие с базами данных в MS Visual Studio»

Цель работы : изучить вспомогательные классы для взаимодействия с базой данных через клиентское приложение(использование технологии ADO.NET).

Задание 1. Используя вспомогательные классы, предоставляемые поставщиком OLE DB (англ. Object Linking and Embedding), установить соединение с БД Колледж и реализовать выполнение запросов, перечисленных ниже.

Запрос 1. Вывести в DataGrid все записи из таблицы Students.

Запрос 2. Для студента с фамилией, заданной в текстовое поле, вывести поля Фамилия(Familia), №группы(№группы) и Бюджет(Budget), т. е. реализовать запрос на выборку, в котором фамилия будет вводится как параметр. Запрос выполняется вызовом обработчика события нажатия на соответствующую кнопку.

Запрос 3. В предыдущем запросе добавить в код конструкцию try..catch и предусмотреть случай, когда нет соединения с БД.

Запрос 4. Вычислить возраст студентов-юношей.

Запрос 5. Для студентов из группы, введенной в дополнительный TextBox, вывести Фамилию и инициалы, разделенные точками.

Запрос 6. Для студентов с заданным годом рождения(дополнительный TextBox), вывести фамилию и №группы

Запрос 7. Вывести фамилии и №группы иногородних студентов.

Примечание1. В предложенной БД Колледж поле Город(Gorod) заполняется только для иногородних студентов.

Запрос 8. Для создания новых записей в таблице программно, необходимо использовать код из следующего примера с методом ExecuteNonQuery(). В данном примере в таблицу Students вставляется новая запись с номером студенческого билета равным 1004, отображаются все записи вместе с добавленной

Задание 2.

В клиентском приложении, созданном в первой части лабораторной работы, необходимо реализовать следующие функции:

1. Вход в систему должен осуществляться через аутентификацию пользователей (аутентификация – это проверка подлинности пользователя путём сравнения введённого им пароля с паролем, сохранённым в базе данных пользователей). Причем аутентификация пользователей, которые были зарегистрированы в системе, реализуется посредством существующего логина и пароля. На форме входа в систему предусмотреть возможность регистрации нового пользователя. Добавить дополнительную таблицу в БД. При создании новой таблицы учесть следующее: в приложении могут быть пользователи с двумя ролями — студент и администратор(сотрудник учебной части). Студент может только просматривать таблицы из БД, а администратор может изменять данные в таблицах через клиентское приложение.
2. Показать расширенные функциональные возможности администратора(сформулировать запрос для нахождения статистической информации за период на отделении, показать возможность изменения данных в таблице(таблицах)).
3. Реализовать импорт данных из форм в MS Excel(использовать Microsoft.Office.Interop.Excel)
4. Действия пользователя, связанные с информационной безопасностью, необходимо запротоколировать (для каждого пользователя, вошедшего в систему, необходимо сохранять информацию о входе и выходе из системы в текстовом файле в следующем формате: логин пользователя, дата /время входа и выхода из системы).

Код:

Разметка базового окна:

```
<Window x:Class="DataBase.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Title="MainWindow" Height="450" Width="800">
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="10*" />
        <RowDefinition Height="1*" />
        <RowDefinition Height="1*" />
        <RowDefinition Height="1*" />
    </Grid.RowDefinitions>
    <DataGrid CellEditEnding="Data_OnCellEditEnding" Name="data" />
    <StackPanel Grid.Row="1" VerticalAlignment="Center"
        < HorizontalAlignment="Stretch" Orientation="Horizontal">
        <TextBlock>Фамилия:</TextBlock>
        <TextBox Name="Familia" Text="" Width="300" />
        <Button Click="Search"> Поиск</Button>
        <CheckBox Name="YoungMen" Checked="ToggleButton_OnChecked"
            < Unchecked="ToggleButton_OnUnchecked"
                Content="Юноши с возрастом" Margin="50, 0, 0, 0" />
    </StackPanel>
    <StackPanel Grid.Row="2" VerticalAlignment="Center"
        < HorizontalAlignment="Stretch" Orientation="Horizontal">
        <CheckBox Checked="GroupSearch_OnChecked"
            < Unchecked="GroupSearch_OnUnchecked" Content="Поиск по группе"
                Name="GroupSearch" />
        <TextBlock Name="GroupText" Visibility="Collapsed" Margin="20, 0, 0,
            < 0" Text="Номер группы:" />
        <TextBox Visibility="Collapsed" Name="Group" Width="300" />
        <Button Visibility="Collapsed" Name="GroupButton"
            < Click="SearchByGroup"> Поиск</Button>
        <CheckBox Margin="10, 0, 0, 0" Name="OutOfTown"
            < Checked="OutOfTown_OnChecked"
                Unchecked="OutOfTown_OnUnchecked" Content="Иногородние
                    < студенты" />
        <Button Click="Reform1" Name="Reform1Btn" Margin="10, 0, 0, 0"
            < Content="Образовательная реформа № 1" />
    </StackPanel>
```

```

<StackPanel Grid.Row="3" VerticalAlignment="Center"
    ↳ HorizontalAlignment="Stretch" Orientation="Horizontal">
    <CheckBox Unchecked="YearSearch_OnUnchecked"
        ↳ Checked="YearSearch_OnChecked"
            Content="Поиск по году рождения" Name="YearSearch" />
    <TextBlock Name="YearText" Visibility="Collapsed" Margin="20, 0, 0,
        ↳ 0" Text="Год рождения:" />
    <TextBox Visibility="Collapsed" Name="Year" Width="300" />
    <Button Visibility="Collapsed" Click="YearButton_OnClick"
        ↳ Name="YearButton"> Поиск</Button>
    <Button Click="Reform2" Name="Reform2Btn" Margin="10, 0, 0, 0"
        ↳ Content="Образовательная реформа № 2" />
    <Button Click="Exit" Name="ExitBtn" IsCancel="True" Margin="10, 0, 0,
        ↳ 0" Content="Войти в другой аккаунт" />
    <Button Click="Import" Name="ExcelImportBtn" IsCancel="True"
        ↳ Margin="10, 0, 0, 0" Content="Import to Excel" />

</StackPanel>
</Grid>
</Window>

```

Разметка окна авторизации:

```

<Window x:Class="DataBase.AuthWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expressionblend/2008"
    mc:Ignorable="d"
    Title="AuthWindow" Height="600" Width="800" MinHeight="600"
    ↳ MinWidth="400">

<Grid>
    <Grid.RowDefinitions>
        <RowDefinition />
        <RowDefinition />
        <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <TextBlock Margin="10" Foreground="Chocolate" FontSize="18"
        ↳ TextAlignment="Center" Name="ErrorMessage">

```

```

        Grid.Column="1" VerticalAlignment="Bottom" />
<StackPanel HorizontalAlignment="Stretch" VerticalAlignment="Center">
    <Grid.Row="1" Grid.Column="1">
        <DockPanel Margin="10" HorizontalAlignment="Stretch">
            <TextBlock>Логин</TextBlock>
            <TextBox Name="Login" Margin="28,0,10,0">
                <!-- HorizontalAlignment="Stretch" />
            </TextBox>
        </DockPanel>
        <DockPanel Margin="10" HorizontalAlignment="Stretch">
            <TextBlock>Пароль</TextBlock>
            <PasswordBox Name="Password" Margin="20,0,10,0">
                <!-- HorizontalAlignment="Stretch" />
            </PasswordBox>
        </DockPanel>
        <DockPanel Name="SecPasswordPanel" Visibility="Collapsed" Margin="10">
            <TextBlock>Пароль 2</TextBlock>
            <PasswordBox Name="SecondPassword" Margin="10,0,10,0">
                <!-- HorizontalAlignment="Stretch" />
            </PasswordBox>
        </DockPanel>
        <Button Name="AuthBtn" Click="Auth" Margin="10" Content="Вход" />
        <Button Click="ToReg" Name="SwitchBtn" Margin="10" Content="Перейти к
            <!-- регистрации" />
    </Grid>
</StackPanel>
</Grid>
</Window>
```

Код логики окна авторизации:

```

using System;
using System.Collections.Generic;
using System.Data.OleDb;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Windows;

namespace DataBase;

public partial class AuthWindow : Window
{
    private Mode _mode = Mode.Auth;

    private Dictionary<Mode, string> modeToQuery = new()
```

```

{
    {Mode.Auth, "SELECT user_id, role FROM users WHERE login = @login AND
→   password = @pass"},

    {Mode.Reg, "INSERT INTO users ([login], [password]) VALUES (@login,
→   @pass)"}
};

public AuthWindow()
{
    InitializeComponent();
}

private void Auth(object sender, RoutedEventArgs e)
{
    try
    {
        var connection =
            new OleDbConnection(
                @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\user\

→ \Desktop\labs\04.01\db_lab1 DataBase\db\college.mdb;Persist Security
→ Info=False");

        connection.Open();
        using (var mySha256 = SHA256.Create())
        {
            if (Login.Text.Trim() == "")
            {
                ErrorMessage.Text = "Логин не может быть пустым";
                return;
            }

            if (Password.Password.Trim() == "")
            {
                ErrorMessage.Text = "Пароль не должен быть пустым";
                return;
            }

            if (_mode == Mode.Reg)
            {
                if (Password.Password != SecondPassword.Password)
                {
                    ErrorMessage.Text = "Пароли не совпадают";
                }
            }
        }
    }
}

```

```

        return;
    }

    var checkThisLoginCommand =
        new OleDbCommand("SELECT user_id FROM users WHERE login =
← @login", connection);
    checkThisLoginCommand.Parameters.Add("@login",
← OleDbType.VarChar, 80).Value = Login.Text;
    if (checkThisLoginCommand.ExecuteScalar() != null)
    {
        ErrorMessage.Text = "Такой логин уже существует";
        return;
    }
}

var passwordHash = string.Join("", mySha256.ComputeHash(new
← UTF8Encoding().GetBytes(Password.Password)).Select(b => $"{b:X}")
    .ToArray()).ToLowerInvariant();
var command = new OleDbCommand(modeToQuery[_mode], connection);
command.Parameters.Add("@login", OleDbType.VarChar, 80).Value =
← Login.Text;
command.Parameters.Add("@pass", OleDbType.VarChar, 80).Value =
← passwordHash;
if (_mode == Mode.Reg)
{
    command.ExecuteNonQuery();
    ToReg(null, null);
    return;
}

var user = command.ExecuteReader();
if (!user.HasRows)
{
    ErrorMessage.Text = "Неверный логин или пароль";
}
else
{
    user.Read();
    var userId = user.GetInt32(0);
    var roleId = user.GetInt32(1);

    var window = new MainWindow(userId, roleId);

```

```

        window.Show();
        Close();
    }
}

catch (Exception exception)
{
    MessageBox.Show("База данных не найдена или что-то пошло не так.");
    Close();
}
}

private void ToReg(object? sender, RoutedEventArgs? e)
{
    ErrorMessage.Text = "";
    switch (_mode)
    {
        case Mode.Auth:
            _mode = Mode.Reg;
            SecPasswordPanel.Visibility = Visibility.Visible;
            SwitchBtn.Content = "Перейти к авторизации";
            AuthBtn.Content = "Зарегистрироваться";
            break;
        case Mode.Reg:
            _mode = Mode.Auth;
            SwitchBtn.Content = "Перейти к регистрации";
            AuthBtn.Content = "Вход";
            SecPasswordPanel.Visibility = Visibility.Collapsed;
            break;
    }
}
}

```

Код логики базового окна:

```

using System;
using System.Windows;
using System.Data;
using System.Data.OleDb;
using System.IO;
using System.Linq;
using System.Windows.Controls;
using DataTable = System.Data.DataTable;
using Window = System.Windows.Window;

```

```

using DocumentFormat.OpenXml;
using DocumentFormat.OpenXml.Packaging;
using DocumentFormat.OpenXml.Spreadsheet;
using Microsoft.Win32;

namespace DataBase;

/// <summary>
/// Interaction logic for MainWindow.xaml
/// </summary>
public partial class MainWindow : Window
{
    private const string QueryGetMalesWithAge =
        "SELECT *, DateDiff(\"yyyy\", Datarogr, Date()) AS [Возраст] FROM
        Students WHERE Students.Pol = 'M' AND Familia LIKE ?";

    private const string BaseQuery = "SELECT * FROM Students WHERE Familia LIKE
        ?";

    private const string GroupQuery =
        "SELECT Familia + \" \" + Left([Imya], 1) + \".\" + Left([Otchestvo], 1)
        + \".\" AS [Фамилия и инициалы] FROM Students WHERE Familia LIKE ? AND Pgr =
        @group";

    private const string YearQuery =
        "SELECT Familia, [Pgr] FROM Students WHERE Year([Datarogr]) = @year AND
        Familia LIKE ?";

    private const string OutOfTownQuery =
        "SELECT Familia, [Pgr] FROM Students WHERE Gorod <> \"\" AND Familia LIKE
        ?";

    private string _queryNow;

    private const string ConnectionString =
        @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\user\Desktop\lab_1\
        s\04.01\db_lab1 DataBase\db\college.mdb;Persist Security
        Info=False";
}

private int _year;
private OleDbConnection connection = new(ConnectionString);
private Role _role;

```

```

private DataTable _dataTable;
private OleDbDataAdapter _adapter;
private int _userId;

public MainWindow()
{
    InitializeComponent();
    ReloadTablebyCommand(Query: BaseQuery);
}

public MainWindow(int userId, int role) : this()
{
    _role = Constants.RoleByInt[role];
    _userId = userId;
    if (_role == Role.User)
    {
        data.IsReadOnly = true;
        Reform1Btn.IsEnabled = false;
        Reform2Btn.IsEnabled = false;
        ExcelImportBtn.IsEnabled = false;
    }

    using (StreamWriter writer = new("log.txt", true))
    {
        writer.WriteLineAsync($"{userId} logged in in {DateTime.Now}");
    }
}

private void OpenConnection()
{
    try
    {
        connection.Open();
    }
    catch (Exception _)
    {
        MessageBox.Show("База данных не найдена или что-то пошло не так.");
        Close();
    }
}

private void ReloadTable(OleDbCommand command)
{

```

```

        OpenConnection();
        _dataTable = new DataTable();
        _adapter = new OleDbDataAdapter(command);
        connection.Close();
        _adapter.Fill(_dataTable);
        data.ItemsSource = _dataTable.DefaultView;
    }

public void ReloadTablebyCommand(string? Query = null, string Familia = "%")
{
    _queryNow = Query ?? _queryNow;
    var command = new OleDbCommand(Query ?? _queryNow, connection);
    command.Parameters.AddWithValue("?", OleDbType.VarChar, 80).Value =
    $"#{Familia.ToLowerInvariant()}%";

    ReloadTable(command);
}

public void ReloadTablebyCommand(string groupName, string? Query = null,
                                string Familia = "%")
{
    _queryNow = Query ?? _queryNow;
    var command = new OleDbCommand(_queryNow, connection);
    command.Parameters.AddWithValue("?", OleDbType.VarChar, 80).Value =
    $"#{Familia.ToLowerInvariant()}%";

    command.Parameters.AddWithValue("@group", OleDbType.VarChar, 6).Value = groupName;
    ReloadTable(command);
}

public void ReloadTablebyCommand(int year, string? Query = null, string
                                Familia = "%")
{
    _queryNow = Query ?? _queryNow;
    var command = new OleDbCommand(_queryNow, connection);
    command.Parameters.AddWithValue("@year", year);
    command.Parameters.AddWithValue("?", OleDbType.VarChar, 80).Value =
    $"#{Familia.ToLowerInvariant()}%";

    ReloadTable(command);
}

private void Search(object sender, RoutedEventArgs e)

```

```

{
    if (Group.Visibility == Visibility.Visible)
        ReloadTablebyCommand(groupName: Group.Text, Familia:
    ↵ Familia.Text.Trim() == "" ? "%" : Familia.Text);
    else if (Year.Visibility == Visibility.Visible && int.TryParse(Year.Text,
    ↵ out _year))
        ReloadTablebyCommand(_year, Familia: Familia.Text.Trim() == "" ? "%" :
    ↵ : Familia.Text);
    else
        ReloadTablebyCommand(Familia: Familia.Text.Trim() == "" ? "%" :
    ↵ Familia.Text);
}

private void ToggleButton_OnChecked(object sender, RoutedEventArgs e)
{
    ReloadTablebyCommand(QueryGetMalesWithAge, Familia.Text.Trim() == "" ?
    ↵ "%" : Familia.Text);
}

private void ToggleButton_OnUnchecked(object sender, RoutedEventArgs e)
{
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    ↵ Familia.Text);
}

private void SearchByGroup(object sender, RoutedEventArgs e)
{
    ReloadTablebyCommand(Group.Text, GroupQuery, Familia.Text.Trim() == "" ?
    ↵ "%" : Familia.Text);
}

private void GroupSearch_OnChecked(object sender, RoutedEventArgs e)
{
    GroupText.Visibility = Visibility.Visible;
    Group.Visibility = Visibility.Visible;
    GroupButton.Visibility = Visibility.Visible;
    HideCheckboxes();
    ReloadTablebyCommand(Group.Text, GroupQuery, Familia.Text.Trim() == "" ?
    ↵ "%" : Familia.Text);
}

private void GroupSearch_OnUnchecked(object sender, RoutedEventArgs e)

```

```

{
    GroupText.Visibility = Visibility.Collapsed;
    Group.Visibility = Visibility.Collapsed;
    GroupButton.Visibility = Visibility.Collapsed;
    ShowCheckboxes();
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    Familia.Text);
}

private void HideCheckboxes()
{
    YoungMen.IsChecked = false;
    YoungMen.Visibility = Visibility.Collapsed;
    OutOfTown.IsChecked = false;
    OutOfTown.Visibility = Visibility.Collapsed;
}

private void ShowCheckboxes()
{
    YoungMen.Visibility = Visibility.Visible;
    OutOfTown.Visibility = Visibility.Visible;
}

private void YearSearch_OnChecked(object sender, RoutedEventArgs e)
{
    HideCheckboxes();
    GroupSearch.IsChecked = false;
    GroupSearch.IsEnabled = false;
    YearText.Visibility = Visibility.Visible;
    Year.Visibility = Visibility.Visible;
    YearButton.Visibility = Visibility.Visible;
    ReloadTablebyCommand(0, YearQuery, Familia.Text.Trim() == "" ? "%" :
    Familia.Text);
}

private void YearButton_OnClick(object sender, RoutedEventArgs e)
{
    if (int.TryParse(Year.Text, out _year))
        ReloadTablebyCommand(_year, Familia: Familia.Text.Trim() == "" ? "%" :
    : Familia.Text);
}

private void YearSearch_OnUnchecked(object sender, RoutedEventArgs e)

```

```

{
    ShowCheckboxes();
    GroupSearch.IsEnabled = true;
    YearText.Visibility = Visibility.Collapsed;
    Year.Visibility = Visibility.Collapsed;
    YearButton.Visibility = Visibility.Collapsed;
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    → Familia.Text);
}

private void OutOfTown_OnChecked(object sender, RoutedEventArgs e)
{
    ReloadTablebyCommand(OutOfTownQuery, Familia.Text.Trim() == "" ? "%" :
    → Familia.Text);
}

private void OutOfTown_OnUnchecked(object sender, RoutedEventArgs e)
{
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    → Familia.Text);
}

private void Reform1(object sender, RoutedEventArgs e)
{
    OpenConnection();
    new OleDbCommand("UPDATE Students SET Gorod = '\u0413. \u041a\u043b\u043f\u043d\u0438\u043d\u0430', Budget = 1
    → WHERE POL = '\u041c' AND Budget = 0",
        connection).ExecuteNonQuery();
    connection.Close();
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    → Familia.Text);
}

private void Reform2(object sender, RoutedEventArgs e)
{
    OpenConnection();
    new OleDbCommand("DELETE FROM Students WHERE Budget = 0",
    → connection).ExecuteNonQuery();
    connection.Close();
    ReloadTablebyCommand(BaseQuery, Familia.Text.Trim() == "" ? "%" :
    → Familia.Text);
}

```

```

    private void Data_OnCellEditEnding(object? sender,
→     DataGridViewCellEventArgs e)
{
    // MessageBox.Show($"{e.Column.Header} = {(e.EditingElement)}");
    // new OleDbCommand($"UPDATE Students SET {e.Column.Header} =
→ {e.EditingElement} ")
}

private void Exit(object sender, RoutedEventArgs e)
{
    new AuthWindow().Show();
    Close();
}

/* Функция взята из документации и примеров использования этой библиотеки:
   https://learn.microsoft.com/en-us/office/open-xml/how-to-insert-text-into-a-
→ cell-in-a-spreadsheet#:~:text=it.%20%0A%20%20%20%20private%20static%20Cell-,I
→ nsertCellInWorksheet,-(string%20columnName%2C%20uint
*/
private static Cell InsertCellInWorksheet(string columnName, uint rowIndex,
→ WorksheetPart worksheetPart)
{
    var worksheet = worksheetPart.Worksheet;
    var sheetData = worksheet.GetFirstChild<SheetData>()!;
    var cellReference = columnName + rowIndex;

    // If the worksheet does not contain a row with the specified row index,
→ insert one.
    Row row;
    if (sheetData.Elements<Row>().Where(r => r.RowIndex! == rowIndex).Count()
→ != 0)
    {
        row = sheetData.Elements<Row>().Where(r => r.RowIndex! ==
→ rowIndex).First();
    }
    else
    {
        row = new Row() {RowIndex = rowIndex};
        sheetData.Append(row);
    }

    // If there is not a cell with the specified column name, insert one.

```

```

        if (row.Elements<Cell>().Where(c => c.CellReference!.Value == columnName
→ + rowIndex).Count() > 0)
    {
        return row.Elements<Cell>().Where(c => c.CellReference!.Value ==
→ cellReference).First();
    }
    else
    {
        // Cells must be in sequential order according to CellReference.
→ Determine where to insert the new cell.

        Cell refCell = null;
        foreach (var cell in row.Elements<Cell>())
            if (cell.CellReference!.Value!.Length == cellReference.Length)
                if (string.Compare(cell.CellReference.Value, cellReference,
→ true) > 0)
            {
                refCell = cell;
                break;
            }

        var newCell = new Cell() {CellReference = cellReference};
        row.InsertBefore(newCell, refCell);

        worksheet.Save();
        return newCell;
    }
}

private void Import(object sender, RoutedEventArgs e)
{
    // Application excelApp = new ();
    // Workbook workbook = excelApp.Workbooks.Add();
    /* Из-за того что умные люди в майкрософте, очень умные и не сделали
→ нормальных способов для COM dependends
    Мы будем использовать адекватный вариант - OpenDocument(это официальная
→ библиотека от microsoft, так
    что ручки чистые)
    */
}

SaveFileDialog fileDialog = new();

fileDialog.InitialDirectory =
→ Environment.GetEnvironmentVariable("USERHOME");

```

```

;

fileDialog.Filter = "Excel files (*.xlsx)|*.xlsx";
fileDialog.FilterIndex = 1;
string filePath;
if (fileDialog.ShowDialog() == true)
    filePath = fileDialog.FileName;
else
    return;

var spreadsheetDocument =
    SpreadsheetDocument.Create(filePath,
    SpreadsheetDocumentType.Workbook);
var workbookpart = spreadsheetDocument.AddWorkbookPart();
workbookpart.Workbook = new Workbook();

var worksheetPart = workbookpart.AddNewPart<WorksheetPart>();
worksheetPart.Worksheet = new Worksheet(new SheetData());
var sheets = spreadsheetDocument.WorkbookPart!.Workbook.AppendChild(new
Sheets());
var sheet = new Sheet
{
    Id = spreadsheetDocument.WorkbookPart!.GetIdOfPart(worksheetPart),
    SheetId = 1, Name = "Ваша таблица";
    sheets.Append(sheet);
    uint rowId = 1;
    uint columnId = 0;
    foreach ( DataColumn column in _dataTable.Columns)
    {
        var cell = InsertCellInWorksheet(Constants.ColumnNames[(int)
columnId++], rowId, worksheetPart);
        cell.CellValue = new CellValue(column.ToString());
        cell.DataType = new EnumValue<CellValues>(CellValues.String);
    }
};

rowId++;
foreach ( DataRow row in _dataTable.Rows)
{
    columnId = 0;
    foreach ( var item in row.ItemArray)
    {
        var cell = InsertCellInWorksheet(Constants.ColumnNames[(int)
columnId++], rowId, worksheetPart);
        cell.CellValue = new CellValue(item.ToString());
        // По хорошему надо сделать преобразование всех типов
    }
}

```

```

        cell.DataType = new EnumValue<CellValues>(item is int ?
→   CellValues.Number : CellValues.String);
    }

    rowId++;
}

workbookpart.Workbook.Save();
spreadsheetDocument.Close();
}

protected override void OnClosed(EventArgs e)
{
    using (StreamWriter writer = new("log.txt", true))
    {
        writer.WriteLineAsync($"({_userId} logged out in {DateTime.Now})");
    }

    base.OnClosed(e);
}
}

```

Константы:

```

using System.Collections.Generic;

namespace DataBase;

public enum Mode
{
    Auth,
    Reg
}

public enum Role
{
    User = 1,
    Admin = 2
}

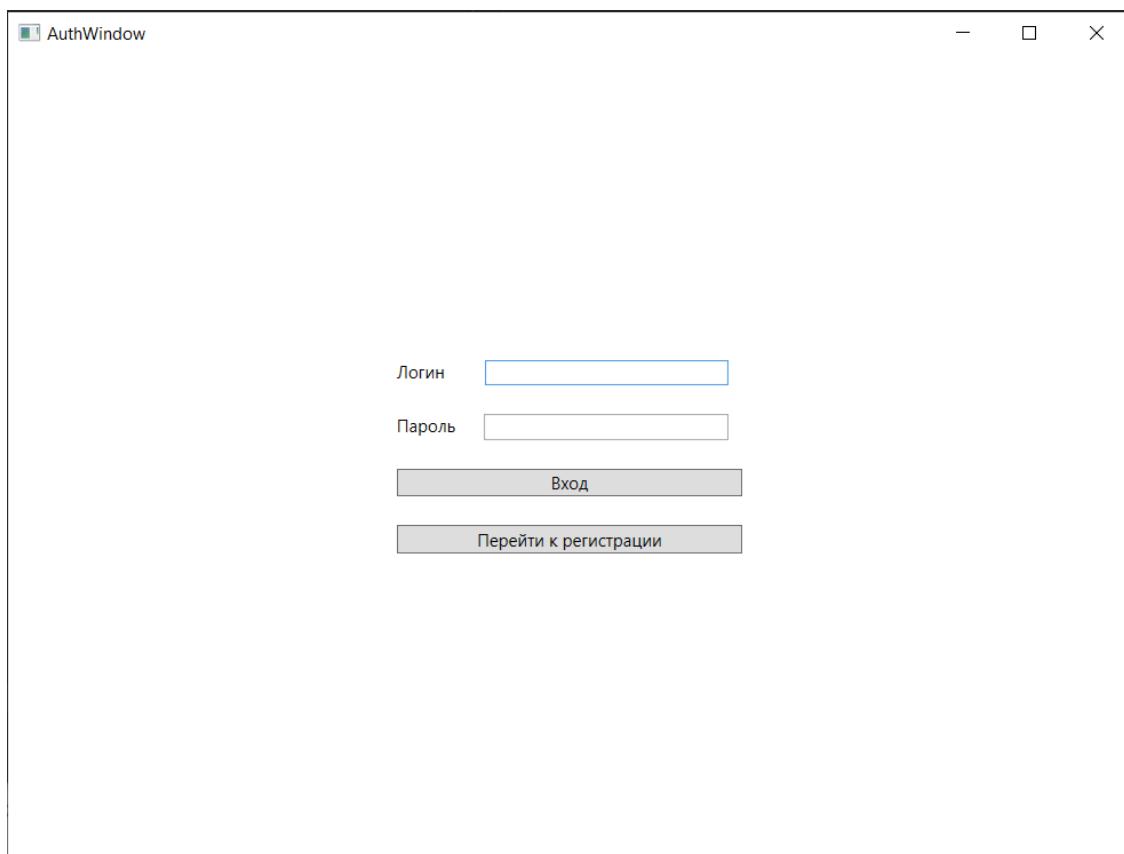
public static class Constants
{

```

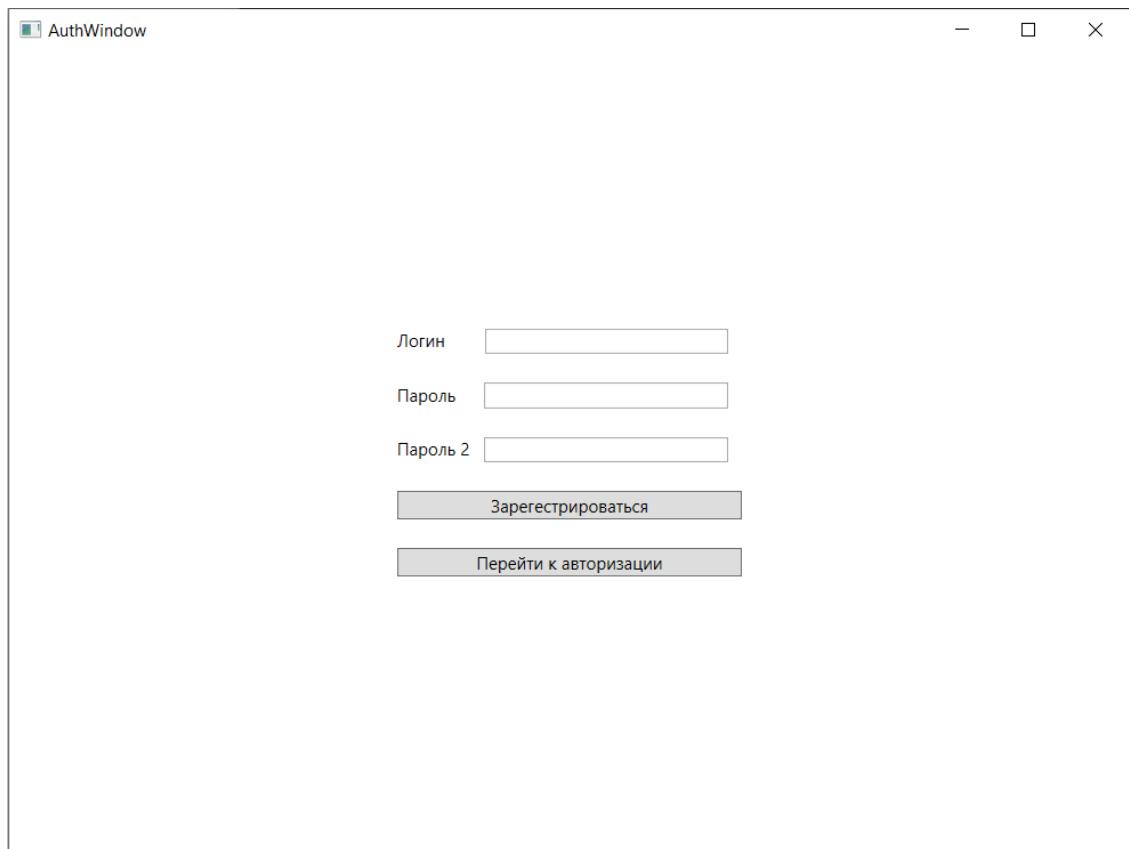
```
public static Dictionary<int, Role> RoleByInt = new() {{2, Role.Admin}, {1, Role.User}};
public static List<string> ColumnNames = new() {"A", "B", "C", "D", "E", "F",
"Г", "H", "I", "J", "K", "L", "M"};
}
```

Демонстрация работы приложения:

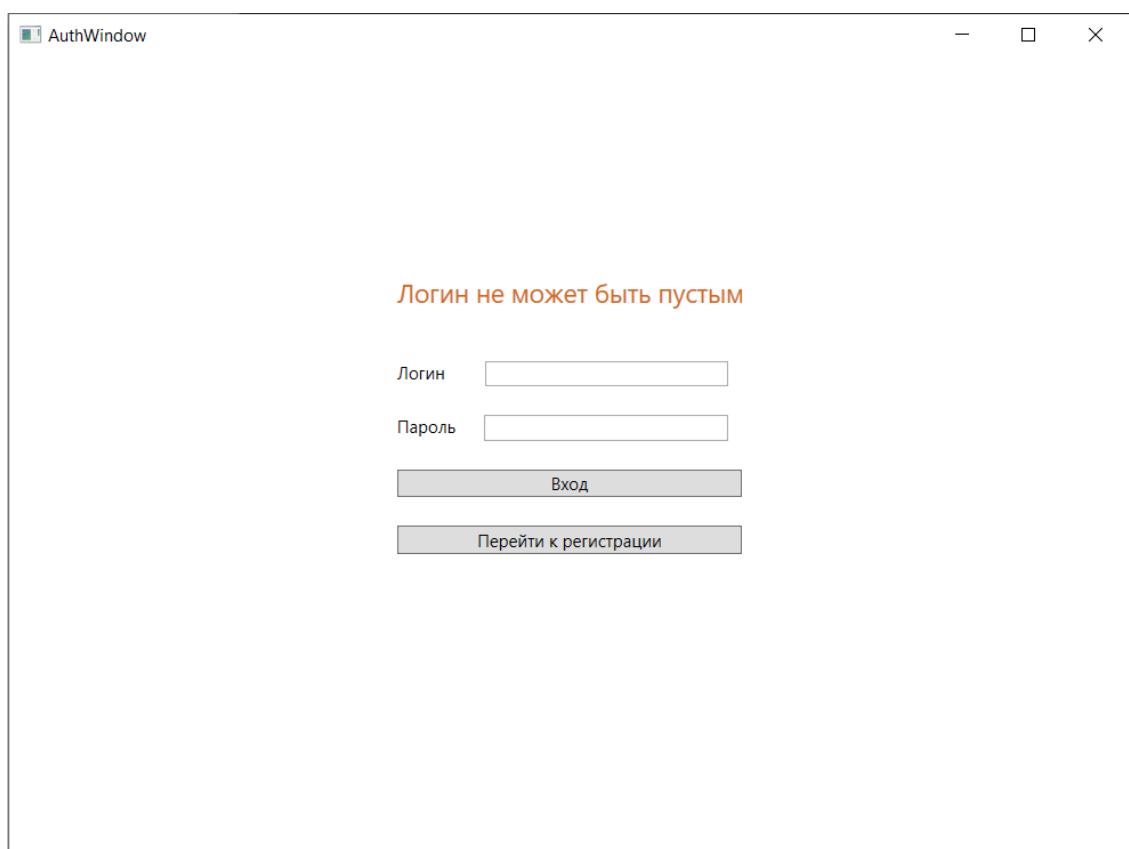
Окно авторизации:



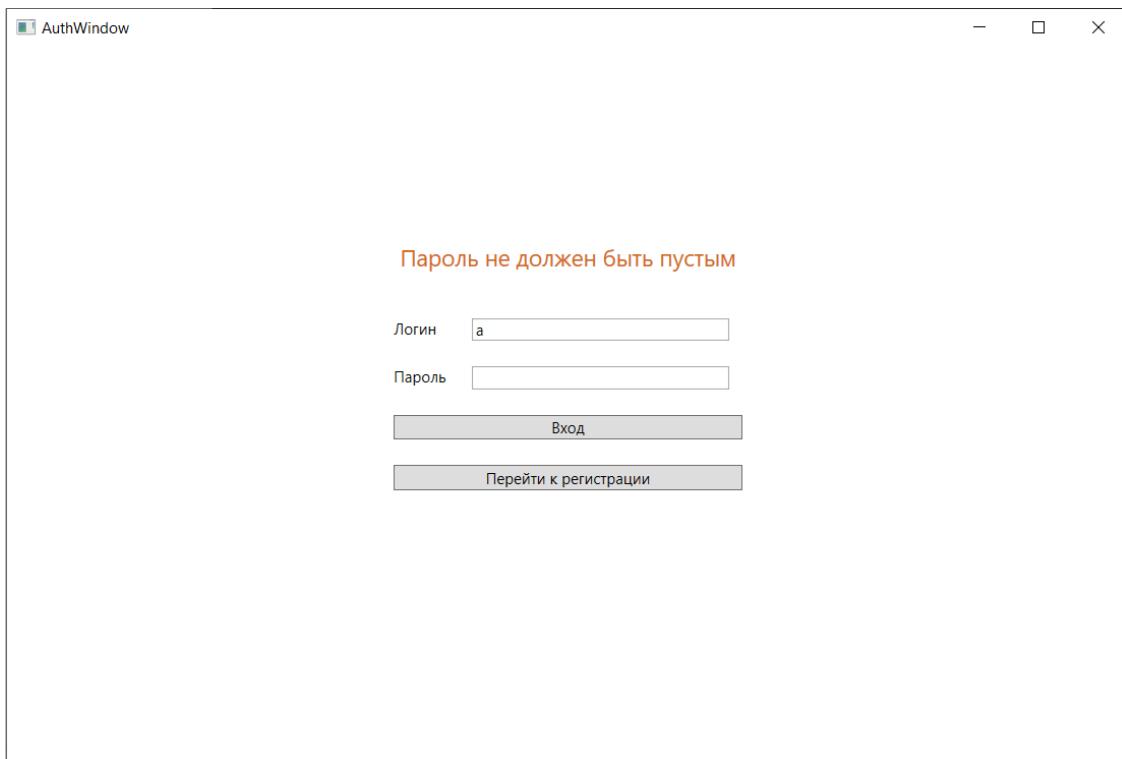
Переключение на регистрацию:



Проверка на пустой логин:



Проверка на пустой пароль:



Вид программы от обычного пользователя:

A screenshot of a Windows application window titled "MainWindow". The main area is a data grid displaying a list of students with columns: №St, Familia, Imya, Otchestvo, Adres, Gorod, Dataroggd, №Pr, Pol, and Budget. The data grid lists numerous entries, such as Terentyev Vячеслав Сергеевич from ul. Pobedy and Klyavin Mаксим Павлович from ul. Marata. At the bottom of the grid, there are search and filter controls: "Фамилия:" with a dropdown arrow, a "Поиск" (Search) button, a "Юноши с возрастом" (Teenagers with age) checkbox, and several other checkboxes for "Поиск по группе" (Search by group), "Иногородние студенты" (Out-of-town students), "Образовательная реформа № 1" (Educational reform № 1), "Поиск по году рождения" (Search by birth year), "Образовательная реформа № 2" (Educational reform № 2), "Войти в другой аккаунт" (Log in to another account), and "Import to Excel".

Демонстрация работы поиска:

MainWindow

NºSt	Familia	Imya	Otchestvo	Adres	Gorod	Datograd	Nºgr	Pol	Budget
21806	Терентьев	Вячеслав	Сергеевич	ул. Победы		9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
23456	Соловьев	Михаил	Сергеевич	ул. Курчатова		12/21/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
26795	Головлев	Михаил	Валерьевич	ул. Костюшко		5/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25370	Дмитриев	Андрей	Валерьевич	ул. Фрунзе		10/15/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21045	Кренев	Ярослав	Сергеевич	Московский пр.		3/3/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
24336	Терентьева	Наталья	Сергеевна	ул. Победы		9/9/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
23467	Соболеев	Илья	Дмитриевич	пр. Ставек		3/10/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
21544	Жиляев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input type="checkbox"/>
29865	Кухарева	Галина	Григорьевна	ул. Зины П.	г. Пушкин	5/13/1997 12:00:00 AM	31M	X	<input type="checkbox"/>

Фамилия: Юноши с возрастом

Поиск по группе Иногородние студенты

Поиск по году рождения

MainWindow

NºSt	Familia	Imya	Otchestvo	Adres	Gorod	Datograd	Nºgr	Pol	Budget
29608	Цой	Виктор	Робертович	пр. Ветеранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>

Фамилия: Юноши с возрастом

Поиск по группе Иногородние студенты

Поиск по году рождения

Демонстрация работы запроса возраста юнош:

MainWindow

Возраст	№St	Familia	Imya	Otchestvo	Adres	Gorod	Datarogr	№gr	Pol	Budget
24	21806	Терентьев	Вячеслав	Сергеевич	ул. Победы		9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	21678	Адвокатов	Владимир	Игорьевич	ул. Благодатная		1/23/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
25	23456	Соловьев	Михаил	Сергеевич	ул. Курчатова		12/21/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	26795	Головлев	Михаил	Валерьевич	ул. Костошко		5/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	25370	Дмитриев	Андрей	Валерьевич	ул. Фрунзе		10/15/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	27456	Розонов	Дмитрий	Сергеевич	ул. Костошко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	28954	Клявин	Максим	Павлович	ул. Марата	г. Гатчина	4/27/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
25	24786	Шубин	Александр	Сергеевич	Невский пр.		2/1/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
25	21045	Кренев	Ярослав	Сергеевич	Московский пр.		3/3/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25	21876	Шапцов	Александер	Сергеевич	ул. Варшавская		6/11/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
25	24735	Полховский	Максим	Генадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
25	20087	Таран	Семен	Игорьевич	ул. Тележная		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
25	21178	Дроздов	Максим	Алексеевич	пр. Художников		4/12/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
25	23467	Соболев	Илья	Дмитриевич	пр. Стachek		3/10/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
25	21544	Жиляев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input type="checkbox"/>
25	24659	Куликов	Юрий	Михайлович	ул. Ж. Диокло		9/26/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
25	23460	Кононец	Валерий	Иванович	21-Линия		7/19/1997 12:00:00 AM	31M	M	<input type="checkbox"/>
25	26849	Платонов	Николай	Павлович	8-Линия		6/19/1997 12:00:00 AM	31M	M	<input type="checkbox"/>
25	29608	Цой	Виктор	Робертович	пр. Ветеранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты [Образовательная реформа № 1](#)

Поиск по году рождения [Образовательная реформа № 2](#) [Войти в другой аккаунт](#) [Import to Excel](#)

MainWindow

Возраст	№St	Familia	Imya	Otchestvo	Adres	Gorod	Datarogr	№gr	Pol	Budget
25	20087	Таран	Семен	Игорьевич	ул. Тележная		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты [Образовательная реформа № 1](#)

Поиск по году рождения [Образовательная реформа № 2](#) [Войти в другой аккаунт](#) [Import to Excel](#)

Демонстрация работы поиска по группе:

MainWindow

Фамилия и инициалы	
Терентьев В. С.	
Адвокатов В. И.	
Соловьев М. С.	
Головлев М. В.	
Дмитриев А. В.	
Розонов Д. С.	
Клявин М. П.	
Шубин А. С.	
Кренев Я. С.	
Шапцов А. С.	

Фамилия: Поиск

Поиск по группе Номер группы: 31Р Образовательная реформа № 1

Поиск по году рождения Войти в другой аккаунт Import to Excel

Демонстрация работы поиска иногородних студентов:

MainWindow

Familia	Nºgr
Клявин	31Р
Жилев	31Т
Карабицина	31М
Кухарева	31М

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты

Поиск по году рождения Войти в другой аккаунт Import to Excel

Демонстрация работы поиска по году рождения:

Fамилия	№гр
Адвокатов	31Р
Соловьев	31Р
Головлев	31Р
Дмитриев	31Р
Розонов	31Р
Клягин	31Р
Шубин	31Р
Кренев	31Р
Шапцов	31Р
Полховский	31Т
Таран	31М
Журина	31Т
Филиппова	31Т
Терентьева	31Т
Куницина	31Т
Дроздов	31Т
Соболеев	31Т
Жиляев	31Т
Карабицина	31М
Куликов	31М
Кухарева	31М
Кононец	31М
Попова	31М
Платонов	31М
Петрова	31М
Шишина	31М

Фамилия: Постк

Поиск по группе [Образовательная реформа № 1](#)

Поиск по году рождения Год рождения: 1997 [Образовательная реформа № 2](#) [Войти в другой аккаунт](#) [Import to Excel](#)

Проверки при регистрации:

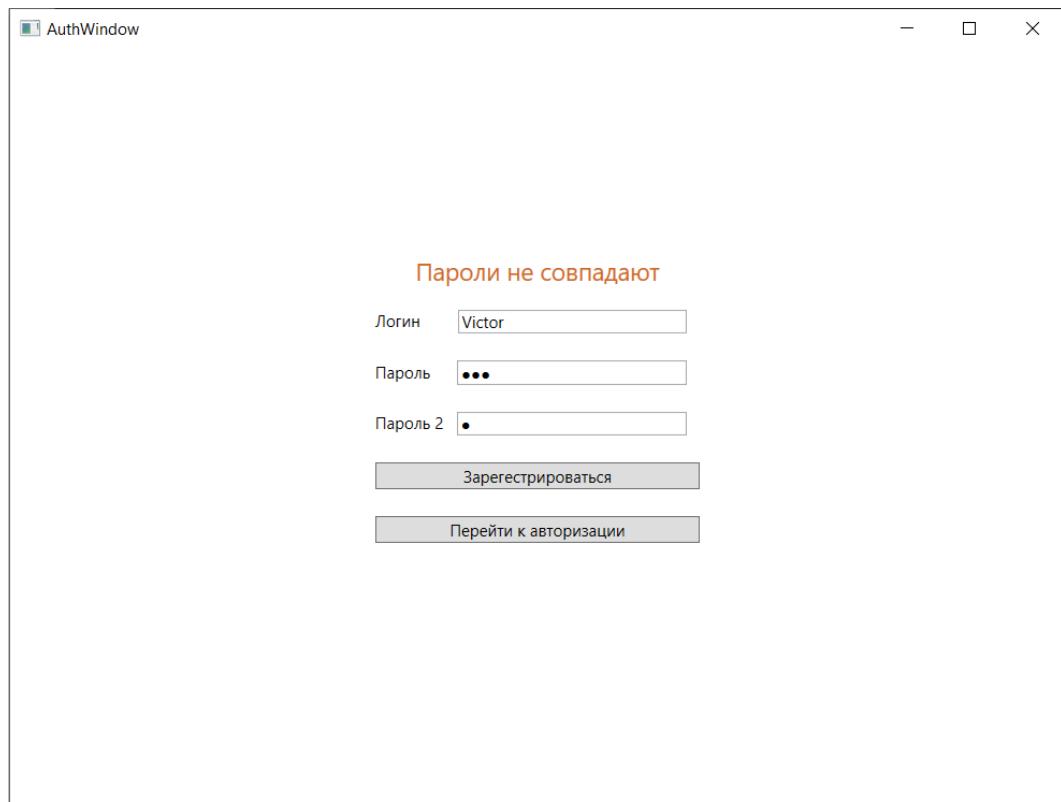
- Логин уже существует

AuthWindow

Такой логин уже существует

Логин	<input type="text" value="a"/>
Пароль	<input type="password" value="•"/>
Пароль 2	<input type="password" value="•"/>
<input type="button" value="Зарегистрироваться"/>	
<input type="button" value="Перейти к авторизации"/>	

- Не совпадение паролей



Демонстрация хранения данных:

- Пользователей

	user_id	login	password	users	role
1	admin	admin	8c6976e5b541415bde98bd4dee15dfb167a9c873fc4bb8a81f6f2ab48a918		2
2	3 a	a978112ca1bbdcfac231b39a23dc4da786eff8147c4e72b9807785afeed48bb			1
3	4 asd	688787d8ff144c502c7f5cffaafe2cc588d86079f9de88304c26b0cb99ce91c6			1
4	5 vasya	7a639b5f083943e8bd81f41eb44025b0df1d189d618993f10624a2ace			1
5	6 kirya	6b86b273ff34fce19d6b804eff5a3f5747ada4ea22fd49c01e52ddb7875b4b			1
6	7 as	f4bf9f7fcbedaba0392f108c59d8f4a38b3838efb6487738171b54475c2ade8			1
7	8 Victor	866d116d21281a3a9375c6deca295bfe67c9989581578b1747b59da428e479			1

- Ролей

	role_id	role_name
1	1	user
2	2	admin
*		(No)

Демонстрация работы LOG'ов:

```
3 logged in in 10/17/2022 12:12:43 PM
3 logged out in 10/17/2022 12:12:45 PM
3 logged in in 10/17/2022 12:12:53 PM
3 logged out in 10/17/2022 12:12:55 PM
2 logged in in 10/17/2022 12:13:12 PM
2 logged out in 10/17/2022 12:13:41 PM
3 logged in in 10/17/2022 12:15:17 PM
3 logged out in 10/17/2022 12:15:21 PM
5 logged in in 10/17/2022 12:46:22 PM
6 logged in in 10/17/2022 12:49:08 PM
6 logged out in 10/17/2022 12:49:14 PM
2 logged in in 10/17/2022 12:49:43 PM
2 logged out in 10/17/2022 12:50:24 PM
3 logged in in 10/17/2022 1:19:29 PM
3 logged out in 10/17/2022 1:19:39 PM
7 logged in in 10/17/2022 1:27:35 PM
7 logged out in 10/17/2022 1:27:48 PM
2 logged in in 10/17/2022 1:28:13 PM
2 logged out in 10/17/2022 1:30:47 PM
3 logged in in 10/17/2022 1:51:58 PM
3 logged out in 10/17/2022 1:55:00 PM
8 logged in in 10/17/2022 1:57:03 PM
8 logged out in 10/17/2022 1:57:06 PM
```

Вид программы от администратора и работы образовательных реформ:

- Образовательная реформа №1

Перемещение всех юношей без бюджета в Колпино и перевод их на бюджет

N ^o St	Familia	Имя	Отчество	Adres	Gorod	Datargd	N ^o gr	Pol	Budget
21806	Терентьев	Вячеслав	Сергеевич	ул. Победы	г. Колпино	9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21678	Адвокатов	Владимир	Игоревич	ул. Благодатная	г. Колпино	1/23/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
23456	Соловьев	Михаил	Сергеевич	ул. Курчатова		12/21/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
26795	Головлев	Михаил	Валерьевич	ул. Костюшко		5/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25370	Дмитриев	Андрей	Валерьевич	ул. Фрунзе		10/15/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
27456	Розонов	Дмитрий	Сергеевич	ул. Костюшко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
28854	Кивин	Максим	Павлович	ул. Марата	г. Колпино	4/27/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
24786	Шубин	Александр	Сергеевич	Невский пр.	г. Колпино	2/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21045	Кренев	Ярослав	Сергеевич	Московский пр.		3/3/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21876	Шапцов	Александр	Сергеевич	ул. Варшавская	г. Колпино	6/11/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
24735	Полховский	Максим	Геннадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
20887	Таран	Семен	Игоревич	ул. Тележкина		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
24556	Журина	Мария	Сергеевна	пр. Обухова		4/30/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
29952	Филиппов	Кристина	Сергеевна	ул. Кузнецкая		9/28/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
24396	Терентьев	Наталья	Аркадьевна	ул. Тоббэни		9/9/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
28167	Куницина	Ольга	Артемьевна	пр. Большевиков		11/1/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
26547	Денисюк	Юлия	Алексеевна	Кировский пр.		8/15/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
2178	Дородова	Мария	Александровна	пр. Художников		4/12/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
23467	Соболев	Илья	Дмитриевич	пр. Степана Разина		2/70/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
21544	Жилев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
27690	Карабичева	Ольга	Аркадьевна	ул. Тоббэни	г. Павловка	12/14/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
24659	Куликов	Юрий	Михайлович	ул. Ж. Дюкло		9/26/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
28865	Кударева	Галина	Григорьевна	ул. Зины П.	г. Пушкин	5/13/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
23460	Кононец	Валерий	Иванович	21-Линия	г. Колпино	7/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
27166	Попов	Елена	Борисовна	ул. Победы		3/4/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
26849	Платонов	Николай	Павлович	ул. Линия	г. Колпино	6/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21347	Петрова	Людмила	Филипповна	Лиговский пр.		5/12/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
21375	Шашкина	Рамса	Ивановна	Лиговский пр.		6/9/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
29608	Цой	Виктор	Робертович	пр. Петерранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21474	Смирнова	Татьяна	Михайловна	пр. Ленина		8/30/1997 12:00:00 AM	31M	X	<input type="checkbox"/>

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты Образовательная реформа №1

Поиск по году рождения Образовательная реформа №2 Войти в другой аккаунт Import to Excel

- Образовательная реформа №2

Отчисление всех с контрактной формы

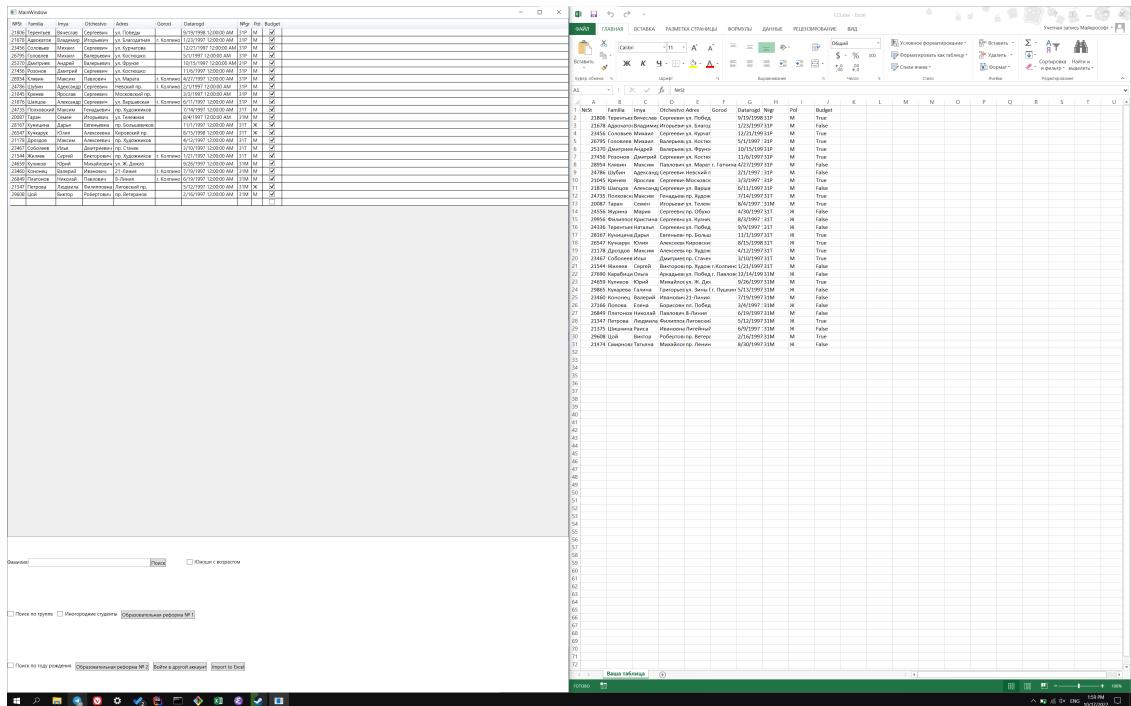
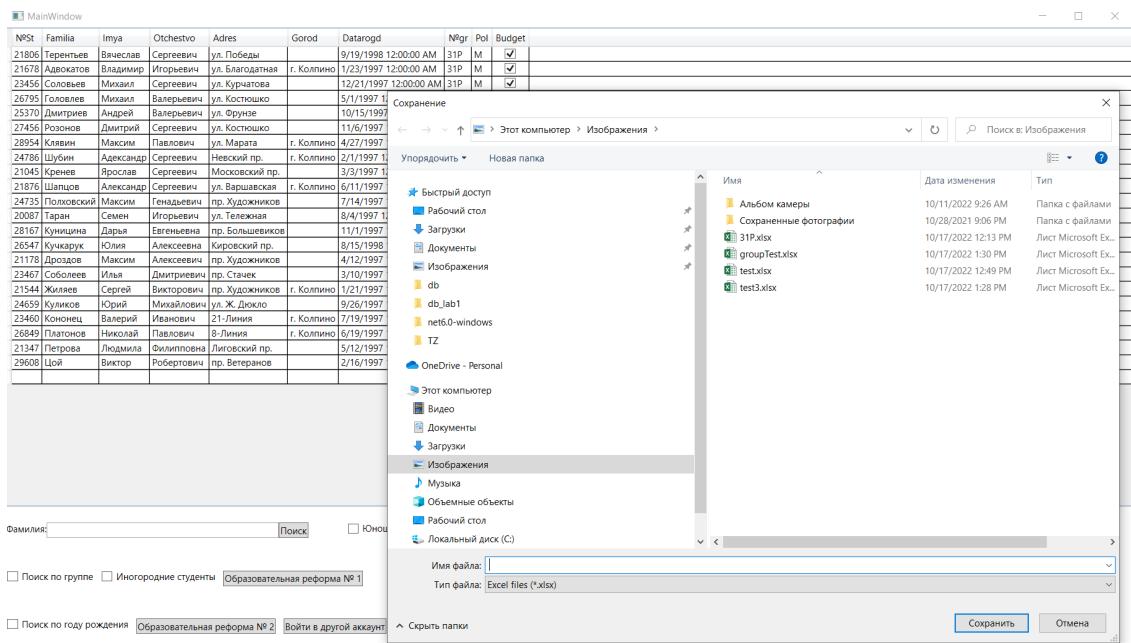
N ^o St	Familia	Имя	Отчество	Adres	Gorod	Datargd	N ^o gr	Pol	Budget
21806	Терентьев	Вячеслав	Сергеевич	ул. Победы	г. Колпино	9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21678	Адвокатов	Владимир	Игоревич	ул. Благодатная	г. Колпино	1/23/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
23456	Соловьев	Михаил	Сергеевич	ул. Курчатова		12/21/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
26795	Головлев	Михаил	Валерьевич	ул. Костюшко		5/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
25370	Дмитриев	Андрей	Валерьевич	ул. Фрунзе		10/15/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
27456	Розонов	Дмитрий	Сергеевич	ул. Костюшко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
28854	Кивин	Максим	Павлович	ул. Марата	г. Колпино	4/27/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
24786	Шубин	Александр	Сергеевич	Невский пр.	г. Колпино	2/1/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21045	Кренев	Ярослав	Сергеевич	Московский пр.		3/3/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21876	Шапцов	Александр	Сергеевич	ул. Варшавская	г. Колпино	6/11/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
24735	Полховский	Максим	Геннадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
20887	Таран	Семен	Игоревич	ул. Тележкина		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
28167	Куницина	Ольга	Артемьевна	пр. Большевиков		11/1/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
26547	Денисюк	Юлия	Алексеевна	Кировский пр.		8/15/1998 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
2178	Дородова	Мария	Александровна	пр. Художников		3/10/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
21544	Жилев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
24659	Куликов	Юрий	Михайлович	ул. Ж. Дюкло		9/26/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
23460	Кононец	Валерий	Иванович	21-Линия	г. Пушкин	7/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
26849	Платонов	Николай	Павлович	ул. Линия	г. Колпино	6/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21347	Петрова	Людмила	Филипповна	Лиговский пр.		5/12/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
29608	Цой	Виктор	Робертович	пр. Петерранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты Образовательная реформа №1

Поиск по году рождения Образовательная реформа №2 Войти в другой аккаунт Import to Excel

Демонстрация работы импорта в Excel:



2 Лабораторная работа № 2

2.1 Часть 1

н Задание 1. Изучить подход code first.

Определим в проекте класс User, объекты которого будут храниться в базе данных. Установим подключение к БД и добавим данные в БД через приложение(на момент запуска приложения БД не существует).

Код:

- Интерфейс (FirstTask.xaml):

```
<Window x:Class="SQLLite_lab.MainWindow"
       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
```

```

    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Title="MainWindow" Height="450" Width="800">
<Grid>
    <DataGrid Name="DataGrid" />
</Grid>
</Window>

```

- Класс User (User.cs)

```

namespace SQLite_lab;

public class User
{
    public int Id { get; set; }

    public string Name { get; set; }

    public int Age { get; set; }
}

```

- Логика (FirstTask.xaml.cs)

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using Microsoft.EntityFrameworkCore;

namespace SQLite_lab;

/// <summary>
/// Interaction logic for FirstTask.xaml

```

```

/// </summary>
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
        Load();
    }

    private void Load()
    {
        User user = new() {Name = "Вася", Age = 10, Id = 1};
        var db = ApplicationContext.GetContext();
        db.Database.EnsureCreated();
        db.Users.Add(user);
        db.SaveChanges();
        DataGrid.ItemsSource = db.Users.ToList();
    }
}

```

Работа приложения:

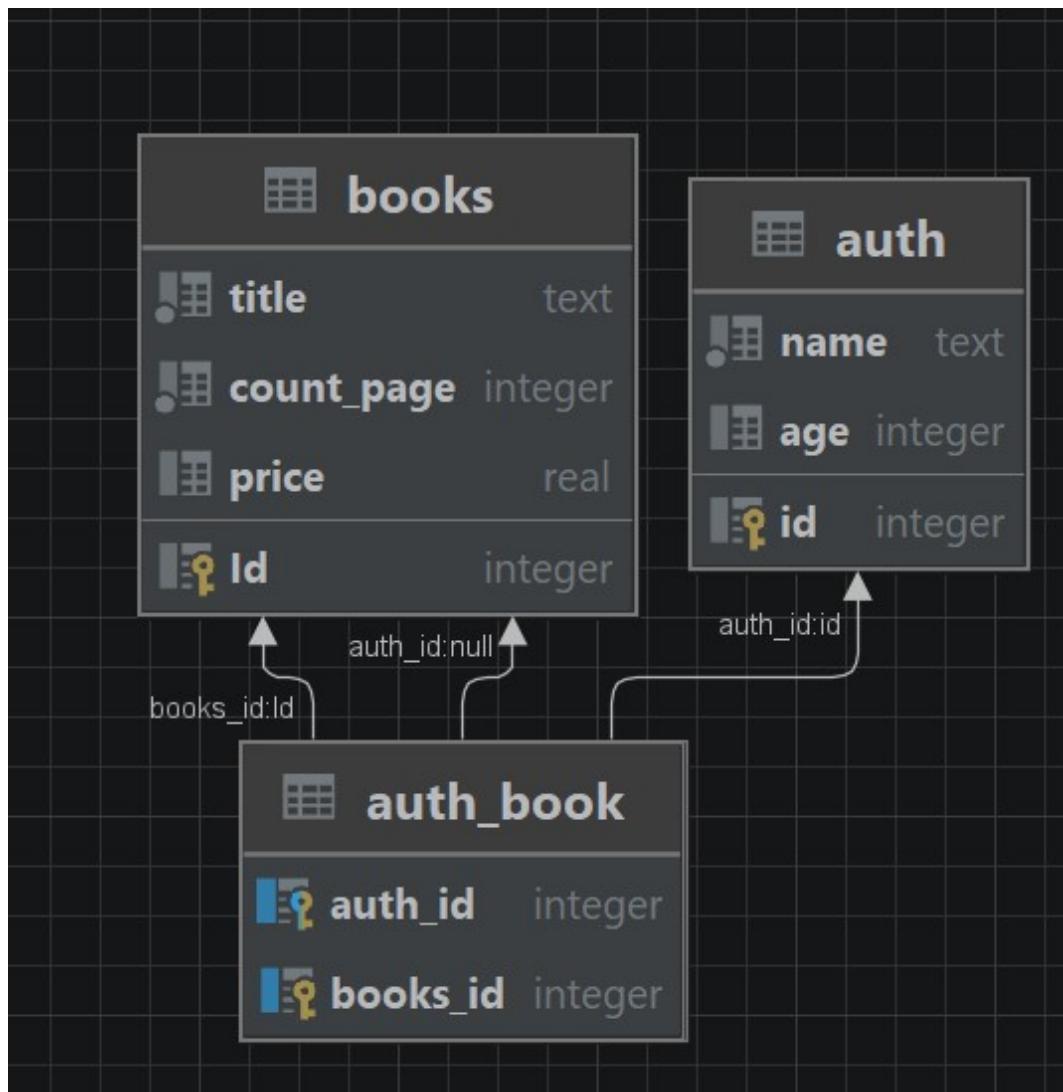
Id	Name	Age
1	Вася	10

Задание 2. Изучить подход database first.

Создать БД из трех связанных таблиц в DB Browser for SQLite(Таблица Авторы, Книги, Книги_Авторы). Используя команду обратного проектирования, получить классы сущностей и контекстов на основе схемы существующей базы данных.

Код:

- База данных:



- Консольная команда использованная для преобразование БД в .cs файлы:

```

dotnet ef dbcontext scaffold "Data Source=C:\Users\user\Desktop\labs\04.01\SQLite-lab\SQLite-lab\res\films.db" Microsoft.EntityFrameworkCore.Sqlite
  -c AcmeDataContext --project .\SQLite-lab\SQLite-lab.csproj
  
```

- Полученные файлы на основе базы данных:

- BooksDataContext.cs

```

using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata;
  
```

```

namespace SQLite_lab;
  
```

```

public partial class BooksDataContext : DbContext
{
  
```

```

private static BooksDataContext? _context;

public BooksDataContext()
{
}

public BooksDataContext(DbContextOptions<BooksDataContext> options)
    : base(options)
{
}

public static BooksDataContext GetContext()
{
    if (_context == null) _context = new BooksDataContext();
    return _context;
}

public virtual DbSet<Auth> Auths { get; set; } = null!;
public virtual DbSet<AuthBook> AuthBooks { get; set; } = null!;
public virtual DbSet<Book> Books { get; set; } = null!;

protected override void OnConfiguring(DbContextOptionsBuilder
→ optionsBuilder)
{
    if (!optionsBuilder.IsConfigured)
    {
#warning To protect potentially sensitive information in your connection
→ string, you should move it out of source code. You can avoid
→ scaffolding the connection string by using the Name= syntax to read
→ it from configuration - see https:
→ //go.microsoft.com/fwlink/?linkid=2131148. For more guidance on
→ storing connection strings, see
→ http://go.microsoft.com/fwlink/?LinkId=723263.
        optionsBuilder.UseSqlite(
            "Data Source=C:\\\\Users\\\\user\\\\Desktop\\\\labs\\\\04.01\\\\SQLi_"
→ te-lab\\\\SQLite-lab\\\\res\\\\Books.db");
    }
}

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Auth>(entity =>
    {
        entity.ToTable("auth");
    });
}

```

```

entity.Property(e => e.Id).HasColumnName("id");

entity.Property(e => e.Age).HasColumnName("age");

entity.Property(e => e.Name).HasColumnName("name");
});

modelBuilder.Entity<AuthBook>(entity =>
{
    entity.HasKey(e => new { e.AuthId, e.BooksId });

    entity.ToTable("auth_book");

    entity.Property(e => e.AuthId).HasColumnName("auth_id");

    entity.Property(e => e.BooksId).HasColumnName("books_id");

    entity.HasOne(d => d.Auth)
        .WithMany(p => p.AuthBooks)
        .HasForeignKey(d => d.AuthId)
        .OnDelete(DeleteBehavior.ClientSetNull);

    entity.HasOne(d => d.AuthNavigation)
        .WithMany(p => p.AuthBooks)
        .HasForeignKey(d => d.AuthId)
        .OnDelete(DeleteBehavior.ClientSetNull);
});

modelBuilder.Entity<Book>(entity =>
{
    entity.ToTable("books");

    entity.Property(e =>
        e.CountPage).HasColumnName("count_page");

    entity.Property(e => e.Price).HasColumnName("price");

    entity.Property(e => e.Title).HasColumnName("title");
});

OnModelCreatingPartial(modelBuilder);
}

```

```

        partial void OnModelCreatingPartial(ModelBuilder modelBuilder);
    }

- Book.cs

using System;
using System.Collections.Generic;

namespace SQLite_lab;

public partial class Book
{
    public Book()
    {
        AuthBooks = new HashSet<AuthBook>();
    }

    public long Id { get; set; }
    public string Title { get; set; } = null!;
    public long CountPage { get; set; }
    public double? Price { get; set; }

    public virtual ICollection<AuthBook> AuthBooks { get; set; }
}

• Auth.cs

using System;
using System.Collections.Generic;

namespace SQLite_lab;

public partial class Auth
{
    public Auth()
    {
        AuthBooks = new HashSet<AuthBook>();
    }

    public long Id { get; set; }
    public string Name { get; set; } = null!;
    public long? Age { get; set; }

    public virtual ICollection<AuthBook> AuthBooks { get; set; }
}

```

- AuthBook.cs

```

using System;
using System.Collections.Generic;

namespace SQLite_lab;

public partial class AuthBook
{
    public long AuthId { get; set; }
    public long BooksId { get; set; }

    public virtual Auth Auth { get; set; } = null!;
    public virtual Book AuthNavigation { get; set; } = null!;
}

```

- Интерфейс (SecondTask.xaml)

```

<Window x:Class="SQLite_lab.SecondTask"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        mc:Ignorable="d"
        Title="SecondTask" Height="450" Width="800">

    <Grid>
        <TabControl>
            <TabItem Header="Авторы">
                <DockPanel LastChildFill="True">
                    <StackPanel DockPanel.Dock="Bottom"
                               Orientation="Horizontal">
                        <TextBlock>Имя: </TextBlock>
                        <TextBox Margin="10,0" Name="Name" Text="{Binding
                                         NewAuthor.Name}" Width="100"></TextBox>
                        <TextBlock>Возраст: </TextBlock>
                        <TextBox Margin="10,0" Name="Age" Text="{Binding
                                         NewAuthor.Age}" Width="100"></TextBox>
                        <Button Click="ButtonBase_OnClick"
                               Content="Сохранить"></Button>
                </StackPanel>
                <DataGrid Name="AuthorsGrid" />
            </DockPanel>
        </TabItem>
    </TabControl>
</Grid>

```

```

        </TabItem>
        <TabItem Header="Книги">
            <DataGrid Name="BooksGrid" />
        </TabItem>
        <TabItem Header="Авторы/Книги">
            <DataGrid Name="AuthorsBooksGrid" />
        </TabItem>
    </TabControl>

</Grid>

</Window>

```

- Логика (AuthorsViewModel.cs)

```

using System;
using System.Windows;
using Microsoft.Data.Sqlite;
using Microsoft.EntityFrameworkCore;

namespace SQLite_lab;

public class AuthorsViewModel
{
    public Auth NewAuthor { get; set; } = new Auth();

    public void Save()
    {
        var dbContext = BooksDataContext.GetContext();
        dbContext.Auths.Add(NewAuthor);
        try
        {
            dbContext.SaveChanges();
        }
        catch (DbUpdateException e)
        {
            if ((e.InnerException as SqliteException).SqliteExtendedErrorCode
                == 275)
                MessageBox.Show("Возраст слишком маленький.");
            else
                MessageBox.Show("Имя у вас дурацкое.");
        }
    }
}

```

```
    }  
}
```

- Связка (SecondTask.xaml.cs)

```
using System.Linq;  
using System.Windows;  
  
namespace SQLite_lab;  
  
public partial class SecondTask : Window  
{  
    private AuthorsViewModel viewModel = new AuthorsViewModel();  
    public SecondTask()  
    {  
        InitializeComponent();  
        Load();  
    }  
  
    private void Load()  
    {  
        var db = BooksDataContext.GetContext();  
        AuthorsGrid.ItemsSource = db.Auths.ToList();  
        BooksGrid.ItemsSource = db.Books.ToList();  
        AuthorsBooksGrid.ItemsSource = db.AuthBooks.ToList();  
        DataContext = viewModel;  
    }  
  
    private void ButtonBase_OnClick(object sender, RoutedEventArgs e)  
    {  
        viewModel.Save();  
        var db = BooksDataContext.GetContext();  
        AuthorsGrid.ItemsSource = db.Auths.ToList();  
    }  
}
```

Работа приложения:

- Экраны:

SecondTask

Авторы Книги Авторы/Книги

Id	Name	Age	AuthBooks
1	Victor Petrov	98	
2	Vasya	123	
3	Vasya	123	
4	Кирилл	19	
5	Test2	19	
6	Федя		
7	Вася	45	

Имя: Возраст: Сохранить

SecondTask

Авторы Книги Авторы/Книги

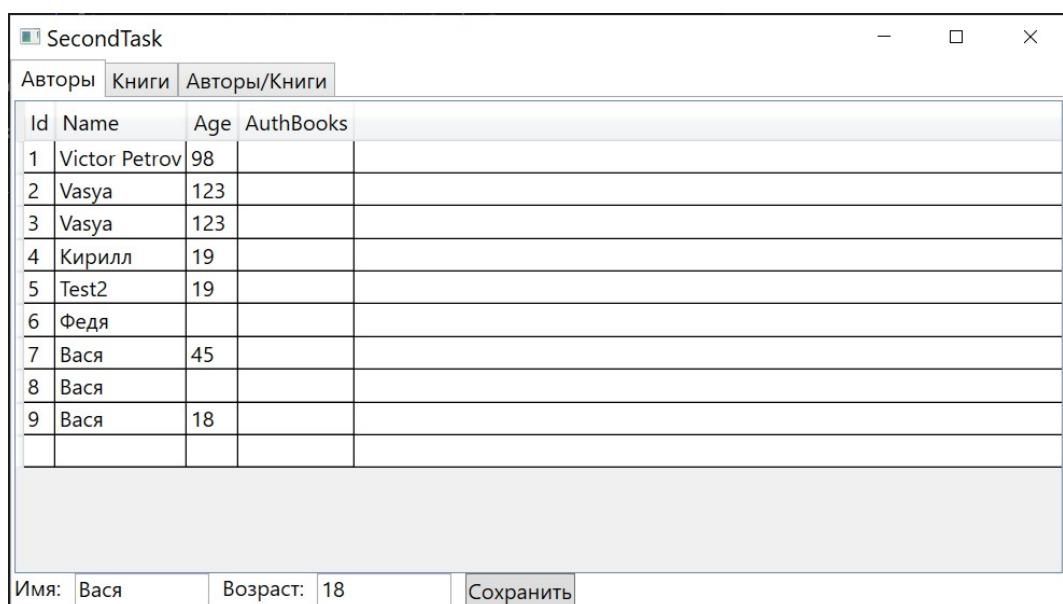
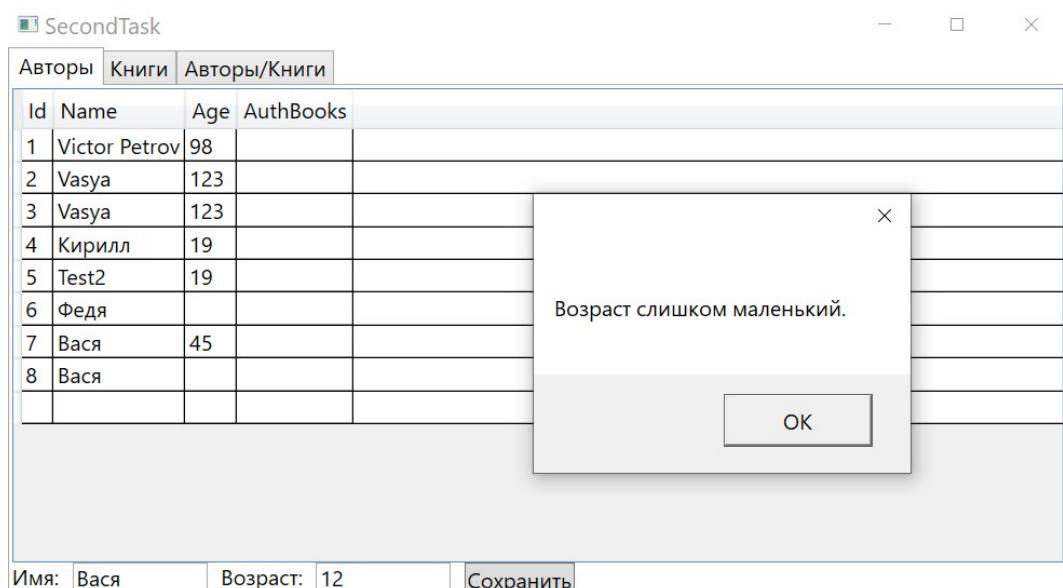
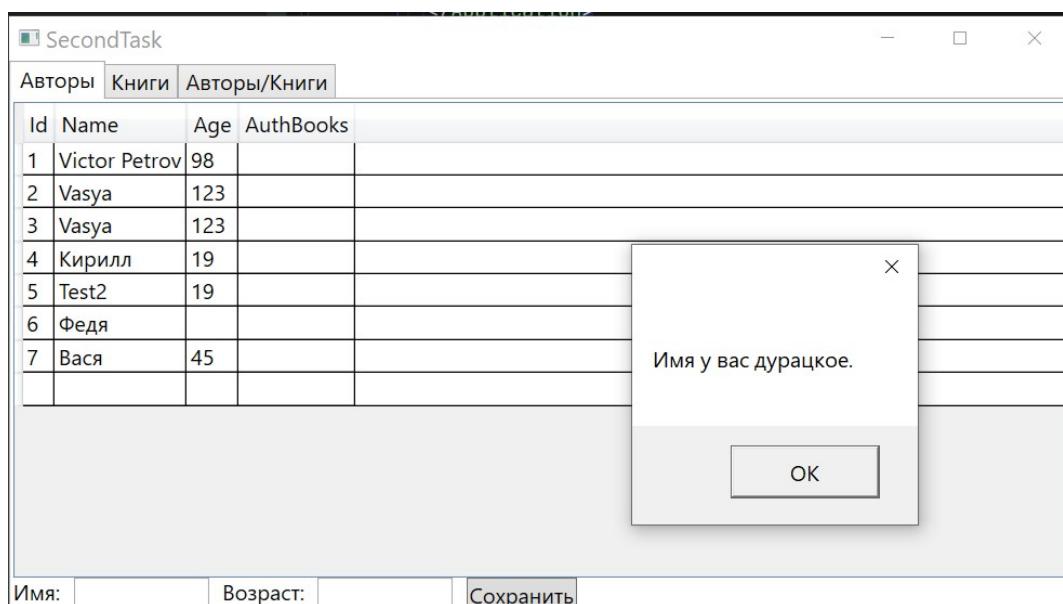
Id	Title	CountPage	Price	AuthBooks
1	Tutorial for C++	200	10000	

SecondTask

Авторы Книги Авторы/Книги

AuthId	BooksId	Auth	AuthNavigation
1	1	SQLite_lab.Auth	SQLite_lab.Book
1	100	SQLite_lab.Auth	SQLite_lab.Book

- Добавление автора:



2.2 Часть 2 (Индивидуальное задание)

1. Уточнить вариант предметной области у преподавателя.
2. Создать базу данных с тремя связанными таблицами с помощью скрипта в DB Browser for SQLite. Скрипт представить в отчете.
3. Используя подход database first, создать настольное приложение, которое будет взаимодействовать с созданной БД.
4. Реализовать функции добавления, изменения и удаления данных в созданные таблицы(через интерфейс приложения).
5. Показать работу ограничений(check и FOREIGN KEY).
6. Сформулировать запросы:
 - на выборку,
 - на использование статистических функций,
 - на соединение таблиц.
7. Добавить поле в одну из таблиц для хранения даты. Показать работу с датами в SQLite.

Задание №19: БД Компьютерной фирмы.

Таблицы:

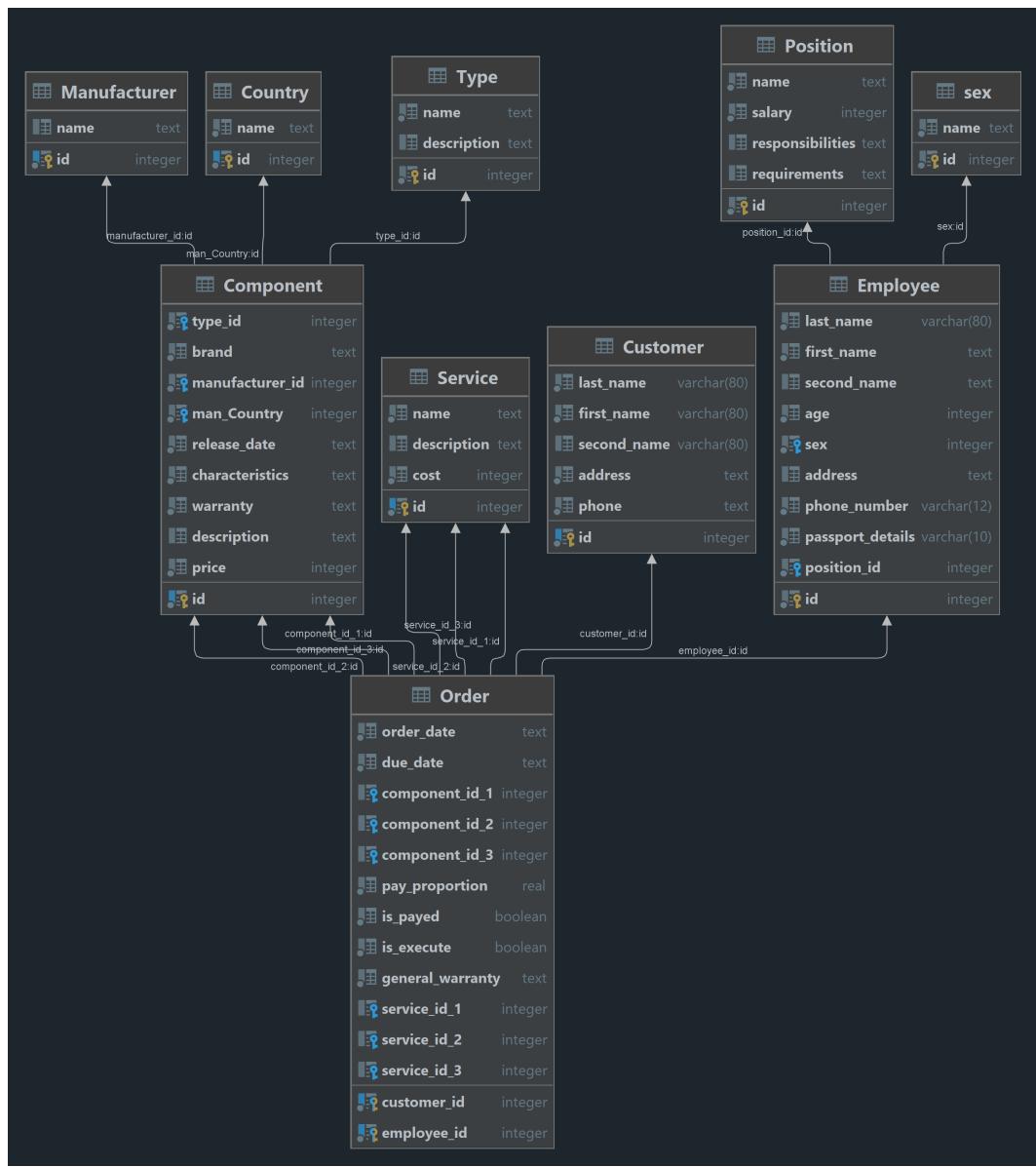
1. Сотрудники (Код сотрудника, ФИО, Возраст, Пол, Адрес, Телефон, Паспортные данные, Код должности).
2. Должности (Код должности, Наименование должности, Оклад, Обязанности, Требования)
3. Виды комплектующих (Код вида, Наименование, Описание)
4. Комплектующие (Код комплектующего, Код вида, Марка, Фирма производитель, Страна производитель, Дата выпуска, Характеристики, Срок гарантии, Описание, Цена)
5. Заказчики (Код заказчика, ФИО, Адрес, Телефон).
6. Услуги (Код услуги, Наименование, Описание, Стоимость)
7. Заказы (Дата заказа, Дата исполнения, Код заказчика, Код комплектующего 1, Код комплектующего 2, Код комплектующего 3, Доля предоплаты, Отметка об оплате, Отметка об исполнении, Общая стоимость, Срок общей гарантии, Код услуги 1, Код услуги 2, Код услуги 3, Код сотрудника).

Запросы:

1. Отобразить заказы отдельных заказчиков;
2. Отобразить комплектующие определенного производителя.

Код:

1. База данных:



2. DbClasses - библиотека классов, куда я сгенерировал работу классы для работы с БД

(a) Component.cs

```

using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Component
{
    public long Id { get; set; }

    public long TypeId { get; set; }

    public string Brand { get; set; } = null!;

    public string Description { get; set; }

    public int Price { get; set; }

    public string Warranty { get; set; }

    public string Characteristics { get; set; }

    public string ReleaseDate { get; set; }

    public int ComponentId1 { get; set; }

    public int ComponentId2 { get; set; }

    public int ComponentId3 { get; set; }

    public int ServiceId1 { get; set; }

    public int ServiceId2 { get; set; }

    public int ServiceId3 { get; set; }

    public string Address { get; set; }

    public string Firstname { get; set; }

    public string Lastname { get; set; }

    public string Secondname { get; set; }

    public int Age { get; set; }

    public int Salary { get; set; }

    public string Responsibilities { get; set; }

    public string Requirements { get; set; }

    public string Sex { get; set; }

    public string PassportDetails { get; set; }

    public int PositionId { get; set; }
}

```

```

public long ManufacturerId { get; set; }

public long ManCountry { get; set; }

public string ReleaseDate { get; set; } = null!;

public string Characteristics { get; set; } = null!;

public string Warranty { get; set; } = null!;

public string? Description { get; set; }

public long Price { get; set; }

public virtual Country ManCountryNavigation { get; set; } = null!;

public virtual Manufacturer Manufacturer { get; set; } = null!;

public virtual ICollection<Order> OrderComponentId1Navigations {
    get; } = new List<Order>();

public virtual ICollection<Order> OrderComponentId2Navigations {
    get; } = new List<Order>();

public virtual ICollection<Order> OrderComponentId3Navigations {
    get; } = new List<Order>();

public virtual Type Type { get; set; } = null!;
}

```

(b) Country.cs

```

using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Country
{
    public long Id { get; set; }

    public string Name { get; set; } = null!;

    public virtual ICollection<Component> Components { get; } = new
        List<Component>();

```

```
    }

(c) Customer.cs

using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Customer
{
    public long Id { get; set; }

    public string LastName { get; set; } = null!;

    public string FirstName { get; set; } = null!;

    public string? SecondName { get; set; }

    public string Address { get; set; } = null!;

    public string Phone { get; set; } = null!;

    public virtual ICollection<Order> Orders { get; } = new
        List<Order>();
}
```

(d) CustomerOrder.cs

```
using System;
using System.Collections.Generic;

namespace DbContext;

public partial class CustomerOrder
{
    public long? Id { get; set; }

    public string? LastName { get; set; }

    public string? FirstName { get; set; }

    public string? SecondName { get; set; }

    public string? OrderDate { get; set; }

    public string? DueDate { get; set; }
```

```

    public long? CustomerId { get; set; }

    public long? ComponentId1 { get; set; }

    public long? ComponentId2 { get; set; }

    public long? ComponentId3 { get; set; }

    public double? PayProportion { get; set; }

    public byte[]? IsPayed { get; set; }

    public byte[]? IsExecute { get; set; }

    public string? GeneralWarranty { get; set; }

    public long? ServiceId1 { get; set; }

    public long? ServiceId2 { get; set; }

    public long? ServiceId3 { get; set; }

    public long? EmployeeId { get; set; }
}

```

(e) DbContext.cs

```

using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;

namespace DbContext;

public partial class DbContext :
    Microsoft.EntityFrameworkCore.DbContext
{
    public DbContext()
    {

    }

    public DbContext(DbContextOptions<DbContext> options)
        : base(options)
    {
    }
}

```

```

public virtual DbSet<Component> Components { get; set; }

public virtual DbSet<Country> Countries { get; set; }

public virtual DbSet<Customer> Customers { get; set; }

public virtual DbSet<CustomerOrder> CustomerOrders { get; set; }

public virtual DbSet<Employee> Employees { get; set; }

public virtual DbSet<Manufacture2Component> Manufacture2Components {
→   get; set; }

public virtual DbSet<Manufacturer> Manufacturers { get; set; }

public virtual DbSet<Order> Orders { get; set; }

public virtual DbSet<Position> Positions { get; set; }

public virtual DbSet<Service> Services { get; set; }

public virtual DbSet<Sex> Sexes { get; set; }

public virtual DbSet<Type> Types { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder
→   optionsBuilder)
#warning To protect potentially sensitive information in your connection
→   string, you should move it out of source code. You can avoid
→   scaffolding the connection string by using the Name= syntax to read
→   it from configuration - see https:
→   //go.microsoft.com/fwlink/?LinkId=2131148. For more guidance on
→   storing connection strings, see
→   http://go.microsoft.com/fwlink/?LinkId=723263.
    => optionsBuilder.UseSqlite("Data Source=C:\\\\Users\\\\user\\\\Desktop\\\\
→   p\\\\labs\\\\04.01\\\\db_lab2_ind\\\\company.db");

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Component>(entity =>
    {
        entity.ToTable("Component");
    });
}

```

```

entity.HasIndex(e => e.Id, "IX_Component_id").IsUnique();

entity.Property(e => e.Id).HasColumnName("id");
entity.Property(e => e.Brand).HasColumnName("brand");
entity.Property(e =>
    e.Characteristics).HasColumnName("characteristics");
    entity.Property(e =>
        e.Description).HasColumnName("description");
        entity.Property(e =>
            e.ManCountry).HasColumnName("man_Country");
            entity.Property(e =>
                e.ManufacturerId).HasColumnName("manufacturer_id");
                entity.Property(e => e.Price).HasColumnName("price");
                entity.Property(e =>
                    e.ReleaseDate).HasColumnName("release_date");
                    entity.Property(e => e.TypeId).HasColumnName("type_id");
                    entity.Property(e => e.Warranty)
                        .HasDefaultValueSql("'1 mec.'")
                        .HasColumnName("warranty");

entity.HasOne(d => d.ManCountryNavigation).WithMany(p =>
    p.Components)
    .HasForeignKey(d => d.ManCountry)
    .OnDelete(DeleteBehavior.ClientSetNull);

entity.HasOne(d => d.Manufacturer).WithMany(p =>
    p.Components)
    .HasForeignKey(d => d.ManufacturerId)
    .OnDelete(DeleteBehavior.ClientSetNull);

entity.HasOne(d => d.Type).WithMany(p => p.Components)
    .HasForeignKey(d => d.TypeId)
    .OnDelete(DeleteBehavior.ClientSetNull);
});

modelBuilder.Entity<Country>(entity =>
{
    entity.ToTable("Country");

    entity.HasIndex(e => e.Id, "IX_Country_id").IsUnique();

    entity.Property(e => e.Id).HasColumnName("id");

```

```

        entity.Property(e => e.Name).HasColumnName("name");
    });

modelBuilder.Entity<Customer>(entity =>
{
    entity.ToTable("Customer");

    entity.HasIndex(e => e.Id, "IX_Customer_id").IsUnique();

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e => e.Address).HasColumnName("address");
    entity.Property(e => e.FirstName)
        .HasColumnType("varchar(80)")
        .HasColumnName("first_name");
    entity.Property(e => e.LastName)
        .HasColumnType("varchar(80)")
        .HasColumnName("last_name");
    entity.Property(e => e.Phone).HasColumnName("phone");
    entity.Property(e => e.SecondName)
        .HasColumnType("varchar(80)")
        .HasColumnName("second_name");
});

modelBuilder.Entity<CustomerOrder>(entity =>
{
    entity
        .HasNoKey()
        .ToView("CustomerOrders");

    entity.Property(e =>
        e.ComponentId1).HasColumnName("component_id_1");
    entity.Property(e =>
        e.ComponentId2).HasColumnName("component_id_2");
    entity.Property(e =>
        e.ComponentId3).HasColumnName("component_id_3");
    entity.Property(e =>
        e.CustomerId).HasColumnName("customer_id");
    entity.Property(e => e.DueDate).HasColumnName("due_date");
    entity.Property(e =>
        e.EmployeeId).HasColumnName("employee_id");
    entity.Property(e => e.FirstName)
        .HasColumnType("varchar(80)")
        .HasColumnName("first_name");
}
);

```

```

        entity.Property(e =>
    ↳ e.GeneralWarranty).HasColumnName("general_warranty");
        entity.Property(e => e.Id).HasColumnName("id");
        entity.Property(e => e.IsExecute)
            .HasColumnType("boolean")
            .HasColumnName("is_execute");
        entity.Property(e => e.IsPayed)
            .HasColumnType("boolean")
            .HasColumnName("is_payed");
        entity.Property(e => e.LastName)
            .HasColumnType("varchar(80)")
            .HasColumnName("last_name");
        entity.Property(e =>
    ↳ e.OrderDate).HasColumnName("order_date");
        entity.Property(e =>
    ↳ e.PayProportion).HasColumnName("pay_proportion");
        entity.Property(e => e.SecondName)
            .HasColumnType("varchar(80)")
            .HasColumnName("second_name");
        entity.Property(e =>
    ↳ e.ServiceId1).HasColumnName("service_id_1");
        entity.Property(e =>
    ↳ e.ServiceId2).HasColumnName("service_id_2");
        entity.Property(e =>
    ↳ e.ServiceId3).HasColumnName("service_id_3");
    });

modelBuilder.Entity<Employee>(entity =>
{
    entity.ToTable("Employee");

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e => e.Address).HasColumnName("address");
    entity.Property(e => e.Age).HasColumnName("age");
    entity.Property(e =>
    ↳ e.FirstName).HasColumnName("first_name");
    entity.Property(e => e.LastName)
        .HasColumnType("varchar(80)")
        .HasColumnName("last_name");
    entity.Property(e => e.PassportDetails)
        .HasColumnType("varchar(10)")
        .HasColumnName("passport_details");
    entity.Property(e => e.PhoneNumber)

```

```

        .HasColumnType("varchar(12)")
        .HasColumnName("phone_number");
    entity.Property(e =>
        e.PositionId).HasColumnName("position_id");
        entity.Property(e =>
        e.SecondName).HasColumnName("second_name");
        entity.Property(e => e.Sex).HasColumnName("sex");

    entity.HasOne(d => d.Position).WithMany(p => p.Employees)
        .HasForeignKey(d => d.PositionId)
        .OnDelete(DeleteBehavior.ClientSetNull);

    entity.HasOne(d => d.SexNavigation).WithMany(p =>
        p.Employees)
        .HasForeignKey(d => d.Sex)
        .OnDelete(DeleteBehavior.ClientSetNull);
    });

modelBuilder.Entity<Manufacture2Component>(entity =>
{
    entity
        .HasNoKey()
        .ToView("Manufacture2Component");

    entity.Property(e => e.Brand).HasColumnName("brand");
    entity.Property(e =>
        e.Characteristics).HasColumnName("characteristics");
        entity.Property(e =>
        e.Description).HasColumnName("description");
        entity.Property(e =>
        e.ManCountry).HasColumnName("man_Country");
        entity.Property(e =>
        e.ManufacturerId).HasColumnName("manufacturer_id");
        entity.Property(e => e.Name).HasColumnName("name");
        entity.Property(e => e.Price).HasColumnName("price");
        entity.Property(e =>
        e.ReleaseDate).HasColumnName("release_date");
        entity.Property(e => e.TypeId).HasColumnName("type_id");
        entity.Property(e => e.Warranty).HasColumnName("warranty");
    });

modelBuilder.Entity<Manufacturer>(entity =>
{

```

```

entity.ToTable("Manufacturer");

entity.HasIndex(e => e.Id, "IX_Manufacturer_id").IsUnique();

entity.Property(e => e.Id).HasColumnName("id");
entity.Property(e => e.Name).HasColumnName("name");
});

modelBuilder.Entity<Order>(entity =>
{
    entity.HasKey(e => new {e.CustomerId, e.EmployeeId});

    entity.ToTable("Order");

    entity.Property(e =>
        e.CustomerId).HasColumnName("customer_id");
    entity.Property(e =>
        e.EmployeeId).HasColumnName("employee_id");
    entity.Property(e =>
        e.ComponentId1).HasColumnName("component_id_1");
    entity.Property(e =>
        e.ComponentId2).HasColumnName("component_id_2");
    entity.Property(e =>
        e.ComponentId3).HasColumnName("component_id_3");
    entity.Property(e => e.DueDate).HasColumnName("due_date");
    entity.Property(e =>
        e.GeneralWarranty).HasColumnName("general_warranty");
    entity.Property(e => e.IsExecute)
        .HasDefaultValueSql("false")
        .HasColumnType("boolean")
        .HasColumnName("is_execute");
    entity.Property(e => e.IsPayed)
        .HasDefaultValueSql("false")
        .HasColumnType("boolean")
        .HasColumnName("is_payed");
    entity.Property(e =>
        e.OrderDate).HasColumnName("order_date");
    entity.Property(e =>
        e.PayProportion).HasColumnName("pay_proportion");
    entity.Property(e =>
        e.ServiceId1).HasColumnName("service_id_1");
    entity.Property(e =>
        e.ServiceId2).HasColumnName("service_id_2");
}
);

```

```

        entity.Property(e =>
    →   e.ServiceId3).HasColumnName("service_id_3");

            entity.HasOne(d => d.ComponentId1Navigation).WithMany(p =>
    →   p.OrderComponentId1Navigations)
                .HasForeignKey(d => d.ComponentId1);

            entity.HasOne(d => d.ComponentId2Navigation).WithMany(p =>
    →   p.OrderComponentId2Navigations)
                .HasForeignKey(d => d.ComponentId2);

            entity.HasOne(d => d.ComponentId3Navigation).WithMany(p =>
    →   p.OrderComponentId3Navigations)
                .HasForeignKey(d => d.ComponentId3);

        entity.HasOne(d => d.Customer).WithMany(p => p.Orders)
            .HasForeignKey(d => d.CustomerId)
            .OnDelete(DeleteBehavior.ClientSetNull);

        entity.HasOne(d => d.Employee).WithMany(p => p.Orders)
            .HasForeignKey(d => d.EmployeeId)
            .OnDelete(DeleteBehavior.ClientSetNull);

        entity.HasOne(d => d.ServiceId1Navigation).WithMany(p =>
    →   p.OrderServiceId1Navigations)
                .HasForeignKey(d => d.ServiceId1);

        entity.HasOne(d => d.ServiceId2Navigation).WithMany(p =>
    →   p.OrderServiceId2Navigations)
                .HasForeignKey(d => d.ServiceId2);

        entity.HasOne(d => d.ServiceId3Navigation).WithMany(p =>
    →   p.OrderServiceId3Navigations)
                .HasForeignKey(d => d.ServiceId3);
    });

    modelBuilder.Entity<Position>(entity =>
{
    entity.ToTable("Position");

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e => e.Name).HasColumnName("name");

```

```

        entity.Property(e =>
    →   e.Requirements).HasColumnName("requirements");
        entity.Property(e =>
    →   e.Responsibilities).HasColumnName("responsibilities");
        entity.Property(e => e.Salary).HasColumnName("salary");
    });

modelBuilder.Entity<Service>(entity =>
{
    entity.ToTable("Service");

    entity.HasIndex(e => e.Id, "IX_Service_id").IsUnique();

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e => e.Cost).HasColumnName("cost");
    entity.Property(e =>
    →   e.Description).HasColumnName("description");
    entity.Property(e => e.Name).HasColumnName("name");
});

modelBuilder.Entity<Sex>(entity =>
{
    entity.ToTable("sex");

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e => e.Name).HasColumnName("name");
});

modelBuilder.Entity<Type>(entity =>
{
    entity.ToTable("Type");

    entity.Property(e => e.Id).HasColumnName("id");
    entity.Property(e =>
    →   e.Description).HasColumnName("description");
    entity.Property(e => e.Name).HasColumnName("name");
});

OnModelCreatingPartial(modelBuilder);
}

partial void OnModelCreatingPartial(ModelBuilder modelBuilder);
}

```

(f) Employee.cs

```
using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Employee : object
{
    public long Id { get; set; }

    public string LastName { get; set; } = null!;

    public string FirstName { get; set; } = null!;

    public string? SecondName { get; set; }

    public long Age { get; set; }

    public long Sex { get; set; }

    public string? Address { get; set; }

    public string PhoneNumber { get; set; } = null!;

    public string PassportDetails { get; set; } = null!;

    public long PositionId { get; set; }

    public virtual ICollection<Order> Orders { get; } = new
    {
        List<Order>();
    }

    public virtual Position Position { get; set; } = null!;

    public virtual Sex SexNavigation { get; set; } = null!;
}
```

(g) Manufacture2Component.cs

```
using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Manufacture2Component
{
```

```

public string? Name { get; set; }

public long? TypeId { get; set; }

public string? Brand { get; set; }

public long? ManufacturerId { get; set; }

public long? ManCountry { get; set; }

public string? ReleaseDate { get; set; }

public string? Characteristics { get; set; }

public string? Warranty { get; set; }

public string? Description { get; set; }

public long? Price { get; set; }
}

```

(h) Manufacturer.cs

```

using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Manufacturer
{
    public long Id { get; set; }

    public string? Name { get; set; }

    public virtual ICollection<Component> Components { get; } = new
        List<Component>();
}

```

(i) Order.cs

```

using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Order
{

```

```
public string OrderDate { get; set; } = null!;

public string DueDate { get; set; } = null!;

public long CustomerId { get; set; }

public long? ComponentId1 { get; set; }

public long? ComponentId2 { get; set; }

public long? ComponentId3 { get; set; }

public double PayProportion { get; set; }

public byte[] IsPayed { get; set; } = null!;

public byte[] IsExecute { get; set; } = null!;

public string GeneralWarranty { get; set; } = null!;

public long? ServiceId1 { get; set; }

public long? ServiceId2 { get; set; }

public long? ServiceId3 { get; set; }

public long EmployeeId { get; set; }

public virtual Component? ComponentId1Navigation { get; set; }

public virtual Component? ComponentId2Navigation { get; set; }

public virtual Component? ComponentId3Navigation { get; set; }

public virtual Customer Customer { get; set; } = null!;

public virtual Employee Employee { get; set; } = null!;

public virtual Service? ServiceId1Navigation { get; set; }

public virtual Service? ServiceId2Navigation { get; set; }

public virtual Service? ServiceId3Navigation { get; set; }
```

```
}
```

(j) Position.cs

```
using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Position
{
    public long Id { get; set; }

    public string Name { get; set; } = null!;

    public long Salary { get; set; }

    public string? Responsibilities { get; set; }

    public string? Requirements { get; set; }

    public virtual ICollection<Employee> Employees { get; } = new
    ← List<Employee>();
}
```

(k) Service.cs

```
using System;
using System.Collections.Generic;

namespace DbContext;

public partial class Service
{
    public long Id { get; set; }

    public string Name { get; set; } = null!;

    public string? Description { get; set; }

    public long Cost { get; set; }

    public virtual ICollection<Order> OrderServiceId1Navigations { get;
    ← } = new List<Order>();

    public virtual ICollection<Order> OrderServiceId2Navigations { get;
    ← } = new List<Order>();
}
```

```
    public virtual ICollection<Order> OrderServiceId3Navigations { get;  
        } = new List<Order>();  
    }  
}
```

(l) Sex.cs

```
using System;  
using System.Collections.Generic;  
  
namespace DbContext;  
  
public partial class Sex  
{  
    public long Id { get; set; }  
  
    public string Name { get; set; } = null!;  
  
    public virtual ICollection<Employee> Employees { get; } = new  
        List<Employee>();  
}
```

(m) Type.cs

```
using System;  
using System.Collections.Generic;  
  
namespace DbContext;  
  
public partial class Type  
{  
    public long Id { get; set; }  
  
    public string Name { get; set; } = null!;  
  
    public string? Description { get; set; }  
  
    public virtual ICollection<Component> Components { get; } = new  
        List<Component>();  
}
```

3. Основное приложение, которое работает с DbClasses

(a) Константы (Constants.cs)

```
using System;  
using System.Collections.Generic;  
using System.Linq;
```

```

using System.Windows.Controls;
using DbContext;
using Microsoft.EntityFrameworkCore;

namespace SuperDBApp;

public static class Constants
{
    public static DbContext DbDataContext = new();

    /*public static Dictionary<string, string> NameToTableName = new()
    {
        {"Сотрудники", "Employees"},
        {"Компоненты", "Components"},
        {"Должности", "Positions"},
        {"Заказы", "Order"},
        {"Сервисы", "Services"},
        {"Заказчики и их заказы", "CustomerOrders"},
        {"Производитель и компоненты", "Manufacture2Components"}
    };*/
}

public static Frame Frame = new();
public static MainWindowViewModel MainWindowViewModel = new();
}

```

(b) Стили (App)

- App.xaml

```

<Application x:Class="SuperDBApp.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             StartupUri="MainWindow.xaml">

    <Application.Resources>
        <Style TargetType="TextBlock">
            <Setter Property="FontFamily" Value="Roboto" />
            <Setter Property="FontSize" Value="14" />
            <Setter Property="Foreground" Value="#191C1B" />
        </Style>
        <Style TargetType="ToggleButton">
            <Setter Property="BorderThickness" Value="0" />
        <Style.Resources>
            <Style TargetType="Border">
                <Setter Property="Border.CornerRadius"
                       Value="16,16, 16, 16" />
            </Style>
        </Style.Resources>
    </Style>
</Application.Resources>

```

```

        </Style>
    </Style.Resources>
    <Style.Triggers>
        <Trigger Property="IsChecked" Value="True">
            <Setter Property="Background" Value="#CEE8E2" />
            <Setter Property="Foreground" Value="#081F1C" />
        </Trigger>
        <Trigger Property="IsChecked" Value="False">
            <Setter Property="Background" Value="Transparent" />
            <Setter Property="Foreground" Value="#404946" />
        </Trigger>
    </Style.Triggers>
</Style>
</Application.Resources>
</Application>

```

- App.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Threading.Tasks;
using System.Windows;

```

```
namespace SuperDBApp;
```

```

/// <summary>
/// Interaction logic for App.xaml
/// </summary>
public partial class App : Application
{
}

```

(c) Главное окно (MainWindow)

- MainWindow.xaml

```

<Window x:Class="SuperDBApp.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"

```

```

Title="Крутая компания" Height="500" Width="800"
MinHeight="500"
MinWidth="800" Background="#FBFDFB">
<Grid>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="2*" />
        <ColumnDefinition Width="5*" />
        <ColumnDefinition Width="2*" />

    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="20" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <ComboBox SelectionChanged="Box_OnSelected" Name="Box">
        <!-- SelectedIndex="0"-->
        <TextBlock>Сотрудники</TextBlock>
        <TextBlock>Компоненты</TextBlock>
        <TextBlock>Должности</TextBlock>
        <TextBlock>Заказы</TextBlock>
        <TextBlock>Сервисы</TextBlock>
        <TextBlock>Заказчики</TextBlock>
        <TextBlock>Заказчики и их заказы</TextBlock>
        <TextBlock>Производитель и компоненты</TextBlock>
    </ComboBox>
    <TextBlock Grid.Row="0" Grid.Column="1" FontFamily="Roboto">
        <!-- TextAlignment="Center" VerticalAlignment="Center" -->
        <!-- FontSize="20" -->
        Крутая компания©
    </TextBlock>
    <Rectangle Grid.Row="1" Fill="#DBE5E1" />
    <StackPanel Grid.Row="1" Name="NavigationView" Margin="12">
        <!-- HorizontalAlignment="Stretch" Orientation="Vertical" -->
        <ToggleButton Click="ToAddTable" IsChecked="{Binding NavButtonsChecked[0]}" Height="32">Добавить в
            таблицу</ToggleButton>
        <ToggleButton Click="ToViewTable" IsChecked="{Binding NavButtonsChecked[1]}" Margin="0, 10, 0, 0">
            Height="32">
            Просмотр таблицы
        </ToggleButton>
        <ToggleButton Click="ToQueryOne" IsChecked="{Binding NavButtonsChecked[2]}" Margin="0, 10, 0, 0">

```

```

    Height="32">
Запрос 1.
</ToggleButton>
<ToggleButton Click="ToQueryTwo" IsChecked="{Binding
→ NavButtonsChecked[3]}" Margin="0, 10, 0, 0"
    Height="32">
Запрос 2.
</ToggleButton>
<ToggleButton Click="ToQueryThree" IsChecked="{Binding
→ NavButtonsChecked[4]}" Margin="0, 10, 0, 0"
    Height="32">
Запрос 3.
</ToggleButton>
</StackPanel>
<Frame NavigationUIVisibility="Hidden" Margin="0,10,0,0"
→ Grid.Column="1" Grid.Row="1" Grid.ColumnSpan="2"
    Name="SFrame" />
</Grid>
</Window>

```

- MainWindow.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace SuperDBApp;

/// <summary>
/// Interaction logic for MainWindow.xaml
/// </summary>
public partial class MainWindow : Window
{

```

```

private readonly MainWindowViewModel _viewModel =
    Constants.MainWindowViewModel;

public MainWindow()
{
    InitializeComponent();
    Constants.Frame = SFrame;
    DataContext = _viewModel;
    _viewModel.ChangeToView(Box.Text);
}

private void ToAddTable(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToAddTable(Box.Text);
}

private void ToViewTable(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToView(Box.Text);
}

private void Box_OnSelected(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToView(((TextBlock) ((ComboBox)
    sender).SelectedItem).Text);
}

private void ToQueryOne(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToQueryOne();
}

private void ToQueryTwo(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToQueryTwo();
}

private void ToQueryThree(object sender, RoutedEventArgs e)
{
    _viewModel.ChangeToQueryThree();
}

```

- MainWindowViewModel.cs

```

using System;
using System.ComponentModel;
using System.Diagnostics;
using System.Dynamic;
using System.Windows.Controls;

namespace SuperDBApp;

public class MainWindowViewModel : INotifyPropertyChanged
{
    public event PropertyChangedEventHandler? PropertyChanged;

    protected void NotifyPropertyChanged(string propertyName)
    {
        if (PropertyChanged != null)
            PropertyChanged(this, new
                PropertyChangedEventArgs(propertyName));
    }

    public bool ReturnButtonVis { get; set; } = false;
    private bool[] _navButtonsChecked = {false, true, false, false,
    → false};
    public bool[] NavButtonsChecked => _navButtonsChecked;

    public void ChangeToView(string comboBoxSelected)
    {
        Console.WriteLine(comboBoxSelected);
        _navButtonsChecked = new[] {false, true, false, false, false};
        NotifyPropertyChanged("NavButtonsChecked");
        if (Constants.Frame.CanGoBack ||
    → Constants.Frame.CanGoForward) Constants.Frame.RemoveBackEntry();
        Constants.Frame.Navigate(new TableView(comboBoxSelected));
    }

    public void ChangeToAddTable(string comboBoxSelected)
    {
        _navButtonsChecked = new[] {true, false, false, false, false};
        NotifyPropertyChanged("NavButtonsChecked");
        if (Constants.Frame.CanGoBack ||
    → Constants.Frame.CanGoForward) Constants.Frame.RemoveBackEntry();

        switch (comboBoxSelected)

```

```

    {
        case "Сотрудники":
            Constants.Frame.Navigate(new AddEmployee());
            break;
        case "Компоненты":
            Constants.Frame.Navigate(new AddComponent());
            break;
        default:
            ChangeToView(comboBoxSelected);
            break;
    }
}

public void ChangeToQueryOne()
{
    _navButtonsChecked = new[] {false, false, true, false, false};
    NotifyPropertyChanged("NavButtonsChecked");

    if (Constants.Frame.CanGoBack ||
        Constants.Frame.CanGoForward) Constants.Frame.RemoveBackEntry();
    Constants.Frame.Navigate(new QueryOne());
}
}

public void ChangeToQueryTwo()
{
    _navButtonsChecked = new[] {false, false, false, true, false};
    NotifyPropertyChanged("NavButtonsChecked");
    if (Constants.Frame.CanGoBack ||
        Constants.Frame.CanGoForward) Constants.Frame.RemoveBackEntry();
    Constants.Frame.Navigate(new QueryTwo());
}
}

public void ChangeToQueryThree()
{
    _navButtonsChecked = new[] {false, false, false, false, true};
    NotifyPropertyChanged("NavButtonsChecked");
    if (Constants.Frame.CanGoBack ||
        Constants.Frame.CanGoForward) Constants.Frame.RemoveBackEntry();
    Constants.Frame.Navigate(new QueryThree());
}
}

```

(d) Просмотр таблиц (TableView)

- TableView.xaml

```

<Page x:Class="SuperDBApp.TableView"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      ↵  n"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility"
      ↵  y/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      mc:Ignorable="d"
      Title="TableView">
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="20*" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <DataGrid IsReadOnly="True" Name="DataG"
              ↵  AutoGenerateColumns="True"
              CanUserAddRows="False" AutoGeneratedColumns="DataG_"
              ↵  OnAutoGeneratedColumns"
              AutoGeneratingColumn="TheDataGrid_OnAutoGeneratingC"
              ↵  olumn" ItemsSource="{Binding
              ↵  Data}">
        <DataGrid.Resources>
        </DataGrid.Resources>
    </DataGrid>
    <Button Click="Delete_OnClick" Name="DeleteBTN"
            ↵  HorizontalAlignment="Center" Grid.Row="1">Удалить</Button>
</Grid>
</Page>

```

- TableView.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Controls.Primitives;
using System.Windows.Data;
using DbContext;

```

```

namespace SuperDBApp;
```

```

public partial class TableView : Page
```

```

{

    private string tableName;
    private List<DataGridColumn> _toDelete = new();




    public TableView()
    {
        InitializeComponent();
    }

    /**
     * Отображение таблицы
     */
    public TableView(string tableName = "") : this()
    {
        this.tableName = tableName;
        /*data = Constants.DbDataContext.GetType().GetProperty(Constants.NameToTableName[tableName]!)
        .GetValue(Constants.DbDataContext);*/

        t = dbSetType.MakeGenericType(data.GetType().GetProperty("Entity").GetValue(data).GetType());
        */

        // Я люблю C#. (нет)
        switch (tableName)
        {
            case "Сотрудники":
                DataG.ItemsSource =
                    Constants.DbDataContext.Employees.ToList();
                break;
            case "Компоненты":
                DataG.ItemsSource =
                    Constants.DbDataContext.Components.ToList();
                break;
            case "Должности":
                DataG.ItemsSource =
                    Constants.DbDataContext.Positions.ToList();
                break;
            case "Заказы":
                DataG.ItemsSource =
                    Constants.DbDataContext.Orders.ToList();
                break;
        }
    }
}

```

```

        break;
    case "Сервисы":
        DataG.ItemsSource =
    → Constants.DbDataContext.Services.ToList();
        break;
    case "Заказчики":
        DataG.ItemsSource =
    → Constants.DbDataContext.Customers.ToList();
        break;
    case "Заказчики и их заказы":
        DataG.ItemsSource =
    → Constants.DbDataContext.CustomerOrders.ToList();
        DeleteBTN.Visibility = Visibility.Collapsed;
        break;
    case "Производитель и компоненты":
        DataG.ItemsSource =
    → Constants.DbDataContext.Manufacture2Components.ToList();
        DeleteBTN.Visibility = Visibility.Collapsed;
        break;
    }
}

private void TheDataGridView_AutoGeneratingColumn(object? sender,
    → DataGridViewAutoGeneratingColumnEventArgs e)
{
    Console.WriteLine();
    if (e.PropertyType.AssemblyQualifiedName!.Contains("System.Collections.Generic"))
    → ||
    → e.PropertyType.AssemblyQualifiedName!.Contains("DbContext"))
        _ToDelete.Add(e.Column);
}

private void OnEdit(object? sender, EventArgs e)
{
    switch (tableName)
    {
        case "Сотрудники":
            Constants.Frame.Navigate(
                new
    → AddEmployee(Constants.DbDataContext.Employees.Find((long)
    → ((Button) sender!).Tag)!));

```

```

                break;
            case "Компоненты":
                Constants.Frame.Navigate(
                    new
                AddComponent(Constants.DbDataContext.Components.Find((long)
                ((Button) sender!).Tag)!));
                break;
            }
        }

    private void DataG_OnAutoGeneratedColumns(object? sender,
    EventArgs e)
{
    foreach (var gridColumn in _ToDelete)
    DataG.Columns.Remove(gridColumn);

    if (tableName == "Сотрудники" || tableName == "Компоненты")
    {
        var gridViewColTem = new DataGridTemplateColumn();
        DataTemplate dataTemplate = new();
        var factory1 = new
        FrameworkElementFactory(typeof(Button));
        factory1.SetValue(ContentControl.ContentProperty,
        "Изменить");
        factory1.AddHandler(ButtonBase.ClickEvent, new
        RoutedEventHandler(OnEdit));
        Binding binding = new("Id");
        binding.Mode = BindingMode.Default;
        factory1.SetValue(TagProperty, binding);
        dataTemplate.VisualTree = factory1;
        gridViewColTem.CellTemplate = dataTemplate;
        DataG.Columns.Add(gridViewColTem);
    }
}

private void Delete_OnClick(object sender, RoutedEventArgs e)
{
    if (DataG.SelectedItems.Count == 0)
    {
        MessageBox.Show("Для удаления выделите хотя бы один
        элемент");
    }
    else

```

```

{
    if (MessageBox.Show($"Вы точно хотите удалить
→ {DataG.SelectedItems.Count} элементов?", "",

        MessageBoxButtons.YesNo) == MessageBoxResult.No)

        return;

    switch (tableName)
    {

        case "Сотрудники":
            foreach (var employee in DataG.SelectedItems)
                Constants.DbDataContext.Employees.Remove((Employee)
→ employee)
→ employee);

            break;

        case "Компоненты":
            foreach (var component in DataG.SelectedItems)
                Constants.DbDataContext.Components.Remove((Component)
→ component)
→ component);

            break;

        case "Должности":
            foreach (var position in DataG.SelectedItems)
                Constants.DbDataContext.Positions.Remove((Position)
→ position)
→ position);

            break;

        case "Заказы":
            foreach (var order in DataG.SelectedItems)
                Constants.DbDataContext.Orders.Remove((Order)
→ order);
→ order);

            break;

        case "Сервисы":
            foreach (var service in DataG.SelectedItems)
                Constants.DbDataContext.Services.Remove((Service)
→ service)
→ service);

            break;

        case "Заказчики":
            foreach (var customer in DataG.SelectedItems)
                Constants.DbDataContext.Customers.Remove((Customer)
→ customer)
→ customer);

            break;
    }
}

```

```

        Constants.DbDataContext.SaveChanges();
        Constants.MainWindowViewModel.ChangeToView(tableName);
    }
}
}

```

(e) Добавление компонента (AddComponent)

- AddComponent.xaml

```

<Page x:Class="SuperDBApp.AddComponent"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:n="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      mc:Ignorable="d"
      Title="AddComponent">
<Grid Margin="10" HorizontalAlignment="Center"
      VerticalAlignment="Stretch">
<StackPanel VerticalAlignment="Stretch">
<StackPanel>
    <TextBlock>Тип:</TextBlock>
    <ComboBox Name="TypeCombo" ItemsSource="{Binding
        Types}" SelectionChanged="TypeSelected" />
</StackPanel>
<StackPanel>
    <TextBlock>Бренд:</TextBlock>
    <TextBox Text="{Binding NewComponent.Brand}" />
</StackPanel>
<StackPanel>
    <TextBlock>Производитель:</TextBlock>
    <ComboBox Name="ManuCombo" ItemsSource="{Binding
        Manufacturers}"
        SelectionChanged="ManufactureSelected" />
</StackPanel>
<StackPanel>
    <TextBlock>Страна-производитель:</TextBlock>
    <ComboBox Name="CountryCombo"
        ItemsSource="{Binding Countries}"
        SelectionChanged="CountrySelected" />
</StackPanel>

```

```

<StackPanel>
    <TextBlock>Дата релиза:</TextBlock>
    <Calendar Name="CalendarIk"
        ↳ SelectedDatesChanged="DateChanged"
        SelectedDate="{Binding Date}"
        ↳ DisplayDate="{Binding Date}" />
</StackPanel>
<StackPanel>
    <TextBlock>Характеристики:</TextBlock>
    <TextBox Text="{Binding
        ↳ NewComponent.Characteristics}" />
</StackPanel>
<StackPanel>
    <TextBlock>Гарантия:</TextBlock>
    <TextBox Text="{Binding NewComponent.Warranty}" />
</StackPanel>
<StackPanel>
    <TextBlock>Описание:</TextBlock>
    <TextBox Text="{Binding
        ↳ NewComponent.Description}" />
</StackPanel>
<StackPanel>
    <TextBlock>Цена:</TextBlock>
    <TextBox Text="{Binding NewComponent.Price}" />
</StackPanel>
<Button Name="SubmitButton" Click="Add"
    ↳ Margin="20">Добавить</Button>
</StackPanel>
</ScrollViewer>
</Grid>
</Page>

```

- AddComponent.xaml.cs

```

using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using DbContext;

namespace SuperDBApp;

public partial class AddComponent : Page
{
    private AddComponentViewModel _addComponentViewModel = new();

```

```

public AddComponent()
{
    InitializeComponent();
    DataContext = _addComponentViewModel;
}

public AddComponent(Component component) : this()
{
    _addComponentViewModel = new AddComponentViewModel(component);
    DataContext = _addComponentViewModel;
    TypeCombo.SelectedItem =
        Constants.DbDataContext.Types.First(p => component.TypeId ==
        p.Id).Name;
    ManuCombo.SelectedItem = Constants.DbDataContext.Manufacturers
        .First(manufacturer => component.ManufacturerId ==
        manufacturer.Id).Name;
    CountryCombo.SelectedItem =
        Constants.DbDataContext.Countries.First(country =>
        country.Id == component.ManCountry).Name;
    _addComponentViewModel.Date =
        DateTime.Parse(component.ReleaseDate);

    SubmitButton.Content = "Сохранить";
}

private void TypeSelected(object sender,
    SelectionChangedEventArgs e)
{
    _addComponentViewModel.SetType((string) ((ComboBox)
        sender).SelectedItem);
}

private void ManufactureSelected(object sender,
    SelectionChangedEventArgs e)
{
    _addComponentViewModel.SetMan((string) ((ComboBox)
        sender).SelectedItem);
}

private void CountrySelected(object sender,
    SelectionChangedEventArgs e)

```

```

{
    _addComponentViewModel.SetManCountry((string) ((ComboBox)
→  sender).SelectedItem);
}

private void Add(object sender, RoutedEventArgs e)
{
    _addComponentViewModel.Add();
}

private void DateChanged(object? sender,
→  SelectionChangedEventArgs e)
{
    _addComponentViewModel.SetDate((sender as
→  Calendar).SelectedDate);
}
}

```

(f) Добавление сотрудника (AddEmployee)

- AddEmployee.xaml

```

<Page x:Class="SuperDBApp.AddEmployee"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      ↵  n"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility"
      ↵  y/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      mc:Ignorable="d"
      Title="AddItem">

<Grid Margin="10" HorizontalAlignment="Center"
      ↵  VerticalAlignment="Stretch">
    <StackPanel VerticalAlignment="Stretch">
        <StackPanel>
            <TextBlock>Имя:</TextBlock>
            <TextBox Text="{Binding NewEmployee.FirstName}" />
        </StackPanel>
        <StackPanel>
            <TextBlock>Фамилия:</TextBlock>
            <TextBox Text="{Binding NewEmployee.LastName}" />
        </StackPanel>
        <StackPanel>
            <TextBlock>Отчество:</TextBlock>
            <TextBox Text="{Binding NewEmployee.SecondName}" />
        </StackPanel>
    </StackPanel>

```

```

<StackPanel>
    <TextBlock>Возраст:</TextBlock>
    <TextBox Text="{Binding NewEmployee.Age}" />
</StackPanel>
<StackPanel>
    <TextBlock>Пол:</TextBlock>
    <ComboBox Name="SexComboBox" ItemsSource="{Binding Sexes}"
              SelectionChanged="Sex_Selected" />
</StackPanel>
<StackPanel>
    <TextBlock>Адрес:</TextBlock>
    <TextBox Text="{Binding NewEmployee.Address}" />
</StackPanel>
<StackPanel>
    <TextBlock>Номер телефона:</TextBlock>
    <TextBox Text="{Binding NewEmployee.PhoneNumber}" />
</StackPanel>
<StackPanel>
    <TextBlock>Паспортные данные:</TextBlock>
    <TextBox Text="{Binding NewEmployee.PassportDetails}"
              />
</StackPanel>
<StackPanel>
    <TextBlock>Должность</TextBlock>
    <ComboBox Name="PositionComboBox"
              ItemsSource="{Binding Positions}"
              SelectionChanged="Position_Selected" />
</StackPanel>
<Button Name="SubmitButton" Click="Add"
        Margin="20">Добавить</Button>
</StackPanel>
</Grid>
</Page>

```

- AddEmployee.xaml.cs

```

using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using DbContext;

namespace SuperDBApp;

public partial class AddEmployee : Page

```

```

    {

        private AddEmployeeViewModel _addEmployeeViewModel = new();

        public AddEmployee()
        {
            InitializeComponent();
            DataContext = _addEmployeeViewModel;
        }

        public AddEmployee(Employee employee) : this()
        {
            _addEmployeeViewModel = new AddEmployeeViewModel(employee);
            DataContext = _addEmployeeViewModel;
            PositionComboBox.SelectedItem =
                Constants.DbDataContext.Positions.First(p => employee.PositionId
                == p.Id).Name;
            SexComboBox.SelectedItem =
                Constants.DbDataContext.Sexes.First(s => employee.Sex ==
                s.Id).Name;
            SubmitButton.Content = "Сохранить";
        }

        private void Sex_Selected(object sender,
        SelectionChangedEventArgs e)
        {
            _addEmployeeViewModel.SetSex((string)((ComboBox)
            sender).SelectedItem);
        }

        private void Position_Selected(object sender,
        SelectionChangedEventArgs e)
        {
            _addEmployeeViewModel.SetPosition((string)((ComboBox)
            sender).SelectedItem);
        }

        private void Add(object sender, RoutedEventArgs e)
        {
            _addEmployeeViewModel.Add();
        }
    }

```

(g) Запрос 1 (QueryOne)
Получение заказчиков и их заказов.

- QueryOne.xaml

```

<Page x:Class="SuperDBApp.QueryOne"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:n="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      mc:Ignorable="d"
      Title="QueryOne">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="20" />
            <RowDefinition Height="*" />
        </Grid.RowDefinitions>
        <ComboBox SelectionChanged="Customer_Changed"
                   Name="CustomerCombo" />
        <DataGrid Name="DataGrid" Grid.Row="1" />
    </Grid>
</Page>

```

- QueryOne.xaml.cs

```

using System.Linq;
using System.Security.Cryptography;
using System.Windows.Controls;
using Microsoft.VisualBasic;

namespace SuperDBApp;

public partial class QueryOne : Page
{
    /**
     * Отобразить заказы отдельных заказчиков;
     */
    public QueryOne()
    {
        InitializeComponent();
        CustomerCombo.ItemsSource =
            Constants.DbDataContext.Customers.Select(customer =>
                $"{customer.Id}, {customer.LastName} {customer.FirstName}
                {customer.SecondName}").ToList();
        DataGrid.ItemsSource = Constants.DbDataContext.Customers.Join(
            Constants.DbDataContext.Orders, customer =>
            customer.Id,

```

```

        order => order.CustomerId, (customer, order) => new
    {
        CustomerName = $"{customer.Id}, {customer.LastName}"
        ↪ {customer.FirstName} {customer.SecondName},
        OrderDate = order.OrderDate,
        DueDate = order.DueDate,
        Warranty = order.GeneralWarranty
    });

    })
}

private void Customer_Changed(object sender,
    ↪ SelectionChangedEventArgs e)
{
    var id = int.Parse((sender as
    ↪ ComboBox).SelectedItem.ToString()!.Split(',', 1)[0]);
    DataGrid.ItemsSource =
    ↪ Constants.DbDataContext.Orders.Where(order => order.CustomerId ==
    ↪ id).ToList();
}
}

```

(h) Запрос 2 (QueryTwo)

Получение компонентов по бренду.

- QueryTwo.xaml

```

<Page x:Class="SuperDBApp.QueryTwo"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    ↪ n"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility"
    ↪ y/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:SuperDBApp"
    mc:Ignorable="d"
    Title="QueryTwo" >
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="20" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <ComboBox SelectionChanged="ManufactureChanged"
        ↪ Name="ManufactureCombo" />
    <DataGrid Name="DataGrid" Grid.Row="1" />
</Grid>

```

```

</Page>
• QueryTwo.xaml.cs

    using System.Linq;
    using System.Windows.Controls;

namespace SuperDBApp;

public partial class QueryTwo : Page
{
    public QueryTwo()
    {
        InitializeComponent();
        ManufactureCombo.ItemsSource =
            Constants.DbDataContext.Manufacturers
                .Select(manufacturer => $"{manufacturer.Id} {manufacturer.Name}")
                .ToList();
    }

    private void ManufactureChanged(object sender,
        SelectionChangedEventArgs e)
    {
        var id = int.Parse((sender as
            ComboBox).SelectedItem.ToString()!.Split(' ')![0]);
        DataGrid.ItemsSource =
            Constants.DbDataContext.Components.Where(component =>
                component.ManufacturerId == id).ToList();
    }
}

```

(i) Запрос 3 (QueryThree)

Получение сотрудника с самой большой зарплатой.

- QueryThree.xaml

```

<Page x:Class="SuperDBApp.QueryThree"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:n="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:SuperDBApp"
      mc:Ignorable="d"
      Title="QueryTwo" >
    <Grid VerticalAlignment="Center">

```

```

        <TextBlock FontSize="20" Name="TextBlock"
        ↵   TextAlignment="Center" TextWrapping="Wrap">Самая большая
        ↵   зарплата: у:</TextBlock>
    </Grid>
</Page>
• QueryThree.xaml.cs

using System.Linq;
using System.Windows.Controls;
using DbContext;

namespace SuperDBApp;

public partial class QueryThree : Page
{
    public QueryThree()
    {
        InitializeComponent();
        Employee employee =
        ↵ Constants.DbDataContext.Employees.AsEnumerable().MaxBy(emp =>
        ↵             Constants.DbDataContext.Positions.First(position =>
        ↵                 position.Id == emp.PositionId).Salary)!;
        Position position =
        ↵ Constants.DbDataContext.Positions.First(position => position.Id
        ↵ == employee.PositionId);
        TextBlock.Text = $"Самая большая зарплата у
        ↵ \'{employee.LastName} {employee.FirstName}
        ↵ {employee.SecondName}\\" на должности \"{position.Name}\", с
        ↵ зарплатой {position.Salary} рублей";
    }
}

```

Демонстрация работы приложения:

1. Отображение таблиц

	Id	LastName	FirstName	SecondName	Age	Sex	Address	PhoneNumber	PassportDetails	PositionId	
1	1	Панков	Василий	Дмитриевич	19	1	Суховский переулок 23	+79627011087	1111 111111	1	Изменить
2	2	Гайкин	Кирилл	Денисович	19	1	Кобринская улица 5	+79118445162	2222 222222	2	Изменить
3	3	Турсунова	Лидия	Сергеевна	24	2	набережная Крутой реки 149	+71111111111	1234 567891	3	Изменить
9	9	Панков	Вася	Дмитриевич	15	1		+79627011087	1111 444444	2	Изменить

Крутая компания

Компоненты

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

	TypeId	Brand	ManufacturerId	ManCountry	ReleaseDate	Characteristics
1	1	Ryzen 5 3600x	2	2	11/10/2020	Общие параметры Модель AMD Ryzen 5 3600X Сокет AM4 Код производителя [100-000000022] Год релиза 2019 Система охлаждения в комплекте нет Термоинтерфейс в комплекте нет Ядро и архитектура Общее количество ядер 6 Максимальное число потоков

Удалить

Крутая компания

Должности

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

	Name	Salary	Responsibilities	Requirements
1	Директор	50000	Руководить	
2	Продавец	40000	Эффективно продавать товары	1. Выполнять скрипты 2. Выполнять нормы продаж 3. Уведомлять покупателей о новых акциях
3	Главный менеджер	92000	Следить за работой подчинённых	1. Оформлять кучу отчётов 2. Следить за работой подчинённых

Удалить

Крутая компания

Заказы

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

OrderDate	DueDate	CustomerId	ComponentId1	ComponentId2	ComponentId3	PayProportion	IsPaid	IsExecute	GeneralWarrant
11/10/2022	11/15/2022	1	1	1	1	1	Byte[] Array	Byte[] Array	1 год
11/10/2022	11/20/2022	2	1			1	Byte[] Array	Byte[] Array	12 лет
9/12/2022	11/30/2023	1	2			1	Byte[] Array	Byte[] Array	2 месяца

Удалить

Крутая компания

Сервисы

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

	Name	Description	Cost
1	Установка процессора		1000
2	Установка Windows 11		10000
3	Установка антивируса		10000
4	Форматирование диска		5000

Удалить

Крутая компания

Заказчики

	Id	LastName	FirstName	SecondName	Address	Phone
1	Гаму́йло	Сергей	Сергеевич	г. Санкт-Петербург	+79111111111	
2	Панков	Владимир	Дмитриевич	г. Санкт-Петербург г. Колпино	+79111112111	
3	Белоцерковский	Марат		г. Владивосток	+79111311111	

Удалить

Крутая компания

Производитель и компоненты

Name	TypeId	Brand	ManufacturerId	ManCountry	ReleaseDate	Characteristics
AMD	1	Ryzen 5 3600x	2	2	11/10/2020	Общие параметры Модель AMD Ryzen 5 3600X Сокет AM4 Код производителя [100-000000022] Год релиза 2019 Система охлаждения в комплекте нет Термоинтерфейс в комплекте нет Ядро и архитектура Общее количество ядер 6 Максимальное число потоков 12

2. Добавление сотрудника

- Добавление без ошибок

Крутая компания

Сотрудники

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Имя:

Фамилия:

Отчество:

Возраст: 0

Пол:

Адрес:

Номер телефона:

Паспортные данные:

Должность:

Добавить

Крутая компания

Сотрудники

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Имя: Виктор

Фамилия: Виктор

Отчество: Виктор

Возраст: 0

Пол: Мужской

Адрес: а

Номер телефона: +79627011087

Паспортные данные: 1111 111111

Должность: Продавец

Добавить

Крутая компания

Сотрудники

Добавить в таблицу

Просмотр таблицы

Запрос 1.

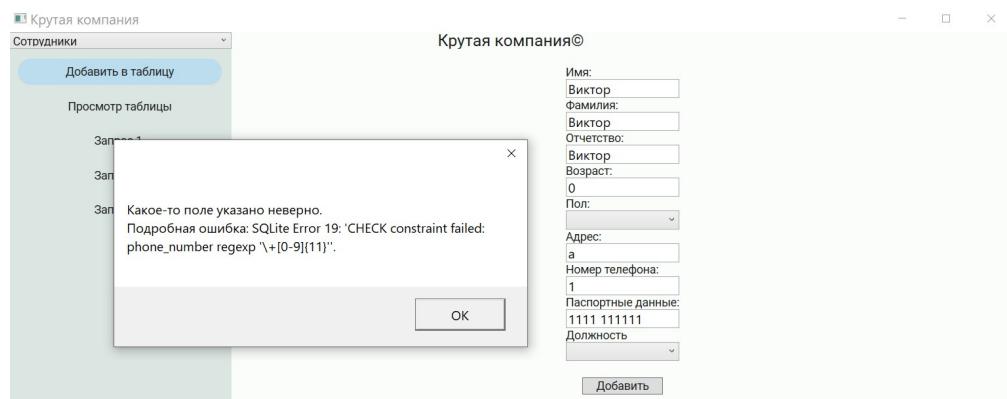
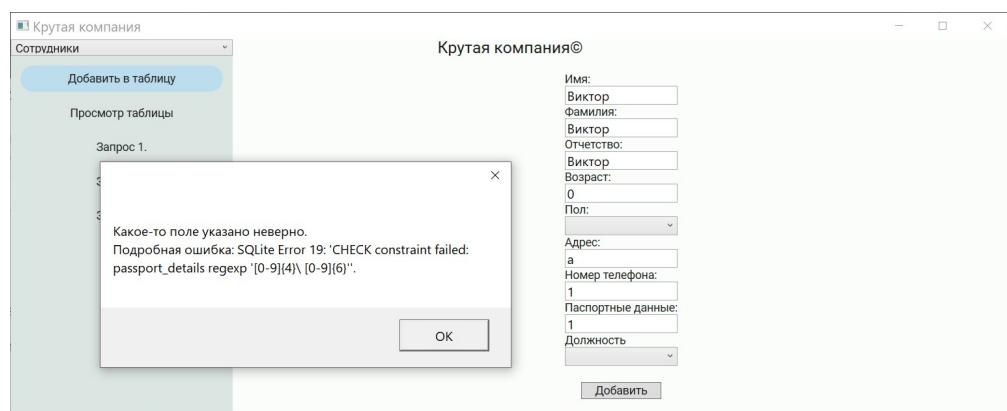
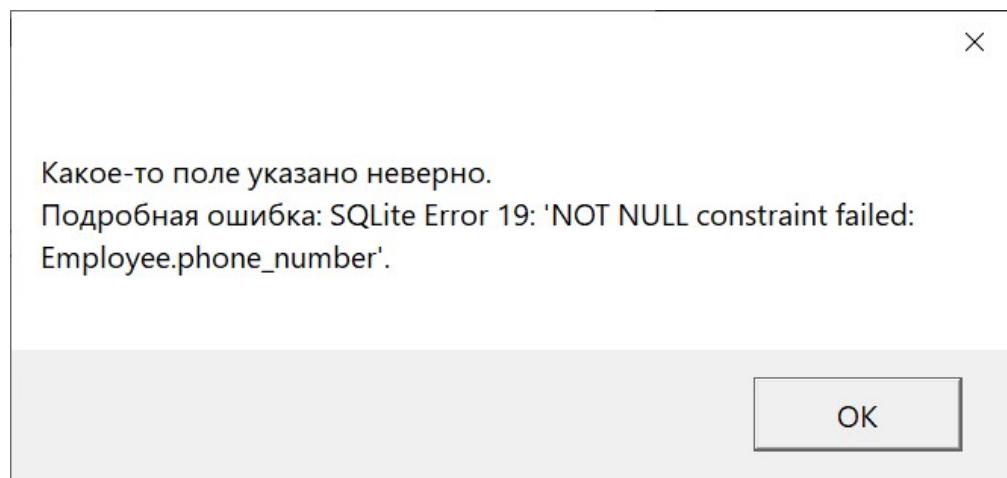
Запрос 2.

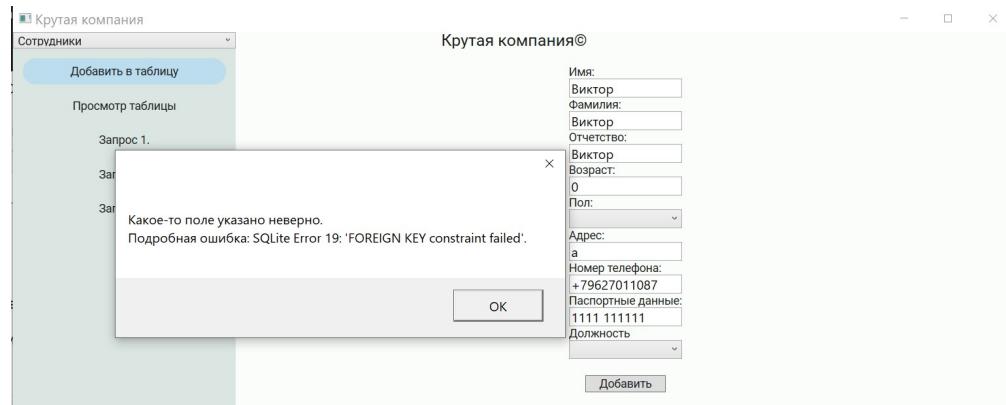
Запрос 3.

1	Панков	Василий	Дмитриевич	19	1	Суховский переулок 23	+79627011087	1111 111111	1	Изменить	
2	Гайкин	Кирилл	Денисович	19	1	Кобринская улица 5	+79118445162	2222 222222	2	Изменить	
3	Турсунова	Лидия	Сергеевна	24	2	набережная Круты реки 149	+71111111111	1234 567891	3	Изменить	
9	Панков	Вася	Дмитриевич	15	1		+79627011087	1111 444444	2	Изменить	
10	Виктор	Виктор	Виктор	0	1	а	+79627011087	1111 111111	2	Изменить	

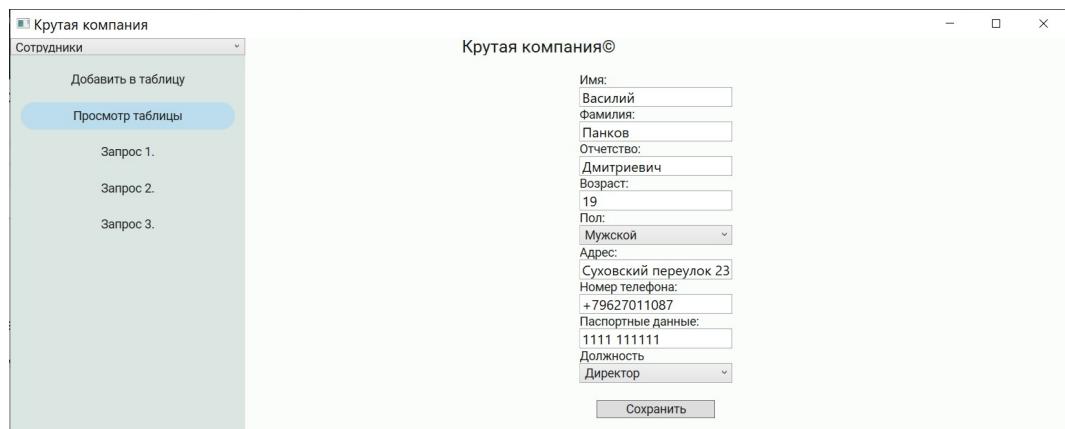
Удалить

- Возможные ошибки

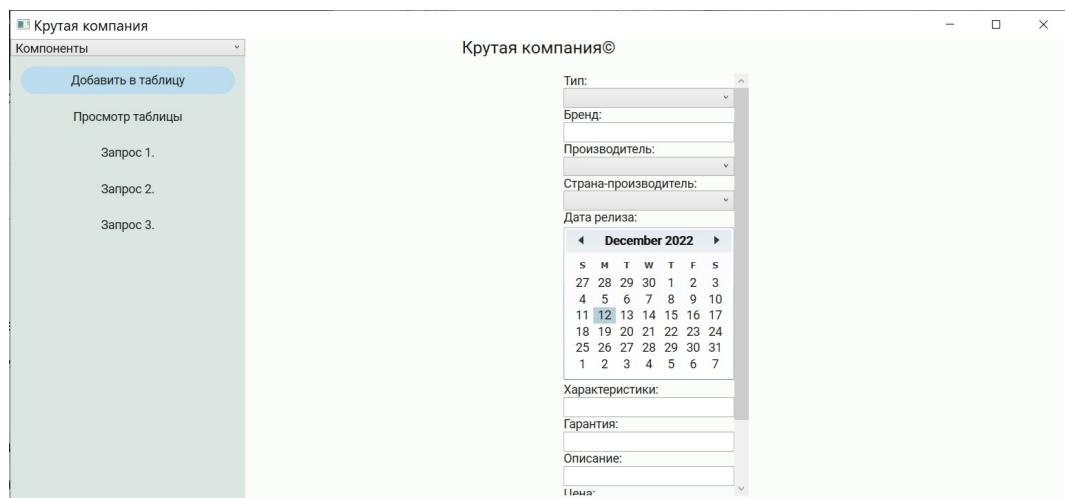


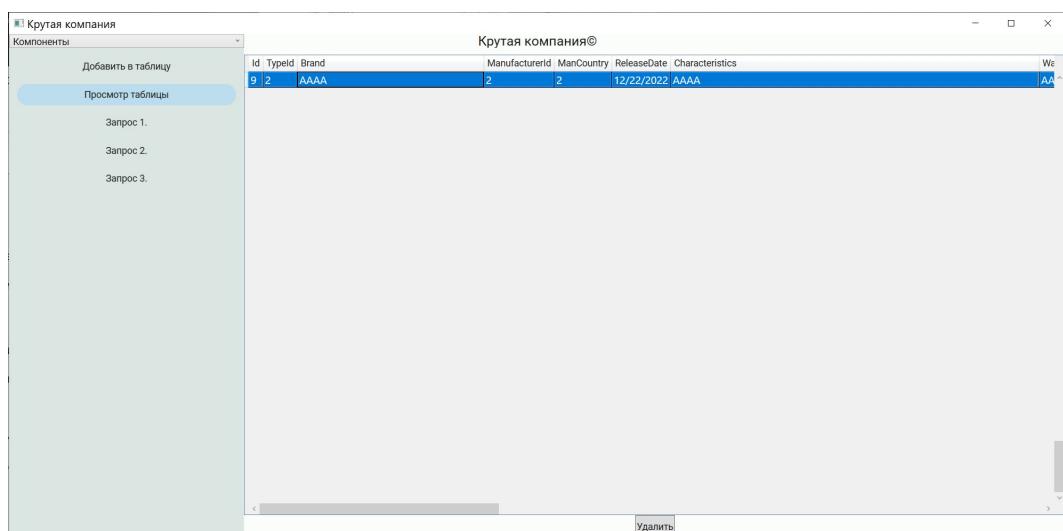
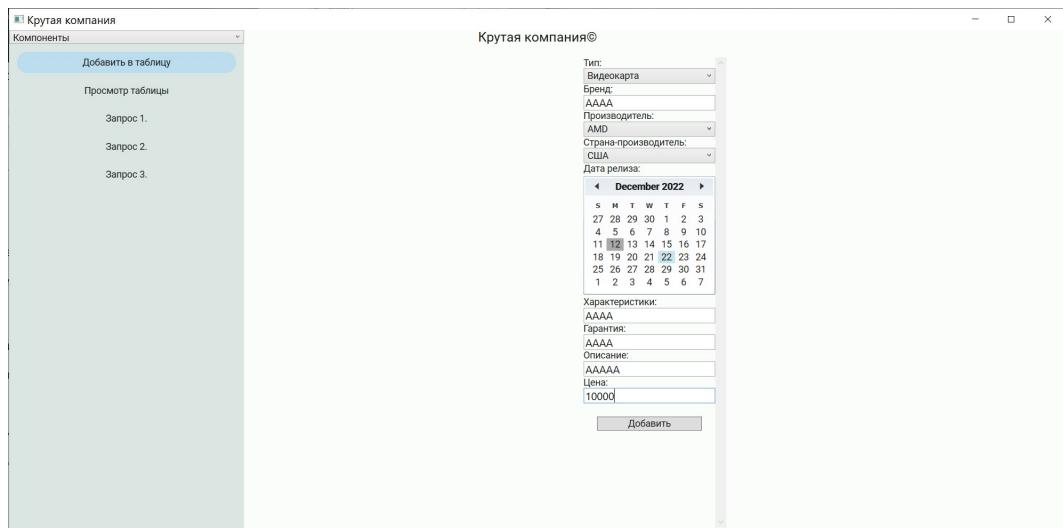


3. Изменение сотрудника

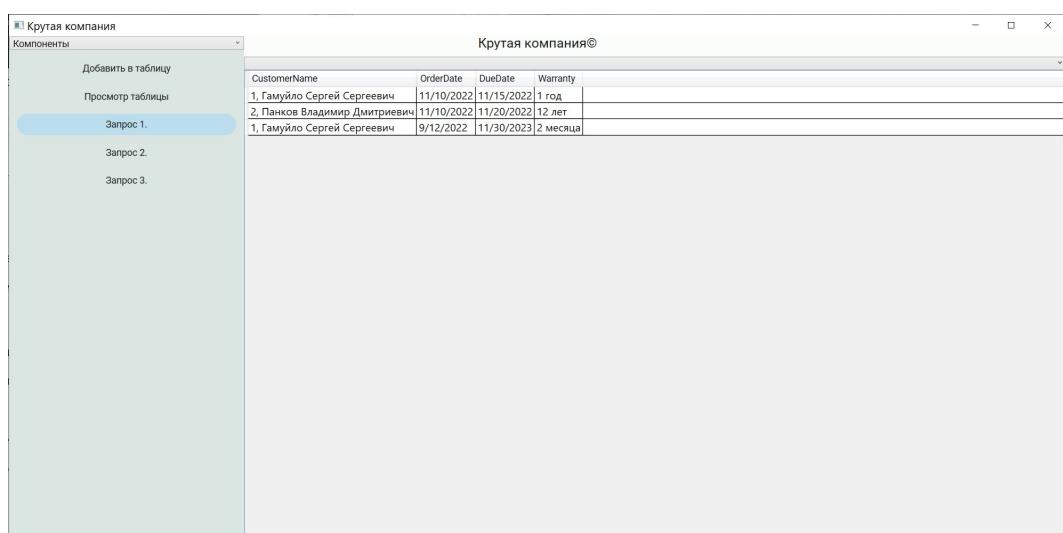


4. Добавление компонента





5. Запрос 1



Крутая компания

Компоненты

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Крутая компания©

1. Гамулю Сергей Сергеевич

OrderDate	DueDate	Customerid	ComponentId1	ComponentId2	ComponentId3	PayProportion	IsPaid	IsExecute	GeneralWarranty	ServiceId1	ServiceId2	ServiceId3	Emplo
11/10/2022	11/15/2022	1	1	1	1	1	Byte[] Array	Byte[] Array	1 год	1	1	1	1
9/12/2022	11/30/2023	1	2			1	Byte[] Array	Byte[] Array	2 месяца				3

6. Запрос 2

Крутая компания

Компоненты

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Крутая компания©

2 AMD

Id	TypeId	Brand	ManufacturerId	ManCountry	ReleaseDate	Characteristics	Warranty	Description
1	1	Ryzen 5 3600x	2	2	11/10/2020	Общие параметры Модель AMD Ryzen 5 3600X Сокет AM4 Код производителя [100-00000002] Год релиза 2019 Система охлаждения в комплекте нет Термointерфейс в комплекте нет Ядро и архитектура Общее количество ядер 6 Максимальное число потоков 12 Количество производительных ядер 6 Количество энергоэффективных ядер нет Объем кэша L2 3 МБ Объем кэша L3 32 МБ Техпроцесс TSMC 7FF Ядро AMD Matisse Частота и возможность разгона	1 год	Хороший процессор для игр и для рабо

7. Запрос 3

Крутая компания

Компоненты

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Крутая компания©

Самая большая зарплата у "Панков Василий Дмитриевич" на должности "Директор", с зарплатой 500000 рублей

3 Лабораторная работа № 3

3.1 Часть 1

Тема: Создание модели БД и работа с физической БД в MySQL Workbench.

Цель работы: Получение навыков с работы с инструментом MySQL Workbench.

Задание №19: БД Компьютерной фирмы.

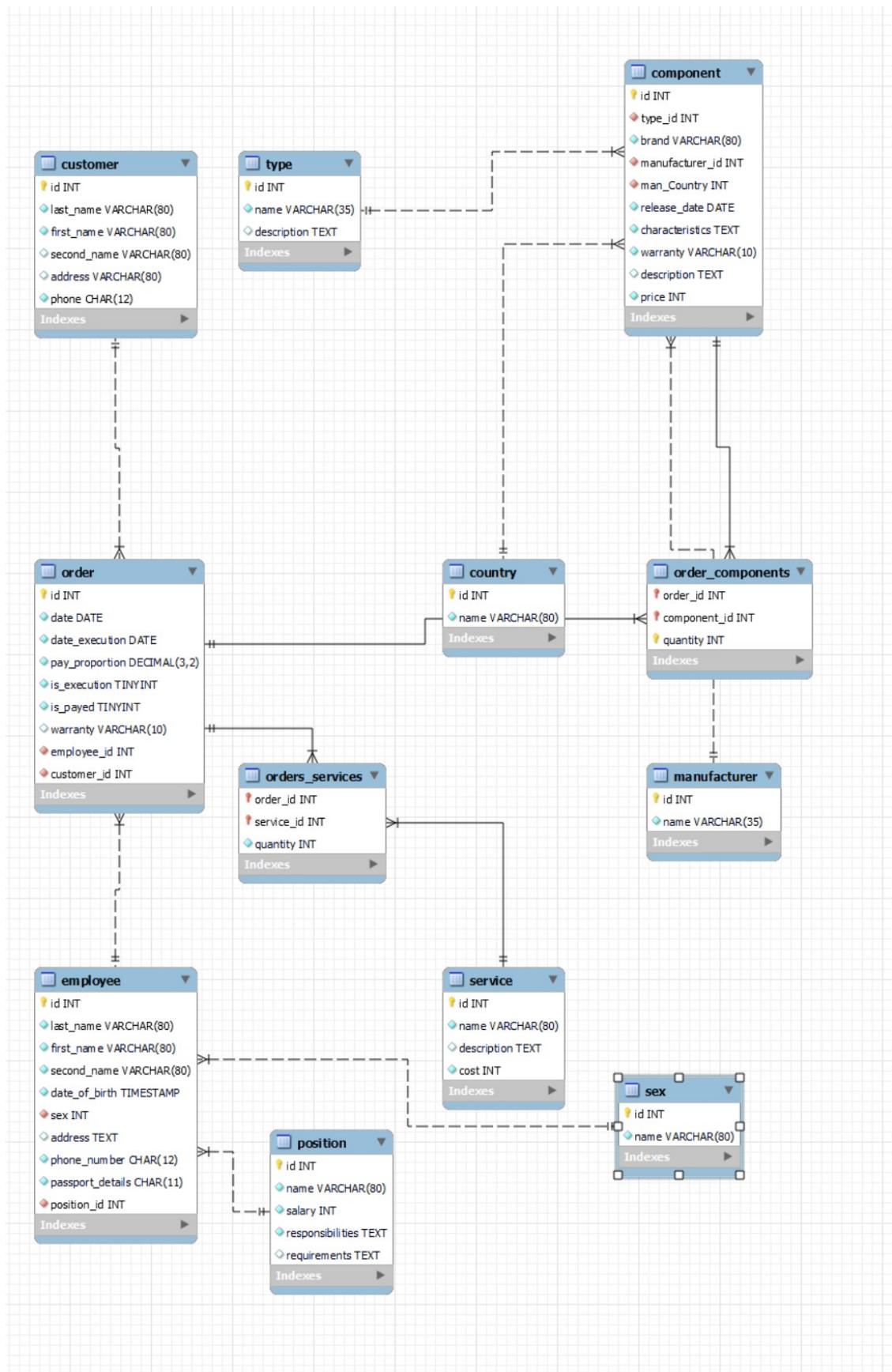
Таблицы:

1. Сотрудники (Код сотрудника, ФИО, Возраст, Пол, Адрес, Телефон, Паспортные данные, Код должности).
 1. Должности (Код должности, Наименование должности, Оклад, Обязанности, Требования).
 2. Виды комплектующих (Код вида, Наименование, Описание).
 3. Комплектующие (Код комплектующего, Код вида, Марка, Фирма производитель, Страна производитель, Дата выпуска, Характеристики, Срок гарантия, Описание, Цена).
1. Заказчики (Код заказчика, ФИО, Адрес, Телефон).
2. Услуги (Код услуги, Наименование, Описание, Стоимость).
3. Заказы (Дата заказа, Дата исполнения, Код заказчика, Код комплектующего 1, Код комплектующего 2, Код комплектующего 3, Доля предоплаты, Отметка об оплате, Отметка об исполнении, Общая стоимость, Срок общей гарантии, Код услуги 1, Код услуги 2, Код услуги 3, Код сотрудника).

Запросы:

1. Отобразить заказы отдельных заказчиков;
2. Отобразить комплектующие определенного производителя.

Диаграмма созданных таблиц:



Заполненные таблицы:

1. Sex (Пол)

	id	name
1	1	Мужской
2	2	Женский
3	3	Неопределённый
4	4	Боевой вертолёт
5	5	Старый динозавр

2. Position (Должность)

	id	name	salary	responsibilities	requirements
1	1	Директор	500000	Руководить	<null>
2	2	Продавец	40000	Эффективно продавать товары	1. Выполнять скрипты=2. Выполнять нормы продаж=3. Уведомлять покупателей о продаже
3	3	Главный менеджер	92000	Следить за работой подчинённых	1. Оформлять кучу отчётов=2. Следить за работой подчинённых
4	5	Стажёр	10000	Учиться работать	Должен работать под надзором сотрудника-наставника.
5	6	Младший менеджер	30000	Бумажная работа	Оформляет товары и занимается большей бумажной работой чем главный менеджер

3. Employee (Сотрудник)

	id	last_name	first_name	second_name	date_of_birth	sex	address	phone_number	passport_details	position_id
1	1	Ланков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Сухойский переулок 23	+79627811887	1111 111111	1
2	2	Тайкин	Кирill	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222 222222	2
3	3	Турсунова	Лидия	Сергееvna	1988-11-06 12:11:18	1	набережная Кругой реки 149	+71111111111	1234 567891	3
4	9	Ланков	Вася	Дмитриевич	2006-12-16 12:11:32	1 <null>		+79627811887	1111 444444	2
5	10	Виктор	Виктор		2004-07-02 12:11:41	1 a		+79627811887	1111 111111	2

4. Type (Тип компонента)

	id	name	description
1	1	Процессор	<null>
2	2	Видеокарта	<null>
3	3	HDD	<null>
4	4	SSD	<null>
5	5	Корпус	<null>
6	6	RGB подсветка	<null>
7	7	Мышка	<null>
8	8	Клавиатура	<null>
9	9	Геймпад	<null>
10	10	Принтер	<null>
11	11	Ноутбук	<null>

5. Manufacturer (Производители)

	 id	 name
1	1	ASUS
2	2	AMD
3	3	INTEL
4	4	NVIDIA
5	5	Google
6	6	Xiaomi
7	7	Huawei
8	8	HP
9	9	Palit
10	10	Seagate
11	11	Kingston
12	12	Zalman

6. Country (Страна)

	id	name
1	1	Россия
2	2	США
3	3	Китай
4	4	Франция
5	5	Австралия

7. Component (Компоненты)

	id	type_id	brand	manufacturer_id	nan_Country	release_date	characteris...	warranty	description	price
1	1	1	Ryzen 5 3600x	2	2	2022-12-14	Общие параметры: Модел...	1 год	Хороший процессор для м...	15000
2	2	2	ROG-STRIX-RX6700XT-012G-GAMING	1	3	2018-12-13	Графический процессор.	1 год	Видеокарта нового покол...	69000
3	3	3	IronWolf ST4000VN008	10	4	2020-12-18	<Форм-фактор: 3.5 " * 06...	1 год	Готовый к работе устройс...	10000
4	4	4	A400 S4608537/480G	11	5	2019-12-12	Объем накопителя: 480 Гб...	3 года	Накопитель SSD KINGSTON ...	2800
5	5	5	15	12	3	2022-06-10	Конфигурация корпуса: ...	1 год	о корпус ATX ZALMAN 15 и...	5500
6	6	3	Test 2	1	2	2017-07-21	XnJ	ddsa	dasd	0
7	7	1	19	3	2	2022-08-11	Кругой	1 мес.	<null>	0
8	8	2	AAAA	2	2	2020-12-22	AAAA	AAAA	AAAAA	10000

8. Customer (Покупатель)

	id	last_name	first_name	second_name	address	phone
1	1	Гаму́йло	Сергей	Сергеевич	г. Санкт-Петербург	+79111111111
2	2	Ланков	Владимир	Дмитриевич	г. Санкт-Петербург г. Колпино	+79111112111
3	3	Белоцерковский	Марат	<null>	<null>	+79999999999
4	4	Куриличев	Михаил	<null>	<null>	+12345678901
5	5	Бобров	Фёдор	Николаевич	<null>	+79856789909

9. Service (Сервисы)

	id	name	description	cost
1	1	Установка процессора	<null>	1000
2	2	Установка Windows 11	<null>	10000
3	3	Установка антивируса	<null>	10000
4	4	Форматирование диска	<null>	5000
5	5	Оптимизация вашего ПК	<null>	1000000

10. Order (Заказы)

	id	date	date_execution	pay_proportion	is_execution	is_paid	warranty	employee_id	customer_id
1	1	2022-12-08	2022-12-23	1.00	0	1	12 лет	1	1
2	2	2022-12-01	2022-12-24	0.50	0	1	1 год	1	1
3	3	2021-12-09	2023-12-01	0.30	1	1	2 дня	2	4
4	4	2022-12-10	2022-12-31	1.00	0	1	8 мес	2	2
5	6	2022-12-07	2022-12-29	1.00	0	1	нет	10	5

11. order_components (Компоненты по заказам)

	order_id	component_id	quantity
1	1	1	1
2	3	1	1
3	2	2	1
4	2	3	1
5	4	4	4

12. order_services (Сервисы по заказам)

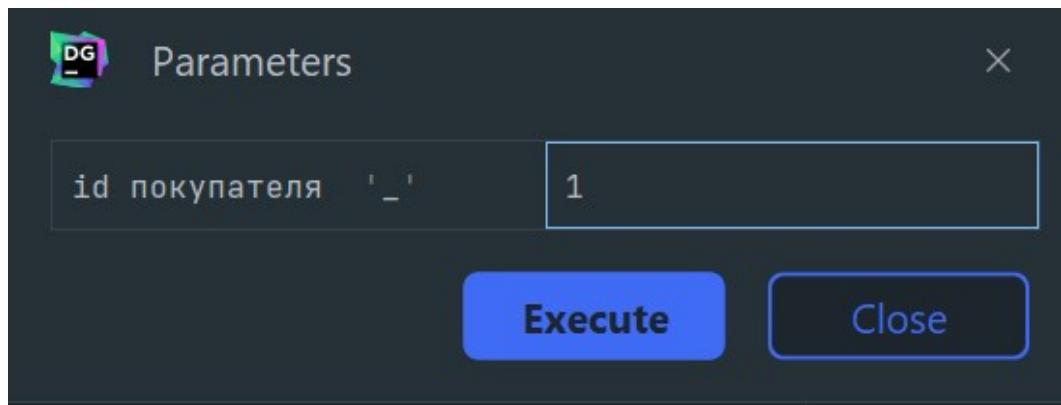
	order_id	service_id	quantity
1	1	1	1
2	1	2	1
3	4	1	1
4	3	3	3
5	2	3	4

Запросы:

1. Отобразить заказы отдельных заказчиков;

```

SELECT customer_id,
       last_name,
       first_name,
       second_name,
       `order`.id as 'order_id',
       date,
       date_execution,
       pay_proportion,
       is_execution,
       is_payed,
       warranty,
       employee_id
FROM `order`
      JOIN customer c on c.id = '${id покупателя}' AND `order`.customer_id
      ↵   = c.id;
  
```

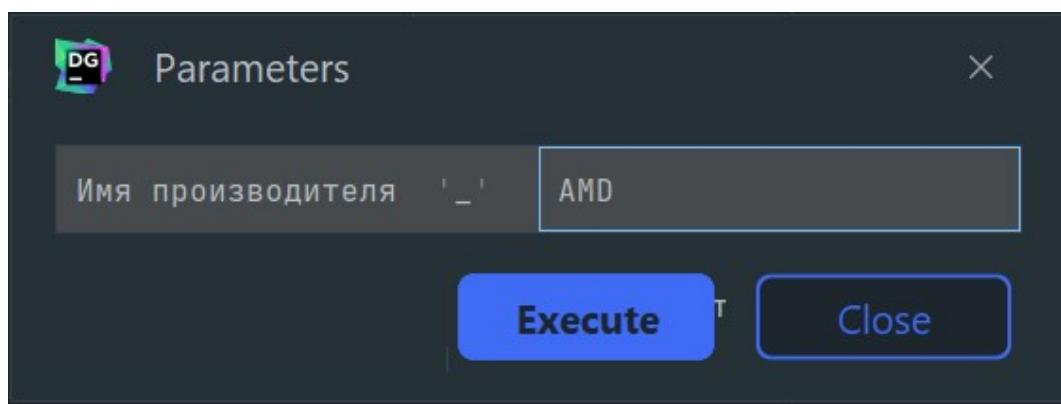


	customer_id	last_name	first_name	second_name	order_id	date	date_execution	pay_proportion	is_execution	is_paid	warranty	employee_id
1	1	Ганыло	Сергей	Сергеевич	1	2022-12-08	2022-12-25	1.00	0	1	12 мес.	1
2	1	Ганыло	Сергей	Сергеевич	2	2022-12-01	2022-12-24	0.50	0	1	1 год	2

2. Отобразить комплектующие определенного производителя.

```

SELECT name as 'Имя производителя',
       component.id,
       type_id,
       brand,
       manufacturer_id,
       man_Country,
       release_date,
       characteristics,
       warranty,
       description,
       price
FROM component
      JOIN manufacturer m on m.name = '${Имя производителя}' AND
      ↳ component.manufacturer_id = m.id;
    
```



	Имя производителя	id	type_id	brand	manufacturer_id	man_Country	release_date	characteristics	warranty	description	price
1	AMD	1	1	Ryzen 5 3600x	2	2	2022-12-14	Общие параметры/Модель/AMD Ryzen 5 3600X/Сокет... 1 год	Хороший процессор для игр и для работы	15000	
2	AMD	8	2	AAAA	2	2	2028-12-22	AAAA	AAAA	18000	

3.2 Часть 2

Тема: работа с данными в формате json. СерIALIZАЦИЯ и дЕСЕРИАЛИЗАЦИЯ объектов.

Цель работы: получение практических навыков при работе с форматом json.

Задание 1. СерIALIZАЦИЯ

- Представьте данные таблицы Auths в формате json(используйте библиотеку Newtonsoft.Json)
- Представьте данные одной из таблиц индивидуального задания в формате json(используйте библиотеку Newtonsoft.Json)

Код:

1. Класс Auth

```
namespace ExampleJson;

public record Auth(int AuId, string AuFname, string AuLname, string Phone,
    string City, string State, int Zip);
```

2. Окно MainWindow

(a) MainWindow.xaml

```
<Window x:Class="ExampleJson.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility"
    mc:Ignorable="d"
    Title="MainWindow" Height="450" Width="800">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="10*"/>
            <RowDefinition Height="1*"/>
        </Grid.RowDefinitions>
        <TextBox TextWrapping="Wrap" Name="RichTextBox" Margin="10"
            IsReadOnly="True"></TextBox>
        <Button Grid.Row="1" Click="ButtonBase_OnClick"
            VerticalAlignment="Center"
            HorizontalAlignment="Center">Показать JSON</Button>
    </Grid>
</Window>
```

(b) MainWindow.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using System.Threading.Tasks;
using System.Windows;
```

```

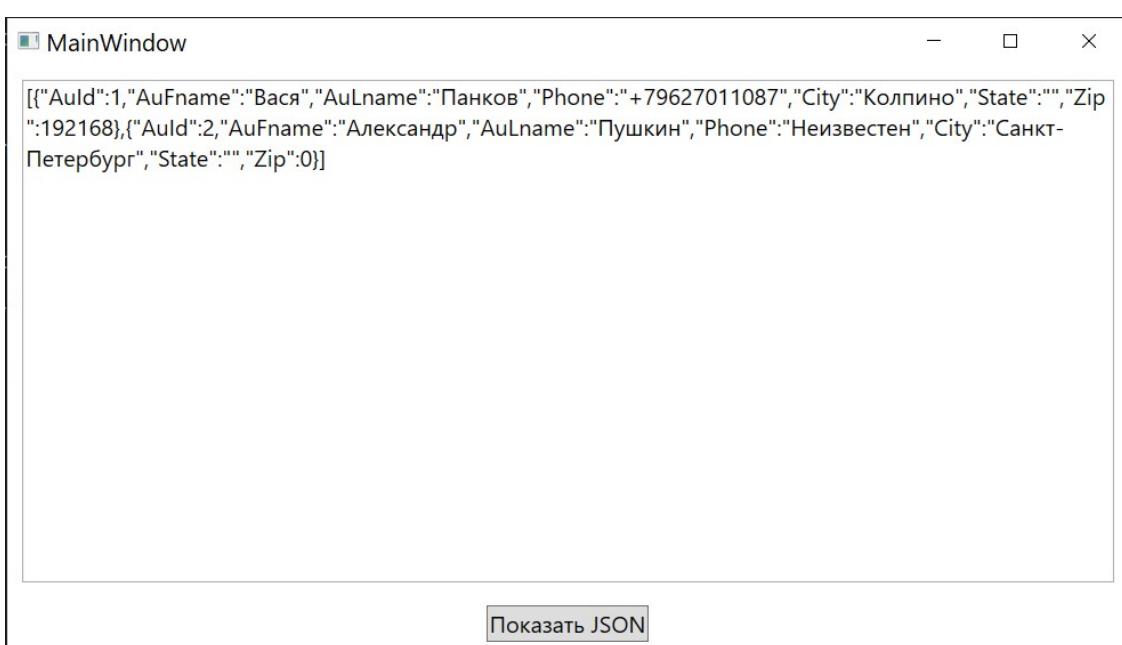
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using Newtonsoft.Json;

namespace ExampleJson
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void ButtonBase_OnClick(object sender, RoutedEventArgs e)
        {
            List<Auth> auths = new()
            {
                new Auth(1, "Вася", "Панков", "+79627011087", "Колпино",
→ "", 192168),
                new Auth(2, "Александр", "Пушкин", "Неизвестен",
→ "Санкт-Петербург", "", 0)
            };
            RichTextBox.Text = JsonConvert.SerializeObject(auths);
        }
    }
}

```

Демонстрация работы приложения:



Задание 2. Десериализация

Используя различные ресурсы в сети, десериализовать данные, полученные в формате json. Самостоятельно найти бесплатный ресурс, который может предоставить различные данные в формате json(сведения о погоде, странах, фильмах, валютах и т.д.).

Вспомогательные классы

Класс WebRequest представляет запрос информации для отправки по определенному URI. Идентификатор URI передается в качестве параметра методу Create().

КлассWebResponse представляет данные, извлекаемые из сервера. Вызов метода WebRequest.GetResponse() приводит к отправке запроса веб-серверу и к созданию объектаWebResponse для просмотра возвращенных данных.

Был использован <https://cataas.com/>, который выдаёт случайную фотографию кота.

1. Получил Json на <https://cataas.com/cat?json=true>

```
{"tags":["caracal","kitten","floppa","ears","gif"],"createdAt":"2022-05-01T20
↳ :51:36.618Z","updatedAt":"2022-10-11T07:52:32.699Z","validated":true,"own
↳ er":null,"file":"626ef2d97f254a0017b564ef.gif","mimetype":"image/gif","_
↳ size":2352578,"_id":"SbbeZwoC81vSTzBX","url":"/cat/SbbeZwoC81vSTzBX"}
```

2. На основе его Rider создал класс:

```
namespace WebJsonExample;

public class Cat
{
    public string[] tags { get; set; }
    public string createdAt { get; set; }
    public string updatedAt { get; set; }
    public bool validated { get; set; }
    public string owner { get; set; }
    public string file { get; set; }
    public string mimetype { get; set; }
    public int? size { get; set; }
    public string?_id { get; set; }
    public string url { get; set; }
}
```

3. Создал WPF-приложение представленное ниже

Код:

- MainWindows.xaml

```
<Window x:Class="WebJsonExample.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:WebJsonExample"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800">
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="*"/>
        <RowDefinition Height="*"/>
        <RowDefinition Height="5*"/>
        <RowDefinition Height="*"/>
        <RowDefinition Height="5*"/>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
```

```

        <ColumnDefinition Width="*"></ColumnDefinition>
        <ColumnDefinition Width="*"></ColumnDefinition>
    </Grid.ColumnDefinitions>
    <Button Click="ButtonBase_OnClick" Grid.ColumnSpan="2"
        → HorizontalAlignment="Center" VerticalAlignment="Center"
        → >Отправить запрос</Button>
    <TextBlock Grid.Row="1" VerticalAlignment="Bottom"
        → TextAlignment="Center">Полученный JSON</TextBlock>
    <TextBlock Grid.Column="1" Grid.Row="1" VerticalAlignment="Bottom"
        → TextAlignment="Center">Полученная информация</TextBlock>
    <TextBox TextWrapping="Wrap" Name="JsonBox" Grid.Row="2"
        → Margin="10"></TextBox>
    <TextBox Name="JsonInfoBox" Grid.Row="2" Grid.Column="2"
        → Margin="10"></TextBox>
    <TextBlock VerticalAlignment="Center" FontSize="20" Grid.Row="3"
        → Grid.ColumnSpan="2" TextAlignment="Center"> Картинка</TextBlock>
    <Image Grid.ColumnSpan="2" Name="Image" Grid.Row="4"></Image>
</Grid>
</Window>

```

- MainWindow.xaml.cs

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace WebJsonExample
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>

```

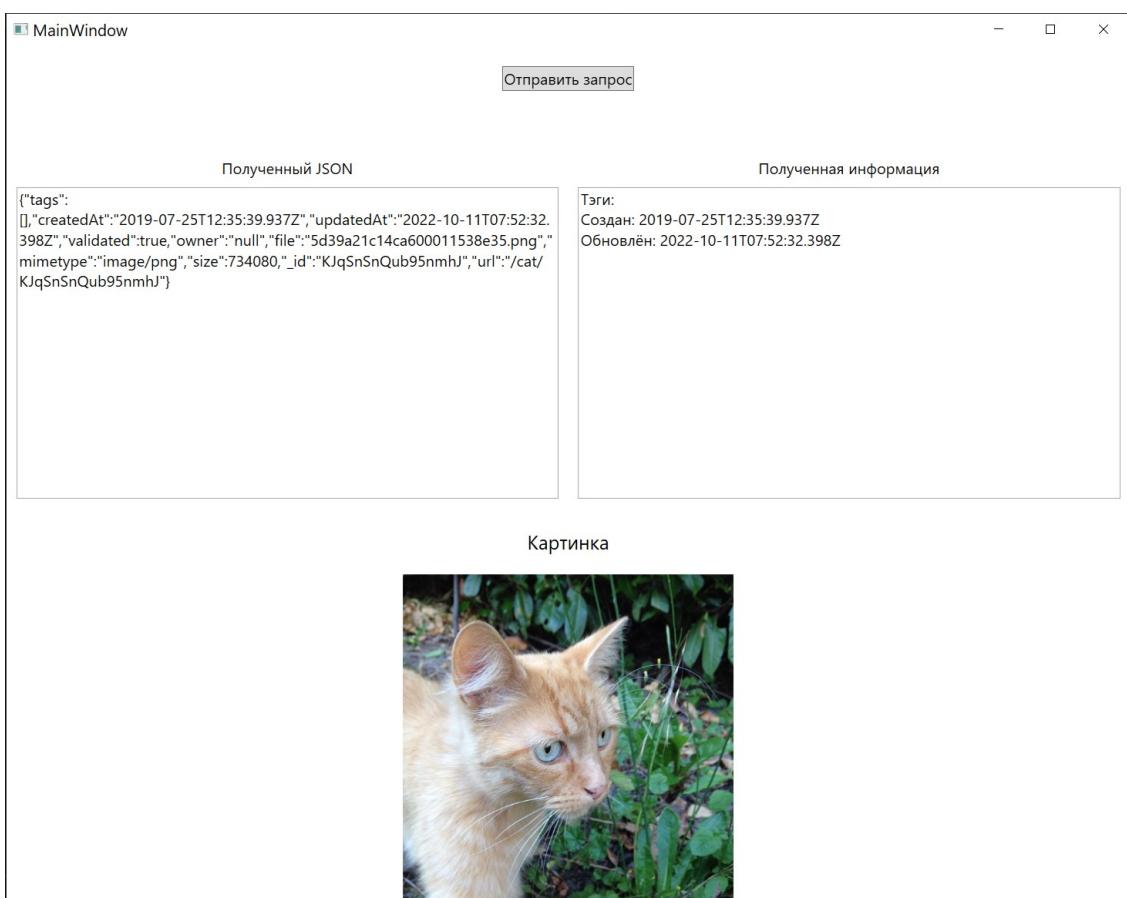
```

public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
    }

    private void ButtonBase_OnClick(object sender, RoutedEventArgs e)
    {
        var response =
            WebRequest.Create("https://cataas.com/cat?json=true").GetResponse();
        Stream dataStream = response.GetResponseStream();
        // Open the stream using a StreamReader for easy access.
        StreamReader reader = new StreamReader(dataStream);
        // Read the content.
        string jsonText = reader.ReadToEnd();
        JsonBox.Text = jsonText;
        Cat? cat =
            Newtonsoft.Json.JsonConvert.DeserializeObject(jsonText, typeof(Cat)) as
        Cat;
        JsonInfoBox.Text = $"Тэги: {String.Join(", ", cat!.tags)}";
    }
}

```

Демонстрация работы приложения:



MainWindow

Полученный JSON

```
{"tags": ["computer"], "createdAt": "2016-11-30T09:32:46.377Z", "updatedAt": "2022-10-11T07:52:32.189Z", "validated": true, "owner": null, "file": "595f2809557291a9750ebf43.jpeg", "mimetype": "image/jpeg", "size": 269527, "_id": "aOkTbVU5PTWrzwah", "url": "/cat/aOkTbVU5PTWrzwah"}
```

Отправить запрос

Полученная информация

Тэги: computer
Создан: 2016-11-30T09:32:46.377Z
Обновлён: 2022-10-11T07:52:32.189Z

Картинка



MainWindow

Полученный JSON

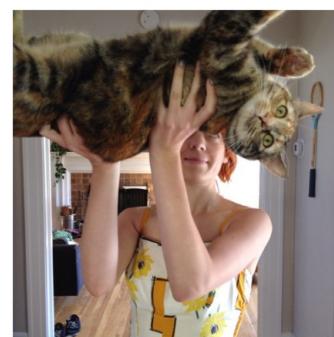
```
{"tags": [], "createdAt": "2019-07-25T12:46:31.715Z", "updatedAt": "2022-10-11T07:52:32.419Z", "validated": true, "owner": null, "file": "5d39a4a714ca600011538e77.png", "mimetype": "image/png", "size": 617816, "_id": "gz6fb8va4tyU7Lpr", "url": "/cat/gz6fb8va4tyU7Lpr"}
```

Отправить запрос

Полученная информация

Тэги:
Создан: 2019-07-25T12:46:31.715Z
Обновлён: 2022-10-11T07:52:32.419Z

Картинка



Вывод: Я научился работать JSON и с базой данной MySql.

3.3 Дополнительное задание

1. Выполнить экспорт данных в формате csv, json и xml из трех таблиц (выбрать любые 3 таблицы из индивидуального задания). В отчете представить скриншоты с содержимым экспортованных файлов.

Таблица сотрудников (employee):

id	last_name	first_name	second_name	date_of_birth	sex	address	pho
1	Панков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Суховский переулок 23	+79
2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79
3	Турсунова	Лидия	Сергеевна	1980-11-06 12:11:18	2	набережная Крутой реки 149	+71
9	Панков	Вася	Дмитриевич	2006-12-16 12:11:32	1	NULL	+79
10	Виктор	Виктор	Виктор	2004-07-02 12:11:41	1	а	+79

Полученный экспорт в CSV:

Таблица стран (country):

id	name
1	Россия
2	США
3	Китай
4	Франция
5	Австралия

Полученный экспорт в JSON:

```
[  
  {  
    "id": 1,  
    "name": "Россия"  
  },  
  {  
    "id": 2,  
    "name": "США"  
  },  
  {  
    "id": 3,  
    "name": "Китай"  
  },  
  {  
    "id": 4,  
    "name": "Франция"  
  },  
  {  
    "id": 5,  
    "name": "Австралия"  
  }  
]
```

Таблица должностей (position):

id	name	salary	responsibilities	requirements
1	Директор	500000	Руководить	NULL
2	Продавец	40000	Эффективно продавать товары	1. Выполнять скрипты 2. Выполнять нормы продаж
3	Главный менеджер	92000	Следить за работой подчинённых	1. Оформлять кучу отчётов 2. Следить за работой подчинённых
5	Стажёр	10000	Учиться работать	Должен работать под надзором сотрудников
6	Младший менеджер	30000	Бумажная работа	Оформляет товары и занимается бюджетированием

Полученный экспорт в XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<data>

<row>
    <id>1</id>
    <name>Директор</name>
    <salary>500000</salary>
    <responsibilities>Руководить</responsibilities>
    <requirements>null</requirements>
</row>

<row>
    <id>2</id>
    <name>Продавец</name>
    <salary>40000</salary>
    <responsibilities>Эффективно продавать товары</responsibilities>
    <requirements>1. Выполнять скрипты
2. Выполнять нормы продаж
3. Уведомлять покупателей о новых акциях</requirements>
</row>

<row>
    <id>3</id>
```

```

<name>Главный менеджер</name>

<salary>92000</salary>

<responsibilities>Следить за работой подчинённых</responsibilities>

<requirements>1. Оформлять кучу отчётов
2. Следить за работой подчинённых</requirements>

</row>

<row>

<id>5</id>

<name>Стажёр</name>

<salary>10000</salary>

<responsibilities>Учиться работать</responsibilities>

<requirements>Должен работать под надзором сотрудника-наставника.</requirements>

</row>

<row>

<id>6</id>

<name>Младший менеджер</name>

<salary>30000</salary>

<responsibilities>Бумажная работа</responsibilities>

<requirements>Оформляет товары и занимается большей бумажной работой чем
↪ главный менеджер.</requirements>

</row>
</data>

```

2. Поработать со всеми строковыми функциями, разобранными на лекции. В качестве параметра в строковых функциях использовать Фамилию, имя, отчество студента. В отчете представить скрин-

шоты с результатом работы функций в MySQL.

Текст запроса:

```
SELECT CONCAT_WS(' ', 'Меня', 'зовут:', CONCAT(UPPER('п'), LOWER('АНКОВ'))),
→  'Вася,', 'кстати', 'длина', 'моего',
   'имени',
   CONCAT('', LENGTH('Вася'), ',', 'а', 'буква', 'Я', 'находится', 'на',
→  LOCATE('я', 'Вася'), 'позиции.',
   'Давайте', 'поиграемся', 'с', 'моей', 'фамилией:',
   LOWER(CONCAT(RIGHT('Панков', 4), SUBSTRING('Панков', 2, 2)),
→  SUBSTRING_INDEX('Панков', 'н', 1),
   LEFT('Панков', 3), '.')), 'Наверное с Кириллом вы
→  незнакомы',
   'но его имя отлично преобразуется в', CONCAT(REPLACE('Кирилл', 'рилл',
→  'лька'), ','),
   'а ещё классно пишется задом наперёд:', REVERSE('Кирилл')) AS 'RESULT';
```

Полученный ответ:

RESULT

Меня зовут: Панков Вася, кстати длина моего имени: 8, а буква Я находится на 4 позиции.

Давайте поиграемся с моей фамилией: нкованпапан.

Наверное с Кириллом вы незнакомы, но его имя отлично преобразуется в Килька, а ещё классно пишется задом наперёд.

3. Поработать с функциями из Таблицы 1. В качестве параметра выбирается дата рождения студента. В отчете представить скриншоты с результатом работы функций в MySQL.

Таблица 1: Пример работы некоторых функций

Функция	Результат
DAYOFMONTH('2018-05-25')	25
DAYOFWEEK('2018-05-25')	6
DAYOFYEAR('2018-05-25')	145
MONTH('2018-05-25')	5
YEAR('2018-05-25')	2018
QUARTER('2018-05-25')	2
WEEK('2018-05-25', 1)	21
LAST_DAY('2018-05-25')	2018-05-31
DAYNAME('2018-05-25')	Friday
MONTHNAME('2018-05-25')	May

Текст запроса:

```
SELECT CONCAT_WS(' ', 'А вы знали, что я родился', DAYOFMONTH(birthday),
→  MONTHNAME(birthday), YEAR(birthday),
   'года, в', CONCAT('', DAYNAME(birthday), '.'), 'Это кстати',
→  DAYOFYEAR(birthday), 'день и', WEEK(birthday), 'неделя в году.\n',
   'А в серьёзной бухгалтерии это', QUARTER(birthday), 'квартал года.', 'Эх,
→  что вам ещё рассказать, что я родился в',
   MONTH(birthday), 'месяц по счёту или то что последний день этого месяца',
```

```

CONCAT(' ', DAYOFMONTH(LAST_DAY(birthday)), '.'),
'Видимо придётся завершить рассказ.') AS RESULT FROM (SELECT '2003-11-06'
↪ AS birthday) AS birthday;

```

Полученный ответ:

RESULT

А вы знали, что я родился 6 November 2003 года, в Thursday. Видимо придётся завершить рассказ. Это кстати 310 день и 44 неделя в году. А в серьёзной бухгалтерии это 4 квартал года. Эх, что вам ещё рассказать, что я родился в 11 месяц по счёту или то что последний день этого месяца 30.

4. Поработать с функциями DATE_ADD, DATE_SUB, DATEDIFF. Один из параметров функции - дата рождения студента. В отчете представить скриншоты с результатом работы функций в MySQL.

Текст запросов:

```

SELECT DATE_ADD('2003-11-06', INTERVAL 2 DAY) AS 'День рождения моей прабабушки';

SELECT YEAR(DATE_SUB('2003-11-06', INTERVAL 24 YEAR)) AS 'Год рождения моих
↪ родителей';

SELECT DATEDIFF('2009-10-20', '2003-11-06') AS 'РАЗНИЦА В ДНЯХ МЕЖДУ РОЖДЕНИЕМ
↪ МНОЙ И МОИМ БРАТОМ';

```

Полученные ответы:

День рождения моей прабабушки
2003-11-08

Год рождения моих родителей
1979

РАЗНИЦА В ДНЯХ МЕЖДУ РОЖДЕНИЕМ МНОЙ И МОИМ БРАТОМ
2175

5. Для одной из таблиц индивидуального задания, где имеется дата рождения(сотрудника, клиента и т.д), выполнить запрос, в котором рядом с датой рождения(тип date) будет столбец с датой рождения с типом timestamp. В отчете представить скриншоты с результатом работы функций в MySQL.

Текст запросов:

```

SELECT date_of_birth AS TIMESTAMP, CAST(date_of_birth as DATE) AS DATE FROM
↪ employee;

```

Полученный ответ:

TIMESTAMP	DATE
2003-12-15 12:10:59	2003-12-15
1999-07-17 12:11:08	1999-07-17
1980-11-06 12:11:18	1980-11-06
2006-12-16 12:11:32	2006-12-16
2004-07-02 12:11:41	2004-07-02