

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

подпись, дата

инициалы, фамилия

Санкт-Петербург 2023

СОДЕРЖАНИЕ

1 Лабораторная работа № 1 «Взаимодействие с базами данных в MS Visual Studio».....	2
2 Лабораторная работа № 2	28
2.1 Часть 1	28
2.2 Часть 2 (Индивидуальное задание)	39
3 Лабораторная работа № 3	84
3.1 Часть 1	84
3.2 Часть 2	91
3.3 Дополнительное задание	98
4 Лабораторная работа № 4 «Создание запросов в MySQL Workbench.»	105
4.1 Индивидуальное задание	114
5 Лабораторная работа № 5 «Процедуры и функции в MySQL»	121
5.1 Самостоятельная работа	125

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(№gr) 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 дополнитель-

ную 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" />
        </DockPanel>
        <DockPanel Margin="10" HorizontalAlignment="Stretch">
            <TextBlock>Пароль</TextBlock>
            <PasswordBox Name="Password" Margin="20,0,10,0"
                ↳  HorizontalAlignment="Stretch" />
        </DockPanel>
        <DockPanel Name="SecPasswordPanel" Visibility="Collapsed"
            ↳  Margin="10" HorizontalAlignment="Stretch">
            <TextBlock>Пароль 2</TextBlock>
            <PasswordBox Name="SecondPassword" Margin="10,0,10,0"
                ↳  HorizontalAlignment="Stretch" />
        </DockPanel>
        <Button Name="AuthBtn" Click="Auth" Margin="10" Content="Вход" />
        <Button Click="ToReg" Name="SwitchBtn" Margin="10"
            ↳  Content="Перейти к регистрации" />
    </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 №gr =
        @group";

    private const string YearQuery =
        "SELECT Familia, №gr FROM Students WHERE Year([Datarogr]) = @year AND
        Familia LIKE ?";

    private const string OutOfTownQuery =
        "SELECT Familia, №gr 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\labs\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.Add("?", 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.Add("?", 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.Add("?", 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 = true;
    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 = \'г. Колпино\', Budget = 1
→ WHERE POL = \'M\' 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,
→ DataGridCellEditEndingEventArgs 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,InsertCellInWorksheet,-(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
}

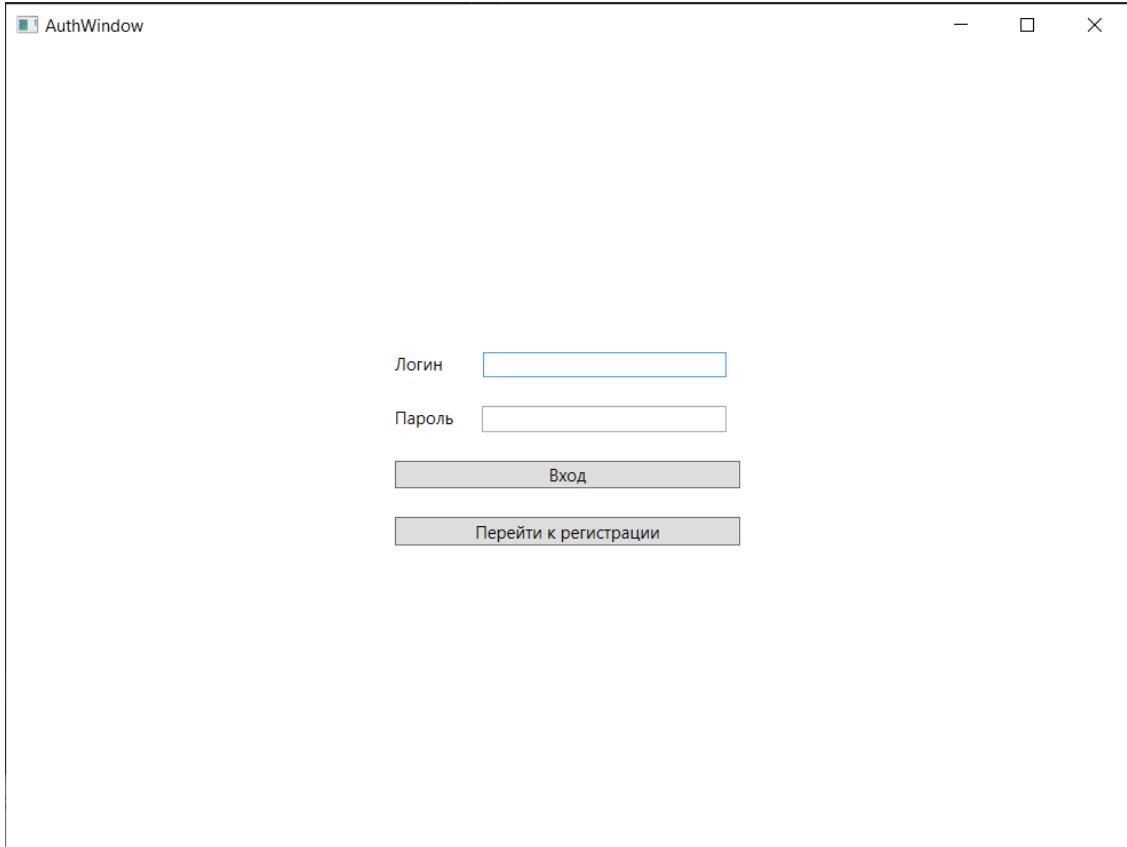
public enum Role
{
    User = 1,
    Admin = 2
}

public static class Constants
{

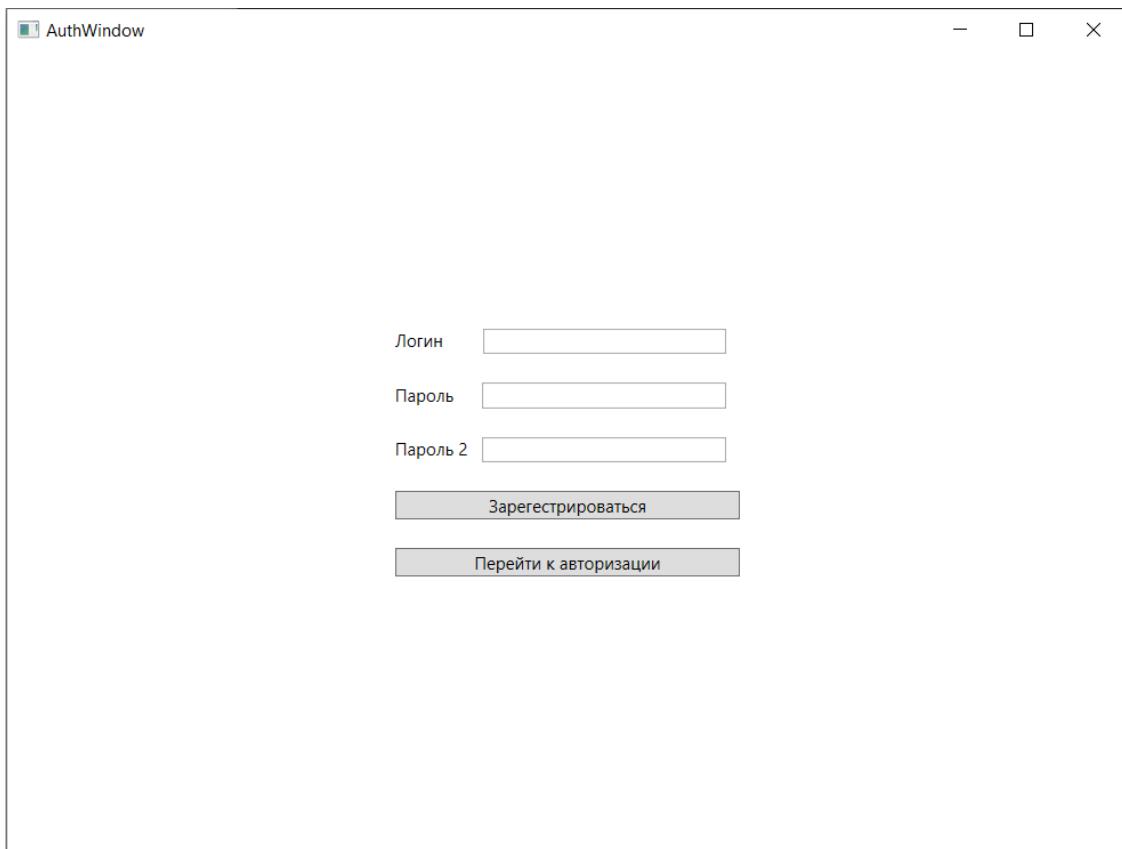
```

```
    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",
→   "G", "H", "I", "J", "K", "L", "M"};
}
```

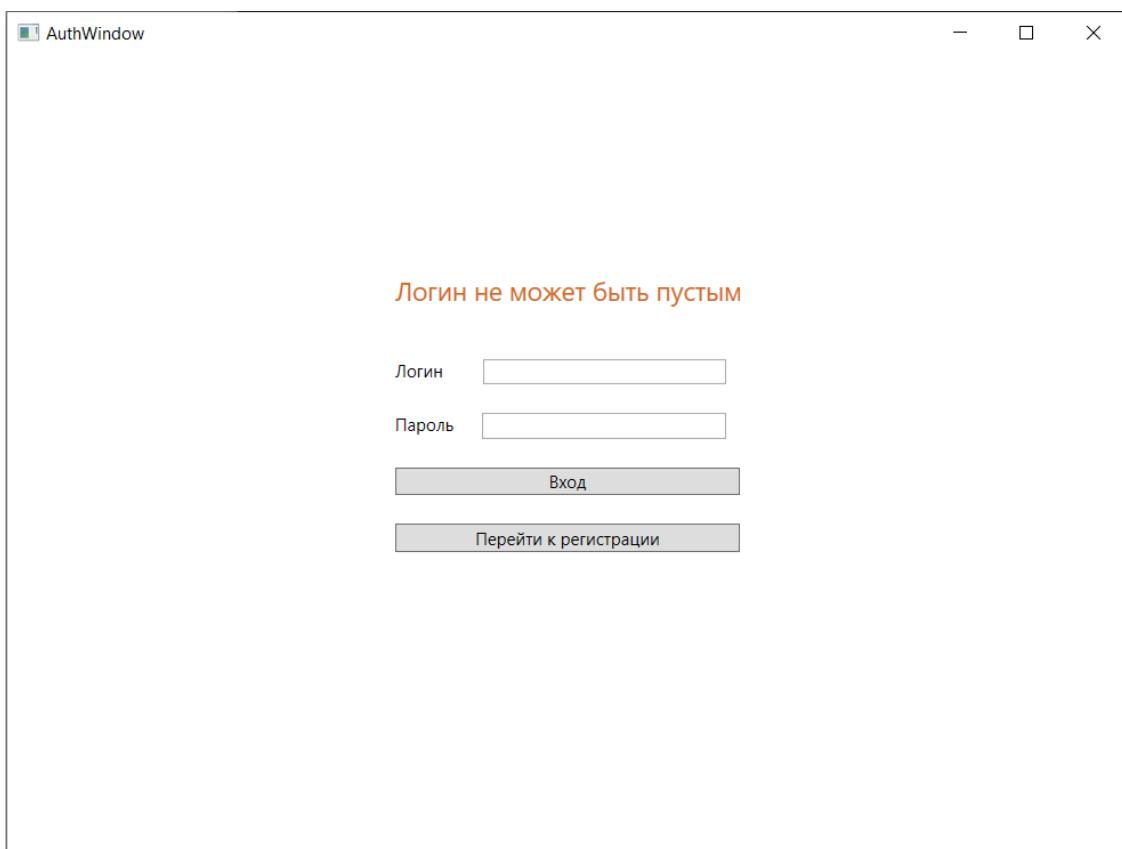
Демонстрация работы приложения:
Окно авторизации:



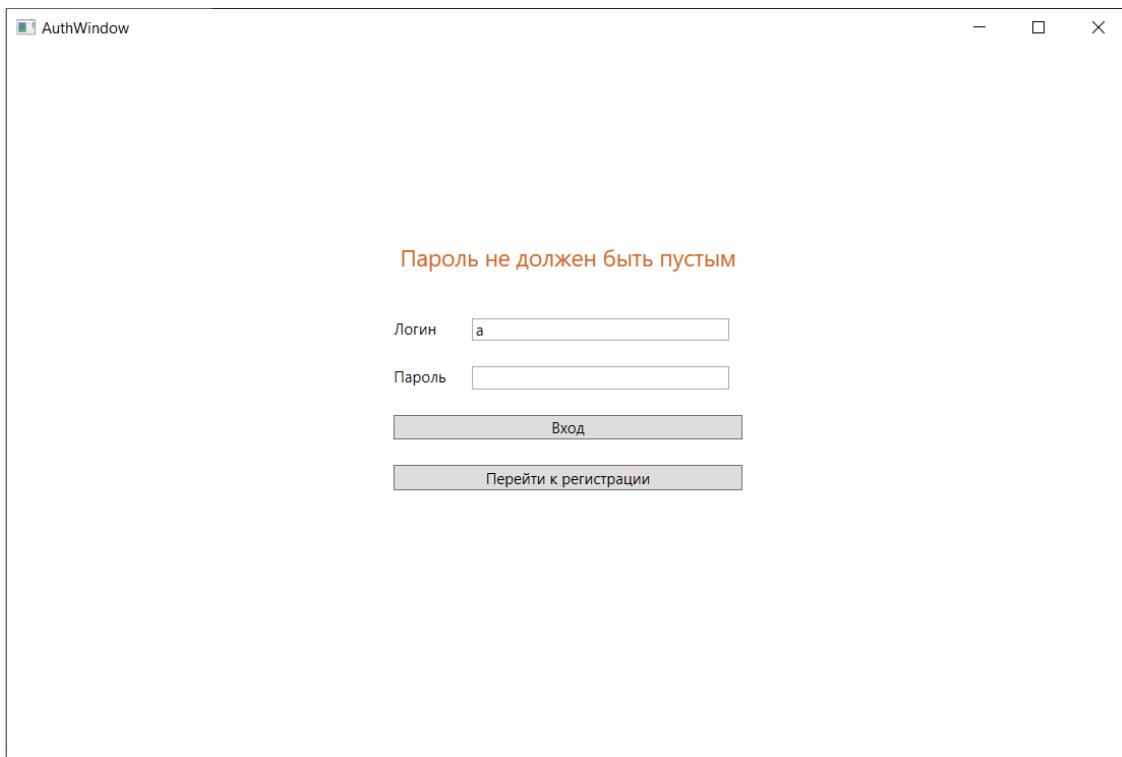
Переключение на регистрацию:



Проверка на пустой логин:



Проверка на пустой пароль:



Вид программы от обычного пользователя:

№	St.	Familia	Imya	Otchestvo	Adres	Gorod	Dataroggd	№gr	Pol	Budget
21806		Терентьев	Вячеслав	Сергеевич	ул. Победы		9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21678		Адвокатов	Владимир	Игорьевич	ул. Благодатная		1/23/1997 12:00:00 AM	31P	M	<input 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. Костищко		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. Костищко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
28954		Клягин	Максим	Павлович	ул. Марата	г. Гатчина	4/27/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
24786		Шубин	Александр	Сергеевич	Невский пр.		2/1/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
21045		Кренев	Ярослав	Сергеевич	Московский пр.		3/3/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21876		Шапцов	Александер	Сергеевич	ул. Варшавская		6/11/1997 12:00:00 AM	31P	M	<input type="checkbox"/>
24735		Полковский	Максим	Генадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
20087		Таран	Семен	Игорьевич	ул. Тележная		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	X	<input type="checkbox"/>
29956		Филиппова	Кристина	Сергеевна	ул. Кузнецковская		8/3/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
24336		Терентьева	Наталия	Сергеевна	ул. Победы		9/9/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
28167		Кунинина	Дарья	Евгеньевна	пр. Большевиков		11/1/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
26547		Кучкарук	Юлия	Алексеевна	Кировский пр.		8/15/1998 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
21178		Дроздов	Максим	Алексеевич	пр. Художников		4/12/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
23467		Соболев	Илья	Дмитриевич	пр. Стасек		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"/>
27690		Карабицена	Ольга	Аркадьевна	ул. Победы	г. Павловск	12/14/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
24659		Куликов	Юрий	Михайлович	ул. Ж. Диюло		9/26/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
29865		Кухарева	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		7/19/1997 12:00:00 AM	31M	M	<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

Демонстрация работы поиска:

MainWindow

NºSt	Familia	Imya	Otchestvo	Adres	Gorod	Dataroggd	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	Ж	<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	Ж	<input type="checkbox"/>

Фамилия: Юноши с возрастом

Поиск по группе Иногородние студенты

Поиск по году рождения

MainWindow

NºSt	Familia	Imya	Otchestvo	Adres	Gorod	Dataroggd	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	Datagrд	№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. Костюшко		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. Костюшко		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. Стacheк		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	Datagrд	№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

Фамилии 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

Демонстрация работы поиска иногородних студентов:

MainWindow

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

Фамилия: Поиск Юноши с возрастом

Поиск по группе Иногородние студенты

Поиск по году рождения

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
Карабиццина	31М
Куликов	31М
Кухарева	31М
Кононец	31М
Попова	31М
Платонов	31М
Петрова	31М
Шишина	31М

Фамилия: Пойск

Поиск по группе Образовательная реформа № 1

Поиск по году рождения Год рождения: 1997 Образовательная реформа № 2

Проверки при регистрации:

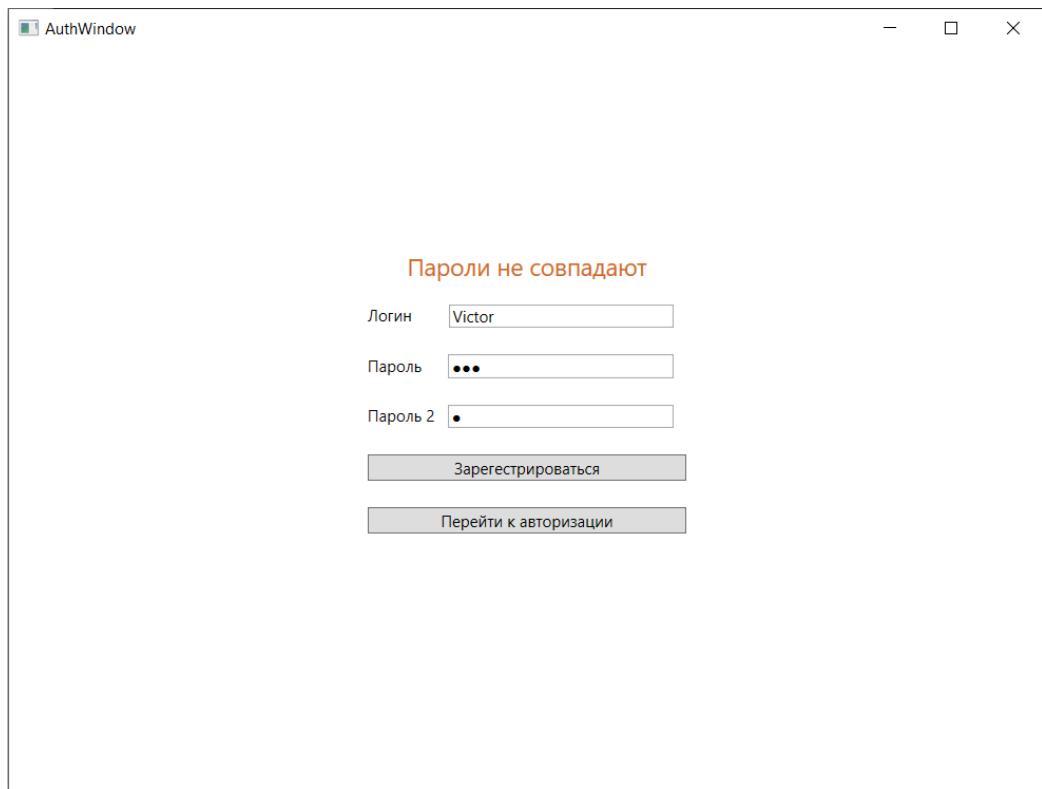
1. Логин уже существует

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"/>	

2. Не совпадение паролей



Демонстрация хранения данных:

- Пользователей

users			
user_id	login	password	role
2	admin	8c6976e5b541415bde98bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918	2
3	a	ca978112ca1bdcfaac231b39a23dc4da786ef8f8147c4e72b9807785afee48b	1
4	asd	688787d8ff1144c5027f5cffaafe2cc58d86079f9de88304c26b0cb99ce91c6	1
5	vasya	7a639b5f083943e8bdd1f41eb44025b0df1d189d618993f10624a2ace	1
6	kirya	6b86b273ff34fce19d6b804eff5a3f5747ada4ea22f1d49c01e52ddb785b4b	1
7	as	f4bf9f7fcbedaba0392f108c59d8f4a3b83838efb6487738171b54475c2ade8	1
8	Victor	866d116d21281a3a9375c6deca295bfe67c9989581578b1747b59da428e479	1

- Ролей

	role_id	role_name	
	1	user	
	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

№№	Familia	Имя	Отчество	Adres	Город	Datadogr	Ngr	Pol	Budget
21806	Терентьев	Вячеслав	Сергеевич	ул. Победы	г. Колпино	9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21678	Андроников	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	Головин	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	Розанов	Дмитрий	Сергеевич	ул. Костюшко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
28954	Клягин	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	Креин	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	Полковской	Максим	Геннадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
20097	Таран	Семен	Игоревич	ул. Тележинской		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
24556	Журина	Мария	Сергеевна	пр. Сбокова		4/30/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
29956	Олиппикова	Кристина	Сергеевна	ул. Кунцевская		8/3/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
24336	Терентьева	Наталья	Сергеевна	ул. Победы		9/9/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
28167	Кунинина	Дарья	Евгеньевна	ул. Большевиков		11/1/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
26547	Кучмарук	Юлия	Алексеевна	Кировский пр.		8/15/1998 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
21178	Дровдев	Максим	Алексеевич	пр. Художников		4/12/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
23467	Соболевец	Илья	Дмитриевич	пр. Станек		3/10/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
21544	Кильев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
27699	Карачинич	Ольга	Аркадьевна	ул. Победы	г. Павловск	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"/>
28656	Куракова	Галина	Тригорьевна	ул. Зины П.	г. Пушкин	5/13/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
23460	Конинец	Валерий	Иванович	21-Линия	г. Колпино	10/1/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
27166	Корнилова	Лена	Борисовна	ул. Победы		3/4/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
26849	Платонов	Николай	Павлович	8-Линия	г. Колпино	6/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21347	Петров	Лидия	Филипповна	Лиговский пр.		5/12/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
21375	Шишкова	Ранис	Ивановна	Лиговский пр.		6/9/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
29608	Цой	Виктор	Робертович	пр. Ветеранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21474	Смирнова	Татьяна	Михайлова	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

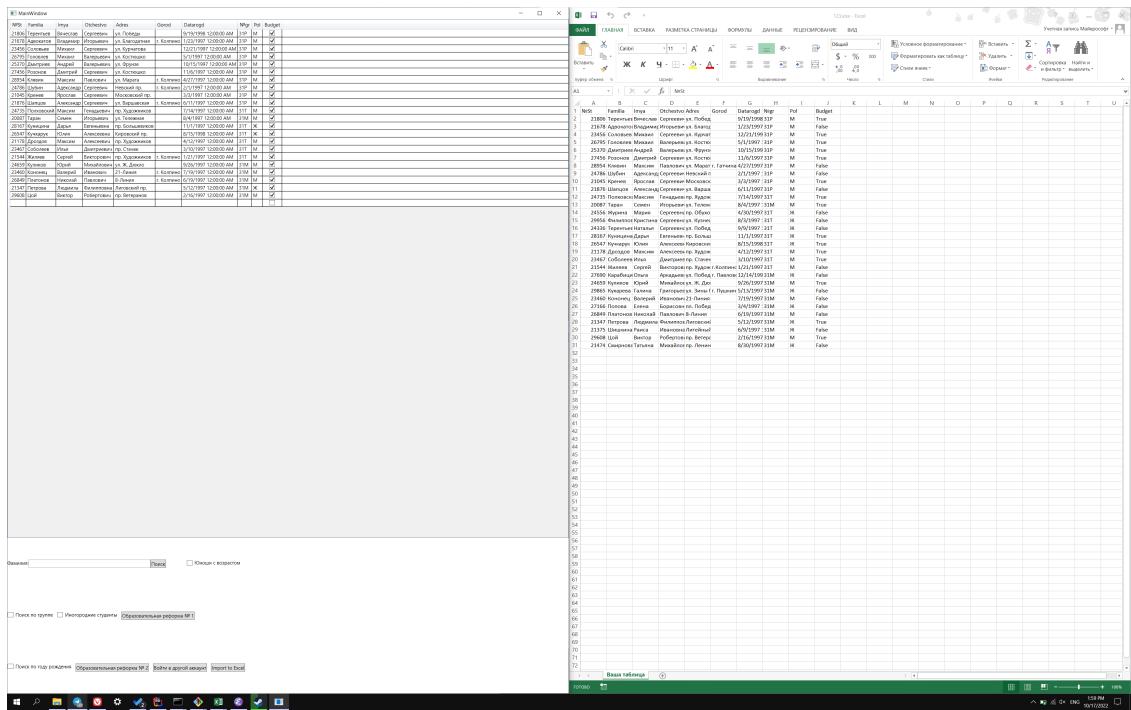
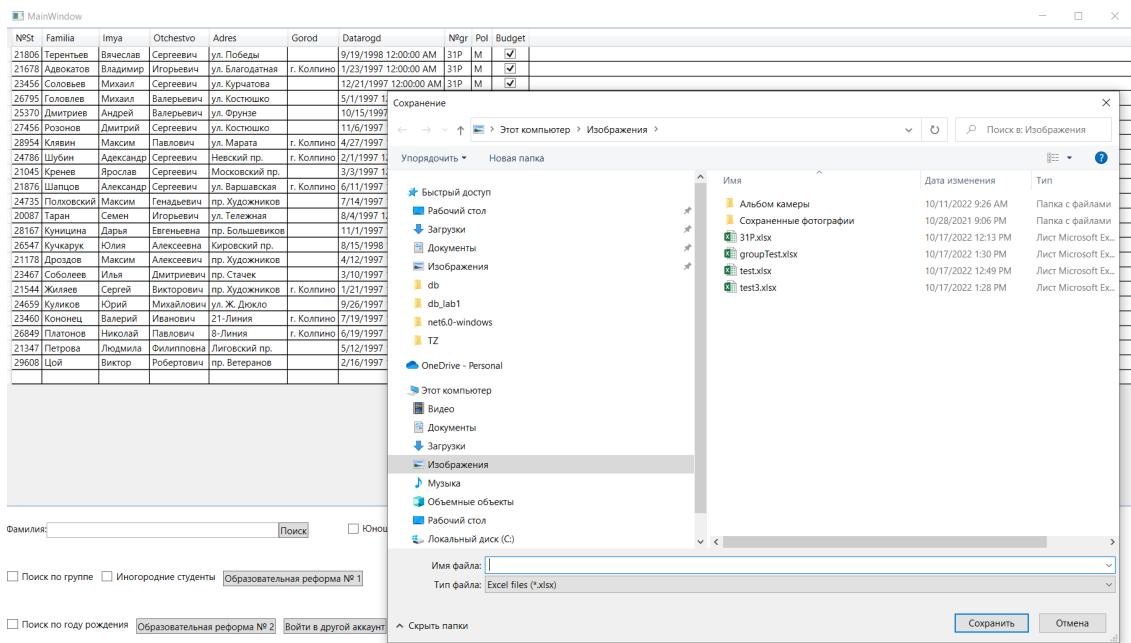
№№	Familia	Имя	Отчество	Adres	Город	Datadogr	Ngr	Pol	Budget
21806	Терентьев	Вячеслав	Сергеевич	ул. Победы	г. Колпино	9/19/1998 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
21678	Андроников	Владимир	Игоревич	ул. Благодатная	г. Колпино	1/23/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
23456	Соловьев	Михаил	Сергеевич	ул. Курчатова		12/21/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
26795	Головин	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	Розанов	Дмитрий	Сергеевич	ул. Костюшко		11/6/1997 12:00:00 AM	31P	M	<input checked="" type="checkbox"/>
28954	Клягин	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	Креин	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	Полковской	Максим	Геннадьевич	пр. Художников		7/14/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
20097	Таран	Семен	Игоревич	ул. Тележинской		8/4/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
24556	Журина	Мария	Сергеевна	пр. Сбокова		4/30/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
29956	Олиппикова	Кристина	Сергеевна	ул. Кунцевская		8/3/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
24336	Терентьева	Наталья	Сергеевна	ул. Победы		9/9/1997 12:00:00 AM	31T	X	<input type="checkbox"/>
28167	Кунинина	Дарья	Евгеньевна	ул. Большевиков		11/1/1997 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
26547	Кучмарук	Юлия	Алексеевна	Кировский пр.		8/15/1998 12:00:00 AM	31T	X	<input checked="" type="checkbox"/>
21178	Дровдев	Максим	Алексеевич	пр. Художников		4/12/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
23467	Соболевец	Илья	Дмитриевич	пр. Станек		3/10/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
21544	Кильев	Сергей	Викторович	пр. Художников	г. Колпино	1/21/1997 12:00:00 AM	31T	M	<input checked="" type="checkbox"/>
27699	Карачинич	Ольга	Аркадьевна	ул. Победы	г. Павловск	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"/>
28656	Куракова	Галина	Тригорьевна	ул. Зины П.	г. Пушкин	5/13/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
23460	Конинец	Валерий	Иванович	21-Линия	г. Колпино	7/19/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
27166	Корнилова	Лена	Борисовна	ул. Победы		3/4/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
26849	Платонов	Николай	Павлович	8-Линия	г. Колпино	6/19/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21347	Петров	Лидия	Филипповна	Лиговский пр.		5/12/1997 12:00:00 AM	31M	X	<input checked="" type="checkbox"/>
21375	Шишкова	Ранис	Ивановна	Лиговский пр.		6/9/1997 12:00:00 AM	31M	X	<input type="checkbox"/>
29608	Цой	Виктор	Робертович	пр. Ветеранов		2/16/1997 12:00:00 AM	31M	M	<input checked="" type="checkbox"/>
21474	Смирнова	Татьяна	Михайлова	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

Демонстрация работы импорта в Excel:



2 Лабораторная работа № 2

2.1 Часть 1

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="SQLite_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/" 
        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:

The screenshot shows a standard Windows application window titled "MainWindow". Inside the window, there is a DataGrid control displaying a table with three columns: "Id", "Name", and "Age". The table has one data row with values: Id=1, Name="Вася", and Age=10. There are also two empty rows below the data row.

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\\\\SQLite-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/" 
        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>
            <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:

The screenshot shows a Windows application window titled "SecondTask". At the top, there are three tabs: "Авторы" (Authors), "Книги" (Books), and "Авторы/Книги" (Authors/Books). The "Авторы" tab is selected. Below the tabs is a data grid with columns: "Id", "Name", "Age", and "AuthBooks". The grid contains 7 rows of data. At the bottom of the window, there is a horizontal input bar with three fields: "Имя:" (Name:), "Возраст:" (Age:), and a "Сохранить" (Save) button.

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	

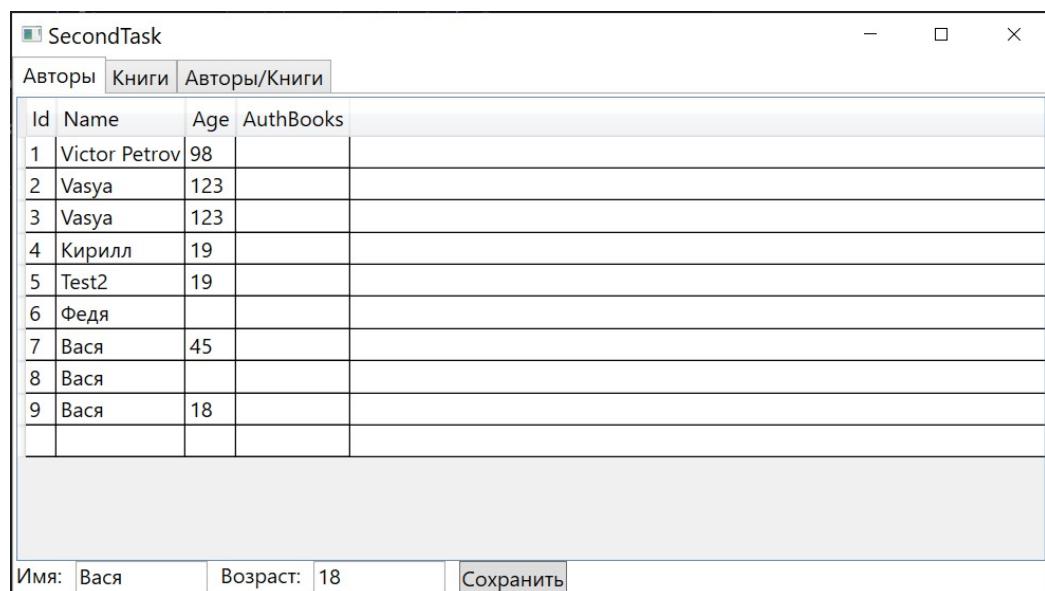
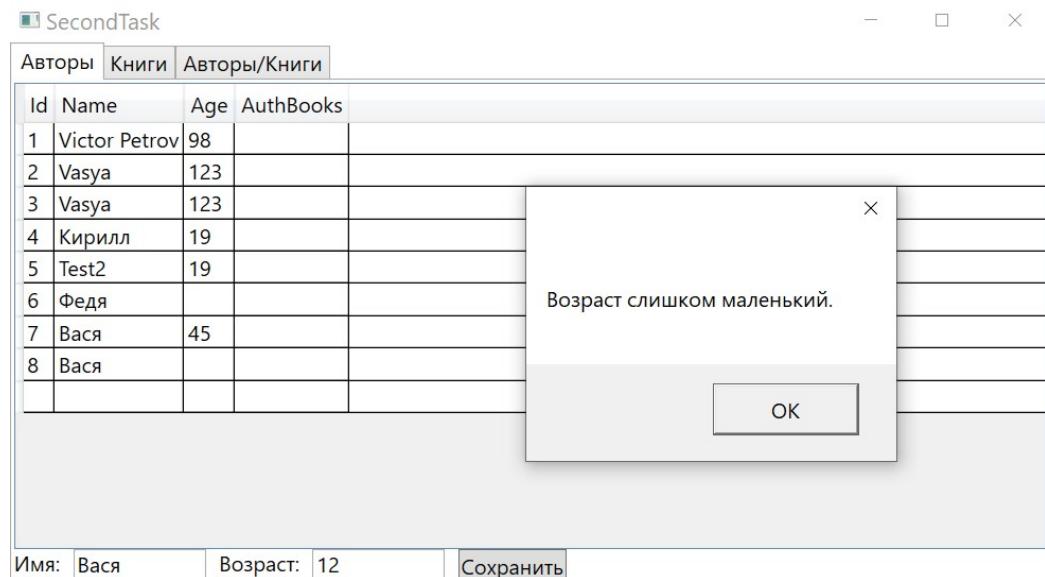
- Добавление автора:

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 у вас дурацкое.

Имя: | Возраст: | Сохранить



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:

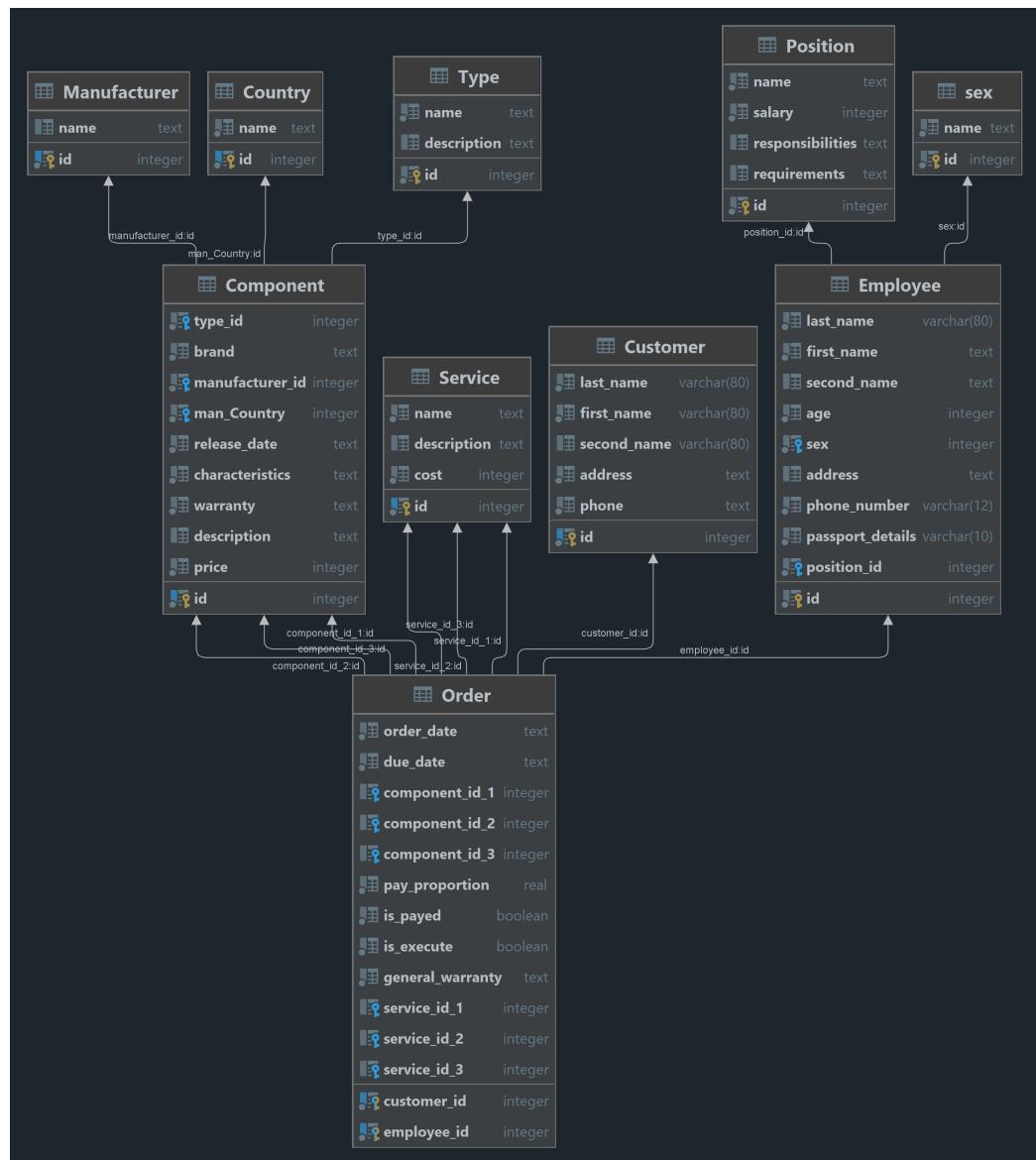
1. Сотрудники (Код сотрудника, ФИО, Возраст, Пол, Адрес, Телефон, Паспортные данные, Код должности).
2. Должности (Код должности, Наименование должности, Оклад, Обязанности, Требования)
3. Виды комплектующих (Код вида, Наименование, Описание)
4. Комплектующие (Код комплектующего, Код вида, Марка, Фирма производитель, Страна производитель, Дата выпуска, Характеристики, Срок гарантия, 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 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\\\\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 DbContext = 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.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"
       ↳ action"
       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="Круглая 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="NavView" 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">
    Sampoc 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"
      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="TableView">
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="20*" />
        <RowDefinition Height="*" />

    </Grid.RowDefinitions>
    <DataGrid IsReadOnly="True" Name="DataG"
    ↳ AutoGenerateColumns="True"
        CanUserAddRows="False" AutoGeneratedColumn 
    ↳ ns="DataG_OnAutoGeneratedColumns"
    
```

```

        AutoGeneratingColumn="TheDataGrid_OnAutoG_ ]
        ↵   eneratingColumn"
        ↵   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(Consi
        ↵   nts.NameToTableName[tableName]!)
        .GetValue(Constants.DbDataContext);

        t = dbSetType.MakeGenericType(data.GetType().GetProperty("E
        ↵   ntityType").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;
            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 TheDataGrid_OnAutoGeneratingColumn(object? sender,
        DataGridAutoGeneratingColumnEventArgs e)
    {
        Console.WriteLine();
    }
}

```

```

                if (e.PropertyType.AssemblyQualifiedName!.Contains("System. ")
→     Collections.Generic")
→     ||
→         e.PropertyType.AssemblyQualifiedName!.Contains("DbC "
→     ontext"))
→             _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 dataGridColTem = 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;
            dataGridColTem.CellTemplate = dataTemplate;
    }
}

```

```

        DataG.Columns.Add(dataGridColTem);
    }

}

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?", "",

→ MessageBoxButton.YesNo) ==
→ MessageBoxButtonResult.No)
            return;
        switch (tableName)
        {
            case "Сотрудники":
                foreach (var employee in
→ DataG.SelectedItems)
                    Constants.DbDataContext.Emp
→ loyees.Remove((Employee)
→ employee);
                break;
            case "Компоненты":
                foreach (var component in
→ DataG.SelectedItems)
                    Constants.DbDataContext.Com
→ ponents.Remove((Component)
→ component);
                break;
            case "Должности":
                foreach (var postition in
→ DataG.SelectedItems)
                    Constants.DbDataContext.Pos
→ itions.Remove((Position)
→ postition);
                break;
            case "Заказы":
                foreach (var order in
→ DataG.SelectedItems)
                    Constants.DbDataContext.Org
→ ers.Remove((Order)
→ order);
                break;
        }
    }
}

```

```

        case "Сервисы":
            foreach (var service in
                DataG.SelectedItems)
                Constants.DbDataContext.Ser...
            vices.Remove((Service)
            service);
            break;
        case "Заказчики":
            foreach (var service in
                DataG.SelectedItems)
                Constants.DbDataContext.Cus...
            tomers.Remove((Customer)
            service);
            break;
    }

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

```

(e) Добавление компонента (AddComponent)

- AddComponent.xaml

```

<Page x:Class="SuperDBApp.AddComponent"
      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/" 
      mc:Ignorable="d"
      Title="AddComponent">
<Grid Margin="10" HorizontalAlignment="Center"
      VerticalAlignment="Stretch">
    <ScrollViewer>
        <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>
    </ScrollViewer>
</Grid>

```

```

        </StackPanel>
        <StackPanel>
            <TextBlock>Производитель:</TextBlock>
            ↳ k>
            <ComboBox Name="ManuCombo"
                ↳ ItemsSource="{Binding
                ↳ Manufactures}"
                    SelectionChanged=
                        ↳ "ManufactureS"
                        ↳ elected"
                        ↳ />
        </StackPanel>
        <StackPanel>
            <TextBlock>Страна-производитель:</TextBlock>
            ↳ extBlock>
            <ComboBox Name="CountryCombo"
                ↳ ItemsSource="{Binding
                ↳ Countries}" SelectionChanged="C
                ↳ ountrySelected"
                ↳ />
        </StackPanel>
        <StackPanel>
            <TextBlock>Дата релиза:</TextBlock>
            <Calendar Name="CalendarIk" Selecte
                ↳ dDatesChanged="DateChanged"
                    SelectedDate="{Bi
                    ↳ nding Date}"
                    ↳ DisplayDate=""
                    ↳ {Binding
                    ↳ Date}" />
        </StackPanel>
        <StackPanel>
            <TextBlock>Характеристики:</TextBlock>
            ↳ ck>
            <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"
      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="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>
            <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}" />
        </StackPanel>
    </StackPanel>

```

```

    SelectionChanged="Position_Choiced"
    ↵   n_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:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/"
      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
    }).ToList();
}

private void Customer_Changed(object sender,
    SelectionChangedEventArgs e)
{
    var id = int.Parse((sender as
        ComboBox).SelectedItem.ToString()!.Split(',') [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"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/" 
    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: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}\"
        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	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 Круой реки 149	+71111111111	1234 567891	3	Изменить
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.

Крутая компания©

	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.

Крутая компания©

	Name	Salary	Responsibilities	Requirements
1	Директор	500000	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.

Крутая компания©

OrderDate	DueDate	CustomerId	ComponentId1	ComponentId2	ComponentId3	PayProportion	IsPayed	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.

Крутая компания©

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

Удалить

Крутая компания

Заказчики

	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

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.

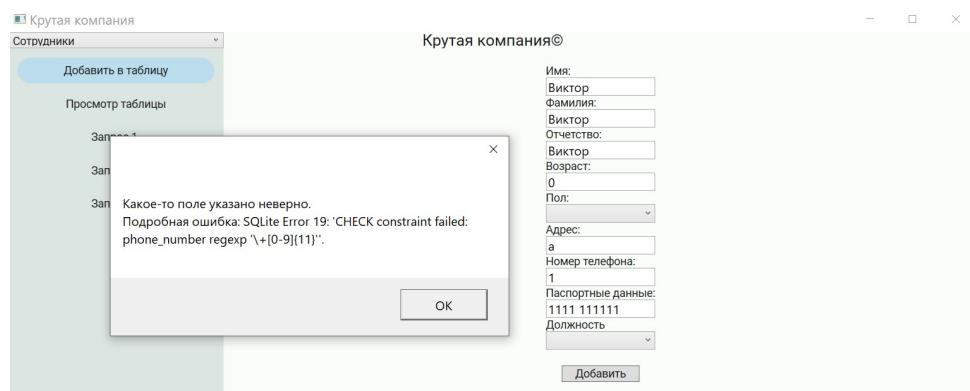
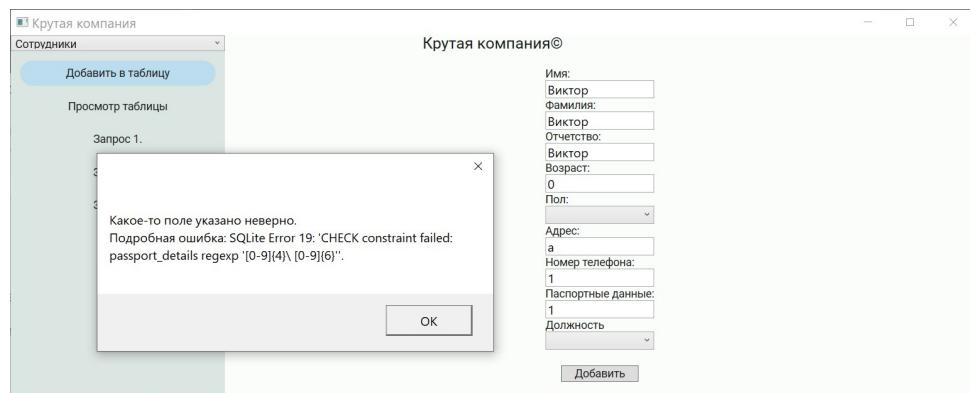
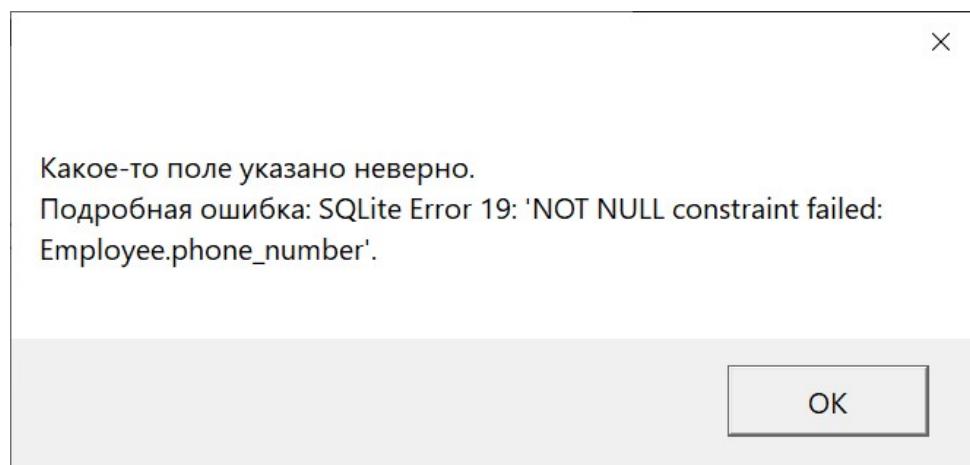
Запрос 3.

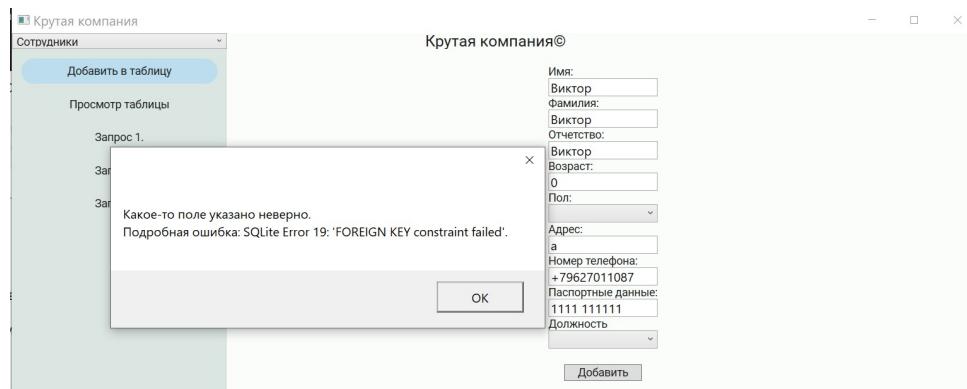
Крутая компания®

	Id	LastName	FirstName	SecondName	Age	Sex	Address	PhoneNumber	PassportDetails	PositionId	
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

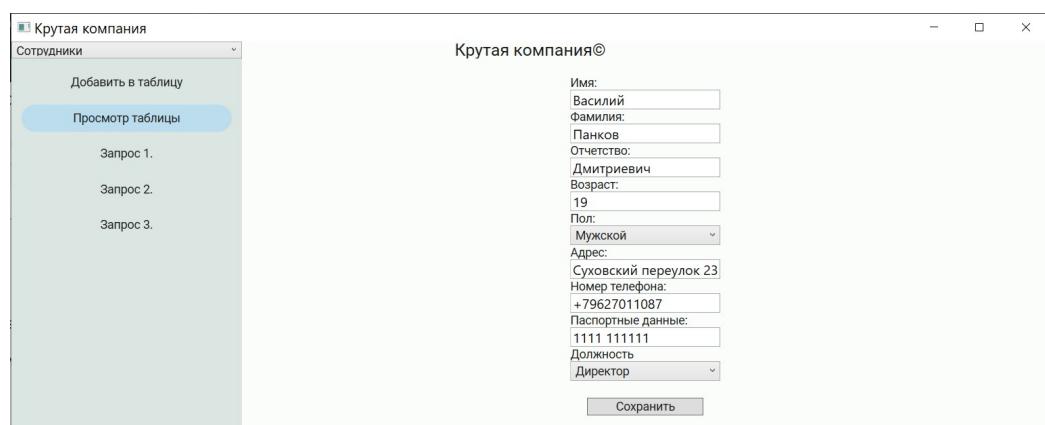
Удалить

- Возможные ошибки

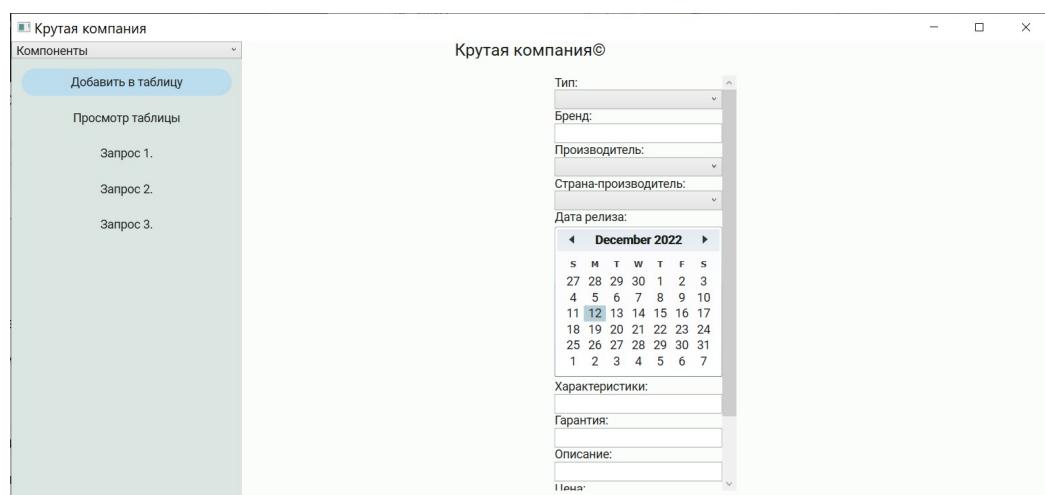


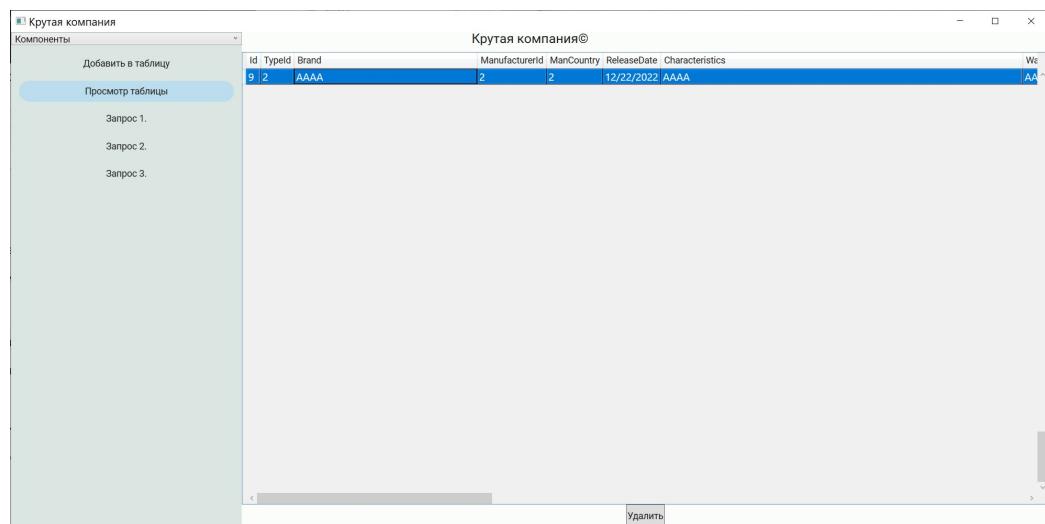
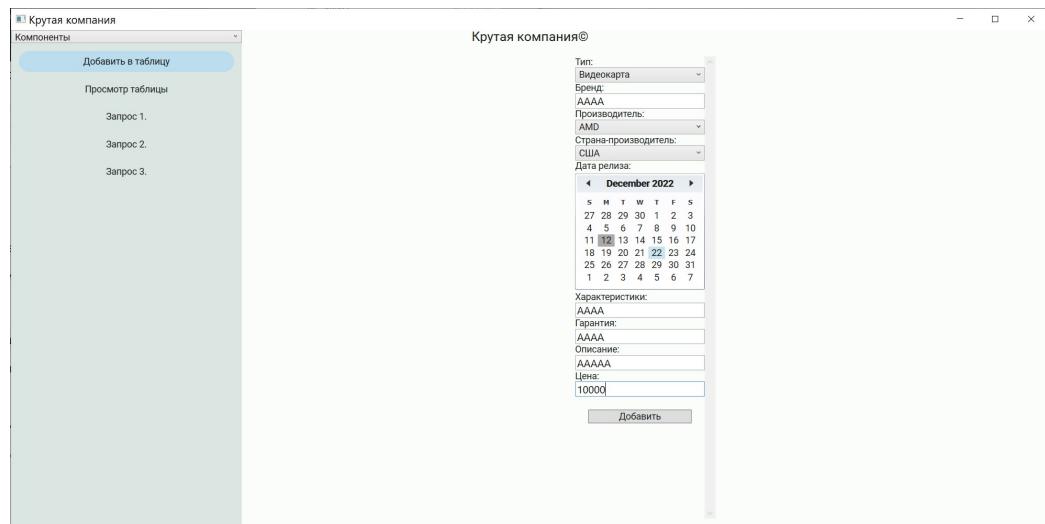


3. Изменение сотрудника

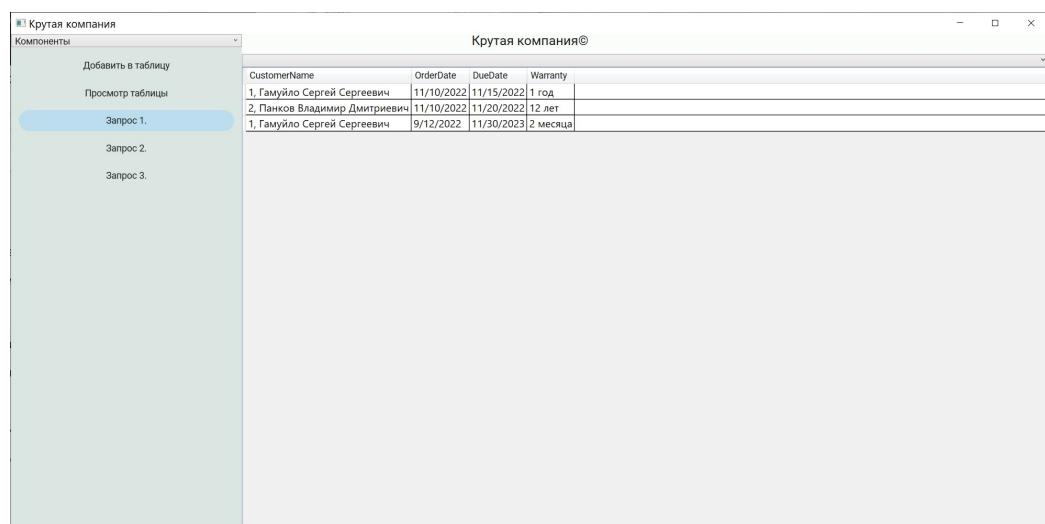


4. Добавление компонента





5. Запрос 1



Крутая компания

Компоненты

Добавить в таблицу

Просмотр таблицы

Запрос 1.

Запрос 2.

Запрос 3.

Крутая компания®

1. Гамчило Сергей Сергеевич

OrderDate	DueDate	CustomerId	ComponentId1	ComponentId2	ComponentId3	PayProportion	IsPayed	IsExecute	GeneralWarranty	ServiceId1	ServiceId2	ServiceId3	Emplo
11/10/2022	11/15/2022	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			

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 Термоинтерфейс 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 работы

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.

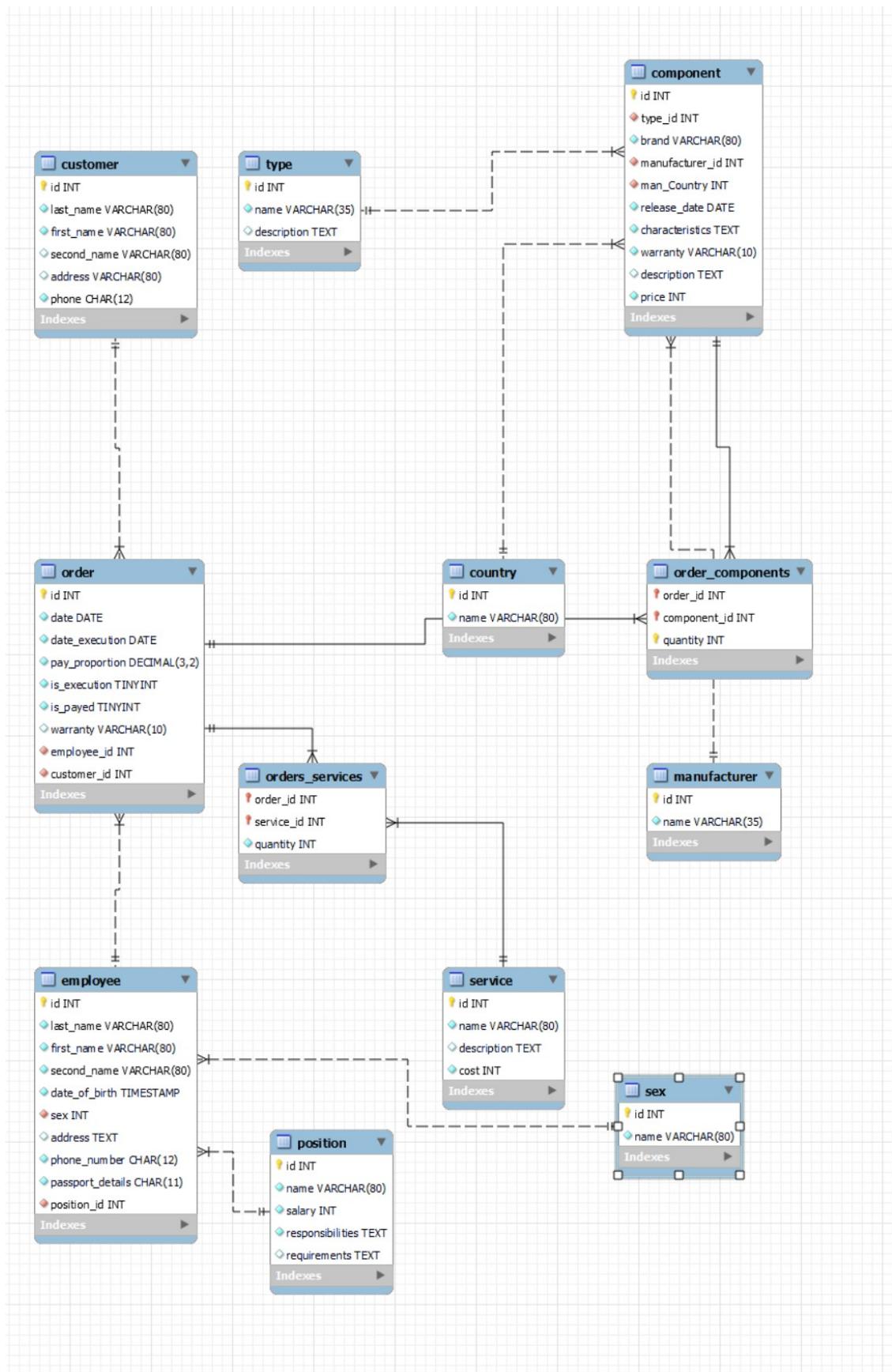
Таблицы:

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 гарантии, 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	Учится 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	Суховский переулок 23	+79627011087	1111 111111	1
2	2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222 222222	2
3	3	Турсунова	Лидия	Сергееvна	1988-11-06 12:11:18	2	набережная Кругой реки 149	+71111111111	1234 567891	3
4	9	Ланков	Вася	Дмитриевич	2006-12-16 12:11:32	1 <null>		+79627011087	1111 444444	2
5	10	Виктор	Виктор	Виктор	2004-07-02 12:11:41	1 a		+79627011087	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	man_country	release_date	characteris...	warranty	description	price
1	1	1	Ryzen 5 3600x	2	2022-12-14	Общие параметры: Модел... 1 год	Хороший процессор для иг...	15000		
2	2	2	ROG-STRIX-RX6700XT-012G-GAMING	1	2012-12-13	Графический процессор. 1 год	Видеокарта нового покол...	60000		
3	3	3	IronWolf ST4000VN008	10	2020-12-18	>Фактор: 3.5 " * 06. 1 год	Готовое к работе устройс...	10000		
4	4	4	A400 SA400S37/4886	11	2019-12-12	Объем накопителя: 480 ... 3 года	Накопитель SSD KINGSTON ...	2800		
5	5	5	15	12	2022-06-10	Конфигурация корпуса... 1 год	корпус ATX ZALMAN 15 и...	5500		
6	6	3	Test 2	1	2017-07-21	Хпз	ddsa	dasd	0	
7	7	1	19	3	2022-08-11	Кругой	1 мес.	<null>	0	
8	8	2	AAAA	2	2028-12-22	AAAAA	AAAAA	AAAAAA	10000	

8. Customer (Покупатель)

	id	last_name	first_name	second_name	address	phone
1	1	Гамуило	Сергей	Сергеевич	г. Санкт-Петербург	+79111111111
2	2	Панков	Владимир	Дмитриевич	г. Санкт-Петербург г. Колпино	+79111112111
3	3	Белоцерковский	Марат	<null>	<null>	+79999999999
4	4	Куриличев	Михаил	<null>	<null>	+123456789801
5	5	Бобров	Федор	Николаевич	<null>	+79056789909

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	2022-12-25	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

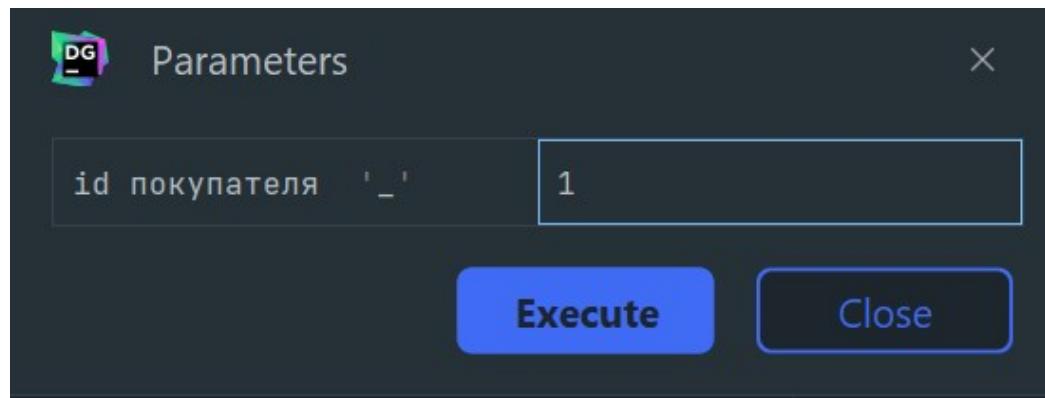
12. order_services (Сервисы по заказам)

	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;
```

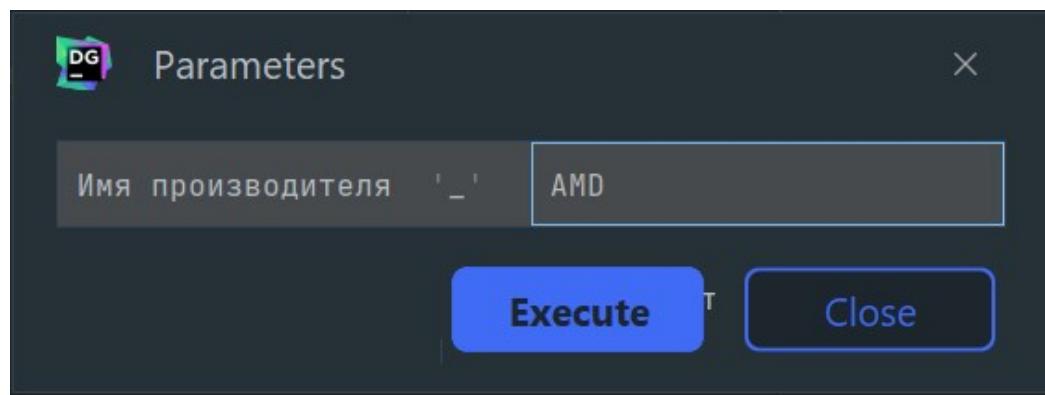


	customer_id	last_name	first_name	second_name	order_id	date	data_execution	pay_proportion	is_execution	is_paid	warranty	employee_id
1	1	Гануло	Сергей	Сергеевич	1	2022-12-08	2022-12-23	1.00	0	1 12 мес		1
2	1	Гануло	Сергей	Сергеевич	2	2022-12-01	2022-12-24	0.50	0	1 1 год		1

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 5600x	2	2	2022-12-14	Быстро параметры: Модель: AMD Ryzen 5 5600x/Сокет: 1 Гнездо	1 год	Кодовый процессор для игр и для работы	18000
2	AMD	2	2	X670	2	2	2022-12-21	AM4	AM4	AM4	18000

3.2 Часть 2

Тема: работа с данными в формате json. Сериализация и десериализация объектов.

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

Задание 1. Сериализация

- 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"
        ↳ action"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expressionblend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        ↳ ility/2006"
        xmlns:local="clr-namespace:ExampleJson"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="10*"/></RowDefinition>
            <RowDefinition Height="1*"/></RowDefinition>
        </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()`.

Класс `WebResponse` представляет данные, извлекаемые из сервера. Вызов метода `WebRequest.GetResponse()` приводит к отправке запроса веб-серверу и к созданию объекта `WebResponse` 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,"owner":"nul ]  
→ l","file":"626ef2d97f254a0017b564ef.gif","mimetype":"image/gif","size":235257 ]  
→ 8,"_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>  
        <RowDefinition Height="*"/></RowDefinition>  
        <RowDefinition Height="5*"/></RowDefinition>
```

```

        <RowDefinition Height="*" />
        <RowDefinition Height="5*" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="*" />
        <ColumnDefinition Width="*" />
    </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 Name="JsonInfoBox" Grid.Row="2" Grid.Column="2"
        Margin="10" />
    <TextBlock VerticalAlignment="Center" FontSize="20" Grid.Row="3"
        Grid.ColumnSpan="2" TextAlignment="Center">
        Картишка</TextBlock>
    <Image Grid.ColumnSpan="2" Name="Image" Grid.Row="4" />
</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)}
Создан: {cat!.createdAt}
Обновлён: {cat!.updatedAt}";
        Image.Source = new BitmapImage(new
        → Uri("https://cataas.com" + cat.url));
    }
}

```

Демонстрация работы приложения:



MainWindow

Отправить запрос

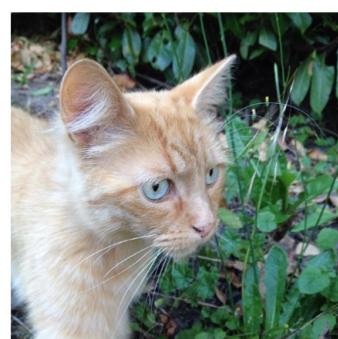
Полученный JSON

```
{"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"}
```

Полученная информация

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

Картинка



MainWindow

Отправить запрос

Полученный JSON

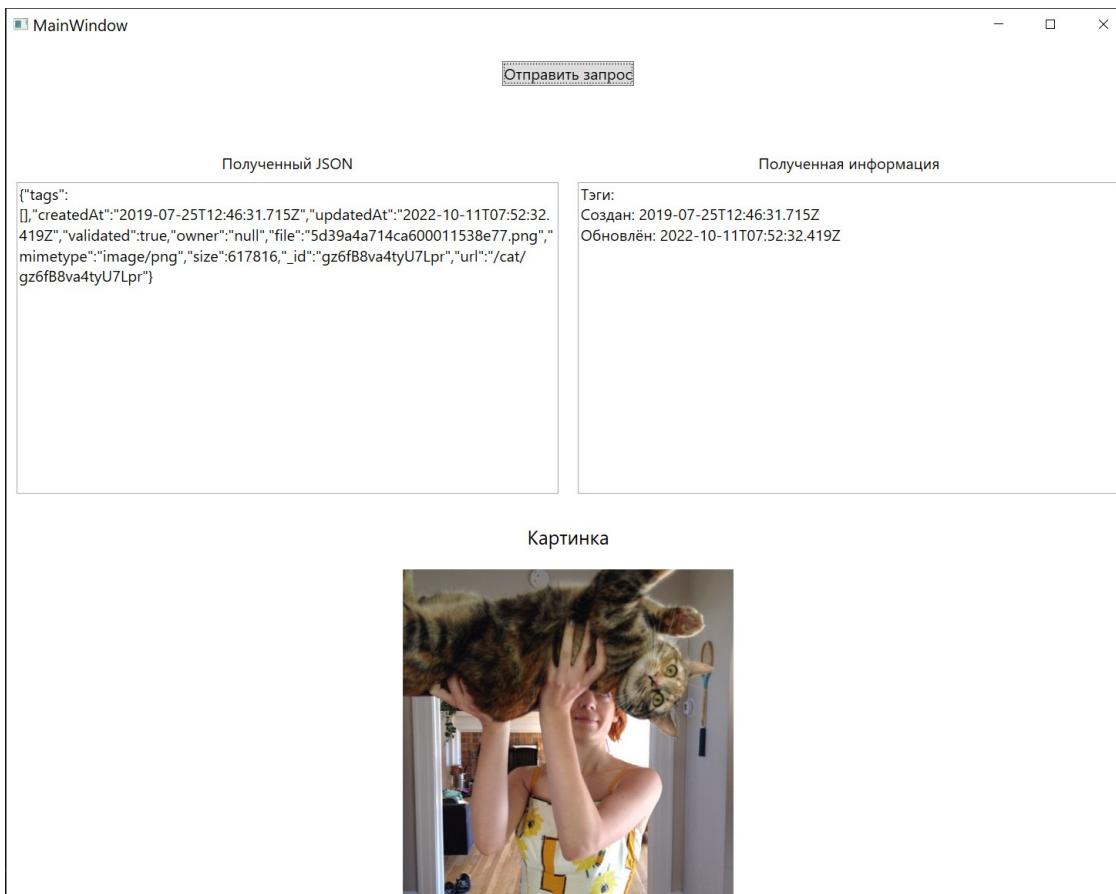
```
{"tags": ["computer"], "createdAt": "2016-11-30T09:32:46.377Z", "updatedAt": "2022-10-11T07:52:32.189Z", "validated": true, "owner": null, "file": "595f2809557291a9750ebf43.jpeg", "mimetype": "image/jpeg", "size": 269527, "_id": "aOkTbVU5PTWrzwah", "url": "/cat/aOkTbVU5PTWrzwah"}
```

Полученная информация

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

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 1: Таблица сотрудников

1	Панков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Суховский переулок 23	+79627011087	1111	1
2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222	2
3	Турсунова	Лидия	Сергеевна	1980-11-06 12:11:18	2	набережная Крутой реки 149	+711111111111	1234	3
9	Панков	Вася	Дмитриевич	2006-12-16 12:11:32	1	NULL	+79627011087	1111	2
10	Виктор	Виктор	Виктор	2004-07-02 12:11:41	1	a	+79627011087	1111	2

Полученный экспорт в CSV:

```
id,last_name,first_name,second_name,date_of_birth,sex,address,phone_number,passport_data |  
↪  ils,position_id  
1,Панков,Василий,Дмитриевич,2003-12-15 12:10:59,1,Суховский переулок  
↪  23,+79627011087,1111 111111,1  
2,Гайкин,Кирилл,Денисович,1999-07-17 12:11:08,1,Кобринская улица 5,+79118445162,2222  
↪  222222,2  
3,Турсунова,Лидия,Сергеевна,1980-11-06 12:11:18,2,набережная Крутой реки  
↪  149,+71111111111,1234 567891,3  
9,Панков,Вася,Дмитриевич,2006-12-16 12:11:32,1,,+79627011087,1111 444444,2  
10,Виктор,Виктор,Виктор,2004-07-02 12:11:41,1,a,+79627011087,1111 111111,2
```

Таблица 2: Таблица стран

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 3: Таблица должностей

id	name	salary	responsibilities	requirements
1	Директор	1000000	Руководить	NULL
2	Продавец	24000	Эффективно продавать товары	1. Выполнять скрипты 2. Выполнять нормы продаж 3. Уведомлять покупателей о новых акциях
3	Главный менеджер	55200	Следить за работой подчинённых	1. Оформлять кучу отчётов 2. Следить за работой подчинённых
5	Стажёр	6000	Учиться работать	Должен работать под надзором сотрудника-наставника.
6	Младший менеджер	18000	Бумажная работа	Оформляет товары и занимается большей бумажной работой чем главный менеджер.

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 скриншоты 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: нкован-папан. Наверное с Кириллом вы незнакомы, но его имя отлично преобразуется в Килька, а ещё классно пишется задом наперёд: лириК

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 5: 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',
'A 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 заувершить 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. 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. Видимо придётся заувершить рассказ.

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
2004-01-24 00:00:00	2004-01-24

4 Лабораторная работа № 4 «Создание запросов в MySQL Workbench.»

Цель работы: практическое освоение команд SELECT, INSERT, UPDATE и DELETE; работа со строковыми функциями MySQL и функциями для работы с датой и временем.

1) Создайте три таблицы в новой базе данных.

SQL – скрипт:

Создание таблицы Продавец (Seller)

```
create table Seller
(
    id_sell      int primary key not null,
    last_name    varchar(80) not null,
    first_name   varchar(80) not null,
    second_name  varchar(80) not null,
    city_sell    varchar(30),
    head         varchar(40),
    comm_sell    decimal,
    sales_plan   int
);
```

Создание таблицы Заказчик (Customer)

```
create table Customer
(
    id_cust      int primary key not null,
    last_name    varchar(80) not null,
    first_name   varchar(80) not null,
    second_name  varchar(80) not null,
    city_cust    varchar(30),
    rate_cust    int,
    credit       int
);
```

Создание таблицы Заказ (Order)

```
create table shop.`order`
(
    id_order     int primary key not null,
    sum_order    decimal,
    date_order   datetime,
    id_sell      int      not null,
    id_customer  int      not null,
```

```

constraint order_customer_null_fk
    foreign key (id_customer) references shop.customer (id_cust),
constraint order_seller_null_fk
    foreign key (id_sell) references shop.seller (id_sell)
);

```

2) Используйте следующий SQL-скрипт для заполнения таблиц.

```

INSERT INTO shop.seller (id_sell, last_name, first_name, second_name, city_sell, head,
→ comm_sell, sales_plan)
VALUES (2, 'Приморская', 'Алина', 'Петровна', 'Москва', 'Талимов Е.Е.', 18, 150);

INSERT INTO shop.seller (id_sell, last_name, first_name, second_name, city_sell, head,
→ comm_sell, sales_plan)
VALUES (3, 'Петухов', 'Игорь', 'Максимович', 'Екатеринбург', 'Федулов А.С.', 18, 300);

INSERT INTO shop.seller (id_sell, last_name, first_name, second_name, city_sell, head,
→ comm_sell, sales_plan)
VALUES (4, 'Терехова', 'Нина', 'Семеновна', 'Москва', 'Рахимов П.Г.', 18, 100);

INSERT INTO shop.seller (id_sell, last_name, first_name, second_name, city_sell, head,
→ comm_sell, sales_plan)
VALUES (5, 'Никифорова', 'Анастасия', 'Петровна', 'Екатеринбург', 'Федулов А.С.', 18,
→ 200);

```

	id_sell	last_name	first_name	second_name	city_sell	head	comm_sell	sales_plan
1	2	Приморская	Алина	Петровна	Москва	Талимов Е.Е.	18	150
2	3	Петухов	Игорь	Максимович	Екатеринбург	Федулов А.С.	18	300
3	4	Терехова	Нина	Семеновна	Москва	Рахимов П.Г.	18	100
4	5	Никифорова	Анастасия	Петровна	Екатеринбург	Федулов А.С.	18	200

Рисунок 1 – Таблица продавцов

```

INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)
VALUES (1, 'Петров', 'Олег', 'Дмитриевич', 'Магадан', 83, 150000);
INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)
VALUES (2, 'Антипов', 'Валерий', 'Петрович', 'Екатеринбург', 45, 75000);
INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)
VALUES (3, 'Берегов', 'Владимир', 'Евгеньевич', 'Москва', 96, 500000);
INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)
VALUES (4, 'Чернов', 'Антон', 'Петрович', 'Тула', 62, 100000);
INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)

```

```

VALUES (5, 'Голубев', 'Иосиф', 'Николаевич', 'Тбилиси', 78, 120000);
INSERT INTO customer (id_cust, last_name, first_name, second_name, city_cust, rate_cust,
→ credit)
VALUES (6, 'Романов', 'Глеб', 'Иванович', 'Екатеринбург', 91, 270000);

```

	id_cust	last_name	first_name	second_name	city_cust	rate_cust	credit
1	1	Петров	Олег	Дмитриевич	Магадан	83	150000
2	2	Антипов	Валерий	Петрович	Екатеринбург	45	75000
3	3	Берегов	Владимир	Евгеньевич	Москва	96	500000
4	4	Чернов	Антон	Петрович	Тула	62	100000
5	5	Голубев	Иосиф	Николаевич	Тбилиси	78	120000
6	6	Романов	Глеб	Иванович	Екатеринбург	91	270000

Рисунок 2 – Таблица заказчиков

```

INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (1, 50000, '2010.06.12', 2, 2);
INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (2, 30000, '2010.06.12', 2, 2);
INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (3, 70000, '2010.06.13', 2, 1);
INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (4, 60000, '2010.06.13', 5, 1);
INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (5, 35000, '2010.06.14', 4, 5);
INSERT INTO shop.`order` (id_order, sum_order, date_order, id_sell, id_customer)
VALUES (6, 70000, '2010.06.14', 4, 3);

```

	id_order	sum_order	date_order	id_sell	id_customer
1	1	50000	2010-06-12 00:00:00	2	2
2	2	30000	2010-06-12 00:00:00	2	2
3	3	70000	2010-06-13 00:00:00	2	1
4	4	60000	2010-06-13 00:00:00	5	1
5	5	35000	2010-06-14 00:00:00	4	5
6	6	70000	2010-06-14 00:00:00	4	3

Рисунок 3 – Таблица заказов

Создайте и выполните следующие запросы к созданным таблицам

1. Напишите запрос, который вывел бы все строки из таблицы “Заказчики”, для которых номер заказчика =5.

```
SELECT * FROM customer WHERE id_cust = 5;
```

2. Напишите запрос, который вывел бы коды всех продавцов в текущем порядке из таблицы “Заказы”, без каких бы то ни было повторений.

```
SELECT distinct id_sell FROM `order`;
```

RESULTS:						
id_cust	last_name	first_name	second_name	city_cust	rate_cust	credit
5	Голубев	Иосиф	Николаевич	Тбилиси	78	120000

Рисунок 4 – Результат выполнения запроса №1

id_sell
2
4
5

Рисунок 5 – Результат выполнения запроса №2

3. Напишите запрос к таблице “Заказчики”, который содержит всех заказчиков с рейтингом < 100, если они не находятся в Москве.

```
SELECT * FROM customer WHERE rate_cust < 100 AND city_cust != 'Москва';
```

RESULTS:						
id_cust	last_name	first_name	second_name	city_cust	rate_cust	credit
1	Петров	Олег	Дмитриевич	Магадан	83	150000
2	Антипов	Валерий	Петрович	Екатеринбург	45	75000
4	Чернов	Антон	Петрович	Тула	62	100000
5	Голубев	Иосиф	Николаевич	Тбилиси	78	120000
6	Романов	Глеб	Иванович	Екатеринбург	91	270000

Рисунок 6 – Результат выполнения запроса №3

4. Напишите запрос, который мог бы вывести все заказы на 12 или 13 июня.

```
SELECT * FROM `order` WHERE MONTH(date_order) = 6 AND (DAYOFMONTH(date_order) = 12 OR
→ DAYOFMONTH(date_order) = 13);
```

5. Напишите запрос, который выбрал бы наименьшую сумму для каждого заказчика

```
SELECT CONCAT(c.last_name, " ", c.first_name, " ", c.second_name) AS 'ФИО',
→ MIN(sum_order) FROM `order` JOIN customer c ON c.id_cust = `order`.id_customer
→ GROUP BY id_customer
```

6. Напишите запрос, который выбрал бы высший рейтинг в каждом городе.

RESULTS:				
<code>id_order</code>	<code>sum_order</code>	<code>date_order</code>	<code>id_sell</code>	<code>id_customer</code>
1	50000	2010-06-12 00:00:00	2	2
2	30000	2010-06-12 00:00:00	2	2
3	70000	2010-06-13 00:00:00	2	1
4	60000	2010-06-13 00:00:00	5	1

Рисунок 7 – Результат выполнения запроса № 4

RESULTS:	
<code>ФИО</code>	<code>MIN(sum_order)</code>
Антипов Валерий Петрович	30000
Петров Олег Дмитриевич	60000
Голубев Иосиф Николаевич	35000
Берегов Владимир Евгеньевич	70000

Рисунок 8 – Результат запроса № 5

```
SELECT city_cust as 'Город', MAX(rate_cust) AS 'Рейтинг' FROM customer GROUP BY
→ city_cust;
```

7. Напишите запрос, который бы выводил всех заказчиков, обслуживаемых продавцом с комиссионными выше 12%. Выведите имя заказчика, имя продавца, и ставку комиссионных продавца.

```
SELECT CONCAT_WS(' ', c.last_name, c.first_name, c.second_name) AS 'Имя заказчика',
       CONCAT_WS(' ', s.last_name, s.first_name, s.second_name) AS 'Имя продавца',
       s.comm_sell
  FROM `order`
    JOIN customer c ON c.id_cust = id_customer
    JOIN seller s on s.id_sell = `order`.id_sell AND s.comm_sell > 12;
```

8. Напишите запрос, который вычислил бы сумму комиссионных продавца для каждого заказа заказчика с рейтингом выше 80.

```
SELECT CONCAT_WS(' ', c.last_name, c.first_name, c.second_name) AS 'Заказчик',
       SUM(s.sales_plan * (s.comm_sell / 100)) as 'Сумма комиссионных'
  FROM `order`
    JOIN seller s on s.id_sell = `order`.id_sell
    JOIN customer c on c.id_cust = `order`.id_customer
   AND c.rate_cust > 80
 GROUP BY id_customer;
```

RESULTS:

Город	Рейтинг
Магадан	83
Екатеринбург	91
Москва	96
Тула	62
Тбилиси	78

Рисунок 9 – Результат работы запроса № 6

RESULTS:

Имя заказчика	Имя продавца	comm_sell
Антипов Валерий Петрович	Приморская Алина Петровна	18
Антипов Валерий Петрович	Приморская Алина Петровна	18
Петров Олег Дмитриевич	Приморская Алина Петровна	18
Голубев Иосиф Николаевич	Терехова Нина Семеновна	18
Берегов Владимир Евгеньевич	Терехова Нина Семеновна	18
Петров Олег Дмитриевич	Никифорова Анастасия Петровна	18

Рисунок 10 – Результаты выполнения запроса № 7

RESULTS:

Заказчик	Сумма комиссионных
Петров Олег Дмитриевич	63.0000
Берегов Владимир Евгеньевич	18.0000

Рисунок 11 – Результат выполнения запроса № 8

9. Напишите запрос, который бы выбрал общую сумму всех заказов для каждого продавца, у которого эта общая сумма больше, чем сумма наибольшего заказа в таблице “Заказы”.

```
SELECT CONCAT_WS(' ', s.last_name, s.first_name, s.second_name) AS 'Продавец',
→ sum(sum_order) FROM `order`
JOIN seller s ON s.id_sell = `order`.id_sell GROUP by s.id_sell HAVING sum(sum_order)
→ > max(sum_order);
```

RESULTS:

Продавец	sum(sum_order)
Приморская Алина Петровна	150000
Терехова Нина Семеновна	105000

Рисунок 12 – Результат запроса №9

10. Напишите команду SELECT, использующую подзапрос, которая выберет фамилии и коды всех заказчиков с максимальными для их городов рейтингами.

```
SELECT id_cust, last_name
FROM customer cust1
WHERE rate_cust = (SELECT MAX(rate_cust) FROM customer cust2 WHERE cust1.city_cust =
→ cust2.city_cust);
```

11. Напишите запрос, который выберет всех продавцов (по их фамилии и коду), имеющих в своих городах заказчиков, которых они не обслуживают.

```
SELECT DISTINCT s.id_sell, s.last_name
FROM customer
JOIN seller s on s.city_sell = city_cust
WHERE NOT EXISTS(SELECT o.id_order FROM `order` o WHERE o.id_sell = s.id_sell AND
→ o.id_customer = customer.id_cust);
```

12. Напишите запрос для извлечения всех продавцов, которые имеют заказчиков с рейтингом 45.

```
SELECT DISTINCT CONCAT_WS(' ', s.last_name, s.first_name, s.second_name) AS 'Продавец'
→ FROM `order` JOIN seller s ON s.id_sell = `order`.id_sell
JOIN customer c ON c.id_cust = id_customer AND c.rate_cust > 45;
```

13. Напишите команду, которая бы увеличила на двадцать процентов комиссионные всех продавцов, имеющих план продаж выше, чем 300.

RESULTS:

<u>id_cust</u>	<u>last_name</u>
1	Петров
3	Берегов
4	Чернов
5	Голубев
6	Романов

Рисунок 13 – Результат запроса № 10

RESULTS:

<u>id_sell</u>	<u>last_name</u>
3	Петухов
4	Терехова
2	Приморская

Рисунок 14 – Результат выполнения запроса № 11

RESULTS:

Продавец

Приморская Алина Петровна
Никифорова Анастасия Петровна
Терехова Нина Семеновна

Рисунок 15 – Результат выполнения запроса № 12

```
UPDATE seller SET comm_sell = comm_sell * 2 WHERE sales_plan > 300;
```

Так как у нас в таблице не было значений комиссионных больше 300, то ничего не изменилось.

14. Напишите команду, которая бы удалила все заказы заданного заказчика из таблицы “Заказы”.

```
DELETE FROM `order` WHERE id_customer = (SELECT id_cust FROM customer WHERE last_name =  
↪ 'Берегов');
```

	id_order	sum_order	date_order	id_sell	id_customer
1	1	50000	2010-06-12 00:00:00	2	2
2	2	30000	2010-06-12 00:00:00	2	2
3	3	70000	2010-06-13 00:00:00	2	1
4	4	60000	2010-06-13 00:00:00	5	1
5	5	35000	2010-06-14 00:00:00	4	5
6	6	70000	2010-06-14 00:00:00	4	3

Рисунок 16 – Таблица до запроса 14

	id_order	sum_order	date_order	id_sell	id_customer
1	1	50000	2010-06-12 00:00:00	2	2
2	2	30000	2010-06-12 00:00:00	2	2
3	3	70000	2010-06-13 00:00:00	2	1
4	4	60000	2010-06-13 00:00:00	5	1
5	5	35000	2010-06-14 00:00:00	4	5

Рисунок 17 – Таблица после запроса 14

15. Продавец X оставил компанию. Переназначьте его заказчиков продавцу Y.

```
UPDATE `order` SET id_sell = (SELECT id_sell FROM seller WHERE last_name = "Приморская")  
↪ WHERE id_sell = 4;
```

	id_order	sum_order	date_order	id_sell	id_customer
1	1	50000	2010-06-12 00:00:00	2	2
2	2	30000	2010-06-12 00:00:00	2	2
3	3	70000	2010-06-13 00:00:00	2	1
4	4	60000	2010-06-13 00:00:00	5	1
5	5	35000	2010-06-14 00:00:00	4	5
6	6	70000	2010-06-14 00:00:00	4	3

Рисунок 18 – Таблица до запроса 15

	id_order	sum_order	date_order	id_sell	id_customer
1	1	50000	2010-06-12 00:00:00	2	2
2	2	30000	2010-06-12 00:00:00	2	2
3	3	70000	2010-06-13 00:00:00	2	1
4	4	60000	2010-06-13 00:00:00	5	1
5	5	35000	2010-06-14 00:00:00	2	5
6	6	70000	2010-06-14 00:00:00	2	3

Рисунок 19 – Таблица после запроса 15

4.1 Индивидуальное задание

Создайте и выполните запросы к БД, созданной в лабораторной работе №1 (Индивидуальные задания). Запросы должны быть следующих типов:

2 запроса с подзапросами;

1. Запрос на нахождения сотрудников одного пола

```
SELECT id AS 'ID', CONCAT_WS(' ', last_name, first_name, second_name) AS 'ФИО' FROM
→ employee
WHERE sex = (SELECT sex.id FROM sex WHERE name = 'Женский');
```

RESULTS:	
ID	ФИО
3	Турсунова Лидия Сергеевна

Рисунок 20 – Результат выполнения первого запроса с подзапросами

2. Запрос на нахождения компонентов, созданных опр. компанией.

```
SELECT * FROM component WHERE manufacturer_id = (SELECT manufacturer.id FROM
→ manufacturer WHERE name = 'AMD');
```

2 запроса на добавление;

1. Добавление нового сотрудника

<code>id</code>	<code>type_id</code>	<code>brand</code>	<code>manufacturer_id</code>	<code>man_Country</code>	<code>release_date</code>	<code>characteristics</code>
1	1	Ryzen 5 3600X	2	2	2022-12-14	Общие параметры\nМодель\nAMD Ryzen 5 3600X\nСокет\nAM4\nКод производителя\n[100-00]
8	2	AAAA	2	2	2020-12-22	AAAA

Рисунок 21 – Результат выполнения второго запроса с подзапросами

```
INSERT INTO employee (last_name, first_name, second_name, date_of_birth, sex, address,
→ phone_number, passport_details,
position_id)
VALUES ('Викторов', 'Кузьма', 'Васильевич', '2004-01-24', 1, 'г. Санкт-Петербург',
→ '+71111111187', '1111 111111', 6);
```

	<code>id</code>	<code>last_name</code>	<code>first_name</code>	<code>second_name</code>	<code>date_of_birth</code>	<code>sex</code>	<code>address</code>	<code>phone_number</code>	<code>passport_detail</code>
1	1	Ланков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Суховский переулок 23	+79627011087	1111 111111
2	2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222 222222
3	3	Турсунова	Лидия	Сергеевна	1980-11-06 12:11:18	2	набережная Кругой реки 149	+711111111111	1234 567891
4	9	Ланков	Вася	Дмитриевич	2006-12-16 12:11:32	1	<null>	+79627011087	1111 444444
5	10	Виктор	Виктор	Виктор	2004-07-02 12:11:41	1	а	+79627011087	1111 111111
6	13	Викторов	Кузьма	Васильевич	2004-01-24 00:00:00	1	г. Санкт-Петербург	+71111111187	1111 111111

Рисунок 22 – Результаты выполнения первого запроса на добавление

2. Добавление новой услуги

```
INSERT INTO service (name, cost) VALUES ('Установка игр', 10000);
```

	<code>id</code>	<code>name</code>	<code>description</code>	<code>cost</code>
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
6	6	Установка игр	<null>	10000

Рисунок 23 – Результаты выполнения второго запроса на добавление

2 запроса на удаление;

1. Удаление сотрудника, с самым коротким адресом

```
DELETE FROM employee WHERE CHAR_LENGTH(address) = (SELECT m FROM (SELECT
→ MIN(CHAR_LENGTH(address)) as m FROM employee) AS cringe);
```

2. В связи законами о браке, запрете ЛГБТ и т.п., удалите все полы, кроме Мужской и Женский.

```
DELETE FROM sex WHERE name NOT IN ('Мужской', 'Женский');
```

2 запроса на изменение;

1. В мире кризис в IT, поэтому уменьшите зарплату всем сотрудника на 40%, но директору увеличьте на 100%.

```
UPDATE position SET salary = IF(name = 'Директор', salary * 2, salary * 0.6);
```

	<code>id</code>	<code>last_name</code>	<code>first_name</code>	<code>second_name</code>	<code>date_of_birth</code>	<code>sex</code>	<code>address</code>	<code>phone_number</code>	<code>passport_details</code>	<code>position_id</code>
1	1	Ланков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Сухойский переулок 23	+79627011087	1111 111111	1
2	2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222 222222	2
3	3	Турсунова	Лидия	Сергеевна	1980-11-06 12:11:18	2	набережная Кроткой реки 149	+71111111111	1234 567891	3
4	9	Ланков	Вася	Дмитриевич	2006-12-16 12:11:32	1 <null>		+79627011087	1111 444444	2
5	10	Виктор	Виктор	Виктор	2004-07-02 12:11:41	1 а		+79627011087	1111 111111	2
6	13	Викторов	Кузьма	Васильевич	2004-01-24 00:00:00	1 г. Санкт-Петербург		+71111111187	1111 111111	6

Рисунок 24 – До первого запроса на удаление

	<code>id</code>	<code>last_name</code>	<code>first_name</code>	<code>second_name</code>	<code>date_of_birth</code>	<code>sex</code>	<code>address</code>	<code>phone_number</code>	<code>passport_details</code>	<code>position_id</code>
1	1	Ланков	Василий	Дмитриевич	2003-12-15 12:10:59	1	Сухойский переулок 23	+79627011087	1111 111111	1
2	2	Гайкин	Кирилл	Денисович	1999-07-17 12:11:08	1	Кобринская улица 5	+79118445162	2222 222222	2
3	3	Турсунова	Лидия	Сергеевна	1980-11-06 12:11:18	2	набережная Кроткой реки 149	+71111111111	1234 567891	3
4	9	Ланков	Вася	Дмитриевич	2006-12-16 12:11:32	1 <null>		+79627011087	1111 444444	2
5	13	Викторов	Кузьма	Васильевич	2004-01-24 00:00:00	1 г. Санкт-Петербург		+71111111187	1111 111111	6

Рисунок 25 – После первого запроса на удаление

	<code>id</code>	<code>name</code>
1	1	Мужской
2	2	Женский
3	3	Неопределённый
4	4	Боевой вертолёт
5	5	Старый динозавр

Рисунок 26 – До второго запроса на удаление

	<code>id</code>	<code>name</code>
1	1	Мужской
2	2	Женский

Рисунок 27 – После второго запроса на удаление

	<code>id</code>	<code>name</code>	<code>salary</code>	<code>responsibilities</code>	<code>requirements</code>
1	1	Директор	500000	Руководить	<null>
2	2	Продавец	40000	Эффективно продавать товары	1. Выполнять скрипты 2. Выполнять нормы продаж 3. Уведомлять покупателей о новых акциях
3	3	Главный менеджер	92000	Следить за работой подчинённых	1. Оформлять кучу чеков 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.

Рисунок 28 – До выполнения первого запроса на изменение

	<code>id</code>	<code>name</code>	<code>salary</code>	<code>responsibilities</code>	<code>requirements</code>
1	1	Директор	1000000	Руководить	<null>
2	2	Продавец	24000	Эффективно продавать товары	1. Выполнять скрипты-2. Выполнять нормы продаж-3. Уведомлять покупателей о новых акциях
3	3	Главный менеджер	55200	Следить за работой подчинённых	1. Формировать кучу отчётов 2. Следить за работой подчинённых
4	5	Стажёр	6000	Учится работать	Должен работать под надзором сотрудника-наставника.
5	6	Младший менеджер	18000	Бумажная работа	Оформляет товары и занимается большей бумажной работой чем главный менеджер.

Рисунок 29 – После выполнения первого запроса на изменение

2. Всё таки закон об замене англицизмов приняли, поэтому ко всем незнакомым нам слова в типах устройств добавить описание "Мы сами не знаем, что это такое кроме слова Мышка.

```
UPDATE type SET description = 'Мы сами не знаем, что это такое' WHERE name != 'Мышка';
```

	<code>id</code>	<code>name</code>	<code>description</code>
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>

Рисунок 30 – До выполнения второго запроса на изменение

2 запроса с использованием статистических функций;

1. Необходимо найти покупателя, который потратил у нас больше всех денег.

```
SELECT CONCAT_WS(' ', c.last_name, c.first_name, c.second_name) as ФИО, SUM(s.cost) +
→ SUM(c2.price) as 'Потраченные деньги'
FROM `order`
JOIN customer c ON c.id = `order`.customer_id
JOIN order_components oc ON `order`.id = oc.order_id
JOIN orders_services os ON `order`.id = os.order_id
JOIN service s ON s.id = os.service_id
JOIN component c2 ON c2.id = oc.component_id
GROUP BY customer_id
ORDER BY SUM(s.cost) + SUM(c2.price) DESC LIMIT 1;
```

	id	name	description
1	1	Процессор	Мы сами не знаем, что это такое
2	2	Видеокарта	Мы сами не знаем, что это такое
3	3	HDD	Мы сами не знаем, что это такое
4	4	SSD	Мы сами не знаем, что это такое
5	5	Корпус	Мы сами не знаем, что это такое
6	6	RGB подсветка	Мы сами не знаем, что это такое
7	7	Мышка	<null>
8	8	Клавиатура	Мы сами не знаем, что это такое
9	9	Геймпад	Мы сами не знаем, что это такое
10	10	Принтер	Мы сами не знаем, что это такое
11	11	Ноутбук	Мы сами не знаем, что это такое

Рисунок 31 – После выполнения второго запроса на изменение

RESULTS:	
ФИО	Потраченные деньги
Гамуйло Сергей Сергеевич	131000

Рисунок 32 – Результат выполнения первого запроса с использованием
стат. функций

2. Найдите среднюю цену всего что есть у нас в магазине

```
SELECT (s_count_sum.s + c_count_sum.su) / (c_count_sum.co + s_count_sum.c) as "Средняя
→ цена всего"
FROM (SELECT SUM(price) su, COUNT(*) co FROM component) c_count_sum join (SELECT
→ SUM(cost) as s, COUNT(*) as c FROM service) s_count_sum ON TRUE;
```

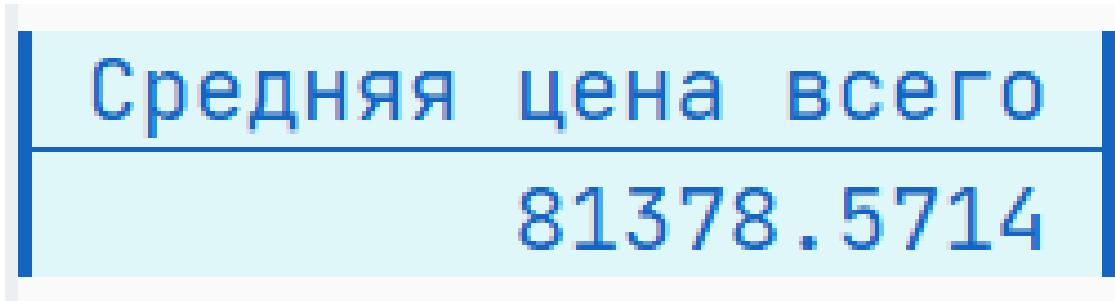


Рисунок 33 – Результат выполнения второго запроса с использованием стат. функций

Число полученное на калькуляторе:

81378.5714286

2 запроса с использованием строковых функций;

1. Необходимо создать запрос, который создаст артикул для каждого компонента, который состоит в виде ID, тип, имя производителя, бренд (1ААА).

```
SELECT CONCAT_WS(' ', t.id, LEFT(t.name, 1), LEFT(m.name, 1), LEFT(brand, 1)) as Артикул
FROM component
JOIN type t on t.id = component.type_id
JOIN manufacturer m on m.id = component.manufacturer_id;
```

2. Сократите фамилии сотрудников следующим образом: Панков -> П4в

```
SELECT CONCAT_WS(' ', LEFT(last_name, 1), CHAR_LENGTH(last_name) - 2, RIGHT(last_name,
→ 1)) AS 'Сокращённые фамилии'
FROM employee;
```

Контрольные вопросы:

1. Синтаксис запроса на добавление

```
INSERT INTO table_name(column) VALUES(col_val);
```

1. Синтаксис запроса на удаление

RESULTS:
Артикул
1ПАР
2ВАР
3HSI
4SKA
5КZi
3НАТ
1ПIi
2ВАА

Рисунок 34 – Результат выполнения первого запроса с использованием строковых функций

RESULTS:
Сокращённые фамилии
П4в
Г4н
Т7а
П4в
В6в

Рисунок 35 – Результат выполнения второго запроса с использованием строковых функций

```
DELETE FROM table_name ;
```

3. Синтаксис запроса на изменение

```
UPDATE table_name SET col = col_val;
```

4. Какие способы соединения таблиц были использованы в работе.

JOIN, а также с помощью условия.

5. Какие строковые функции были использованы в данной работе?

CONCAT_WS, LEFT, CHAR_LENGTH, RIGHT

6. Какие функции для работы с датой и временем были использованы в данной работе.

DAYOFMONTH, MONTH

5 Лабораторная работа № 5 «Процедуры и функции в MySQL»

Цель работы: получение практических навыков при создании и использовании хранимых процедур и функций.

ЗАДАНИЕ 1. Создайте хранимую процедуру с параметрами для вставки записей в таблицу user(поле comment не заполняется)

Код создания процедуры:

```
CREATE DEFINER=current_user PROCEDURE `insert_user`(`username_` varchar(45), `password_` varchar(45))
BEGIN
    INSERT INTO user(username_, password_) VALUES (username_, password_);
END;
```

Код использования процедуры:

```
CALL insert_user('Test', 'Test');
```

	iduser	username	password	comment
1	14	Test	Test	<null>

Рисунок 36 – Результат выполнения процедуры insert_user

ЗАДАНИЕ 2. Создайте хранимую процедуру для вставки записей в таблицу user. Параметром процедуры является пароль. Если длина пароля

меньше 6 символов, то в таблицу user в поле password вставляется пароль, а в поле comment вставляется комментарий – «Короткий», в противном случае вставляется пароль и комментарий - «Нормальный».

Код создания процедуры:

```
CREATE PROCEDURE insert_user_pwd(IN paswrd varchar(45))
BEGIN
    IF CHAR_LENGTH(paswrd) < 6 THEN
        INSERT INTO user(password, `comment`) VALUES (paswrd, 'Короткий');
    ELSE
        INSERT INTO user(password, `comment`) VALUES (paswrd, 'Нормальный');
    END IF;
END;
```

Код использования процедуры:

```
CALL insert_user_pwd("small");
CALL insert_user_pwd("longest");
```

RESULTS:			
iduser	username	password	comment
14	Test	Test	NULL
17	NULL	small	Короткий
18	NULL	longest	Нормальный

Рисунок 37 – Результат выполнения insert_user_pws

ЗАДАНИЕ 3. Используя тип параметра out, создать процедуру, которая возвращает максимальную длину пароля в таблице user.

Код создания процедуры:

```
CREATE PROCEDURE get_max_len_pwd(OUT len integer)
BEGIN
    SELECT MAX(CHAR_LENGTH(password)) INTO len FROM user;
END;
```

Код использования процедуры:

```
SET @max_len = NULL;
CALL get_max_len_pwd(@max_len);
SELECT @max_len;
```

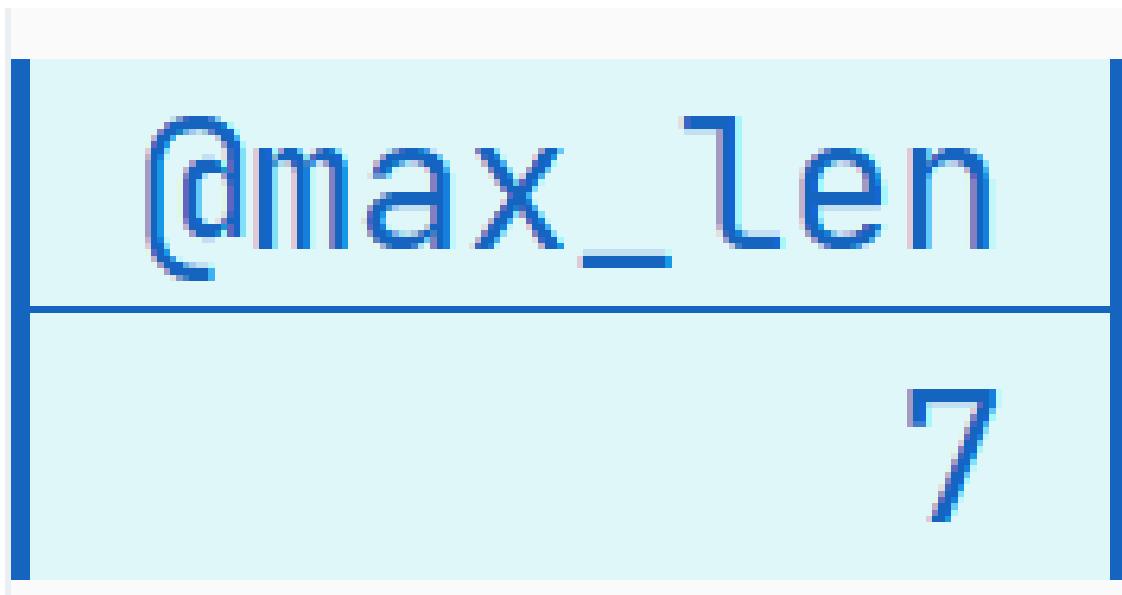


Рисунок 38 – Результат выполнения get_max_len_pwd

ЗАДАНИЕ 4. Создать хранимую функцию, которая вернет количество нормальных паролей.

Код создания функции:

```
CREATE DEFINER=current_user FUNCTION get_count_normal_pwd()
RETURNS int(11) READS SQL DATA
BEGIN
    return (SELECT COUNT(*) FROM user WHERE `comment` = 'Нормальный');
END;
```

Код использования функции:

```
SELECT get_count_normal_pwd();
```

RESULTS:

iduser	username	password	comment
14	Test	Test	NULL
17	NULL	small	Короткий
18	NULL	longest	Нормальный

Рисунок 39 – Исходные данные

ЗАДАНИЕ 5. Создать процедуру, которая удаляет в таблице user пользователя с заданным username.

Код создания процедуры:

RESULTS:

```
get_count_normal_pwd()
```

1

Рисунок 40 – Результат выполнения функции get_count_normal_pwd()

```
CREATE PROCEDURE del_user_by_username(IN username_ varchar(45))
BEGIN
    DELETE FROM user WHERE username = username_;
END;
```

Код вызова процедуры:

```
SET @username_to_del = 'delete_user';
-- call insert_user(@username_to_del, 'delete');
call del_user_by_username(@username_to_del);
```

RESULTS:

iduser	username	password	comment
14	Test	Test	NULL
17	NULL	small	Короткий
18	NULL	longest	Нормальный
21	delete_user	delete	NULL

Рисунок 41 – Таблица user после выполнения insert_user

RESULTS:

iduser	username	password	comment
14	Test	Test	NULL
17	NULL	small	Короткий
18	NULL	longest	Нормальный

Рисунок 42 – Таблица user после выполнения del_user_by_username

5.1 Самостоятельная работа

Используйте запросы, выполненные в лабораторной работе №2 для создания хранимых процедур. Наличие и количество параметров в процедурах необходимо обсудить с преподавателем.

Задание 1. Создайте хранимую процедуру с параметром для выполнения следующего запроса.

```
SELECT * FROM customer WHERE id_cust = 5;
```

Задание 2. Создайте хранимую процедуру с параметром для выполнения следующего запроса.

```
SELECT * FROM customer WHERE rate_cust < 100 AND city_cust != 'Москва';
```

```
SELECT * FROM `order` WHERE MONTH(date_order) = 6 AND (DAYOFMONTH(date_order) = 12  
OR DAYOFMONTH(date_order) = 13);
```