Оглавление

[Предметная область 1](#_Toc165280662)

[Разработка ER диаграммы 5](#_Toc165280663)

[Разработка базы данных 7](#_Toc165280664)

[Создание БД таблицами 7](#_Toc165280665)

[Организация связей между таблицами 8](#_Toc165280666)

[Добавление данных 9](#_Toc165280667)

[Запросы 18](#_Toc165280668)

[Разработка приложения 22](#_Toc165280669)

[Создание ИС 22](#_Toc165280670)

[Подключение БД к ИС 24](#_Toc165280671)

[Сортировка и фильтрация. 29](#_Toc165280672)

[Добавления и удаление позиций в ИС 36](#_Toc165280673)

[Вывод 43](#_Toc165280674)

[ПРИЛОЖЕНИЕ А 44](#_Toc165280675)

[ПРИЛОЖЕНИЕ Б 60](#_Toc165280676)

[ПРИЛОЖЕНИЕ В 71](#_Toc165280677)

[ПРИЛОЖЕНИЕ Г 84](#_Toc165280678)

[ПРИЛОЖЕНИЕ Д 97](#_Toc165280679)

[ПРИЛОЖЕНИЕ E 114](#_Toc165280680)

# Предметная область

Предметная область включает в себя информацию обо всех дошкольных учреждениях города. В базу данных включены все дошкольные учреждения города, список работников детских садов, список групп в детских садах, список детей, сведения о родителях. Каждая таблица в базе данных играет важную роль в хранении и организации информации о дошкольных учреждениях, их работниках, детях и родителях:

1. Personal (Список работников):

В базе данных эта таблица содержит информацию о различных должностях сотрудников в детском саду, ФИО сотрудников, контакты и стаж работы.

Подробное описание должностей сотрудников в детском саду:

* Педагоги: воспитатели, обученные и опытные специалисты, отвечающие за образовательный процесс детей.
* Медицинский персонал: медсестры, отслеживающие здоровье детей и оказывающие первую помощь при необходимости.
* Повара: специалисты по приготовлению питательных и безопасных блюд для детей.
* Административный персонал: сотрудники, занимающиеся организационными вопросами и взаимодействием с родителями.
* Уборщицы: работники, поддерживающие чистоту и порядок в помещениях детского сада.
* Директоры: руководители детского сада, отвечающие за общее управление и развитие учреждения.
* Психологи: специалисты, помогающие детям и их семьям в решении эмоциональных и социальных проблем.
* Помощники воспитателей: сотрудники, поддерживающие воспитателей в их ежедневной работе с детьми.

Kindergartens (Детские сады):

Здесь содержится основная информация о каждом детском саде, такая как:

* Название детского сада: уникальное наименование учреждения.
* Адрес и контактная информация: местоположение и реквизиты для связи с садом.
* Директор: руководитель учреждения, ответственный за его функционирование и развитие.

Parents (Сведения о родителях):

В этой таблице содержится информация о родителях или законных опекунах детей, которая включает в себя:

* Информация о родителях: ФИО, контактные данные, данные о месте работы и другие релевантные сведения.
* Инструкции и предпочтения: индивидуальные пожелания и требования родителей относительно ухода и воспитания их детей в детском саду.

Эффективное управление этой информацией позволяет обеспечить безопасность, эффективное обучение и взаимодействие с родителями в дошкольных учреждениях.

Childrens (Списки детей):

В этой таблице хранятся данные о каждом ребенке, включенном в определенную группу, которая включает в себя:

* Персональную информацию: полное имя, дата рождения, место рождения, данные о родителях.
* Состояние здоровья: информация о физическом и эмоциональном здоровье, аллергии, медицинские особенности.
* Информацию о входе в сад: дата поступления.

Groups (Группы в детских садах):

Эта таблица содержит информацию о каждой группе детей в детском саду. Сюда входит:

* Название групп: наименование каждой группы детского сада.
* Уровень группы: отличительные особенности, связанные с возрастом или образовательными целями.
* Связь с другими группами: информация о передаче детей из одной группы в другую по мере их роста и развития.

Детский сад играет огромную роль в формировании личности ребенка, подготовке его к школе, развитии навыков самостоятельности и социализации.

В детском саду осуществляется комплексное воспитание и образование детей. Педагоги создают специальные условия для физического, эмоционального, социального и интеллектуального развития детей.

Детский сад является местом, где дети впервые начинают общаться со сверстниками и взрослыми, учатся соблюдать правила поведения, развивают навыки социального взаимодействия. Дети занимаются различными видами творчества (рисование, лепка, музыка, танцы и т. д.), что способствует развитию их творческого потенциала.

# Разработка ER диаграммы

1. Добавляем все таблицы – тема «БД детского сада» включающая в себя несколько таблиц для хранения информации о персонале, родителях, детях, группах и самих детских садах.

* аутентификация администратора: Таблица для аутентификации администраторов системы. Содержит информацию о сотрудниках, их учетных данных (имя пользователя и пароль).
* дети: В этой таблице хранится информация о детях, посещающих детский сад. Включает данные о ребенке (ФИО, дата рождения), информацию о родителе (ID\_родителя), дату поступления в сад и принадлежность к группе.
* группа детей: Содержит информацию о группах детского сада. Каждая группа имеет свой уникальный идентификатор (ID\_группы), принадлежит определенному саду (ID\_сада), имеет название, уровень и может иметь связь с старшей группой.
* Сад: В этой таблице хранится информация о детских садах. Включает уникальный идентификатор сада (ID\_сада), данные о директоре (ID\_директора), название, адрес и контактные данные сада.
* родители: Содержит информацию о родителях детей. Каждый родитель имеет уникальный идентификатор (ID\_родителя), ФИО, контактные данные и дополнительную информацию.
* персонал: Таблица для хранения данных о сотрудниках детского сада. Включает в себя уникальный идентификатор сотрудника (ID\_сотрудника), ФИО, должность, контактные данные и опыт работы.

2. Проводим логический анализ, какими связями будут соединены таблицы.

3. Строим связи между таблицами



Рисунок 1. ERD диаграмма

# Разработка базы данных

## Создание БД таблицами

Создаем запрос на создание базы данных Kindergartens1 с таблицами Admin\_Auth, Childrens, Groups, Kindergartens, Parents, Personal

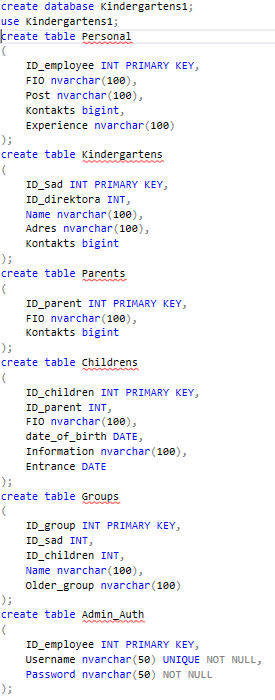


Рисунок 2. Запрос на создание базы данных с таблицами

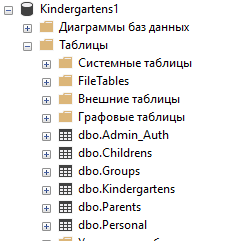


Рисунок 3. Результат запросов

## Организация связей между таблицами

В таблице Admin\_Auth:

* Основной ключ - ID\_employee. Связан с таблицей Personal по полю ID\_employee.

В Таблица Childrens:

* Внешний ключ - ID\_parent, связан с таблицей Parents по полю ID\_parent.
* Внешний ключ - ID\_group, связан с таблицей Groups по полю ID\_group.

В Таблица Groups:

* Внешний ключ - ID\_sad, связан с таблицей Kindergartens по полю ID\_Sad.
* Внешний ключ - Older\_group, связан с таблицей Personal по полю ID\_employee.

В Таблица Kindergartens:

* Внешний ключ - ID\_direktora, связан с таблицей Personal по полю ID\_employee.

В результате 6 внешних ключей, связывающих таблицы

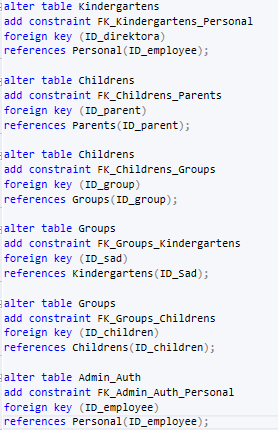


Рисунок 3. Запрос на создание внешних ключей

## Добавление данных

Для таблицы таблицы Personal (Персонал) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Personal FIO, Post, Kontakts, Experience) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 23 строк, значит будет 23 скобок с данными.

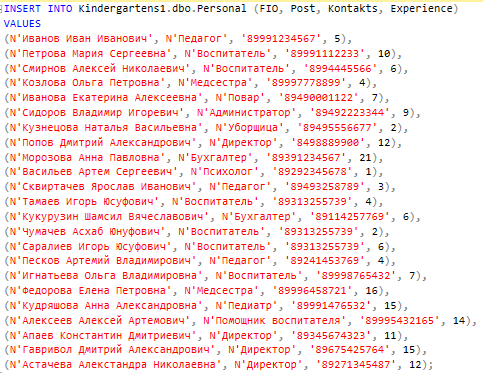


Рисунок 4. Запрос на добавление данных в таблицу Personal

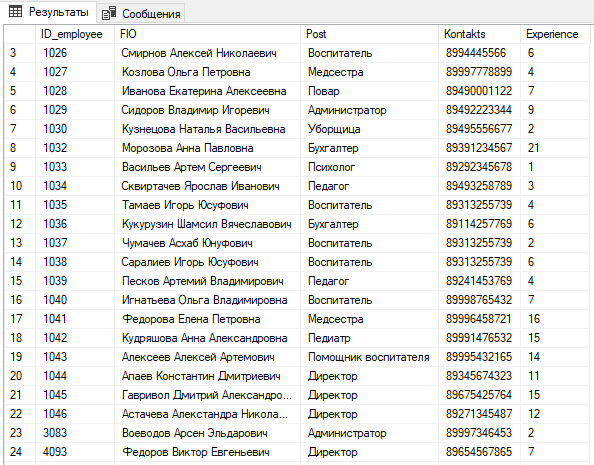
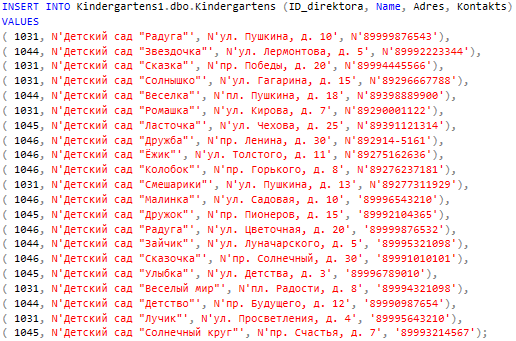
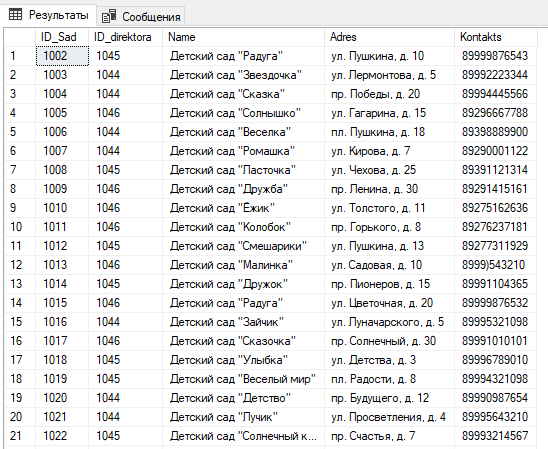


Рисунок 5. Результат добавления данных в таблицу Personal

Для таблицы таблицы Kindergartens (Сады) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Kindergartens (ID\_direktora, Name, Adres, Kontakts) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 21 строк, значит будет 21 скобок с данными.

Рисунок 6. Запрос на добавление данных в таблицу Kindergartens

Рисунок 7. Результат добавления данных в таблицу Kindergartens

Для таблицы таблицы Parents (Родители) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Parents (ID\_parent, FIO, Kontakts, Informations) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 20 строк, значит будет 20 скобок с данными.

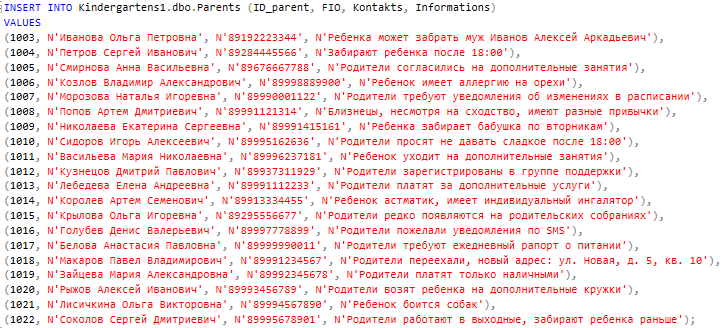


Рисунок 8. Запрос на добавление данных в таблицу Parents

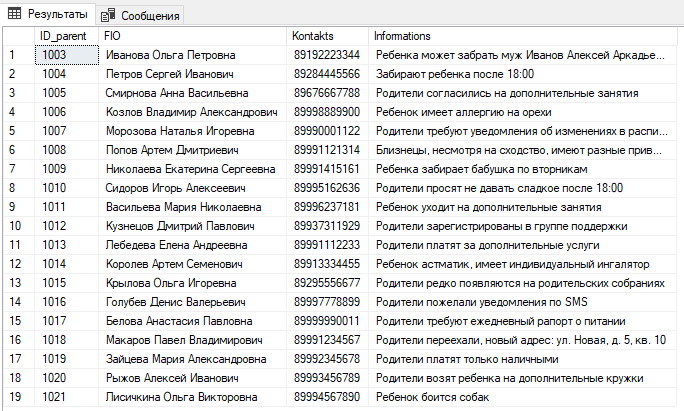


Рисунок 9. Результат добавления данных в таблицу Parents

Для таблицы таблицы Childrens (Дети) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Childrens ( ID\_parent, FIO, date\_of\_birth, Information, Entrance) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 40 строк, значит будет 40 скобок с данными.



Рисунок 10. Запрос на добавление данных в таблицу Childrens



Рисунок 11. Результат добавления данных в таблицу Childrens

Для таблицы таблицы Groups (Группы) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Groups ( ID\_sad, Name, Level\_group, Older\_group) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 20 строк, значит будет 20 скобок с данными.

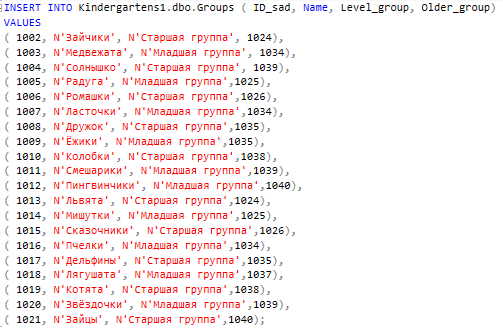


Рисунок 12. Запрос на добавление данных в таблицу Groups

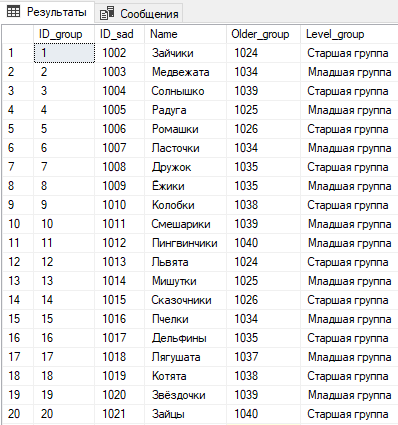


Рисунок 13. Результат добавления данных в таблицу Groups

Для таблицы таблицы Admin\_Auth (аутентификация администратора) мы используем конструкцию:

INSERT INTO Kindergartens1.dbo.Groups (ID\_employee, Username, Password) VALUES (прописываем одну строку нужных данных) и так скобок пишем столько, сколько требуется. В данном случае нам нужно 2 строк, значит будет 2 скобок с данными

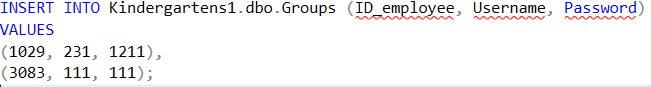


Рисунок 14. Запрос на добавление данных в таблицу Admin\_Auth

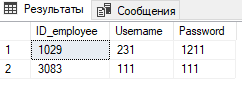


Рисунок 15. Результат добавления данных в таблицу Admin\_Auth

# Запросы

1. запрос на выборку по таблице «Списки детей» и просмотреть всех детей  
   с фамилией на букву «П». Выполняем с помощью запроса: SELECT \* FROM Childrens WHERE FIO LIKE 'П%'

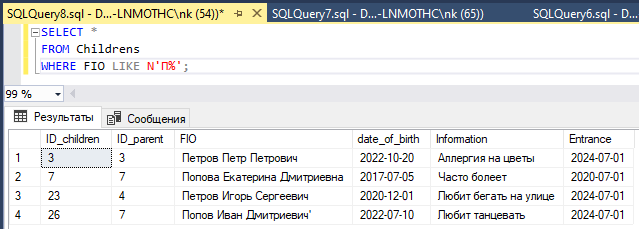


Рисунок 16. Результат выполнение первого запроса

1. запрос на выборку по таблице «Списки детей» и просмотреть всех  
   детей, у которых фамилия начинается на букву «В», «П», «С». Выполняем с помощью запроса: SELECT \* FROM Childrens WHERE FIO LIKE ' С%' OR FIO LIKE ' В %' OR FIO LIKE ' П %'

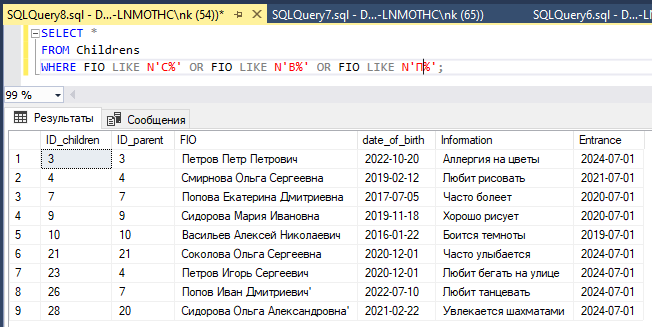


Рисунок 17. Результат выполнение второго запроса

1. запрос на выборку по таблице «Работники дет. сада» и просмотреть всех работников с фамилией на букву «М» и «С» у которых стаж работы более 5 лет. Выполняем с помощью запроса: SELECT \* FROM Personal WHERE FIO LIKE 'М%' OR FIO LIKE 'С%' AND Experience > 5

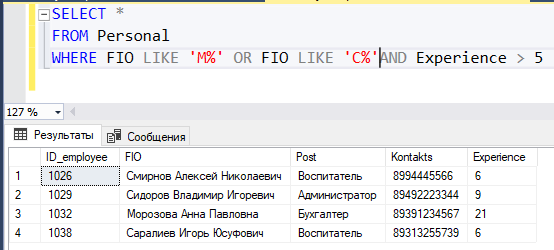


Рисунок 18. Результат выполнение третьего запроса

1. Создать запрос, которое будет хранить список детей, которые  
   поступили в текущем году, и количество детей было не менее 30. Выполняем с помощью запроса: SELECT \* FROM [Kindergartens1].dbo.Childrens WHERE YEAR(Entrance) = 2024

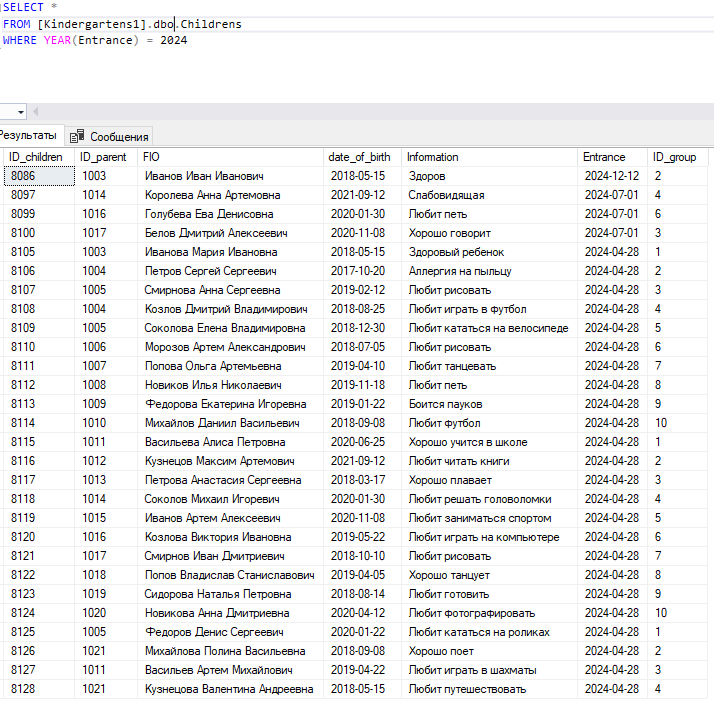


Рисунок 19. Результат выполнение четвертого запроса

1. Создать запрос, которое будет выводить список родителей, у которых номер телефона начинается на «+7904». Выполняем с помощью запроса: SELECT \* FROM [dbo].[Parents] WHERE [Kontakts] LIKE '890%'

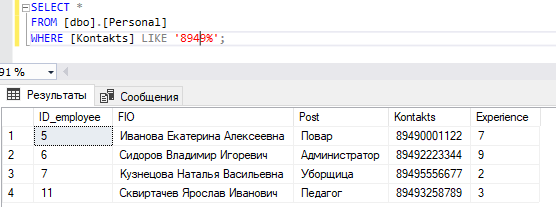


Рисунок 20. Результат выполнение пятого запроса

# Разработка приложения

## Создание ИС

1.Заходим в Visual Studio

2.На этапе создания проекта, выбираем Windows Forms

3.Присваиваем имя проекту

4.Добавляем фон, button, label и 2 textbox на главную форму

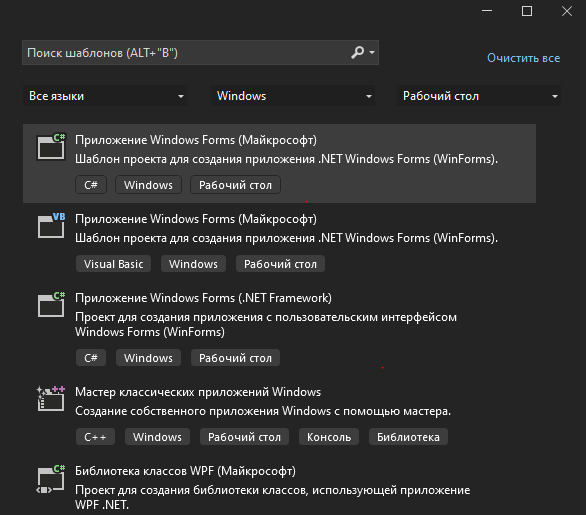


Рисунок 21. Выбор шаблона

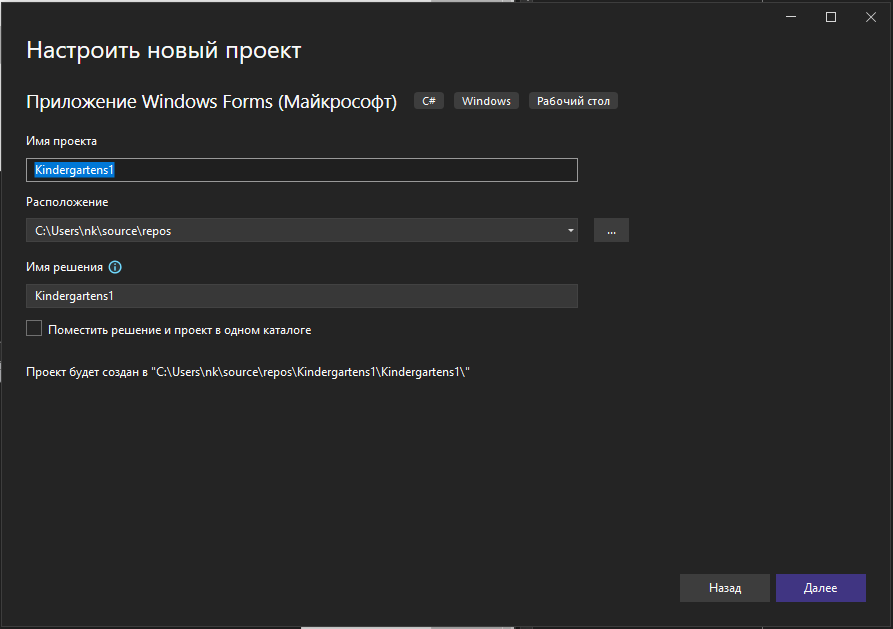


Рисунок 22. Присваиваем имя проекту

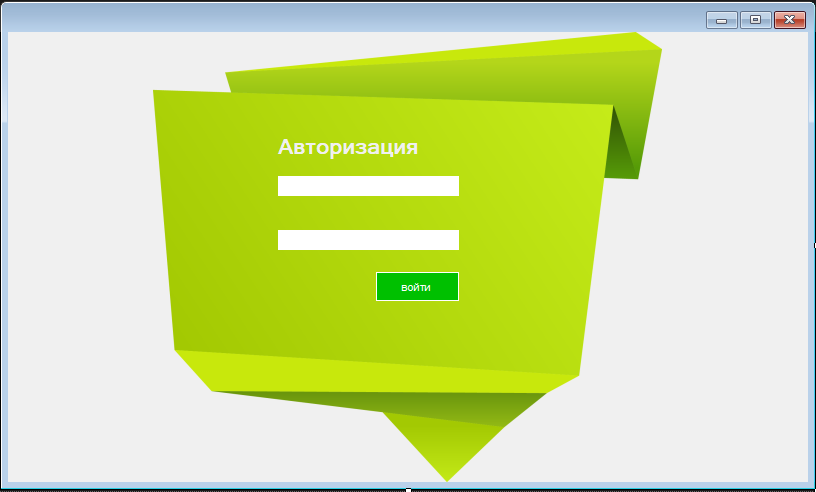


Рисунок 23. Добавляем на первую форму элементы и фон

## Подключение БД к ИС

1.Открываем вкладку «Источник данных»

2.Нажимаем «добавить новый источник данных»

3.Выбираем тип источника – базы данных

4.Указываем путь до базы

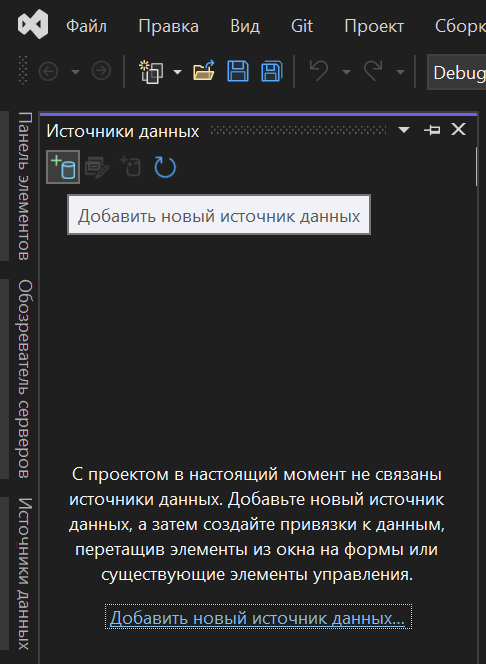


Рисунок 24. Добавление источника данных.

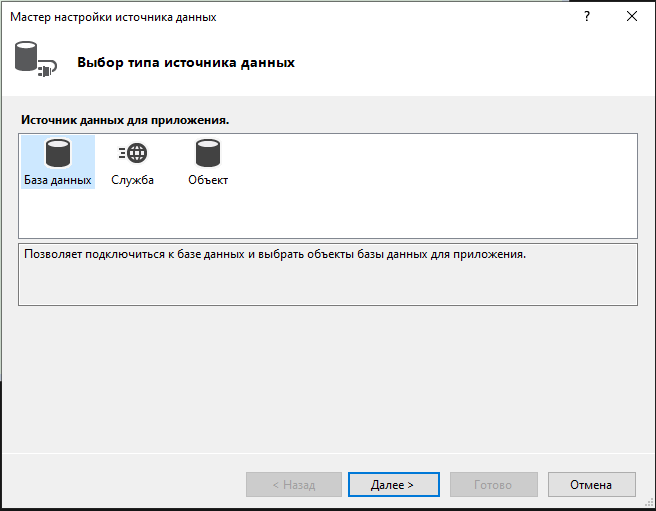


Рисунок 25. Выбор типа источника данных

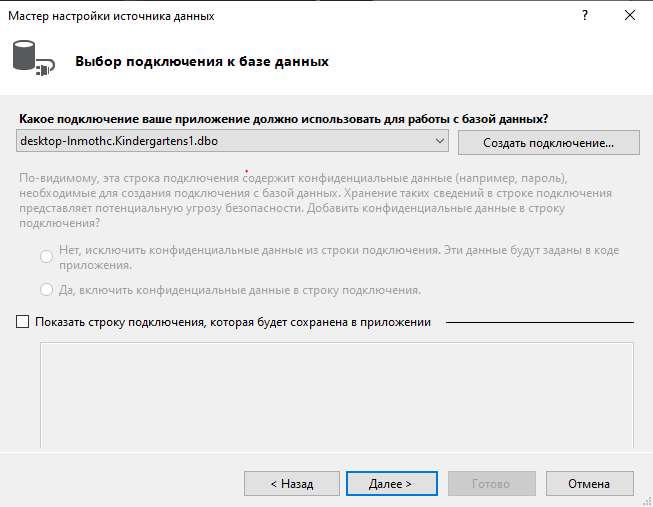


Рисунок 26. Указание пути до базы данных

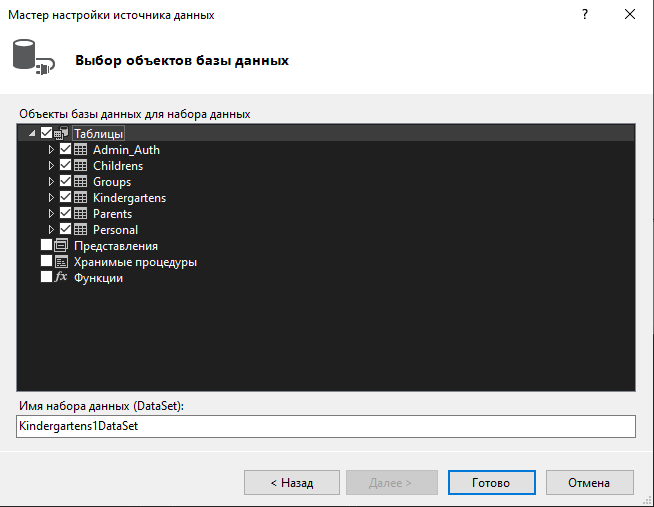


Рисунок 27. Выбор таблиц базы данных

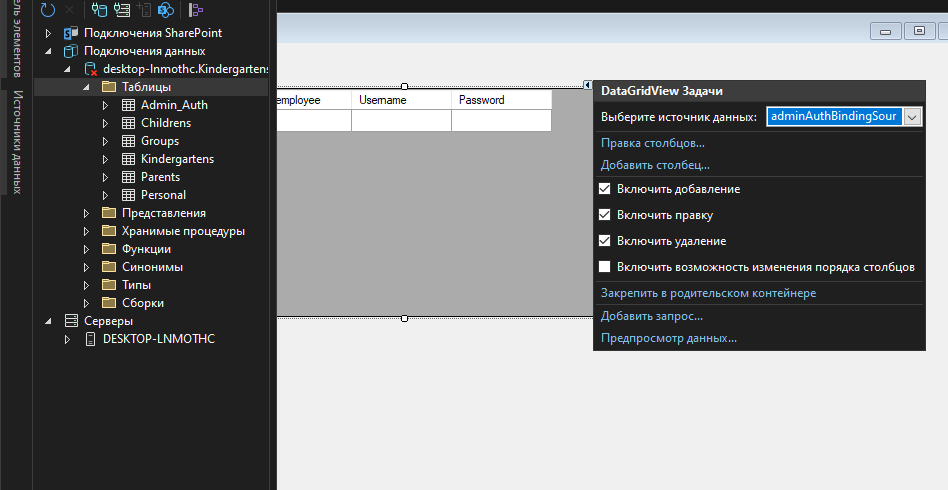


Рисунок 28. Выбор источника данных Admin\_Auth в DataGridView

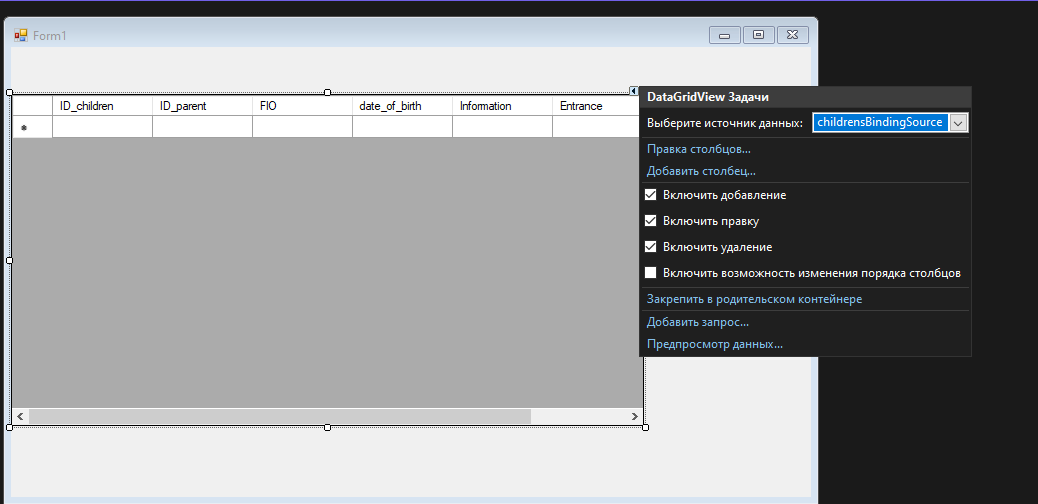


Рисунок 29. Добавление источника данных таблицы Childrens

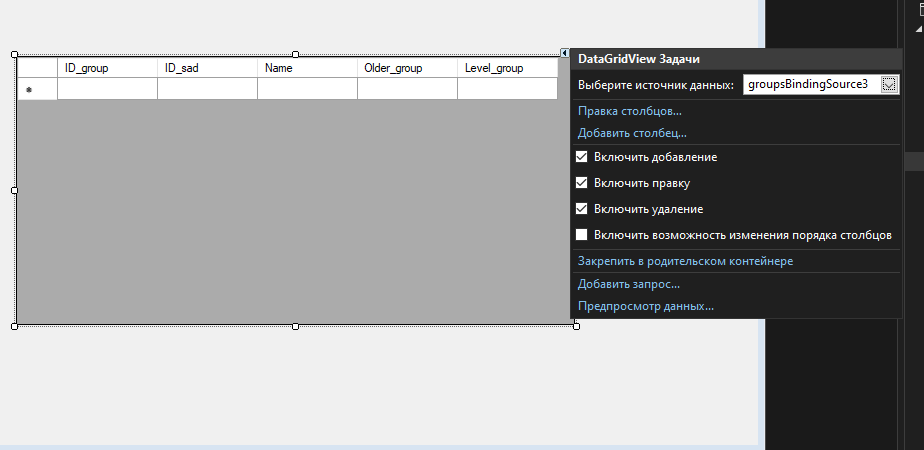


Рисунок 30. Добавление источника данных таблицы Groups

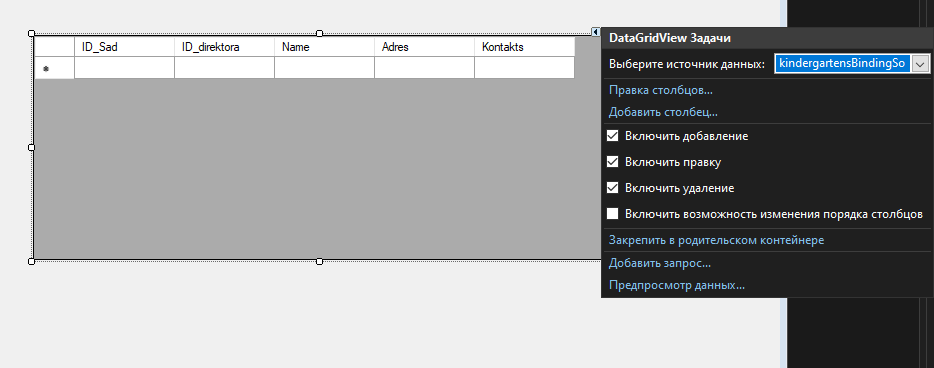


Рисунок 31. Добавление источника данных таблицы Kindergartens

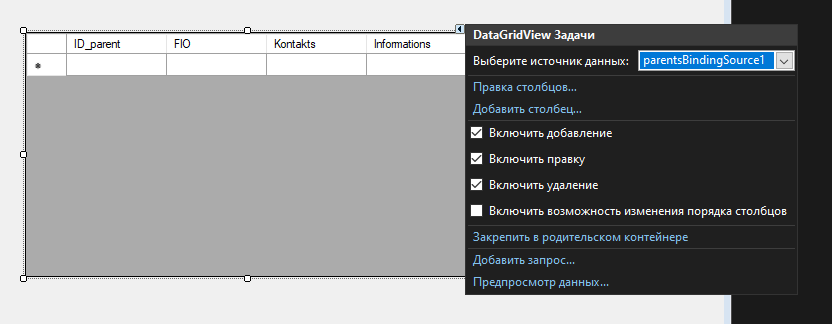


Рисунок 32. Добавление источника данных таблицы Parents

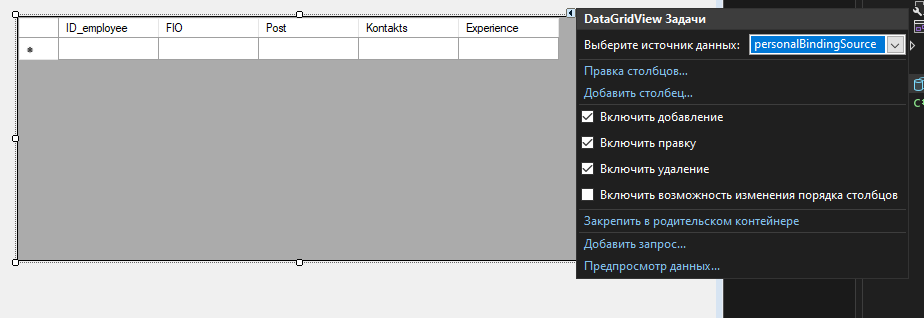


Рисунок 33. Добавление источника данных таблицы Personal

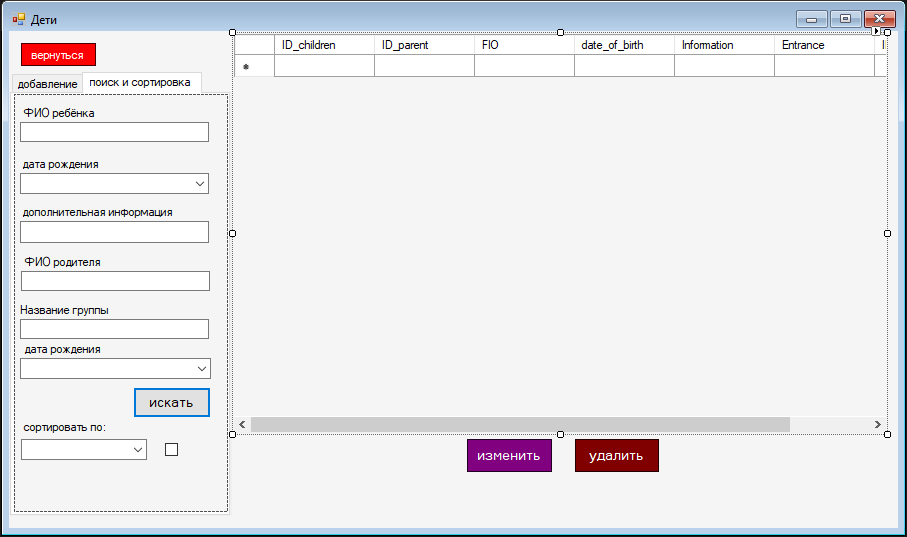


Рисунок 34. Добавление элементов на форму

## Сортировка и фильтрация.

1. Добавляем combobox и checkbox на форму для создания фильтрации
2. Добавляем функционал на элементы
3. Выбираем в combobox нужную таблицу, нажимаем на checkbox, сортировка будет работать от большего к меньшему и на оборот

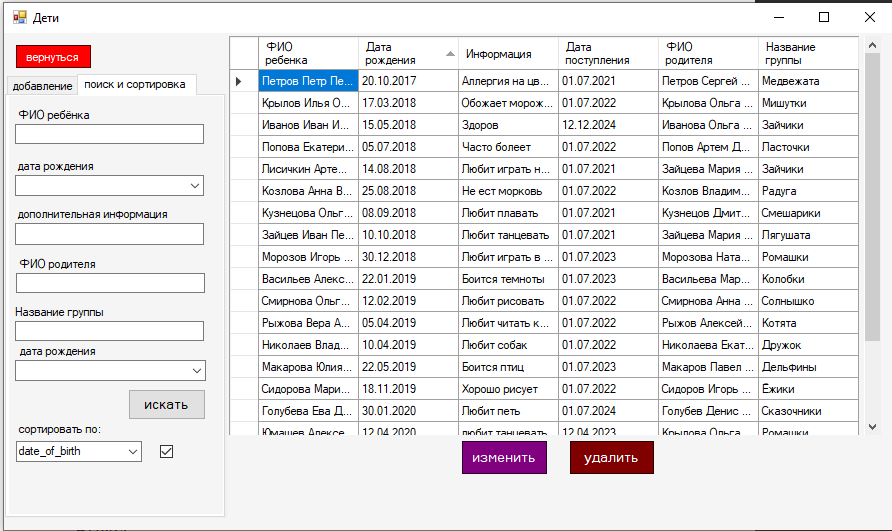


Рисунок 35. Сортировка в форме Дети по дате рождения от большего у меньшему

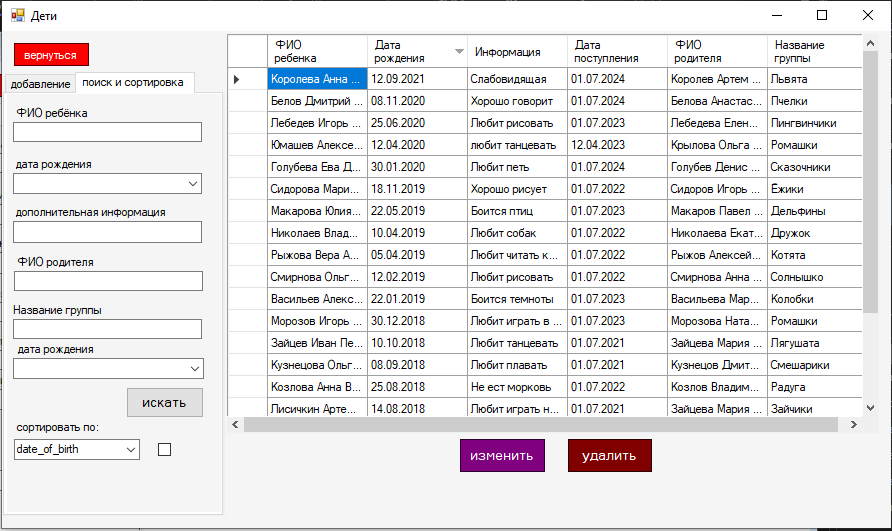


Рисунок 36. Сортировка в форме Дети по дате рождения от меньшего к большему

Код см. в ПРИЛОЖЕНИИ А

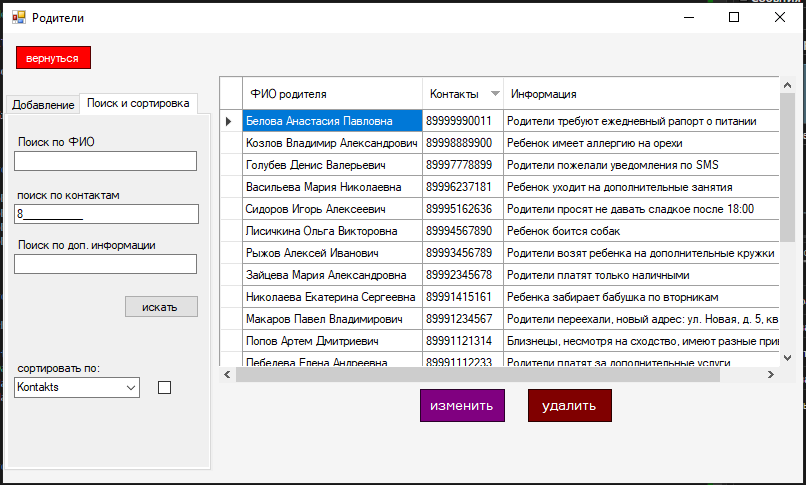


Рисунок 37. Сортировка в форме Родители по номеру телефона от меньшего к большему

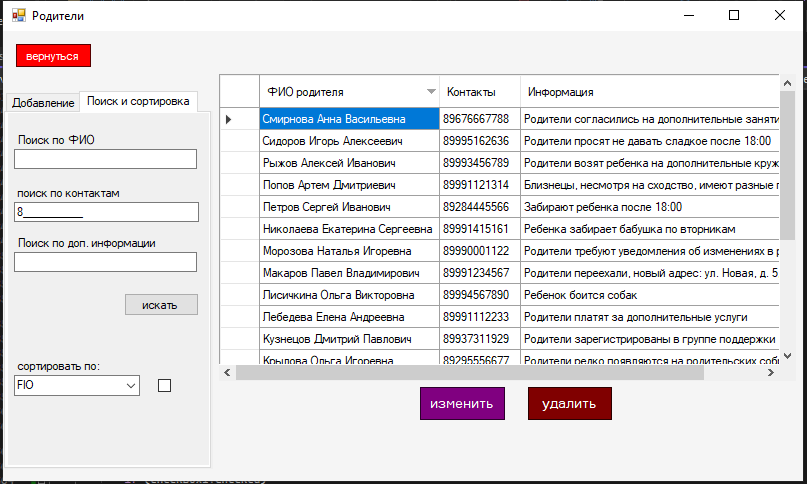


Рисунок 38 Сортировка в форме Родители по номеру телефона от большего к меньшему

Код см. в ПРИЛОЖЕНИИ Б

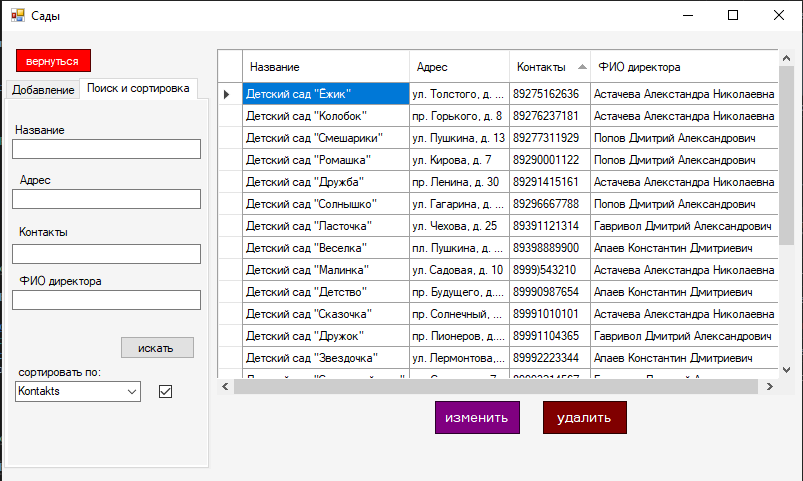


Рисунок 39. Сортировка в форме Сад по номеру телефона от меньшего к большему

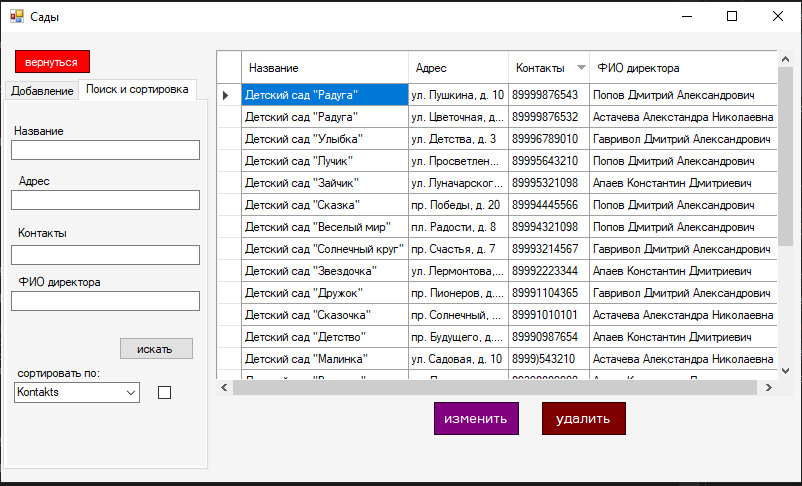


Рисунок 40. Сортировка в форме Сад по номеру телефона от большего к меньшему

Код см. в ПРИЛОЖЕНИИ В

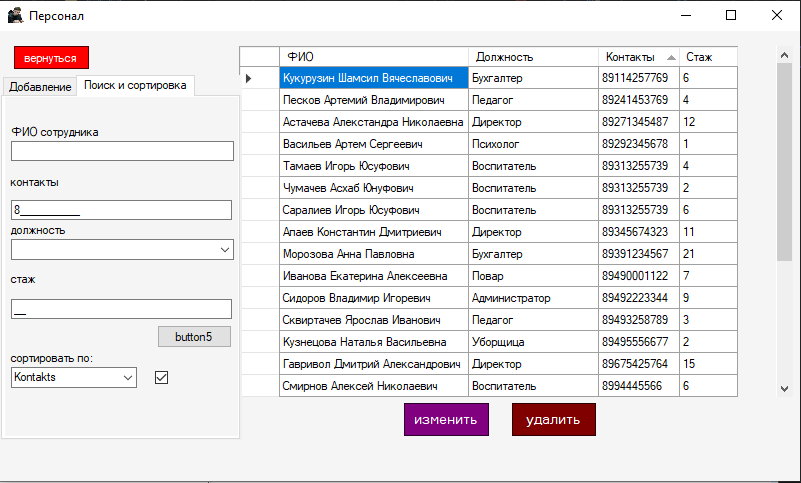


Рисунок 41. Сортировка в форме Персонал по номеру телефона от меньшего к большему

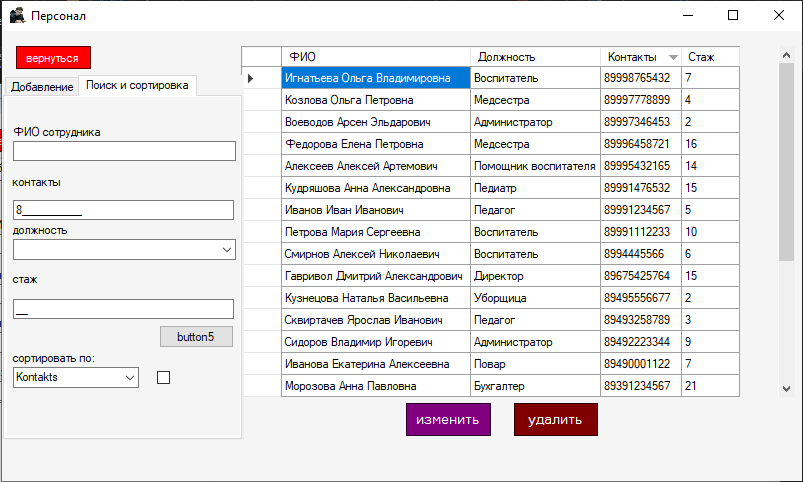


Рисунок 42. Сортировка в форме Персонал по номеру телефона от большего к меньшему

Код см. в ПРИЛОЖЕНИИ Г

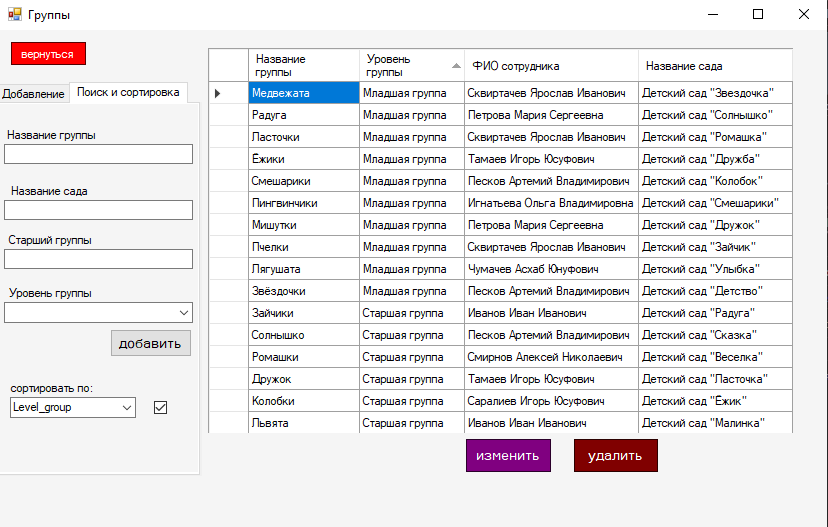


Рисунок 43. Сортировка в форме Группы по уровню группы от младшей к старшей

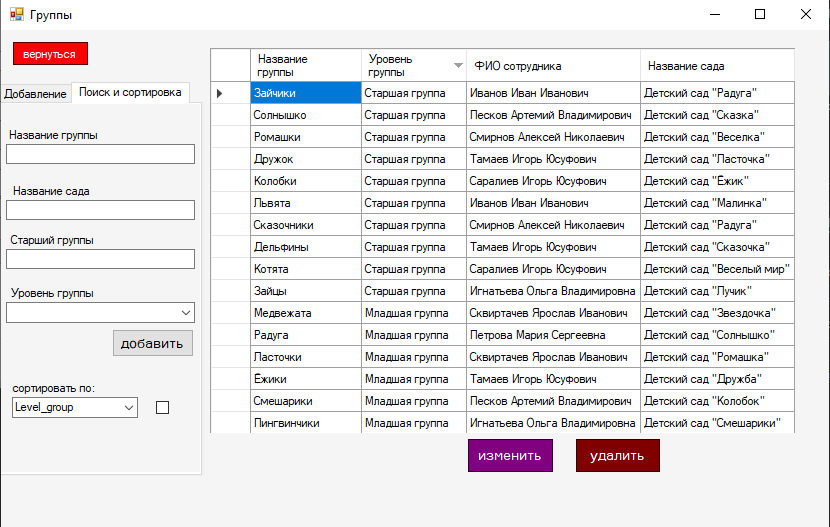


Рисунок 44. Сортировка в форме Группы по уровню группы от старшей к младшей

Код см. в ПРИЛОЖЕНИИ Д

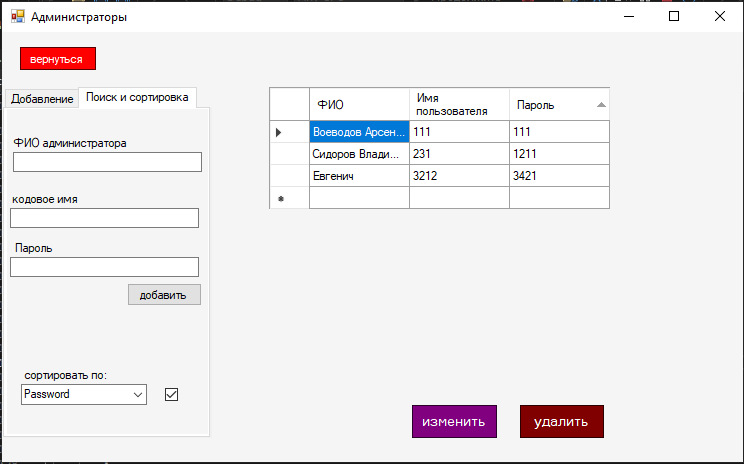


Рисунок 45. Сортировка в форме Администраторы по паролю от меньшего к большему

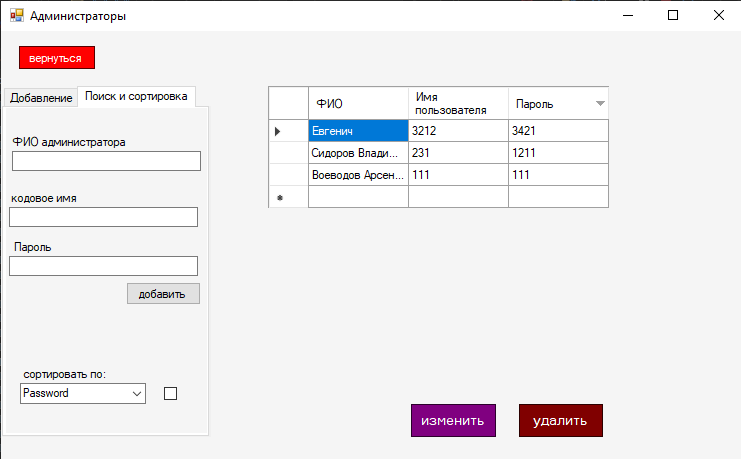


Рисунок 46. Сортировка в форме Администраторы по паролю от большего к меньшему

Код см. в ПРИЛОЖЕНИИ Е

## 

## Добавления и удаление позиций в ИС

1. Добавляем 2 button на форму для добавления и удаления
2. Добавляем функционал на элементы
3. Записываем данные в имеющиеся textbox, нажимаем кнопку сохранить, данные появляются в datagridview и сохраняются в БД
4. Выбираем нужную строку в datagridview и нажимаем удалить, строка удаляется в таблице и БД

При нажатии на удаление выскакивает табличка о подтверждении

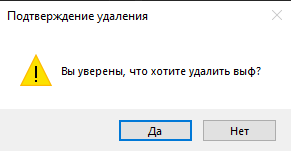


Рисунок 47. Подтверждение удаления

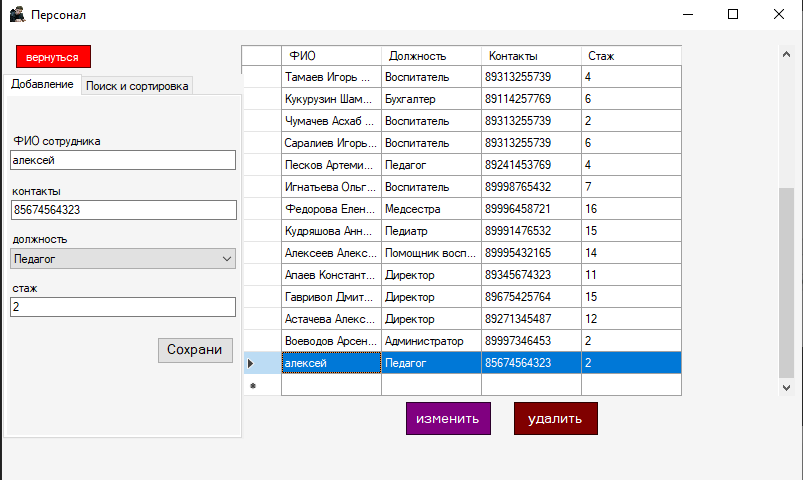


Рисунок 48. Проверка добавление персонала в форме Персонал

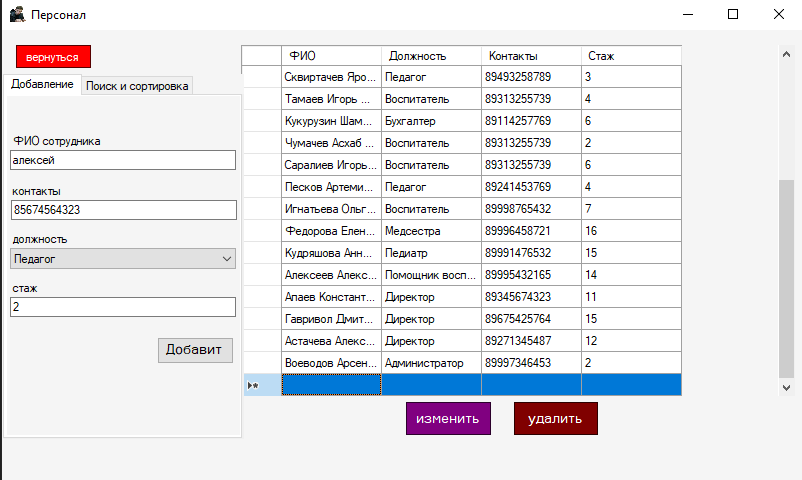


Рисунок 49. Проверка удаления персонала в форме Персонал

Код см. в ПРИЛОЖЕНИИ Г

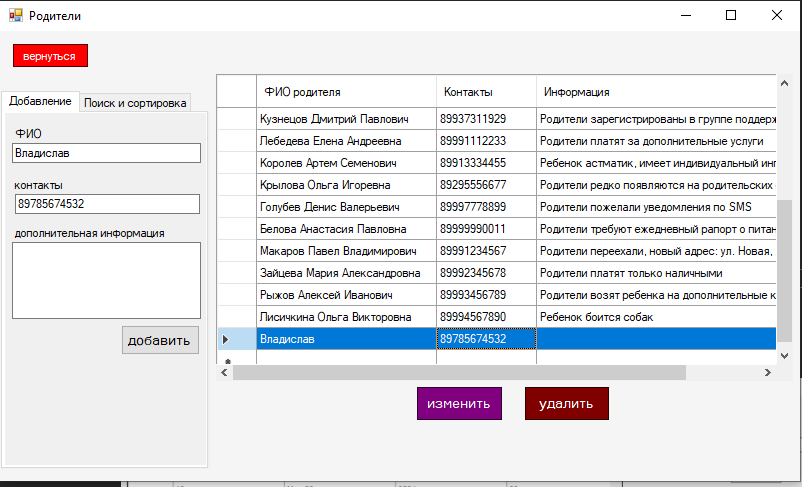


Рисунок 50 Проверка добавления родителя в форме Родители

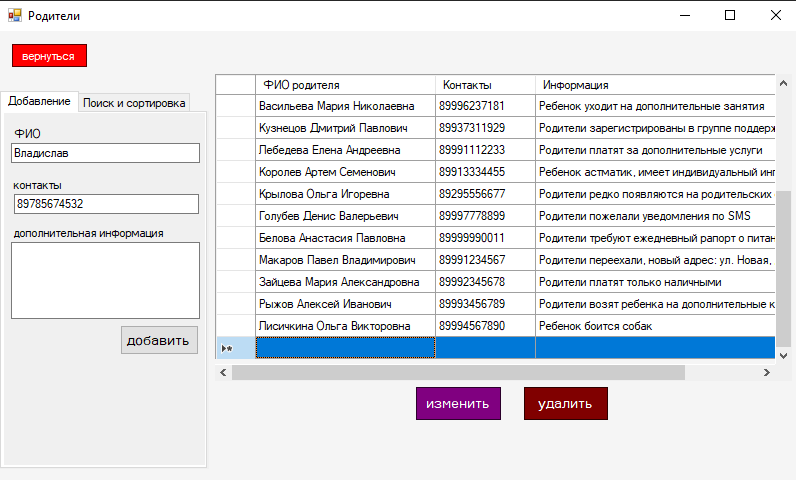


Рисунок 51. Проверка удаления родителя в форме Родители

Код см. в ПРИЛОЖЕНИИ Б

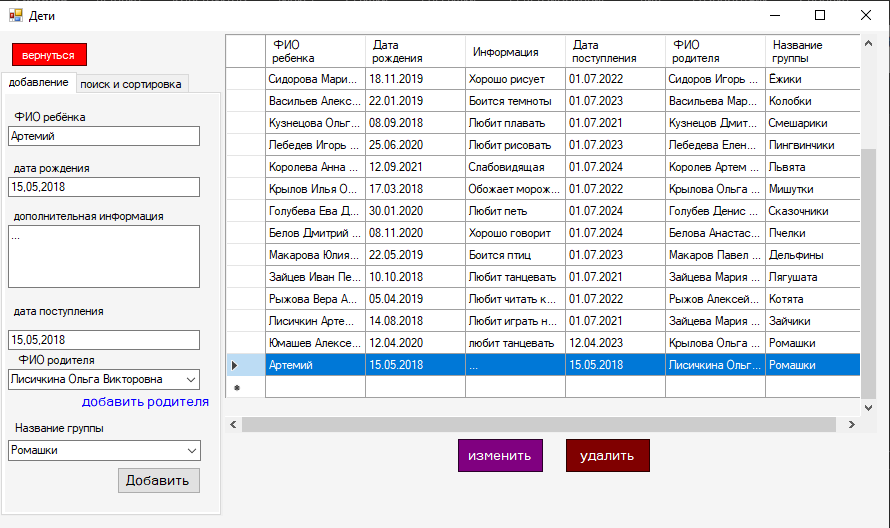


Рисунок 52. Проверка добавления ребенка в форме Дети

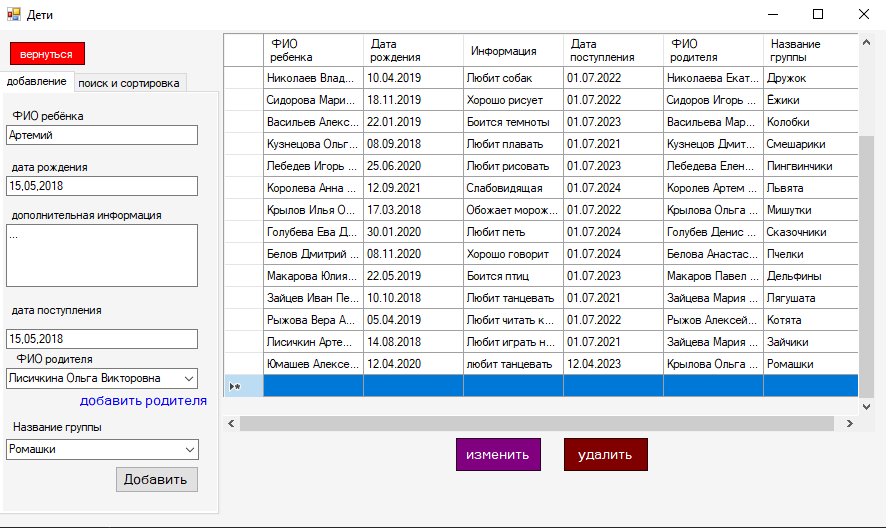


Рисунок 53. Проверка удаления ребенка в форме Дети

Код см. в ПРИЛОЖЕНИИ А

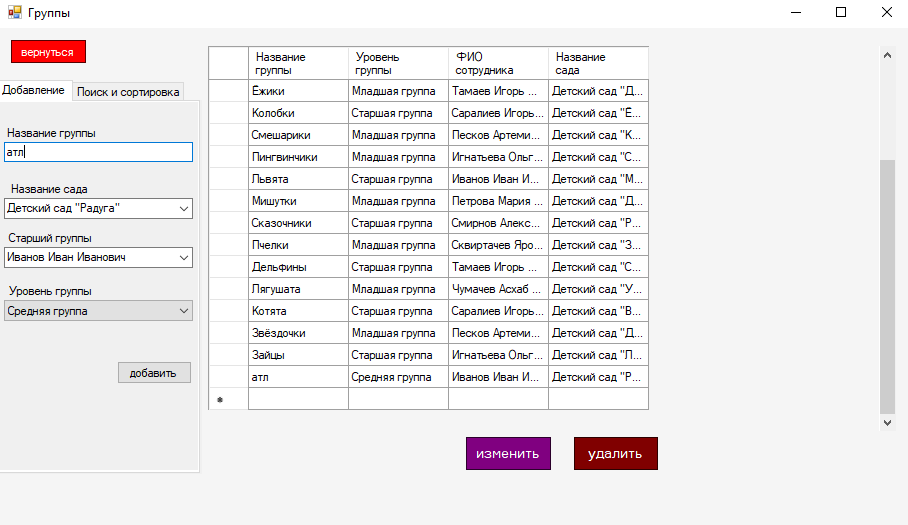


Рисунок 54. Проверка добавления группы в форме Группы

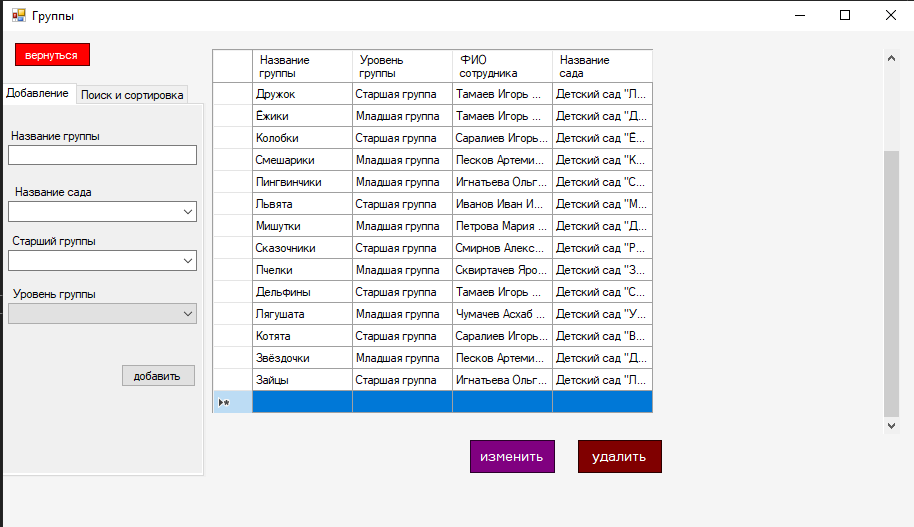


Рисунок 55. Проверка удаления группы в форме Группы

Код добавления и удаления в ПРИЛОЖЕНИИ Д

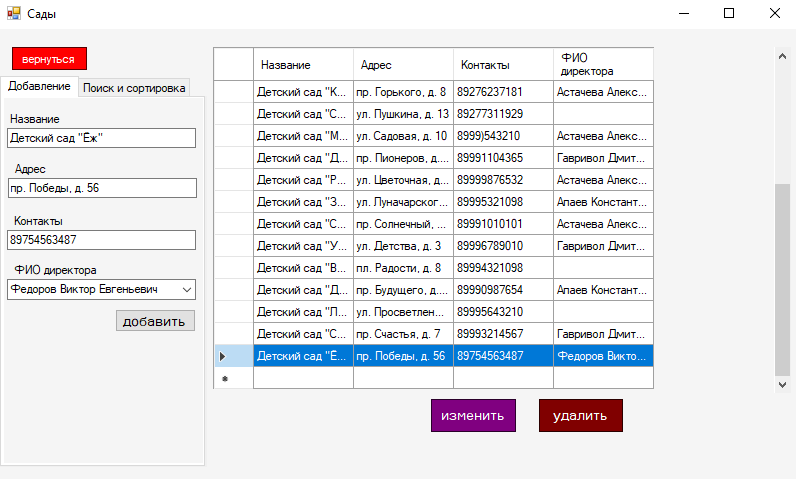


Рисунок 56. Проверка добавления сада в форме Сады

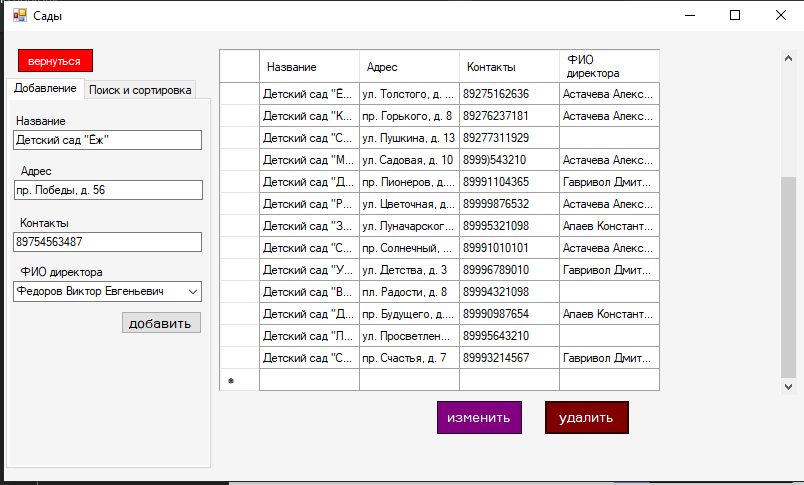


Рисунок 57. Проверка удаления сада в форме Сады

Код добавления и удаления в ПРИЛОЖЕНИИ В

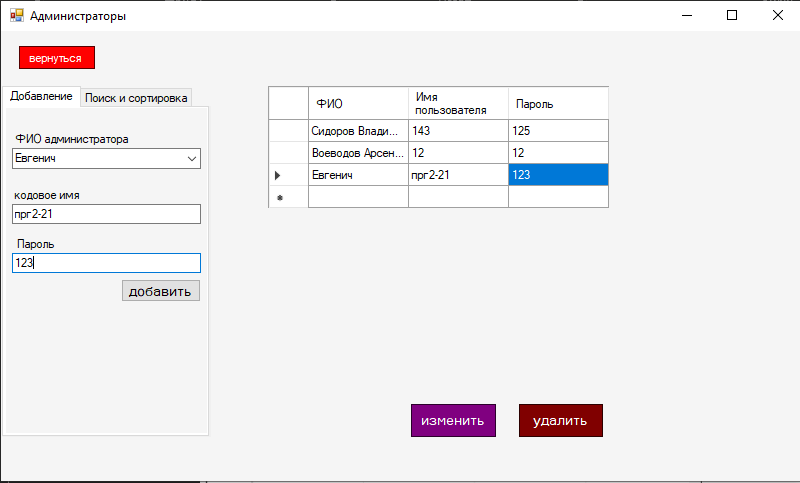


Рисунок 58. Проверка добавления администратора в форме Администраторы

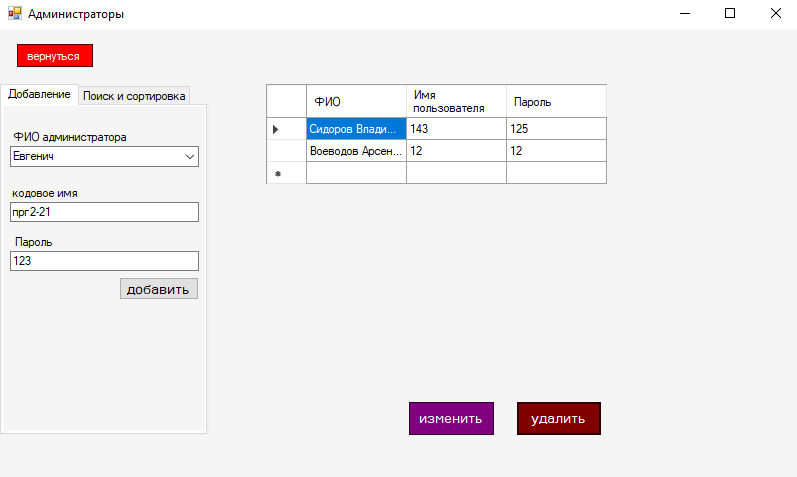


Рисунок 59. Проверка удаления администратора в форме Администраторы

Код добавления и удаления в ПРИЛОЖЕНИИ Е

# Вывод

Разработанное приложение представляет собой инструмент для управления информацией о детях, их родителях, персонале, детских садах, группах и администраторах этих садов. Благодаря нему стало возможным эффективное хранение данных, а также выполнение операций добавления, изменения, и удаления информации.

Также оно позволяет вносить и редактировать данные о родителях, включая контактную информацию и особые запросы по воспитанию детей, что способствует лучшему взаимодействию между детским садом и родителями.

Система также включает возможность управления информацией о персонале детского сада, включая педагогический и вспомогательный персонал, их контактные данные, стаж работы и профессиональные навыки.

Благодаря функциональности по управлению данными о детских садах, включая их местоположение, контактную информацию, а также особенности организации и инфраструктуры, приложение облегчает процесс поиска и выбора детского сада для родителей.

Также приложение предоставляет возможность управления информацией о группах детей в детских садах, включая их состав, режим занятий, а также связи с персоналом.

Система позволяет управлять данными об администраторах детских садов, включая информацию о правах доступа и административных функциях.

В целом, разработанное приложение значительно упрощает и оптимизирует управление информацией в детских садах, что способствует повышению эффективности и качества воспитательного процесса и создает более комфортные условия для детей и их родителей.

ПРИЛОЖЕНИЕ А

Childrens

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace kyrsovaia

{

public partial class Childrens : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public Childrens()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

LoadChildrenData();

FillComboBox3();

dataGridView1.Columns["FIO\_children"].HeaderText = "ФИО ребенка";

dataGridView1.Columns["date\_of\_birth"].HeaderText = "Дата рождения";

dataGridView1.Columns["Information"].HeaderText = "Информация";

dataGridView1.Columns["Entrance"].HeaderText = "Дата поступления";

dataGridView1.Columns["FIO\_parent"].HeaderText = "ФИО родителя";

dataGridView1.Columns["Group\_Name"].HeaderText = "Название группы";

}

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

{

this.groupsTableAdapter.Fill(this.kindergartens1DataSet.Groups);

this.childrensTableAdapter.Fill(this.kindergartens1DataSet.Childrens);

LoadParentsData();

LoadGroupsData();

}

private void LoadChildrenData()

{

string query = "SELECT c.FIO AS FIO\_children, c.date\_of\_birth, c.Information, c.Entrance, p.FIO AS FIO\_parent, g.Name AS Group\_Name " +

"FROM Childrens c " +

"LEFT JOIN Parents p ON c.ID\_parent = p.id\_parent " +

"LEFT JOIN Groups g ON c.ID\_group = g.ID\_group";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

FillYearsComboBoxes();

}

private void LoadParentsData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT id\_parent, FIO FROM Parents";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable parentsTable = new DataTable();

try

{

connection.Open();

adapter.Fill(parentsTable);

comboBox1.DataSource = parentsTable;

comboBox1.DisplayMember = "FIO";

comboBox1.ValueMember = "id\_parent";

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о родителях: " + ex.Message);

}

}

}

private void LoadGroupsData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT ID\_group, Name FROM Groups";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable groupsTable = new DataTable();

try

{

connection.Open();

adapter.Fill(groupsTable);

comboBox2.DataSource = groupsTable;

comboBox2.DisplayMember = "Name";

comboBox2.ValueMember = "ID\_group";

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о родителях: " + ex.Message);

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

string FIO = textBox1.Text;

DateTime dateOfBirth;

if (!DateTime.TryParse(maskedTextBox1.Text, out dateOfBirth))

{

MessageBox.Show("Некорректный формат даты рождения.");

return;

}

string Information = textBox3.Text;

DateTime Entrance;

if (!DateTime.TryParse(maskedTextBox2.Text, out Entrance))

{

MessageBox.Show("Некорректный формат даты поступления.");

return;

}

int idParents = 0;

if (comboBox1.SelectedValue != null)

{

idParents = Convert.ToInt32(comboBox1.SelectedValue);

}

int iDgroup = 0;

if (comboBox2.SelectedValue != null)

{

iDgroup = Convert.ToInt32(comboBox2.SelectedValue);

}

if (button3.Text == "Сохранить")

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string primaryKeyValue = dataGridView1.Rows[selectedIndex].Cells["FIO\_children"].Value.ToString();

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateChildQuery = "UPDATE Childrens SET FIO = @FIO, date\_of\_birth = @dateOfBirth, Information = @Information, Entrance = @Entrance, ID\_parent = @idParents, ID\_group = @iDgroup WHERE FIO = @primaryKeyValue";

SqlCommand updateChildCommand = new SqlCommand(updateChildQuery, connection); updateChildCommand.Parameters.AddWithValue("@FIO", FIO); updateChildCommand.Parameters.AddWithValue("@dateOfBirth", dateOfBirth); updateChildCommand.Parameters.AddWithValue("@Information", Information); updateChildCommand.Parameters.AddWithValue("@Entrance", Entrance); updateChildCommand.Parameters.AddWithValue("@idParents", idParents); updateChildCommand.Parameters.AddWithValue("@iDgroup", iDgroup); updateChildCommand.Parameters.AddWithValue("@primaryKeyValue", primaryKeyValue);

int rowsAffected = updateChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно обновлены.");

button3.Text = "Добавить";

LoadChildrenData();

}

else {MessageBox.Show("Ошибка при обновлении данных.");

}

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertChildQuery = "INSERT INTO Childrens ( FIO, date\_of\_birth, Information, Entrance, ID\_parent, ID\_group) " + "VALUES ( @FIO, @dateOfBirth, @Information, @Entrance, @idParents, @iDgroup)";

SqlCommand insertChildCommand = new SqlCommand(insertChildQuery, connection); insertChildCommand.Parameters.AddWithValue("@FIO", FIO); insertChildCommand.Parameters.AddWithValue("@dateOfBirth", dateOfBirth); insertChildCommand.Parameters.AddWithValue("@Information", Information);

insertChildCommand.Parameters.AddWithValue("@Entrance", Entrance); insertChildCommand.Parameters.AddWithValue("@idParents", idParents); insertChildCommand.Parameters.AddWithValue("@iDgroup", iDgroup);

int rowsAffected = insertChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadChildrenData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

textBox1.Clear();

maskedTextBox1.Clear();

textBox3.Clear();

maskedTextBox2.Clear();

}

private void label6\_Click(object sender, EventArgs e)

{

Parents parents = new Parents();

Hide();

parents.Show();

}

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

{

Form1 form1 = new Form1();

Hide();

form1.ShowDialog();

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string fio = dataGridView1.Rows[selectedIndex].Cells["FIO\_children"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить " + fio + "?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

DeleteChildData(fio);

LoadChildrenData();

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string fio = dataGridView1.Rows[selectedIndex].Cells["FIO\_children"].Value.ToString();

string dateOfBirth = dataGridView1.Rows[selectedIndex].Cells["date\_of\_birth"].Value.ToString();

string information = dataGridView1.Rows[selectedIndex].Cells["Information"].Value.ToString();

string entrance = dataGridView1.Rows[selectedIndex].Cells["Entrance"].Value.ToString();

textBox1.Text = fio;

maskedTextBox1.Text = dateOfBirth;

textBox3.Text = information;

maskedTextBox2.Text = entrance;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void DeleteChildData(string fio)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string deleteQuery = "DELETE FROM Childrens WHERE FIO = @fio";

SqlCommand deleteCommand = new SqlCommand(deleteQuery, connection);

deleteCommand.Parameters.AddWithValue("@fio", fio);

deleteCommand.ExecuteNonQuery();

}

}

private void FillYearsComboBoxes()

{

for (int year = DateTime.Now.Year; year >= 2010; year--)

{

comboBox4.Items.Add(year.ToString());

comboBox5.Items.Add(year.ToString());

}

}

private void button5\_Click(object sender, EventArgs e)

{

string query = "SELECT c.FIO AS FIO\_children, c.date\_of\_birth, c.Information, c.Entrance, p.FIO AS FIO\_parent, g.Name AS Group\_Name " +

"FROM Childrens c " +

"LEFT JOIN Parents p ON c.ID\_parent = p.id\_parent " +

"LEFT JOIN Groups g ON c.ID\_group = g.ID\_group " +

"WHERE 1=1";

if (!string.IsNullOrEmpty(textBox5.Text))

query += $" AND c.FIO LIKE '%{textBox5.Text}%'";

if (comboBox4.SelectedIndex != -1 && comboBox4.SelectedItem.ToString() != "Все")

{

string selectedYear = comboBox4.SelectedItem.ToString();

query += $" AND YEAR(c.date\_of\_birth) = {selectedYear}";

}

if (!string.IsNullOrEmpty(textBox7.Text))

query += $" AND c.Information LIKE '%{textBox7.Text}%'";

if (!string.IsNullOrEmpty(textBox9.Text))

query += $" AND p.FIO LIKE '%{textBox9.Text}%'";

if (!string.IsNullOrEmpty(textBox10.Text))

query += $" AND g.Name LIKE '%{textBox10.Text}%'";

if (comboBox5.SelectedIndex != -1 && comboBox5.SelectedItem.ToString() != "Все")

{

string selectedYear = comboBox5.SelectedItem.ToString();

query += $" AND YEAR(c.Entrance) = {selectedYear}";

}

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

private void FillComboBox3()

{

comboBox3.Items.Clear();

comboBox3.Items.Add("date\_of\_birth");

comboBox3.Items.Add("Entrance");

comboBox3.SelectedIndex = 0;

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + " DESC";

dataGridView1.DataSource = dataView;

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem.ToString() == "date\_of\_birth" || comboBox3.SelectedItem.ToString() == "Entrance")

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

dataGridView1.DataSource = dataView;

}

}

}

}

# ПРИЛОЖЕНИЕ Б

Parents

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

using static

namespace kyrsovaia

{

public partial class Parents : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public Parents()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

LoadParentsData();

dataGridView1.Columns["FIO"].HeaderText = "ФИО родителя";

dataGridView1.Columns["Kontakts"].HeaderText = "Контакты";

dataGridView1.Columns["Informations"].HeaderText = "Информация";

}

private void LoadParentsData()

{

string query = "SELECT FIO, Kontakts, Informations FROM Parents";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

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

{

FillComboBox3();

}

private void button3\_Click(object sender, EventArgs e)

{

string FIO = textBox1.Text;

string Kontakts = maskedTextBox1.Text;

string Information = textBox3.Text;

if (button3.Text == "Сохранить")

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string primaryKeyValue = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateChildQuery = "UPDATE Parents SET FIO = @FIO, Kontakts = @Kontakts, Informations = @Informations WHERE FIO = @primaryKeyValue";

SqlCommand insertParentsCommand = new SqlCommand(updateChildQuery, connection); insertParentsCommand.Parameters.AddWithValue("@FIO", FIO);

insertParentsCommand.Parameters.Add("@Kontakts", SqlDbType.NVarChar, -1).Value = Kontakts; insertParentsCommand.Parameters.AddWithValue("@Informations", Information); insertParentsCommand.Parameters.AddWithValue("@primaryKeyValue", primaryKeyValue);

int rowsAffected = insertParentsCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно обновлены.");

button3.Text = "Добавить";

LoadParentsData();

}

else

{

MessageBox.Show("Ошибка при обновлении данных.");

}

}

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertChildQuery = "INSERT INTO Parents ( FIO, Kontakts, Informations) " + "VALUES (@FIO, @Kontakts, @Informations)";

SqlCommand insertChildCommand = new SqlCommand(insertChildQuery, connection); insertChildCommand.Parameters.AddWithValue("@FIO", FIO); insertChildCommand.Parameters.AddWithValue("@Kontakts", Kontakts); insertChildCommand.Parameters.AddWithValue("@Informations", Information);

int rowsAffected = insertChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadParentsData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

textBox1.Clear();

maskedTextBox1.Clear();

textBox3.Clear();

}

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

{

Form1 form1 = new Form1();

Hide();

form1.ShowDialog();

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string fio = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

string dateOfBirth = dataGridView1.Rows[selectedIndex].Cells["Kontakts"].Value.ToString();

string information = dataGridView1.Rows[selectedIndex].Cells["Informations"].Value.ToString();

textBox1.Text = fio;

maskedTextBox1.Text = dateOfBirth;

textBox3.Text = information;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

string parentFIO = dataGridView1.SelectedRows[0].Cells["FIO"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить " + parentFIO + "?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

UpdateChildrenParentLink(parentFIO);

DeleteParent(parentFIO);

LoadParentsData();

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void UpdateChildrenParentLink(string parentFIO)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateChildrenQuery = "UPDATE Childrens SET ID\_parent = NULL WHERE ID\_parent IN (SELECT id\_parent FROM Parents WHERE FIO = @parentFIO)";

SqlCommand updateChildrenCommand = new SqlCommand(updateChildrenQuery, connection);

updateChildrenCommand.Parameters.AddWithValue("@parentFIO", parentFIO);

updateChildrenCommand.ExecuteNonQuery();

}

}

private void DeleteParent(string parentFIO)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string deleteParentQuery = "DELETE FROM Parents WHERE FIO = @parentFIO";

SqlCommand deleteParentCommand = new SqlCommand(deleteParentQuery, connection);

deleteParentCommand.Parameters.AddWithValue("@parentFIO", parentFIO);

deleteParentCommand.ExecuteNonQuery();

}

}

private void button5\_Click(object sender, EventArgs e)

{

string fio = textBox4.Text;

string kontakts = maskedTextBox2.Text;

string informations = textBox6.Text;

SearchParentsData(fio, kontakts, informations);

}

private void SearchParentsData(string fio, string kontakts, string informations)

{

string query = "SELECT FIO, Kontakts, Informations FROM Parents WHERE 1=1";

if (!string.IsNullOrWhiteSpace(fio))

query += $" AND FIO LIKE '%{fio}%'";

if (!string.IsNullOrWhiteSpace(kontakts))

query += $" AND Kontakts LIKE '%{kontakts}%'";

if (!string.IsNullOrWhiteSpace(informations))

query += $" AND Informations LIKE '%{informations}%'";

using (sqlDataAdapter = new SqlDataAdapter(query, sqlConnection))

{

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

}

dataGridView1.DataSource = dataTable;

}

private void FillComboBox3()

{

comboBox3.Items.Clear();

comboBox3.Items.Add("FIO");

comboBox3.Items.Add("Kontakts");

comboBox3.SelectedIndex = 0;

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + " DESC";

dataGridView1.DataSource = dataView;

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem.ToString() == "FIO" || comboBox3.SelectedItem.ToString() == "Kontakts")

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

dataGridView1.DataSource = dataView;

}

}

}

}

# ПРИЛОЖЕНИЕ В

SAD

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace kyrsovaia

{

public partial class Sad : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public Sad()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

LoadSadData();

dataGridView1.Columns["Name"].HeaderText = "Название";

dataGridView1.Columns["Adres"].HeaderText = "Адрес";

dataGridView1.Columns["Kontakts"].HeaderText = "Контакты";

dataGridView1.Columns["FIO"].HeaderText = "ФИО директора";

}

private void LoadSadData()

{

string query = "SELECT K.Name, K.Adres, K.Kontakts, P.FIO " +

"FROM Kindergartens AS K " +

"LEFT JOIN Personal AS P ON K.ID\_direktora = P.ID\_employee";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

private void LoadPersonalData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT ID\_employee, FIO FROM Personal WHERE Post = 'Директор'";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable directorsTable = new DataTable();

try

{

connection.Open();

adapter.Fill(directorsTable);

if (directorsTable.Rows.Count > 0)

{

comboBox1.DataSource = directorsTable;

comboBox1.DisplayMember = "FIO";

comboBox1.ValueMember = "ID\_employee";

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о директорах: " + ex.Message);

}

}

}

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

{

LoadPersonalData();

FillComboBox3();

}

private void button3\_Click(object sender, EventArgs e)

{

string Name = textBox1.Text;

string Kontakts = textBox2.Text;

string Adres = textBox3.Text;

if (button3.Text == "Сохранить")

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string primaryKeyValue = dataGridView1.Rows[selectedIndex].Cells["Name"].Value.ToString();

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateChildQuery = "UPDATE Kindergartens SET Name = @Name, Adres = @Adres, Kontakts = @Kontakts, ID\_direktora = @ID\_direktora WHERE Name = @primaryKeyValue";

SqlCommand updateSadCommand = new SqlCommand(updateChildQuery, connection); updateSadCommand.Parameters.AddWithValue("@Name", Name); updateSadCommand.Parameters.AddWithValue("@Adres", Adres); updateSadCommand.Parameters.AddWithValue("@Kontakts", Kontakts); updateSadCommand.Parameters.AddWithValue("@ID\_direktora", comboBox1.SelectedValue);

updateSadCommand.Parameters.AddWithValue("@primaryKeyValue", primaryKeyValue);

int rowsAffected = updateSadCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно обновлены.");

button3.Text = "Добавить";

LoadSadData();

}

else

{

MessageBox.Show("Ошибка при обновлении данных.");

}

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertChildQuery = "INSERT INTO Kindergartens (Name, Adres, Kontakts, ID\_direktora) " +

"VALUES (@Name, @Adres, @Kontakts, @ID\_employee)";

SqlCommand insertChildCommand = new SqlCommand(insertChildQuery, connection); insertChildCommand.Parameters.AddWithValue("@Name", Name); insertChildCommand.Parameters.AddWithValue("@Adres", Adres); insertChildCommand.Parameters.AddWithValue("@Kontakts", Kontakts); insertChildCommand.Parameters.AddWithValue("@ID\_employee", comboBox1.SelectedValue);

int rowsAffected = insertChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadSadData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

}

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

{

Form1 form1 = new Form1();

Hide();

form1.ShowDialog();

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

string kindergartensName = dataGridView1.SelectedRows[0].Cells["Name"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить " + kindergartensName + "?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

DeleteSad(kindergartensName);

LoadSadData();

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string name = dataGridView1.Rows[selectedIndex].Cells["Name"].Value.ToString();

string adres = dataGridView1.Rows[selectedIndex].Cells["Adres"].Value.ToString();

string kontakts = dataGridView1.Rows[selectedIndex].Cells["Kontakts"].Value.ToString();

string fio = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

textBox1.Text = name;

textBox2.Text = adres;

textBox3.Text = kontakts;

comboBox1.Text = fio;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void DeleteSad(string name)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string deleteGroupsQuery = "DELETE FROM Groups WHERE ID\_sad IN (SELECT ID\_Sad FROM Kindergartens WHERE Name = @name)";

SqlCommand deleteGroupsCommand = new SqlCommand(deleteGroupsQuery, connection);

deleteGroupsCommand.Parameters.AddWithValue("@name", name);

deleteGroupsCommand.ExecuteNonQuery();

string deleteKindergartensQuery = "DELETE FROM Kindergartens WHERE Name = @name";

SqlCommand deleteKindergartensCommand = new SqlCommand(deleteKindergartensQuery, connection);

deleteKindergartensCommand.Parameters.AddWithValue("@name", name);

deleteKindergartensCommand.ExecuteNonQuery();

}

}

private void button5\_Click(object sender, EventArgs e)

{

string Name = textBox4.Text;

string Adres = textBox6.Text;

string Kontakts = maskedTextBox1.Text;

string FIO = textBox7.Text;

string query = "SELECT K.Name, K.Adres, K.Kontakts, P.FIO " +

"FROM Kindergartens AS K " +

"LEFT JOIN Personal AS P ON K.ID\_direktora = P.ID\_employee " +

"WHERE 1 = 1";

if (!string.IsNullOrWhiteSpace(Name))

query += " AND K.Name LIKE @Name";

if (!string.IsNullOrWhiteSpace(Adres))

query += " AND K.Adres LIKE @Adres";

if (!string.IsNullOrWhiteSpace(Kontakts))

query += " AND K.Kontakts LIKE @Kontakts";

if (!string.IsNullOrWhiteSpace(FIO))

query += " AND P.FIO LIKE @FIO";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Name", "%" + Name + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Adres", "%" + Adres + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Kontakts", "%" + Kontakts + "%");

sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@FIO", "%" + FIO + "%");

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении поиска: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

private void FillComboBox3()

{

comboBox3.Items.Clear();

comboBox3.Items.Add("Kontakts");

comboBox3.SelectedIndex = 0;

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + " DESC";

dataGridView1.DataSource = dataView;

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem.ToString() == "Kontakts")

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

dataGridView1.DataSource = dataView;

}

}

}

}

# ПРИЛОЖЕНИЕ Г

Personal

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace kyrsovaia

{

public partial class Personal : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public Personal()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

LoadPersonalsData();

comboBox1.Items.Add("Директор");

comboBox1.Items.Add("Уборщица");

comboBox1.Items.Add("Педагог");

comboBox1.Items.Add("Воспитатель");

comboBox1.Items.Add("Медсестра");

comboBox1.Items.Add("Бухгалтер");

comboBox1.Items.Add("Помощник воспитателя");

comboBox1.Items.Add("Повар");

comboBox1.Items.Add("Администратор");

comboBox1.Items.Add("Педиатр");

comboBox2.Items.Add("Директор");

comboBox2.Items.Add("Уборщица");

comboBox2.Items.Add("Педагог");

comboBox2.Items.Add("Воспитатель");

comboBox1.Items.Add("Медсестра");

comboBox2.Items.Add("Бухгалтер");

comboBox2.Items.Add("Помощник воспитателя");

comboBox2.Items.Add("Повар");

comboBox2.Items.Add("Администратор");

comboBox2.Items.Add("Педиатр");

comboBox1.DropDownStyle = ComboBoxStyle.DropDownList; // чтобы пользователь не мог вводить свои значения

comboBox1.SelectedIndexChanged += comboBox1\_SelectedIndexChanged;

dataGridView1.Columns["FIO"].HeaderText = "ФИО";

dataGridView1.Columns["Post"].HeaderText = "Должность";

dataGridView1.Columns["Kontakts"].HeaderText = "Контакты";

dataGridView1.Columns["Experience"].HeaderText = "Стаж";

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

string selectedValue = comboBox1.SelectedItem.ToString();

switch (selectedValue)

{

case "Директор":

break;

case "Уборщица":

break;

case "Педагог":

break;

case "Воспитатель":

break;

case "Медсестра":

break;

case "Бухгалтер":

break;

case "Помощник воспитателя":

break;

case "Повар":

break;

case "Администратор":

break;

case "Педиатр":

break;

default:

break;

}

}

private void LoadPersonalsData()

{

string query = "SELECT FIO, Post, Kontakts, Experience FROM Personal";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

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

{

FillComboBox3();

}

private void button3\_Click(object sender, EventArgs e)

{

string FIO = textBox1.Text;

string Post = comboBox1.Text;

string Kontakts = textBox3.Text;

string Experience = textBox2.Text;

if (button3.Text == "Сохранить")

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string primaryKeyValue = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertPersonalQuery = "UPDATE Personal SET FIO = @FIO, Post = @Post, Kontakts = @Kontakts, Experience = @Experience WHERE FIO = @primaryKeyValue";

SqlCommand updatePersonalCommand = new SqlCommand(insertPersonalQuery, connection);

updatePersonalCommand.Parameters.AddWithValue("@FIO", FIO);

updatePersonalCommand.Parameters.AddWithValue("@Post", Post);

updatePersonalCommand.Parameters.AddWithValue("@Kontakts", Kontakts);

updatePersonalCommand.Parameters.AddWithValue("@Experience", Experience);

updatePersonalCommand.Parameters.AddWithValue("@primaryKeyValue", primaryKeyValue);

int rowsAffected = updatePersonalCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

button3.Text = "Добавить";

LoadPersonalsData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertChildQuery = "INSERT INTO Personal ( FIO, Post, Kontakts, Experience) " + "VALUES (@FIO, @Post, @Kontakts, @Experience)";

SqlCommand insertChildCommand = new SqlCommand(insertChildQuery, connection);

insertChildCommand.Parameters.AddWithValue("@FIO", FIO);

insertChildCommand.Parameters.AddWithValue("@Post", Post);

insertChildCommand.Parameters.AddWithValue("@Kontakts", Kontakts);

insertChildCommand.Parameters.AddWithValue("@Experience", Experience);

int rowsAffected = insertChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadPersonalsData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

textBox1.Clear();

textBox3.Clear();

}

}

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

{

Form1 form1 = new Form1();

Hide();

form1.ShowDialog();

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string FIO = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

string Post = dataGridView1.Rows[selectedIndex].Cells["Post"].Value.ToString();

string Kontakts = dataGridView1.Rows[selectedIndex].Cells["Kontakts"].Value.ToString();

string Experience = dataGridView1.Rows[selectedIndex].Cells["Experience"].Value.ToString();

textBox1.Text = FIO;

comboBox1.Text = Post;

textBox3.Text = Kontakts;

textBox2.Text = Experience;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string personalFIO = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить данные?", "Предупреждение", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

UpdateSadTable(personalFIO);

UpdatePersonalSadLink(personalFIO);

DeletePersonalData(personalFIO);

LoadPersonalsData();

}

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void UpdateSadTable(string personalFIO)

{

string updateSadQuery = "UPDATE Groups SET Older\_group = NULL WHERE Older\_group IN (SELECT ID\_sad FROM Kindergartens WHERE ID\_direktora = (SELECT ID\_employee FROM Personal WHERE FIO = @personalFIO))";

using (SqlConnection connection = new SqlConnection(Connect))

{

SqlCommand updateSadCommand = new SqlCommand(updateSadQuery, connection); updateSadCommand.Parameters.AddWithValue("@personalFIO", personalFIO);

try

{

connection.Open();

updateSadCommand.ExecuteNonQuery();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при обновлении таблицы Groups: " + ex.Message);

}

}

}

private void UpdatePersonalSadLink(string personalFIO)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateChildrenQuery = "UPDATE Kindergartens SET ID\_direktora = NULL WHERE ID\_direktora IN (SELECT ID\_employee FROM Personal WHERE FIO = @personalFIO)";

SqlCommand updateChildrenCommand = new SqlCommand(updateChildrenQuery, connection);

updateChildrenCommand.Parameters.AddWithValue("@personalFIO", personalFIO);

updateChildrenCommand.ExecuteNonQuery();

}

}

private void DeletePersonalData(string fio)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string checkAdminUsageQuery = "SELECT COUNT(\*) FROM Admin\_Auth WHERE ID\_employee = (SELECT ID\_employee FROM Personal WHERE FIO = @fio)";

SqlCommand checkAdminUsageCommand = new SqlCommand(checkAdminUsageQuery, connection);

checkAdminUsageCommand.Parameters.AddWithValue("@fio", fio);

int adminUsageCount = (int)checkAdminUsageCommand.ExecuteScalar();

if (adminUsageCount > 0)

{

MessageBox.Show("Администратор используется в форме администратора. Удаление невозможно.");

return;

}

string updateGroupsQuery = "UPDATE Groups SET Older\_group = NULL WHERE Older\_group IN (SELECT ID\_employee FROM Personal WHERE FIO = @fio)";

SqlCommand updateGroupsCommand = new SqlCommand(updateGroupsQuery, connection);

updateGroupsCommand.Parameters.AddWithValue("@fio", fio);

updateGroupsCommand.ExecuteNonQuery();

string deleteQuery = "DELETE FROM Personal WHERE FIO = @fio";

SqlCommand deleteCommand = new SqlCommand(deleteQuery, connection);

deleteCommand.Parameters.AddWithValue("@fio", fio);

deleteCommand.ExecuteNonQuery();

}

}

private void button5\_Click(object sender, EventArgs e)

{

string FIO = textBox4.Text;

string Kontakts = maskedTextBox1.Text;

string Post = comboBox2.Text;

string Experience = maskedTextBox2.Text;

string query = "SELECT FIO, Post, Kontakts, Experience FROM Personal WHERE 1 = 1";

if (!string.IsNullOrWhiteSpace(FIO))

query += " AND FIO LIKE @FIO";

if (!string.IsNullOrWhiteSpace(Kontakts))

query += " AND Kontakts LIKE @Kontakts";

if (!string.IsNullOrWhiteSpace(Post))

query += " AND Post LIKE @Post";

if (!string.IsNullOrWhiteSpace(Experience))

query += " AND Experience LIKE @Experience";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@FIO", "%" + FIO + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Kontakts", "%" + Kontakts + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Post", "%" + Post + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Experience", "%" + Experience + "%");

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении поиска: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

private void FillComboBox3()

{

comboBox3.Items.Clear();

comboBox3.Items.Add("Kontakts");

comboBox3.Items.Add("Post");

comboBox3.Items.Add("Experience");

comboBox3.SelectedIndex = 0;

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + " DESC";

dataGridView1.DataSource = dataView;

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem.ToString() == "Kontakts" || comboBox3.SelectedItem.ToString() == "Post" || comboBox3.SelectedItem.ToString() == "Experience")

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

dataGridView1.DataSource = dataView;

}

}

}

}

# ПРИЛОЖЕНИЕ Д

Groups

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace kyrsovaia

{

public partial class Groups : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public Groups()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

LoadGroupData();

LoadOlderGroupData();

LoadSadData();

comboBox3.Items.Add("Младшая группа");

comboBox3.Items.Add("Средняя группа");

comboBox3.Items.Add("Старшая группа");

comboBox4.Items.Add("Младшая группа");

comboBox4.Items.Add("Средняя группа");

comboBox4.Items.Add("Старшая группа");

comboBox3.DropDownStyle = ComboBoxStyle.DropDownList;

comboBox3.SelectedIndexChanged += comboBox3\_SelectedIndexChanged;

dataGridView1.Columns["GroupName"].HeaderText = "Название группы";

dataGridView1.Columns["Level\_group"].HeaderText = "Уровень группы";

dataGridView1.Columns["EmployeeName"].HeaderText = "ФИО сотрудника";

dataGridView1.Columns["KindergartenName"].HeaderText = "Название сада";

if (dataGridView1.Columns.Contains("ID\_Sad"))

{

dataGridView1.Columns["ID\_Sad"].Visible = false;

}

}

private void LoadGroupData()

{

string query = "SELECT g.ID\_Sad, g.Name AS GroupName, g.Older\_group, g.Level\_group, p.FIO AS EmployeeName, k.Name AS KindergartenName " +

"FROM Groups g " +

"JOIN Personal p ON g.Older\_group = p.ID\_employee " +

"JOIN Kindergartens k ON g.ID\_sad = k.ID\_Sad";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection);

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataTable.Columns.Remove("Older\_group");

if (dataGridView1.Columns.Contains("ID\_Sad"))

{

dataGridView1.Columns["ID\_Sad"].Visible = false;

}

dataGridView1.DataSource = dataTable;

}

private void LoadOlderGroupData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT ID\_employee, FIO FROM Personal";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable parentsTable = new DataTable();

try

{

connection.Open();

adapter.Fill(parentsTable);

comboBox2.DataSource = parentsTable;

comboBox2.DisplayMember = "FIO";

comboBox2.ValueMember = "ID\_employee";

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о родителях: " + ex.Message);

}

}

}

private void LoadSadData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT ID\_Sad, Name FROM Kindergartens";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable parentsTable = new DataTable();

try

{

connection.Open();

adapter.Fill(parentsTable);

comboBox1.DataSource = parentsTable;

comboBox1.DisplayMember = "Name";

comboBox1.ValueMember = "ID\_Sad";

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о родителях: " + ex.Message);

}

}

}

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

{

this.groupsTableAdapter.Fill(this.kindergartens1DataSet.Groups);

FillComboBox3();

}

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

{

Form1 form = new Form1();

Hide();

form.Show();

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem != null)

{

string selectedValue = comboBox3.SelectedItem.ToString();

switch (selectedValue)

{

case "Младшая группа":

break;

case "Средняя группа":

break;

case "Старшая группа":

break;

default:

break;

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

string NameGroup = textBox1.Text;

int NameSad = Convert.ToInt32(comboBox1.SelectedValue);

int dadGroup = Convert.ToInt32(comboBox2.SelectedValue);

string level = comboBox3.Text;

if (button3.Text == "Сохранить")

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string oldNameGroup = dataGridView1.Rows[selectedIndex].Cells["GroupName"].Value?.ToString();

if (oldNameGroup != null)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateQuery = "UPDATE Groups SET ID\_sad = @NameSad, Name = @NameGroup, Older\_group = @dadGroup, Level\_group = @level WHERE Name = @oldNameGroup";

SqlCommand updateCommand = new SqlCommand(updateQuery, connection); updateCommand.Parameters.AddWithValue("@NameSad", NameSad); updateCommand.Parameters.AddWithValue("@NameGroup", NameGroup); updateCommand.Parameters.AddWithValue("@dadGroup", dadGroup); updateCommand.Parameters.AddWithValue("@level", level); updateCommand.Parameters.AddWithValue("@oldNameGroup", oldNameGroup);

int rowsAffected = updateCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно обновлены.");

button3.Text = "Добавить";

LoadGroupData();

}

else

{

MessageBox.Show("Ошибка при обновлении данных.");

}

}

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string insertChildQuery = "INSERT INTO Groups (ID\_sad, Name, Older\_group, Level\_group) " + "VALUES (@NameSad, @NameGroup, @dadGroup, @level)";

SqlCommand insertChildCommand = new SqlCommand(insertChildQuery, connection); insertChildCommand.Parameters.AddWithValue("@NameSad", NameSad); insertChildCommand.Parameters.AddWithValue("@NameGroup", NameGroup); insertChildCommand.Parameters.AddWithValue("@dadGroup", dadGroup);

insertChildCommand.Parameters.AddWithValue("@level", level);

int rowsAffected = insertChildCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadGroupData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

textBox1.Clear();

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string nameGroup = dataGridView1.Rows[selectedIndex].Cells["GroupName"].Value.ToString();

string idSad = dataGridView1.Rows[selectedIndex].Cells["ID\_sad"].Value.ToString();

string olderGroup = dataGridView1.Rows[selectedIndex].Cells["EmployeeName"].Value.ToString();

string levelGroup = dataGridView1.Rows[selectedIndex].Cells["Level\_group"].Value.ToString();

textBox1.Text = nameGroup;

comboBox1.Text = idSad;

comboBox2.Text = olderGroup;

comboBox3.Text = levelGroup;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string nameGroup = dataGridView1.Rows[selectedIndex].Cells["GroupName"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить " + nameGroup + "?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

DeleteGroup(nameGroup);

LoadGroupData();

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void DeleteGroup(string nameGroup)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateNullQuery = "UPDATE Groups SET Older\_group = NULL WHERE Older\_group = (SELECT ID\_employee FROM Personal WHERE FIO = @nameGroup)";

SqlCommand updateNullCommand = new SqlCommand(updateNullQuery, connection); updateNullCommand.Parameters.AddWithValue("@nameGroup", nameGroup);

updateNullCommand.ExecuteNonQuery();

string deleteQuery = "DELETE FROM Groups WHERE Name = @nameGroup";

SqlCommand deleteCommand = new SqlCommand(deleteQuery, connection);

deleteCommand.Parameters.AddWithValue("@nameGroup", nameGroup);

deleteCommand.ExecuteNonQuery();

LoadGroupData();

textBox1.Clear();

comboBox1.SelectedIndex = -1;

comboBox2.SelectedIndex = -1;

comboBox3.SelectedIndex = -1;

}

}

private void button5\_Click(object sender, EventArgs e)

{

string NameGroup = textBox2.Text;

string NameSad = textBox3.Text;

string DadGroup = textBox4.Text;

string LevelGroup = comboBox4.Text;

string query = "SELECT g.ID\_sad, g.Name AS GroupName, g.Older\_group, g.Level\_group, p.FIO AS EmployeeName, k.Name AS KindergartenName " +

"FROM Groups g " +

"JOIN Personal p ON g.Older\_group = p.ID\_employee " +"JOIN Kindergartens k ON g.ID\_sad = k.ID\_Sad " + "WHERE 1 = 1";

if (!string.IsNullOrWhiteSpace(NameGroup))

query += " AND g.Name LIKE @NameGroup";

if (!string.IsNullOrWhiteSpace(NameSad))

query += " AND k.Name LIKE @NameSad";

if (!string.IsNullOrWhiteSpace(DadGroup))

query += " AND p.FIO LIKE @DadGroup";

if (!string.IsNullOrWhiteSpace(LevelGroup))

query += " AND g.Level\_group LIKE @LevelGroup";

sqlDataAdapter = new SqlDataAdapter(query, sqlConnection); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@NameGroup", "%" + NameGroup + "%");

sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@NameSad", "%" + NameSad + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@DadGroup", "%" + DadGroup + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@LevelGroup", "%" + LevelGroup + "%");

dataTable = new DataTable();

try

{

sqlConnection.Open();

sqlDataAdapter.Fill(dataTable);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении поиска: " + ex.Message);

}

finally

{

sqlConnection.Close();

}

dataGridView1.DataSource = dataTable;

}

private void FillComboBox3()

{

comboBox5.Items.Clear();

comboBox5.Items.Add("Level\_group");

comboBox5.SelectedIndex = 0;

}

private void comboBox5\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox5.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + " DESC";

dataGridView1.DataSource = dataView;

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox5.SelectedItem != null)

{

string columnName = comboBox5.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

if ( columnName == "Level\_group")

{

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

}

dataGridView1.DataSource = dataView;

}

}

}

}

# ПРИЛОЖЕНИЕ E

Администраторы

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace kyrsovaia

{

public partial class admin : Form

{

private SqlConnection sqlConnection;

private SqlDataAdapter sqlDataAdapter;

private DataTable dataTable;

private string Connect = @"Data Source=localhost;Initial Catalog=Kindergartens1;Integrated Security=True";

public admin()

{

InitializeComponent();

sqlConnection = new SqlConnection(Connect);

FillComboBox3();

}

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

{

LoadAdminData();

LoadPersonalData();

dataGridView1.Columns["FIO"].HeaderText = "ФИО";

dataGridView1.Columns["Username"].HeaderText = "Имя пользователя";

dataGridView1.Columns["Password"].HeaderText = "Пароль";

}

private void LoadAdminData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT p.FIO, a.Username, a.Password FROM Admin\_Auth a INNER JOIN Personal p ON a.ID\_employee = p.ID\_employee";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable adminTable = new DataTable();

try

{

connection.Open();

adapter.Fill(adminTable);

dataGridView1.DataSource = adminTable;

dataTable = adminTable;

FillComboBox3();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных из таблицы Admin\_Auth: " + ex.Message);

}

}

}

private void LoadPersonalData()

{

using (SqlConnection connection = new SqlConnection(Connect))

{

string query = "SELECT ID\_employee, FIO FROM Personal WHERE Post = 'Администратор'";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable personalTable = new DataTable();

try

{

connection.Open();

adapter.Fill(personalTable);

if (personalTable.Rows.Count > 0)

{

comboBox2.DataSource = personalTable;

comboBox2.DisplayMember = "FIO";

comboBox2.ValueMember = "ID\_employee";

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при загрузке данных о персонале: " + ex.Message);

}

}

}

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

{

Form1 form1 = new Form1();

Hide();

form1.Show();

}

private void button3\_Click(object sender, EventArgs e)

{

int ID\_employee = 0;

string KodName = textBox5.Text;

string password = textBox1.Text;

if (comboBox2.SelectedValue != null)

{

ID\_employee = Convert.ToInt32(comboBox2.SelectedValue);

}

if (button3.Text == "Сохранить")

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string primaryKeyValue = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string updateQuery = "UPDATE Admin\_Auth SET Username = @Username, Password = @Password WHERE ID\_employee = @ID\_employee";

SqlCommand updateCommand = new SqlCommand(updateQuery, connection); updateCommand.Parameters.AddWithValue("@Username", KodName); updateCommand.Parameters.AddWithValue("@Password", password); updateCommand.Parameters.AddWithValue("@ID\_employee", ID\_employee);

int rowsAffected = updateCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно обновлены.");

button3.Text = "Добавить";

LoadAdminData();

}

else

{

MessageBox.Show("Ошибка при обновлении данных.");

}

}

}

else

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string checkQuery = "SELECT COUNT(\*) FROM Admin\_Auth WHERE ID\_employee = @ID\_employee";

SqlCommand checkCommand = new SqlCommand(checkQuery, connection); checkCommand.Parameters.AddWithValue("@ID\_employee", ID\_employee);

int existingRecords = (int)checkCommand.ExecuteScalar();

if (existingRecords > 0)

{

MessageBox.Show("Запись с таким администратором уже существует.");

return;

}

string insertQuery = "INSERT INTO Admin\_Auth (ID\_employee, Username, Password) VALUES (@ID\_employee, @Username, @Password)";

SqlCommand insertCommand = new SqlCommand(insertQuery, connection); insertCommand.Parameters.AddWithValue("@ID\_employee", ID\_employee);

insertCommand.Parameters.AddWithValue("@Username", KodName); insertCommand.Parameters.AddWithValue("@Password", password);

int rowsAffected = insertCommand.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены.");

LoadAdminData();

}

else

{

MessageBox.Show("Ошибка при добавлении данных.");

}

}

}

textBox1.Clear();

textBox3.Clear();

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

int selectedIndex = dataGridView1.SelectedRows[0].Index;

string FIO = dataGridView1.Rows[selectedIndex].Cells["FIO"].Value.ToString();

string KodName = dataGridView1.Rows[selectedIndex].Cells["Username"].Value.ToString();

string password = dataGridView1.Rows[selectedIndex].Cells["Password"].Value.ToString();

comboBox2.Text = FIO;

textBox5.Text = KodName;

textBox1.Text = password;

button3.Text = "Сохранить";

}

else

{

MessageBox.Show("Выберите строку для редактирования данных.");

}

}

private void button4\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0)

{

string username = dataGridView1.SelectedRows[0].Cells["Username"].Value.ToString();

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить " + username + "?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

DeleteAdminData(username);

LoadAdminData();

}

}

else

{

MessageBox.Show("Выберите строку для удаления данных.");

}

}

private void DeleteAdminData(string username)

{

using (SqlConnection connection = new SqlConnection(Connect))

{

connection.Open();

string deleteQuery = "DELETE FROM Admin\_Auth WHERE Username = @Username";

SqlCommand deleteCommand = new SqlCommand(deleteQuery, connection); deleteCommand.Parameters.AddWithValue("@Username", username);

deleteCommand.ExecuteNonQuery();

LoadAdminData();

}

}

private void button5\_Click\_1(object sender, EventArgs e)

{

string FIO = textBox4.Text;

string Username = textBox3.Text;

string Password = textBox2.Text;

string query = "SELECT p.FIO, a.Username, a.Password FROM Admin\_Auth a INNER JOIN Personal p ON a.ID\_employee = p.ID\_employee WHERE 1 = 1";

if (!string.IsNullOrWhiteSpace(FIO))

query += " AND p.FIO LIKE @FIO";

if (!string.IsNullOrWhiteSpace(Username))

query += " AND a.Username LIKE @Username";

if (!string.IsNullOrWhiteSpace(Password))

query += " AND a.Password LIKE @Password";

using (SqlConnection connection = new SqlConnection(Connect))

{

sqlDataAdapter = new SqlDataAdapter(query, connection); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@FIO", "%" + FIO + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Username", "%" + Username + "%"); sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@Password", "%" + Password + "%");

dataTable = new DataTable();

try

{

connection.Open();

sqlDataAdapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении поиска: " + ex.Message);

}

}

}

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

private void FillComboBox3()

{

comboBox3.Items.Clear();

comboBox3.Items.Add("Username");

comboBox3.Items.Add("Password");

comboBox3.SelectedIndex = 0;

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (comboBox3.SelectedItem.ToString() == "Username" || comboBox3.SelectedItem.ToString() == "Password")

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName;

if (checkBox1.Checked)

{

dataView.Sort += " ASC";

}

else

{

dataView.Sort += " DESC";

}

dataGridView1.DataSource = dataView;

}

else

{

MessageBox.Show("Сортировка доступна только по столбцам 'date\_of\_birth' и 'Entrance'.");

}

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

string columnName = comboBox3.SelectedItem.ToString();

DataView dataView = dataTable.DefaultView;

dataView.Sort = columnName + (checkBox1.Checked ? " ASC" : " DESC");

dataGridView1.DataSource = dataView;

}

}

}

}