ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ профессиональное

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

**Пермский химико-технологический техникум**

Специальность 09.02.07

«Информационные системы и программирование»

**КУРСОВОЙ ПРОЕКТ**

**Междисциплинарный курс:** «МДК.11.01 Технология разработки и защиты баз данных»

**Тема:** «Проектирование базы данных для детского центра творчества «Капитошка»»

**Выполнил** студент гр.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (А.А. Каменских)

**Проверил(а)**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Л.Ф. Абдалова)

*2024*

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc170347306)

[Описательная часть 4](#_Toc170347307)

[1. Описание предметной области. Постановка задачи 4](#_Toc170347308)

[2. Выбор средств/методологии проектирования. Выбор СУБД 8](#_Toc170347309)

[3. Построение инфологической (концептуальной) модели предметной области 11](#_Toc170347310)

[4. Проектирование логической структуры БД 12](#_Toc170347311)

[5. Проектирование физической структуры базы данных 13](#_Toc170347312)

[Практическая часть 14](#_Toc170347313)

[6. Разработка интерфейса приложения 14](#_Toc170347314)

[6.1 Организация ввода-вывода данных в БД 14](#_Toc170347315)

[6.2 Организация манипулирования данными БД 16](#_Toc170347316)

[6.3 Реализация запросов, получение отчетов 18](#_Toc170347317)

[7. Администрирование БД (использование механизма триггеров) 19](#_Toc170347318)

[8. Заключение 20](#_Toc170347319)

[9. Список литературы 21](#_Toc170347320)

[10. Приложение 22](#_Toc170347321)

# **ВВЕДЕНИЕ**

Базы данных дают возможность множеству пользователей в одно и то же время быстро и безопасно получать доступ к данным и запрашивать их, используя развитую логику и язык запросов.

Любая современная организация не может обойтись без базы данных.

Детские центры творчества играют ключевую роль в развитии детей, предоставляя им возможность раскрыть себя, с помощью разнообразных образовательных и развлекательных программ. Проектирование базы данных является необходимым для детского центра творчества. БД поможет распределить нагрузку и не загружать сервер, а также поспособствует эффективной работе приложения и учету информации о посетителях, сотрудниках, занятиях, другим аспектам деятельности организации.

Целью курсового проекта являются проектирование базы данных и разработка удобного приложения для центра детского творчества.

Задачами курсового проекта являются:

* анализ предметной области;
* разработка логической и концептуальной модели БД;
* реализация логической модели в конкретной СУБД;
* создание приложения для работы в БД;
* разработка технической документации.

# **Описательная часть**

# **1. Описание предметной области. Постановка задачи**

**Предметной областью является центр детского творчества «Капитошка». Центр детского творчества (ЦДТ)** представляет собой учреждение для детей и подростков, в котором организуются развивающие занятия. Посещение таких центров длится несколько часов, имеет ряд направлений и секций, которые и посещает ребенок, обеспечивает удовлетворение образовательных потребностей детей и подростков в области внешкольной деятельности.

Для оптимизации работы приложения для персонала необходимо структурировать информацию, разработав базу данных, которая содержит в себе следующие данные:

1. данные о детях: ФИО, дата рождения и информация о законном представителе;
2. данные о должностях: наименование должностей;
3. данные о законных представителях: ФИО, дата рождения, место работы и номер телефона;
4. данные о заявлениях на зачисления в ЦДТ: номер заявления, ФИО ребенка, ФИО законного представителя, дата начала и дата окончания, направление занятия, группа, которую будет посещать ребенок, и педагог;
5. данные о группах: название группы, наименование направлений занятий и ФИО педагога. За каждым педагогом закреплена группа;
6. данные о педагогах: ФИО, дата рождения, номер телефона, логин, пароль. Логин и пароль предоставляется педагогам для входа в систему, что позволит им иметь доступ к нужным для них ресурсам
7. данные о мероприятиях ЦДТ: наименование группы, ФИО педагога, дата начала и дата окончания. Мероприятия, такие как конкурсы и выставки, проходят внутри ЦДТ, ответственные за мероприятия педагоги. Проведение мероприятий позволяет детям показать свои таланты и то, чему их научили в центре творчества.

Основные функции системы представлены на функционально диаграмме (Рис.1, Рис. 2):

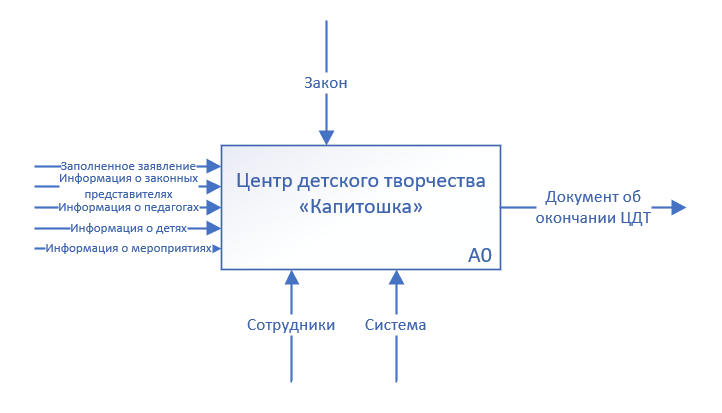


Рис. 1 – Функциональная диаграмма А0.

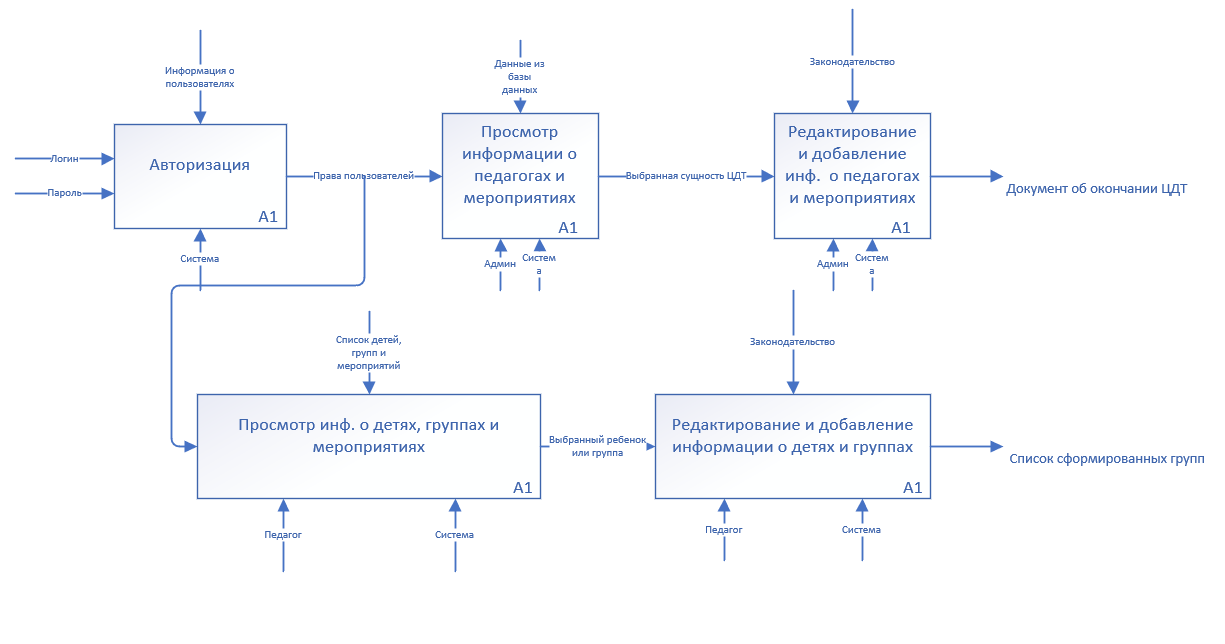


Рис. 2 – Функциональная диаграмма А1

Для достижения данных требований, функции системы будут распределены между СУБД и программным кодом. Доступные функции в системе для отдельных ролей отображены на диаграмме прецедентов (Рис. 3):

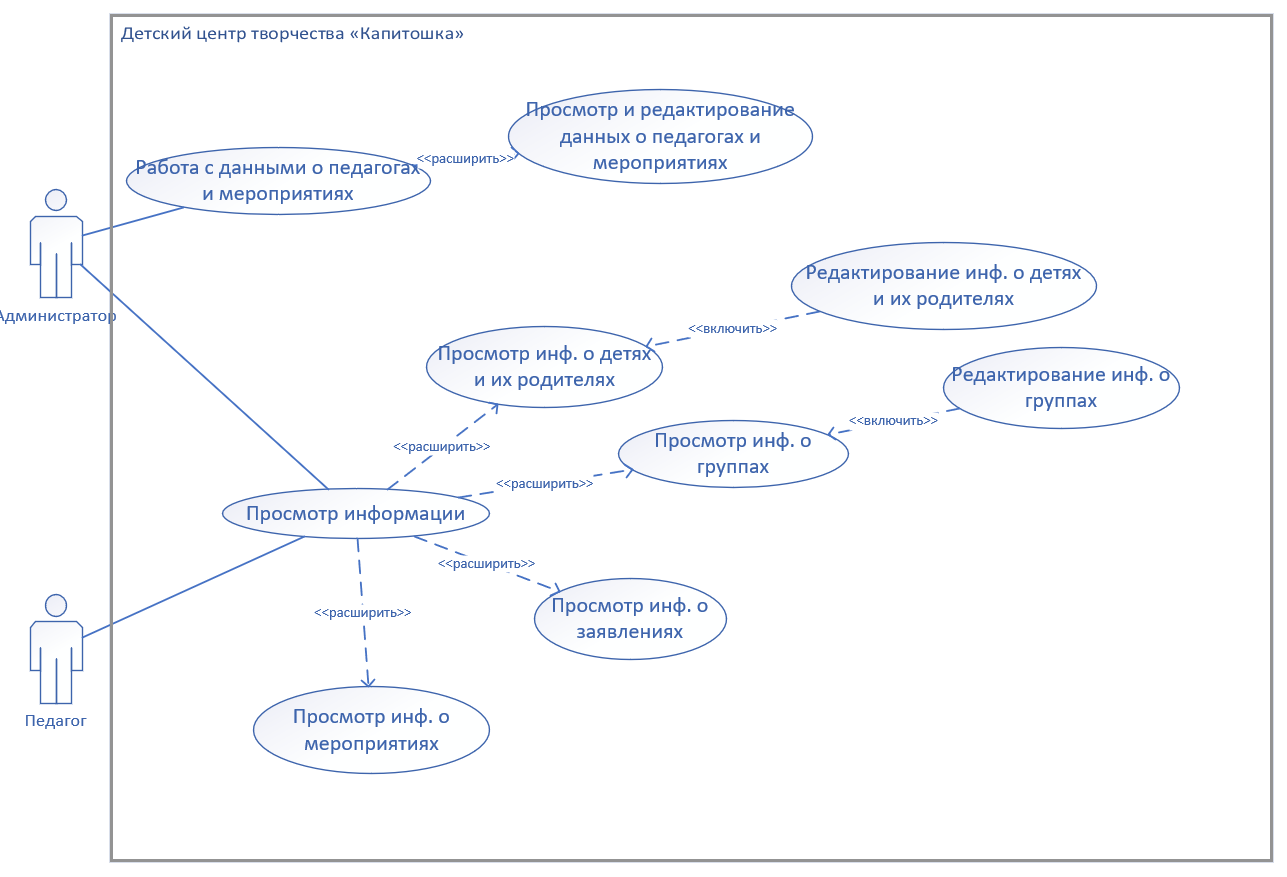


Рис. 3 – Диаграмма прецедентов

# **2. Выбор средств/методологии проектирования. Выбор СУБД**

Для проектирования баз данных выбор пал на семантическую модель. Семантическая модель баз данных – это высокоуровневая модель, которая описывает структуру данных, их смысл и взаимосвязи в информационной системе. Она сосредотачивается на смысле данных, их значениях и смысловых связях, а не на конкретной структуре хранения данных. Семантическая модель данных обладает несколькими преимуществами:

1. **Понятность и ясность:** Семантическая модель уделяет особое внимание смыслу данных, их значениям и взаимосвязям. Это делает модель более понятной для различных участников проекта, что способствует лучшему пониманию предметной области.
2. **Согласованность:** Создание семантической модели помогает установить единое представление данных между всеми заинтересованными сторонами. Это способствует созданию общего языка для обсуждения данных и их взаимосвязей.
3. **Удобство анализа:** Семантическая модель облегчает анализ данных, поскольку она фокусируется на смысле и значении данных. Это помогает выявить скрытые взаимосвязи и обнаружить потенциальные проблемы еще на этапе проектирования.
4. **Основа для разработки:** Семантическая модель является основой для разработки дальнейших уровней моделирования данных, таких как логическая и физическая модели. Она помогает строить более точные и эффективные базы данных.
5. **Гибкость:** Семантическая модель позволяет легко вносить изменения в структуру данных и их взаимосвязи, поскольку она описывает смысл данных, а не их конкретную реализацию.

ERD или ER-диаграмма (Схема «сущность-связь») — это разновидность блок-схемы, где показано, как разные «сущности» (объекты) связаны между собой внутри системы. ER-диаграммы (или ER-модели) полагаются на стандартный набор символов, включая прямоугольники, ромбы, овалы и соединительные линии, для отображения сущностей, их атрибутов и связей. Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи — глаголов. ER-диаграммы состоят из нескольких основных компонентов:

1. Сущности – объекты, о которых хранятся данные в БД;
2. Атрибуты – характеристики сущностей (описание свойств);
3. Взаимосвязи – связи между отдельными сущностями.

Выбор СУБД – ключевой выбор курсового проекта для управления базой данных. При выборе технологий проектирования, одной из ключевых точек является выбор системы управления базами данных (СУБД). Существует множество различных СУБД на рынке, каждая из которых имеет свои преимущества и недостатки.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название СУБД** | **Стоимость** | **Платформа** | **Тип программы** | **Доступ** |
| Microsoft SQL Server Management Studio | Можно скачать бесплатно | Windows | Серверная | Клиент-Сервер |
| My SQL | Можно скачать бесплатно | Windows, Linux, macOS | Серверная | Сетевой доступ |
| MS Access | Можно скачать только платно | Windows | Настольная | Локальный доступ |

Таблица 1 – Сравнительный анализ СУБД.

Сравнив между собой несколько СУБД выбор пал на Microsoft SQL Server. Microsoft SQL Server – это одна из самых популярных реляционных систем управления базами данных (СУБД) на рынке. SQL Server обладает механизмами, обеспечивающими надежность и безопасность данных, имеет высокую производительность при выполнении запросов и обработке транзакций, поддерживает стандартные языки запросов SQL, а также языки программирования, что делает его гибким для разработки различных приложений.

# **3. Построение инфологической (концептуальной) модели предметной области**

Концептуальная модель базы данных – это некая наглядная диаграмма, нарисованная в принятых обозначениях и подробно показывающая связь между объектами и их характеристиками.

На основе описания предметной области была построена концептуальная модель (Рис. 4).

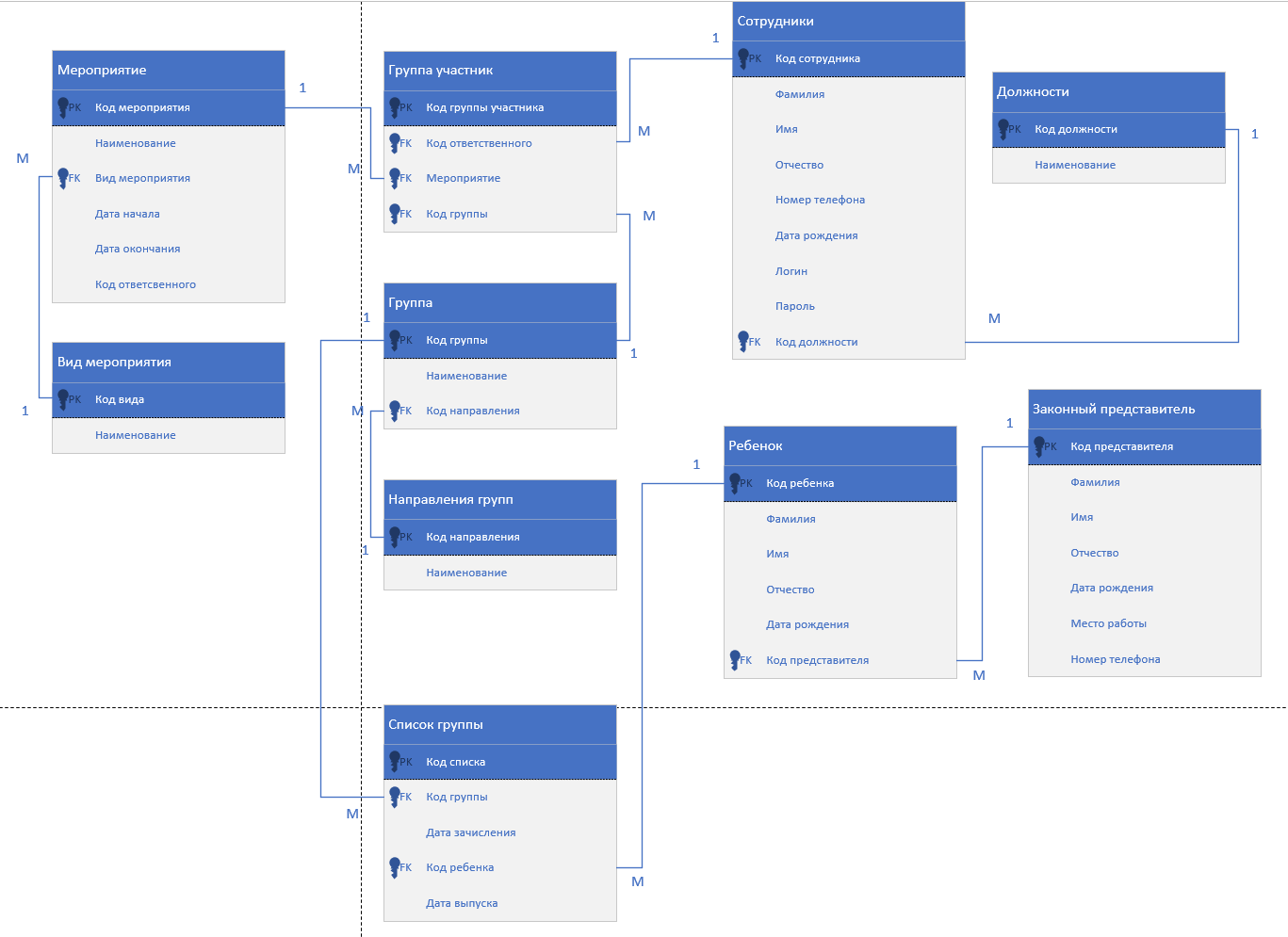


Рис. 4 – Концептуальная модель базы данных.

# **4. Проектирование логической структуры БД**

Логическая (даталогическая) модель представляет собой модель базы данных, которая не привязана к конкретной СУБД. В ней выделяют основные объекты БД и определяют связи между этими объектами. Иногда определятся типы данных отдельных объектов.

На основе описания предметной области была построена логическая модель (Рис. 5).

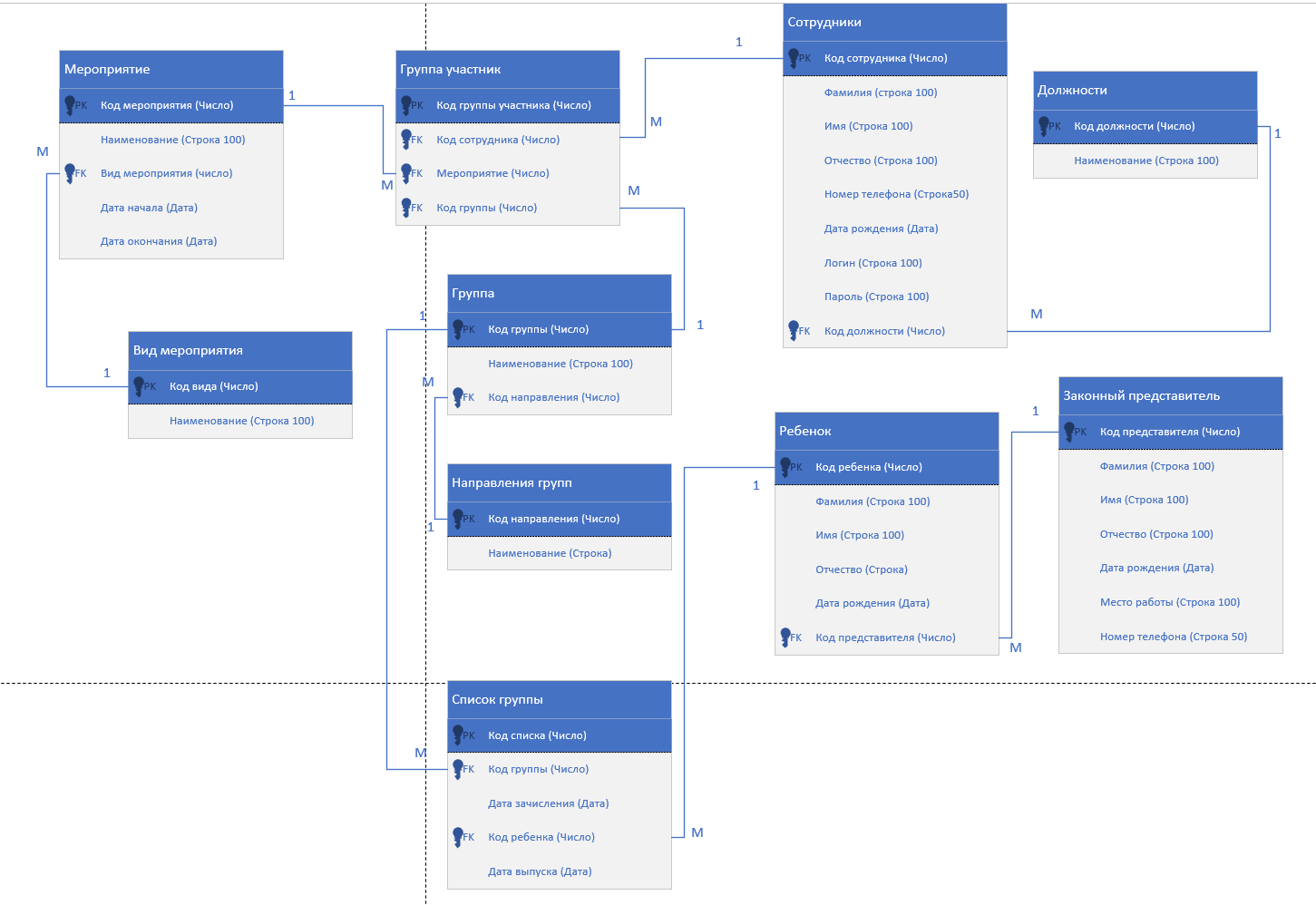


Рис. 5 – Логическая модель базы данных

# **5. Проектирование физической структуры базы данных**

Физическая структура баз данных включает в себя способы организации и хранения данных на уровне хранилища. Она определяет, как данные фактически хранятся на диске и каким образом они доступны для чтения и записи.

На основе логической модели была построена физическая модель базы данных (Рис. 6)

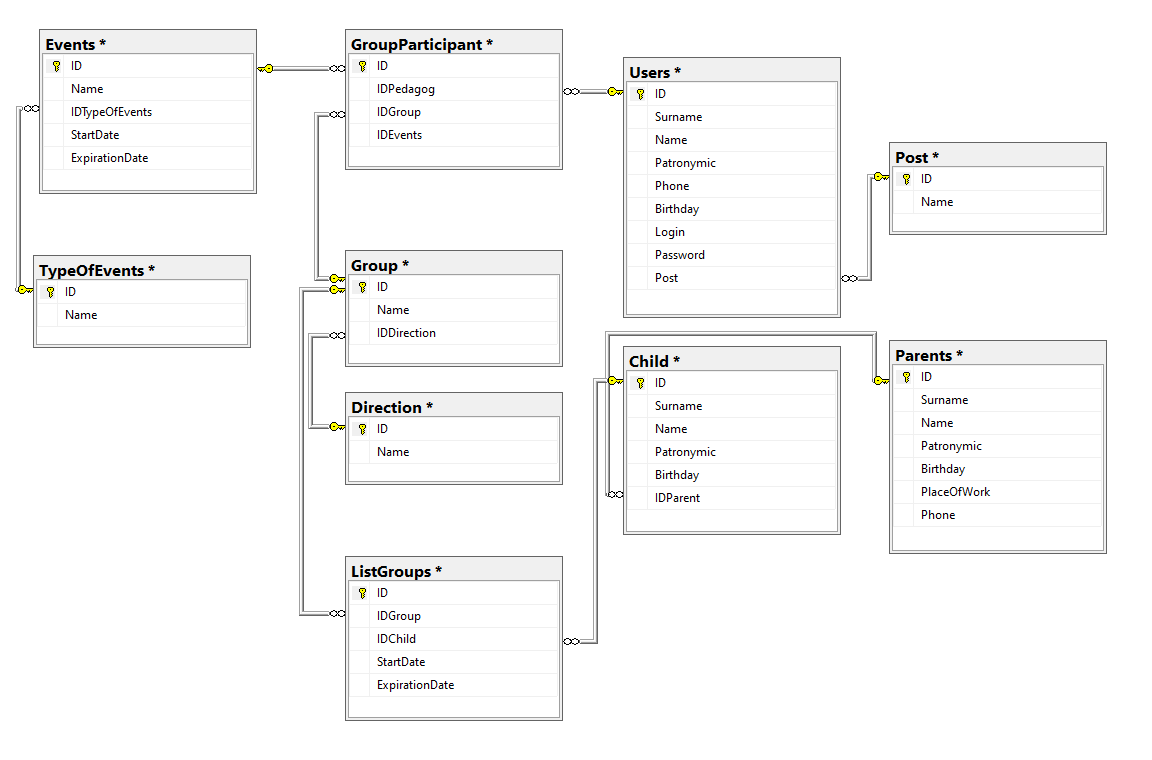


Рис. 6 – Физическая модель базы данных

# **Практическая часть**

# **6. Разработка интерфейса приложения**

## 6.1 Организация ввода-вывода данных в БД

Для полноценной работы с приложением пользователю необходимо пройти авторизацию. В окне авторизации пользователь вводит логин и пароль из базы данных, после успешной авторизации система выводит сообщение, которое содержит в себе ФИО и должность сотрудника (Рис.7).

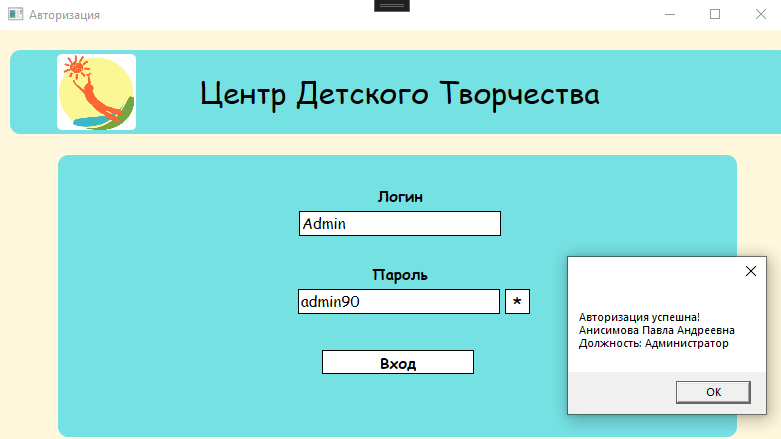


Рис. 7 – Ввод данных в окне авторизации.

Администратору доступна функция изменения информации о сотрудниках. Для этого необходимо выбрать сотрудника из списка и в полях, расположенных с правой стороны экрана, поменять то, что необходимо, после чего, нажать кнопку «Изменить» (Рис. 8).

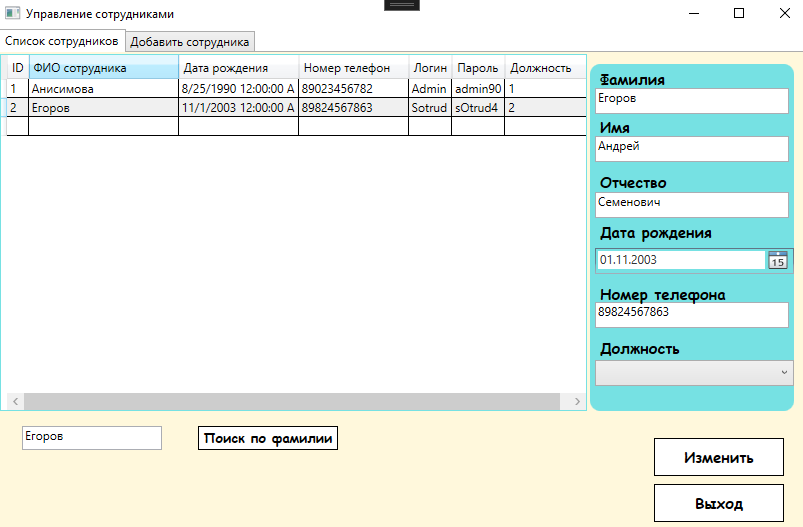


Рис. 8 – Вывод данных о сотруднике из базы данных.

Окно администратора с выводом данных о мероприятяих центра детского творчества и базы данных и изменение этих данных (Рис. 9).

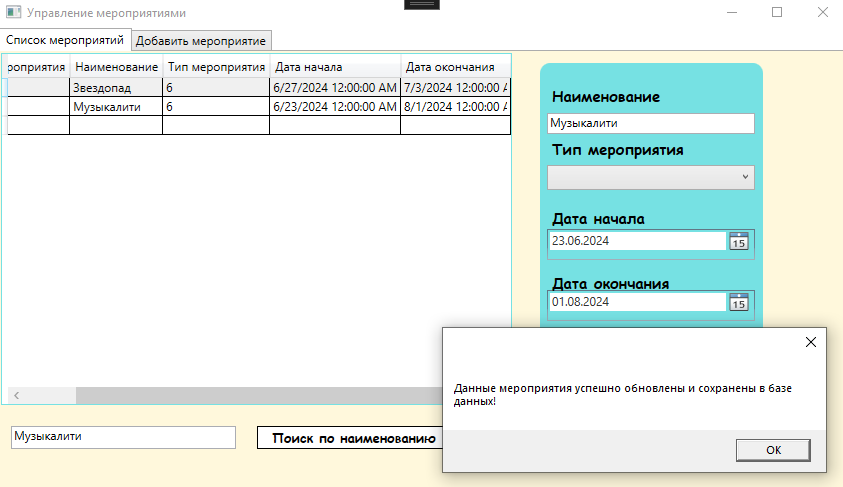


Рис. 9 – Вывод данных и изменение данных о мероприятиях.

## 6.2 Организация манипулирования данными БД

Окно администратора с поиском сотрудника по фамилии и вывод данных нужного сотрудника в поля для вывода (Рис. 10).

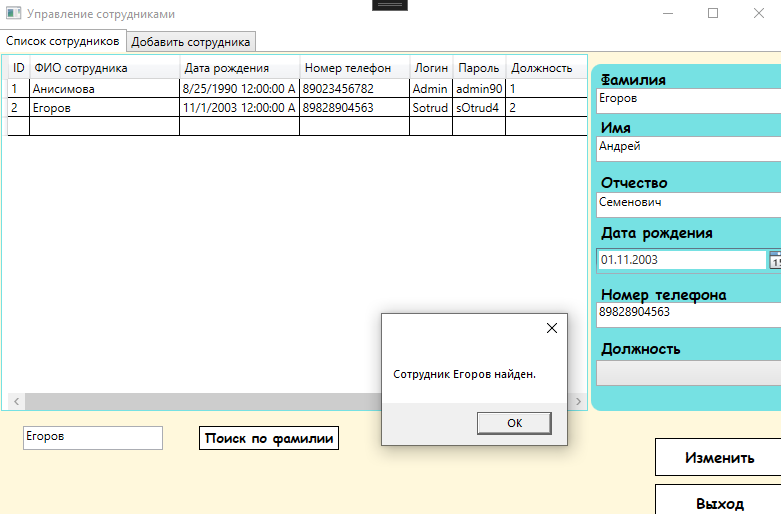


Рис. 10 – Поиск сотрудника и вывод данных о нем.

Окно администратора с возможностью добавления данных о новых сотрудниках в базу данных (Рис. 11).

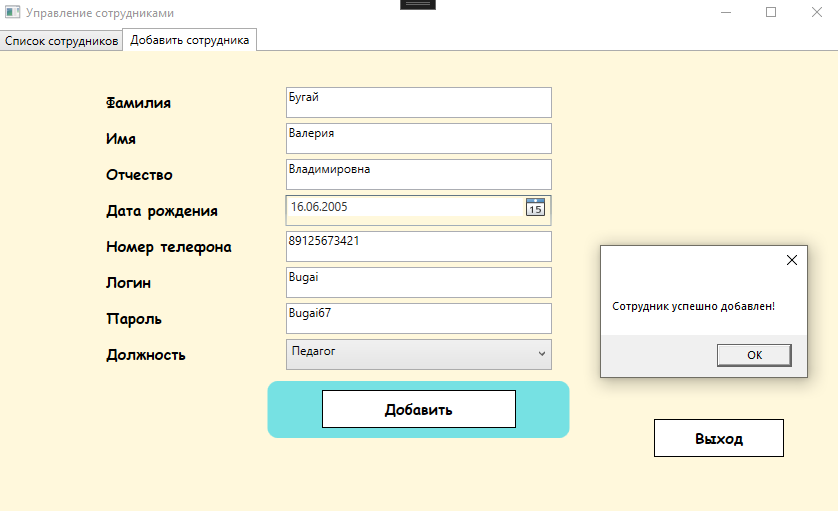


Рис. 11 – Добавление данных о новом сотруднике в базу данных.

Окно администратора с возможностью обновления данных о сотрудниках в базу данных (Рис. 12)

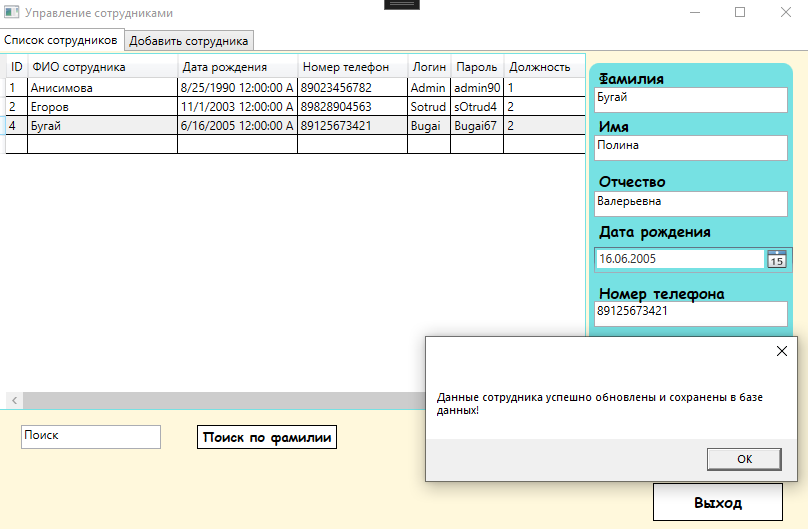


Рис. 12 – Обновление данных о сотруднике в базе данных.

## 6.3 Реализация запросов, получение отчетов

Окно администратора с создание договора об обучении в центре детского творчества (Рис. 13)

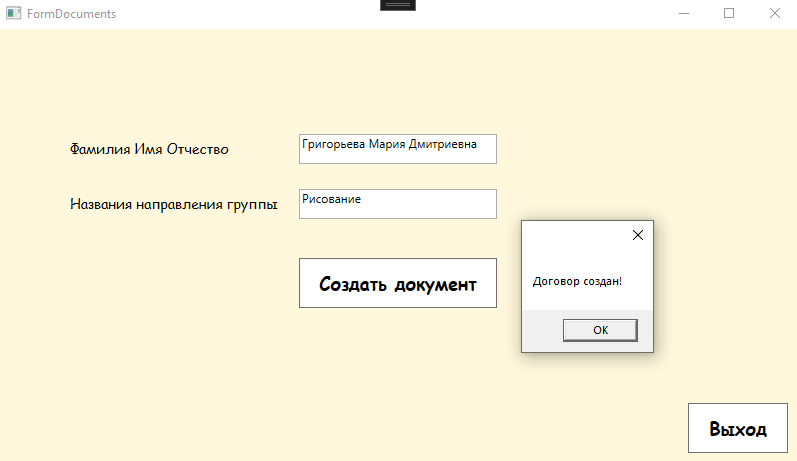


Рис. 13 – Создание договора об обучении в центре детского творчества в Word-документе.

# **7. Администрирование БД (использование механизма триггеров)**

В бызе данных не разработано использование механизма треггеров.

# **8. Заключение**

В рамках данного курсового проекта была разработана информационная база для Центра Детского Творчества, представляющая собой комплексное решение для эффективного управления деятельностью учреждения. Спроектированная база данных позволяет эффективно хранить и обрабатывать информацию о группах, учениках, сотрудниках, мероприятиях и других аспектах деятельности Центра.

Проект направлен на автоматизацию работы Центра Детского Творчества, что способствует его развитию, делает его более современным, открытым и привлекательным для детей, родителей и педагогов, улучшая как внутренние процессы, так и качество предоставляемых услуг.

# **9. Список литературы**

* + - 1. Федорова, Г. Н.Основы проектирование баз данных. М.: Издательский центр "Академия", 2020. - 224 с.
      2. Волк, В. К. Базы данных. Проектирование, программирование, управление и администрирование. Санкт - Петербург: Лань, 2020. — 244 с.
      3. М.: Издательский центр "Академия", 2020. - 224 с.
      4. habr.com [Электронный ресурс] URL: https://habr.com/ru/companies/first/articles/755832/

(Дата обращения: 24.06.2024)

* + - 1. skillbox.ru [Электронный ресурс] URL: https://skillbox.ru/media/code/sqlzaprosy-osnovnye-komandy-dlya-upravleniya-bazami-dannykh/

(Дата обращения: 29.05.2024)

# **10. Приложение**

Созданный договор, выделенные поля заполняются автоматически при его создании:

**Договор об образовании №\_**

**на обучение по дополнительным общеобразовательным программам**

Документ от 18.06.2024 13:26:20 свидетельстует о том, что Иванов Иван Иванович поступил(а) Центр Детского Творчества «Капитошка», и пройдет обучение в соответствии с программой дополнительного общеобразовательного обучения в группе с направлением «Название».

**Права Исполнителя, Заказчика и Учащегося**

2.1. **Исполнитель вправе**:

2.1.1. Самостоятельно осуществлять образовательный процесс, устанавливать системы оценок, формы, порядок и периодичность проведения промежуточной аттестации Учащегося.

2.1.2. Применять к Учащемуся меры поощрения и меры дисциплинарного взыскания в соответствии с законодательством Российской Федерации, учредительными документами Исполнителя, настоящим Договором и локальными нормативными актами Исполнителя.

2.2. **Заказчик вправе** получать информацию от Исполнителя по вопросам организации и обеспечения надлежащего предоставления услуг, предусмотренных разделом I настоящего Договора.

2.2. **Заказчик вправе** получать информацию от Исполнителя по вопросам организации и обеспечения надлежащего предоставления услуг, предусмотренных разделом I настоящего Договора:

- выбирать с учетом мнения ребенка, а также с учетом рекомендаций психолого-медико-педагогической комиссии (при их наличии) формы обучения, в образовательной организации, язык, языки образования, дополнительные общеобразовательные общеразвивающие программы из перечня, предлагаемого организацией;

- знакомиться с уставом образовательной организации, лицензией на осуществление образовательной деятельности, с учебно-программной документацией и другими документами, регламентирующими организацию и осуществление образовательной деятельности;

- знакомиться с содержанием образования, используемыми методами обучения и воспитания, образовательными технологиями, а также с достижениями учащихся;

- защищать права и законные интересы учащихся;

- получать информацию о всех видах планируемых обследований (психологических, психолого-педагогических) учащихся, давать согласие на проведение таких обследований или участие в таких обследованиях, отказаться от их проведения или участия в них, получать информацию о результатах проведенных обследований учащихся;

- принимать участие в управлении образовательной организации, в форме, определяемой уставом этой организации;

2.3. **Учащемуся предоставляются права** в соответствии с частью 1 статьи 34 Федерального закона от 29 декабря 2012г. №273-ФЗ «Об образовании в Российской Федерации». Учащийся также **вправе:**

2.3.1. Получать информацию от Исполнителя по вопросам организации и обеспечения надлежащего предоставления услуг, предусмотренных разделом I настоящего Договора.

2.3.2. Обращаться к Исполнителю по вопросам, касающимся образовательного процесса.

2.3.3. Пользоваться в порядке, установленном локальными нормативными актами, имуществом Исполнителя, необходимым для освоения образовательной программы.

2.3.4. Принимать в порядке, установленном локальными нормативными актами, участие в социально-культурных, оздоровительных и иных мероприятиях, организованных Исполнителем.

2.3.5. Получать полную и достоверную информацию об оценке своих знаний, умений, навыков и компетенций, а также о критериях этой оценки.

Код «Окно Авторизации»:

public static class GlobalVariables

{

public static string UserLogin;

public static string UserPassword;

}

private void Entrance\_Click(object sender, RoutedEventArgs e)

{

string login = textBoxLogin.Text;

string password = textBoxPassword.Text;

using (var Context = new capitoshkaEntities())

{

try

{

var user = Context.Users.FirstOrDefault(u => u.Login == login);

if (user != null)

{

if (user.Password != password)

{

MessageBox.Show("Введен неверный пароль");

}

else

{

if (user != null)

{

// Проверка, является ли пользователь Администратором

if (user.Post == 1)

{

string rol = "Должность: Администратор";

MessageBox.Show($"Авторизация успешна!\n" + user.Surname + " " + user.Name + " " + user.Patronymic + "\n" + rol);

Admin admin = new Admin();

admin.Show();

Close();

GlobalVariables.UserLogin = textBoxLogin.Text;

GlobalVariables.UserPassword = textBoxPassword.Text;

}

// Проверка, является ли пользователь Педагогом

else if (user.Post == 2)

{

string rol = "Должность: Педагог";

MessageBox.Show($"Авторизация успешна!\n" + user.Surname + " " + user.Name + " " + user.Patronymic + "\n" + rol);

Pedagog ped = new Pedagog();

ped.Show();

this.Close();

GlobalVariables.UserLogin = textBoxLogin.Text;

GlobalVariables.UserPassword = textBoxPassword.Text;

}

}

}

}

else

{

MessageBox.Show("Пользователь с такими данными не найден. Зарегистрируйтесь, чтобы продолжить.");

}

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка при обработке запроса: {ex.Message}");

}

}

}

private void BtnMAska\_Click(object sender, RoutedEventArgs e)

{

if (textBoxPassword.Visibility == Visibility.Visible)

{

paswordPB.Visibility = Visibility.Visible;

textBoxPassword.Visibility = Visibility.Collapsed;

}

else

{

paswordPB.Visibility = Visibility.Collapsed;

textBoxPassword.Visibility = Visibility.Visible;

}

//скрытие пароля

}

private void textBoxPassword\_TextChanged(object sender, TextChangedEventArgs e)

{

paswordPB.Password = textBoxPassword.Text;

}

private void paswordPB\_TextInput(object sender, TextCompositionEventArgs e)

{

textBoxPassword.Text = paswordPB.Password;

}

Окно «Администратор. Работа с сотрудниками»:

capitoshkaEntities db = new capitoshkaEntities();

List<Users> users = new List<Users>();

public WorkngWithUsers()

{

InitializeComponent();

ListUsers.ItemsSource = db.Users.ToList();

var post = db.Post.ToList();

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

{

ComboBoxPost.Items.Add(post[i].Name);

}

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

{

ComboBoxPostAdd.Items.Add(post[i].Name);

}

}

private void Add\_Click(object sender, RoutedEventArgs e)

{

try

{

if (string.IsNullOrWhiteSpace(TextBoxSurnameAdd.Text) || string.IsNullOrWhiteSpace(TextBoxNameAdd.Text) || string.IsNullOrWhiteSpace(TextBoxPatronimycAdd.Text) || string.IsNullOrWhiteSpace(DataBirthdayAdd.Text) || string.IsNullOrWhiteSpace(TextBoxPhoneAdd.Text) || string.IsNullOrWhiteSpace(ComboBoxPostAdd.Text) || string.IsNullOrWhiteSpace(TextBoxLoginAdd.Text) || string.IsNullOrWhiteSpace(TextBoxPassAdd.Text))

{

MessageBox.Show("Все поля обязательны к заполнению!");

}

else

{

var us = new Users

{

Surname = TextBoxSurnameAdd.Text,

Name = TextBoxNameAdd.Text,

Patronymic = TextBoxPatronimycAdd.Text,

Birthday = DataBirthdayAdd.SelectedDate,

Phone = TextBoxPhoneAdd.Text,

Login = TextBoxLoginAdd.Text,

Password = TextBoxPassAdd.Text,

Post = ComboBoxPostAdd.SelectedIndex + 1,

};

db.Users.Add(us);

db.SaveChanges();

MessageBox.Show("Сотрудник успешно добавлен!");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private Users selectedPerson;

private void Edit2\_Click(object sender, RoutedEventArgs e)

{

if (selectedPerson != null)

{

// Вносим изменения в выбранный объект

selectedPerson.Surname = TextBoxSurname.Text;

selectedPerson.Name = TextBoxName.Text;

selectedPerson.Patronymic = TextBoxPatronimyc.Text;

selectedPerson.Birthday = DataBirthday.SelectedDate;

selectedPerson.Phone = TextBoxPhohe.Text;

ComboBoxPost.Text = Convert.ToString(dolSelection);

// Обновляем информацию в ListView

ListUsers.Items.Refresh();

try

{

db.SaveChanges();

MessageBox.Show("Данные сотрудника успешно обновлены и сохранены в базе данных!");

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка при сохранении данных: {ex.Message}");

}

}

}

private void Exit3\_Click(object sender, RoutedEventArgs e)

{

Admin admin = new Admin();

admin.Show();

Close();

}

private void search\_Click(object sender, RoutedEventArgs e)

{

try

{

string фамилия = search\_surname.Text;

// Ищем сотрудника в базе данных по фамилии

selectedPerson = db.Users.FirstOrDefault(p => p.Surname == фамилия);

if (selectedPerson != null)

{

// Если сотрудник найден, отображаем его данные в полях ввода

TextBoxSurname.Text = selectedPerson.Surname;

TextBoxName.Text = selectedPerson.Name;

TextBoxPatronimyc.Text = selectedPerson.Patronymic;

DataBirthday.SelectedDate = selectedPerson.Birthday;

TextBoxPhohe.Text = selectedPerson.Phone;

ComboBoxPost.Text = Convert.ToString(dolSelection);

}

else

{

MessageBox.Show("Сотрудник с такой фамилией не найден.");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private int dolSelection;

private void ComboBoxPost\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

try

{

if (ComboBoxPost.SelectedItem != null)

{

dolSelection = (int)ComboBoxPost.SelectedValue;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void ListUsers\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListUsers.SelectedItem != null)

{

// Сохраняем выбранный объект

selectedPerson = (Users)ListUsers.SelectedItem;

// Отображаем данные в полях ввода

TextBoxSurname.Text = selectedPerson.Surname;

TextBoxName.Text = selectedPerson.Name;

TextBoxPatronimyc.Text = selectedPerson.Patronymic;

DataBirthday.SelectedDate = selectedPerson.Birthday;

TextBoxPhohe.Text = selectedPerson.Phone;

ComboBoxPost.Text = Convert.ToString(dolSelection);

}

}

private void Exit1\_Click(object sender, RoutedEventArgs e)

{

Admin admin = new Admin();

admin.Show();

Close();

}

}

Окно «Администратор. Работа с мероприятиями»:

capitoshkaEntities db = new capitoshkaEntities();

public WorkingWithEvents()

{

InitializeComponent();

ListData.ItemsSource = db.Events.ToList();

var type = db.TypeOfEvents.ToList();

for (int i = 5; i < type.Count; i++)

{

Type.Items.Add(type[i].Name);

}

}

private Events selectedEv;

private void search\_Click(object sender, RoutedEventArgs e)

{

try

{

string name = search\_Name.Text;

// Ищем мероприятие в базе данных по наименованию

selectedEv = db.Events.FirstOrDefault(p => p.Name == name);

if (selectedEv != null)

{

// Если мероприятие найдено, отображаем его данные в полях ввода

Name.Text = selectedEv.Name;

Type.Text = Convert.ToString(selectedEv.IDTypeOfEvents);

StartData.SelectedDate = selectedEv.StartDate;

ExpiratioData.SelectedDate = selectedEv.ExpirationDate;

}

else

{

MessageBox.Show("Мероприятие с таким названием не найдено.");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void Edit2\_Click(object sender, RoutedEventArgs e)

{

string typeOfEvents = Convert.ToString(selectedEv.IDTypeOfEvents);

if (selectedEv != null)

{

// Вносим изменения в выбранный объект

selectedEv.Name = Name.Text;

typeOfEvents = Type.Text;

selectedEv.StartDate = StartData.SelectedDate;

selectedEv.ExpirationDate = ExpiratioData.SelectedDate;

// Обновляем информацию в ListView

ListData.Items.Refresh();

try

{

db.SaveChanges();

MessageBox.Show("Данные мероприятия успешно обновлены и сохранены в базе данных!");

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка при сохранении данных: {ex.Message}");

}

}

}

private int typeSelection;

private void Type\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ComboBoxTypeAdd.SelectedItem != null)

{

typeSelection = (int)ComboBoxTypeAdd.SelectedValue;

}

}

private void ListData\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

try

{

if (ListData.SelectedItem != null)

{

// Сохраняем выбранный объект

selectedEv = (Events)ListData.SelectedItem;

// Отображаем данные в полях ввода

Name.Text = selectedEv.Name;

Type.Text = Convert.ToString(selectedEv.IDTypeOfEvents);

StartData.SelectedDate = selectedEv.StartDate;

ExpiratioData.SelectedDate = selectedEv.ExpirationDate;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void Exit\_Click(object sender, RoutedEventArgs e)

{

Admin admin = new Admin();

admin.Show();

Close();

}

private void Add\_Click(object sender, RoutedEventArgs e)

{

try

{

if (ComboBoxTypeAdd.SelectedItem != null)

{

int usll = (int)ComboBoxTypeAdd.SelectedValue;

var qwerty = db.TypeOfEvents.FirstOrDefault(x => x.ID == usll);

using (var context = new capitoshkaEntities())

{

var ev = new Events

{

Name = TextBoxNameAdd.Text,

IDTypeOfEvents = typeSelection,

StartDate = DataStartAdd.SelectedDate,

ExpirationDate = DataExpirationAdd.SelectedDate

};

context.Events.Add(ev);

context.SaveChanges();

MessageBox.Show("Мероприятие успешно добавлено!");

}

}

else

{

MessageBox.Show("Ошибка");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void Exit3\_Click(object sender, RoutedEventArgs e)

{

Admin admin = new Admin();

admin.Show();

Close();

}

}

Окно «Педагог. Работа с учениками и родителями»:

capitoshkaEntities db = new capitoshkaEntities();

Child child = new Child();

public WorkingWithChild()

{

InitializeComponent();

ListChild.ItemsSource = db.Child.ToList();

var par = db.Parents.ToList();

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

{

ComboBoxParent.Items.Add(par[i].Surname);

}

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

{

ComboBoxParentAdd.Items.Add(par[i].Surname);

}

}

private Child selectedch;

private void Edit\_Click(object sender, RoutedEventArgs e)

{

if (selectedch != null)

{

// Вносим изменения в выбранный объект

selectedch.Surname = TextBoxSurname.Text;

selectedch.Name = TextBoxName.Text;

selectedch.Patronymic = TextBoxPatronimyc.Text;

selectedch.Birthday = DataBirthday.SelectedDate;

selectedch.IDParent = Convert.ToInt32(ComboBoxParent.Text);

// Обновляем информацию в ListView

ListChild.Items.Refresh();

try

{

db.SaveChanges();

MessageBox.Show("Данные о ребенке успешно обновлены и сохранены в базе данных!");

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка при сохранении данных: {ex.Message}");

}

}

}

private void search\_Click(object sender, RoutedEventArgs e)

{

try

{

string surname = search\_Surname.Text;

// Ищем мероприятие в базе данных по наименованию

selectedch = db.Child.FirstOrDefault(p => p.Surname == surname);

if (selectedch != null)

{

// Если мероприятие найдено, отображаем его данные в полях ввода

TextBoxSurname.Text = selectedch.Surname;

TextBoxName.Text =selectedch.Name;

TextBoxPatronimyc.Text = selectedch.Patronymic;

DataBirthday.SelectedDate = selectedch.Birthday;

ComboBoxParent.Text = Convert.ToString(selectedch.IDParent);

}

else

{

MessageBox.Show("Ребенка с такой фамилией не найдено.");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void Exit\_Click(object sender, RoutedEventArgs e)

{

Pedagog pedagog = new Pedagog();

pedagog.Show();

Close();

}

private void Exit3\_Click(object sender, RoutedEventArgs e)

{

Pedagog pedagog = new Pedagog();

pedagog.Show();

Close();

}

private void Add\_Click(object sender, RoutedEventArgs e)

{

try

{

if (string.IsNullOrWhiteSpace(TextBoxSurnameAdd.Text) || string.IsNullOrWhiteSpace(TextBoxNameAdd.Text) || string.IsNullOrWhiteSpace(TextBoxPatronimycAdd.Text) || string.IsNullOrWhiteSpace(DataBirthdayAdd.Text) || string.IsNullOrWhiteSpace(ComboBoxParentAdd.Text))

{

MessageBox.Show("Все поля обязательны к заполнению!");

}

else

{

using (var context = new capitoshkaEntities())

{

var us = new Child

{

Surname = TextBoxSurnameAdd.Text,

Name = TextBoxNameAdd.Text,

Patronymic = TextBoxPatronimycAdd.Text,

Birthday = DataBirthdayAdd.SelectedDate,

IDParent = ComboBoxParentAdd.SelectedIndex + 1,

};

context.Child.Add(us);

context.SaveChanges();

MessageBox.Show("Ребенок успешно добавлен!");

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void ListChild\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

try

{

if (ListChild.SelectedItem != null)

{

// Сохраняем выбранный объект

selectedch = (Child)ListChild.SelectedItem;

// Отображаем данные в полях ввода

TextBoxSurname.Text = selectedch.Surname;

TextBoxName.Text = selectedch.Name;

TextBoxPatronimyc.Text = selectedch.Patronymic;

DataBirthday.SelectedDate = selectedch.Birthday;

ComboBoxParent.Text = Convert.ToString(selectedch.IDParent);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private int parSelection;

private void ComboBoxParent\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ComboBoxParent.SelectedItem != null)

{

parSelection = (int)ComboBoxParent.SelectedValue;

}

}

private void AddPar\_Click(object sender, RoutedEventArgs e)

{

try

{

if (string.IsNullOrWhiteSpace(TextBoxSurnameAddPar.Text) || string.IsNullOrWhiteSpace(TextBoxNameAddPar.Text) || string.IsNullOrWhiteSpace(TextBoxPatronimycAddPar.Text) || string.IsNullOrWhiteSpace(DataBirthdayAddPar.Text) || string.IsNullOrWhiteSpace(TextBoxPlaceOfWorkAddPar.Text) || string.IsNullOrWhiteSpace(TextBoxPhoneAddPar.Text))

{

MessageBox.Show("Все поля обязательны к заполнению!");

}

else

{

using (var context = new capitoshkaEntities())

{

var pr = new Parents

{

Surname = TextBoxSurnameAdd.Text,

Name = TextBoxNameAdd.Text,

Patronymic = TextBoxPatronimycAdd.Text,

Birthday = DataBirthdayAdd.SelectedDate,

PlaceOfWork = TextBoxPlaceOfWorkAddPar.Text,

Phone = TextBoxPhoneAddPar.Text,

};

context.Parents.Add(pr);

context.SaveChanges();

MessageBox.Show("Родитель успешно добавлен!");

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

}

Окно «Администратор. Создание договора»:

capitoshkaEntities db = new capitoshkaEntities();

public FormDocuments()

{

InitializeComponent();

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

Admin admin = new Admin();

admin.Show();

Close();

}

private void Repwo(string subToReplace, string text, Word.Document worddoc)

{

var range = worddoc.Content;

range.Find.ClearFormatting();

range.Find.Execute(FindText: subToReplace, ReplaceWith: text);

}

private void Btn\_Click(object sender, RoutedEventArgs e)

{

try

{

if (string.IsNullOrWhiteSpace(fio.Text) || string.IsNullOrWhiteSpace(name.Text))

{

MessageBox.Show("Все поля обязательны к заполнению!");

}

else

{

string d = DateTime.Now.ToString();

//создание документа в ворде

var WordApp = new Word.Application();

WordApp.Visible = false;

var Worddoc = WordApp.Documents.Open(Environment.CurrentDirectory + @"\documentss.docx");

Repwo("{data}", d, Worddoc);

Repwo("{fio}", "Иванов Иван Иванович", Worddoc);

string data = DateTime.Now.ToString();

Repwo("{name} ", "Название", Worddoc);

Worddoc.SaveAs2(Environment.CurrentDirectory + @"\documentss.docx");

Worddoc.Close();

WordApp.Quit();

MessageBox.Show("Договор создан!");

}

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка при сохранении данных: {ex.Message}");

}

}

Код «Педагог. Добавление групп»

capitoshkaEntities db = new capitoshkaEntities();

List<Group> group = new List<Group>();

public ManagingGrpStd()

{

InitializeComponent();

var type = db.Direction.ToList();

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

{

ComboBoxTypeAdd.Items.Add(type[i].Name);

}

}

private void Add\_Click(object sender, RoutedEventArgs e)

{

try

{

if (string.IsNullOrWhiteSpace(TextBoxNameAdd.Text) || string.IsNullOrWhiteSpace(ComboBoxTypeAdd.Text))

{

MessageBox.Show("Все поля обязательны к заполнению!");

}

else

{

using (var context = new capitoshkaEntities())

{

var gr = new Group

{

Name = TextBoxNameAdd.Text,

IDDirection = typeSelection,

};

context.Group.Add(gr);

context.SaveChanges();

MessageBox.Show("Группа успешно добавлена!");

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private int typeSelection;

private void ComboBoxTypeAdd\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

try

{

if (ComboBoxTypeAdd.SelectedItem != null)

{

typeSelection = (int)ComboBoxTypeAdd.SelectedValue;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

private void Exit\_Click(object sender, RoutedEventArgs e)

{

Pedagog ped = new Pedagog();

ped.Show();

Close();

}