**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФГБОУ ВО «БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Информатика и программное обеспечение»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине

**«Базы данных»**

**Тема:** *АС “Отдел кадров”*

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст.преп. Коптенок Е.В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

Студент гр. О-18-ПРИ-РПС-Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Подгорняк А.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

**БРЯНСК 2021**

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc62078317)

[1. Анализ предметной области 4](#_Toc62078318)

[1.1. Описание предметной области 4](#_Toc62078319)

[1.2. Функциональные требования 4](#_Toc62078320)

[2. Проектирование базы данных 5](#_Toc62078321)

[2.1. Основные положения по проектированию баз данных 5](#_Toc62078322)

[2.2. Концептуальная модель базы данных 6](#_Toc62078323)

[2.3. Структурная схема базы данных 7](#_Toc62078324)

[3. Программная реализация базы данных 9](#_Toc62078325)

[3.1. Хранимая процедура, хранимая функция и представление 9](#_Toc62078326)

[4. Программная реализация системы 12](#_Toc62078327)

[4.1. Реализация основных элементов проекта 12](#_Toc62078332)

[5. Тестирование 15](#_Toc62078333)

[5.1. Описание методики тестирования 15](#_Toc62078334)

[5.2. Тест работы в штатных условиях 15](#_Toc62078335)

[5.3. Тест работы в экстремальных условиях, с некорректными данными 17](#_Toc62078336)

[6. Руководство пользователя 19](#_Toc62078337)

[6.1. Установка и запуск программы 19](#_Toc62078338)

[Заключение 20](#_Toc62078339)

[Список использованной литературы 21](#_Toc62078340)

# Введение

Разрабатываемая программная система имеет окно подключения к серверу базы данных, окно авторизации пользователя, главное меню, в котором отразим основные функции, помощь пользователю и сведения о программе.

Интерфейс содержит функции заполнения и редактирования базы данных, а также поиска нужной информации, фильтрации и сортировки.

База содержит 8 таблиц, которые будут обеспечивать достаточно рациональный уровень избыточности. Все таблицы базы заполнены данными. Основные таблицы содержат не менее 15 записей.

Также в программной системе доступен просмотр отчёта, созданного при помощи SQL Server Reporting Services.

Выполнение работы включало в себя следующие шаги:

1. Разработка концептуальной модели базы данных;
2. Разработка и создание базы данных на основе полученной концептуальной модели в СУБД MS SQL Server 2018;
3. Тестирование разработанной базы данных;
4. Разработка интерфейса для работы с базой данных в MS Visual Studio 2019.

# Анализ предметной области

## Описание предметной области

Многие компании имеют отдел кадров. Для эффективной работы этого подразделения необходима база данных, содержащая всю информацию о работниках предприятия, способная быстро отдать хранящиеся сведения.

Для решения поставленной задачи была выбрана реляционная база данных. Реляционная база данных — это совокупность отношений, содержащих всю информацию, которая должна храниться в БД. Пользователи воспринимают такую базу данных как совокупность таблиц, связанных между собой.

## Функциональные требования

Программа должна обеспечивать поддержку следующих возможностей обработки информации:

* Просмотр информации. Выводимая на экран информация должна иметь вид таблицы, с возможностью прокрутки содержимого.
* Добавление новой информации к уже имеющейся в базе данных.
* Редактирование введенной информации.
* Удаление информации.
* Поиск информации по определенным полям таблицы.
* Вывод результатов поиска в виде таблицы, разбитой на страницы.
* Составление отчётов.

Программа должна надежно функционировать при неправильных действиях пользователя. Юзер должен получать сообщение о возникновении соответствующей ошибки. Программа должна предлагать пользователю возможность:

* Игнорировать ошибку и продолжить работу с программой;
* Исправить ошибку (если это возможно) и повторно выполнить действие, приведшее к возникновению ошибки;
* Закончить работу с программой.

# Проектирование базы данных

## Основные положения по проектированию баз данных

**База данных***(database)*– электронное хранилище информации. Самые распространенные сейчас – реляционные базы данных – хранят данные в таблицах. Физически база данных представляет собой один или несколько файлов специального формата.

**СУБД, Система Управления Базами Данных** (*DBMS – database management system*) – комплекс программ, предназначенный для создания и сопровождения баз данных. СУБД способна одновременно управлять несколькими базами данных, с которыми в одно и то же время, параллельно, работают многие пользователи. Обычно СУБД включает в себя следующие программы:

* ***ядро СУБД*** – программа или служба, которая постоянно «висит» в памяти и занимается обслуживанием поступивших от пользователей запросов на обработку данных (найти данные по критерию поиска, вставить, удалить или изменить строки таблицы и т.д.).
* ***программы для создания и сопровождения баз данных*** – интерактивные программы, где можно просмотреть содержимое баз данных, находящихся под управлением данной СУБД, отредактировать его, зарегистрировать новых пользователей базы данных и разрешить им работать с определенными таблицами, создавать и удалять базы данных и др..
* ***мониторы производительности*** – программы, с помощью которых можно просмотреть в реальном времени загрузку СУБД, статистику наиболее часто выполняющихся запросов, выявить запросы, требующие большого времени на выполнение и их оптимизировать.

**Приложение базы данных** (*database application*) – это программа, с помощью которой пользователи работают с базой данных. Приложение обычно пишется на языках высокого уровня (Java, Delphi, С++, C#) или с применением Web-технологий (HTML и язык web-сценариев: PHP, Python, JavaScript, Perl).

Приложение направляет SQL-запросы к СУБД, в ответ получает данные и отображает их в удобном для пользователя виде.

## Концептуальная модель базы данных

Концептуальная модель определяет информацию о предметной области в формализованной абстрактной форме. Модель должна быть представлена в графическом виде.

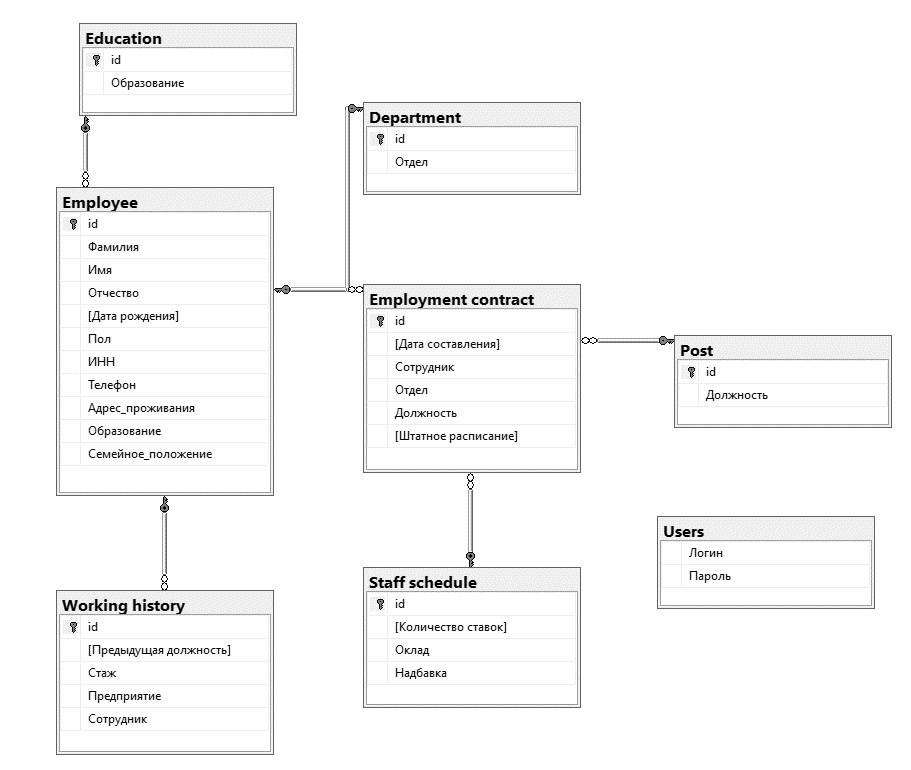


Рис. 1. Концептуальная модель базы данных

## Структурная схема базы данных

Для решения поставленной задачи была создана база данных «Human Resources» содержащая следующие активные таблицы:

1. Таблица рабочих «Employee»:
   1. id (int) – первичный ключ;
   2. Фамилия (nchar(20)) – фамилия рабочего;
   3. Имя (nchar(20))-имя рабочего;
   4. Отчество(nchar(20))-отчество рабочего;
   5. Дата рождения (date)–день рождения рабочего;
   6. Пол(nchar(1)) – гендер рабочего;
   7. ИНН (nchar(12)) – идентификационный номер налогоплательщика;
   8. Телефон (nchar(12)) – мобильный телефон рабочего;
   9. Адрес\_проживания (nvarchar(50)) - место жительство рабочего;
   10. Образование (int) – номер образования из таблицы-справочника “Education”;
   11. Семейное\_положение(nvarchar(30)) – статус рабочего в семье;
2. Трудовой договор «Employment contract»:
   1. id (int) – первичный ключ;
   2. Дата составления (date) – дата заключения трудового договора;
   3. Сотрудник (int) – id сотрудника из таблицы “Employee”;
   4. Отдел (int) - id отдела из таблицы “Department”;
   5. Должность (int) - id должности из таблицы “Post”;
   6. Штатное расписание (int) – id расписания из таблицы “Staff schedule”;
3. Трудовая история «Working history»:
   1. id (int) – первичный ключ;
   2. Предыдущая должность (nchar(50)) – прежнее место работы, если имеется;
   3. Стаж (int) – количество лет отработанных на должности;
   4. Предприятие (nchar(50)) – наименование прежнего места работы;
   5. Сотрудник (int) – id сотрудника из таблицы “Employee”;
4. Штатное расписание «Staff schedule»:
   1. id (int) – первичный ключ;
   2. Количество ставок (int) – количество занимаемых должностей;
   3. Оклад (money) - размер денежной выплаты;
   4. Надбавка (money) – дополнительные выплаты;
5. Образование «Education»:
   1. id (int) – первичный ключ;
   2. Образование (nchar(50)) – полное название имеющегося образования;
6. Департамент «Department»:
   1. id (int) – первичный ключ;
   2. Отдел (nchar(30)) – полное название отдела работника;
7. Должность «Post»:
   1. id (int) – первичный ключ;
   2. Должность (nchar(30)) – полное название занимаемой работником должности;
8. Пользователи «Users»:
   1. Логин (nvarchar(50));
   2. Пароль (nvarchar(50));

# Программная реализация базы данных

## Хранимая процедура, хранимая функция и представление

Хранимая процедура — это специальный тип пакета инструкций Transact-SQL, созданный, используя язык SQL и процедурные расширения. Основное различие между пакетом и хранимой процедурой состоит в том, что последняя сохраняется в виде объекта базы данных. Иными словами, хранимые процедуры сохраняются на стороне сервера, чтобы улучшить производительность и постоянство выполнения повторяемых задач.

Компонент Database Engine поддерживает хранимые процедуры и системные процедуры. Хранимые процедуры создаются таким же образом, как и все другие объекты баз данных, т.е. при помощи языка DDL. Системные процедуры предоставляются компонентом Database Engine и могут применяться для доступа к информации в системном каталоге и ее модификации.

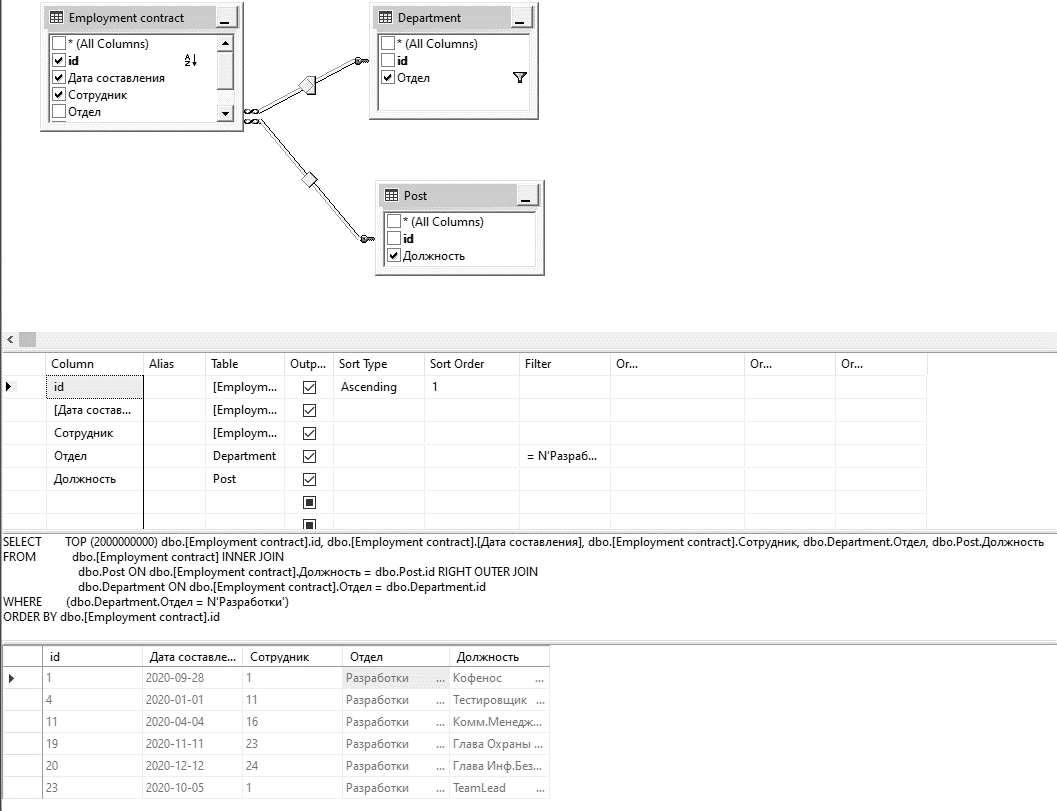
При создании хранимой процедуры можно определить необязательный список параметров. Таким образом, процедура будет принимать соответствующие аргументы при каждом ее вызове. Хранимые процедуры могут возвращать значение, содержащее определенную пользователем информацию или, в случае ошибки, соответствующее сообщение об ошибке.

В рамках данной курсовой работы были созданы хранимая процедура, хранимая функция, представление:

* + 1. представление.
    2. хранимая процедура с входным и выходными параметрами.
    3. хранимая функция с входным и выходным параметром.

Данное представление осуществляет поиск всех сотрудников из отдела разработки.

Протестируем данное представление



**Рис. 2. Результат выполнения представления**

Листинг 1

|  |
| --- |
| ALTER PROCEDURE [dbo].[CountGender](@g nchar)    AS  BEGIN  SET NOCOUNT ON;  SELECT \* from Employee where Пол=@g  END |

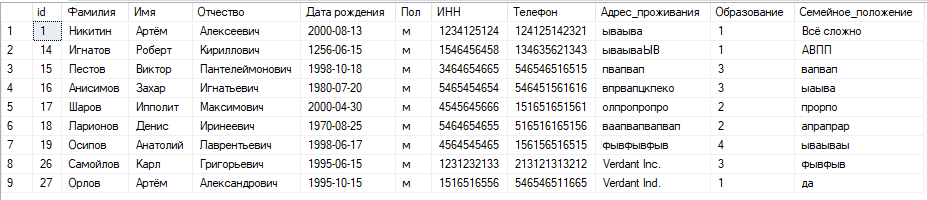
Создадим хранимую процедуру, которая имеет входной и выходные параметры. Данная хранимая процедура осуществляет выборку рабочих по гендеру (см. Листинг 1).

Рис. .Результат выполнения представления

Листинг 2

|  |
| --- |
|  |

Листинг 2. Хранимая функция с входным и выходным параметром

# Программная реализация системы

Программная реализация представляет собой настольное (desktop) приложение, разработанное на основе технологии создания приложений Microsoft .NET с использованием языка программирования C#. В качестве СУБД для реализации хранилища данных, был использован Microsoft SQL Server 2018.

Microsoft .NET – программная технология, для создания как обычных программ, так и веб-приложений, в качестве платформы для разработок создана фирмой Microsoft.

Одной из основных идей Microsoft .NET является совместимость различных приожений, написанных на разных языках. Например, приложение, написанное на C++ для Microsoft .NET, может обратиться к методу класса из библиотеки, написанной на Delphi; на C# можно написать класс, наследующий от класса, написанного на Visual Basic .NET, а исключение, созданное методом, написанным на C#, может быть перехвачено и обработано в Delphi. Благодаря совместимости программных компонентов, написанных на различных языках программирования, при написании плагина для системы, возможен выбор языка программирования, который наиболее подходит для реализации того или иного алгоритма интеллектуального анализа.

Каждая библиотека (сборка) в .NET имеет сведения о своей версии, что позволяет устранить возможные конфликты между разными версиями сборок. Эта возможность позволяет разработчикам плагинов выпускать новые версии плагинов, которые не будут конфликтовать со старыми.

C# – язык программирования, сочетающий объектно-ориентированные и аспектно-ориентированные концепции. Разработан в 1998—2001 в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET.



## Реализация основных элементов проекта

Листинг 3

|  |
| --- |
| private void buttonIn\_Click(object sender, EventArgs e)  {  DataView dv;  string login;  string pass;  dv = human\_ResourcesDataSet.Users.DefaultView;  login = textBoxLogin.Text;  pass = textBoxPass.Text;  dv.RowFilter= string.Format("Логин='{0}' and Пароль='{1}'", login,pass);  if(dv.Count == 1)  {  Hide();  var main = new FormMain(login);  main.ShowDialog();  Close();  }  else  MessageBox.Show("Логин или пароль введены не правильно", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);  } |

Листинг 3. Проверка данных при авторизации

Листинг 4

|  |
| --- |
| private void buttonFindEmployee\_Click(object sender, EventArgs e)  {  try  {  DataView dv;  string col;  string value;  if(comboBoxEmployee.SelectedItem == null)  {  throw new KadrLib.Exception.Exception("Не выбраны значения");  }  if(textBoxEmployee.Text=="")  {  throw new KadrLib.Exception.Exception("Не выбраны значения");  }  dv = human\_ResourcesDataSet.Employee.DefaultView;  if (comboBoxEmployee.SelectedIndex == 0)  col = "Фамилия";  else if (comboBoxEmployee.SelectedIndex == 1)  col = "Имя";  else  col = "Отчество";  value = textBoxEmployee.Text;  dv.RowFilter = string.Format("{0}='{1}'", col, value);  employeeDataGridView.DataSource = dv;  }  catch (System.Exception exception)  {    MessageBox.Show(exception.Message,"Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);  }  } |

Листинг 4. Поиск записей по критериям

Листинг 5

|  |
| --- |
| private void buttonAddSchedule\_Click(object sender, EventArgs e)  {  try  {  DataRow Row = human\_ResourcesDataSet.Staff\_schedule.NewRow();  Row[1] = numericUpDownBid.Value;  Row[2] = numericUpDownPayment.Value;  Row[3] = numericUpDownExtra.Value;  if(numericUpDownBid.Value==0)  {  throw new KadrLib.Exception.Exception("Поле 'Кол-во ставок' должно быть > 0 ");  }else if(numericUpDownPayment.Value==0)  {  throw new KadrLib.Exception.Exception("Поле 'Оклад' должно быть > 0 ");  }  human\_ResourcesDataSet.Staff\_schedule.Rows.Add(Row);  tableAdapterManager.UpdateAll(human\_ResourcesDataSet);  human\_ResourcesDataSet.AcceptChanges();  }catch(System.Exception exception)  {  MessageBox.Show(exception.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);    }  } |

Листинг 5. Добавление записи в таблицу

Листинг 6

|  |
| --- |
| private void buttonGenderSort\_Click(object sender, EventArgs e)  {  try  {  if (comboBoxSortGender.SelectedItem == null)  throw new KadrLib.Exception.Exception("Не выбран гендер");  employeeTableAdapter1.FillByGender(human\_ResourcesDataSet.Employee, comboBoxSortGender.SelectedItem.ToString());  }  catch (System.Exception exception)  {  MessageBox.Show(exception.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);  }  } |

Листинг 6. Сортировка таблицы с использованием хранимой процедуры

# Тестирование

## Описание методики тестирования

Проводим тесты в штатных условиях и в условиях, когда намеренно используем некорректные данные при работе с программной системой.

## Тест работы в штатных условиях

Открываем программу, подключаемся к серверу базы данных, авторизуемся под разными ролями. Просматриваем все таблицы; добавляем, редактируем и удаляем строки в таблицах. Проверяем обработку корректности ввода данных в ячейках всех таблиц. Делаем поиск по критерию в таблицах.

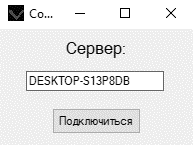
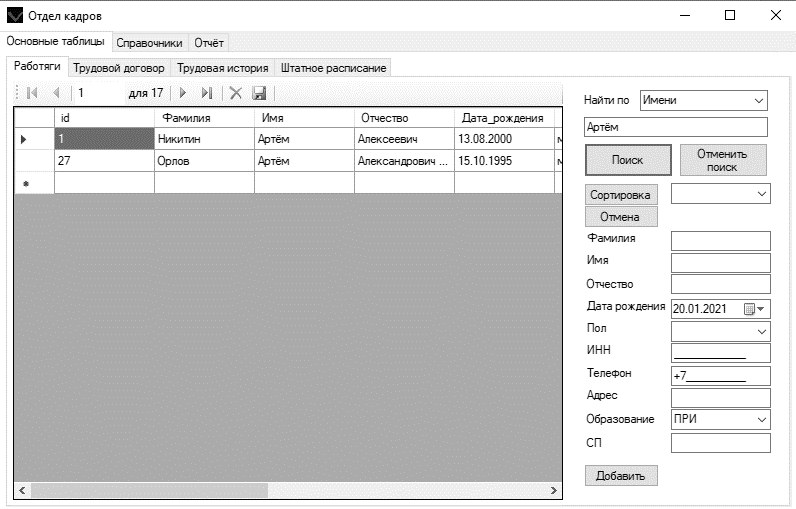


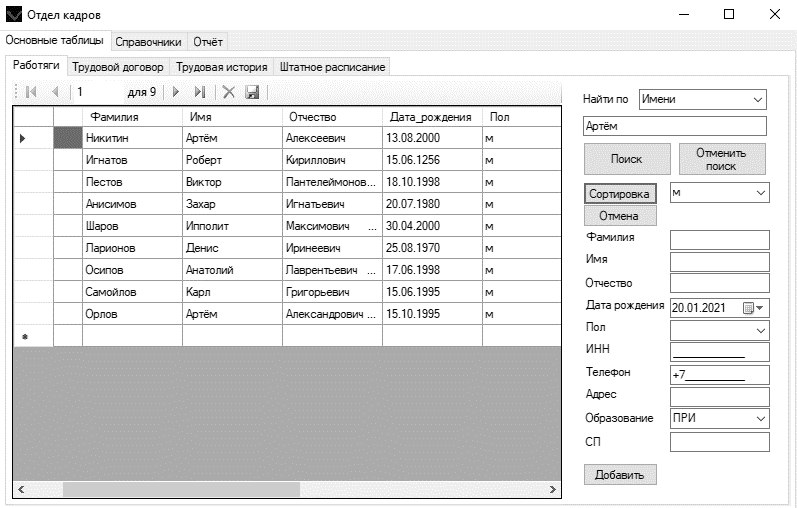
Рис. 4. Окно подключения к серверу



**Рис. 5. Окно авторизации**



**Рис. 6. Поиск по имени в таблице “Employee”**



**Рис. 7. Сортировка значений с помощью хранимой функции**

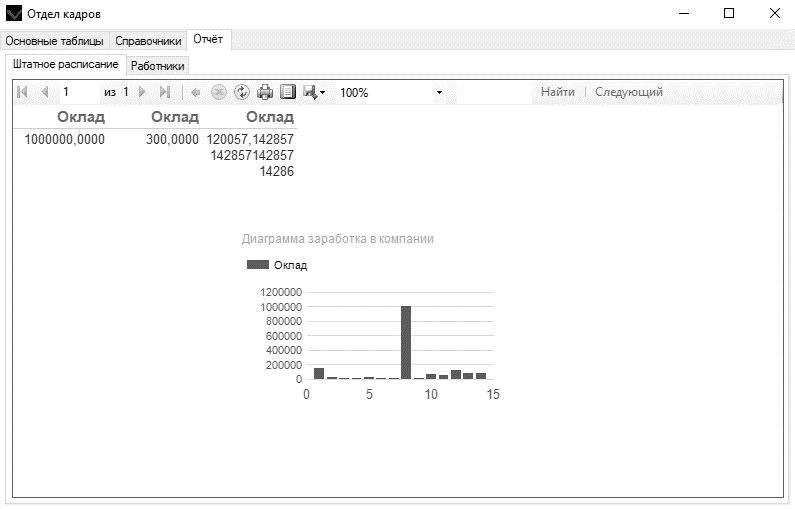


Рис. . Отчёт по таблице “Штатное расписание”

## Тест работы в экстремальных условиях, с некорректными данными

Тестируем программу так же, как и в штатном режиме, но с намеренно некорректными данными.

Тестируем подключение к серверу:

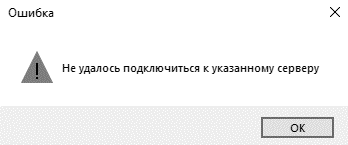
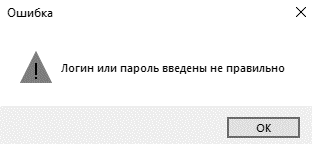


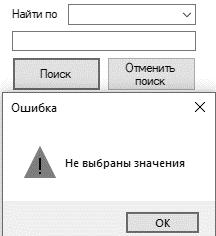
Рис. 9. Подключение к несуществующему серверу

Тестируем вход в базу данных:

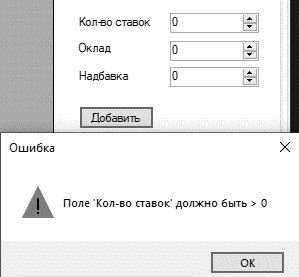


**Рис. 10. Вход в базу данных при неверно введенном пароле\логине**

Тестируем поиск данных в основных таблицах по критерию:



**Рис. 11. Попытка поиска по полю, не введя критерий поиска**



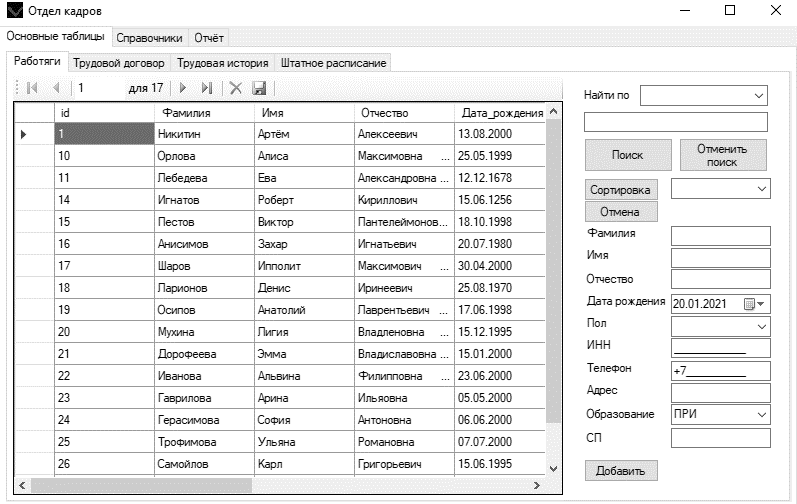
**Рис. 12. Попытка добавить запись, не заполнив данные**

# Руководство пользователя

## Установка и запуск программы

Для того что бы запустить программу, надо открыть exe–файл. После этого открывается окно подключения к серверу, затем окно авторизации, где пользователь должен будет ввести свои данные для входа в систему

При входе от имени администратора в базу данных, ему будет предоставлен полный список возможностей, таких как: изменение записей, удаление записей из таблиц, добавление новых записей, удаление дубликатов из таблицы. А для пользователя лишние элементы скрываются.



**Рис. 13. Окно приложения при входе от имени пользователя**

Также, пользователю или администратору доступен поиск в основных таблицах по определенному критерию. При некорректном значении поиска, или пустом его значении, пользователю будет выведено диалоговое окно с ошибкой:

# Заключение

В данной курсовой работе с помощью СУБД MS SQL Server 2018 была создана база данных с предметной областью «Отдел кадров». Разработанная база данных упрощает и частично автоматизирует хранение информации о работниках.

Так же в ходе данной курсовой работы был разработан программный интерфейс на языке C# c использованием технологии Windows Forms. Данный интерфейс полностью соответствует техническому заданию. Программа работает корректно, никаких ошибок выявлено не было.

# Список использованной литературы

1. Бейли Л. Изучаем SQL / Л. Бейли. ­– СПб.: Изд-во Питер, 2012 – 573c.
2. Бондарев В. Язык SQL. Базовый курс / В. Бондарев. ­– СПб.: Изд-во Postgres Professional, 2017 – 256c.
3. Основы SQL: запросы к базе данных. – Режим доступа: https://interocitor.livejournal.com/17721.html.
4. Руководство по MS SQL Server. – Режим доступа: https://metanit.com/sql/sqlserver/.
5. Введение в базы данных. – Режим доступа: https://compress.ru/article.aspx?id=12460.
6. Джон, Ш. Microsoft Visual C#. Подробное руководство / Ш. Джон. СПб.: Изд-во Питер, 2017. – 848с.
7. Петцольд, Ч. Программирование с использованием Microsoft Windows Forms / Ч. Петцольд. ­– СПб.: Изд-во Питер, 2016 – 432c.