Министерство образования и науки Республики Башкортостан

Государственное бюджетное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2021 г. |

ЗАДАНИЕ

на курсовой проект студенту дневного отделения, группы 18П-1

специальности 09.02.03 Программирование в компьютерных системах

Фамилия, имя, отчество: Командов Максим Олегович

Тема курсового проекта: «Проектирование базы данных для учета заработной платы сотрудников»

Текст задания:

при выполнении курсового проекта должны быть решены следующие задачи:

1. спроектирована база данных;
2. разработана структура программы;
3. реализованы функции расчёта зарплаты, налогов, отчислений и налоговых льгот, формирования расчётного листа.

В результате выполнения курсового проекта должны быть представлены:

1. пояснительная записка, состоящая из следующих разделов:

Введение

1 Постановка задачи

2 Экспериментальный раздел

Заключение

Приложения

Список сокращений

Список источников

1. электронный носитель, содержащий разработанный программный продукт;
2. презентация курсового проекта в электронном виде.

Список рекомендуемых источников:

1. Култыгин, О. П. Култыгин, О. П. Администрирование баз данных. СУБД MS SQL Server [Текст] : учеб. пособ. / О. П. Култыгин. - М.: МФПА, 2012. - 232 с.
2. Фуфаев, Э.В. Базы данных [Текст]: учеб. пособ. для студ. учрежд. сред. проф. образования / Э.В. Фуфаев, Д.Э. Фуфаев. - 6-е изд., стер. - М.: Издательский центр «Академия», 2012.- 320 с.- (Среднее профессиональное образование)
3. Википедия [Электронный ресурс] // Свободная энциклопедия. – Режим доступа: <http://ru.wikipedia.org/wiki/>, свободный

Задание к выполнению получил «29» января 2021 г.

Студент Командов Максим Олегович

Срок окончания «30» мая 2021 г.

Руководитель курсового проекта Р.Ф. Каримова

Задание рассмотрено на заседании цикловой комиссии информатики

«11» января 2021 г.

Председатель цикловой комиссии информатики О.В.Фатхулова

Министерство образования и науки Республики Башкортостан

Государственное бюджетное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

ЗАКЛЮЧЕНИЕ

на курсовой проект

Студент Командов Максим Олегович

Группа 18П-1

Специальность 09.02.03 Программирование в компьютерных системах

Тема Проектирование базы данных для учета заработной платы сотрудников

Объем курсового проекта:

количество листов пояснительной записки

количество листов графической части

Заключение о степени соответствия заданию на курсовое проектирование

Характеристика качеств, проявленных студентом при работе над проектом: самостоятельность, дисциплинированность, умение планировать работу и пользоваться литературным материалом и т.д.

Положительные стороны курсового проекта

Недостатки курсового проекта

Характеристика общетехнической и специальной подготовки студента

Заключение и предлагаемая оценка за курсовой проект

Руководитель курсового проекта Каримова Резида Флюновна

«\_\_\_\_» 2021 г.

Подпись

АННОТАЦИЯ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

40.К2123 -21 09.02.03 КП-П3

Разраб.

Командов М.О.

Провер.

Каримова Р.Ф.

Реценз.

Н. Контр.

Утверд.

Проектирование базы данных для учета заработной платы сотрудников

Лит.

Листов

УКСИВТ 18П-1

Пояснительная записка к курсовому проекту содержит постановку и программу решения задачи «Проектирование базы данных для учета заработной платы сотрудников»

Программа BDZarplata.exe написана на языке C# в среде программирования Visual Studio 2019 с использованием сервера баз данных MS SQL SERVER, предназначена для работы в операционной системе MS Windows 10, отлажена на данных контрольного примера.

ВВЕДЕНИЕ

Среди наиболее трудоемких участков бухгалтерского учета особое место занимают расчеты с персоналом по оплате труда. В роли объектов учета могут выступать десятки, сотни и даже тысячи человек, по каждому из которых нужно учитывать и обрабатывать достаточно большие объемы данных.

Актуальность данной работы обусловлена тем, что расчет заработной платы сотрудников производится бухгалтерами либо с помощью программы «1С-бухгалтерия», либо вручную. Так как программа «1С-бухгалтерия» очень сложна в применении, и ее может освоить не каждый бухгалтер, то расчет заработной платы производится с помощью электронных таблиц. Это довольно трудоемкий процесс, который занимает очень много времени и ресурсов компьютера. Данная курсовая работа, ориентирована на простое освоение и комфортную работу бухгалтеров.

Цель курсового проекта – разработка приложения для упрощения процесса учета заработной платы бухгалтерам, путем автоматизации расчёта заработной платы, а также проверки корректности вводимых данных.

Задачами курсового проекта являются:

* описать предметную область;
* разработать структуру базы данных;
* разработать приложение;
* провести тестирование приложения.

1. Постановка задачи
   1. Описание предметной области

Задача – автоматизировать расчёт зарплаты, содержание работника и суммы налогов, упростить процесс выплаты зарплаты для упрощения работы Бухгалтера

Информационная система должна обеспечивать: ввод, изменение анкетных данных работников, сведения о болезнях, надбавках; ежемесячный перерасчет зарплаты с выдачей ведомости на экран и печать.

С общей суммы зарплаты отчисляется подоходный налог.

Круг пользователей

* Сотрудники Отдела кадров имеют право:
  + добавлять новых работников в БД;
  + изменять ФИО и другие анкетные данные сотрудников;
  + изменять должность работников;
  + изменять статус работника.
* Сотрудники бухгалтерии имеют право:
  + изменять оклад в соответствии с должностью;
  + изменять показатели процента налогов;
  + Изменять константы влияющие на расчет зарплаты;
  + создавать выписки по ЗП;
  + начислять штрафы и надбавки работникам соответствующих отделов.
* Рядовые сотрудники имеют право:
  + получать выписку по ЗП.

Константы

МРОТ 12 792 руб.

Налоги, которые платит предприятие:

Пенсионный фонд. Тариф составляет 22% для работников, доход которых за год не превышает 1,292 млн рублей и 10% — с больших сумм.

Фонд соцстрахования. Платеж составляет 2,9% для работников, которые в ход получают заработную плату в размере до 912 тыс. рублей. Если зарплата больше, на сумму больше предельной платеж не начисляется.

Медицинское страхование. Размер ежемесячного платежа составляет 5,1%

Существует также платеж «на травматизм». Процентная ставка определяется в зависимости от того, в какой сфере деятельности работает компания. Минимальная ставка составляет 0,2%, максимальная 8,5%.

Промежуточные данные

Информация о ЗП:

* Доплата за выход в праздник(выходной);
* Льготы;
* Льготы за детей (ндфл);
* Вычет на ребенка (детей) предоставляется до месяца, в котором доход налогоплательщика, облагаемый по ставке 13% и исчисленный нарастающим итогом с начала года, превысил 350 000 рублей. Вычет отменяется с месяца, когда доход сотрудника превысил эту сумму;
* на первого и второго ребенка – 1400 рублей;
* на третьего и каждого последующего ребенка – 3000 рублей;
* на каждого ребенка-инвалида до 18 лет, или учащегося очной формы обучения, аспиранта, ординатора, интерна, студента в возрасте до 24 лет, если он является инвалидом I или II группы – 12 000 рублей родителям и усыновителям (6 000 рублей – опекунам и попечителям);
* Процент налогов на зарплату в валюте (НДФЛ для физ лиц = 13% \*налоговую базу (налоговая база = оклад+надбавки –налоговый вычет ))
* Процент пенсионных вычетов;
* Итоговая ЗП работника;
* Стоимость содержания работника для предприятия.

Расчеты

Работник получает ЗП согласно окладу и фактически отработанным дням

Найдем дневной доход = оклад / количество рабочих дней в месяце[[1]](#footnote-1)

Найдем фактический дневной доход сотрудника, учитывая статус дня в месяце:

* Если статус сотрудника = «вышел» а статус дня «рабочий» - начисляется дневной доход без изменений;
* Если статус сотрудника = «вышел» а статус дня «выходной» - начисляется дневной доход в 2х кратном объеме;
* Если статус сотрудника = «болеет» а статус дня «рабочий» или «выходной» - начисляется дневной доход в соответствие с стр. 8 ниже;
* В ином случае дневной доход не начисляется.

Найдем фактический оклад в месяц сложив все дневные доходы в конкретном месяце.

Найдем фактический доход= фактический оклад – штрафы +надбавки

Найдем налоговую базу (НБ), изначально равную Фактическому Доходу, используя налоговые вычеты, зависящие от указанных данных в таблице сотрудник:

Если сотрудник имеет спец статус 1, то НБ = НБ - 3000 руб

Если сотрудник имеет спец статус 2, то НБ = НБ - 500 руб

Если сотрудник имеет детей не инвалидов[[2]](#footnote-2) :Если детей менее 3 то НБ=НБ-1400\*(Число детей), Если детей 3 и более то НБ = 2800 + 3000 \* (Число детей-2)

Если сотрудник имеет детей инвалидов НБ = НБ - 12 000 \* (число детей инвалидов)

Если сотрудник имеет опекунство над детьми инвалидами   
НБ = НБ - 6000 \* (число детей инвалидов)

Если сотрудник имеет семейный статус «мать-одиночка» или «отец-одиночка», то выше описанные налоговые вычеты удваиваются, соответственно:

НБ = НБ - (1400 \* (Число детей) \* 2)  
НБ = НБ - (5600 + 3000 \* (Число детей - 2) \* 2)  
НБ = НБ - 12 000 \* (число детей инвалидов) \* 2  
НБ = НБ - 6000 \* (число детей инвалидов) \* 2

Рассчитаем НДФЛ = НБ \* 13% или НДФЛ = НБ – (НБ / 100 \* 13)

Итоговая зарплата («на руки») = фактический доход – НДФЛ

Расчёт содержания сотрудника = фактический доход + ПФР + Соцстрах + МедСтрах + травматизм

ПФР = фактический доход \* 22%

Соцстрах = фактический доход \* 2,9%

Медстрах = фактический доход \* 5,1%

Травматизм = фактический доход \* процент травматизма

Расчёт больничного

Ср. заработок = зарплата за 2 года[[3]](#footnote-3) (24 месяца) / 730 \* размер больничного[[4]](#footnote-4)

Если количество больничных дней ≤ 3: Размер пособия (работодатель) = количество больничных дней\* Ср. заработок

Если количество больничных дней > 3:

Размер пособия (работодатель) = 3 \* Ср. заработок

Размер пособия (ФСС) = (количество больничных дней -3) \* Ср. заработок

В случае если:

* в расчетном периоде у работника нет заработка;
* заработок работника за полный календарный месяц(Оклад) ниже МРОТ;
* стаж работника менее 6 месяцев;
* работник нарушил режим, предписанный врачом;
* больничный лист выдан вследствие алкогольного, наркотического или токсического опьянения.

Расчёт больничного идет из МРОТ:

Если количество больничных дней ≤ 3:  
Размер пособия (работодатель) = количество больничных дней \* МРОТ

Если количество больничных дней > 3:

Размер пособия (работодатель) = 3 \* МРОТ

Размер пособия (ФСС) = (количество больничных дней - 3) \* МРОТ

Итоговый расчет больничного:

НДФЛ(работодатель) = Размер пособия (работодатель) \* 13%

НДФЛ(ФСС) = размер пособия \* 13%

Итоговый размер пособия = Размер пособия (работодатель) - НДФЛ(работодатель) +  
Размер пособия (ФСС) - НДФЛ(ФСС)

* 1. Описание входной информации

Входной информацией для выполнения задачи являются данные непосредственно вносимые сотрудниками Бухгалтерии и отдела кадров:

* Предельные базы для исчисления страховых взносов:
  + Год;
  + Значение базы.
* Справочник (хранящий различные константы):
  + Сумма выплаты за 1 ребенка;
  + Сумма выплаты за 3 и более детей;
  + Сумма выплаты за 1 ребенка инвалида;
  + Сумма выплаты за опекунство над 1 ребенком инвалидом;
  + Процент отчислений в пенсионный фонд;
  + Процент отчислений в фонд соц. Страхования;
  + Процент отчислений в фонд мед. Страхования;
  + Процент НДФЛ;
  + МРОТ.
* Информация о сотрудниках:
  + - Страховой стаж;
    - Табельный номер[[5]](#footnote-5);
    - Ф.И.О.;
    - семейное положение (не обязательно);
    - число здоровых детей (не обязательно);
    - число детей инвалидов (до 18 лет), или учащихся очной формы обучения, аспирантов, ординаторов, интернов, студентов в возрасте до 24 лет, если они являются инвалидами I или II группы (не обязательно);
    - число опекаемых детей инвалидов (не обязательно)[[6]](#footnote-6);
    - счет зачисления (20 символов);
    - код должности.
    - наличие спец статуса:
      * статус предоставляющий налоговый вычет в размере 3000 Ꝑ;
      * статус предоставляющий налоговый вычет в размере 500 Ꝑ [[7]](#footnote-7).
* Информация о должностях и окладах:
  + Код должности;
  + Название должности;
  + Оклад (в единой Валюте);
  + Процент травматизма (от 0,2% до 8,5%);
  + Уровень доступа.
* Информация о ЗП:
  + Табельный номер;
  + Дата выплаты;
  + Надбавки [[8]](#footnote-8)(сумма всех надбавок) (в единой Валюте);
  + Штрафы (сумма всех штрафов) (в единой Валюте)[[9]](#footnote-9).
* График Работы:
  + Дата;
  + Табельный номер;
  + Статус сотрудника (уволен / вышел / не вышел / болеет (другая уважительная причина));
  + Статус дня (рабочий / выходной).
  1. Описание выходной информации

Выходной информацией является расчётный лист

Описание выходных документов представлено в таблице 1.3.1.

Таблица 1.3.1 – Описание выходных документов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Наименование документа (шифр) | Периодичность выдачи документа | Кол-во экз. | Куда передаются | Поля сортировки | Поля группировки | Итоги |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Расчётный лист | Раз в месяц или по мере необходимости | 1 | Сотруднику | - | - | - |

Шаблоны выходных документов представлены в приложении А

* 1. Концептуальное моделирование

Концептуальная модель – это отражение предметной области, для которой разрабатывается база данных.

Можно сказать, что это некая диаграмма с принятыми обозначениями элементов. Так, все объекты, обозначающие вещи, обозначаются в виде прямоугольника. Атрибуты, характеризующие объект – в виде овала, а связи между объектами – ромбами. Мощность связи обозначается стрелками (в направлении, где мощность равна многим – двойная стрелка, а со стороны, где она равна единице - одинарная).

Концептуальная модель базы данных представлена в схеме 1.4.1



Рисунок 1.4.1 – Концептуальная модель БД

* 1. Логическое моделирование

При логическом моделировании происходит окончательное определение структуры данных, определяются ограничения, накладываемые на эти данные, целью которых является обеспечить целостность данных. Наиболее распространенной моделью данных является реляционная модель. В этой модели каждая сущность представляется в виде таблицы.

Логическое моделирование заключается в переходе от диаграммы «сущность-связь» к взаимосвязанным таблицам. Этот переход состоит из следующих шагов:

* каждая простая сущность становится таблицей;
* каждый атрибут становится столбцом таблицы;
* уникальный идентификатор сущности становится ключом таблицы;
* преобразование связи;
* сущности, связанные обязательной связью один к одному можно объединить в одну таблицу;
* связи типа один к одному возможные и связи типа один ко многим реализуются путем переноса ключей атрибутов таблиц соответствующих сущностей, стоящих со стороны один в таблице соответствующих сущностей, стоящих со стороны многие;
* связи типа многие ко многим реализуются при помощи промежуточных таблиц, содержащих ключевые атрибуты связываемых таблиц в качестве внешних ключей.

Схема данных – это структура базы данных, описанная на формальном языке, поддерживаемая СУБД (системой управления базы данных). В реляционных базах данных схема определяет таблицы, поля в каждой таблице и ограничения целостности, такие как первичный и внешний ключи.

Схема данных представлена на рисунке 1.5.1.



Рисунок 1.5.1 – Схема данных БД

* 1. Описание структуры базы данных

Описание структуры базы данных представлено в таблице 1.6.1.

Таблица 1.6.1 - Описание структуры базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описание поля | Тип данных | Размер поля | Тип ключа[[10]](#footnote-10) |
| 1 | 2 | 3 | 4 | 5 |
| Sotrudnik (Сотрудник) | | | | |
| idSotrudnik | Табельный номер сотрудника | INT | 4 | PK |
| full\_name | ФИО сотрудника | Varchar(45) | 46 |  |
| idDoljnost | ID Должности | INT | 4 | FK |
| family\_status | Семейный статус сотрудника | VARCHAR(45) | 46 |  |
| num\_zd\_kids | Количество здоровых детей сотрудника | TINYINT | 1 |  |
| num\_invalid\_kids | Количество детей инвалидов | TINYINT | 1 |  |
| opeka | Число опекаемых детей инвалидов | TINYINT | 1 |  |
| SchetZchisl | Счёт зачисления сотрудника | CHAR(20) | 21 |  |

Продолжение таблицы 1.6.1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | | 2 | | 3 | | 4 | | 5 |
| SpecStatus | | Наличие или отсутствие одного из спец статусов | | TINYINT | | 1 | |  |
| Staj | | Страховой стаж | | INT | | 4 | |  |
| Doljnost(Должность) | | | | | | | | |
| idDoljnost | ID Должности | | INT | | 4 | | PK | |
| title | Название должности | | Varchar(45) | | 46 | |  | |
| Oklad | оклад | | INT | | 4 | |  | |
| Travmat | Процент травматизма | | DECIMIAL(2,1) | |  | |  | |
| AccessLvl | Уровень доступа к данным | | TINYINT | |  | |  | |
| Graphik\_Rabot(График работ) | | | | | | | | |
| DATE | Дата дня | | DATE | | 3 | | PK | |
| idSotrudnik | Табельный номер сотрудника | | INT | | 4 | | PK | |
| StatusSotrud | Статус сотрудника | | CHAR(10) | | 1 | |  | |
| StatusDay | Статус Дня | | CHAR(10) | | 1 | |  | |
| ZP(Заработная Плата) | | | | | | | | |
| DATE | Дата выплаты | | DATE | | 3 | | PK | |
| idSotrudnik | Табельный номер сотрудника | | INT | | 4 | | PK | |
| Nadbav | Сумма всех надбавок | | INT | | 4 | |  | |
| itogZP | Итоговая ЗП сотруднка | | INT | | 4 | |  | |

Продолжение таблицы 1.6.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Base(Предельная база для исчисления страховых взносов) | | | | |
| Y | год | YEAR | 1 | PK |
| Z | Значение базы | INT | 4 |  |
| H(Справочник) | | | | |
| NDFL | НДФЛ | DECIMIAL(5,2) |  |  |
| PFR | ПФР | DECIMIAL(5,2) |  |  |
| FCC | ФСС | DECIMIAL(5,2) |  |  |
| FOMC | ФОМС | INT | 4 |  |
| kid1 | Размер льготы за 1 и 2 ребенка | INT | 4 |  |
| kid3 | Размер льготы за 3 и последующих детей | INT | 4 |  |
| invalid | Размер льготы за ребенка-инвалида | INT | 4 |  |
| invalid\_O | Размер льготы за опеку над ребенком инвалидом | INT | 4 |  |
| MROT | МРОТ | INT | 4 |  |

* 1. Контрольный пример

[контрольный пример.xlsx](контрольный%20пример.xlsx)

Контрольный пример является ручным подсчётом задачи. По составленной программе обрабатываются исходные данные контрольного примера. Полученные результаты сравниваются с известными результатами контрольного примера. При несовпадении результатов производится поиск, исправление ошибок, и снова производится выполнение программы.

Входная информация контрольных примеров представлена в приложении Б.

Выходные данные для контрольных примеров показаны в приложении В.

* 1. Общие требования к программному продукту

Пользователи должны иметь базовые навыки пользования персональным компьютером и знать основы бухгалтерского учета.

Минимальные требования к техническому обеспечению программного продукта следующие:

* Windows 10 64bit;
* Процессор 2,3 ГГц (2 ядра, 4 потока) / Intel core i3-7020U;
* Интегрированное графическое ядро Intel HD Graphics 620 или аналогичная дискретная видеокарта;
* Оперативная память 1 ГБ;
* 50 Мб свободного места на Твердотельном накопителе или Жестком диске;
* .Net Core;
* .NET Framework.

Функциональные возможности программного продукта:

* приложение должно формировать и отображать выходные данные пользователю;
* в приложении должен быть обеспечен просмотр таблиц (справочников) базы данных с возможностью добавления, редактирования, удаления данных.

Требования к надежности:

* пользователь для входа в свою учетную запись должен использовать логин и пароль;
* приложение должно обрабатывать ошибочные действия пользователя и сообщать ему об этом;
* приложение должно обеспечивать контроль входной и выходной информации.

Требования к информационной и программной совместимости: обеспечить работу приложения с таблицами СУБД MS SQL

1. Экспериментальный раздел

2.1 Описание программы

Программа имеет модульную структуру. При ее запуске выполняется проект на BDZarplata.exe. Схема взаимодействия модулей программы представлена на рисунке 2.1.1. Описание модулей и методов представлено в таблице 2.1.1.



* + - 1. Схема взаимодействия модулей

Таблица 2.1.1. – Описание Модулей

|  |  |
| --- | --- |
| Методы | Назначения |
| 1 | 2 |
| DB.cs | |
| void loadDataGrid | Загрузка данных из БД в DataGrid |
| void LoadDataComboBox | Загрузка данных из БД в список |
| void LoadDataListBox | Загрузка данных из БД в список |
| object queryScalar | Выполнение SQL Запроса к БД |
| string queryScalar | Выполнение запроса к БД |
| string[] queryScalar | Выполнение запроса к БД |
| void ReturnTable | Запрос к БД с получением данных из таблицы (таблиц) |
| int queryData | Выполнение внешнего SQl Файла или запроса с возвратом количества строк |
| DB\_Connect.cs | |
| void OpenConnection | Создание соединения с БД |
| void CloseConnection | закрытие соединения с БД |
| bool OpenClouseConnection | Проверка на возможность установки соединения с БД |

Продолжение Таблицы 2.1.1.

|  |  |
| --- | --- |
| 1 | 2 |
| DG.cs | |
| string GetSelectCell | Получение значения выбранной ячейки DataGrid |
| Manager.cs | |
| void UpdateLabel | Обновление текста в Label |
| Procedure.cs | |
| void UpdateTable | Обновление значений указанной таблицы в БД |
| void InsertTable | Добавление значений указанной таблицы в БД |
| MainWindow.xaml.cs | |
| public MainWindow | Переключение на страницу входа |
| void BtnBack\_Click | Возврат на предыдущую страницу |
| Page\_AddRedAnketa.xaml.cs | |
| public Page\_AddRedAnketa | Загрузка формы |
| void CB\_Doljnost\_SelectionChanged | соединение 2 списков для синхронного выбора |
| void intOnly\_PreviewTextInput | Фильтр целочисленных значений |
| void Btn\_Save\_Click | Проверка и внесение данных в таблицу |
| void Btn\_Delete\_Click | Удаление записи |
| Page\_BughalterInfo.xaml.cs | |
| public Page\_BughalterInfo() | Загрузка формы |
| void TabI\_LN\_Initialized | Подгруздка справочника льгот и налогов |

Продолжение Таблицы 2.1.1.

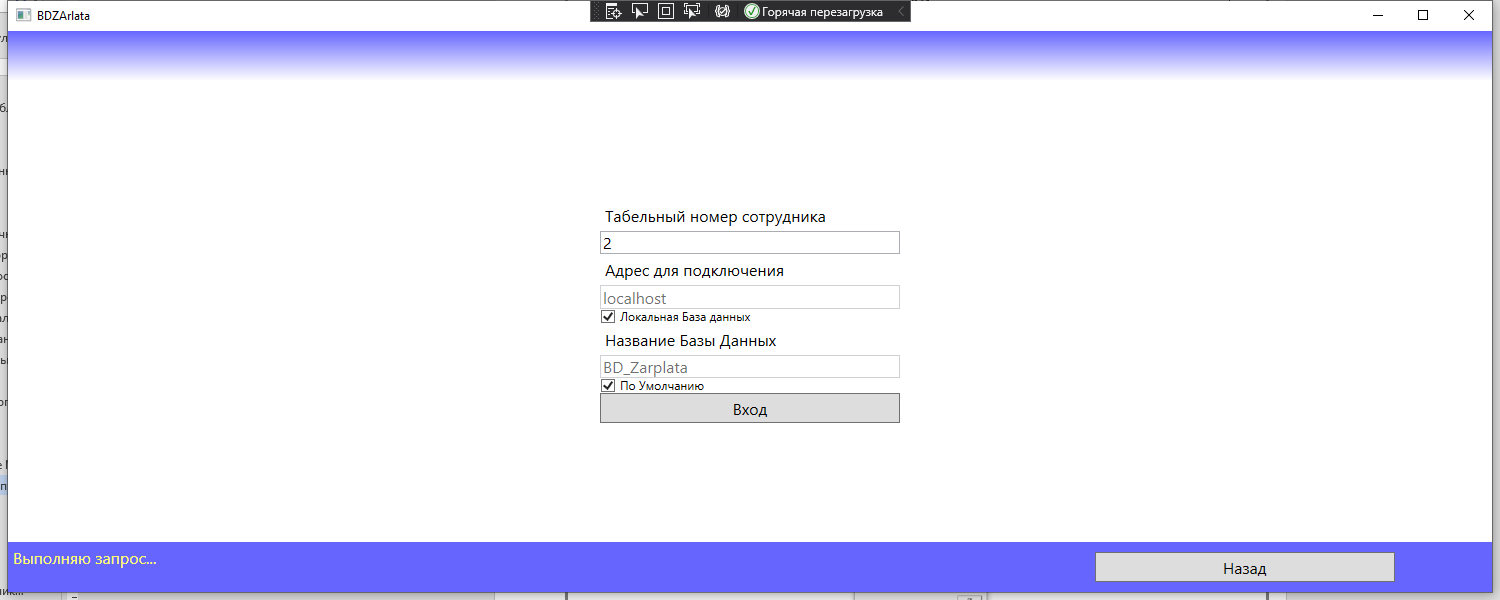
|  |  |
| --- | --- |
| 1 | 2 |
| void Btn\_Save\_Click | сохранение измененных полей таблицы БД |
| void Btn\_Redactir\_Click | Переход на страницу редактирования |
| void intOnly\_PreviewTextInput | Фильтр целочисленных значений |
| void floatOnly\_PreviewTextInput | Фильтр дробных значений |
| void DG\_SotridnikOklad\_SelectionChanged | запись ID выделенной строки |
| void DG\_SotridnikOklad\_MouseDoubleClick | Переход в режим редактирования оклада |
| void TabI\_OkladSotrud\_GotFocus | Смена видимости кнопок , при выборе вкладки |
| void TabI\_Base\_GotFocus |
| void TabI\_LN\_GotFocus |
| void LB\_Sotrud\_FIO2\_SelectionChanged | Подгрузка данных о надбавках и штрафах выбранного сотрудника в DataGrid |
| void LB\_Sotrud\_id2\_SelectionChanged |
| void DG\_NadbavShtraf\_SelectionChanged | Вывод в форму даты , надбавки и штрафов |
| void Btn\_Raschet\_Click | Переход к странице расчета |
| void Btn\_Red\_Nadbav\_Click | Внесение изменений в БД |
| Page\_Bughl\_AddEdit\_base.xaml.cs | |
| void BtnSave\_Click | Сохранение данных |
| void intOnly\_PreviewTextInput | Фильтр целочисленных значений |
| Page\_Login.xaml.cs | |
| void BtnLogin\_Click | Подключение к БД с указынными параметрами |

Продолжение Таблицы 2.1.1.

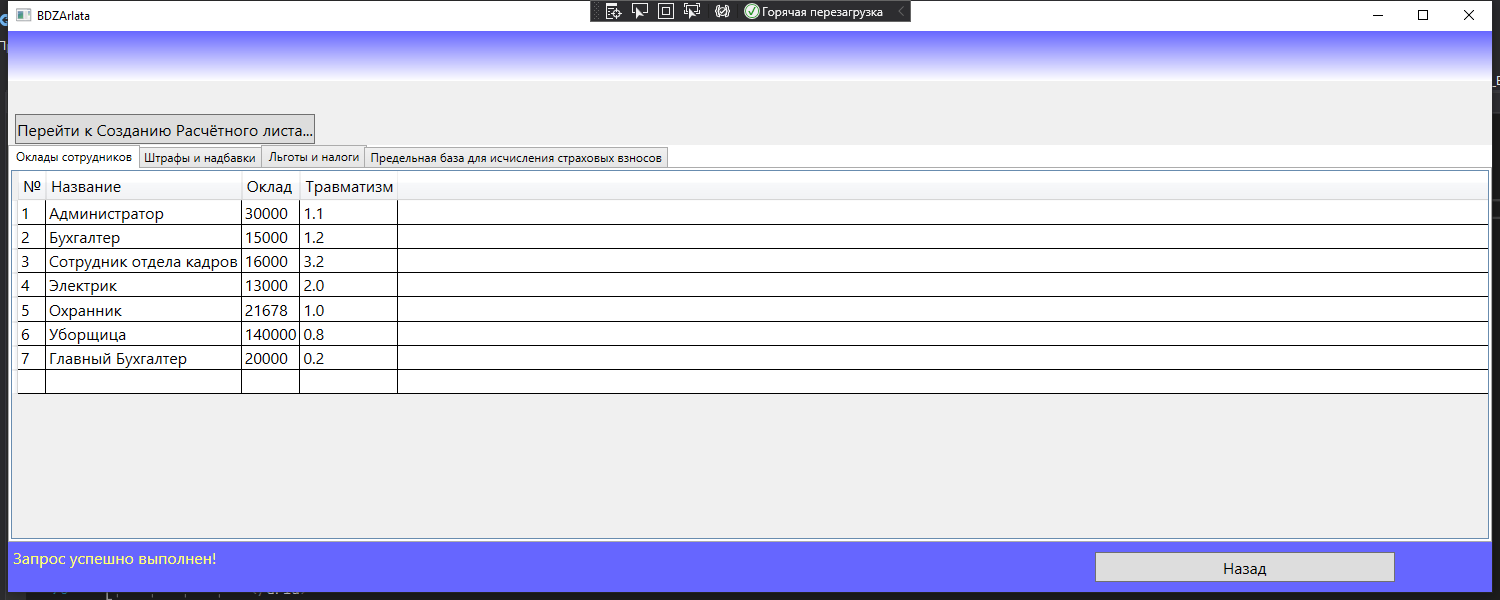
|  |  |
| --- | --- |
| 1 | 2 |
| void CB\_IPPC\_Localhost\_Click | Переключение поля для ввода адреса БД |
| void CB\_BD\_NAmeDef\_Click | Переключение поля для ввода названия БД |
| void BTN\_Raschet\_Click | Расчет зарплаты |
| void BTN\_Export\_Click | Экспорт в Эксель |
| void CB\_SotrudID\_SelectionChanged | Сопоставление ФИО и ID сотрудника |
| void CB\_FIO\_SelectionChanged | Подгрузка информации о сотруднике |
| void CB\_Date\_SelectionChanged | Изменение даты на календаре |
| Page\_SotrudnikMainInfo.xaml.cs | |
| void Btn\_Redactir\_Click | Переход в режим редактирования |
| void Btn\_Save\_Click | сохранение измененных полей таблицы БД |
| void LB\_Sotrud\_FIO\_SelectionChanged | Сопоставление ФИО и ID сотрудника |
| void LB\_Sotrud\_id\_SelectionChanged | Подгрузка данных о Расписании выбранного сотрудника |
| void DG\_Sotrud\_Anketa\_MouseDoubleClick | Переход в режим редактирования |
| void Btn\_Add\_Click | Переход в режим добавления нового сотрудника |
| void DG\_Raspisnie\_SelectionChanged | Подгрузка данных о выбранном дне |
| void BTN\_RedRaspisan\_Click | сохранение измененных полей таблицы БД |

* 1. Протокол тестирования программного продукта

Тестирование входа Бухгалтера, при вводе корректных данных (рисунки 2.2.1–2.2.2)

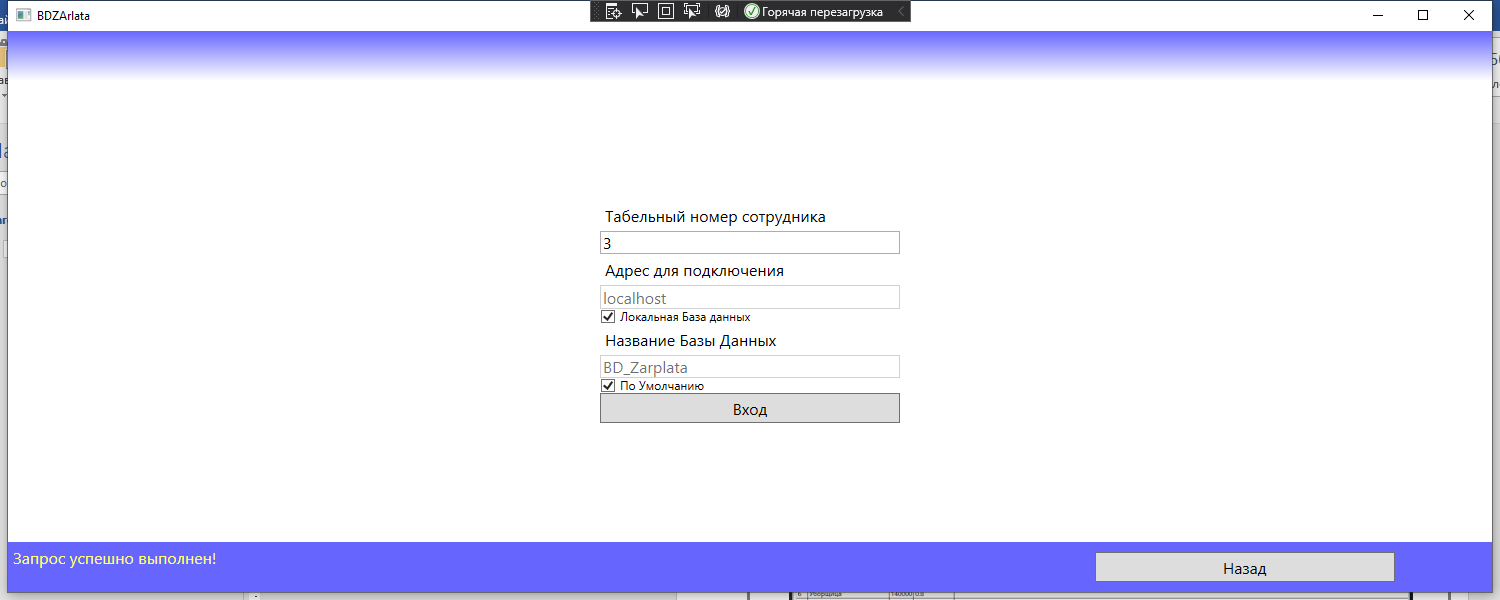


* + - 1. Ввод корректных данных бухгалтера

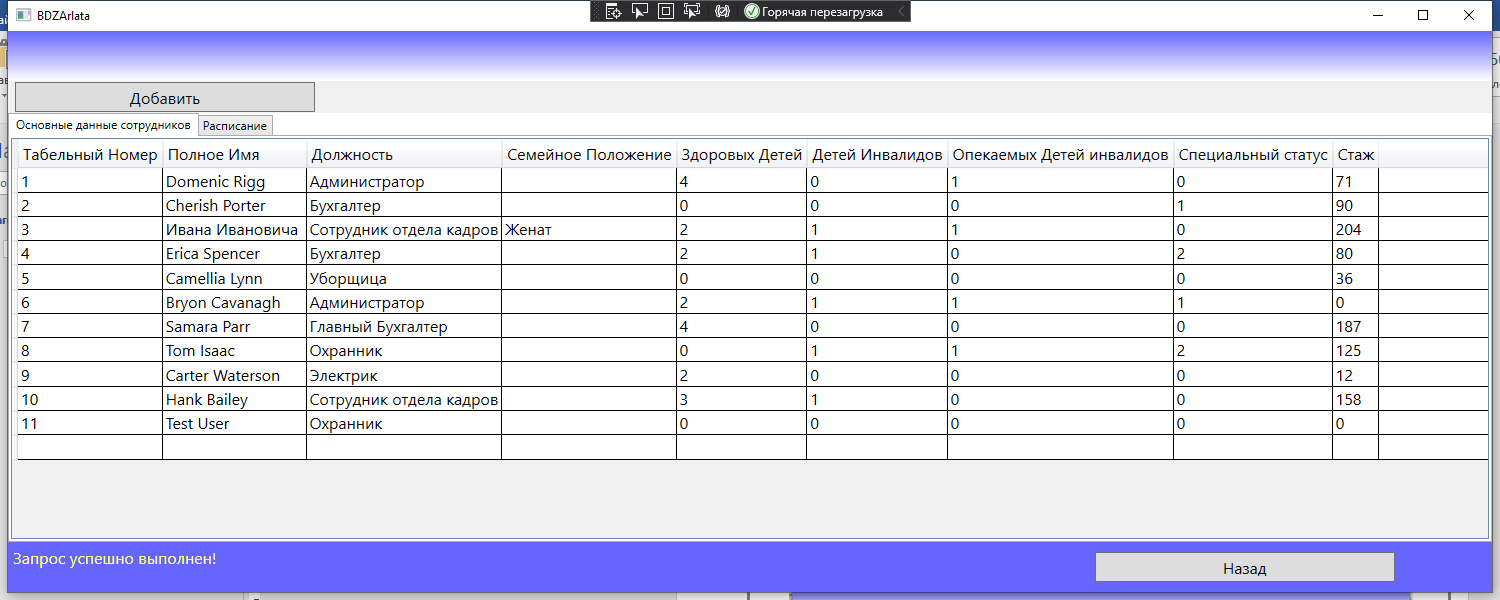


* + - 1. Страница бухгалтера

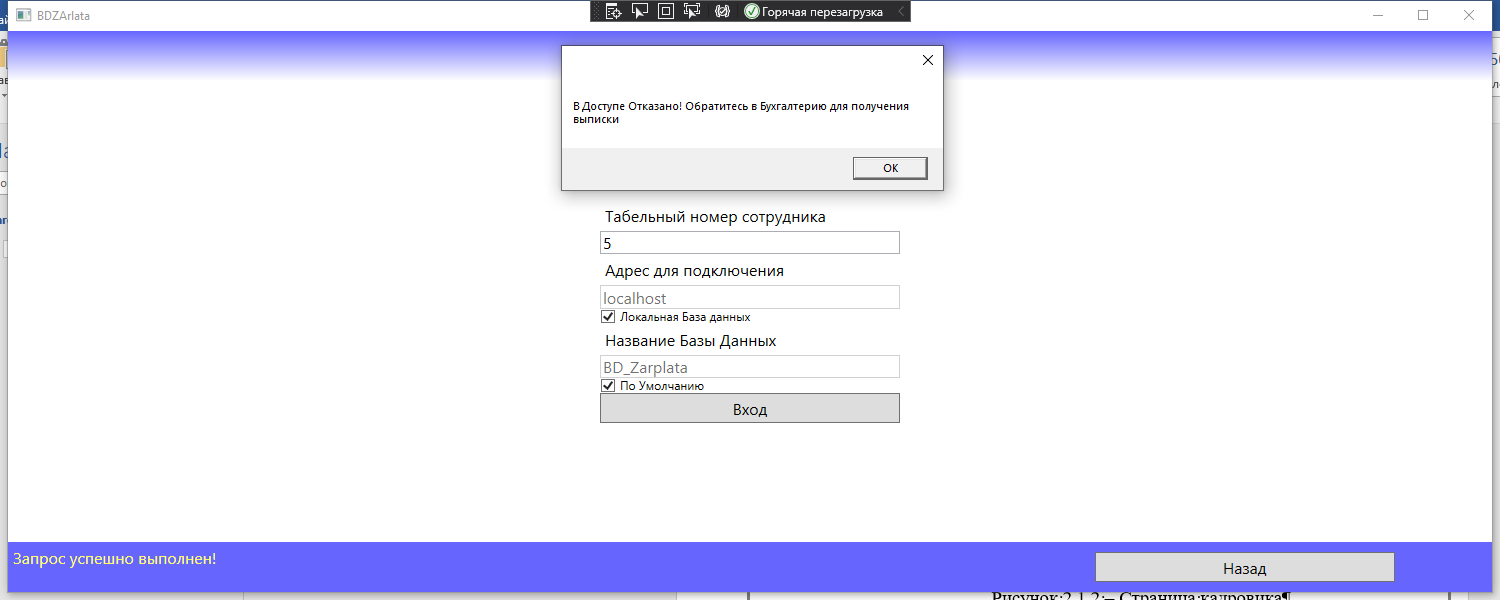
Тестирование входа кадровика при вводе корректных данных (рисунки 2.2.3–2.2.4)



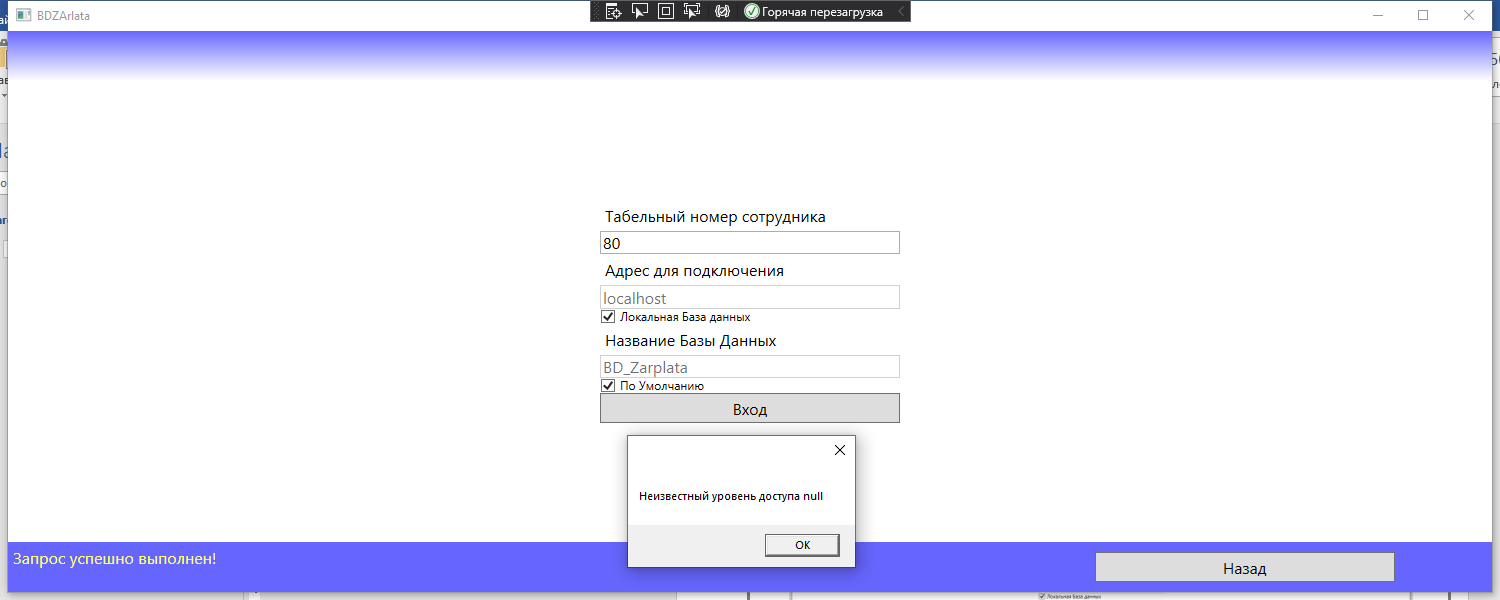
* + - 1. Ввод корректных данных кадровика



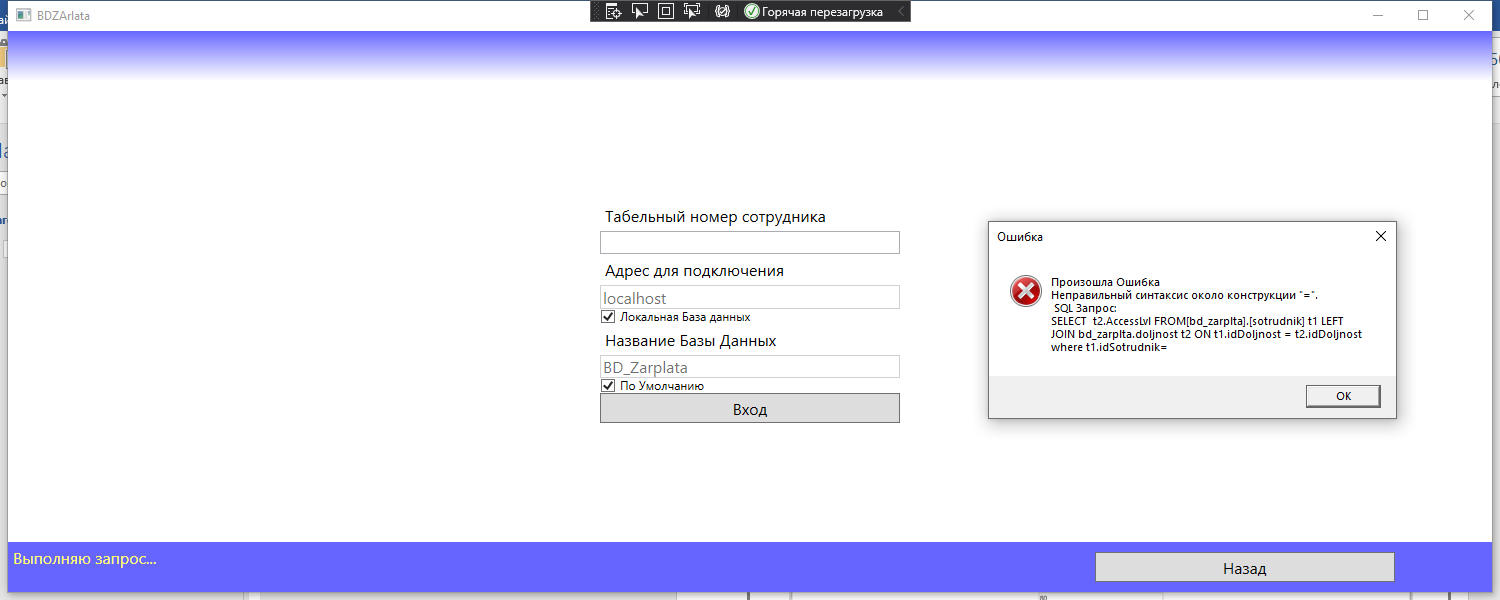
* + - 1. Страница кадровика

Тестирование входа обычного пользователя при вводе корректных данных (рисунок 2.2.5)

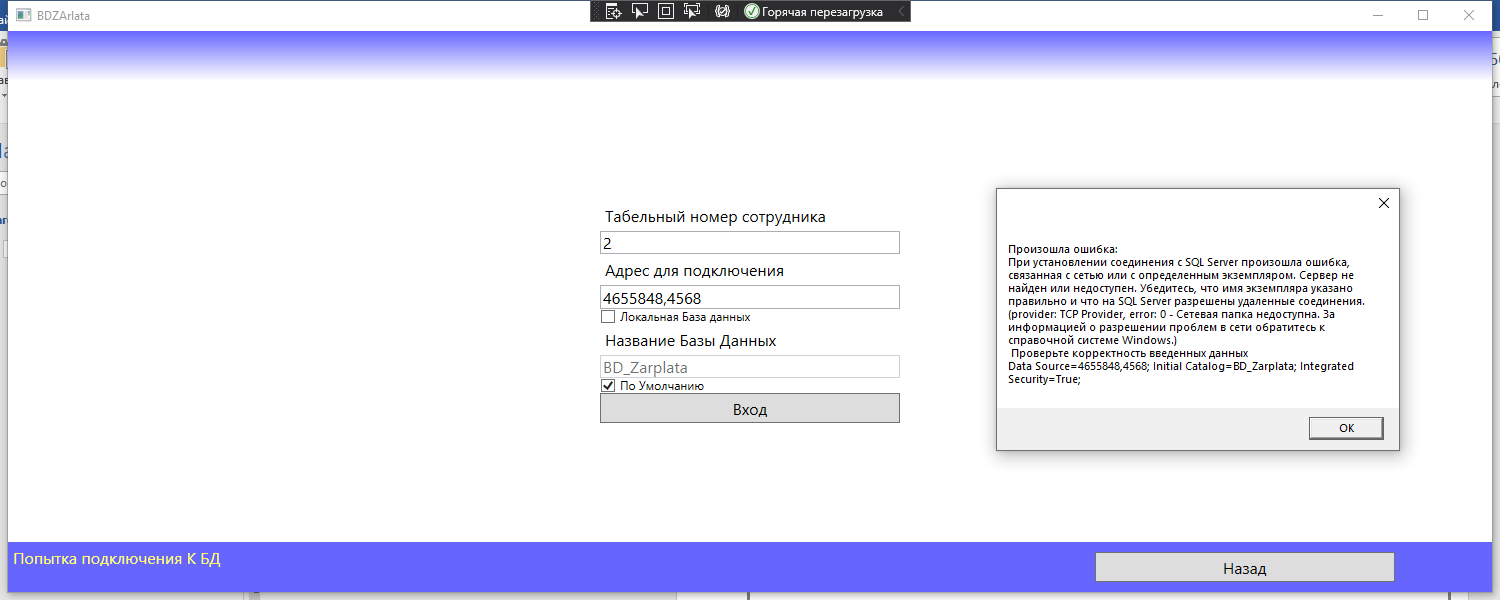
* + - 1. Сообщение об отказанном доступе

Тестирование входа при вводе некорректных данных (рисунки 2.2.6–2.2.10)

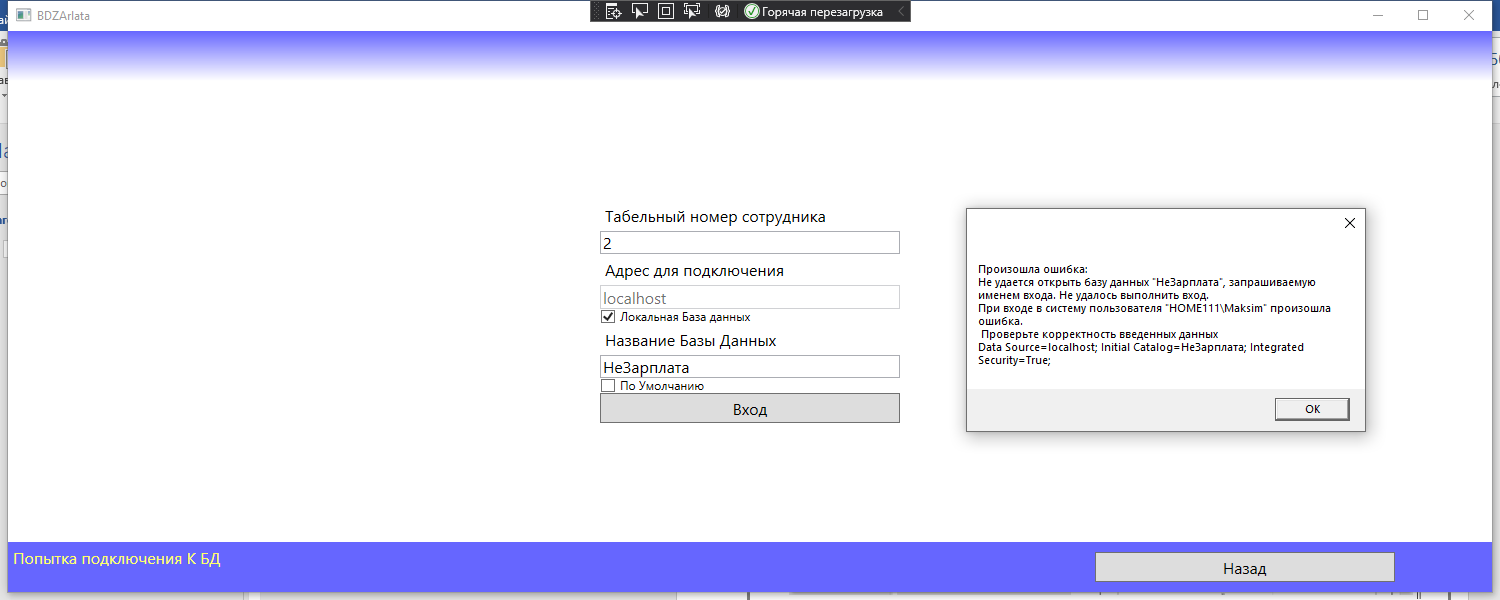
* + - 1. Сообщение об ошибке



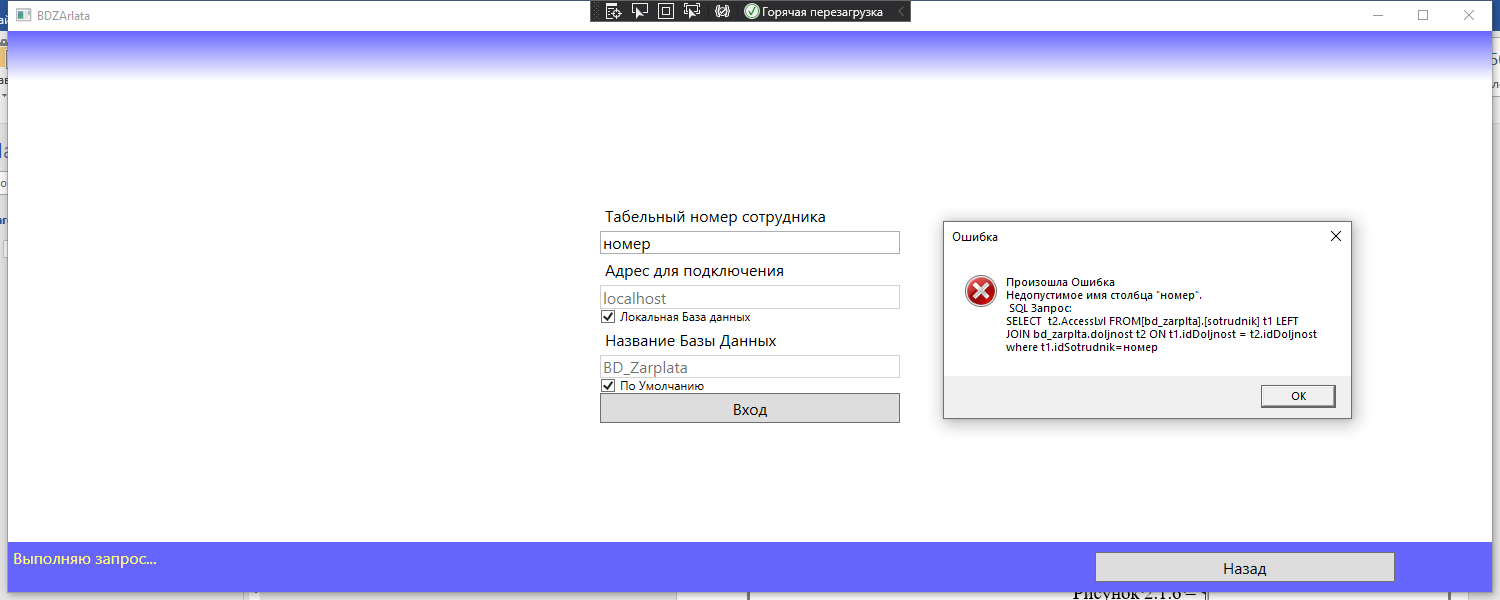
* + - 1. Сообщение об ошибке



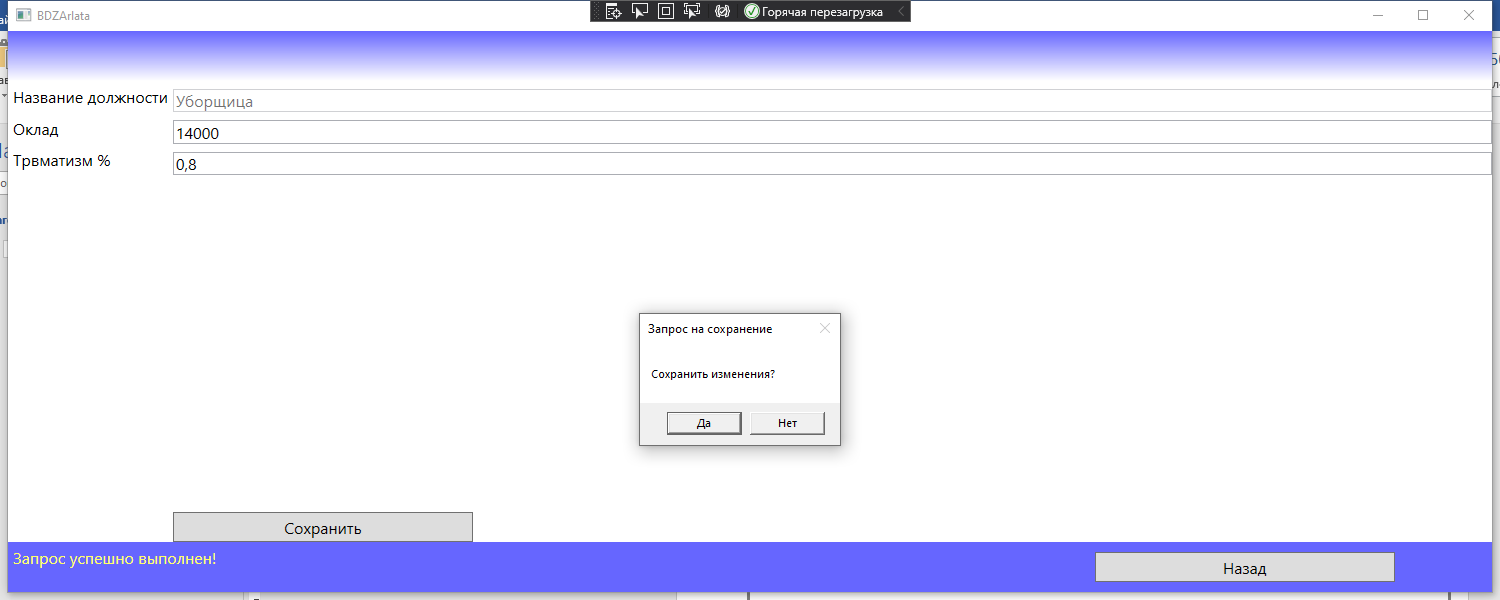
* + - 1. Сообщение об ошибке



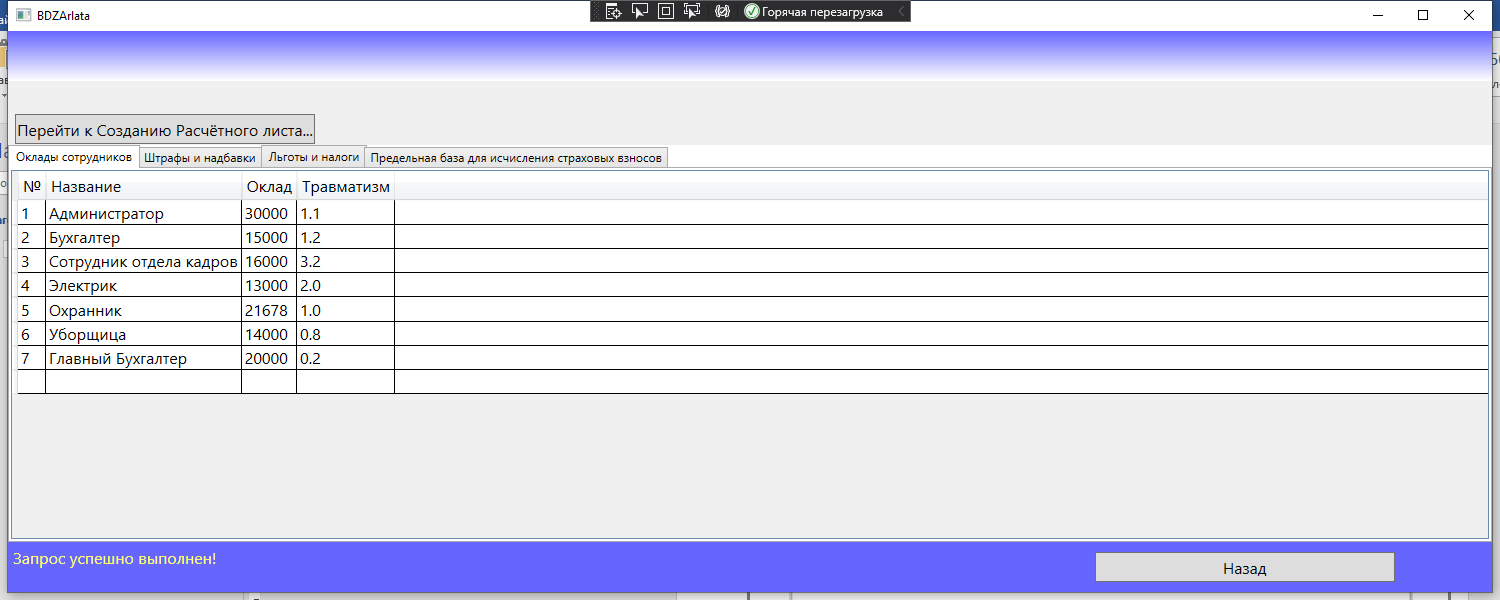
* + - 1. Сообщение об ошибке



* + - 1. Сообщение об ошибке

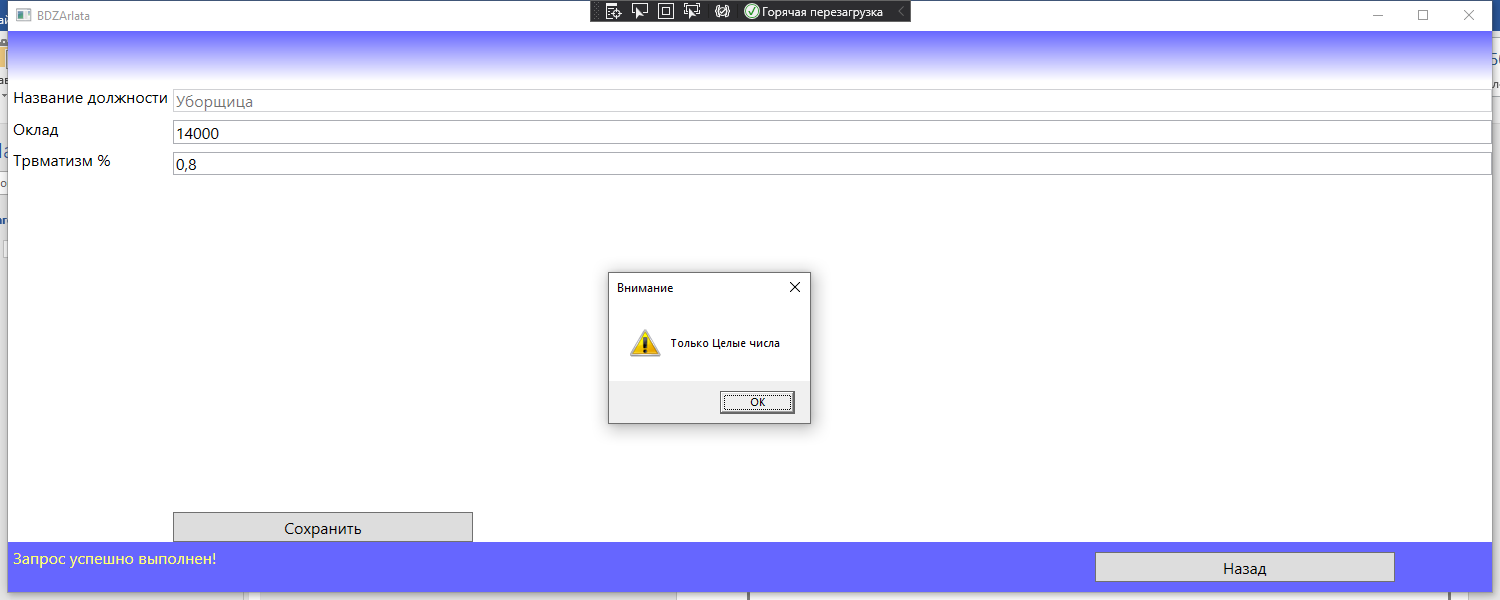
Тестирование изменение оклада при вводе корректных данных (рисунки 2.2.11–2.2.12)

* + - 1. Запрос на сохранение



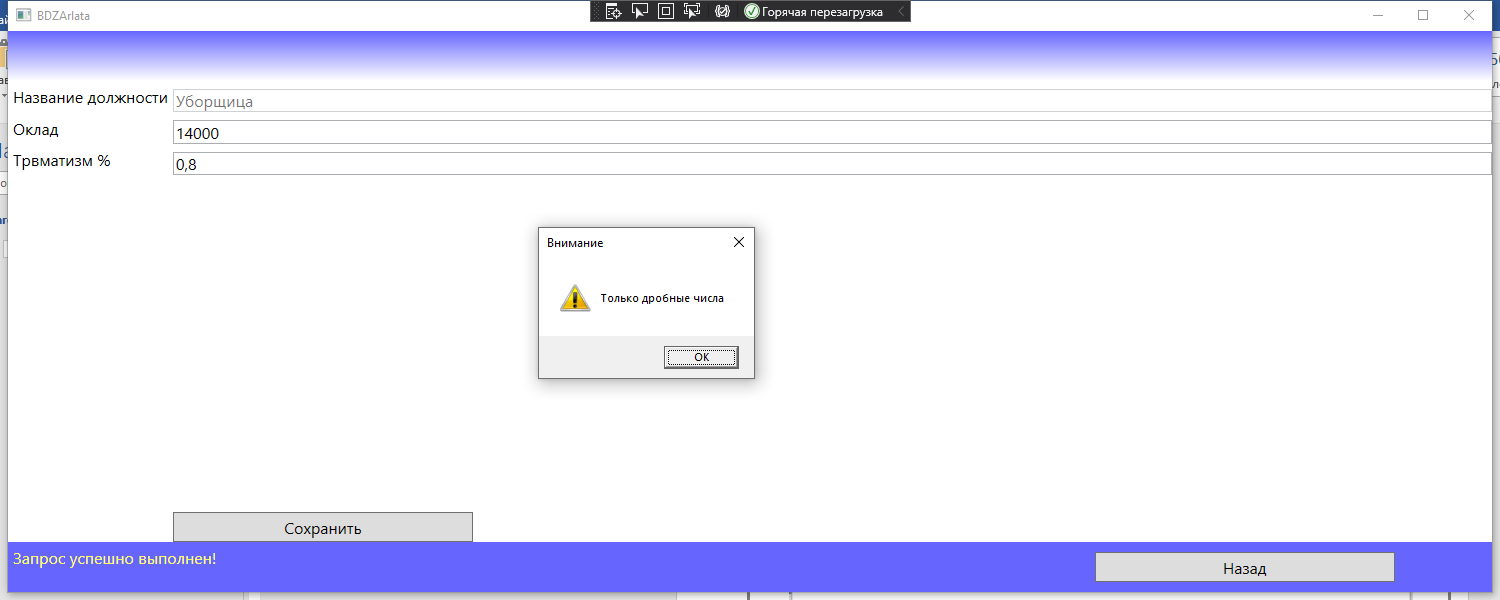
* + - 1. Изменение данных

Тестирование изменения оклада при вводе не корректных данных в поле «Оклад» (рисунок 2.2.13)



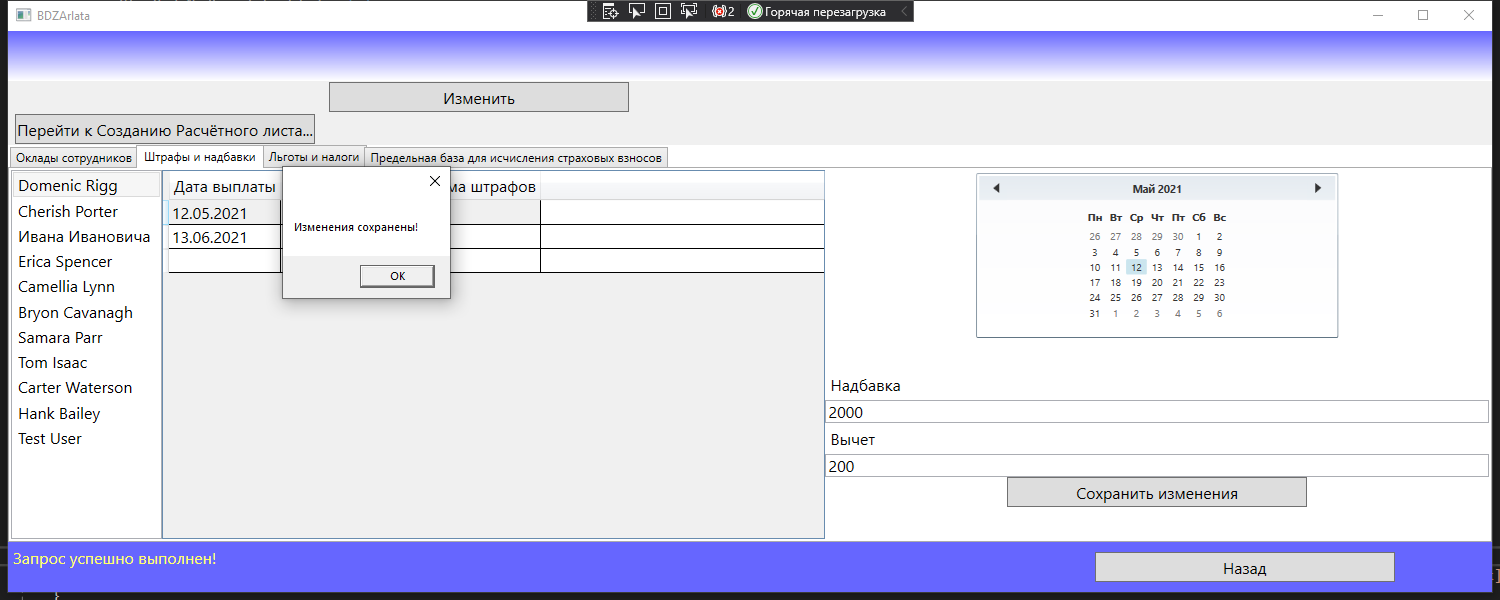
* + - 1. Сообщение об ошибке

Тестирование изменения оклада при вводе не корректных данных в поле «Травматизм%» (рисунок 2.2.14)

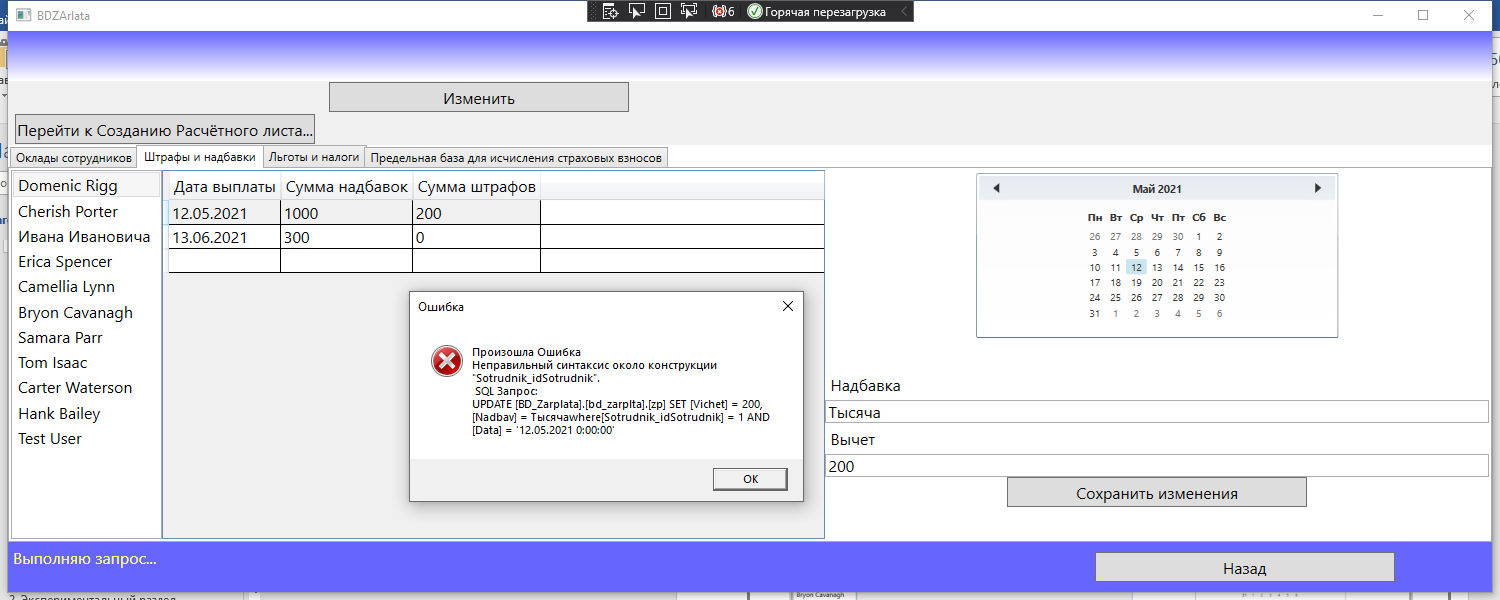


* + - 1. Сообщение об ошибке

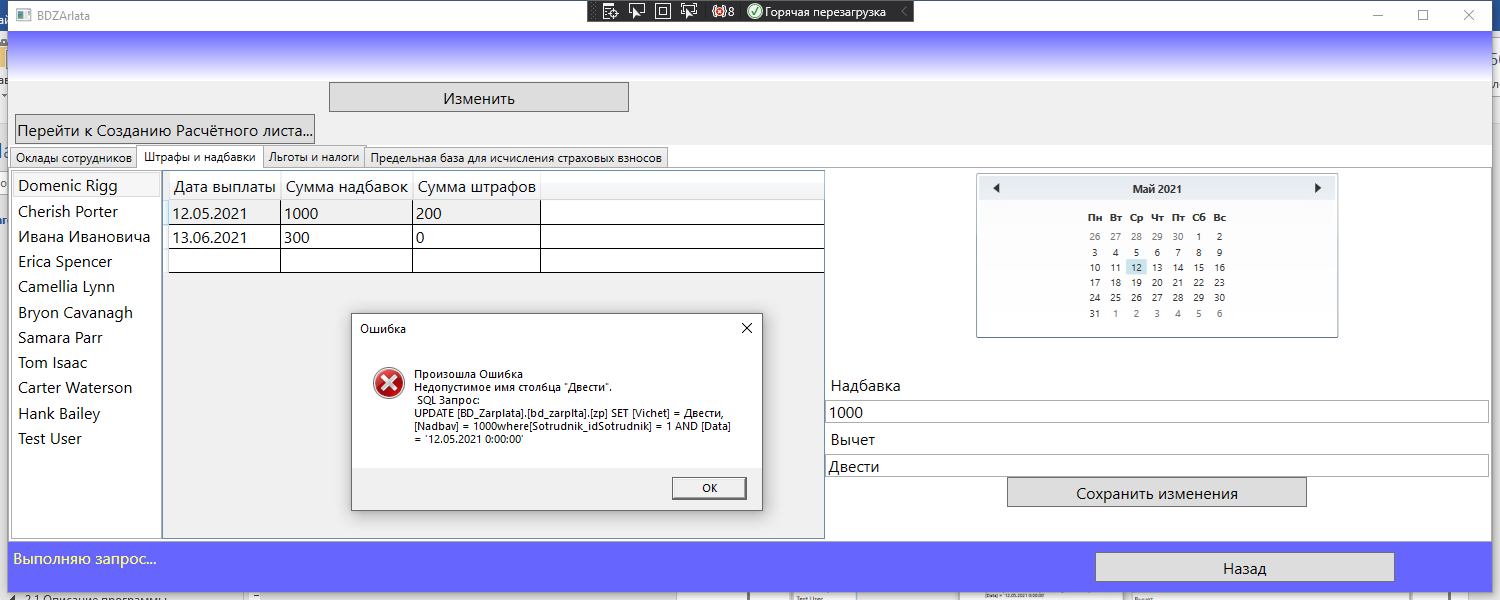
Тестирование штрафов и надбавок при вводе корректных данных (рисунок 2.2.15)



* + - 1. Сообщение об успешном сохранении

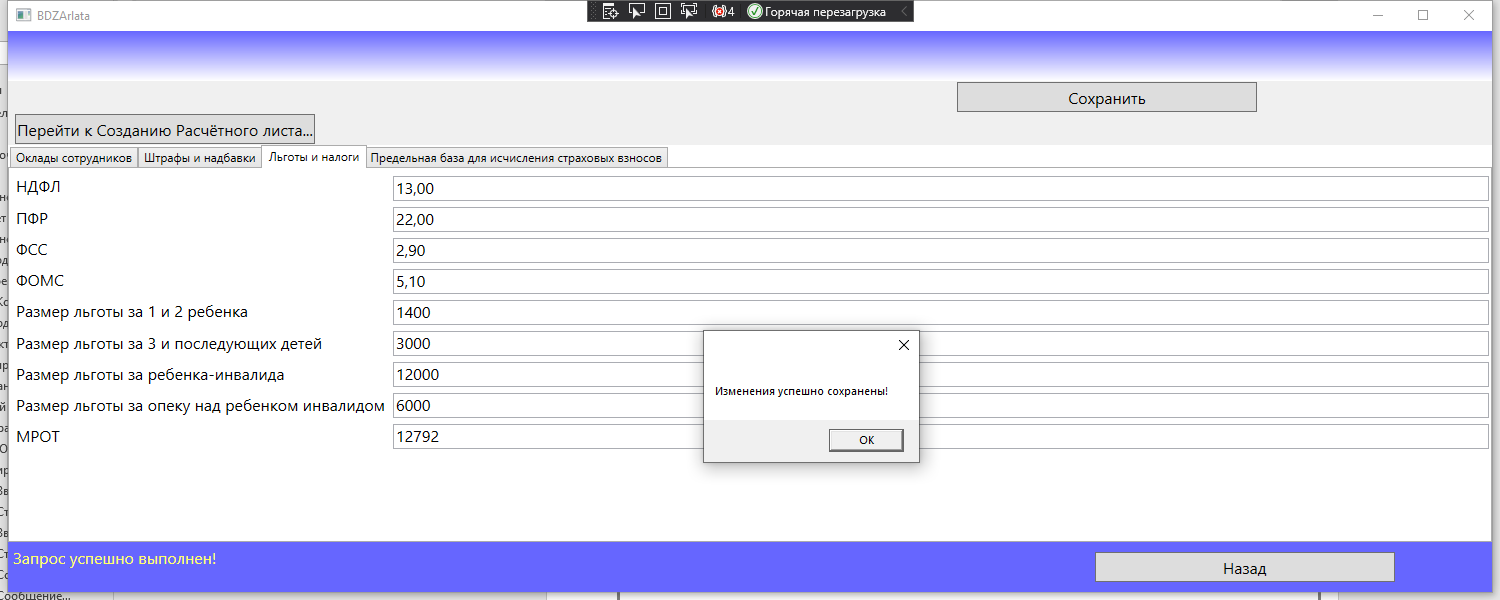
Тестирование штрафов и надбавок при вводе не корректных данных (рисунки 2.2.16–2.2.17)

* + - 1. Сообщение об ошибке



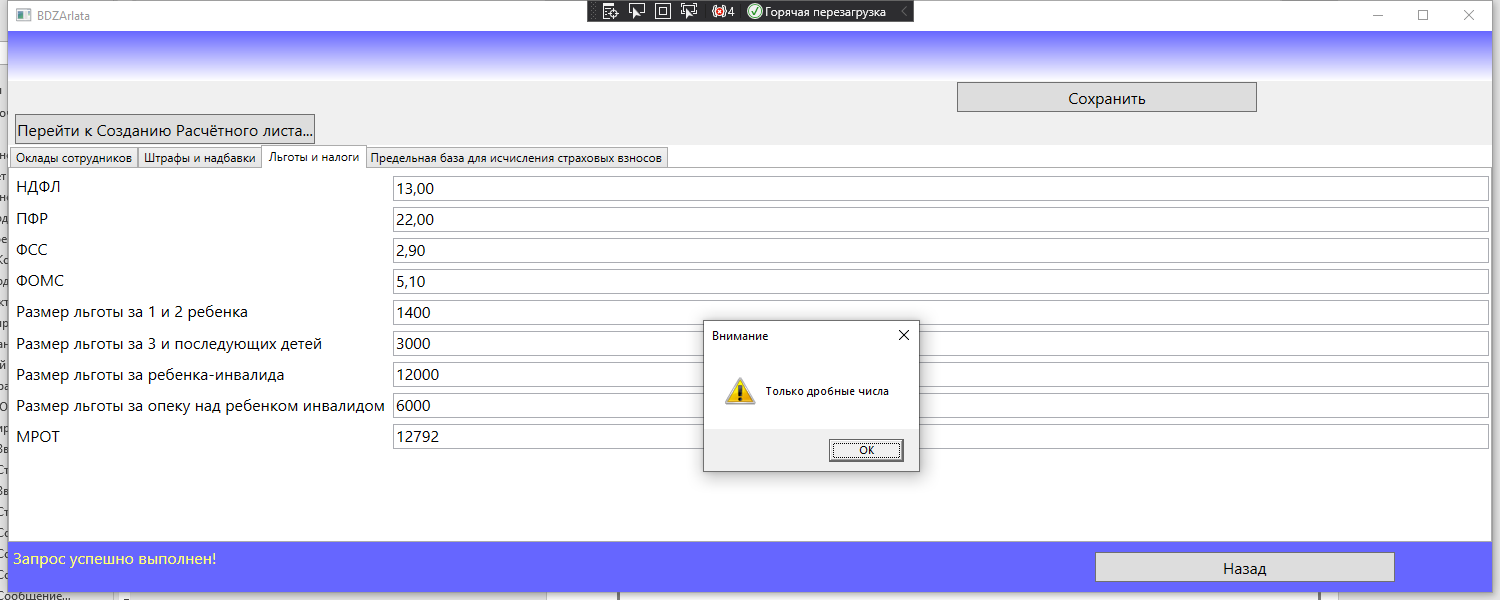
* + - 1. Сообщение об ошибке

Тестирование льгот и налогов при вводе корректных данных (рисунок 2.2.18)



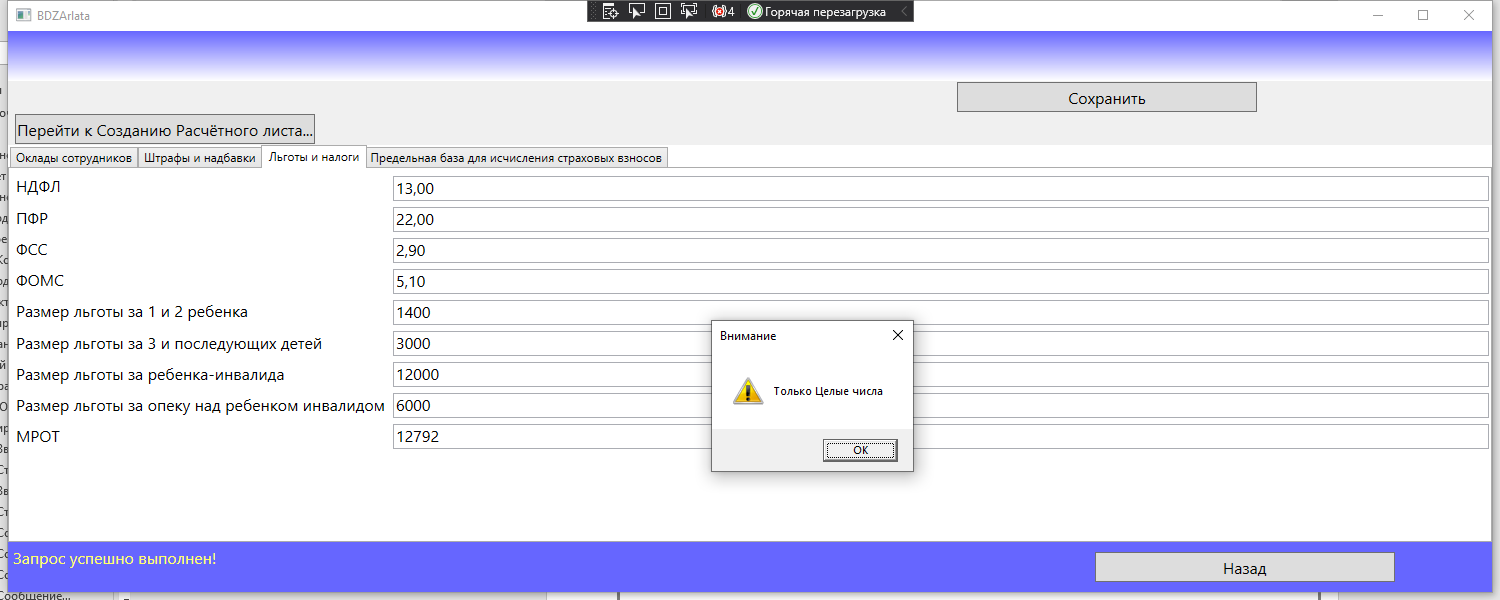
* + - 1. Сообщение об успешном сохранении

Тестирование льгот и налогов при вводе некорректных данных в поля: «НДФЛ», «ПФР», «ФСС» , «ФОМС» (рисунок 2.2.19)



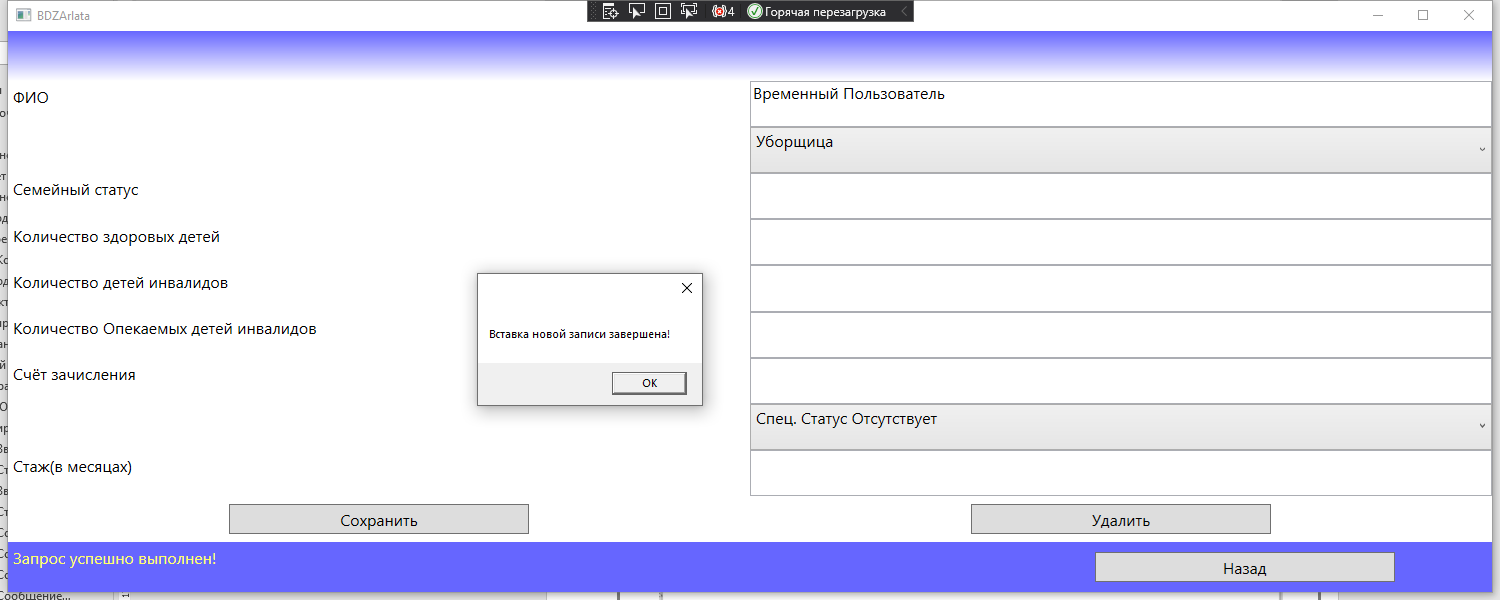
* + - 1. Сообщение об ошибке

Тестирование льгот и налогов при вводе некорректных данных в поля льгот и «МРОТ» (рисунок 2.2.20)



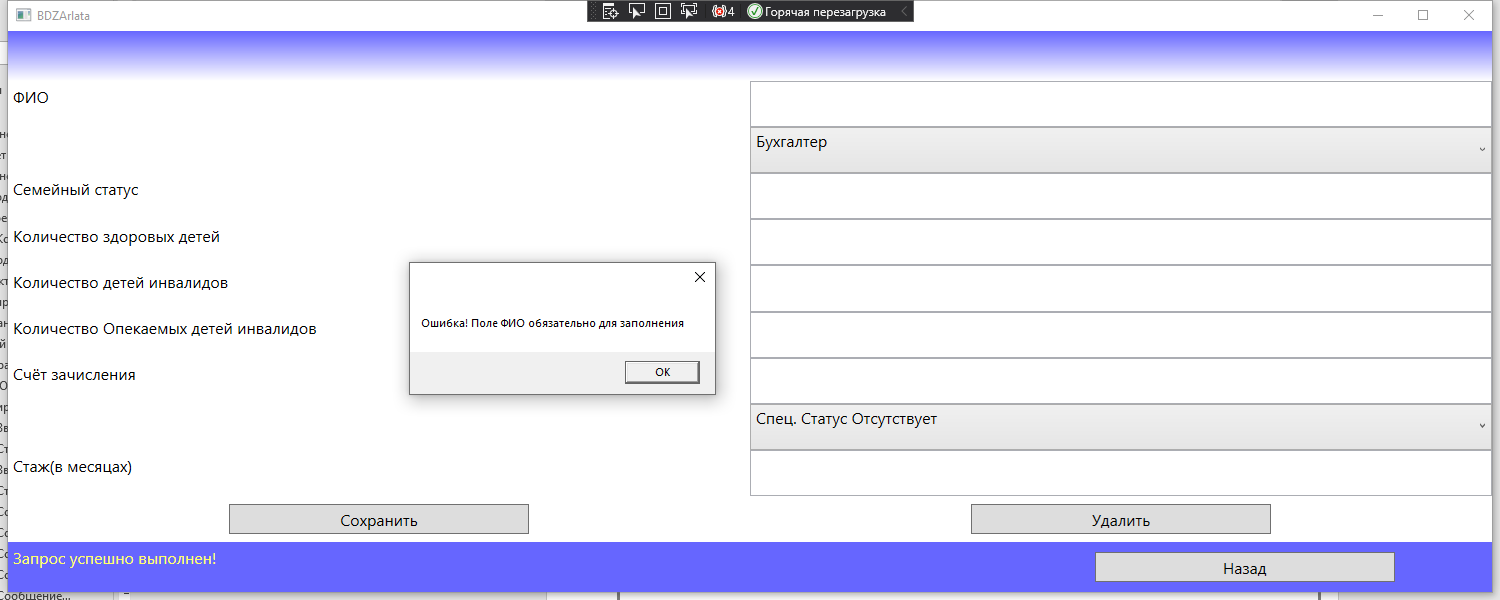
* + - 1. Сообщение об ошибке

Тестирование добавления нового пользователя при заполнении только обязательных полей корректно (рисунок 2.2.21)



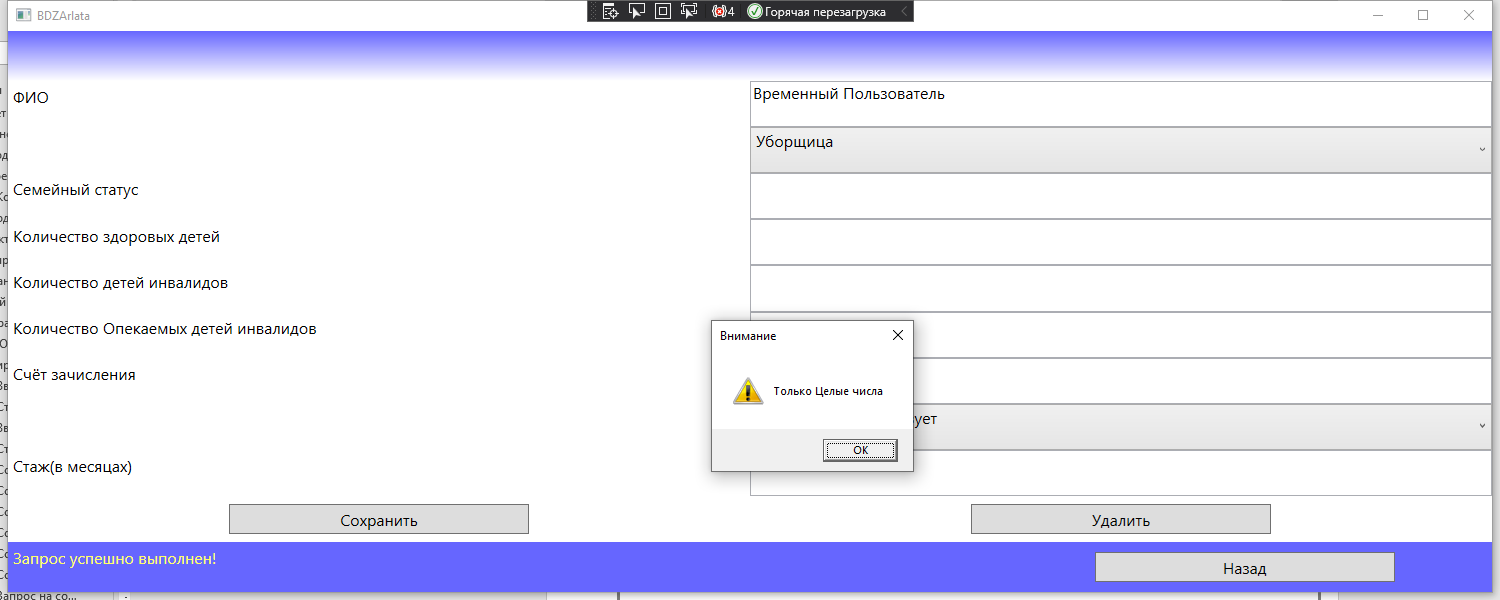
* + - 1. Успешное создание нового пользователя

Тестирование добавления или изменения нового пользователя при некорректном заполнении поля «ФИО» (рисунок 2.2.22)



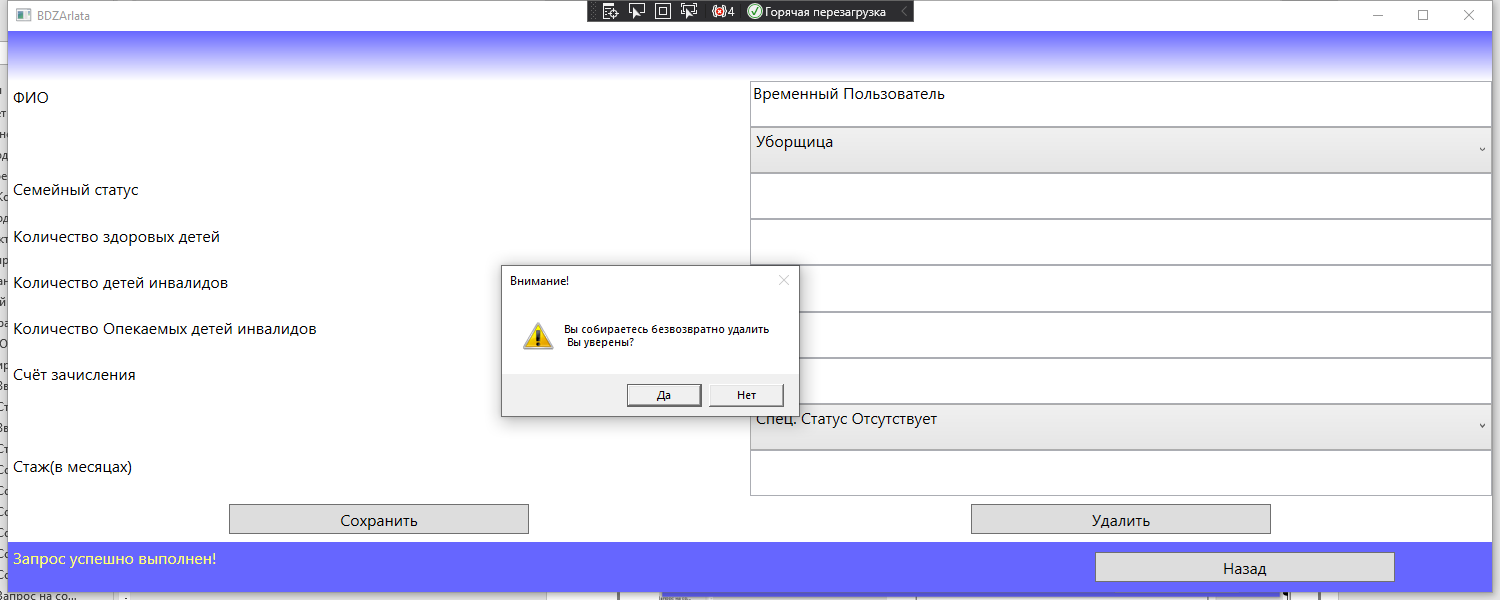
* + - 1. Сообщение об ошибке

Тестирование добавления или изменения нового пользователя при некорректном заполнении числовых полей (рисунок 2.2.23)

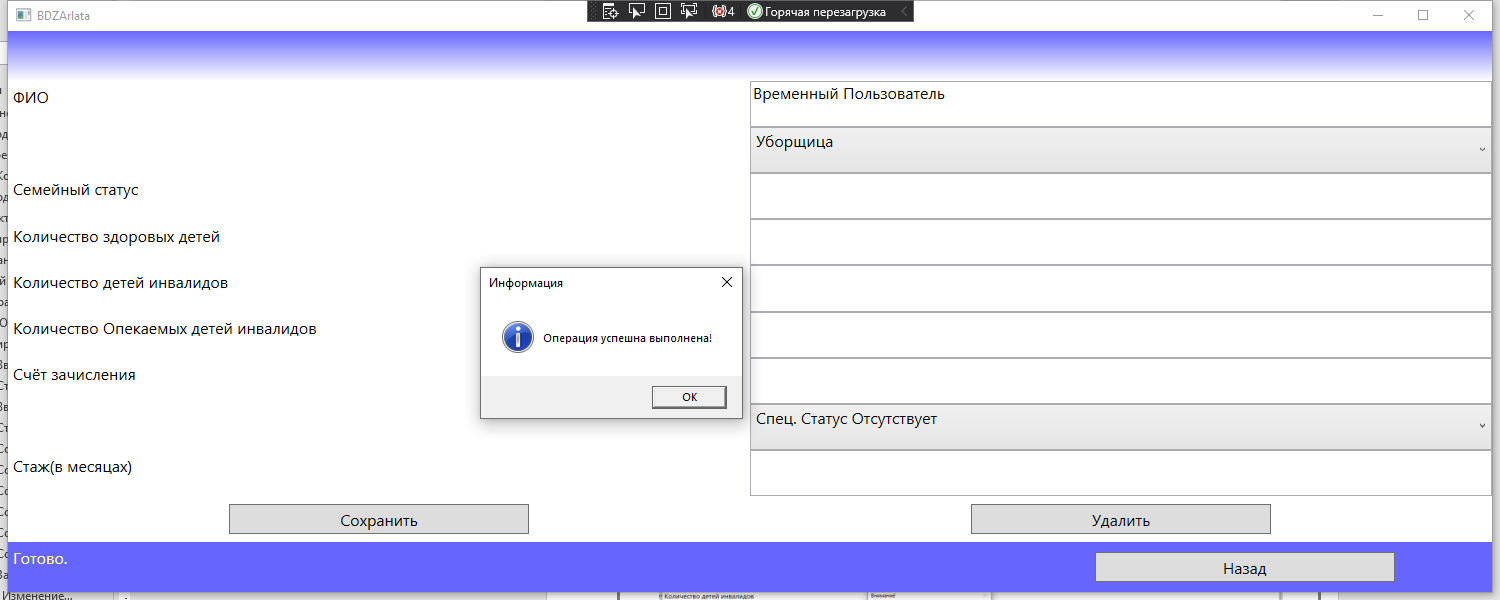


* + - 1. Сообщение об ошибке

Тестирование удаления пользователя (рисунки 2.2.24–2.2.25)

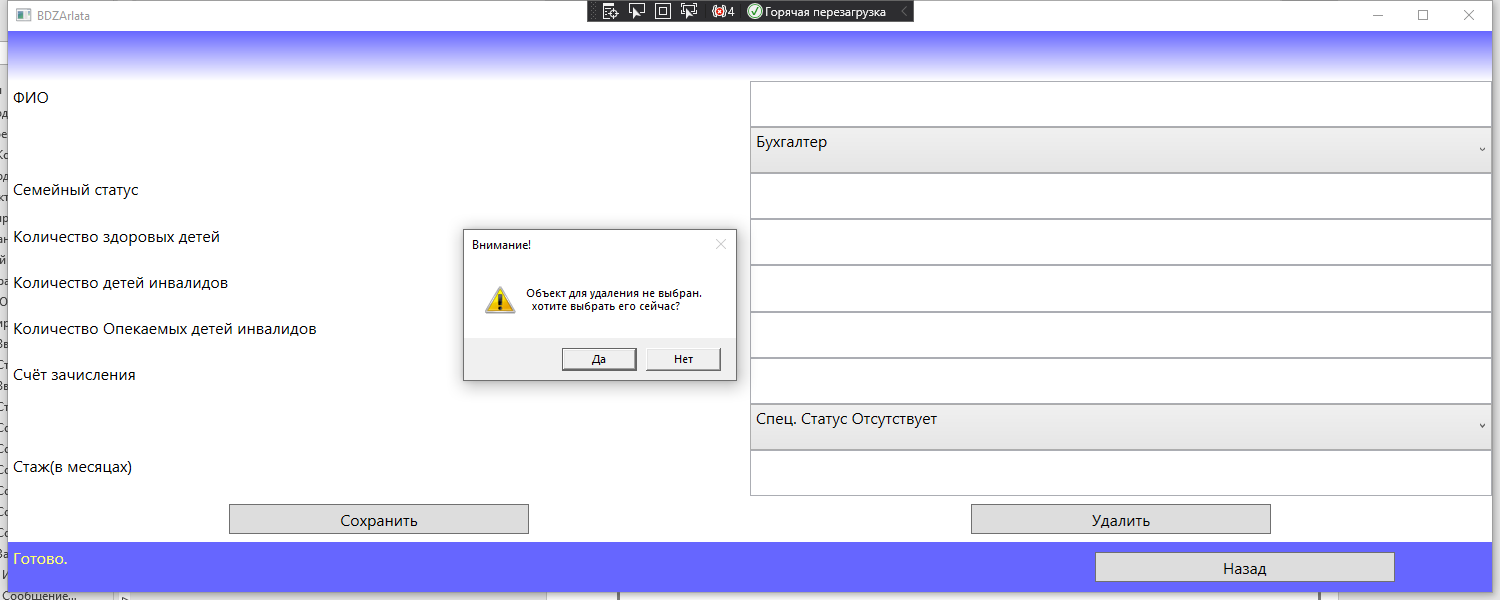


* + - 1. Диалоговое окно



* + - 1. Сообщение об успешном удалении

Тестирование удаления ещё не созданного пользователя (рисунок 2.2.26)



* + - 1. Сообщение о выборе не существующего объекта
  1. Руководство пользователя

Назначение системы

Программа BDZarplata предназначена автоматизировать расчёт зарплаты, содержание работника и суммы налогов, упростить процесс выплаты зарплаты для упрощения работы Бухгалтера.

Условия применения системы

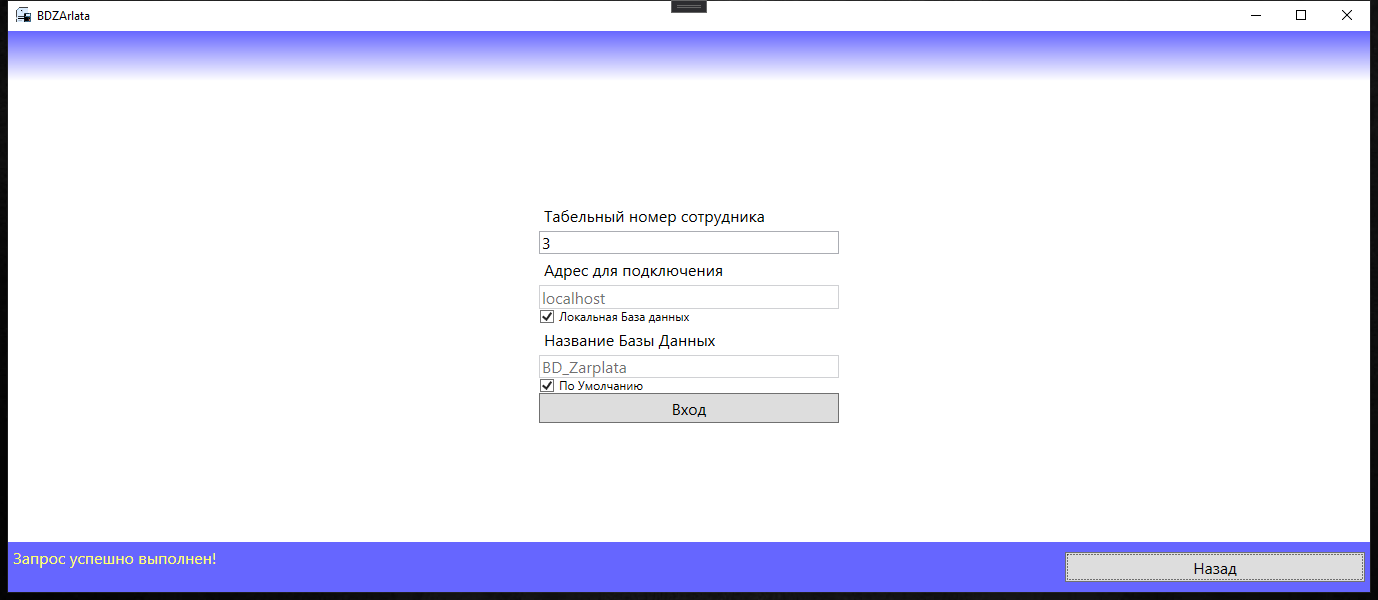
Программное обеспечение разрабатывается для персональной вычислительной техники со следующими характеристиками:

* Windows 10 64bit
* Процессор 2,3 ГГц (2 ядра, 4 потока) / Intel core i3-7020U
* Интегрированное графическое ядро Intel HD Graphics 620 или аналогичная дискретная видеокарта
* Оперативная память 1 ГБ
* 50 Мб свободного места на Твердотельном накопителе или Жестком диске

Программа «BDZarplata» предназначена для пользователей, имеющих как минимум первоначальные навыки работы с графической операционной системой, на которой будет запускаться данная программа, а также владеющих знаниями по бух. учету.

Подготовка системы к работе

Для запуска программы необходимо запустить приложение BDZarplata.exe из каталога, в котором установлен данный программный продукт. После этого открывается окно авторизации для входа в программу.



* + - 1. Окно авторизации

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта были разработаны структура и алгоритм работы WPF-приложения «BDZarplata».

При этом были изучены особенности реализации компонентов WPF для построения клиентских приложений с визуально привлекательными возможностями взаимодействия с пользователем.

Результатом работы стало создание WPF-приложения для базового создания списка работников, их расписания, а также полного расчёта заработной платы.

WPF-приложение написано на языке C# с использованием среды разработки Visual Studio 2019 с использованием языка разметки XAML и системы управления базой данных Microsoft SQL Server

Были проведены опытная эксплуатация и отладочное тестирование WPF приложения. По результатам отладочного тестирования были устранены некоторые недостатки, в частности были обнаружены и исправлены неточности в реализации алгоритма: усовершенствован контроль на входные данные и отформатирован вывод документов.

С помощью приложения на основании данных контрольного примера были получены результаты, которые полностью совпадают с выходной информацией контрольного примера.

Приложение А.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Расчетный листок за 13.05.2021** | | | | | | | |
| **ООО** | | | | | | | |
| **Работник: Cherish Porter** | | | | |  |  |  |
| **Табельный номер: 2** |  |  |  |  | **Должность: Бухгалтер** | | |
| *Вид* | *Период* | *Дни* | *Часы* | *Сумма* | *Вид* | *Период* | *Сумма* |
| *1. Начислено* |  |  |  |  | *2.Удержано* |  |  |
| Оклад |  |  |  |  | НДФЛ по ставке 13% |  |  |
| Дополнительные выплаты |  |  |  |  | Иные удержания |  |  |
| Больничные пособия |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Всего начислено: |  |  |  |  | Всего удержано: |  |  |
| 3. Взносы в ПФР |  |  |  |  | Сумма к выплате |  |  |
| Страховые взносы в ПФР (страховая часть 22%) |  |  |  |  | Зачислено на счёт№ |  |  |
|  |  |  |  |  | Выдано наличными |  |  |
|  |  |  |  |  |  |  |  |

Рисунок А – Шаблон выходного документа

Приложение Б.

Входные данные контрольного примера

Таблица Б.1.1 - Справочник сотрудников

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер сотрудника | ФИО сотрудника | ID Должности | Семейный статус сотрудника | Количество здоровых детей сотрудника | Количество детей инвалидов | Число опекаемых детей инвалидов | Счёт зачисления сотрудника | Наличие или отсутствие одного из спец статусов | Страховой стаж |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | Domenic Rigg | 1 |  | 4 | 0 | 1 |  | 0 | 71 |
| 2 | Cherish Porter | 2 |  | 0 | 0 | 0 |  | 1 | 90 |
| 3 | Tess Morley | 3 |  | 2 | 1 | 1 |  | 0 | 204 |
| 4 | Erica Spencer | 2 |  | 2 | 1 | 0 |  | 2 | 80 |
| 5 | Camellia Lynn | 6 |  | 0 | 0 | 0 |  | 0 | 36 |
| 6 | Bryon Cavanagh | 1 |  | 2 | 1 | 1 |  | 1 | 0 |
| 7 | Samara Parr | 7 |  | 4 | 0 | 0 |  | 0 | 187 |
| 8 | Tom Isaac | 5 |  | 0 | 1 | 1 |  | 2 | 125 |
| 9 | Carter Waterson | 4 |  | 2 | 0 | 0 |  | 0 | 12 |
| 10 | Hank Bailey | 3 |  | 3 | 1 | 0 |  | 0 | 158 |

Таблица Б.1.2 - Справочник Финансовых данных

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| НДФЛ | ПФР | ФСС | ФОМС | Размер льготы за 1 и 2 ребёнка | Размер льготы за 3 и последующих детей | Размер льготы за ребёнка инвалида | Размер льготы за опеку над ребёнком инвалидом | МРОТ |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 13.00 | 22.00 | 2.90 | 5.10 | 1400 | 3000 | 12000 | 6000 | 12792 |

Таблица Б.1.3 - Справочник Налоговой базы

|  |  |
| --- | --- |
| Год | Значение базы |
| 1 | 2 |
| 2020 | 1292000 |
| 2021 | 1465000 |

Таблица Б.1.4 - Справочник Выплат Сотрудникам

|  |  |  |  |
| --- | --- | --- | --- |
| Дата выплаты | Табельный номер сотрудника | Сумма всех надбавок | Сумма всех Вычетов |
| 1 | 2 | 3 | 4 |
| 12.05.2021 | 1 | 1000 | 100 |
| 13.05.2021 | 2 | 0 | 100 |
| 10.05.2021 | 3 | 500 | 400 |
| 11.05.2021 | 4 | 400 | 200 |
| 12.05.2021 | 5 | 600 | 0 |
| 11.05.2021 | 6 | 0 | 200 |
| 10.05.2021 | 7 | 0 | 0 |
| 13.05.2021 | 8 | 500 | 400 |
| 12.05.2021 | 9 | 0 | 0 |
| 11.05.2021 | 10 | 700 | 0 |
| 13.06.2021 | 1 | 300 | 100 |
| 10.06.2021 | 2 | 500 | 100 |
| 11.06.2021 | 3 | 1000 | 0 |
| 11.06.2021 | 4 | 0 | 200 |
| 12.06.2021 | 5 | 0 | 300 |
| 13.06.2021 | 6 | 0 | 0 |

Продолжение таблицы Б.1.4

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 10.06.2021 | 7 | 500 | 300 |
| 11.06.2021 | 8 | 400 | 200 |
| 12.06.2021 | 9 | 800 | 200 |
| 10.05.2021 | 10 | 1000 | 0 |

Таблица Б.1.5 Справочник Должностей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Должности | Название должности | Оклад | Процент травматизма | Уровень доступа к данным |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Администратор | 30000 | 1.1 | 3 |
| 2 | Бухгалтер | 15000 | 1.2 | 1 |
| 3 | Сотрудник отдела кадров | 16000 | 3.2 | 2 |
| 4 | Электрик | 13000 | 2.0 | 0 |
| 5 | Охранник | 21678 | 1.0 | 0 |
| 6 | Уборщица | 13222 | 0.8 | 0 |
| 7 | Главный бухгалтер | 20000 | 0.2 | 1 |

Приложение В.

Выходные данные контрольного примера



Рисунок В.1 – Выходные данные контрольного примера

Приложение Г.

Код программы

DB.cs

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows;

using System.Windows.Controls;

namespace BDZarplata.Classes

{

class DB

{

/// <summary>

/// загрузка данных из БД в DataGrid

/// </summary>

/// <param name="dataGrid"></param>

/// <param name="sql">SQL команда для выполнения (типа SELECT) </param>

public static void loadDataGrid(DataGrid dataGrid, string sql)

{

DB\_Connect.OpenConnection();

SqlDataAdapter mysqldataAdapter = new SqlDataAdapter(sql, DB\_Connect.connectionString);

DataTable dataTable = new DataTable();

mysqldataAdapter.Fill(dataTable);

dataGrid.ItemsSource = dataTable.DefaultView;

DB\_Connect.CloseConnection();

}

/// <summary>

/// Загрузка данных из БД в список

/// </summary>

/// <param name="comboBox">Список куда будет загружены данные из БД</param>

/// <param name="sql">SQL команда по которой происходит выборка</param>

/// <param name="numberCol">Номер столбца из выборки (начиная с 0) который присваивается списку </param>

public static void LoadDataComboBox(ComboBox comboBox, string sql, int numberCol)

{

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(sql, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{ comboBox.Items.Add(reader.GetValue(numberCol).ToString()); }

DB\_Connect.CloseConnection();

}

/// <summary>

/// Загрузка данных из БД в список

/// </summary>

/// <param name="listbox">Список куда будет загружены данные из БД</param>

/// <param name="sql">SQL команда по которой происходит выборка</param>

/// <param name="numberCol">Номер столбца из выборки (начиная с 0) который присваивается списку </param>

public static void LoadDataListBox(ListBox listbox, string sql, int numberCol)

{

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(sql, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{ listbox.Items.Add(reader.GetValue(numberCol).ToString()); }

DB\_Connect.CloseConnection();

}

/// <summary>

/// Выполнение SQL Запроса к БД

/// </summary>

/// <param name="sql">SQL команда для выполнения</param>

/// <returns>Первый столбец первой строки набора результатов или пустая ссылка</returns>

public static object queryScalar(string sql)

{

Manager.UpdateLabel("Выполняю запрос...");

object sqlValue = null;

DB\_Connect.OpenConnection();

try

{

SqlCommand command = new SqlCommand(sql, DB\_Connect.myConnection);

sqlValue = command.ExecuteScalar();

Manager.UpdateLabel("Запрос успешно выполнен!");

}

catch (Exception ex) { MessageBox.Show("Произошла Ошибка при обработке SQL запроса \n" + ex.Message + "\n SQL Запрос: \n" + sql, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); }

finally { DB\_Connect.CloseConnection(); }

if (sqlValue != null)

{

return sqlValue;

}

else

{

sqlValue = "null";

return sqlValue;

}

}

/// <summary>

/// Выполнение запроса к БД C возвратом указанного столбца

/// </summary>

/// <param name="SQL\_Comand">SQL команда для выполнения</param>

/// <param name="Column"> Номер возвращаемого столбца</param>

/// <returns>Указанный столбец первой строки набора результатов или пустая ссылка</returns>

public static string queryScalar(string SQL\_Comand, int Column)

{

try

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(SQL\_Comand, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

string temp = reader[Column].ToString();

Manager.UpdateLabel("Запрос успешно выполнен!");

if (temp != null)

{

return temp;

}

else

{

temp = "null";

return temp;

}

}

catch (Exception ex) { MessageBox.Show("Произошла Ошибка \n" + ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); return ""; }

finally { DB\_Connect.CloseConnection(); }

}

/// <summary>

/// Выполнение запроса к БД

/// </summary>

/// <param name="SQL\_Comand">SQL команда для выполнения</param>

/// <param name="Column"> Название возвращаемого столбца</param>

/// <returns>Указанный столбец первой строки набора результатов или пустая ссылка</returns>

public static string queryScalar(string SQL\_Comand, string Column)

{

try

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(SQL\_Comand, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

string temp = reader[Column].ToString();

Manager.UpdateLabel("Запрос успешно выполнен!");

if (temp != null)

{

return temp;

}

else

{

temp = "";

return temp;

}

}

catch (Exception ex) { MessageBox.Show("Произошла Ошибка \n" + ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); return ""; }

finally { DB\_Connect.CloseConnection(); }

}

/// <summary>

/// Выполнение запроса к БД

/// </summary>

/// <param name="SQL\_Comand">SQL команда для выполнения</param>

/// <param name="Columns">Массив индексов возвращаемых столбцов (начиная с 0) </param>

/// <returns> массив значений Указанных столбцов первой строки набора результатов или пустая ссылка</returns>

public static string[] queryScalar(string SQL\_Comand, int[] Columns)

{

string[] results = new string[Columns.Length];

try

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(SQL\_Comand, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

for (int i = 0; i < Columns.Length; i++)

{

results[i] = reader[Columns[i]].ToString();

}

Manager.UpdateLabel("Запрос успешно выполнен!");

return results;

}

catch (Exception ex)

{

MessageBox.Show("Произошла Ошибка при чтении данных \n" + ex.Message, "Ошибка\n", MessageBoxButton.OK, MessageBoxImage.Error);

results = new[] { "" };

return results;

}

finally { DB\_Connect.CloseConnection(); }

}

/// <summary>

/// Выполнение запроса к БД

/// </summary>

/// <param name="SQL\_Comand">SQL команда для выполнения(Не рекомендуется использовать эту перегрузку если в запросе содержаться столбцы с одинаковым именем)</param>

/// <param name="Columns">Массив названий возвращаемых столбцов </param>

/// <returns> массив значений Указанных столбцов первой строки набора результатов или пустая ссылка</returns>

public static string[] queryScalar(string SQL\_Comand, string[] Columns)

{

string[] results = new string[Columns.Length];

try

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(SQL\_Comand, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

for (int i = 0; i < Columns.Length; i++)

{

results[i] = reader[Columns[i]].ToString();

}

Manager.UpdateLabel("Запрос успешно выполнен!");

return results;

}

catch (Exception ex)

{

MessageBox.Show("Произошла Ошибка при чтении данных \n" + ex.Message, "Ошибка\n", MessageBoxButton.OK, MessageBoxImage.Error);

results = new[] { "" };

return results;

}

finally { DB\_Connect.CloseConnection(); }

}

/// <summary>

/// Запрос к БД с получением данных из таблицы (таблиц)

/// </summary>

/// <param name="SQL\_Comand">SQL команда для выполнения(типа Select)</param>

/// <param name="CountColumn">Количество столбцов для возврата</param>

/// <param name="CountString">Количество строк для возврата(возврат начинается с 1 строки возвращаемым запросом</param>

/// <param name="Results">[c,s]Массив результатов где перебор с - столбца , s- строки</param>

public static void ReturnTable(string SQL\_Comand, out string[,] Results, int CountString = 1, int CountColumn = 1)

{

Results = new string[CountColumn, CountString];

try

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

SqlCommand command = new SqlCommand(SQL\_Comand, DB\_Connect.myConnection);

SqlDataReader reader = command.ExecuteReader();

int s = 0;

while (reader.Read() && s < CountString)

{

for (int с = 0; с < CountColumn; с++)

{

Results[с, s] = reader[с].ToString();

}

s++;

}

Manager.UpdateLabel("Запрос успешно выполнен!");

}

catch (Exception ex)

{

MessageBox.Show("Произошла Ошибка при чтении таблицы \n" + ex.Message, "Ошибка\n", MessageBoxButton.OK, MessageBoxImage.Error);

}

finally { DB\_Connect.CloseConnection(); }

}

/// <summary>

/// Выполнение внешнего SQl Файла или запроса с возвратом количества строк

/// </summary>

/// <param name="sql"></param>

/// <returns> количество задействованных в инструкции строк или -1 если при обработке запроса происходит ошибка </returns>

public static int queryData(string sql)

{

Manager.UpdateLabel("Выполняю запрос...");

DB\_Connect.OpenConnection();

try

{

SqlCommand command = new SqlCommand(sql, DB\_Connect.myConnection);

return command.ExecuteNonQuery();

}

catch (Exception ex)

{

MessageBox.Show("Произошла Ошибка \n" + ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return -1;

}

finally { DB\_Connect.CloseConnection(); }

}

}

}

DB\_Connect.cs

using System;

using System.Data.SqlClient;

using System.Windows;

namespace BDZarplata.Classes

{

public class DB\_Connect

{

public static string connectionString = "Data Source=localhost; Initial Catalog=BD\_Zarplata; Integrated Security=true;";

public static SqlConnection myConnection = new SqlConnection(connectionString);

/// <summary>

/// создание соединения с БД

/// </summary>

public static void OpenConnection()

{

if (myConnection.State == System.Data.ConnectionState.Closed)

{

myConnection.Open();

}

}

/// <summary>

/// закрытие соединения с БД

/// </summary>

public static void CloseConnection()

{

if (myConnection.State == System.Data.ConnectionState.Open)

{

myConnection.Close();

}

}

/// <summary>

/// Проверка на возможность установки соединения с БД

/// </summary>

/// <param name="DataSource">Адрес компьютера на котором храниться БД</param>

/// <param name="InitialCatalog">Название БД</param>

/// <param name="IntegratedSecurity">проверку подлинности</param>

/// <returns></returns>

public static bool OpenClouseConnection(string DataSource = "localhost", string InitialCatalog = "BD\_Zarplata", bool IntegratedSecurity = true)

{

string connectStr = "Data Source=" + DataSource + "; Initial Catalog=" + InitialCatalog + "; Integrated Security=" + IntegratedSecurity + ";";

try

{

SqlConnection userConnection = new SqlConnection(connectStr);

if (userConnection.State == System.Data.ConnectionState.Closed)

{

userConnection.Open();

}

if (userConnection.State == System.Data.ConnectionState.Open)

{

userConnection.Close();

myConnection = userConnection;

}

return true;

}

catch (Exception ex)

{

MessageBox.Show("Произошла ошибка:\n" + ex.Message + "\n Проверьте корректность введенных данных\n" + connectStr);

return false;

}

}

}

}

DG.cs

using System;

using System.Windows.Controls;

namespace BDZarplata.Classes

{

class DG

{

/// <summary>

/// Получение значения выбранной ячейки

/// </summary>

/// <param name="dataGrid1">DataGrid в котором находиться нужная ячейка</param>

/// <param name="selectedColumn">Номер столбца (начиная с 0) в котором находиться ячейка.

/// Присвоить -1 для автоматического определения столбца </param>

/// <returns>значение ячейки или ""</returns>

public static string GetSelectCell(DataGrid dataGrid1, int selectedColumn = -1)

{

try

{

string result = "";

if (selectedColumn == -1) { selectedColumn = dataGrid1.CurrentCell.Column.DisplayIndex; }

var selectedCell = dataGrid1.SelectedCells[selectedColumn];

var cellContent = selectedCell.Column.GetCellContent(selectedCell.Item);

if (cellContent is TextBlock)

{

result = (cellContent as TextBlock).Text;

}

return result;

}

catch (Exception ex) { Manager.UpdateLabel("Ошибка в returnCell, возможно не выбрана ячейка для возврата \n" + ex.Message); return ""; }

}

}

}

Manager.cs

using System.Windows.Controls;

namespace BDZarplata.Classes

{

class Manager

{

public static Frame MainFrame { get; set; }

public static Label LabelStatus { get; set; }

public static ProgressBar MainProgressBar { get; set; }

/// <summary>

/// отладочный вывод сообщений в отдельной Label

/// </summary>

/// <param name="newContent">новое сообщение для вывода</param>

public static void UpdateLabel(string newContent = "")

{

LabelStatus.Content = newContent;

}

}

}

Procedure.cs

using System.Collections.Generic;

using System.Windows;

namespace BDZarplata.Classes

{

class Procedure

{

/// <summary>

/// Обновление значений указанной таблицы в БД

/// </summary>

/// <param name="NameTable">Имя таблицы</param>

/// <param name="ColumnsName">Массив имен столбцов</param>

/// <param name="ColumnsData">Массив значений этих столбцов</param>

/// <param name="Where">Условие отбора</param>

public static void UpdateTable(string NameTable, List<string> ColumnsName, List<string> ColumnsData, string Where)

{

string sqlComand = $"UPDATE {NameTable} SET {ColumnsName[0]} = {ColumnsData[0]}";

if (ColumnsData.Count == ColumnsName.Count)

{

for (int i = 1; i < ColumnsData.Count; i++)

{

sqlComand += $", {ColumnsName[i]} = {ColumnsData[i]}";

}

sqlComand += Where;

Classes.DB.queryScalar(sqlComand);

}

else

{

MessageBox.Show("err UpdateTable \n Количество столбцов не совпадает с количеством значений");

}

}

/// <summary>

/// Добавление значений указанной таблицы в БД

/// </summary>

/// <param name="NameTable">Имя таблицы</param>

/// <param name="ColumnsName">Массив имен столбцов</param>

/// <param name="ColumnsData">Массив значений этих столбцов</param>

public static void InsertTable(string NameTable, List<string> ColumnsName, List<string> ColumnsData)

{

string sqlComand = $"INSERT INTO {NameTable} ( {ColumnsName[0]}";

if (ColumnsData.Count == ColumnsName.Count)

{

for (int i = 1; i < ColumnsName.Count; i++)

{

sqlComand += $", {ColumnsName[i]}";

}

sqlComand += $") VALUES( {ColumnsData[0]} ";

for (int i = 1; i < ColumnsData.Count; i++)

{

sqlComand += $", {ColumnsData[i]}";

}

sqlComand += ")";

DB.queryScalar(sqlComand);

}

else

{

MessageBox.Show("err InsertTable \n Количество столбцов не совпадает с количеством значений");

}

}

}

}

СПИСОК ИСТОЧНИКОВ

1. Федеральный закон от 29.12.2006 N 255-ФЗ (ред. от 29.12.2020) «Об обязательном социальном страховании на случай временной нетрудоспособности и в связи с материнством» Статья 14.
2. Налоговый Кодекс Российской Федерации Статья 218. Стандартные налоговые вычеты
3. "Трудовой кодекс Российской Федерации" статья 136
4. Основы проектирования баз данных: учебник для студ. Учреждений сред. Проф образования Г.Н. Федорова.-М.: Издательский центр «Академия», 2017. – 224 с.
5. Г.Н. Федорова. Разработка и администрирование и защита баз данных: учебник для студ. учреждений сред. проф. образования. –М.: Издательский центр «Академия», 2018.- 288с
6. 1С-Старт [Электронный ресурс]; Предельная величина базы для начисления страховых взносов: новые лимиты в 2019 и 2020 году; авт. 1С-Старт;2021; – Режим доступа: [https://www.regberry.ru/](https://www.regberry.ru/nalogooblozhenie/predelnaya-velichina-bazy-dlya-nachisleniya-strahovyh-vznosov-novye-limity-v-2019-i-2020-godu) , свободный. Загл. с экрана – Яз. рус.
7. Ip-on-line.ru [Электронный ресурс]; Порядок начисления и выплаты зарплаты – Режим доступа <https://ip-on-line.ru/kadry/poryadok-nachisleniya-i-vyplaty-zarplaty.html>, свободный. Загл. с экрана – Яз. рус.
8. subsidii.net [Электронный ресурс]: Оплата больничного листа в 2021 году: сроки выплаты и размер процентов. – Режим доступа: [https://subsidii.net/](https://subsidii.net/posobiya-fss/oplata-bolnichnogo-lista-v-godu-sroki-vyplaty-i-razmer-procentov.html) , свободный. Загл. с экрана – Яз. рус.
9. Ассистентус [Электронный ресурс]; Расчетный листок по заработной плате – Режим доступа <https://assistentus.ru/forma/raschetnyj-listok/>, свободный. Загл. с экрана – Яз. рус.
10. Богданова А.Л. Базы данных. Теория и практика применения (2-е издание) [Электронный ресурс]: учебное пособие/ Богданова А.Л., Дмитриев Г.П., Медников А.В.— Электрон. текстовые данные.— Химки: Российская международная академия туризма, 2016.— 128 c.— Режим доступа: http://www.iprbookshop.ru/47625.— ЭБС «IPRbooks», по паролю
11. ЗАРПЛАТА/Практический журнал для бухгалтеров для расчета зарплаты [Электронный ресурс]: Выплата больничного: порядок и новые сроки в 2021 году; – Режим доступа:[https://www.zarplata-online.ru/](https://www.zarplata-online.ru/art/162435-vyplata-bolnichnogo-sroki-v-2021-godu) , свободный. Загл. с экрана – Яз. рус.
12. КонсультантПлюс / надежная правовая поддержка [Электронный ресурс] ; – Режим доступа: <http://www.consultant.ru>, свободный. Загл. с экрана – Яз. рус.
13. Контур [Электронный ресурс]: Журнал / МРОТ — 2021: изменения; авт. Марина Крицкая Режим доступа: https://kontur.ru/ , свободный. Загл. с экрана – Яз. рус.
14. Контур.Школа [Электронный ресурс] : Расчет и оплата больничного листа в 2021 году; авт. Бусыгина Ю. О. . – Режим доступа: <https://school.kontur.ru/>, свободный. Загл. с экрана – Яз. рус.
15. Култыгин О.П. Администрирование баз данных. СУБД MS SQL Server [Электронный ресурс]: учебное пособие/ Култыгин О.П.— Электрон. текстовые данные.— М.: Московский финансово-промышленный университет «Синергия», 2016.— 232 c.— Режим доступа: http://www.iprbookshop.ru/17009.— ЭБС «IPRbooks», по паролю
16. НАЛОГ-НАЛОГ.ру [Электронный ресурс]; Облагается ли больничный лист (больничный) НДФЛ?;авт. Степанова Наталья.- – Режим доступа: [https://nalog-nalog.ru/](https://nalog-nalog.ru/posobiya/posobie_po_vremennoj_netrudosposobnosti_bolnichnyj/oblagaetsya_li_bolnichnyj_list_bolnichnyj_ndfl/) , свободный. Загл. с экрана – Яз. рус.
17. Основы современных баз данных [Электронный ресурс]: методическая разработка к выполнению лабораторных работ (№1-3)/ — Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2016.— 37 c.— Режим доступа: http://www.iprbookshop.ru/22906.— ЭБС «IPRbooks», по паролю
18. Туманов В.Е. Основы проектирования реляционных баз данных [Электронный ресурс]/ Туманов В.Е.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 502 c.— Режим доступа: http://www.iprbookshop.ru/22431.— ЭБС «IPRbooks», по паролю
19. Упрощенка [Электронный ресурс]: Оплата и расчет больничного листа в 2021 году: изменения и новые правила;2021; – Режим доступа: <https://www.26-2.ru/> , свободный. Загл. с экрана – Яз. рус.
20. Швецов В.И. Базы данных [Электронный ресурс]/ Швецов В.И.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 218 c.— Режим доступа: http://www.iprbookshop.ru/16688.— ЭБС «IPRbooks», по паролю
21. <https://www.klerk.ru/buh/articles/506743/>

1. «количество рабочих дней в месяце» – количество полей в таблице «график Работы» с определенным табельным номером и статусом дня «рабочий» [↑](#footnote-ref-1)
2. распространяется на родителя, супруга (супругу) родителя, усыновителя, опекуна, попечителя, приемного родителя, супруга (супругу) приемного родителя, на обеспечении которых находится ребенок([Статья 218 НК РФ](http://nalog.garant.ru/fns/nk/1458a0a55ea2ab5e42fde990265bcdf4/#block_21814) Пункт 1.4) [↑](#footnote-ref-2)
3. Если превышает предельную базу страховых взносов – берётся значение базы [↑](#footnote-ref-3)
4. Страховой стаж Размер больничного

   менее 5 лет (60 мес.) 60% среднего заработка

   от 5 до 8 лет (60-96 мес) 80% среднего заработка

   8 лет и более (96 мес) 100% среднего заработка [↑](#footnote-ref-4)
5. Поля, выделенные подчеркиванием, являются ключевыми [↑](#footnote-ref-5)
6. распространяется на опекуна, попечителя, приемного родителя, супруга (супругу) приемного родителя, на обеспечении которых находится ребенок [↑](#footnote-ref-6)
7. \*Согласно [статье 218](http://nalog.garant.ru/fns/nk/1458a0a55ea2ab5e42fde990265bcdf4/#block_21814) НК РФ:. Налогоплательщикам, имеющим в соответствии с [подпунктами 1](http://nalog.garant.ru/fns/nk/1458a0a55ea2ab5e42fde990265bcdf4/#block_21811) и [2 пункта 1](http://nalog.garant.ru/fns/nk/1458a0a55ea2ab5e42fde990265bcdf4/#block_2182) настоящей статьи право более чем на один стандартный налоговый вычет, предоставляется максимальный из соответствующих вычетов. [↑](#footnote-ref-7)
8. Денежные поощрения, не связанные с окладом (премии и т.д.)

   Региональные доп. Выплаты

   Доп. выплаты гос. сотрудникам за стаж [↑](#footnote-ref-8)
9. Дисциплинарные, Материальная ответственность и тому подобное [↑](#footnote-ref-9)
10. PK-первичный ключ

    FK-внешний ключ [↑](#footnote-ref-10)