Министерство транспорта Российской Федерации

Федеральное агентство железнодорожного транспорта

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Дальневосточный государственный университет путей сообщения»

Кафедра «Информационные технологии и системы»

ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЕЯТЕЛЬНОСТИ АПТЕЧНОГО СКЛАДА «ЕКЛИНИК»

Курсовая работа

дисциплина «Управление данными»

КР 09.03.04. 15.00.БО221ПИН

Студент \_\_\_\_\_ Кузнецов А.Ю.

*(подпись, дата)*

Заведующий кафедры ИТиС, к.т.н.,  
 доцент Попов М.А.

*(подпись, дата)*

Хабаровск 2023

**Задание на курсовую работу**

**09.03.04** Программная инженерия

«Разработка клиент-серверного приложения»

по дисциплине ***«Управление данными»***

**Студент: \_\_\_Кузнецов А.Ю.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_БО221ПИН группа. Вариант 15**

*Дата выдачи задания:* **«10» февраля 2022 года**

*Срок сдачи:* **до «30» мая 2022 года**

**Инструментарий:**

*Операционная система рабочей станции: Windows 8.1, Windows 10 (по выбору студента).*

*Сервер базы данных: MySQL Server версии выше 8.0 (по выбору студента).*

*Visual Studio 2017/2019/2022 (по выбору студента).*

**Задание на разработку:**

1. Описать деловые процессы, происходящие при работе предприятия (см. свой вариант).

2. Разработать ER-модель (entity-relationship model) – модель «Сущность-связь» для своей предметной области (см.вариант задания).

3. Разработать базу данных с использованием ER-модели.

4. Обеспечить её безопасность с применением механизма ссылочной целостности.

5. Создать и заполнить базу данных (не менее 20 записей на одну основную таблицу и не менее 5 записей на таблицу-справочник).

6. Написать приложение на языке C#, работающее с базой данных и реализующее следующие основные функции:

6.1. Работа в режиме клиент-сервер.

6.2. Просмотр, ввод, редактирование и удаление записей в любой таблице с отслеживанием «авторства».

6.3. Реализация хранимой процедуры.

**Пояснительная записка должна включать следующие основные разделы:**

1. Текст задания (этот бланк).
2. Свой вариант предприятия.
3. Оглавление.
4. Описание предметной области и схемы модели данных.
5. Реализация базы данных.
6. Разработка приложения (руководство разработчика).
7. Описание интерфейса (руководство пользователя).
8. Список литературы.
9. Компакт-диск с формами, отчетами, базой данных и текстовым файлом с паролями.

**Литература.**

1. Гурвиц Г.А. Microsoft Access 2010. Разработка приложений на реальном примере. СПб.:БХВ-Петербург, 2010. – 496 c.: ил. + CD-ROM – (Профессиональное программирование)

2. Смирнов С.Н. Безопасность систем баз данных. М: Гелиос, 2007.- 352 c.: ил.

**Задание выдал доцент кафедры**

**«Информационные технологии и системы» Попов М.А.**

Рецензия

на курсовую работу

студента группы БО221ПИН направления подготовки (специальности)

09.03.04 Программная инженерия

*Ф.И.О. студента*

выполненную по дисциплине: «Управление данными»

на тему: «Разработка клиент-серверного приложения.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Показатели оценки | Да | Нет |
| 1. | Курсовой проект выполнен в соответствии с требованиями и задачами курсового проектирования. |  |  |
| 2. | Курсовой проект выполнен в соответствии с заданием. |  |  |
| 3. | Кроме основных разделов, содержит самостоятельные решения поставленной задачи. |  |  |
| 4. | Программная часть выполнена в полном объеме. |  |  |
| 5. | Программная часть не содержит технических ошибок. |  |  |
| 6. | Пояснительная записка выполнена в соответствии со Стандартом ДВГУПС СТ 03-04. |  |  |
| 7. | Графический материал выполнен в соответствии со Стандартом ДВГУПС СТ 03-04. |  |  |

Курсовой проект при соответствующей защите заслуживает

оценки \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

|  |  |  |
| --- | --- | --- |
| Рецензент: |  |  |

Ст. преподаватель ДВГУПС .

*должность, организация Ф.И.О.*

*степень, звание (при наличии)*

Дата *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Вариант 15

Разработать прикладное программное обеспечение деятельности аптечного склада. Аптечный склад занимается оптовой продажей лекарств больницам и аптекам города. В его ассортименте — тысячи наименований лекарств, а также различных аптечных принадлежностей (градусники, шприцы, бинты и т. д.) Возможна продажа лишь тех лекарств, которые одобрены Минздравом РФ, т. е. имеют регистрационный номер Минздрава РФ. Поступающие лекарства сопровождаются документами — приходными накладными ведомостями. Покупатель получает счет-фактуру на выбранный товар, оплачивает сумму, указанную в ней, и после оплаты получает выходную накладную ведомость, по которой получает выбранный товар.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | GoodsID | Числовой | 10 | Регистрационный номер товара в базе |
| 2 | Name | Текстовый | 40 | Название товара |
| 3 | International | Текстовый | 40 | Международное название лекарства |
| 4 | Begin | Дата/ время | Авто | Дата производства |
| 5 | End | Дата/время | Авто | Годен до |
| 6 | Yes | Логический | 1 | Одобрено Минздравом РФ (да/нет) |
| 7 | RF | Текстовый | 20 | Регистрационный номер Минздрава РФ |
| 8 | Producer | Текстовый | 60 | Данные о производителе |
| 9 | Instructions | Поле Memo | Авто | Инструкция к лекарству |
| 10 | Batch | Текстовый | 20 | Вид упаковки |
| 11 | Seller | Текстовый | 20 | Название поставщика |
| 12 | Address | Текстовый | 60 | Адрес поставщика |
| 13 | Phone | Текстовый | 15 | Телефон поставщика |
| 14 | INN | Текстовый | 10 | ИНН поставщика |
| 15 | Sign | Текстовый | 1 | Признак посредника |
| 16 | Date | Дата/ время | Авто | Дата поступления на склад |
| 17 | Price | Денежный | 10 | Цена товара |
| 18 | GoodsInvoice | Числовой | 4 | Номер приходной накладной ведомости |
| 19 | ClientID | Числовой | 5 | Номер покупателя |
| 20 | Company | Текстовый | 25 | Название покупателя |
| 21 | Address | Текстовый | 60 | Адрес покупателя |
| 22 | Phone | Текстовый | 15 | Телефон покупателя |
| 23 | CountNumber | Числовой | 4 | Номер счет-фактуры |
| 24 | DateStart | Дата/ время | Авто | Дата выписки счетфактуры |
| 25 | Sum | Денежный | 15 | Сумма к уплате |
| 26 | Cash | Логический | 1 | Оплата наличными (да/ нет) |
| 27 | Worker | Текстовый | 60 | Выдавший счетфактуру |
| 28 | Invoice | Числовой | 4 | Номер выходной накладной |
| 29 | INNClient | Текстовый | 10 | ИНН покупателя |
| 30 | Seller | Текстовый | 15 | Фамилия продавца |

Содержание

[Введение 6](#_Toc138013670)

[1 Описание предметной области 7](#_Toc138013671)

[1.1 Проектирование бизнес-процессов 7](#_Toc138013672)

[1.2 Проектирование меню программного комплекса или главной кнопочной формы 8](#_Toc138013673)

[2 Реализация базы данных 10](#_Toc138013674)

[2.1 Логическая модель данных 10](#_Toc138013675)

[2.2 Физическая модель данных 11](#_Toc138013676)

[3 Руководство разработчика 16](#_Toc138013677)

[3.1 Создание хранимых процедур 16](#_Toc138013678)

[3.2 Создание стартовой формы 16](#_Toc138013679)

[3.3 Создание главной формы 17](#_Toc138013680)

[3.4 Создание вкладки «Клиенты» 18](#_Toc138013681)

[3.5 Создание вкладок «Поставщики», «Счёт-фактуры», «Список товаров» 21](#_Toc138013682)

[3.6 Создание страницы «Тарифы» 24](#_Toc138013683)

[3.7 Создание страницы «Расписания» 27](#_Toc138013684)

[4 Руководство пользователя 29](#_Toc138013685)

[Заключение 37](#_Toc138013686)

[Список использованных источников 38](#_Toc138013687)

Введение

Работа с базой данных с помощью «PostgreSQL» и приложением, написанным на языке C# в «Microsoft Visual Studio».

Актуальность работы

В связи с расширением компании, большое значение приобрела идея разработки прикладного программного обеспечения деятельности аптечного склада.

Объектом исследования выступает аптечный склад, деятельностью которого

является оптовая продажа лекарств больницам и аптекам города.

Предмет исследования, создание прикладного программного обеспечения для данного предприятия.

Цель работы – разработка программного обеспечения деятельности аптечного склада. Задачами курсового проекта являются:

* + знакомство с интерфейсом заданной программы;
  + изучение её возможностей;
  + получение знаний по созданию баз данных;
  + обучение заполнению таблиц в созданных базах данных;
  + обучение созданию прикладного программного обеспечения для базы данных;
  + обучение анализу эргономики интерфейса инструментального средства и его возможностей.

Объектом курсового проекта выступает база данных.

Предметом курсового проекта является система управления базами данных.

Теоретической базой курсовой работы послужили информационные источники глобальной сети и учебное пособие СУБД.

Программными средствами, используемыми при выполнении курсового проекта, послужили программы «Microsoft Visual Studio» и «pgAdmin 4».

# Описание предметной области

Основной задачей предприятия является оптовая продажа лекарств больницам и аптекам города. Возможна продажа лишь тех лекарств, которые одобрены Минздравом РФ, т. е. имеют регистрационный номер Минздрава РФ. Поступающие лекарства сопровождаются документами — приходными накладными ведомостями. Покупатель получает счет-фактуру на выбранный товар, оплачивает сумму, указанную в ней, и после оплаты получает выходную накладную ведомость, по которой получает выбранный товар.

## 1.1 Проектирование бизнес-процессов

Деятельность аптечного склада складывается из трех подразделений:

-отдел работы с заказчиками;

-отдел работы с товарами;

-отдел работы с поставщиками.

Первоначально клиент обращается в отдел работы с заказчиками, где составляет заявку на нужный товар. После выбора товара, клиент оплачивает его, а сотрудник магазина выдает ему квитанцию и заносит его в базу данных с уникальным регистрационным номером заказа.

После занесения клиента в БД, его уникальный регистрационный номер заказа передается в отдел работы с заказами. Этот отдел составляет накладную и проверяет, есть ли товар.

Если товара нет на складе, то накладная с уникальным номером товара передается в отдел работы с поставщиками. Этот отдел составляет накладную товаров, которые должны быть на складе, в том числе и зарезервированные. Затем отдел организовывает доставку партии на склад, через различных поставщиков.

Когда товар приходит на склад, отдел работы с заказами его обрабатывает и передает в отдел работы с заказчиками.

## 1.2 Проектирование меню программного комплекса или главной кнопочной формы

В проектируемом программном комплексе разобьем всю имеющуюся информацию на шесть подменю:

- получение поддержки по работе с программой (Поддержка);

- работа с информацией о заказчиках (Оформление заказа);

- работа с информацией о платежах (Платежи);

- работа с информацией о накладных (Накладные);

- работа с информацией о заказах (Формирование заказа);

- работа с информацией о доставках партий товаров на склад (Отдел работы с поставщиками).

Для входа в систему пользователю (сотруднику) необходимо зарегистрироваться, вход в систему будет разрешен только ограниченному числу людей. Нужно ввести свою фамилию и пароль. Если пароль введен неверно, то будет предложено ввести пароль три раза. После успешной регистрации пользователь в главное меню системы, из которого будут доступны все возможные подменю. Термин “работа с информацией” подразумевает возможность просмотра, корректировки и добавления новых записей, сохранения исправлений и возможно печать, если это необходимо в конкретном случае.

Работа с информацией о заказчиках, заключается в создание заказа и присвоение уникального регистрационного номера заказа, в котором хранится вся информация о клиенте. Форма “Оформление заказа” подразумевает наличие определенных полей для занесения в них данных по новому клиенту. Основные поля: регистрационный номер заказа, ФИО заказчика, дата заказа, дата, к которой необходимо выполнить заказ, кол-во товара к доставке, стоимость заказа, оплачен ли заказ, адрес доставки.

Работа с платежами подразумевает три вида работы: поиск уже созданного клиента, добавление нового, а также вывод специальных отчетов по клиентам (например, квитанция об оплате товара).

Поиск заказа осуществляется по регистрационному номеру заказа, введённым в определенные поля или выбранным из предполагаемого списка. Так как каждому клиенту выдается накладная, из этой же формы можно будет посмотреть информацию о заказанных товаров.

Каждой накладной соответствует регистрационный номер заказа, следовательно, из формы “Накладная” нужно перейти к регистрационному номеру заказа клиента и ознакомиться с ним. Тут же нужно вывести список осуществленных платежей и список заказанных товаров. Так как в накладной идет перечень заказанных товаров, из формы “Накладная” можно посмотреть информацию о товаре, нажав кнопку “Просмотр информации о товаре”.

Работа с информацией о товарах состоит из:

- поиска накладной;

- обработки заказа;

- отправки заказа.

Поиск накладной включает в себя печать, корректировку найденной накладной, просмотр соответствующего ей регистрационного номера заказа, просмотр информации о клиенте. Обработка заказа, состоит в поиске нужного товара на складе, либо в оформление заказа его у поставщиков. Отправка заказа, после того как товары поступили на склад, их собирают в одну упаковку и отправляют клиенту.

Работа с информацией о доставках партий товаров на склад заключается в составление накладных на доставку товаров от поставщика, а также в сообщение поставщику о забронированных товаров.

# Реализация базы данных

В pgAdmin 4 была создана база данных «pharma\_storage», которая будет содержать в себе таблицы, связи и хранимые процедуры.

## 2.1 Логическая модель данных

Прежде чем приступить к созданию базы данных, необходимо составить ER-диаграмму, которая представляет собой зависимость и связи сущностей. Данная диаграмма представлена на рисунке 2.1.

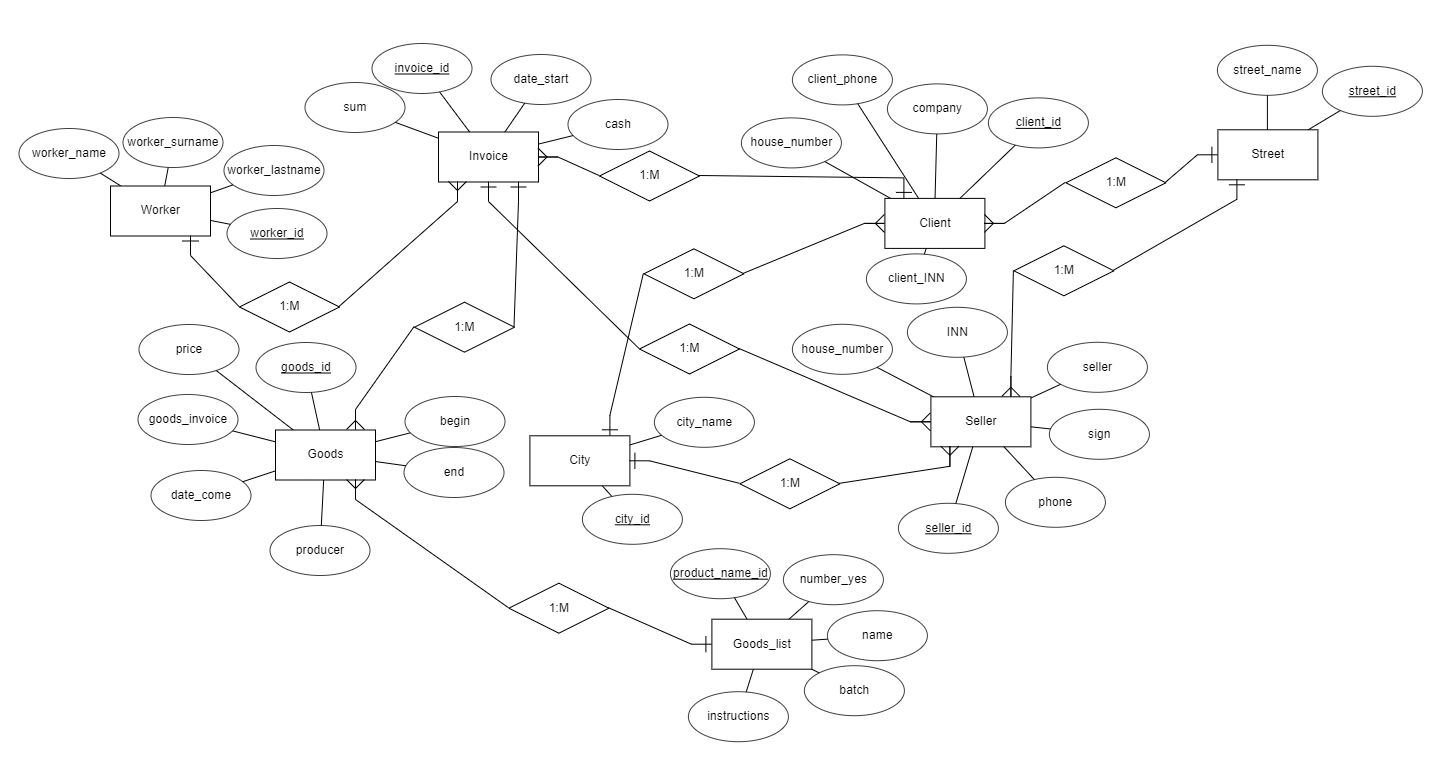


Рисунок 2.1 – ER-диаграмма

## 2.2 Физическая модель данных

Опираясь на логическую модель данных, необходимо было создать физическую модель. Для этого был использован веб-сервис «sqldbm.com». Готовую физическую модель можно наблюдать на рисунке 2.2.

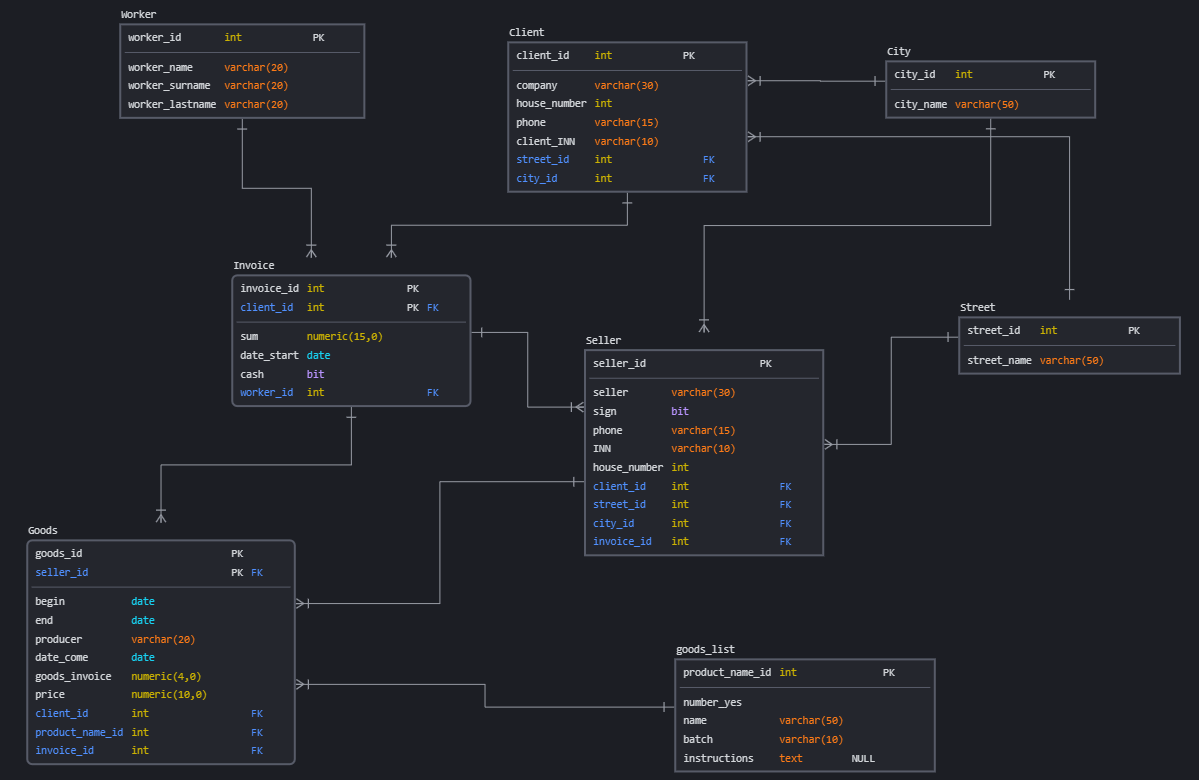


Рисунок 2.2 - Физическая схема данных

На основе данной модели был сгенерирован DDL - скрипт, что представлен ниже:

CREATE TABLE goods\_list

(

product\_name\_id int NOT NULL,

number\_yes boolean NOT NULL,

name varchar(50) NOT NULL,

batch varchar(10) NOT NULL,

instructions text NULL,

CONSTRAINT PK\_PNID PRIMARY KEY ( product\_name\_id )

);

CREATE TABLE Worker

(

worker\_id int NOT NULL,

worker\_name varchar(20) NOT NULL,

worker\_surname varchar(20) NOT NULL,

worker\_lastname varchar(20) NOT NULL,

CONSTRAINT PK\_WorkerId PRIMARY KEY ( worker\_id )

);

CREATE TABLE Street

(

street\_id int NOT NULL,

street\_name varchar(50) NOT NULL,

CONSTRAINT PK\_StreetId PRIMARY KEY ( street\_id )

);

CREATE TABLE City

(

city\_id int NOT NULL,

city\_name varchar(50) NOT NULL,

CONSTRAINT PK\_CityId PRIMARY KEY ( city\_id )

);

CREATE TABLE Client

(

client\_id int NOT NULL,

company varchar(30) NOT NULL,

house\_number int NOT NULL,

phone varchar(15) NOT NULL,

client\_INN varchar(10) NOT NULL,

street\_id int NOT NULL,

city\_id int NOT NULL,

CONSTRAINT PK\_ClientId PRIMARY KEY ( client\_id ),

CONSTRAINT FK\_CityId FOREIGN KEY ( city\_id ) REFERENCES City ( city\_id ) ON DELETE RESTRICT ON UPDATE CASCADE,

CONSTRAINT FK\_StreetId FOREIGN KEY ( street\_id ) REFERENCES Street ( street\_id ) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE INDEX FK\_1 ON Client

(

city\_id

);

CREATE INDEX FK\_2 ON Client

(

street\_id

);

CREATE TABLE Invoice

(

invoice\_id int NOT NULL,

client\_id int NOT NULL,

"sum" numeric(15,0) NOT NULL,

date\_start date NOT NULL,

cash bit NOT NULL,

worker\_id int NOT NULL,

CONSTRAINT PK\_InvoiceClientIds PRIMARY KEY ( invoice\_id, client\_id ),

CONSTRAINT FK\_WorkerId FOREIGN KEY ( worker\_id ) REFERENCES Worker ( worker\_id ) ON DELETE RESTRICT ON UPDATE CASCADE,

CONSTRAINT FK\_ClientId FOREIGN KEY ( client\_id ) REFERENCES Client ( client\_id ) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE INDEX FK\_3 ON Invoice

(

worker\_id

);

CREATE INDEX FK\_4 ON Invoice

(

client\_id

);

CREATE TABLE Seller

(

seller\_id int NOT NULL,

seller varchar(30) NOT NULL,

sign bit NOT NULL,

phone varchar(15) NOT NULL,

INN varchar(10) NOT NULL,

house\_number int NOT NULL,

client\_id int NOT NULL,

street\_id int NOT NULL,

city\_id int NOT NULL,

invoice\_id int NOT NULL,

CONSTRAINT PK\_SellerId PRIMARY KEY ( seller\_id ),

CONSTRAINT FK\_InvoiceClientIds FOREIGN KEY ( invoice\_id, client\_id ) REFERENCES Invoice ( invoice\_id, client\_id ) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK\_CityId FOREIGN KEY ( city\_id ) REFERENCES City ( city\_id ) ON DELETE RESTRICT ON UPDATE CASCADE,

CONSTRAINT FK\_StreetId FOREIGN KEY ( street\_id ) REFERENCES Street ( street\_id ) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE INDEX FK\_5 ON Seller

(

invoice\_id,

client\_id

);

CREATE INDEX FK\_6 ON Seller

(

city\_id

);

CREATE INDEX FK\_7 ON Seller

(

street\_id

);

CREATE TABLE Goods

(

goods\_id int NOT NULL,

seller\_id int NOT NULL,

begin date NOT NULL,

"end" date NOT NULL,

producer varchar(20) NOT NULL,

date\_come date NOT NULL,

goods\_invoice numeric(4,0) NOT NULL,

price numeric(10,0) NOT NULL,

client\_id int NOT NULL,

product\_name\_id int NOT NULL,

invoice\_id int NOT NULL,

CONSTRAINT PK\_GoodsSellerIds PRIMARY KEY ( goods\_id, seller\_id ),

CONSTRAINT FK\_PNId FOREIGN KEY ( product\_name\_id ) REFERENCES goods\_list ( product\_name\_id ) ON DELETE RESTRICT ON UPDATE CASCADE ,

CONSTRAINT FK\_SellerId FOREIGN KEY ( seller\_id ) REFERENCES Seller ( seller\_id ) ON DELETE CASCADE ON UPDATE CASCADE ,

CONSTRAINT FK\_InvoiceClientIds FOREIGN KEY ( invoice\_id, client\_id ) REFERENCES Invoice ( invoice\_id, client\_id ) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE INDEX FK\_8 ON Goods

(

product\_name\_id

);

CREATE INDEX FK\_9 ON Goods

(

seller\_id

);

CREATE INDEX FK\_10 ON Goods

(

invoice\_id,

client\_id

);

Выполнение данного запроса в приложении pgAdmin 4 создало готовую схему данных, представленной на рисунке 2.3.

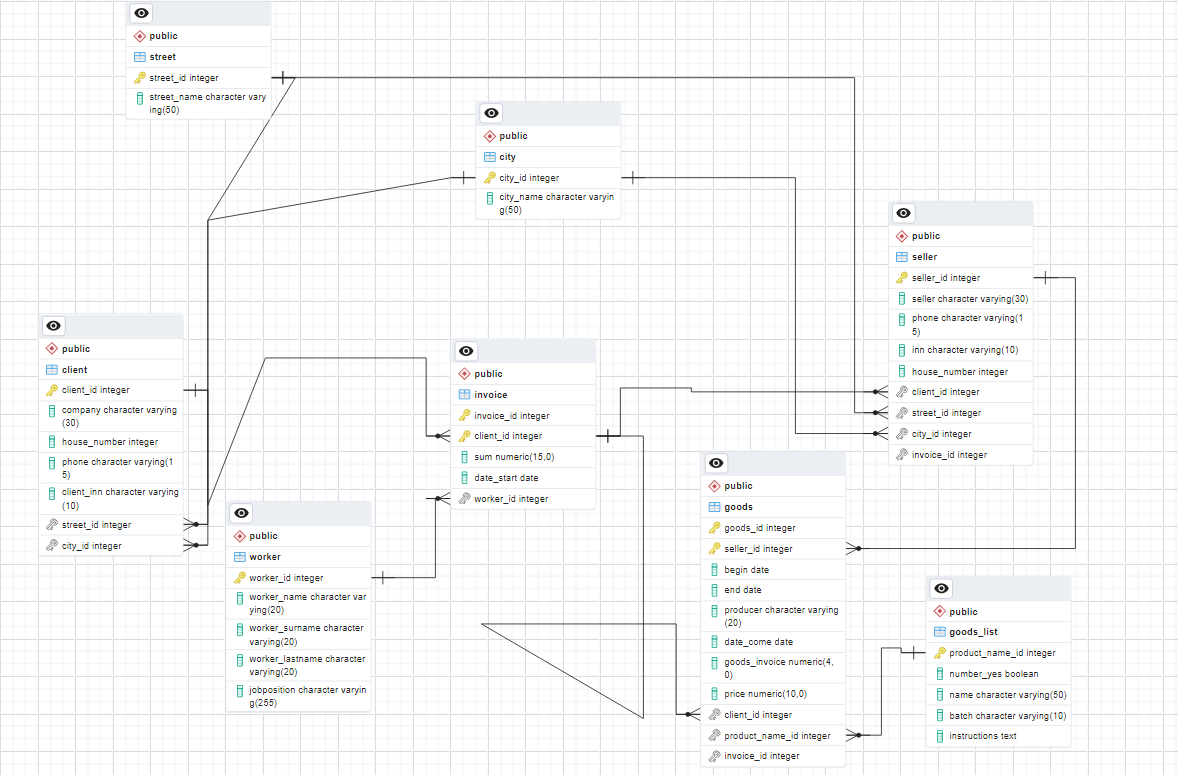


Рисунок 2.3 - Схема данных

# Руководство разработчика

## 3.1 Создание хранимых процедур

Перед началом работы были созданы хранимые процедуры, которые в последствии будут использоваться для создания, удаления и загрузки данных из таблиц. Их список перечислен ниже:

* + workeraddoredit4 – добавление записи в таблицу Worker;
  + delete\_worker – удаление записи из таблицы Worker;

Хранимых процедур всего две, функционал работы с оставшимися таблицами будет представлен ниже.

## 3.2 Создание стартовой формы

Первым шагом было создание стартовой формы, что включает в себя кнопки «Начать работу», «Справка» и «Выйти». При нажатии на кнопку «Начать работу» откроется главная форма (MainForm). Стартовая форма показана на Рисунке 3.1. При нажатии на кнопку «Справка» откроется форма Справка (InfoForm), Рисунок 3.2. Кнопка «Выход» завершает работу программы.

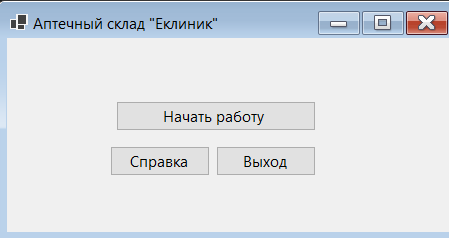


Рисунок 3.1 - Стартовая форма

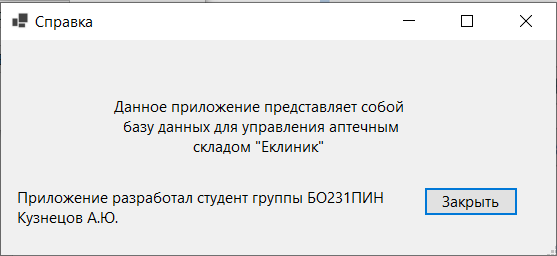


Рисунок 3.2 - Справочная форма

## 3.3 Создание главной формы

Предназначение главной формы – ввод, вывод и редактирование данных. Она изображена на рисунке 3.3. Для формы был создан элемент TabContol, с помощью которого были добавлены вкладки «Клиенты», «Товары», «Поставщики», «Работники склада», «Счёт-фактуры», «Список товаров», «SELECT», «ListView», «DT Filter», «LV Filter».

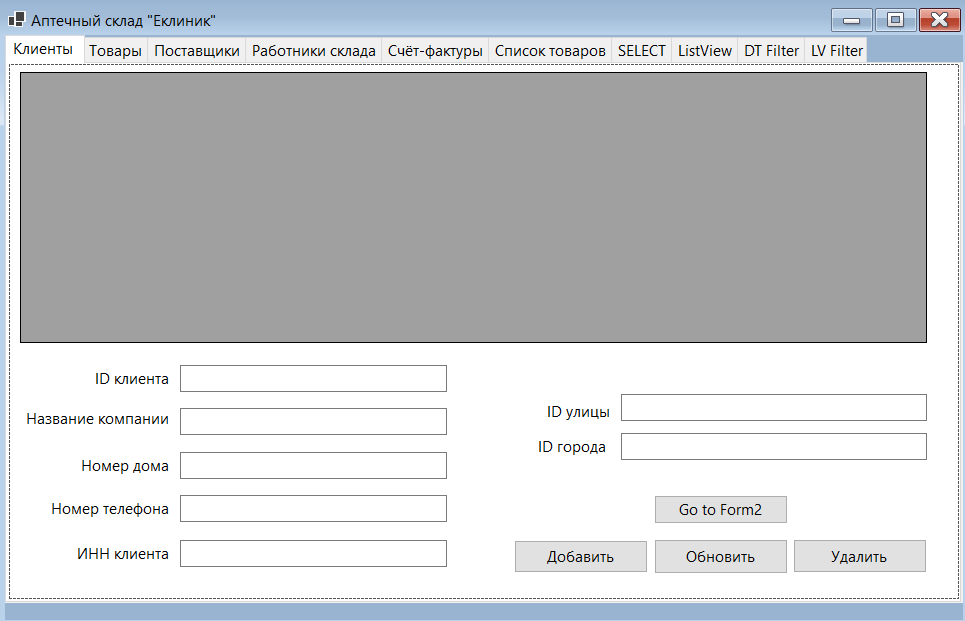


Рисунок 3.3 - Главная форма

## 3.4 Создание вкладки «Клиенты»

Вкладка «Клиенты» предназначена для ввода, редактирования и отображения данных о клиентах Аптечного склада «Еклиник»

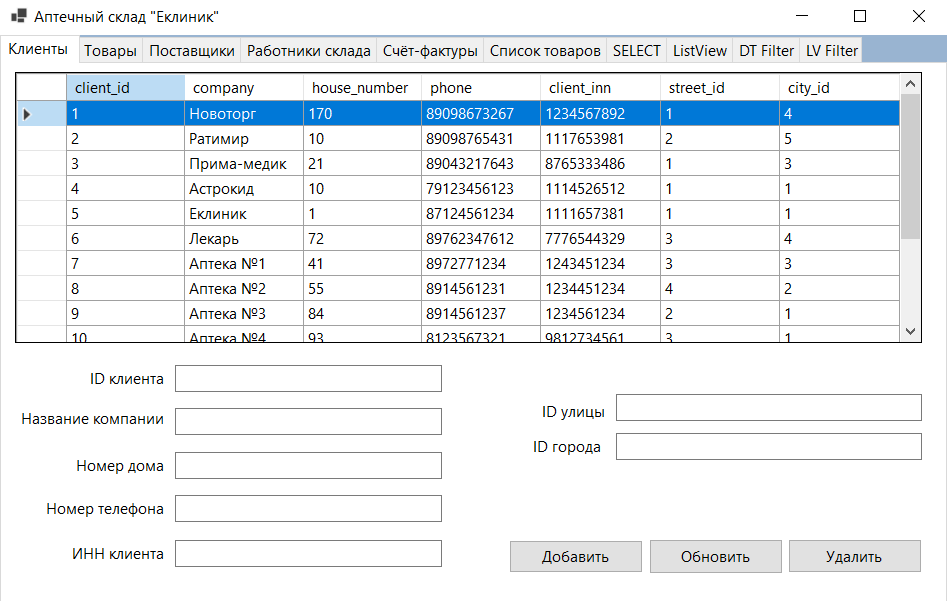


Рисунок 3.4 – Вкладка «Клиенты»

На вкладке присутствуют элементы:

* + TextBox – введение или редактирование данных о клиенте;
  + DataGridView – вывод данных из таблицы client;
  + Кнопки «Добавить» и «Удалить» – добавление и удаление записи из таблицы;
  + Кнопка «Обновить» - перезагрузка списка клиентов;
  + Двойное нажатие на поле клиента заполняет соответствующие поля textbox для редактирования

Чтобы DataGridView отоброжал актуальные данные из таблицы «Клиенты» нужно сначала настроить подключение к базе данных из приложения Microsoft Visual Studio:

private string connstring = String.Format("Server={0};Port={1};" +

"User Id={2};Password={3};Database={4};",

"localhost", 5432, "postgres", "1111", "pharma\_storage"

);

Отображение данных для всех таблиц было выведено в отдельные методы, реализующие SQL – запрос select \* from table\_name. Пример кода для отображения таблицы client:

private void Select\_client()

{

try

{

conn.Open();

sql = @"select \* from client";

cmd = new NpgsqlCommand(sql, conn);

dt = new DataTable();

dt.Load(cmd.ExecuteReader());

conn.Close();

dgvData.DataSource = null;

dgvData.DataSource = dt;

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Ошибка" + ex.Message);

}

}

Для того, чтобы добавить клиента необходимо будет заполнить все текстовые поля и нажать на кнопку «Добавить». Обработчик кнопки добавить:

private void add\_client\_btn\_Click(object sender, EventArgs e)

{

string company = textBoxcompany.Text;

string houseNumber = textBoxHousenum.Text;

string phone = textBoxPhone.Text;

string clientInn = textBoxInn.Text;

string clientId = textBoxId.Text;

string streetId = textBoxStreetId.Text;

string citytId = textBoxCityId.Text;

// Формирование SQL-запроса

string query = "INSERT INTO client (client\_id,company, house\_number, phone, client\_inn,street\_id,city\_id) " +

$"VALUES ('{clientId}','{company}', '{houseNumber}', '{phone}', '{clientInn}','{streetId}','{citytId}');";

// Выполнение SQL-запроса

NpgsqlCommand command = new NpgsqlCommand(query, conn);

conn.Open();

int rowsAffected = command.ExecuteNonQuery();

conn.Close();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены в базу данных.");

}

else

{

MessageBox.Show("Не удалось добавить данные в базу данных.");

}

}

Добавление клиента из приложения изображено на Рисунках 3.5 – 3.7

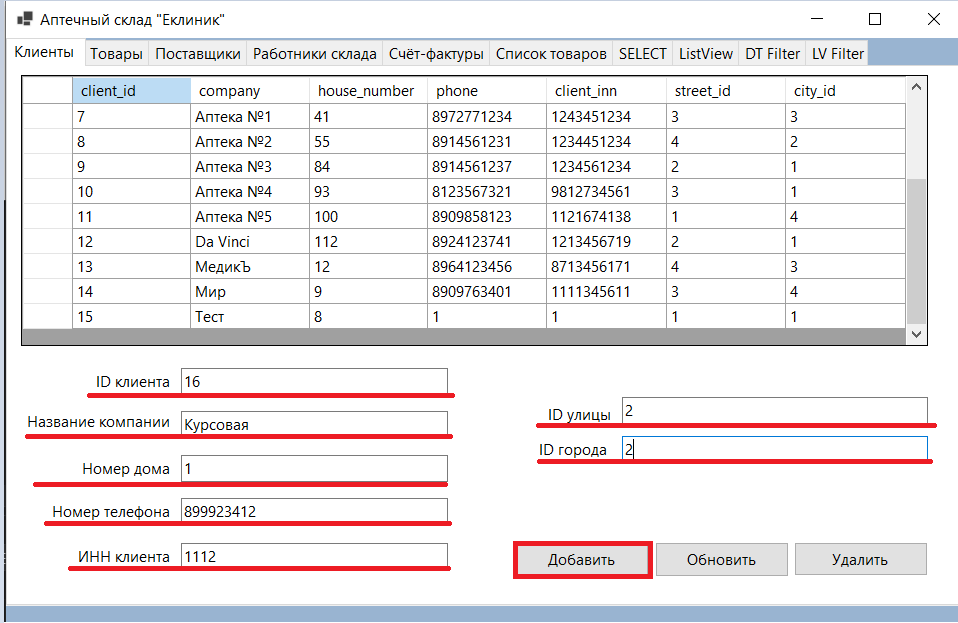


Рисунок 3.5 – Заполнение полей о новом клиенте

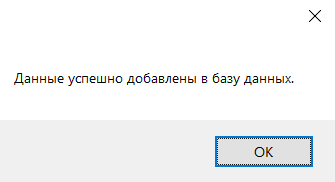


Рисунок 3.6 – Уведомление об успешном добавлении записи

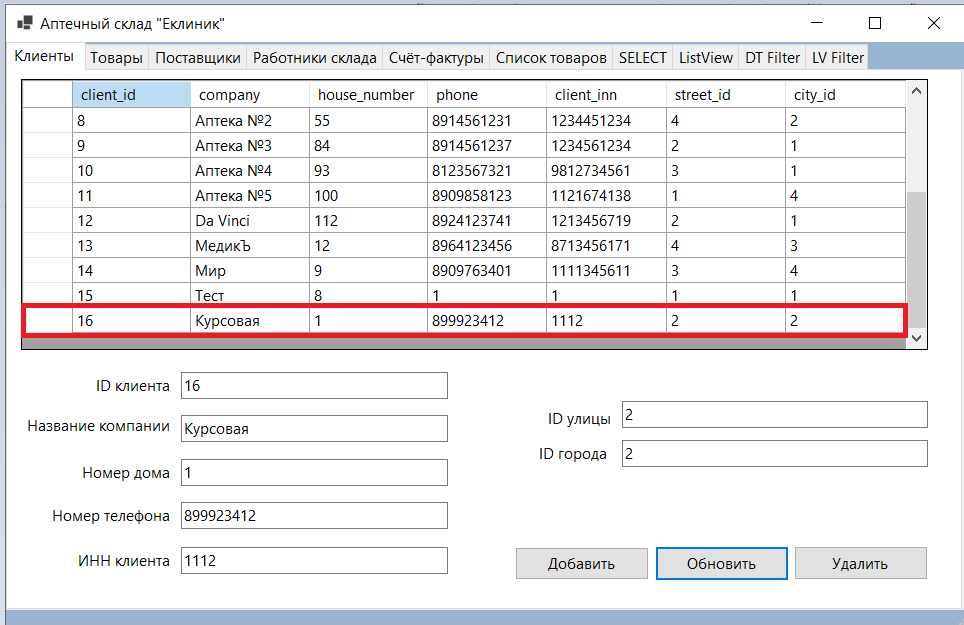


Рисунок 3.7 – Новая запись появилась в таблице

## 3.5 Создание вкладок «Поставщики», «Счёт-фактуры», «Список товаров»

Вкладки «Поставщики», «Счёт-фактуры», «Список товаров» созданы для обычного отображения данных из соответствующих таблиц seller, invoice, goods\_list БД «pharma\_storage», Рисунки 3.8 - 3.10



Рисунок 3.8 – Вкладка «Поставщики»

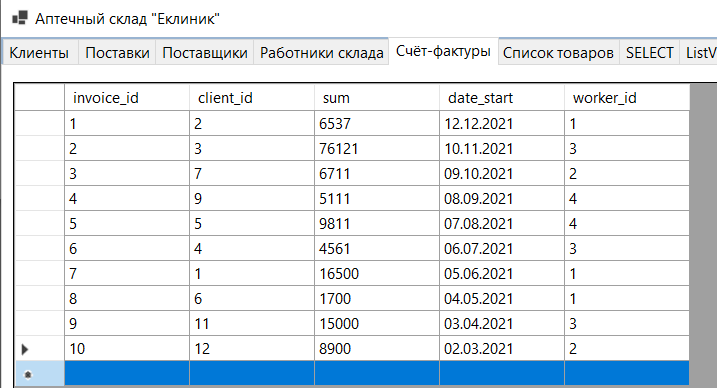


Рисунок 3.9 – Вкладка «Счёт - фактуры»

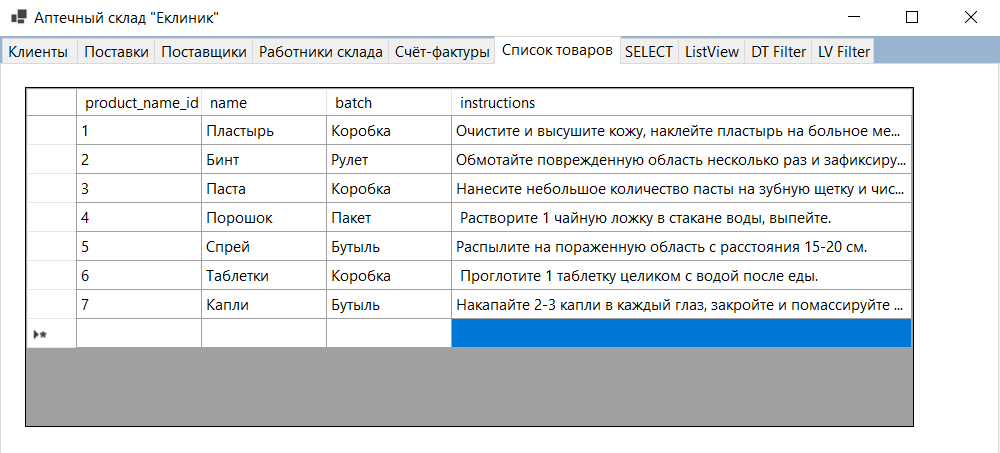


Рисунок 3.10 – Вкладка «Список товаров»

На вкладках присутствует по одному элементу DataGridView соответственно.

* + DataGridView – вывод данных из таблиц seller, invoice, goods\_list

## 3.5 Создание вкладки «Поставки»

Вкладка «Поставки» предназначена для ввода, редактирования и отображения данных о поставках товара, пришедших на склад. Она представлена на рисунке 3.11.

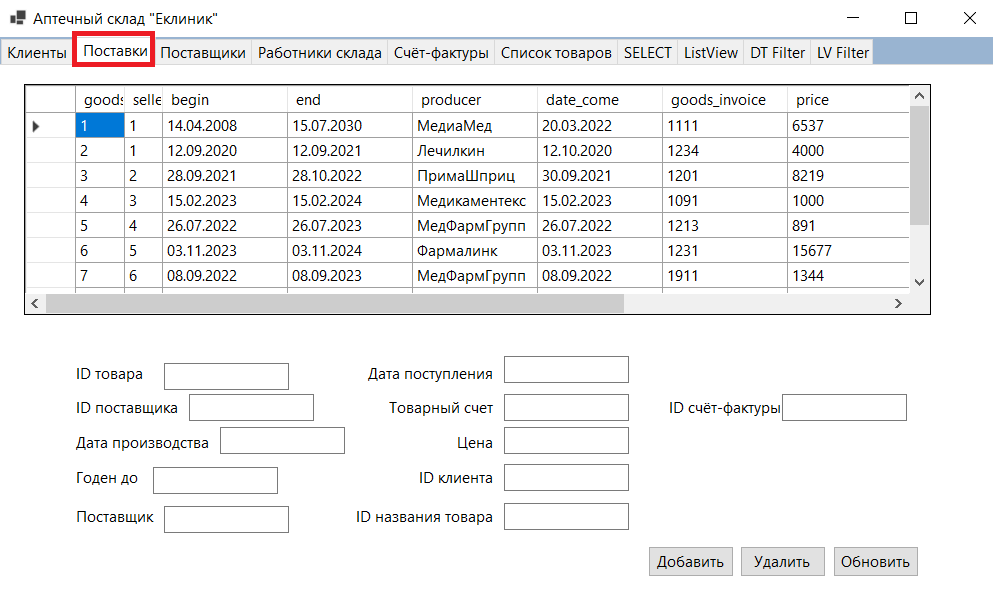


Рисунок 3.11 – Вкладка «Поставки»

На вкладке присутствуют элементы:

* + TextBox – ввод данных о новой поставке;
  + DataGridView – вывод данных из таблицы goods;
  + Кнопка «Добавить» – добавление внесенных значений из полей textbox

в таблицу goods.

* + Кнопка «Обновить» – перезагрузка таблицы;
  + Кнопка «Удалить» – удаление записи из таблицы goods;

Отображение данных таблицы было выведено в отдельный метод, реализующий SQL – запрос select \* from goods. Пример кода для отображения таблицы goods:

private void Select\_goods()

{

try

{

conn.Open();

sql = @"select \* from goods";

cmd = new NpgsqlCommand(sql, conn);

dt = new DataTable();

dt.Load(cmd.ExecuteReader());

conn.Close();

dgvData2.DataSource = null;

dgvData2.DataSource = dt;

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Ошибка" + ex.Message);

}

## }

Для того, чтобы добавить новую поставку необходимо будет заполнить все текстовые поля и нажать на кнопку «Добавить». Обработчик кнопки добавить:

private void button6\_Click(object sender, EventArgs e)

{

string goods\_id = textBoxGoodsId.Text;

string seller\_id = textBoxSellerId.Text;

string begin = textboxBegin.Text;

string end = textBoxEnd.Text;

string producer = textBoxProducer.Text;

string date\_come = textBoxDatecome.Text;

string goods\_invoice = textBoxGoods\_invoice.Text;

string price = textBoxPrice.Text;

string client\_id = textBoxClient\_id.Text;

string product\_name\_id = textBoxProduct\_name\_id.Text;

string invoice\_id = textBoxInvoiceId.Text;

// Формирование SQL-запроса

string query = "INSERT INTO goods (goods\_id,seller\_id, begin, end, producer, date\_come, goods\_invoice, price, client\_id, product\_name\_id, invoice\_id) " +

$"VALUES ('{goods\_id}','{seller\_id}', '{begin}', '{end}', '{producer}','{date\_come}','{goods\_invoice}','{price}','{client\_id}','{product\_name\_id}','{invoice\_id}');";

// Выполнение SQL-запроса

NpgsqlCommand command = new(query, conn);

conn.Open();

int rowsAffected = command.ExecuteNonQuery();

conn.Close();

if (rowsAffected > 0)

{

MessageBox.Show("Данные успешно добавлены в базу данных.");

}

else

{

MessageBox.Show("Не удалось добавить данные в базу данных.");

}

}

Добавление поставки из приложения представлено на Рисунках 3.12 – 3.14

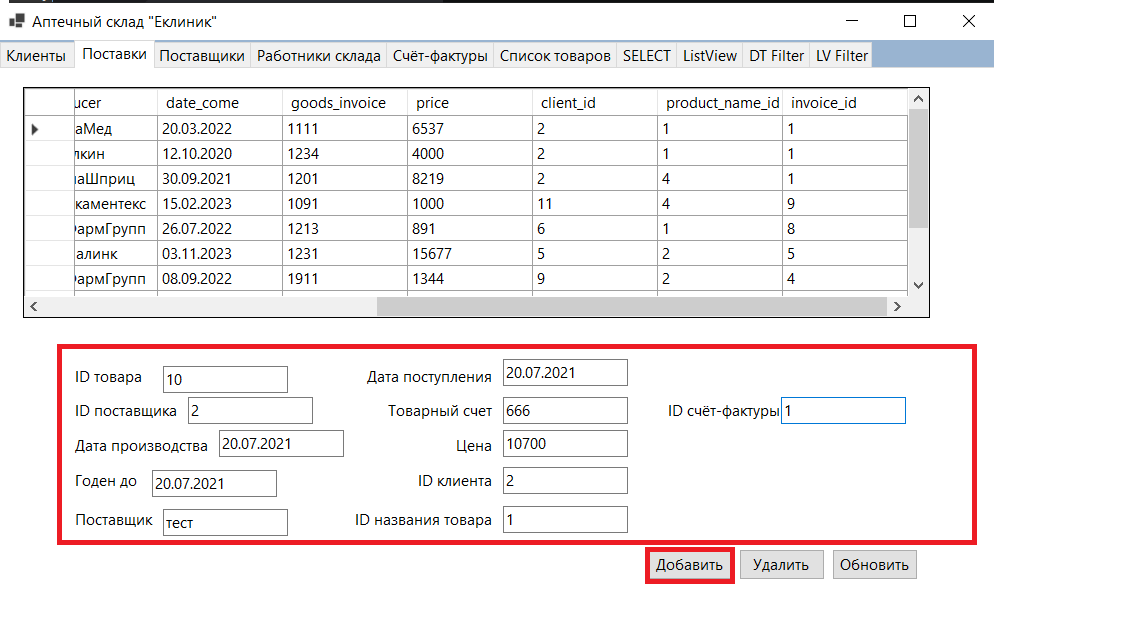


Рисунок 3.12 - Заполнение формы о новой поставке

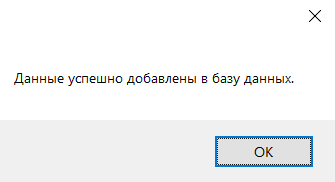


Рисунок 3.13 – Уведомление об успешном добавлении записи

## 

Рисунок 3.14 – Новая запись появилась в таблице

## 3.7 Создание вкладки «Работники склада»

Вкладка «Работники склада» предназначена для добавления, удаления или редактирования информации о работниках аптечного склада «Еклиник». Вкладка продемонстрирована на рисунке 3.15.

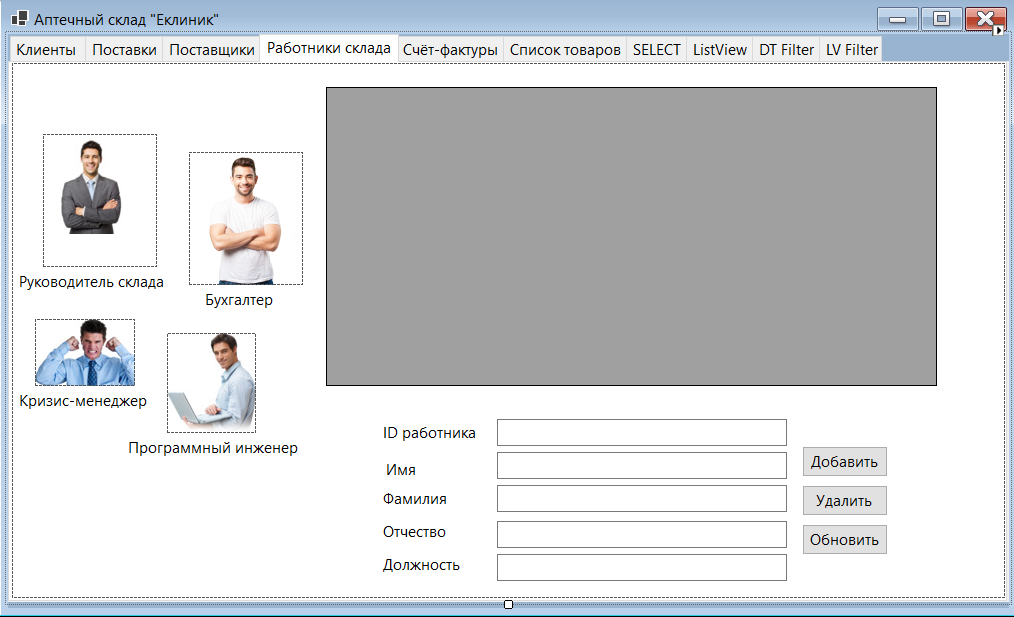


Рисунок 3.15 – Вкладка «Работники склада»

На вкладке представлен набор элементов:

* + DataGridView – вывод данных из таблицы worker;
  + Кнопка «Добавить» – открытие формы «Добавление нового сотрудника». Представлена на рисунке 3.16;
  + Кнопка «Обновить» – перезагрузка списка работников;

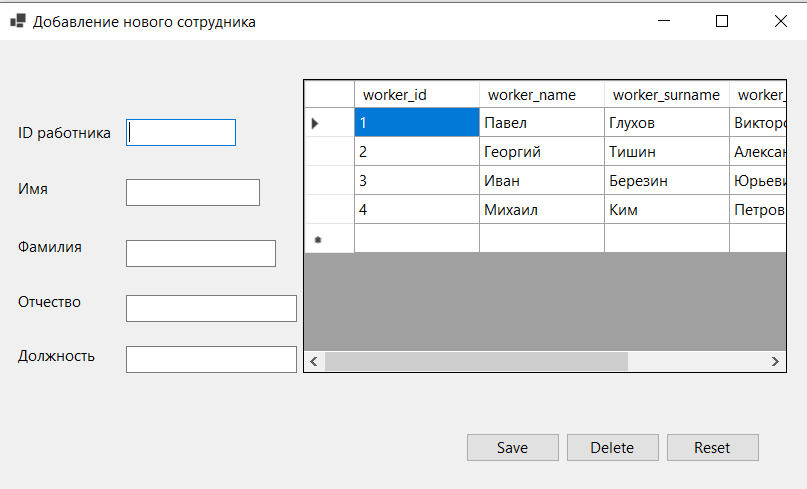


Рисунок 3.16 – Форма «Добавление нового сотрудника»

На вкладке представлен набор элементов:

* + DataGridView – вывод данных из таблицы worker;
  + Кнопка «Save» - сохранение заполненных пользователем полей в БД посредством использования процедуры workeraddoredit4;
  + Кнопка «Обновить» – перезагрузка списка работников;
  + Кнопка «Удалить» – удаление записи из таблицы worker посредством использования процедуры delete\_worker;

Добавление сотрудника в таблицу реализовано с помощью хранимой процедуры workeraddoredit4. SQL-запрос хранимой процедуры:

**BEGIN**

**-- Если существует запись с переданным p\_worker\_id, обновляем ее**

**IF EXISTS (SELECT 1 FROM Worker WHERE worker\_id = p\_worker\_id) THEN**

**UPDATE Worker**

**SET**

**worker\_name = p\_worker\_name,**

**worker\_surname = p\_worker\_surname,**

**worker\_lastname = p\_worker\_lastname,**

**jobposition = p\_jobposition**

**WHERE**

**worker\_id = p\_worker\_id;**

**ELSE**

**-- В противном случае, создаем новую запись**

**INSERT INTO Worker (worker\_id, worker\_name, worker\_surname, worker\_lastname, jobposition)**

**VALUES (p\_worker\_id, p\_worker\_name, p\_worker\_surname, p\_worker\_lastname, p\_jobposition);**

**END IF;**

**END;**

Вызов хранимой процедуры в коде:

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

{

int workerId;

if (!int.TryParse(textBoxID.Text, out workerId))

{

MessageBox.Show("Некорректное значение для ID рабочего.");

return;

}

using (var connection = new NpgsqlConnection(ConnectionString))

{

try

{

connection.Open();

using (var command = new NpgsqlCommand("workeraddoredit4", connection))

{

command.CommandType = CommandType.StoredProcedure;

command.Parameters.AddWithValue("p\_worker\_id", workerId);

command.Parameters.AddWithValue("p\_worker\_name", textBoxName.Text);

command.Parameters.AddWithValue("p\_worker\_surname", textBoxSurname.Text);

command.Parameters.AddWithValue("p\_worker\_lastname", textBoxLastName.Text);

command.Parameters.AddWithValue("p\_jobposition", textBoxJobPosition.Text);

command.ExecuteNonQuery();

}

MessageBox.Show("Рабочий успешно добавлен в таблицу worker.");

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при добавлении рабочего: {ex.Message}");

}

}

}

Удаление сотрудника из таблицы реализовано с помощью хранимой процедуры delete\_worker. SQL-запрос хранимой процедуры:

BEGIN

DELETE FROM Worker WHERE worker\_id = p\_worker\_id;

END;

Вызов хранимой процедуры в коде:

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

{

using (var connection = new NpgsqlConnection(ConnectionString))

{

try

{

connection.Open();

int workerId;

if (!int.TryParse(textBoxID.Text, out workerId))

{

MessageBox.Show("Некорректное значение для ID сотрудника.");

return;

}

using (var command = new NpgsqlCommand("delete\_worker", connection))

{

command.CommandType = CommandType.StoredProcedure;

command.Parameters.AddWithValue("p\_worker\_id", workerId);

command.ExecuteNonQuery();

}

MessageBox.Show("Сотрудник успешно удален из таблицы Worker.");

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при удалении сотрудника: {ex.Message}");

}

}

Добавление и удаление сотрудника из приложения представлены на Рисунках 3.17 – 3.21

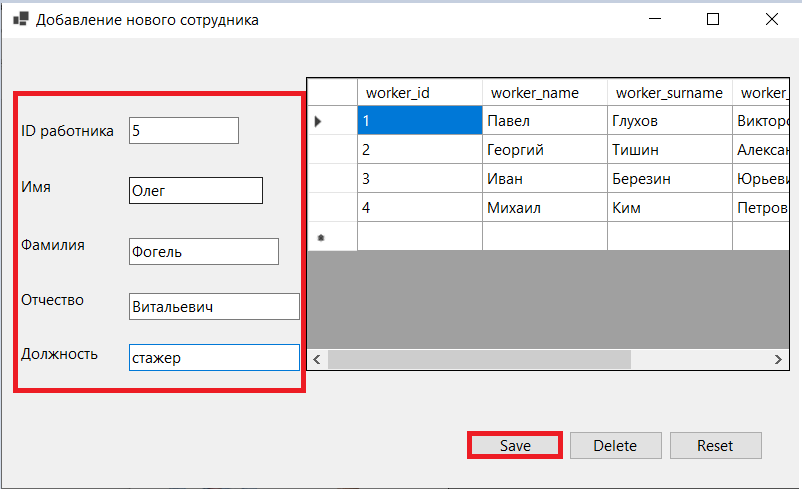


Рисунок 3.17 – Заполнение полей

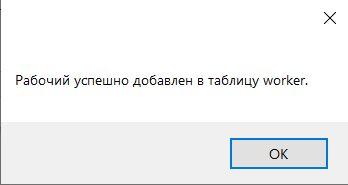


Рисунок 3.18 – Уведомление об успешном добавлении сотрудника

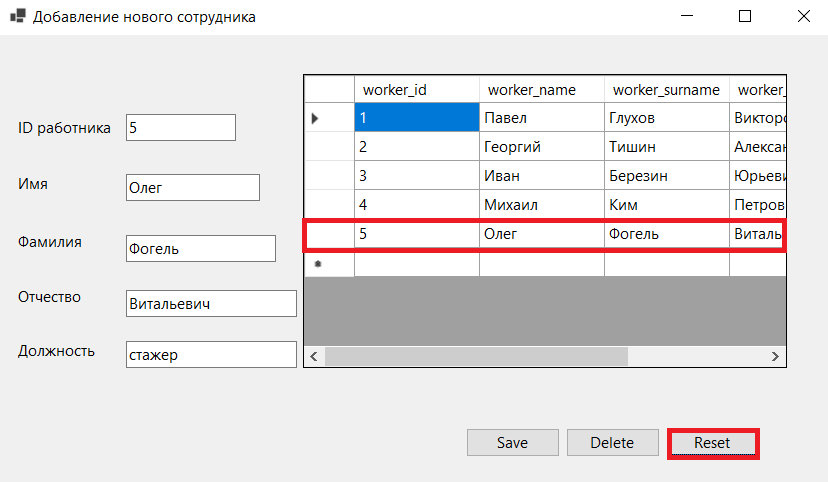


Рисунок 3.20 – Запись о новом сотруднике добавлена в таблицу

Для удаления записи о сотруднике необходимо в форме «Добавление нового сотрудника» в поле «ID сотрудника» указать ID рабочего, которого нужно удалить из БД. Рисунок 3.21 – 3.23

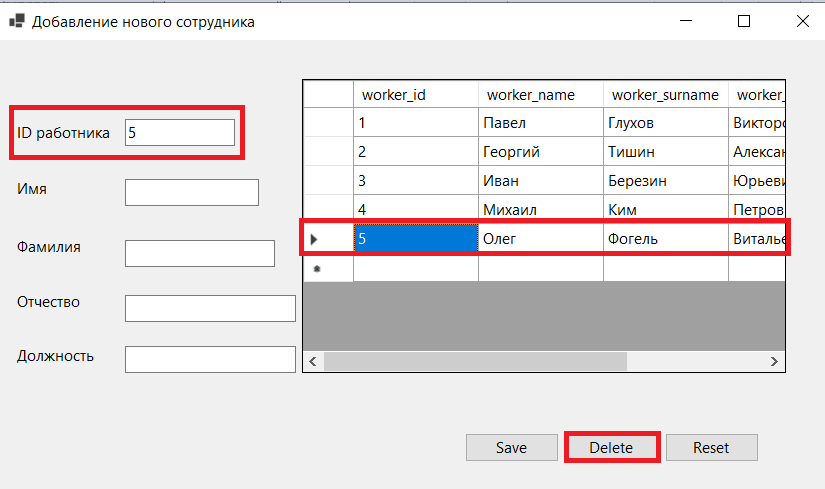


Рисунок 3.21 – Выбор записи для удаления из БД

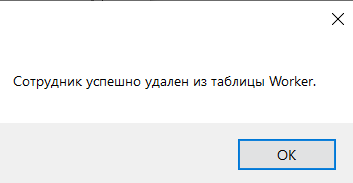


Рисунок 3.22 – Уведомление об успешном удалении сотрудника

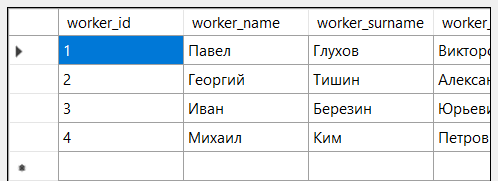


Рисунок 3.23 – Обновление таблицы

# Руководство пользователя

При открытии приложения вас встретит стартовая форма (или же окно) с тремя кнопками. Данное окно можно рассмотреть на рисунке 4.1.

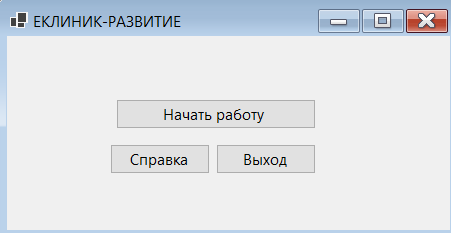


Рисунок 4.1 – Форма «ЕКЛИНИК-РАЗВИТИЕ»

Кнопка «Начать работу» переведёт к главной форме (Рисунок 4.2), откуда можно переходить по вкладкам. Кнопка «Справка» выведет информацию о программе и разработчике (Рисунок 4.3), а «Выйти» - закроет приложение.

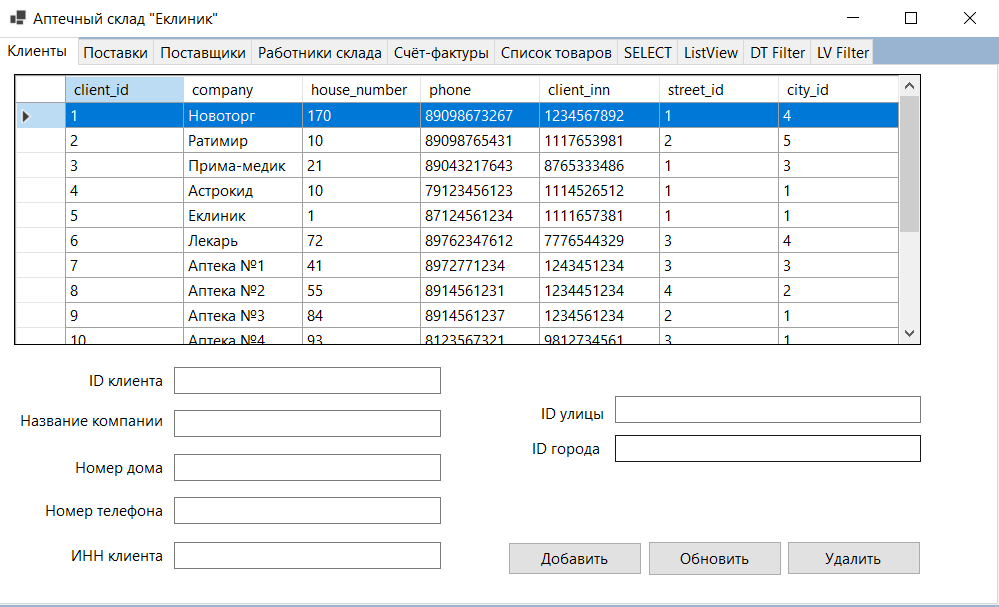


Рисунок 4.2 – Форма «Аптечный склад Еклиник»

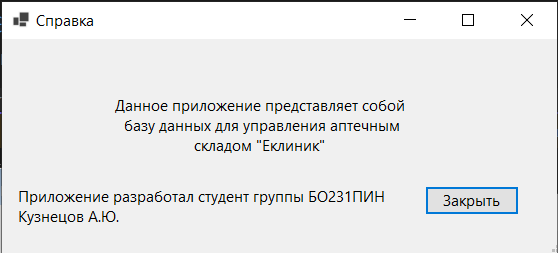


Рисунок 4.3 – Форма «Справка»

На вкладке «Клиенты» можно добавлять, удалять и редактировать информацию о клиентах аптечного склада. Для того, чтобы добавить нового клиента нужно заполнить все поля для ввода данных и нажать на кнопку «Добавить», Рисунок 4.4.

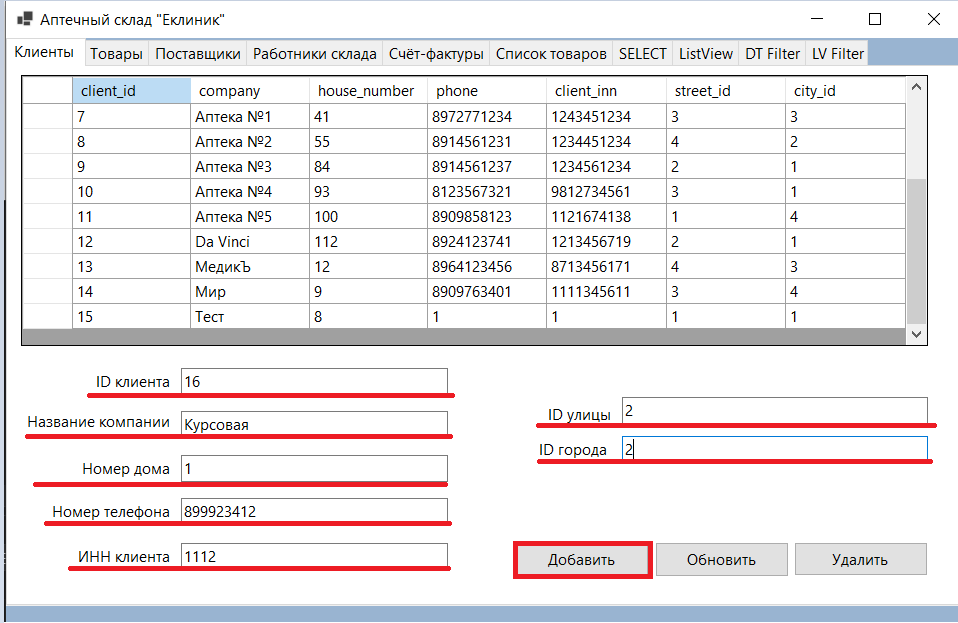


Рисунок 4.4 – Заполнение данных о новом клиенте

При нажатии же на вкладку «Поставки» откроется страница «Поставки», (Рисунок 4.5), на которой можно добавлять информацию о новых поставках и удалять старые поставки. Добавление и удаление записей идентично с вкладкой «Клиенты»

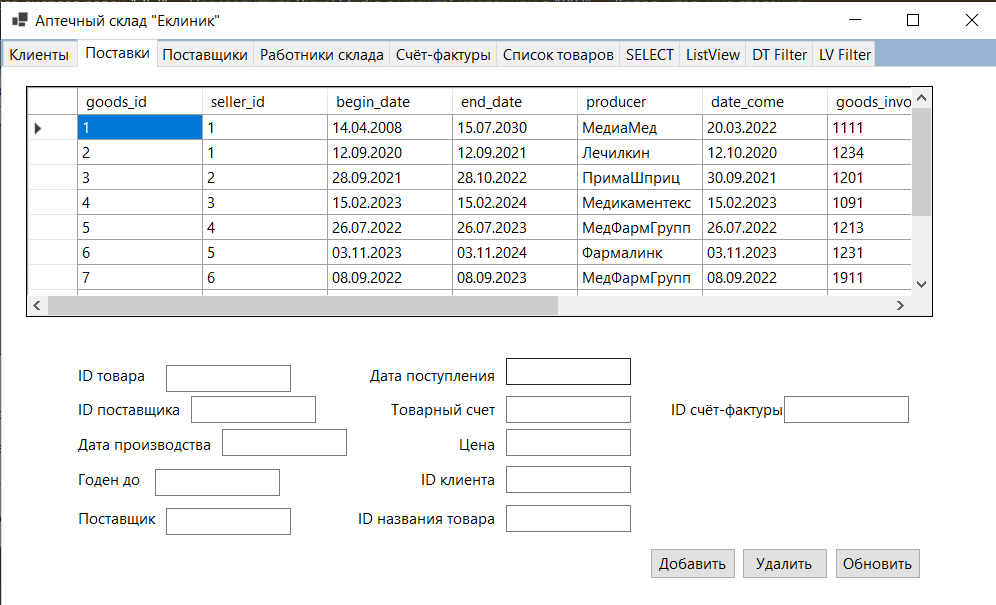


Рисунок 4.5 – Вкладка «Поставки»

На вкладках «Поставщики», «Счёт-фактуры», «Список товаров» представлены на Рисунках 4.6 – 4.8, вкладки предназначены только для просмотра информации, добавлять или удалять данные с них нельзя.

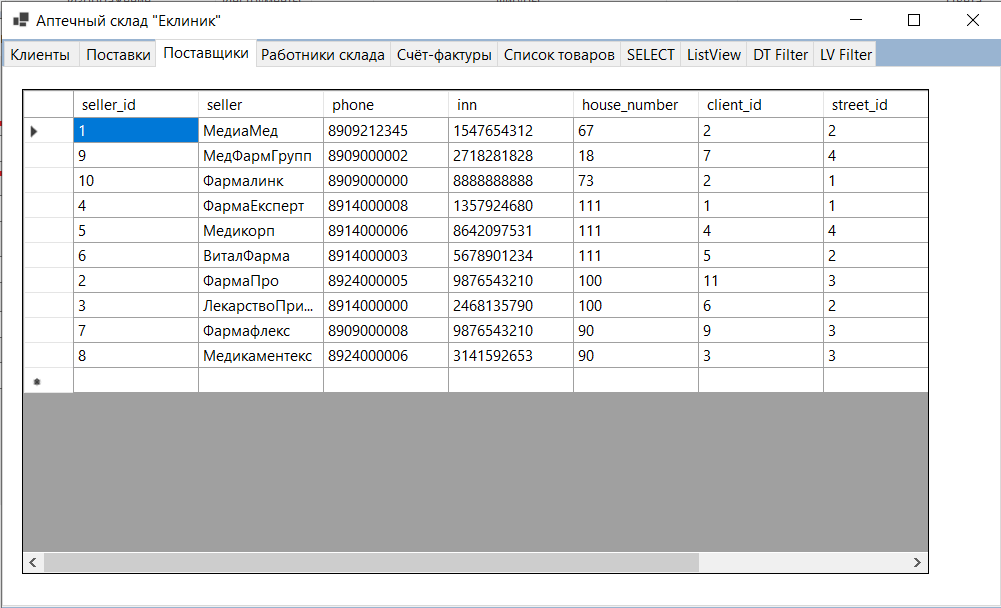


Рисунок 4.6 – Вкладка «Поставщики»

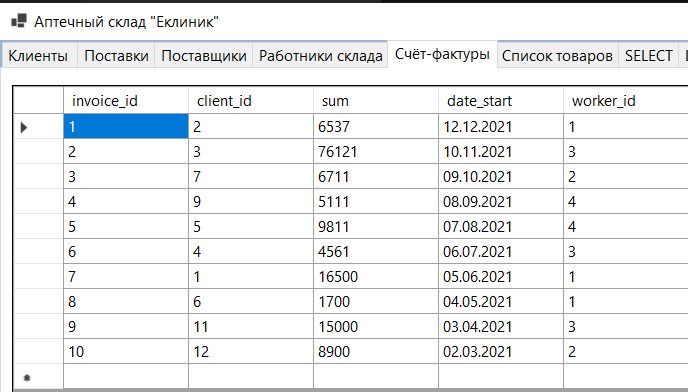


Рисунок 4.7 – Вкладка «Счёт-фактуры»

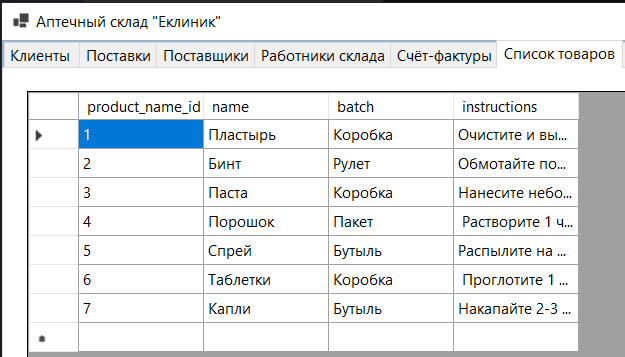


Рисунок 4.8 – Вкладка «Список товаров»

Заключительная вкладка – вкладка «Работники склада» - отображает таблицу с данными о сотрудниках предприятия, которых можно удалять и добавлять в БД, Рисунок 4.9

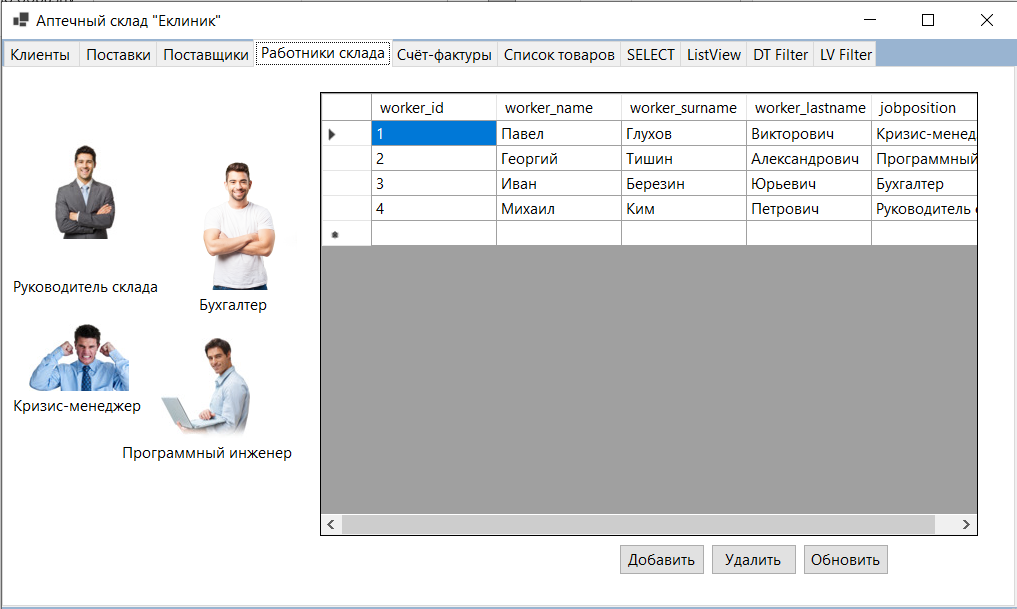


Рисунок 4.9 – Вкладка «Работники склада»

На вкладке присутствует кнопка «Добавить», при нажатии которой откроется форма «Добавление нового сотрудника», Рисунок 4.10.

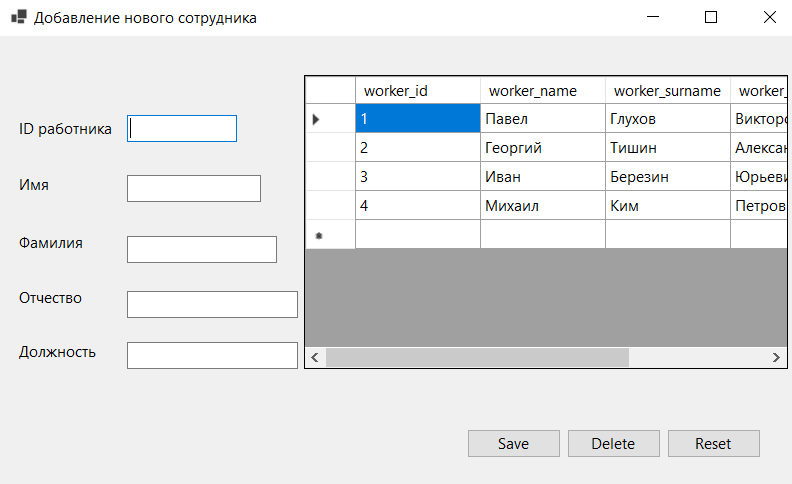


Рисунок 4.10 – Вкладка «Добавление нового сотрудника»

Заключение

В ходе работы над курсовым проектом использовался программный продукт «Microsoft Visual Studio» и «pgAdmin 4». Была создана база данных и клиент-серверное приложение для управления аптечным складом «Еклиник» позволяющее:

* просматривать и редактировать данные о сотрудниках, клиентах, поставках, счет-фактурах;

В результате выполнения курсового проекта была проанализирована предметная область и на её основе реализованы поставленные задачи, а именно:

* был изучен интерфейс «Microsoft Visual Studio» и её возможности;
* были получены знания по созданию баз данных;
* был получен опыт по созданию прикладного программного обеспечения для базы данных;
* был произведён анализ эргономики интерфейса инструментального средства и его возможностей.

Список использованных источников

1. Гурвиц, Г. А. Microsoft Access 2010. Разработка приложений на реальном примере / Г. А. Гурвиц. – Санкт-Петербург: БХВ-Петербург, 2010. – 496 с. : ил. + CD-ROM – (Профессиональное программирование)
2. Смирнов, С.Н. Безопасность систем баз данных / С. Н. Смирнов. – М: Гелиос, 2007. – 352с.