Оглавление

[**1.Проектирование и разработка структуры базы данных 2**](#_Toc466918787)

[**1.1. Постановка задачи 2**](#_Toc466918788)

[**1.2. Описание системы 2**](#_Toc466918789)

[**1.3. Анализ предметной области 2**](#_Toc466918790)

[**1.4. Перечень сокращений 5**](#_Toc466918791)

[**1.5. Создание модели вариантов использования. 5**](#_Toc466918792)

[**1.6. Инфологическая модель базы данных 8**](#_Toc466918793)

[**2. Программная реализация 12**](#_Toc466918794)

[**2.1. SQL – запросы 12**](#_Toc466918795)

[**2.2. Код программы 13**](#_Toc466918796)

[**2.3. Оконные формы приложения 15**](#_Toc466918797)

[**Заключение 19**](#_Toc466918798)

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 21**](#_Toc466918799)

[**Приложение 1. 22**](#_Toc466918800)

# 1.Проектирование и разработка структуры базы данных

* 1. **Постановка задачи**

Необходимо разработать модель распределенной реляционной базы данных предприятия, состоящего из главного производства и нескольких удаленных филиалов. Части предприятия связаны компьютерной сетью, имеют собственные локальные базы данных, могут использовать данные удаленных узлов. Глобальная информация хранится на центральном узле.

* 1. **Описание системы**

Из задания следует, что база данных в нашей работе будет не одна. Будет основная (глобальная) база данных, которая будет содержать общую информацию, к которой должны иметь доступ все подразделения и все пользователи. И несколько удаленных (локальных) баз данных, который будет хранить дополнительную информацию, необходимую для конкретного филиала предприятия.

* 1. **Анализ предметной области**

Система предназначена для работников и посетителей сети аптек. Благодаря этой системе, работники могут быстрее выполнять необходимые действия, система позволяет сотрудникам отсортировать, отфильтровать и сгруппировать уже имеющиеся данные. Система сделает процесс добавления новых данных проще и быстрее. Автоматизированная информационная система позволяет производить поиск по базе данных гораздо быстрее, чем, если это база хранилась в бумажном виде, и поиск производился бы вручную. Также использование автоматизированной системы контролировать целостность и корректность вводимых данных, что приводит к повышению качества работы сети аптек.

Для работы с информационной системой «Сеть аптек» необходим Microsoft Access иVisual Studio

Требование к Microsoft Access.

|  |  |
| --- | --- |
| КОМПОНЕНТ | ТРЕБОВАНИЯ |
| Компьютер и процессор | Процессор с тактовой частотой 1 гигагерц (ГГц) или выше (x86) либо 64-разрядный процессор с набором инструкций SSE2 |
| Оперативная память | 1 ГБ ОЗУ (для 32-разрядной версии) или 2 ГБ ОЗУ (для 64-разрядной версии) |
| Жесткий диск | 3,0 ГБ свободного места на жестком диске |
| Дисплей | Разрешение экрана 1366 x 768 |
| Графика | Для использования аппаратного ускорения графики необходима графическая карта с поддержкой DirectX 10. |
| Операционная система | Windows 7, Windows 8, Windows Server 2008 R2 или Windows Server 2012 |
| Браузер | [Поддерживаемые браузеры для Office 2013 и Office 365](http://office.microsoft.com/ru-ru/products/redir/HA102789344.aspx) |
| Другие | Для использования Интернета в Access 2013 необходимо подключение к Интернету (может быть платным). |

Вывод:

Возможности СУБД Access и Visual Studio позволяют реализовать ИС «Сеть аптек» на ее основе без привлечения других программных продуктов

Программа Access требует минимальные ресурсы компьютера. Программный продукт можно переносить на различных источниках (кроме дискет, которые практически уже нигде не используются), при этом программа будет работать на любом компьютере с установленным офисным пакетом Access. После внедрения возможны изменения, доработки. Это все зависит от пожеланий заказчика.

Цели:

* Хранить информацию
* Ускорить процесс оформления документации
* Ускорить процесс поиска информации
* Ускорить качество и скорость обслуживания
* Ускорить процесс приема и обработки данных

**Дополнительные особенности проектируемой системы:**

*Функциональные возможности:*

* Система должна обеспечивать многопользовательский режим работы.
* Если сервер недоступен в момент выполнения запроса, программа должна выполнить поиск хотя бы по локальной базе данных.
* Программа должна иметь возможность работать в демонстрационном режиме (не регистрированные пользователи могут посмотреть лекарства, и найти в какой из аптек есть это лекарство).

*Удобство использования:*

* Пользовательский интерфейс должен быть Windows-совместимым.
* Изучение интерфейса должно занимать не более 2 минут.

*Надежность.*

* Сервер для программы должен быть в работоспособном состоянии 24 часа в день 7 дней в неделю.
* Программа должна проверять успешность доставки сообщения с данными на сервер.

*Производительность.*

• Сервер должен поддерживать до 100 одновременных подключений клиентов.

*Безопасность.*

* Только «Администратор» может редактировать информацию в глобальной базе данных.
* Незарегистрированные пользователи не могут вносить никаких изменений.

*Удобство поддержки (сопровождение).*

* Наличие документации.
* Читабельный, комментированный код программы, для упрощения модификации программы.
* Создание ежедневных отчетов об ошибках, для того, чтобы быстро находить и исправлять ошибки в работе системы.
  1. **Перечень сокращений**

SQL – язык структурированных запросов.

SQL запрос – команда на языке SQL, производящая некоторое изменение данных в базе данных.

АИС - автоматизированная информационная система.

СУБД – система управления базами данных.

БД - база данных.

OC- операционная система.

* 1. **Создание модели вариантов использования.**

Для начала нужно выделить объекты системы:

* Глобальная база данных;
* Локальная база данных;
* Администратор;
* Гость;
* Продавец.

Опишем задачи, которые выполняет каждый объект.

Объект «Глобальная база данных»:

* Хранит общие данные о лекарственных средствах, цены и описания;
* Хранит информацию о филиалах предприятия;
* Обрабатывает запросы на изменение информации о лекарственных средствах;
* Осуществляет поиск лекарств по запросу покупателей.

Объект «Локальная база данных»:

* Хранит частную информацию о конкретном филиале (расположение, информация о сотрудниках, графики работы и дежурств);
* Хранит дополнительную информацию о медикаментах (расположение конкретного медикамента на складе аптеки);
* Обрабатывает изменение количества товара на складе при продаже и получении товара;
* Осуществляет поиск лекарств по запросу продавца.

Объект «Администратор»:

* Авторизируется при входе в систему;
* Может изменять информацию о лекарственных средствах в глобальной базе данных;
* Создает выборки по различным условиям (например, отчет по продаже товаров в каком-либо из филиалов)
* Поддерживает работоспособность системы.

Объект «Продавец»:

* Авторизируется при входе в систему
* Продает товар и, соответственно, вносит изменения в локальную базу данных;
* Принимает товар и добавляет его в базу данных;
* Ищет нужный товар на складе, для того, чтобы продать его покупателю.

Объект «Гость»:

* Просматривает списки товаров;
* Ищет нужный медикамент по различным параметрам;
* Покупает товар.

На основе этого можно составить модель взаимодействия объектов в системе. Модель взаимодействия представлена на рисунке 1.

Untitled Diagram.png

Рисунок 1. Модель взаимодействия объектов

* 1. **Инфологическая модель базы данных**

Для построения инфологической модели необходимо выделить сущности, которые будут созданы в базе данных.

Базы данных у нас две, поэтому и инфологические модели будет тоже две. Одна для глобальной базы данных, вторая для локальных баз данных, которые будут одинаковыми по структуре, но разными по содержанию.

Сущности глобальной базы данных:

* Лекарства (описание, цена);
* Пользователи;
* Филиалы;
* Продажи.

Сущности локальной базы данных:

* Лекарства (количество, расположение);
* Сотрудники;
* График работы;
* График дежурств;
* Необходимые к закупке товары.

**Теперь для каждой сущности выделим атрибуты.**

Сущность «Лекарства»:

* Номер лекарства;
* Название;
* Производитель;
* Описание;
* Категория;
* Форм-фактор;
* Объем;
* Цена.

Сущность «Пользователи»:

* Номер пользователя;
* Логин;
* Пароль;
* Личные данные;
* Группа.

Сущность «Филиалы»:

* Номер филиала;
* Название;
* Адрес.

Сущность «Продажи»:

* Номер товара;
* Количество;
* Дата и время продажи;
* Номер филиала;
* Номер продавца.

Сущность «Лекарства»:

* Номер лекарства;
* Количество товара на складе;
* Расположение товара на складе.

Сущность «Сотрудники»:

* Номер сотрудника;
* Фамилия;
* Имя;
* Зарплата;
* Дополнительная информация.

Сущность «График работы»:

* Дата;
* Время начала работы;
* Время окончания;
* Время обеда.

Сущность «График дежурств»:

* Дата;
* Время дежурства;
* Номер сотрудника.

Сущность «Товары необходимые к закупке»:

* Номер лекарства;
* Необходимое количество;
* Дата составления заявки.

Далее представим инфологические модели. Они представлены на рисунках 2 и 3 (для глобальной и локальной базы данных).

2.png

Рисунок 2. Инфологическая модель глобальной базы данных

3.png

Рисунок 3. Инфологическая модель локальной базы данных.

4.png

Рисунок 4. Инфологическая модель всей системы

1. **Программная реализация**
   1. **SQL – запросы**
      1. Запрос на добавление данных о продажах:

*INSERT INTO Продажи*

*([Номер товара], [Количество], [Дата], [Время], [Номер филиала], [Номер продавца] )*

*Values*

*(idProduct,selled, 'getDate()', 'getTime()',numberBranche,numberSellers);*

* + 1. Запрос на изменение количество товара на складе:

*Update Лекарства*

*set [Количество товара на складе] = newCount*

*where [Номер лекарства] =idProduct;*

* + 1. Запрос для измение количество товара и его расположение на складе:

*Update Лекарства*

*set [Количество товара на складе] = newCount,*

*[Расположение товара на складе] = 'textBox9.Text'*

*where [Номер лекарства] = idProduct;*

* + 1. Поиск лекарства по номеру:

*SELECT Лекарства.[Количество товара на складе], Лекарства.[Расположение товара на складе]*

*FROM Лекарства*

*where Лекарства.[Номер лекарства] = id;*

* + 1. Запрос для создания отчета по лекарствам и их продажам:

*SELECT Лекарства.Название,SUM (Продажи.Количество),*

*SUM (Продажи.Количество\*Лекарства.Цена)*

*FROM Лекарства*

*INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]*

*GROUP BY Лекарства.Название*

*ORDER BY SUM (Продажи.Количество\*Лекарства.Цена) DESC;*

* + 1. Запрос для создания отчета по продажам в филиалах:

*SELECT Филиалы.Код, Филиалы.Название, SUM(Продажи.Количество), SUM(Лекарства.Цена\*Продажи.Количество)*

*FROM Филиалы*

*INNER JOIN (Лекарства*

*INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]*

*) ON Филиалы.[Код] = Продажи.[Номер продавца]*

*GROUP BY Филиалы.Код, Филиалы.Название*

*ORDER BY SUM(Лекарства.Цена\*Продажи.Количество) DESC;*

* + 1. Запрос для создания отчета по продажам по продавцам:

*SELECT Продажи.[Номер продавца],Продажи.[Номер филиала],*

*SUM(Продажи.Количество),*

*SUM(Продажи.Количество\*Лекарства.Цена)*

*FROM Лекарства*

*INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]*

*GROUP BY Продажи.[Номер филиала], Продажи.[Номер продавца]*

*ORDER BY SUM(Продажи.Количество\*Лекарства.Цена) DESC;*

* + 1. Запрос для получения списка необходимых закупок:

*Select \* From Закупки;*

* + 1. Запрос для получения списка пользователей:

*Select \* From Пользователи;*

* + 1. Запрос для удаления лекарства

*Delete from Лекарства where Код =id;*

* 1. **Код программы**

Для взаимодействия с базой данных необходимо в программе подключиться к базе данных из программного приложения. Для этого в языке программирования существует тип OleDbConnection. Из названия типа следует, что он предназначен для связи с базами данных с типом интерфейса OleDb. Базы данных MS Access работают именно по этому принципу. При работе с другими базами данных, например MySQL, используются другие коннекторы. Для MySQL – MySQLConnector.

Создание объекта коннектора выглядит следующим образом:

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

В команде указывается тип базы данных и ее адрес.

После инициализации объекта происходит подключение. Для этого используется команда open():

conn.Open();

После подключения к базе данных можно производить необходимые действия. Например, выполнять какие-либо SQL-запросы.

Для этого нужно создать объекта типа OleDbCommand. В команде мы указываем наш запрос, и коннектор с помощью которого мы подключены к базе данных:

OleDbCommand cmd = new OleDbCommand("Select \* From Пользователи", conn);

SQL-запросы бывают двух типов. Результатом одних является изменение данных в базе данных, результатом других – получение пользователей некоторой выборки.

В зависимости от типа запроса отличает и модель поведения. Если мы выполняем запрос, который изменяет данные в таблице, используется команда ExecuteNonQuery():

cmd.ExecuteNonQuery();

Если мы выполняем запрос, результатом которого является некоторая выборка, то используем команду ExecuteReader():

cmd.ExecuteReader();

Так как данная команда возвращает данные, необходимо куда-то их сохранять, для этого используется объект типа OleDbDataReader. Он считывает построчно при помощи команды Read() результат выполнения запроса:

using (OleDbDataReader dr = cmd.ExecuteReader()) //выполняем

while (dr.Read()){ //читаем результат

Далее требуется извлечь эти данные, для этого используется команда GetValue():

\_login = dr.GetValue(1).ToString();

После выполнения всех действий с базой данных, необходимо закрыть подключение к ней:

conn.Close();

* 1. **Оконные формы приложения**
     1. **Окно авторизации**

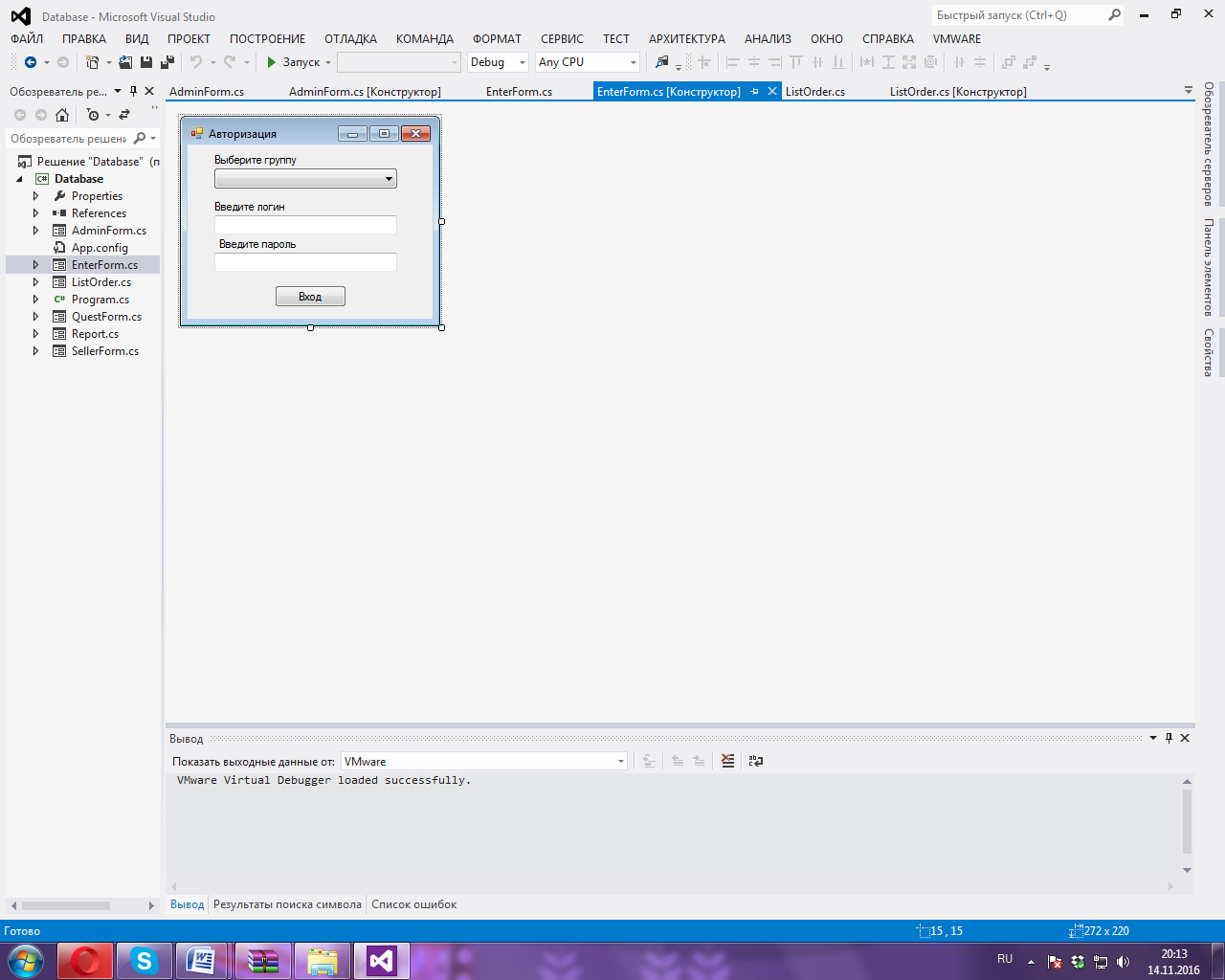


Рисунок 5.Окно авторизации

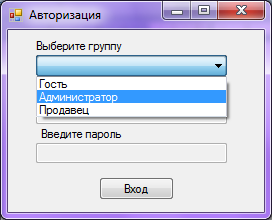
****

Рисунок 6. Выбор группы

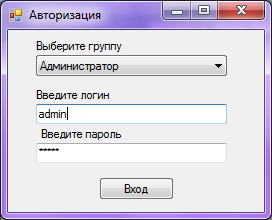


Рисунок 7.Ввод данных

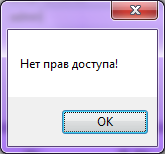
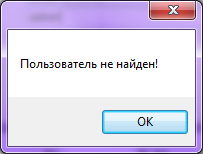
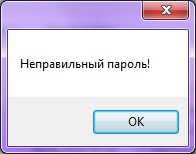


Рисунок 8. Обработка ошибок

* + 1. **Окно администратора**

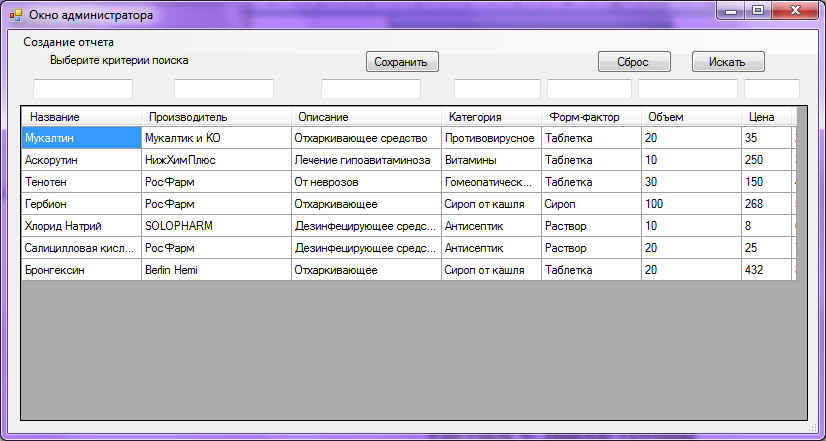


Рисунок 9. Окно администратора

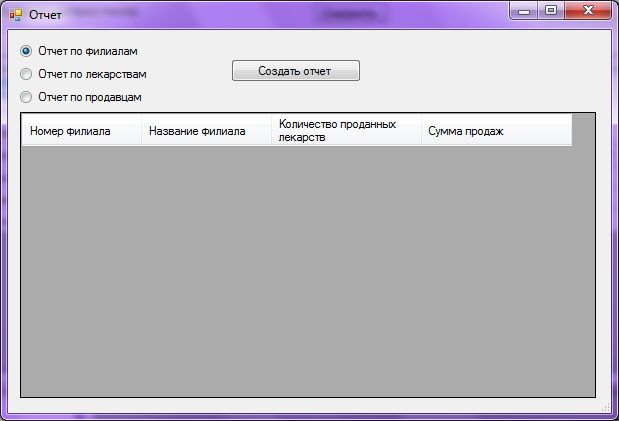
****

Рисунок 10. Создание отчетов

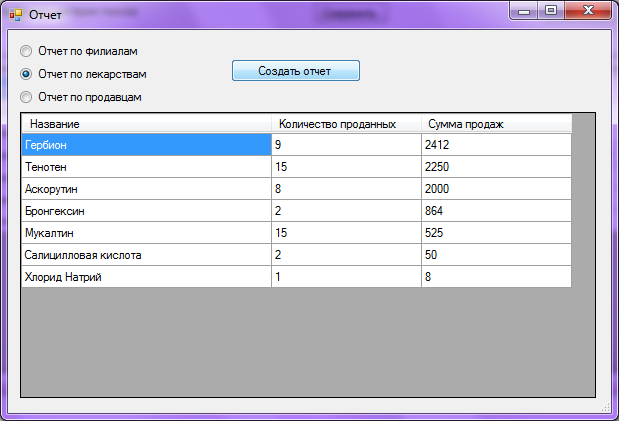
****

Рисунок 11. Отчет

* 1. **Окно гостя**

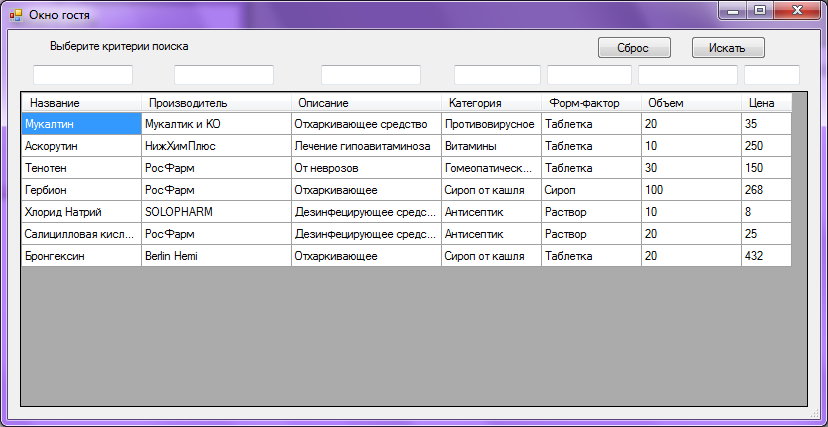


Рисунок 12. Окно гостя

* 1. **Окно продавца**

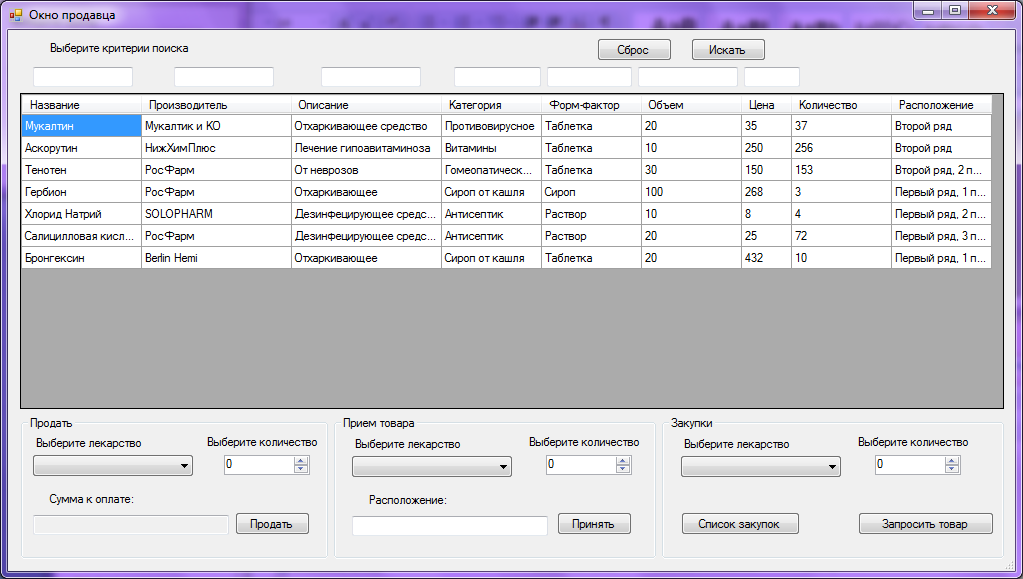
****

Рисунок 13. Окно продавца

# Заключение

В результате работы над курсовым проектом была создана ИС «Распределенная реляционная база данных сети аптек». По созданию интерфейса поставленная цель была достигнута, результатом получился яркий, графический интерфейс для удобного поиска информации и пользования программой.

Созданный программный продукт полностью удовлетворяет потребности заказчика:

* Ведение базы данных;
* Создание отчетов по работе аптек;
* Поиск необходимых лекарств.

Заказчик программного продукта получает полный доступ к исходным данным. В информационной системе «Распределенная реляционная база данных сети аптек» в режиме просмотра возможно выполнение следующих действий:

- Поиск лекарств;

- Просмотр информации;

- Создание отчетов.

В режиме редактирования возможно выполнение следующих действий:

- Редактировать лекарства;

- Удаления лекарства;

- Продавать лекарства;

- Перемещать лекарства;

- Принимать лекарства.

Программа требует минимальные ресурсы компьютера. Программный продукт можно переносить на различных источниках (кроме дискет, которые практически уже нигде не используются), при этом программа будет работать на любом компьютере с установленным офисным пакетом Access и C#. После внедрения возможны изменения, доработки. Это все зависит от пожеланий заказчика.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Гончаров А.Ю. - Access 2003. Самоучитель с примерами. [www.natahaus.ru](http://www.natahaus.ru)
2. Симонович С.В., Евсеев Г.А., Алексеев А.Г. Специальная информатика. Учебное пособие. – М.: АСТ-Пресс, 1998.- 480 с
3. Дейт К. Дж. Введение в системы баз данных. — К.; М.; СПб.: Издательский дом «Вильямс», 1999. — 848 с.
4. Праг К. Н., Ирвин М. Р. Access 2000. Библия пользователя. — М.: Вильямс, 2000. — 1040 с.
5. Microsoft Access 2003. Русская версия. Шаг за шагом. – М.: ЭКОМ, 2006
6. Хансен Г., Хансен Дж. Базы данных: разработка и управление. — М.: БИНОМ, 1999. — 704 с.
7. Ковязин А. С. Востриков. Мир InterBase. М.:КУДИЦ –ОБРАЗ, 2006-  496 с.
8. Орлов В.В.Технологии разработки программных продуктов– СПб.: Питер, 2003 -437 с.
9. Рудаков А.В. Технология разработки программных продуктов – М.: Издательский центр «Академия», 2005-208 с.
10. Рудаков А.В, Федорова Г.Н. Технология разработки программных продуктов. Практикум: учеб. Пособие для студ. учреждений сред.проф. образования – М.: Издательский центр «Академия»; 2010. – 192 с.

# Приложение 1.

Код программы.

*Окно авторизации.*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class EnterForm : Form

{

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

int numberSellers;

public EnterForm()

{

InitializeComponent();

}

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

{

if (comboBox1.SelectedIndex == 0){

textBox1.Enabled = false;

textBox3.Enabled = false;

}

else{

textBox3.Enabled = true;

textBox1.Enabled = true;

}

}

int successLogin(string login, string password, int type){

string \_login, \_password;

int number, group;

OleDbCommand cmd = new OleDbCommand("Select \* From Пользователи", conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader()){ //выполняем

while (dr.Read()){ //читаем результат

\_login = dr.GetValue(1).ToString();

\_password = dr.GetValue(2).ToString();

group = System.Convert.ToInt32(dr.GetValue(4).ToString());

number = System.Convert.ToInt32(dr.GetValue(5).ToString());

numberSellers = System.Convert.ToInt32(dr.GetValue(6).ToString());

if (login == \_login){

if (type == group)

if (password == \_password)

return number;

else{

MessageBox.Show("Неправильный пароль!");

return -1;

}

else{

MessageBox.Show("Нет прав доступа!");

return -1;

}

}

}

}

MessageBox.Show("Пользователь не найден!");

return -1;

}

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

if (textBox1.Text != "" || comboBox1.SelectedIndex == 0){

if (textBox3.Text != "" || comboBox1.SelectedIndex == 0){

if (comboBox1.SelectedIndex == 0){

conn.Close();

QuestForm quest = new QuestForm();

quest.ShowDialog();

}

if (comboBox1.SelectedIndex == 1){

if (successLogin(textBox1.Text, textBox3.Text, 0) >= 0){

conn.Close();

AdminForm admin = new AdminForm();

admin.AdminLogin = textBox1.Text;

admin.ShowDialog();

}

}

if (comboBox1.SelectedIndex == 2){

int branch = successLogin(textBox1.Text, textBox3.Text, 1);

if (branch > 0){

conn.Close();

SellerForm seller = new SellerForm();

seller.sellerLogin = textBox1.Text;

seller.numberBranche = branch;

seller.numberSellers = numberSellers;

seller.ShowDialog();

}

}

}

else

MessageBox.Show("Введите пароль!");

}

else

MessageBox.Show("Введите логин!");

}

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

{

conn.Open();

}

}

}

*Окно администратора:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class AdminForm : Form

{

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

public string AdminLogin;

int rows;

public AdminForm()

{

InitializeComponent();

}

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

{

dataGridView1.ContextMenuStrip = contextMenuStrip1;

load(true);

}

string getCommand(bool type)

{

bool check = false;

string command = "Select \* From Лекарства";

if (type)

return command;

command += " where ";

if (textBox1.Text != "")

{

command += " Название = '" + textBox1.Text + "'";

check = true;

}

if (textBox2.Text != "")

{

if (check)

command += " and ";

command += " Производитель = '" + textBox2.Text + "'";

check = true;

}

if (textBox3.Text != "")

{

if (check)

command += " and ";

command += " Описание = '" + textBox3.Text + "'";

check = true;

}

if (textBox4.Text != "")

{

if (check)

command += " and ";

command += " Категория = '" + textBox4.Text + "'";

check = true;

}

if (textBox5.Text != "")

{

if (check)

command += " and ";

command += " Форм-фактор = '" + textBox5.Text + "'";

check = true;

}

if (textBox6.Text != "")

{

if (check)

command += " and ";

command += " Объем = " + textBox6.Text;

check = true;

}

if (textBox7.Text != "")

command += "and Цена = " + textBox7.Text;

return command;

}

void load(bool type)

{

int i = 0;

dataGridView1.Rows.Clear();

conn.Open();

OleDbCommand cmd = new OleDbCommand(getCommand(type), conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView1.Rows.Add();

dataGridView1[0, i].Value = dr.GetValue(1).ToString();

dataGridView1[1, i].Value = dr.GetValue(2).ToString();

dataGridView1[2, i].Value = dr.GetValue(3).ToString();

dataGridView1[3, i].Value = dr.GetValue(4).ToString();

dataGridView1[4, i].Value = dr.GetValue(5).ToString();

dataGridView1[5, i].Value = dr.GetValue(6).ToString();

dataGridView1[6, i].Value = dr.GetValue(7).ToString();

dataGridView1[7, i].Value = dr.GetValue(0).ToString();

dataGridView1.Rows[i].ReadOnly = true;

i++;

}

}

conn.Close();

}

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

{

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

textBox7.Text = "";

}

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

{

if (textBox1.Text == "" && textBox2.Text == "" && textBox3.Text == "" && textBox4.Text == "" && textBox5.Text == "" && textBox6.Text == "" && textBox7.Text == "")

load(true);

else

load(false);

}

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

{

if (MessageBox.Show("Действительно хотите удалить?", "Предупреждение", MessageBoxButtons.OKCancel, MessageBoxIcon.Stop) == DialogResult.OK)

{

conn.Open();

OleDbCommand cmd = new OleDbCommand("Delete from Лекарства where Код = " + dataGridView1[7, rows].Value.ToString(), conn); //создаем команду

cmd.ExecuteNonQuery();

conn.Close();

}

load(true);

}

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

{

rows = e.RowIndex;

}

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

{

if (MessageBox.Show("Действительно хотите сохранить?", "Предупреждение", MessageBoxButtons.OKCancel, MessageBoxIcon.Stop) == DialogResult.OK)

{

string command = "Update Лекарства set ";

command += " Название = '" + dataGridView1[0, rows].Value.ToString() + "',";

command += " Производитель = '" + dataGridView1[1, rows].Value.ToString() + "',";

command += " Описание = '" + dataGridView1[2, rows].Value.ToString() + "',";

command += " Категория = '" + dataGridView1[3, rows].Value.ToString() + "',";

command += " [Форм-фактор] = '" + dataGridView1[4, rows].Value.ToString() + "',";

command += " Объем = " + dataGridView1[5, rows].Value.ToString()+",";

command += " Цена = " + dataGridView1[6, rows].Value.ToString();

command += " where Код = " + dataGridView1[7, rows].Value.ToString() +";";

conn.Open();

OleDbCommand cmd = new OleDbCommand(command, conn); //создаем команду

cmd.ExecuteNonQuery();

conn.Close();

}

}

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

{

dataGridView1.Rows[rows].ReadOnly = false;

}

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

{

Report newReport = new Report();

newReport.ShowDialog();

}

}

}

*Создание отчетов:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class Report : Form

{

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

public Report()

{

InitializeComponent();

}

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

{

radioButton1.Checked = true;

}

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

{

int i = 0;

string command;

conn.Open();

if (radioButton2.Checked)

{

dataGridView1.Visible = false;

dataGridView2.Visible = true;

dataGridView3.Visible = false;

command = "SELECT Лекарства.Название,SUM (Продажи.Количество), SUM (Продажи.Количество\*Лекарства.Цена) " +

" FROM Лекарства INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]" +

" GROUP BY Лекарства.Название" +

" ORDER BY SUM (Продажи.Количество\*Лекарства.Цена) DESC;";

dataGridView2.Rows.Clear();

OleDbCommand cmd = new OleDbCommand(command, conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView2.Rows.Add();

dataGridView2[0, i].Value = dr.GetValue(0).ToString();

dataGridView2[1, i].Value = dr.GetValue(1).ToString();

dataGridView2[2, i].Value = dr.GetValue(2).ToString();

dataGridView2.Rows[i].ReadOnly = true;

i++;

}

}

}

else

if (radioButton1.Checked)

{

dataGridView1.Visible = true;

dataGridView2.Visible = false;

dataGridView3.Visible = false;

command = "SELECT Филиалы.Код, Филиалы.Название, SUM(Продажи.Количество), SUM(Лекарства.Цена\*Продажи.Количество)"+

" FROM Филиалы INNER JOIN (Лекарства INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]) ON Филиалы.[Код] = Продажи.[Номер продавца]"+

" GROUP BY Филиалы.Код, Филиалы.Название"+

" ORDER BY SUM(Лекарства.Цена\*Продажи.Количество) DESC;";

dataGridView1.Rows.Clear();

OleDbCommand cmd = new OleDbCommand(command, conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView1.Rows.Add();

dataGridView1[0, i].Value = dr.GetValue(0).ToString();

dataGridView1[1, i].Value = dr.GetValue(1).ToString();

dataGridView1[2, i].Value = dr.GetValue(2).ToString();

dataGridView1[3, i].Value = dr.GetValue(3).ToString();

dataGridView1.Rows[i].ReadOnly = true;

i++;

}

}

}

else

{

dataGridView1.Visible = false;

dataGridView2.Visible = false;

dataGridView3.Visible = true;

command = "SELECT Продажи.[Номер продавца],Продажи.[Номер филиала], SUM(Продажи.Количество), SUM(Продажи.Количество\*Лекарства.Цена)"+

" FROM Лекарства INNER JOIN Продажи ON Лекарства.[Код] = Продажи.[Номер товара]"+

" GROUP BY Продажи.[Номер филиала], Продажи.[Номер продавца]"+

" ORDER BY SUM(Продажи.Количество\*Лекарства.Цена) DESC;";

dataGridView3.Rows.Clear();

OleDbCommand cmd = new OleDbCommand(command, conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView3.Rows.Add();

dataGridView3[0, i].Value = dr.GetValue(0).ToString();

dataGridView3[1, i].Value = dr.GetValue(1).ToString();

dataGridView3[2, i].Value = dr.GetValue(2).ToString();

dataGridView3[3, i].Value = dr.GetValue(3).ToString();

dataGridView3.Rows[i].ReadOnly = true;

i++;

}

}

}

conn.Close();

}

}

}

*Окно гостя:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class QuestForm : Form

{

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

public QuestForm()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

string getCommand(bool type)

{

bool check = false;

string command = "Select \* From Лекарства";

if (type)

return command;

command += " where ";

if (textBox1.Text != "")

{

command += " Название = '" + textBox1.Text + "'";

check = true;

}

if (textBox2.Text != "")

{

if (check)

command += " and ";

command += " Производитель = '" + textBox2.Text + "'";

check = true;

}

if (textBox3.Text != "")

{

if (check)

command += " and ";

command += " Описание = '" + textBox3.Text + "'";

check = true;

}

if (textBox4.Text != "")

{

if (check)

command += " and ";

command += " Категория = '" + textBox4.Text + "'";

check = true;

}

if (textBox5.Text != "")

{

if (check)

command += " and ";

command += " Форм-фактор = '" + textBox5.Text + "'";

check = true;

}

if (textBox6.Text != "")

{

if (check)

command += " and ";

command += " Объем = " + textBox6.Text;

check = true;

}

if (textBox7.Text != "")

command += "and Цена = " + textBox7.Text;

return command;

}

void load(bool type)

{

int i=0;

dataGridView1.Rows.Clear();

conn.Open();

OleDbCommand cmd = new OleDbCommand(getCommand(type), conn); //создаем команду

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView1.Rows.Add();

dataGridView1[0,i].Value = dr.GetValue(1).ToString();

dataGridView1[1, i].Value = dr.GetValue(2).ToString();

dataGridView1[2, i].Value = dr.GetValue(3).ToString();

dataGridView1[3, i].Value = dr.GetValue(4).ToString();

dataGridView1[4, i].Value = dr.GetValue(5).ToString();

dataGridView1[5, i].Value = dr.GetValue(6).ToString();

dataGridView1[6, i].Value = dr.GetValue(7).ToString();

dataGridView1.Rows[i].ReadOnly = true;

i++;

}

}

conn.Close();

}

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

{

load(true);

}

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

{

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

textBox7.Text = "";

}

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

{

if (textBox1.Text == "" && textBox2.Text == "" && textBox3.Text == "" && textBox4.Text == "" && textBox5.Text == "" && textBox6.Text == "" && textBox7.Text == "")

load(true);

else

load(false);

}

}

}

*Окно продавца:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class SellerForm : Form

{

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

OleDbConnection connLocal;

public string sellerLogin;

public int numberBranche;

public int numberSellers;

public SellerForm()

{

InitializeComponent();

}

string getCommand(bool type)

{

bool check = false;

string command = "Select \* From Лекарства";

if (type)

return command;

command += " where ";

if (textBox1.Text != "")

{

command += " Название = '" + textBox1.Text + "'";

check = true;

}

if (textBox2.Text != "")

{

if (check)

command += " and ";

command += " Производитель = '" + textBox2.Text + "'";

check = true;

}

if (textBox3.Text != "")

{

if (check)

command += " and ";

command += " Описание = '" + textBox3.Text + "'";

check = true;

}

if (textBox4.Text != "")

{

if (check)

command += " and ";

command += " Категория = '" + textBox4.Text + "'";

check = true;

}

if (textBox5.Text != "")

{

if (check)

command += " and ";

command += " Форм-фактор = '" + textBox5.Text + "'";

check = true;

}

if (textBox6.Text != "")

{

if (check)

command += " and ";

command += " Объем = " + textBox6.Text;

check = true;

}

if (textBox7.Text != "")

command += "and Цена = " + textBox7.Text;

return command;

}

string getCommandById(string id)

{

string command = "SELECT Лекарства.[Количество товара на складе], Лекарства.[Расположение товара на складе] "+

" FROM Лекарства "+

"where Лекарства.[Номер лекарства] = " + id+";";

return command;

}

void load(bool type)

{

int i = 0;

dataGridView1.Rows.Clear();

conn.Open();

connLocal.Open();

OleDbCommand cmd = new OleDbCommand(getCommand(type), conn); //создаем команду

OleDbCommand cmd2;

OleDbDataReader dr2;

comboBox2.Items.Clear();

comboBox1.Items.Clear();

comboBox3.Items.Clear();

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView1.Rows.Add();

comboBox1.Items.Add(dr.GetValue(1).ToString());

comboBox2.Items.Add(dr.GetValue(1).ToString());

comboBox3.Items.Add(dr.GetValue(1).ToString());

dataGridView1[0, i].Value = dr.GetValue(1).ToString();

dataGridView1[1, i].Value = dr.GetValue(2).ToString();

dataGridView1[2, i].Value = dr.GetValue(3).ToString();

dataGridView1[3, i].Value = dr.GetValue(4).ToString();

dataGridView1[4, i].Value = dr.GetValue(5).ToString();

dataGridView1[5, i].Value = dr.GetValue(6).ToString();

dataGridView1[6, i].Value = dr.GetValue(7).ToString();

dataGridView1[7, i].Value = dr.GetValue(0).ToString();

cmd2 = new OleDbCommand(getCommandById(dr.GetValue(0).ToString()),connLocal);

dr2 = cmd2.ExecuteReader();

dr2.Read();

dataGridView1[8, i].Value = dr2.GetValue(0).ToString();

dataGridView1[9, i].Value = dr2.GetValue(1).ToString();

dataGridView1.Rows[i].ReadOnly = true;

i++;

}

}

conn.Close();

connLocal.Close();

}

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

{

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

textBox7.Text = "";

}

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

{

if (textBox1.Text == "" && textBox2.Text == "" && textBox3.Text == "" && textBox4.Text == "" && textBox5.Text == "" && textBox6.Text == "" && textBox7.Text == "")

load(true);

else

load(false);

}

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

{

connLocal= new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=localBase" + numberBranche + ".accdb");

load(true);

}

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

{

numericUpDown1.Maximum = System.Convert.ToDecimal(dataGridView1[8,comboBox1.SelectedIndex].Value);

}

private void numericUpDown1\_ValueChanged(object sender, EventArgs e)

{

textBox8.Text = (System.Convert.ToInt32(numericUpDown1.Value) \* System.Convert.ToInt32(dataGridView1[6, comboBox1.SelectedIndex].Value)).ToString();

}

string getDate()

{

return DateTime.Now.Day + "." + DateTime.Now.Month + "." + DateTime.Now.Year;

}

string getTime()

{

return DateTime.Now.Hour + ":" + DateTime.Now.Minute;

}

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

{

string idProduct = dataGridView1[7, comboBox1.SelectedIndex].Value.ToString();

int selled = System.Convert.ToInt32(numericUpDown1.Value);

int newCount = System.Convert.ToInt32(numericUpDown1.Maximum) - selled;

conn.Open();

connLocal.Open();

string commandSale = "INSERT INTO Продажи ([Номер товара], [Количество], [Дата], [Время], [Номер филиала], [Номер продавца] ) " +

" Values ("+ idProduct+ ", "+ selled+",'"+getDate()+"','"+getTime()+"',"+numberBranche+","+numberSellers+");";

string updateProduct = "Update Лекарства set [Количество товара на складе] = " + newCount + " where [Номер лекарства] = " + idProduct + ";";

OleDbCommand cmd = new OleDbCommand(commandSale, conn); //создаем команду

OleDbCommand cmd2 = new OleDbCommand(updateProduct,connLocal);

cmd.ExecuteNonQuery();

cmd2.ExecuteNonQuery();

connLocal.Close();

conn.Close();

load(true);

}

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

{

string idProduct = dataGridView1[7, comboBox2.SelectedIndex].Value.ToString();

int count = System.Convert.ToInt32(dataGridView1[8, comboBox2.SelectedIndex].Value.ToString());

int newCount = System.Convert.ToInt32(numericUpDown2.Value) + count;

connLocal.Open();

string updateProduct = "Update Лекарства set [Количество товара на складе] = " + newCount + ",[Расположение товара на складе] = '"+textBox9.Text+"' where [Номер лекарства] = " + idProduct + ";";

OleDbCommand cmd2 = new OleDbCommand(updateProduct, connLocal);

cmd2.ExecuteNonQuery();

connLocal.Close();

load(true);

}

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

{

string idProduct = dataGridView1[7, comboBox3.SelectedIndex].Value.ToString();

int newCount = System.Convert.ToInt32(numericUpDown3.Value);

connLocal.Open();

string updateProduct = "Insert into [Закупки] ( [Номер лекарства],[Необходимое количество], [Дата составления]) Values ("+idProduct+","+newCount+",'"+getDate()+"')";

OleDbCommand cmd2 = new OleDbCommand(updateProduct, connLocal);

cmd2.ExecuteNonQuery();

connLocal.Close();

numericUpDown3.Value = 0;

comboBox3.SelectedIndex = 0;

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

textBox9.Text = dataGridView1[9, comboBox2.SelectedIndex].Value.ToString();

}

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

{

ListOrder list = new ListOrder();

list.numberBranch = numberBranche;

list.ShowDialog();

}

}

}

*Окно заявок:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Database

{

public partial class ListOrder : Form

{

OleDbConnection connLocal;

OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=mainBase.accdb");

public int numberBranch;

public ListOrder(){

InitializeComponent();

}

private void List\_Load(object sender, EventArgs e){

connLocal = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=localBase" + numberBranch + ".accdb");

int i = 0;

dataGridView1.Rows.Clear();

connLocal.Open();

conn.Open();

OleDbCommand cmd = new OleDbCommand("Select \* From Закупки", connLocal); //создаем команду

OleDbCommand cmd2;

OleDbDataReader dr2;

using (OleDbDataReader dr = cmd.ExecuteReader())

{ //выполняем

while (dr.Read())

{ //читаем результат

dataGridView1.Rows.Add();

dataGridView1[0, i].Value = dr.GetValue(1).ToString();

dataGridView1[1, i].Value = dr.GetValue(2).ToString();

dataGridView1[2, i].Value = dr.GetValue(3).ToString();

cmd2 = new OleDbCommand("Select Название From Лекарства where Код = " + dataGridView1[0, i].Value.ToString(), conn);

dr2 = cmd2.ExecuteReader();

dr2.Read();

dataGridView1[0, i].Value = dr2.GetValue(0);

dataGridView1.Rows[i].ReadOnly = true;

i++;

}

}

conn.Close();

connLocal.Close();

}

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

{

dataGridView1.Rows.Clear();

connLocal.Open();

OleDbCommand cmd = new OleDbCommand("Delete FROM Закупки", connLocal); //создаем команду

cmd.ExecuteNonQuery();

connLocal.Close();

}

}

}