МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждения высшего образования

«ДАГЕСТАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ, ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И ЭНЕРГЕТИКИ

Кафедра «ПОВТиАС»

Курсовая работа

по программированию на тему: “Факультеты ”

вариант - 21

Выполнил:

студент 2-го курса

группы У233

Агарагимов Абзайдин Магомедрасулович

Проверил:

Пиняскин Владимир Викторович

Махачкала 2023

Введение

Информационная система, занимающаяся автоматическим оформление, отправкой и отслеживанием отпуска различных изделий в значительной мере повысило бы производительность труда, сократило бы количество затрачиваемого времени, бумаги, человеческих ресурсов.

То есть то, для чего ещё 15-20 лет назад требовалось 5-10 человек, поддерживающих постоянную связь через телефон, факсы или электронную почту, что конечно же гораздо быстрее чем какая-нибудь почта, телеграф и так далее, как было лет 50 назад, но всё же это требует больше времени чем одно приложение с понятным “UI/UX” дизайном, единой базой данных и прочими преимуществами современных приложений.

В связи с этим было решено разработать такое программное обеспечение. Оно должно будет облегчить оформление заказов как для заказчиков, так и для отправителей. Облегчить ведение учёта заказов, то есть будет в доступно увидеть, что и как купил ты или у тебя, твоей компании и сделать необходимые выводы.

Данная программа должна будет обрабатывать заказы, сделанные заказчиком и обработанные отправителем, и предоставлять по введённой информации отчёт по совершённым операциям.

Назначение и применение

Программа отпуска изделий предназначена для контроля и учета процесса отпуска товаров/изделий в компаниях различных отраслей. Область применения программы включает:

1. Торговые компании: Программа позволяет контролировать остатки товаров на складе, отслеживать движение товаров при их отпуске, автоматически формировать накладные и счета-фактуры для отгрузки товаров покупателям.

2. Производственные предприятия: Программа позволяет отслеживать и контролировать выпуск продукции со склада на производство, автоматически списывать изделия с учетом их количества и стоимости, формировать отчеты о потребности в материалах и комплектующих изделиях.

3. Складские комплексы: Программа помогает управлять и контролировать складские запасы, отслеживать движение товаров при отпуске и приеме, формировать отчеты о доступных запасах и потребностях в дополнительных поставках.

4. Розничные магазины: Программа позволяет вести учет товаров на прилавках, контролировать и отслеживать их продажи и перемещение между различными точками продажи, автоматически формировать отчеты о востребованности товаров и потребности в дополнительных поставках.

В целом, программа отпуска изделий может быть использована в любой компании, где требуется эффективный контроль и учет процесса отпуска товаров/изделий и ведение оперативного учета складских запасов.

2. Постановка задачи

2.1 Функциональные требования к программе

Информационная система должна предоставлять доступ к инструментам, позволяющим добавлять, просматривать, редактировать, а также удалять записи в таблицах базы данных для сотрудников. Помимо этого, программа должна выдать отчёт о проделанных операциях. Ниже перечислены основные функции приложения:

1. Возможность создания профиля сотрудника и других необходимых данных.

2. Возможность редактирования или удаления созданной заявки на отпуск до её одобрения руководством.

3. Возможность одобрения или отклонения заявки на отпуск руководством.

4. Возможность просмотра календаря отпусков, чтобы видеть, кто и когда находится в отпуске.

5. Возможность генерирования отчётов о предоставленных отпусках по различным параметрам (например, по департаментам, отделам, годам).

6. Возможность сохранения и архивирования заявок на отпуск для последующей проверки или аудита.

7. Возможность ведения и отображения истории отпусков каждого сотрудника.

8. Возможность обеспечения безопасности данных, включая защиту персональной информации сотрудников.

**Доступные разновидности отчётов:**

- для заданного получателя: дата отпуска, изделие, идентификатор склада;

- для заданного года отчетности: идентификатор склада, изделие, цена, количество;

- для заданного изделия: дата отпуска, номер накладной, ФИО получателя и заказчика, адрес, получателя.

2.2 Нефункциональные требования к программе

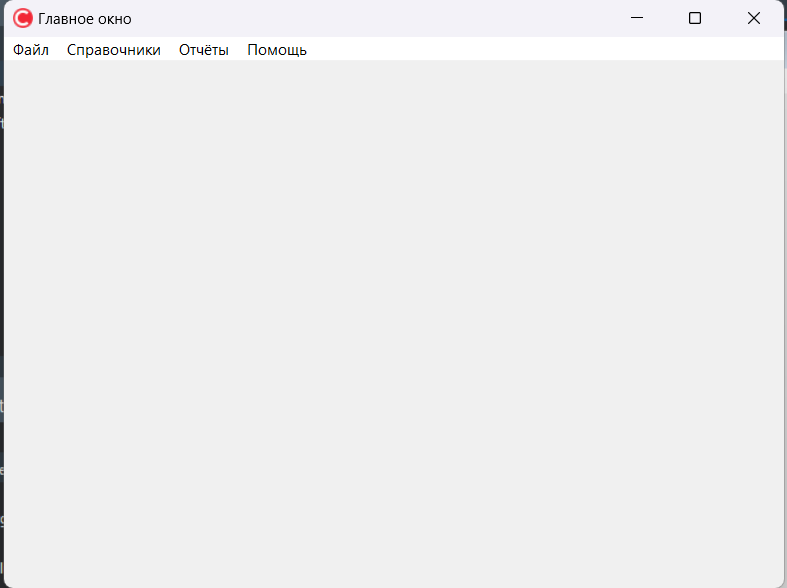


Рис. 2 Графическое отображение главной формы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Файл** | **Отчет** | **Ведомости** | **Справка** | **Выход** |
| **Открыть** | **По получателям** | **Сводная ведомость по получателям** | **О программе** |  |
| **Сохранить** | **По изделиям** |  |  |  |
| **Печать** |  | **Сводная ведомость по складам** |  |  |
| **Выход** |  |  |  |  |

Рис .3. Структура главного меню

# 3.Предполагаемый анализ материально-технической базы

Для работы сотруднику фирмы с программным обеспечением понадобится компьютер.

**Минимальные системные требования**

ОС:Windows 7

Процессор:Intel Pentium Duo IV 2.4 GHz

Память:1 Gb

Видеокарта:NVIDIA GeForce 6600 GT / ATI Radeon 9800 с 512 Mb памяти

Места на диске: 2 Gb

Устройства ввода: мышь, клавиатура, сканер

Устройства вывода: монитор, принтер

Программное приложение: Microsoft Office 2007

**Минимальные системные требования**

Процессор: двухъядерный (Core duo 2 или лучше), Intel Core2Duo @ 1.8ГГц для Windows 7,8

ОЗУ: 1 Гб ( Windows 7,8)

Видео-карта: 256 Мб c 3Д-ускорением и поддержкой DirectX® 9.0c и Shader Model 3.0

Место на диске: около 5 Гб

Устройства ввода: мышь, клавиатура, сканер

Устройства вывода: монитор, принтер

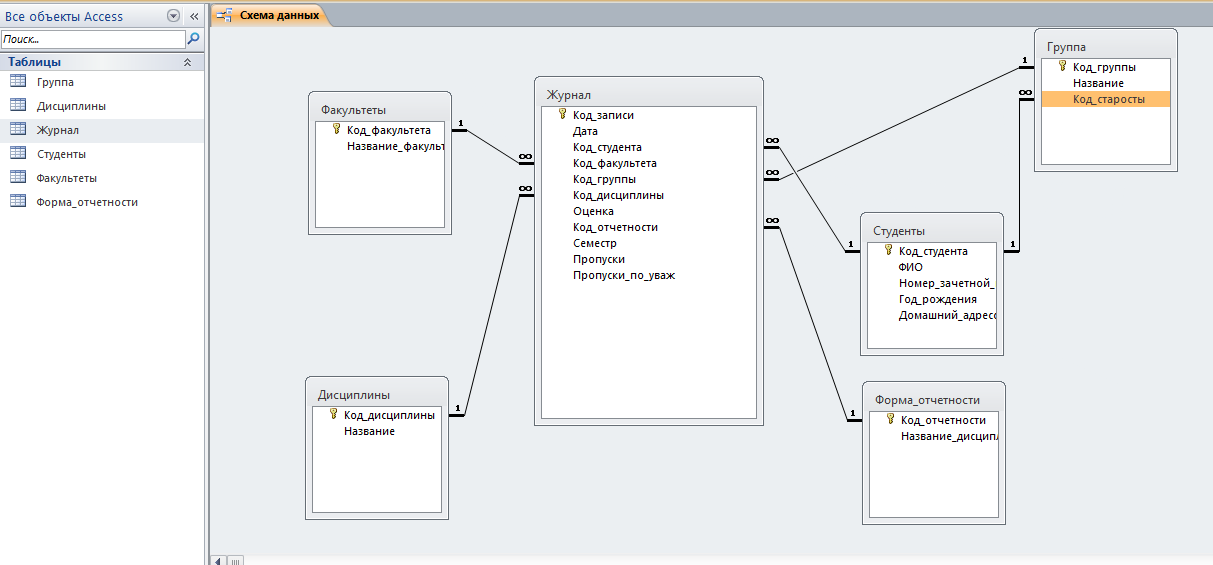
Операционные системы: Windows 7, 8

Microsoft Word (часто — MS Word, WinWord или просто Word) — текстовый процессор, предназначенный для создания, просмотра и редактирования текстовых документов, с локальным применением простейших форм таблично-матричных алгоритмов. Выпускается корпорацией Microsoft в составе пакета Microsoft Office. Первая версия была написана Ричардом Броди (Richard Brodie) для IBM PC, использующих DOS, в 1983 году. Позднее выпускались версии для Apple Macintosh (1984), SCO UNIX и Microsoft Windows (1989). Текущей версией является Microsoft Office Word 2013 для Windows и Microsoft Office Word 2011 для Mac.

Microsoft Excel – программа для работы с электронными таблицами, созданная корпорацией Microsoft для Microsoft Windows, Windows NT и Mac OS. Она предоставляет возможности экономико-статистических расчетов и графические инструменты.

Microsoft Access— это, прежде всего, система управления базами данных (СУБД).

4. Разработка приложения в СУБД MS Access

4.1 Схема данных  Рис.6 Схема данных.

# 4.2 Запросы отчётов

# Запрос отчёта по заказчику работает следующим образом:

# Вводится имя, купившего товар через ИС;

1. На экран подаётся информация в виде: ФИО заказчика, дата регистрации заказа, наименование товара, идентификатор склада, идентификатор операции.

Режим SQL

SELECT recipient.ФИО, journal.registration\_date, product.title, journal.warehouse\_id, journal.id FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((recipient.ФИО)= idx));

Где idx – переменная, хранящая введённое значение.

Результат запроса:

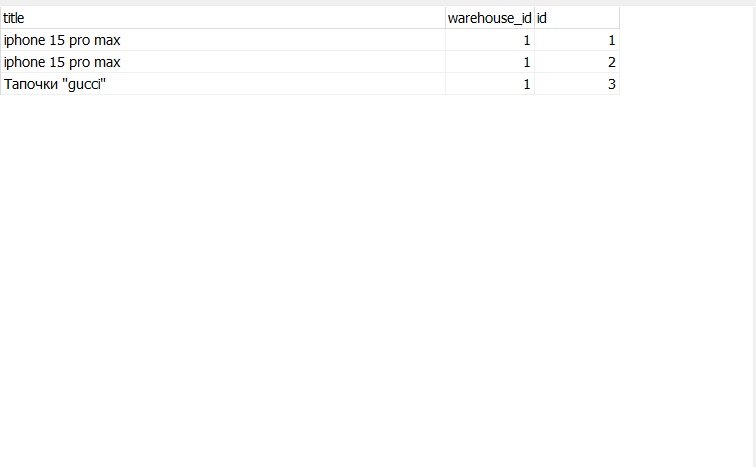
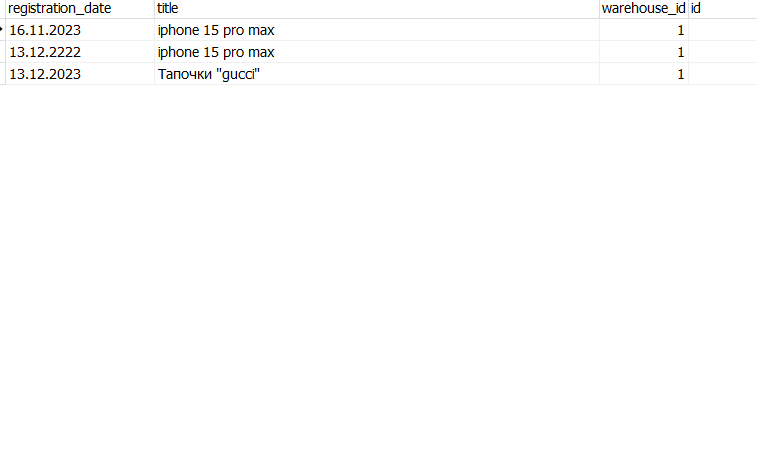
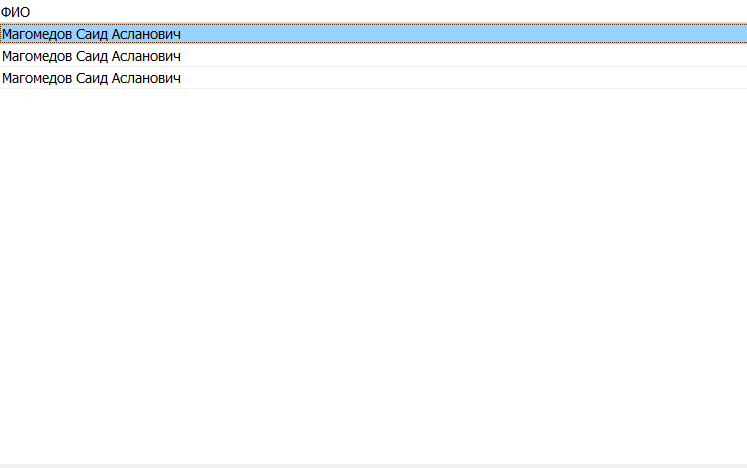
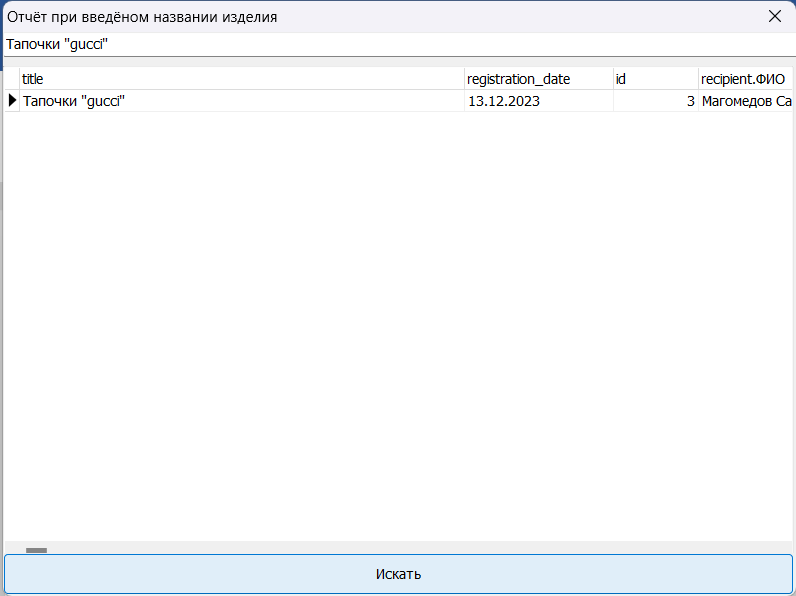


Рис.3, 4 и 5 Результат запроса

Поиск записей по названию изделия и дате совершения покупки.

Рис.6 Конструктор таблицы «Показ журнала по изделиям»

Режим SQL

SELECT product.title, journal.registration\_date, journal.id, recipient.ФИО, recipient.address, sender.ФИО FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((product.title)=idx));

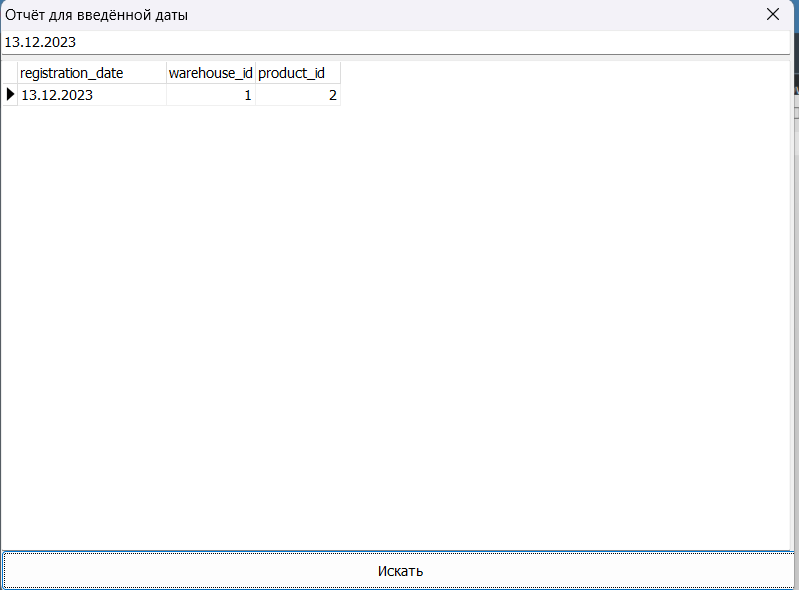


Рис. 7 Результат запроса записей по дате покупки.

Режим SQL

SELECT journal.registration\_date, journal.warehouse\_id, journal.product\_id FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((journal.registration\_date)=idx));

# 4.3 Формы и отчеты

Формы - это средства для ввода и просмотра данных. С помощью форм можно закрыть некоторые поля для несанкционированного ввода, можно разместить специальные элементы управления (счетчики, раскрывающиеся списки, переключатели, флажки и пр.) для автоматизации ввода.

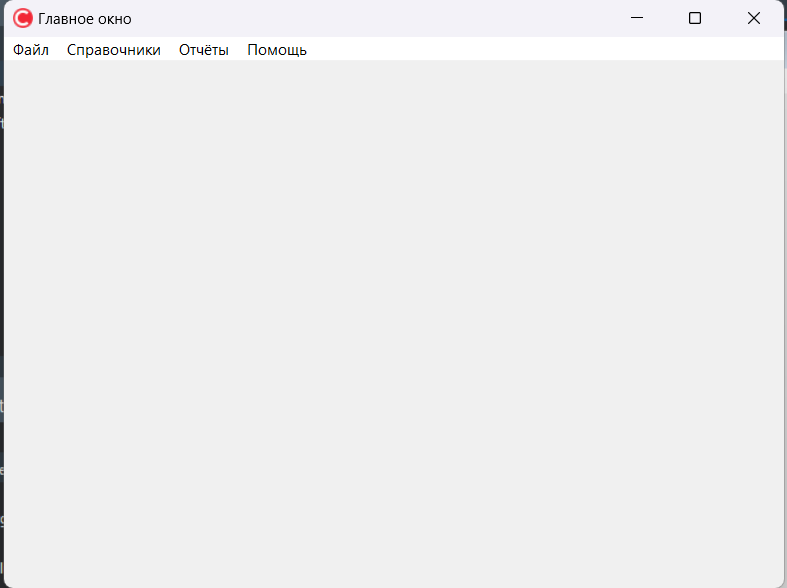


Рис.8 Главная форма

На главной форме показывается меню управления программой, в котором находятся кнопки: “Файл”, “Справочники”, “Отчёты, а также “Помощь”. Особое внимание стоит заострить на “Справочниках” и “Отчётах”, состовляющих большую часть функциональности информационной системы.

В справочниках находятся кнопки, открывающие формы добавления, чтения, обновления и удаления записей соответсвующих названию кнопки таблиц базы данных.

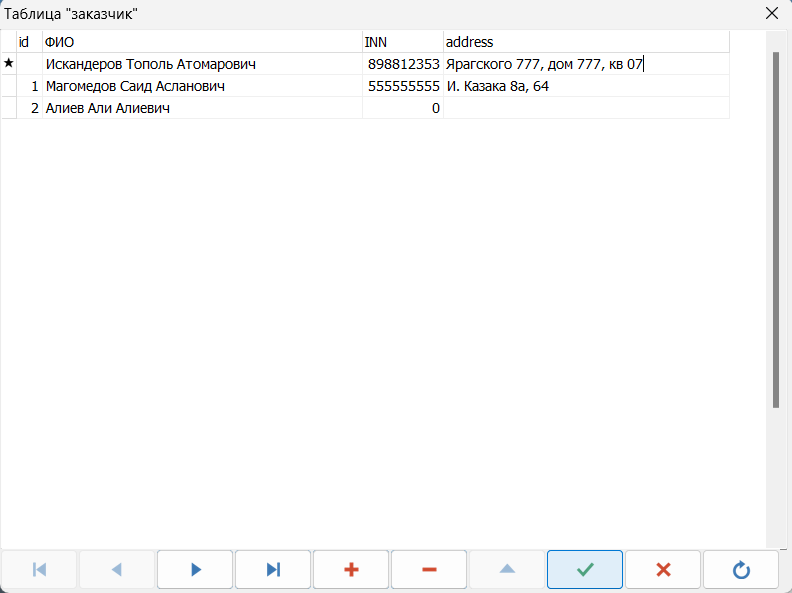


Рис. 9 Регистрация нового пользователя-заказчика

**Отчет** – это форматированное представление данных, которое выводится на экран, в печать или файл. Они позволяют извлечь из базы нужные сведения и представить их в виде, удобном для восприятия, а также предоставляют широкие возможности для обобщения и анализа данных.

При печати таблиц и запросов информация выдается практически в том виде, в котором хранится. Часто возникает необходимость представить данные в виде отчетов, которые имеют традиционный вид и легко читаются. Подробный отчет включает всю информацию из таблицы или запроса, но содержит заголовки и разбит на страницы с указанием верхних и нижних колонтитулов.

Отчет может создаваться с помощью мастера или в режиме конструктора отчетов. Во многих случаях удобно использовать мастера отчетов. Созданный мастером отчет можно доработать в режиме конструктора. Отчеты делятся на разделы, которые в большинстве пакетов программного обеспечения, предназначенных для создания отчетов, называются полосами или разделами.

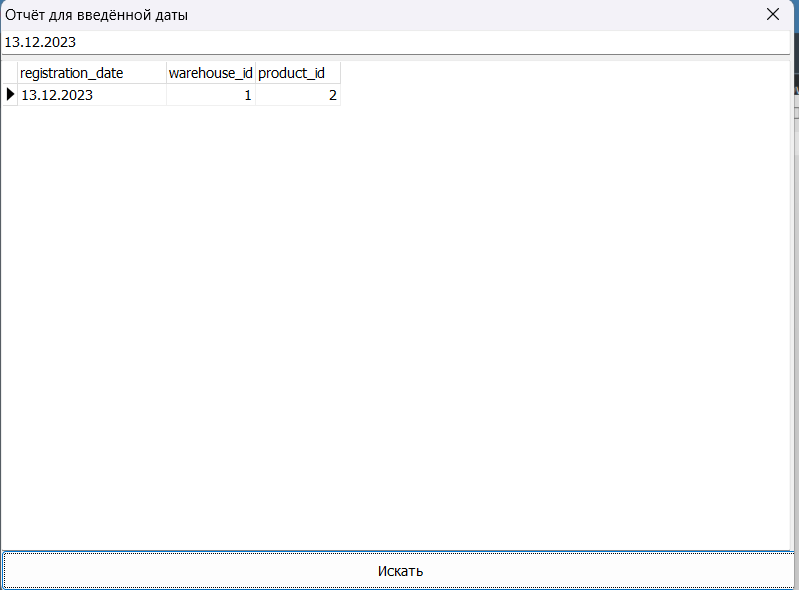


Рис. 10 Отчёт записей за 13 декабря 23года.

5. Защита информации

Существуют следующие способы защиты БД:

1. Кодирование и декодирование.

Кодирование базы данных — это простейший способ защиты. При кодировании базы данных ее файл сжимается и становится недоступным для чтения с помощью служебных программ или текстовых редакторов. Кодирование незащищенной базы данных неэффективно, поскольку каждый сможет открыть такую базу данных и получить полный доступ ко всем ее объектам.

Чтобы приступить к кодированию базы данных, необходимо быть либо ее владельцем, либо, если база данных использует средства защиты, членом группы «Admins» в файле рабочей группы, который содержит учетные записи, используемые для защиты базы данных. Кроме того, базу данных надо открыть в монопольном режиме, для чего необходимо иметь разрешения «открытие/запуск» и «монопольный доступ».

Декодирование базы данных является операцией, обратной кодированию.

1. Отображение и скрытие объектов в окне базы данных.

Другим способом защиты объектов в базе данных от посторонних пользователей является скрытие объектов в окне базы данных. Этот способ защиты является наименее надежным, поскольку относительно просто можно отобразить любые скрытые объекты.

1. Использование параметров запуска.

Параметры запуска позволяют задать такие настройки, как стартовая форма, которая автоматически открывается при открытии базы данных, а также заголовок и значок приложения базы данных. Кроме того, можно скрыть окно базы данных и установить собственную кнопочную форму.

В новой базе данных параметры запуска отсутствуют до тех пор, пока не внесены изменения в диалоговом окне Параметры запуска.

1. Использование пароля.

Другим простейшим способом защиты является установка пароля для открытия базы данных (.mdb). После установки пароля при каждом открытии базы данных будет появляться диалоговое окно, в которое требуется ввести пароль. Только те пользователи, которые введут правильный пароль, смогут открыть базу данных. После открытия базы данных все объекты становятся доступными для пользователя (пока не определены другие типы защиты, описанные ниже в этом разделе). Для базы данных, которая совместно используется небольшой группой пользователей или на автономном компьютере, обычно оказывается достаточно установки пароля.

Можно хранить пароль базы данных в незашифрованном виде. Если это нарушает безопасность защищаемой паролем базы данных, то для защиты базы данных не следует использовать пароль. Вместо этого определите защиту на уровне пользователей, которая помогает управлять доступом к важной информации в базе данных.

Не используйте пароль базы данных, если предполагается выполнять репликацию базы данных. Реплицированные базы данных не могут быть синхронизированы, если определен пароль базы данных.

5. Использование защиты на уровне пользователя.

Наиболее гибкий и распространенный способ реализации средств защиты базы данных называют защитой на уровне пользователя. Защита на уровне пользователя позволяет установить различные уровни доступа к важным данным и объектам в базе данных. Чтобы воспользоваться базой данных, защищенной на уровне пользователя, необходимо ввести пароль при запуске программы. После этого анализируется файл рабочей группы, в котором каждый пользователь идентифицируется уникальным кодом. Уровень доступа и объекты, доступ к которым получает пользователь, зависят от кода и пароля.

Хотя установка защиты на уровне пользователей для большинства баз данных является сложной задачей, мастер защиты позволит быстро и легко защитить базу данных. Более того, благодаря использованию общих схем защиты мастер позволяет уменьшить или даже вообще исключить необходимость использования команды Защита в меню Сервис.

После запуска мастера защиты можно создать собственные группы пользователей и определить разрешения на работу с базой данных и ее таблицами, запросами, формами, отчетами и макросами для различных пользователей или групп пользователей. Также могут быть установлены разрешения на доступ, по умолчанию присваиваемые вновь создаваемым объектам базы данных. Группам и пользователям предоставляются разрешения, определяющие возможность их доступа к каждому объекту базы данных.

6. Запрещение репликации базы данных, установки паролей и настройки параметров запуска пользователями.

В многопользовательской среде часто возникают ситуации, требующие использования средств защиты базы данных. Возможно, потребуется запретить репликацию базы данных. Репликация позволяет пользователям создавать копию общей базы данных, а также добавлять поля и вносить другие изменения в текущую базу данных. Кроме того, может потребоваться запрещение установки пароля базы данных пользователями, поскольку, если это произойдет, никто, не зная пароля, не сможет открыть базу данных. Также следует рассмотреть возможность установки запрета на изменение параметров запуска, которые определяют такие свойства, как настраиваемые меню, настраиваемые панели инструментов и стартовую форму.

Если общая база данных не имеет защиты на уровне пользователей, невозможно запретить пользователям вносить подобные изменения.

При установке защиты на уровне пользователей пользователь или группа для репликации базы данных, установления пароля базы данных и изменения параметров запуска должны иметь такое же разрешение на доступ, как и администратор. Только члены группы «Admins» текущей рабочей группы имеют права администратора.

Если пользователь или группа имеют в настоящий момент разрешение на доступ к базе данных, соответствующее полномочиям администратора, удаление разрешения запретит пользователю или группе внесение изменений. Можно присвоить соответствующее разрешение на доступ пользователю или группе, что позволит им выполнять эти задачи. Невозможно независимо управлять доступом для каждой из таких задач.

Защитить программу можно с помощью пароля, который достаточно ввести один раз за сеанс. Пароль не позволяет несанкционированным пользователям редактировать, копировать, экспортировать, удалять программу VBA, а также вырезать из нее и вставлять в нее фрагменты текста.

Если программа VBA не содержит ошибок и работает правильно, ее текст можно удалить из базы данных, сохранив базу как MDB-файл. Это позволяет защитить интеллектуальную собственность и предотвратить изменения в структуре форм, отчетов и модулей базы данных.

8. Шифрование данных (всей базы или отдельных таблиц) применяется для того, чтобы другие программы не могли прочитать данные. Шифрование исходных текстов программ позволяет скрыть от несанкционированного пользователя описание соответствующих алгоритмов.

# Заключение

В ходе работы над курсовым проектом изучена предметная область в сфере образования, а именно часть работы ректора для подсчета студентов в высшем учебном заведении, добавление и удаление студентов. Определенны и выявлены функциональные и нефункциональные требования к программе. После чего спланирован предполагаемый анализ материально-технической базы. Технология проектирования реализована в Borland C++ выстроена физическая модель, в которой определенны связи между объектами, после чего данные поместили в формате Access.

Приложение реализовано на языке С++, для создания использованы такие объекты как: таблицы, запросы, формы и отчеты. В таблицах хранится вся информация по данной предметной области. Для реализации поставленной задачи были использованы запросы, с помощью которых можно выполнять различные действия с приложением.

С помощью приложения можно:

1.Добавлять, удалять, редактировать операции отпуска товаров со склада;

2.Формировать отчеты и ведомости по заданным запросам.

# Список литературы

1. Исаев Г.Н. Проектирование информационных систем. Учебное пособие. -М: [Омега-Л](http://www.labirint.ru/pubhouse/600/), 2015,424с.
2. Архангельский А.Я. Программирование в C++ Builder . -М: Бином-2010. 1304С.
3. Избачков Ю.С., Петров В.Н Информационные системы. –СПб: Питер. 2006, 656с.
4. Гурвиц Г.А. Microsoft Access 2007. Разработка приложений. - БХВ-Петербург, 2010.
5. Вирджиния А. Базы данных Microsoft Access. - Эком , 2007
6. Кошелев В. Е. Access 2007. Эффективное использование.- Бином-Пресс,2009
7. Гринченко Н. Н. , Гусев Е. В. , Макаров Н. П. Проектирование баз данных. СУБД Microsoft Access. Учебное пособие. - Москва, 2007
8. Карчевский Е.М. Филиппов И.Е.. Access 2007 в примерах.  -«Казанский федеральный университет», 2011
9. Г. Гурвиц. Microsoft Access 2007.Разработка приложений на реальном примере, 2007
10. Мартьянова А. В..  Базы данных и знаний. -АГТУ, 2007
11. Ржеуцкая С.Ю. БД. Язык SQL –ВОГТУ, 2007.

# Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем

# Приложение

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit6.h"

#include "Unit7.h"

#include "Unit8.h"

#include "Unit9.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N10Click(TObject \*Sender)

{

Form1->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N11Click(TObject \*Sender)

{

Form3->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N12Click(TObject \*Sender)

{

Form4->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N13Click(TObject \*Sender)

{

Form5->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N9Click(TObject \*Sender)

{

Form6->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N17Click(TObject \*Sender)

{

Form8->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N18Click(TObject \*Sender)

{

Form7->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N19Click(TObject \*Sender)

{

Form9->Show();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit6.h"

#include "Unit7.h"

#include "Unit8.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm7 \*Form7;

//---------------------------------------------------------------------------

\_\_fastcall TForm7::TForm7(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::Button1Click(TObject \*Sender)

{

ADOQuery1->Active=false;

ADOQuery1->SQL->Clear();

ADOQuery1->SQL->Add("SELECT product.title, journal.registration\_date, journal.id, recipient.ФИО, recipient.address, sender.ФИО FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((product.title)=idx));");

ADOQuery1->Parameters->ParamByName("idx")->Value = Edit1->Text;

ADOQuery1->Active = true;

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit6.h"

#include "Unit7.h"

#include "Unit8.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm8 \*Form8;

//---------------------------------------------------------------------------

\_\_fastcall TForm8::TForm8(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm8::Button1Click(TObject \*Sender)

{

ADOQuery1->Active=false;

ADOQuery1->SQL->Clear();

ADOQuery1->SQL->Add("SELECT recipient.ФИО, journal.registration\_date, product.title, journal.warehouse\_id, journal.id FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((recipient.ФИО)= idx));");

ADOQuery1->Parameters->ParamByName("idx")->Value = Edit1->Text;

ADOQuery1->Active = true;

}

//---------------------------------------------------------------------------

#pragma hdrstop

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit6.h"

#include "Unit7.h"

#include "Unit8.h"

#include "Unit9.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm9 \*Form9;

//---------------------------------------------------------------------------

\_\_fastcall TForm9::TForm9(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm9::Button1Click(TObject \*Sender)

{

ADOQuery1->Active=false;

ADOQuery1->SQL->Clear();

ADOQuery1->SQL->Add("SELECT journal.registration\_date, journal.warehouse\_id, journal.product\_id FROM warehouse INNER JOIN (sender INNER JOIN (recipient INNER JOIN (product INNER JOIN journal ON product.id = journal.product\_id) ON recipient.id = journal.recipient\_id) ON sender.id = journal.sender\_id) ON warehouse.id = journal.warehouse\_id WHERE (((journal.registration\_date)=idx));");

ADOQuery1->Parameters->ParamByName("idx")->Value = Edit1->Text;

ADOQuery1->Active = true;

}