Приднестровский государственный университет им. Т.Г. Шевченко

Рыбницкий филиал

*Кафедра информатики и программной инженерии*

**КУРСОВАЯ РАБОТА**

**по дисциплине**

**«базы данных»**

**на тему:**

**«РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ ДОКУМЕНТООБОРОТА**

**ДЕКАНАТА»**

Выполнил:

cтудент III курса

направления «Программная инженерия»

профиля «Разработка программно-информационных систем»

Толстов И. И.

Проверил:

ст. преподаватель

Ляху А. А.

Рыбница, 2016

Оглавление

[Введение 3](#_Toc453019539)

[1. Описание СУБД 4](#_Toc453019540)

[2. Проектирование физической структуры базы данных 6](#_Toc453019541)

[3. Организация ввода информации в базу данных. Организация корректировки базы данных. 9](#_Toc453019542)

[4. Реализация запросов, получение отчетов 14](#_Toc453019543)

[5. Интерфейс программного продукта и структура главного меню 19](#_Toc453019544)

[6. Реализация проекта в среде СУБД 22](#_Toc453019545)

[7. Тестирование и отладка программной подсистемы 24](#_Toc453019546)

[Заключение 25](#_Toc453019547)

[Список литературы 26](#_Toc453019548)

# Введение

Основные идеи современных информационных технологии базируются на концепции баз данных (БД). Согласно данной концепции, основой информационной технологии являются данные, организованные в БД, адекватно отражающие реалии действительности в той или иной предметной области и обеспечивающие пользователя актуальной информацией в соответствующей предметной области.

Без определенного программного обеспечения возникают сложности при ведении документации, хранении информации о студентах, преподавателях, успеваемости, посещаемости и т. д.

Целью данной курсовой работы является создание приложения в СУБД *Visual Fox Pro 9.0 SP1*, позволяющего автоматизировать документооборот деканата.

Для достижения данной цели были поставлены следующие задачи:

1. Изучить теоретические сведения, необходимые для решения данной задачи.
2. Систематизировать и обобщить полученные знания.
3. Создать физическую структуру базы данных.
4. Создать формы, необходимые для осуществления ключевых действий с базой данных.
5. Реализовать несколько видов отчетов.
6. Провести тестирование и отладку программы.

Задачи курсовой работы определили ее структуру. Вначале приводится описание СУБД, затем проектирование физической структуры программы, организация ввода/вывода, реализация отчетов и тестирование программного продукта.

# Описание СУБД

*Visual FoxPro 9.0 SP1* – популярный и мощный инструмент для разработки приложений, работающих с базами данных. При помощи данной среды разработано очень большое количество статистических, экономических и других приложений, связанных с расчетами и хранением информации.

*Visual FoxPro 9.0 SP1* это событийно-управляемая среда для быстрой разработки приложений по обработке информации, которая может быть использована в различных экономических и расчетных системах. Основной задачей приложения является поддержка одной или нескольких связанных между собой таблиц с данными, выборочное извлечение информации из этих таблиц, анализ информации и графическое представление итоговых данных. Таблицы организуются в базы данных. Для работы с различными данными создается наиболее подходящий для каждого конкретного случая интерфейс, содержащий всевозможные меню, множество управляемых окон и разнотипных элементов управления, благодаря чему экран не загромождается лишней информацией и в то же время вся необходимая информация является легко доступной. Приложение также имеет средства для создания и тиражирования различной отчетной, технической, финансовой и рекламной документации, этикеток, визитных карточек всевозможных размеров, расцветок, стилей оформления, удовлетворяющих самым современным требованиям и изыскательным вкусам. Также приложения *Visual* *FoxPro* 9.0 *SP1* могут обмениваться данными с другими приложениями Windows, функционировать в среде Интернет и оформляться как веб­­­-службы.

Обладая собственным внутренним механизмом управления реляционной БД, тесной взаимосвязью между языком и данными, полноценными возможностями объектно-ориентированного программирования и широким спектром функций *Misrosoft* *Visual* *FoxPro* 9.0 позволяет создавать производительные, масштабируемые БД-ориентированные решения (настольные, клиент-серверные и *Web*) с поддержкой баз данных с таблицами объемом до 2 Гб. При этом *Visual* *FoxPro* 9.0 выгодно отличается от других инструментов *Microsoft* умеренными системными требованиями и высокой эффективностью разрабатываемых приложений (производительность, размеры БД и программного кода). *Visual* *FoxPro* последние годы применяется и для мобильных решений. Теперь на его базе можно разрабатывать и приложения для компьютеров под управлением *Windows* *XP* *Tablet* *PC*.

Значительное число новшеств связано с механизмом управления данными. Все годы существования *Misrosoft* *Visual* *FoxPro* особое внимание уделяется поддержке *SQL*-запросов и взаимодействию с *MS* *SQL* *Server*. С этой целью в версии 9.0 реализованы новые типы данных, сняты многие ограничения *SQL*-языка, введены дополнительные типы индексов, усилена работа с удаленными данными и т. д.

*Visual* *FoxPro* в силу своей предметной нацеленности всегда отличался достаточно мощной системой формирования отчетов. Эти возможности расширены за счет новых архитектурных решений, повышающих возможности управления выводом и форматирования данных.

# Проектирование физической структуры базы данных

Логическая структура реляционной базы данных определяется совокупностью логически взаимосвязанных реляционных таблиц. Каждая реляционная таблица имеет структуру, определяемую реквизитным составом одного из информацион­ных объектов полученной ИЛМ. Логические связи таблиц соответствуют струк­турным связям между объектами.

Логическая структура реляционной базы данных, построенная на основе получен­ной ИЛМ, приведена на рис. 1. На этой схеме реляционные таблицы пред­ставлены структурой, определяемой составом и последовательностью полей (атри­бутов). Ключевые поля отмечены знаком \*. Логические связи изображены линиями между одинаковыми ключами связи.

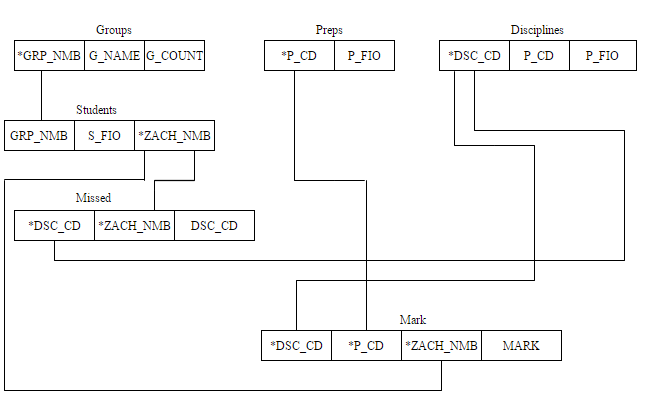


Рис. 1. Логическая структура базы данных

Логическая структура отображает характер связей между объектами системы. При переходе с логического уровня моделирования на физический отображается характер связей между объектами системы.

База данных является интегрированной совокупностью недублируемых данных, на основе которых решаются задачи данной предметной области. Проектирование баз данных на основе реляционной модели имеет ряд важных преимуществ перед другими моделями:

* независимость логической структуры от физического и пользовательского представления;
* гибкость структуры базы данных – конструктивные решения не ограничивают возможности выполнять в будущем самые разнообразные запросы.

Структура базы данных в среде *Microsoft* *Visual* *FoxPro* 9.0 *SP1* имеет вид, представленный на рисунке 2.

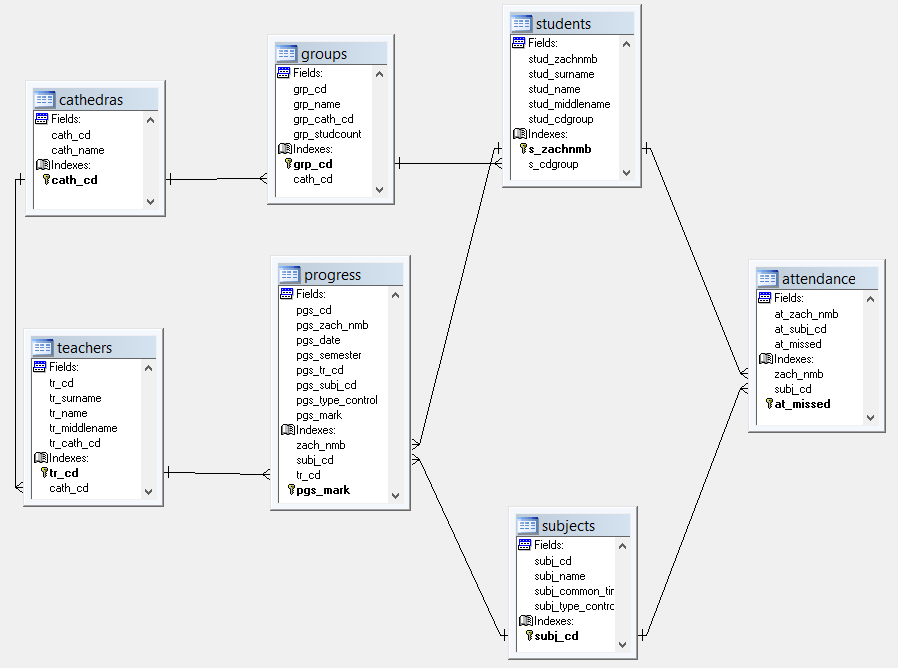


Рис.1. Структура базы данных

Разработанная база данных содержит в себе 7 таблиц, основное назначение которых – хранение полной информации для автоматизации учета движения товаров музыкального магазина (табл. 1).

На данном этапе разработана оптимальная структура базы данных для полноценной работы пользователя. Выделены сущности с набором основных и ключевых атрибутов, определены логические связи между ними. Таким образом, создана физическая модель базы данных. Итогом данного этапа является сгенерированная база данных в виде таблиц, индексов и связей между таблицами.

Таблица 1

Семантика сущностей базы данных

|  |  |  |
| --- | --- | --- |
| **Наименование таблицы** | **Ключевой атрибут** | **Назначение таблицы** |
| cathedras | cath\_cd | Содержит код и название каждой кафедры |
| teachers | tr\_cd | Содержит код преподавателей, ФИО, принадлежность к кафедре |
| groups | grp\_cd | Содержит код каждой группы, принадлежность к кафедре |
| students | s\_zachnmb | Содержит номер зачётной книжки студента, ФИО, принадлежность к группе |
| subjects | subj\_cd | Информация о дисциплинах, читаемых в ВУЗе |
| attendance | zach\_nmb + subj\_cd | Информация о пропущенных часах студента по каждой дисциплине |
| progress | zach\_nmb + subj\_cd + tr\_cd | Содержит информацию об успеваемости студентов |

(Источник: авторская разработка)

# Организация ввода информации в базу данных. Организация корректировки базы данных.

**Пользовательские формы.** При построении форм, была выбрана приятная цветовая гамма и учтена выразительность техники, которая способствует работе пользователя. При проектировании форм придерживались следующих принципов:

* 1. Экранные формы должны иметь уникальные и информативные заголовки.
  2. Все поля необходимо снабдить соответствующими разъяснительными подписями.
  3. Обязательные элементы должны находиться в верхней части экрана. Элементы на экране необходимо упорядочить по степени важности.
  4. Пользователи должны вводить код только один раз и не должны ничего запоминать или записывать при переходе от одной экранной формы к другой.

Назначение любой базы данных заключается в получении пользователем базы необходимой ему информации. Кроме того, пользователю должны быть предоставлены возможности пополнения базы данных вновь возникшей информацией и её коррекции в случае изменения тех или иных компонент, хранящихся в базе данных.

В *Visual* *Fox* *Pro* существует два формата отображения содержимого сущности – в виде таблицы и в виде формы. Формы являются мощным и гибким средством представления информации. Форма позволяет объединять поля в группы по определенным признакам, что облегчает восприятие информации. Она заметно упрощает процесс ввода данных, а также процесс их редактирования. Самый простой вариант формы позволяет вводить данные только в одну таблицу. В разработанной информационной системе примером такой формы может служить формы справочников.

Для оптимальной работы пользователя разработаны формы для ввода и редактирования данных, а также для построения отчетов. Ниже приведены основные формы.

Форма «Кафедры». Данная форма предназначена для просмотра и визуализации данных о кафедрах (рис. 3).

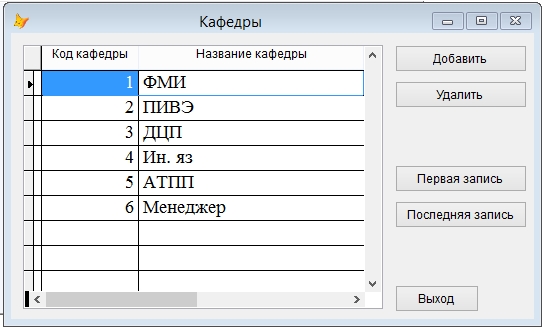


Рис.2. Форма «Кафедры»

**Форма «Группы»**. Данная форма предназначена для просмотра и визуализации данных о группах (рис. 4). **Форма «Преподаватели»**. Данная форма предназначена для просмотра и визуализации данных о преподавателях (рис. 5). **Форма «Предметы»**. Данная форма предназначена для просмотра и визуализации данных о предметах (рис. 6). **Форма «Студенты»**. Данная форма предназначена для просмотра и визуализации данных о студентах (рис. 7). **Форма «Посещаемость»**. Данная форма предназначена для просмотра и визуализации данных о посещаемости студентов (рис. 8). **Форма «Успеваемость»**. Данная форма предназначена для просмотра и визуализации данных об успеваемости каждого студента (рис. 9).

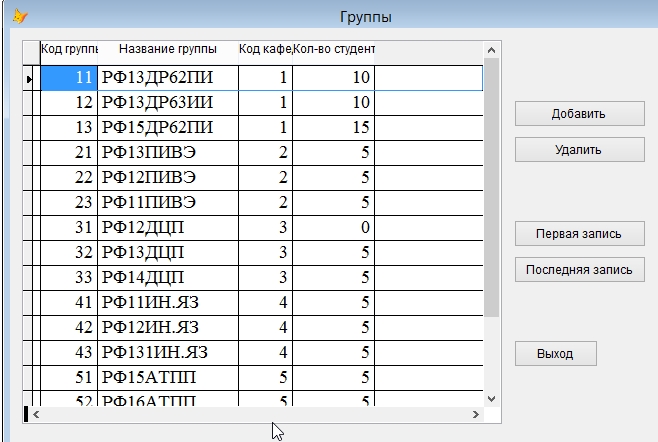


Рис.4. Форма «Группы»

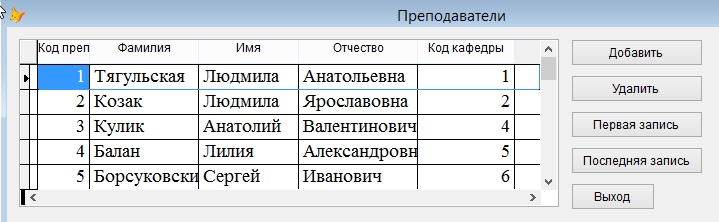


Рис.3. Форма «Преподаватели»

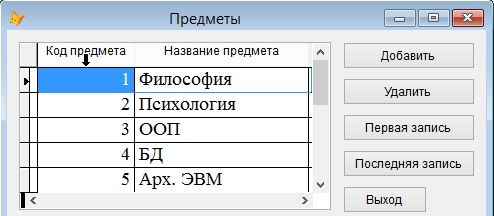


Рис.6. Форма «Предметы»

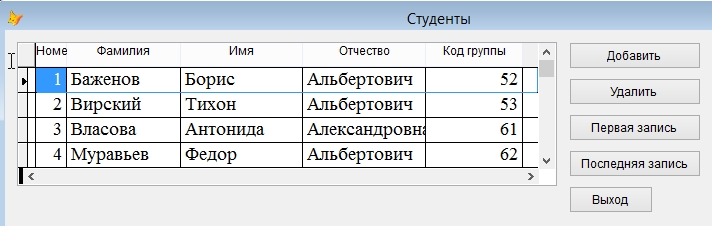


Рис.7. Форма «Студенты»

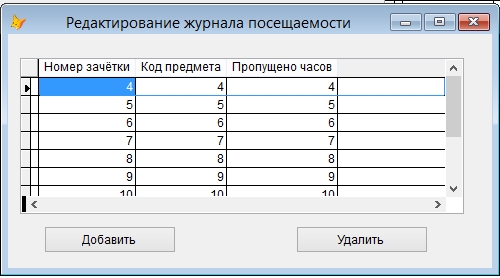


Рис.8. Форма «Посещаемость»

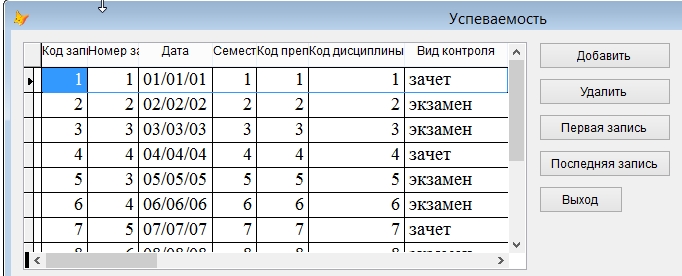


Рис.9. Форма «Успеваемость»

Также разработаны формы для формирования отчетов, которые описываются в следующем разделе.

Последней была разработана форма «О разработчике» (рис. 10). В ней содержится информация о разработчике приложения, версии, а также предназначение программного обеспечения.

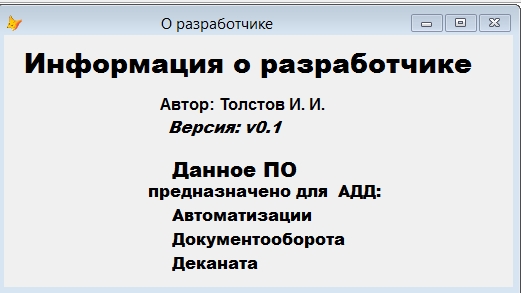


Рис.10. Форма «О разработчике»

Необходимо отметить, что практически во всех формах, кроме формы «О разработчике», функциональная реализация похожа друг на друга. На каждой форме существует следующие функциональные возможности: добавить запись, удалить запись, перейти к первой записи, перейти к последней записи. Реализация форм основана на представлении данных через компонент «Grid».

# Реализация запросов, получение отчетов

Отчет представляет собой форматированное представление данных, выводимое на экран, принтер или файл. Отчет может быть представлен в табличном виде или в свободной форме.

Форма «Отчёт по посещаемости» (рис. 11) предназначена для формирования отчета по посещаемости студента из журнала посещаемости. Пользователю предлагается ввести номер зачётки студента и вызвать функциональную возможность «Сформировать отчёт», тем самым активировать функцию формирования.

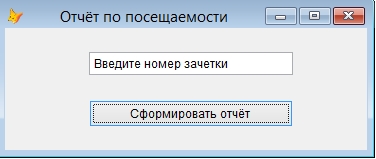


Рис.11. Форма «Отчёт по посещаемости»

Программный код кнопки «Сформировать отчёт»:

IF thisform.text1.Value == "Введите номер зачетки" THEN

MESSAGEBOX("Введите номер зачётки!")

RETURN

ENDIF

IF thisform.text1.Value == "" THEN

MESSAGEBOX("Введите номер зачётки!")

RETURN

ENDIF

IF VAL(thisform.text1.Value) <= 0 THEN

MESSAGEBOX("Введите номер зачётки > 0!")

RETURN

ENDIF

SELECT attendance.at\_zach\_nmb, students.stud\_zachnmb, students.stud\_surname, attendance.at\_subj\_cd, subjects.subj\_cd, subjects.subj\_name, attendance.at\_missed;

FROM attendance, students, subjects;

WHERE at\_zach\_nmb == VAL(thisform.text1.value);

AND stud\_zachnmb == VAL(thisform.text1.value);

AND subj\_cd == at\_subj\_cd;

INTO CURSOR all;

ORDER BY attendance.at\_zach\_nmb

\*BROWSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

oWord = CREATEOBJECT('Word.Application')

oWord.Documents.Add()

oWord.visible=.T.

oWord.Selection.TypeText("Отчет посещаемости по номеру зачётки: " + thisform.text1.Value)

oWord.Selection.TypeParagraph

oWord.Selection.Tables.Add(oWord.Selection.Range,RECCOUNT()+ 1, 4)

i = 1

WITH oWord.Selection

.Tables(1).Cell(i,1).select()

.TypeText("Номер зачётки")

.Tables(1).Cell(i,2).select()

.TypeText("ФИО")

.Tables(1).Cell(i,3).select()

.TypeText("Название предмета")

.Tables(1).Cell(i,4).select()

.TypeText("Пропущено часов")

ENDWITH

SCAN

i = i + 1

WITH oWord.Selection

.Tables(1).Cell(i,1).select()

.TypeText(STR(all.stud\_zachnmb))

.Tables(1).Cell(i,2).select()

.TypeText (all.stud\_surname)

.Tables(1).Cell(i,3).select()

.TypeText (all.subj\_name)

.Tables(1).Cell(i,4).select()

.TypeText(STR(all.at\_missed))

ENDWITH

ENDSCAN

WITH oWord.Selection.Tables(1).Borders

.InsideLineStyle = 1

.OutsideLineStyle = 1

ENDWITH

WITH oWord.Selection.Tables(1).Columns

.autofit()

ENDWITH

Результат работы вышеуказанного кода представлен на рисунке 12.

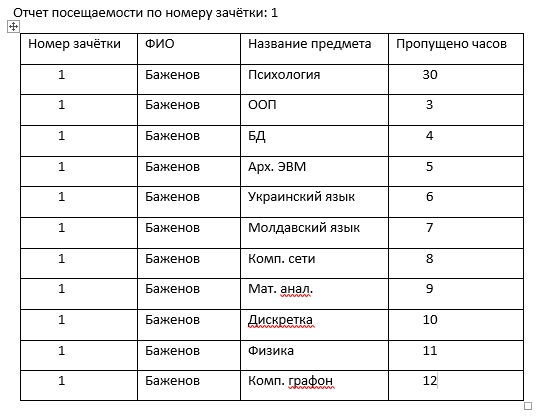


Рис. 12. Результат работы первого отчета в *MS Word 2016*.

Форма «Отчёт по успеваемости» (рис. 13) позволяет сформировать отчёт по успеваемости конкретного студента. Отчет может быть выведен на печать с помощью *Microsoft* *Office* *Word*. Пользователю, как и в предыдущем случае, необходимо ввести номер зачётки и вызвать функциональную возможность «Сформировать отчёт», тем самым активировать функцию формирования.

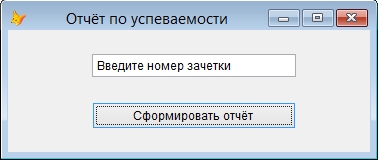


Рис.13. Форма ввода для формирования отчета

Установлены следующие ограничения:

* Невозможно ввести отрицательный номер зачётной книжки.

Программный код кнопки «Сформировать отчёт»:

IF thisform.text1.Value == "Введите номер зачетки" THEN

MESSAGEBOX("Введите номер зачётки!")

RETURN

ENDIF

IF thisform.text1.Value == "" THEN

MESSAGEBOX("Введите номер зачётки!")

RETURN

ENDIF

IF VAL(thisform.text1.Value) <= 0 THEN

MESSAGEBOX("Введите номер зачётки > 0!")

RETURN

ENDIF

SELECT subjects.subj\_cd, subjects.subj\_name, students.stud\_zachnmb, students.stud\_surname, progress.pgs\_zach\_nmb, progress.pgs\_subj\_cd, progress.pgs\_type\_control, progress.pgs\_mark;

FROM subjects, students, progress;

WHERE pgs\_zach\_nmb == VAL(thisform.text1.value);

AND stud\_zachnmb == VAL(thisform.text1.value);

AND subj\_cd == pgs\_subj\_cd;

INTO CURSOR all;

ORDER BY progress.pgs\_zach\_nmb

\*BROWSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

oWord = CREATEOBJECT('Word.Application')

oWord.Documents.Add()

oWord.visible=.T.

oWord.Selection.TypeText("Отчет посещаемости по номеру зачётки: " + thisform.text1.Value)

oWord.Selection.TypeParagraph

oWord.Selection.Tables.Add(oWord.Selection.Range,RECCOUNT()+ 1, 5)

i = 1

WITH oWord.Selection

.Tables(1).Cell(i,1).select()

.TypeText("Номер зачётки")

.Tables(1).Cell(i,2).select()

.TypeText("ФИО")

.Tables(1).Cell(i,3).select()

.TypeText("Название предмета")

.Tables(1).Cell(i,4).select()

.TypeText("Вид контроля")

.Tables(1).Cell(i,5).select()

.TypeText("Оценка")

ENDWITH

SCAN

i = i + 1

WITH oWord.Selection

.Tables(1).Cell(i,1).select()

.TypeText(STR(all.stud\_zachnmb))

.Tables(1).Cell(i,2).select()

.TypeText (all.stud\_surname)

.Tables(1).Cell(i,3).select()

.TypeText (all.subj\_name)

.Tables(1).Cell(i,4).select()

.TypeText(all.pgs\_type\_control)

.Tables(1).Cell(i,5).select()

.TypeText(all.pgs\_mark)

ENDWITH

ENDSCAN

WITH oWord.Selection.Tables(1).Borders

.InsideLineStyle = 1

.OutsideLineStyle = 1

ENDWITH

WITH oWord.Selection.Tables(1).Columns

.autofit()

ENDWITH

Результат работы вышеуказанного кода представлен на рисунке 14.

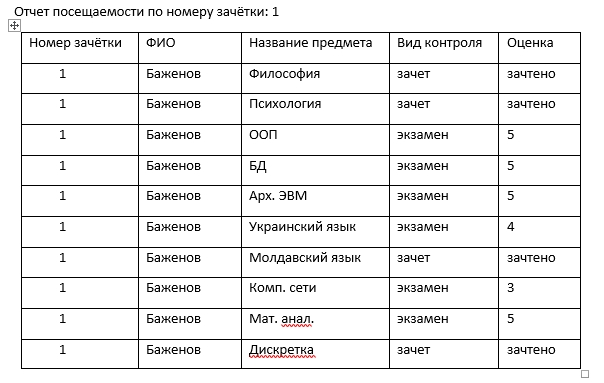


Рис. 14. Результат работы второго отчета в *MS Word 2016*

# Интерфейс программного продукта и структура главного меню

При разработке интерфейса использовалась концепция согласованности. Концепция согласованности состоит в том, что при работе с компьютером у пользователя формируется система ожидания одинаковых реакций на одинаковые действия, что постоянно подкрепляет пользовательскую модель интерфейса.

Так же использовались свойства его конкретности и наглядности. Это осуществлялось применением плана панели, использованием цветов и другой выразительной техники. Идеи и концепции затем обретают физическое выражение на экране, с которым непосредственно общается пользователь.

Программный интерфейс разработан с учетом требований:

* полнота и краткость отображаемой информации;
* удобное расположение информации на экране;
* оперативность работы.

Структура меню приложения во многом также определяет удобство работы пользователя и является одним из важных факторов, влияющих на эффективность его труда. Программа, выполняющая все возложенные на нее функции, обладающая высоким быстродействием может быть полностью непригодной для работы из-за неправильно разработанного интерфейса. Наиболее практичными и удобными с точки зрения пользователя можно считать системы, имеющие экранный интерфейс, построенный на основе системы раскрывающихся меню. Команды обработки данных вызываются из системы меню, присутствующего на экране постоянно. Все команды подсистемы распределены по функциональному признаку на группы и в главном меню присутствуют собственно наименования групп команд. Выбрав группу, пользователь получает доступ к списку команд группы.

Таким образом, создается система многоуровневого меню. Применение такой идеологии обеспечивает удобство ориентировки в системе.

Меню создаваемого приложения организовано на основе классификации необходимых пользователю документов – форм и отчетов. Основное меню организовано следующим образом (рис. 15).

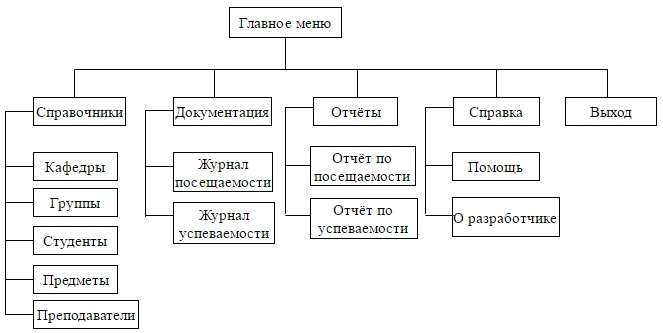


Рис. 15. Структура меню

**Первый уровень иерархии** представлен главным окном, которое состоит из меню и рабочей области программы.

**Второй уровень иерархии** представлен главным меню, которое содержит пункты:

* Справочники – служат для ввода справочной информации в базу данных деканата.
* Документация – служит для ввода оперативно-учетной информации в базу данных деканата;
* Отчёты ­­– служат для формирования необходимых отчетов.
* Cправка – содержит информацию по использованию программного обеспечения, а также информацию о его разработчике.
* Выход – выход из программы.

Интерфейс подсистемы построен таким образом, что не потребуют дополнительного времени от пользователей на его изучение.

Фон, окна меню, и текст отвечают всем требованиям, предъявляемым к цветовому оформлению; цветовая палитра подобрана так, чтобы глаза сотрудников как можно меньше уставали при работе с приложением.

**Пользовательское меню.** Хорошо написанное законченное приложение может использоваться пользователем любой квалификации. Обычно такое приложение имеет свое собственное меню, которое заменяет основное меню Visual FoxPro и содержит команды, предназначенные для выполнения конкретных задач.

Меню создаваемого приложения организовано на основе классификации необходимых пользователю документов – справочники, документация, отчеты, сервис, справка. Основное меню организовано следующим образом (рис. 16).

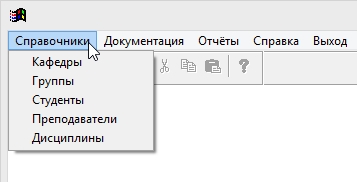


Рис. 16. Главное меню приложения

# Реализация проекта в среде СУБД

При выполнении курсовой работы разработано автономное приложение, запуск которого осуществляется файлом *autodeanydocu.exe*

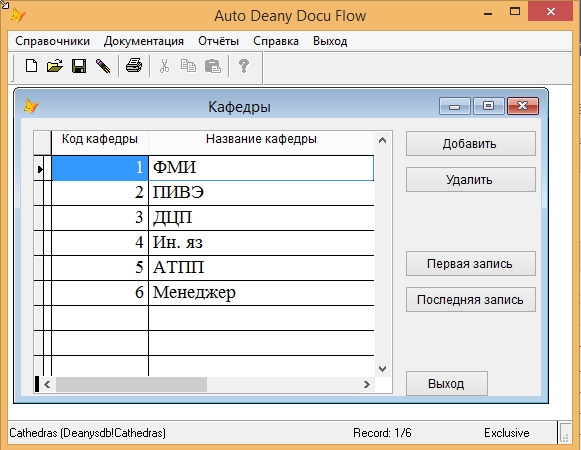


Рис. 17. Запущенный проект с активной формой справочника кафедр

При реализации программного приложения информация о разработчике указана в пункте меню «Справка». Для проекта определена стандартная пиктограмма для выполняемого файла, для этого использовали встроенное средство *Visual* *FoxPro*. Также была введена общая информация о разработчике и компании в *Application Builder*.

В процессе реализации были осуществлены некоторые изменения, которые коснулись логической структуры базы данных, таблиц, пользовательских форм и отчетов:

1. Добавлена таблица «Кафедры», чтобы связать группы и преподавателей определённой кафедрой.
2. На стадии реализации структуры базы данных во многих таблицах были изменены названия полей, некоторые поля были удалены и некоторые добавлены.
3. По необходимости были увеличены размеры (длина) некоторых полей.
4. Полностью удалена таблица «Преподаватель-дисциплина» ввиду ненадобности. Данная таблица просто соединяла ключевые поля из вышеуказанных таблиц, чтобы использовать одно поле в таблице «Успеваемость». Это можно сделать и напрямую с помощью двух полей.
5. Удалено поле «Зарплата» из таблицы «Преподаватели», т. к. для реализации данного поля необходимо учитывать приватность данных. Для достижения поставленной цели нет особой необходимости реализовывать данное поле.

# Тестирование и отладка программной подсистемы

После завершения реализации программного продукта было проведено его тестирование, что является неотъемлемой частью процесса предварительного внедрения. Программный продукт подвергся тщательному тестированию, что очень важно в процессе разработки любого программного продукта. Тестирование проводилось на локальной машине под управлением операционной системы *MS* *Windows 8.1 x64*.

В результате тестирования были выявлены следующие ошибки, которые можно подразделить на группы:

1. Ошибки, заключающиеся в неправильном выборе критериев для получения информации из базы данных.
2. Ошибки, связанные с некорректным взаимодействием элементов между собой.
3. Ошибки, связанные с несоответствием типов полей.
4. Ошибки в формулах вычислений.

Данные ошибки были устранены при помощи стандартного окна отладки и трассировки Visual FoxPro (Debugger). Данное средство позволяет провести выполнение программы по шагам с любого установленного места, отслеживать изменение тех или иных переменных, быстро выявить ошибку и устранить ее.

Анализ результатов тестирования показал, что приложение работает достаточно безотказно, позволяя пользователю вводить, удалять и изменять записи, в соответствии с установленными правилами. Расчеты производятся правильно и быстро.

# Заключение

В настоящее время проблема эффективного управления персоналом и рационального использования ресурсов стоит особенно остро. При этом наибольшее внимание уделяется аспектам формализации решения таких задач, как отбор кадров, аттестация, оценка деятельности, раннее выявление работников с определенными управленческими способностями, повышение обоснованности решений в области кадровой политики и др.

В процессе выполнения курсовой работы поставленная цель была достигнута. Реализовано автономное приложение в среде разработки *Visual Fox Pro 9.0 SP1*. Были созданы основные формы, необходимые для выполнения функций работы с базой данных. С помощью программы *WinCHM* скомпилирован файл справки по программе.

В результате проделанной работы была сделано следующее:

1. Построена структура базы данных.
2. Дано описание среды разработки и разработан основной программный продукт.
3. Описан пользовательский интерфейс и структура главного меню.
4. Созданы и описаны необходимые формы и отчеты.
5. Описан программный код кнопок, использовавшихся на представленных формах.
6. База данных заполнена в соответствии с тестовыми данными. По окончании работы над программным продуктом он подвергся предварительному тестированию.

**Проведенные исследования позволяют сделать вывод о том, что документооборот в жизни ВУЗов имеет существенное значение. При сбоях в данном процессе снижается продуктивность большинства подразделений ВУЗа. Для предотвращения таких проблем, необходимо качественно осуществить организацию документооборота.**

# Список литературы

1. Л.К. Скодорова, А.А. Ляху "Visual FoxPro 9.0 SP1 (Часть I, Часть II) лабораторный практикум. " Рыбница, 2010.
2. Карпова Т. С. Базы данных: модели, разработка, реализация. СПб.:, 2001. 304 с.
3. Ульман Дж. Основы систем баз данных. М.: Финансы и статистика, 1983. 334 с.
4. Тиори Т., Фрай Дж. Проектирование структур баз данных: В 2 т. М.: Мир, 1985.
5. Информационные системы: Учебное пособие для студентов по специальности «Информационные системы в экономике»/ под ред. Волкова В.Н. – СПб.: Издательство СПБГТУ, 1998.
6. Каратыгин С.А., Тихонов А.Ф., Тихонова Л.Н. «Visual FoxPro 6.0» – М.: ЗАО «Издательство БИНОМ», 2000.
7. Ложе И. Информационные системы. Методы и средства. – М.: Издательство «Мир», 1979.
8. Тихонов А.Ф., Тихонова Л.Н. «Visual FoxPro 5.0 (без проблем!)» – М.: Восточная Книжная Компания, 1997.
9. Методические указания для выполнению лабораторных работ по Visual FoxPro 9.0./ сост. Л.К. Скодорова, А.А. Ляху. – Рыбница, 2010.
10. Методические указания к выполнению курсовой работы по курсу «Базы данных» / сост. Л.К. Скодорова, А.А. Ляху. – Рыбница, 2010.