|  |  |  |  |
| --- | --- | --- | --- |
| **Министерство образования и науки Российской Федерации**  Федеральное государственное бюджетное образовательное учреждение  высшего образования  **Московский государственный университет технологии и управления**  **имени К.Г. Разумовского (Первый казачий университет)**  **Университетский колледж информационных технологий** | | | |
| Специальность 09.02.03 Программирование в компьютерных системах | | | |
| КУРСОВОЙ ПРОЕКТ | | | |
| ПМ.02. Разработка и администрирование баз данных  МДК.02.02 Технология разработки и защиты баз данных | | | |
| на тему  Создание и ведение базы данных приёмной комиссии колледжа  **Пояснительная записка**  **УКИТ 09.02.03.2016.304.032ПЗ** | | | |
| Группа | | *П-304* |  |
| Студент | | *Личная подпись (личная подпись)* | Орищенко А.Э. |
| Руководители проекта | | *Личная подпись (личная подпись)* | Ларионова Е.А. |
|  |

**2016**

**Введение**

Курсовой проект посвящен разработке автоматизации обработки информации об абитуриентах. Приёмная комииссия это уполномоченное собрание представителей среднего или высшего образовательного учреждения во главе с председателем приёмной комиссии, занимающееся регулировкой процесса приема студентов в учебное заведени.

Основная цель проекта – создание системы, автоматизировать процесс обработки данных, и выдачи отчетов и статистики по абитуриентам колледжа.

Анализ существующих систем показал, что для данной системы лучше использовать 3 роли базы данных, для соответствующего распределения задач между персоналом осуществляющим работу с данной базой данных.

Данная система позволяет пользователям обрабатывать и хранить информацию об абитуриентах, а также сгенерироват документацию связанную приемом студентов. Система позволяет работать одновременно многим пользовательй.

1. **Основная часть**
   1. **Анализ предметной области**

Приёмная комиссия колледжа - уполномоченное собрание представителей среднего или высшего образовательного учреждения во главе с председателем приёмной комиссии, занимающееся регулировкой процесса перехода учащихся от начальной и средней общеобразовательных ступеней к средне-специальной. Комиссия создаётся для организации приема документов от поступающих, проведения вступительных испытаний, конкурсного отбора и зачисления. Решение о зачислении на бюджет основываются на результатах сдачи ГИА\ЕГЭ, и на проходном бале. Так же они получают документы от абитуриентов, и делают отчеты связанные с поступающими абитуриентами.

Абитуриенты - это выпускники школ и других средних учебных заведений, решившие поступать в данный колледж. Основная функция ИС- обеспечить хранение и оперативную обработку всей поступающей информации в ходе приёмной кампании, а также подготовка документов: списков, справок, ведомостей, отчетов и т.д. Информационное взаимодействие: передача абитуриентами анкетных данных приемной комиссии, информирование абитуриентов об условиях приёма, приём экзаменов и выставление оценок. Для организации эффективного ведения документации приемной комиссии требуется наличие централизованного хранения информации, а также свободного доступа к ней. Существенной проблемой является быстрый и результативный поиск необходимой информации среди огромного объема данных. Применение электронного решения проблем позволяет уменьшить время поиска информации и способствует оптимальному взаимодействию в области создания и контроля документов.

Документы, необходимые для поступления:

1. Заявление;
2. Аттестат или ксерокопия аттестата;
3. Паспорт абитуриента или ксерокопия паспорта;
4. Паспорт одного из родителей или ксерокопия паспорта;
5. Фото 6 штук (3х4);
6. Медицинская справка по форме 086/у;
7. Выписка результатов ОГЭ;

Приемная комиссия получает данный пакет документов от абитуриента, и заносит их в БД для дальнейшей обработки.

Проходной балл – минимальное количество баллов, необходимое для зачисления на место соответствующей специальности, определяется средним баллом аттестата последнего по рейтингу абитуриента входящего в список прошедших студентов, ограниченного числом мест на соответствующей специальности. Полупроходной балл – рассчитывается если несколько человек с одинаковым средним баллом претендуют на меньшее количество мест, тогда решение о прохождении на место определенной специальности основывается на:

1. Подготовительные курсы.
2. Результаты ОГЭ.

ОГЭ (ГИА-9) - это основной обязательный вид экзамена в 9 классе средней школы, аналог ЕГЭ для 9-х классов. Служит для контроля знаний, полученных учащимися за 9 лет, а также для приёма в техникумы.

* 1. **Жизненный цикл базы данных**

Жизненный цикл - совокупность этапов, которые проходит база данных на своём пути от создания до окончания использования, состоит из 7 этапов:

* предварительное планирование;
* проверка осуществимости;
* определение требований;
* концептуальное проектирование;
* логическое проектирование;
* физическое проектирование;
* оценка работы и поддержка базы данных;
  1. **Защита баз данных**

В информационной системе должны быть предусмотрены меры по защите информации. Должна быть предусмотрена аутентификация пользователе, и соответственно должны быть предусмотрены роли пользователей базы данных, в которых предусмотрены ограничения доступа к базе данных. Пользователь из каждой группы не должен иметь права выполнять функционал не предусмотренный для данной группы пользователей. Кроме того, необходимо принимать меры защиты от внедрения SQL скриптов в программу, так называемых SQL-инъекций, и других методов доступа к данным которые не предусмотренные разработчиком базы данных.

* 1. **Выбор и характеристика СУБД**

Для реализации курсового проекта был выбрана в роли СУБД Microsoft SQL Server 2008 R2, так как только данная СУБД установлена на компьютерах на которые рассчитывалась данная база данных. У данной СУБД есть ряд минусов, а точнее отсутствие кроссплатформенности, а также высокую нагрузку, особенно заметную при запуске на слабых машинах.

* 1. **Выбор и характеристика среды разработки приложения**

Для реализации была выбрана Delphi 7, так как только данная СУБД установлена на компьютерах на которые рассчитывалась данная база данных, и поскольку данная среда разработки освоенна мной на высоком уровне.

Также было использована пользовательского пакета AlphaControl, для изменения графики, путем изменения стандартной среды Delphi7.

1. **Специальная часть**
   1. **Постановка задачи**

Разработать приложение для хранения и обработки информации об абитуриентах, так же предусмотреть ограничение прав доступа к базе данных. Предусмотреть возможность добавления абитуриента, и редактирование и удаление данных об абитуриенте, а также просмотр с поиском и сортировкой. Предусмотреть возможность получения отчетов, и получения статистики как по определенным специальностям так и по колледжу в целом. Пользовательский интерефейс должен быть удобен и понятен для пользователя.

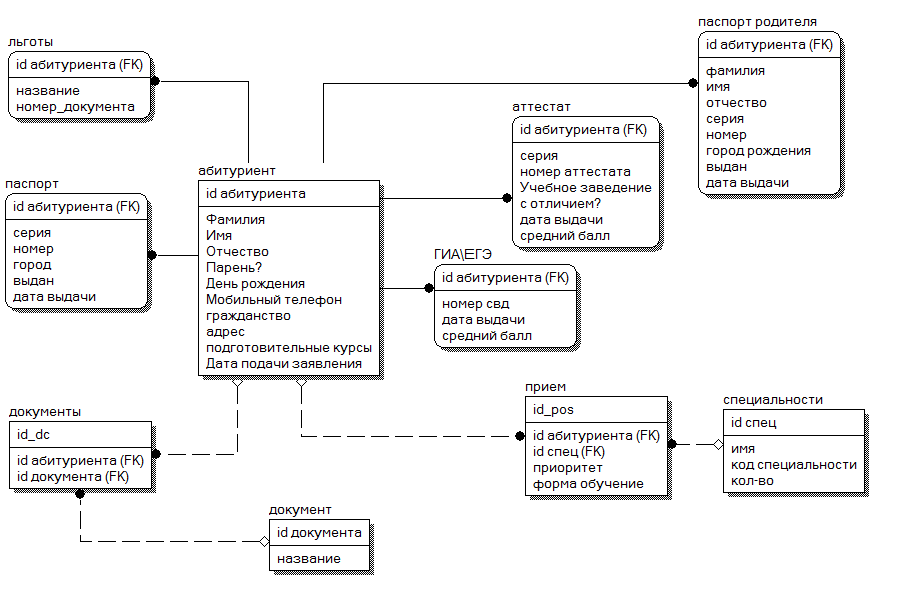
* 1. **Архитектура информационной системы**

В данном курсовом проекте реализована двухзвенная клиент-серверная архитектура. В данной архитектуре база данных находится находиться на сервере, а на клиентском компьютере находится только приложение для работы с базой данных, при этом оно обращается напрямую к базе данных.

Для работы с данными используется реляционный способ доступа, что снижает нагрузку на сеть. Пользователи клиентских машин посылают запрос на сервер и получают те данные, которые им нужны. Вся обработка запросов выполняется на удаленном сервере, при этом приложение обращается к базе данных с запросом на выполнение хранимых процедур.

* 1. **Логическая модель БД**

Рисунок 1. Логическая модель.



* 1. **Нормализация таблиц**

Всякая нормализованная таблица автоматически считается таблицей в первой нормальной форме, сокращенно 1НФ. Таким образом, строго говоря, "нормализованная" и "находящаяся в 1НФ" означают одно и то же. Однако на практике термин "нормализованная" часто используется в более узком смысле – "полностью нормализованная", который означает, что в проекте не нарушаются никакие принципы нормализации. Теперь в дополнение к 1НФ можно определить дальнейшие уровни нормализации – вторую нормальную форму (2НФ), третью нормальную форму (3НФ) и т.д.

* 1. **Описание таблиц**
     1. Таблица Abit\_prow, хранит статистику по специальностям.

Таблица 1, поля таблицы Abit\_prow

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| ID | Идентификатор | INT |
| Name\_spec | Название специальности | VARCHAR(100) |
| Places | Количество мест на специальности | INT |
| Prowed | Количество прошедших на специальность | INT |
| Ne\_prowed | Количество не прошедших на специальность | INT |
| Proh\_ball | Проходной балл на данную специальность | DECIMAL(3,2) |

* + 1. Таблица Abiture, хранит основные сведенья об абитуриентах.

Таблица 2, поля таблицы Abiture

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Surname | Фамилия абитуриента | VARCHAR(50) |
| First\_name | Имя абитуриента | VARCHAR(30) |
| Middle\_name | Отчество абитуриента | VARCHAR(30) |
| Is\_male | Пол абитуриента | BIT |
| Birthday | День рождения абитуриента | DATE |
| Phone | Номер телефона | VARCHAR(11) |
| Address | Адрес проживания | VARCHAR(50) |
| Training\_courses | Подготовительные курсы | BIT |
| Date\_of\_application | Дата подачи заявления | DATE |

* + 1. Таблица Atestat, хранит сведенья об аттестатах.

Таблица 3, поля таблицы Atestat

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Numb\_at | Номер аттестата | INT |
| Institution | Учебное заведение | VARCHAR(50) |
| S\_otl | Аттестат с отличием | BIT |
| Avgn | Средний балл аттестата | DECIMAL(3,2) |
| Date\_of\_issue | Дата выдачи аттестата | DATE |

* + 1. Таблица Benefits, хранит сведенья об льготах.

Таблица 4, поля таблицы Benefits

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Name | Название льготы | VARCHAR(50) |
| Document\_number | Номер документа | VARCHAR(20) |

* + 1. Таблица Document, хранит сведенья о документах.

Таблица 5, поля таблицы Document

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Numb\_doc | Идентификатор документа | INT |
| Name | Название документа | VARCHAR(50) |

* + 1. Таблица Documents, хранит сведенья о том какие документы сдали абитуриенты.

Таблица 6, поля таблицы Documents

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_dc | Идентификатор сданного документа | INT |
| Id\_statement | Идентификатор абитуриента | INT |
| Numb\_doc | Номер документа | INT |

* + 1. Таблица OGE, хранит сведенья о результатах ОГЭ.

Таблица 7, поля таблицы OGE

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Numb\_sv | Номер свидетельства | INT |
| Avg\_oge | Средний балл ОГЭ | DECIMAL(3,2) |
| Date\_of\_issue | Дата выдачи свидетельства | DATE |

* + 1. Таблица Pasp\_rod, хранит сведенья о паспортах родителей.

Таблица 8, поля таблицы Pasp\_rod

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Series | Серия паспорта родителя | INT |
| Numb | Норме паспорта для родителей | INT |
| Surname | Фамилия родителя | VARCHAR(50) |
| First\_name | Имя родителя | VARCHAR(30) |
| Middle\_name | Отчество родителя | VARCHAR(30) |
| Birthday | День рождения родителя | DATE |
| Place\_of\_birth | Место рождения родителя | VARCHAR(70) |

* + 1. Таблица Pasport, хранит сведенья о паспортах.

Таблица 9, поля таблицы Pasport

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_statement | Идентификатор абитуриента | INT |
| Series | Серия паспорта абитуриента | INT |
| Numb | Норме паспорта абитуриента | INT |
| Place\_of\_birth | Место рождения абитуриента | VARCHAR(70) |
| Issued\_by | Кем выдан паспорт | VARCHAR(100) |
| Date\_of\_issue | Дата выдачи паспорта | DATE |

* + 1. Таблица Priem, хранит сведенья выбранных специальностях.

Таблица 10, поля таблица Priem

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_pos | Идентификатор | INT |
| Id\_statement | Идентификатор абитуриента | INT |
| Id\_spec | Идентификатор специальности | INT |
| Priority | Приоритет | INT |

* + 1. Таблица Specialty, хранит специальности.

Таблица 11, поля таблицы Specialty

|  |  |  |
| --- | --- | --- |
| Название поля | Описание | Тип данных |
| Id\_spec | Идентификатор специальности | INT |
| Name\_spec | Название специальности | VARCHAR(150) |
| Code\_spec | Код специальности | VARCHAR(20) |
| Places | Мест на специальности | INT |

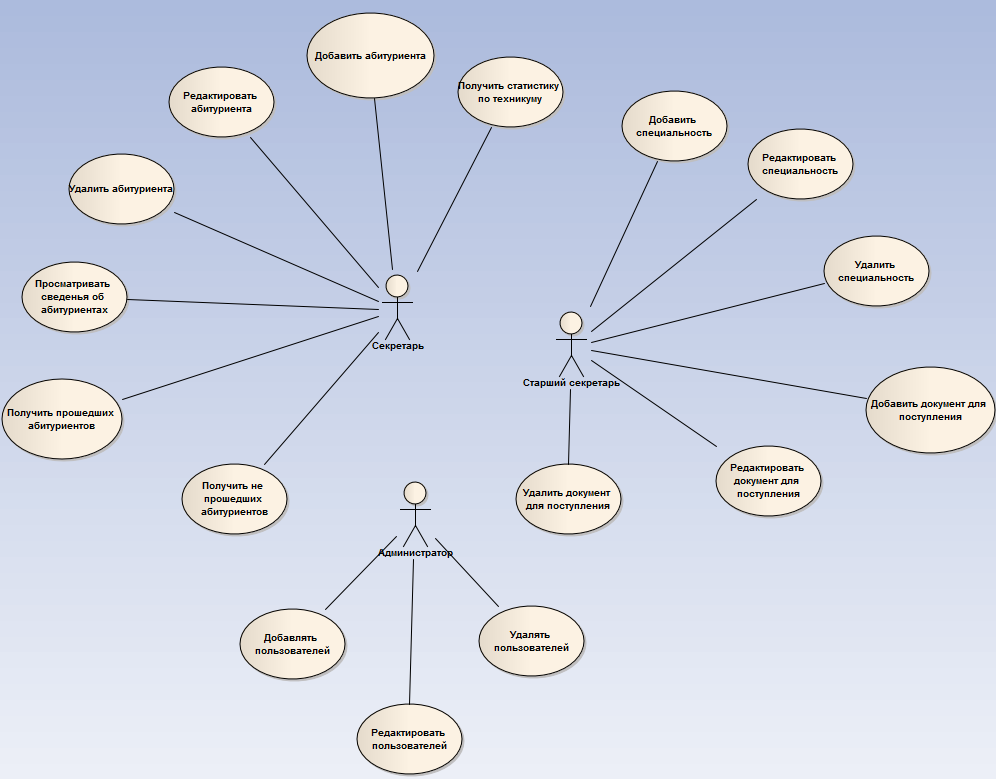
* 1. Другие объекты базы данных

1. Таблица 12, Объекты базы данных

|  |  |
| --- | --- |
| Имя объекта | Описание |
| Add\_doc | Добавление документов |
| Add\_priem | Добавление заявлений на специальности |
| Sel\_title | Выборка основных сведений с поиском |
| Priem\_doz | Получение списка специальностей на которые данный абитуриент подал заявление |
| Sel\_avg\_atest | Выборка студентов по среднему баллу аттестата, с приоритетом и по определенной специальности. |
| Prowedwie | Выборка студентов прошедших на определенную специальность, с выбранным приоритетом |
| Proh\_ball | Получение проходного балла на определенную специальность с выбранным приоритетом |
| Doomsayer |  |
| Polu\_proh | Выборка студентов набравших полупроходной балл на определенную специальность, с выбранным приоритетом |
| Sel\_count\_kurs | Получение количество студентов проходящих подготовительные курсы |
| Ne\_prowedwie | Выборка студентов не прошедших на определенную специальность |
| Gen\_count\_prow | Генерация количество прошедших студентов на определенную специальность |
| Gen\_all\_prow | Генерация количество прошедших студентов на все специальности |
| See\_all\_table | Выборка всех данных об абитуриенте |
| Add\_users | Создать пользователя |
| Update\_users | Изменить пользователя |
| Drop\_users | Удалить пользователя |
| Sel\_user\_deathwing | Вьюшка пользователей БД |

* 1. Разработка приложения
     1. **Структурная схема приложения**

Рисунок 2. Структурная схема приложения.



* + 1. **Используемая технология доступа к данным**

Для подключения к MS SQL Server используется технология ADO.Также реализована возможность выбора сревера базы данных находяшейся в локальной сети из списка доступных, при условии доступа к данным серверам.

Подключение идет через объект ADOConnection, а для получения данных используется ADOQuery, через который путем запросов приложение получает данные, и отображает их в DBGrid компонентах, и стандартных компонентах.

* + 1. **Описание процесса отладки приложения**

В целом серьезных ошибок в ходе написания программы не было, были допущены в основном опечатки, выборки не тех столбцов или вызовы не правильных процедур, либо не было прав на ее выполнение.

* + 1. **Защита информационной системы**

В проекте была предусмотрена защита от SQL-инъекций, путем использования SQL-параметров безопасных типов, и использование параметризованного ввода с хранимыми процедурами. Также в программе предусмотрена авторизация пользователей с вводом логина и пароля, и разграничением прав доступа к данным ролями пользователей что ограничивает возможность работы с базой данных. Для управления пользователями существует отдельная группа пользователей «Admin», в которой реализована возможность создания, редактирования, а также удаления существующих пользователей.

* 1. **Инструкция пользователя**
     1. **Общие сведения об информационной системе**

ИС предназначена для автоматизированной обработки информации, выдачи отчетов и справок, и удобности доступа к информации, а также экономии на бумажных носителях. Система позволяет ускорить процесс получения информации об абитуриентах, дает возможность автоматически рассчитывать прошедших абитуриентов, а также выводить данный отчет на печать. Также в приложении предусмотрен сброс пользовательских настроек.

* + 1. **Требования к техническим средствам**
* Процессор Intel i5-4690K или лучше;
* Оперативная память не менее 2Гб;
* Монитор, клавиатура и мышь;
  + 1. **Требования к программным средствам**
* ОС Microsoft Windows XP или боле поздняя;
* Microsoft SQL Server 2008 R2;
  + 1. **Настройка информационной системы**
* Установка пакета модифицирующего Delphi (находится в папке ./D7\_style/)
  + Папка с компонентами должна быть зарегистрирована в Delphi. (Tools->Environment Options->Library->Library Path)
  + Открываем в Delphi файл acntd7\_r.dpk.
  + В открывшемся окне нажимаем "Build".
  + Открываем файл acnt7.dpk.
  + В открывшемся окне нажимаем "Install".
  + Открываем файл AlphaDB7.dpk.
  + В открывшемся окне нажимаем "Install".
* Создание базы данных из скрипта CREATE\_BD.sql.(Проверена раюота на MS SQL Server 2008 R2 и MS SQL Server 2014)
  + Открываем SQL Server.
  + Выбираем в меню Файл->Открыть->Файл, далее выбираем местоположение файла (по умолчанию ./SQL/CREATE\_BD.sql) и открываем его.
  + Далее нажимаем "Выполнить", база данных создана.

**Стандартные пользователи:**

**Секретарь**

Имя: secr\_user

Пароль: bubl11

**Старший секретарь**

Имя: star\_user

Пароль: doom22

**Администратор**

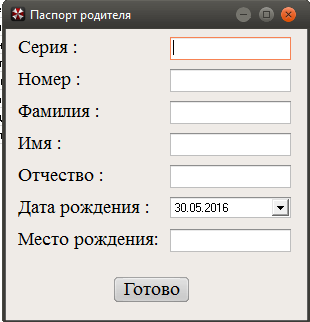
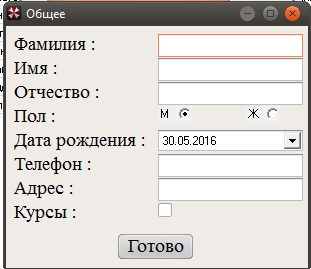
Имя: admn\_user

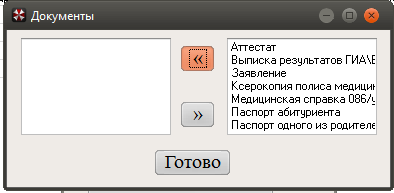
Пароль: slay33

**Настройки** (Хранятся в ini-файле, по умолчанию ./program/Config.ini)

* [DataBase]-> IPDef - хранит адресс MSSQLServer`а который будет использоваться по умолчанию. (также происходит поиск доступных устройств в локальной сети)
* [Auth]-> UserLogin - хранит логин авторизовавшегося вошедшего пользователя, c этого устройства. (используется при входе чтобы упростить повторную авторизацию пользователя)
* [Design]-> SkinID - хранит номер стиля который был выбран последним (для авторизации).
* [StyleID]-> <Имя пользователя> - хранит номер стиля который выбрал данный пользователь.
  + 1. **Формы ввода**

Рисуноки 3,4,5,6,7,8 Общее





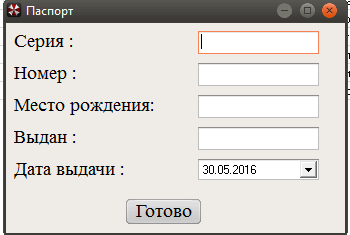
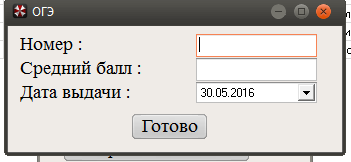
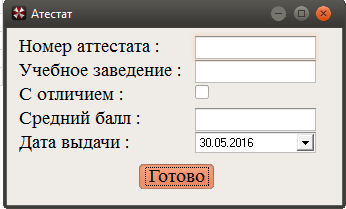


Рисунок 9. Главная форма.

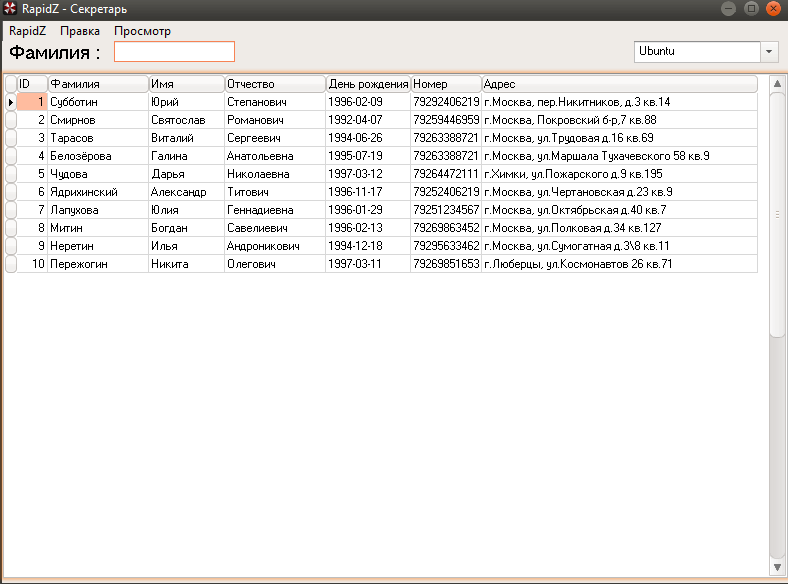


Рисунок 10. Проходной балл.

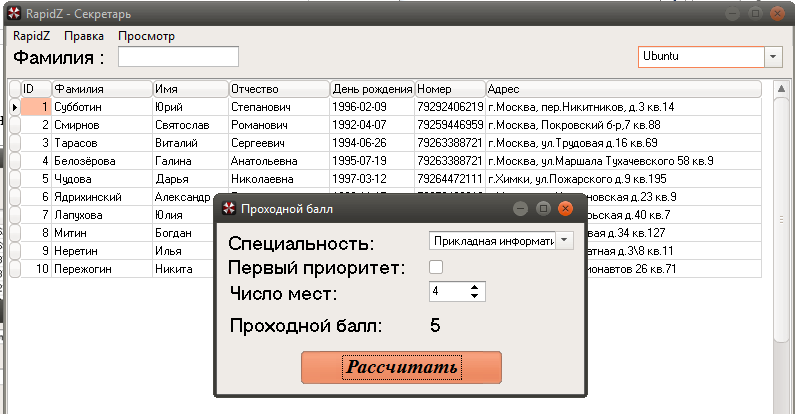
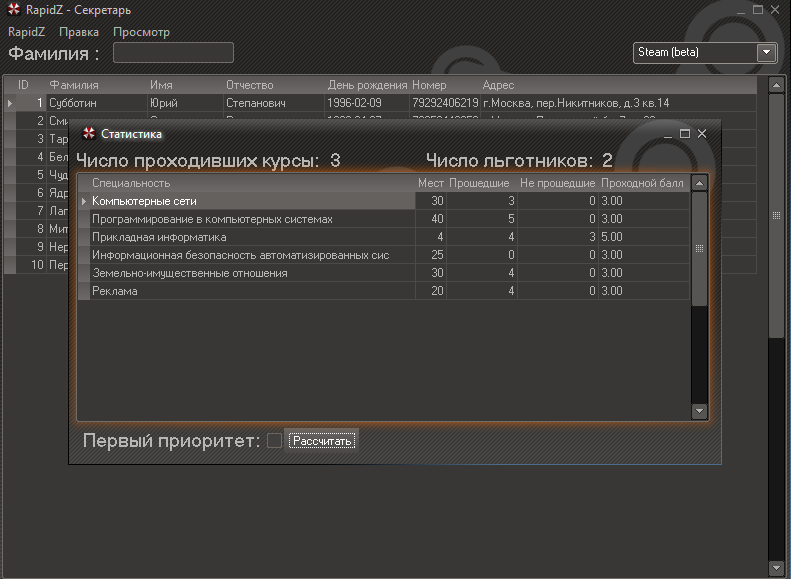


Рисунок 11. Статистика.



* + 1. **Отчеты**

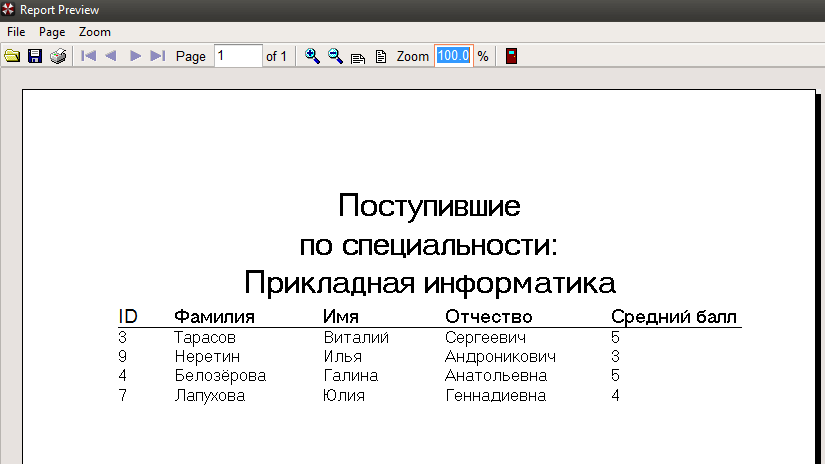
Рисунок 12. Поступившие на специальность. 

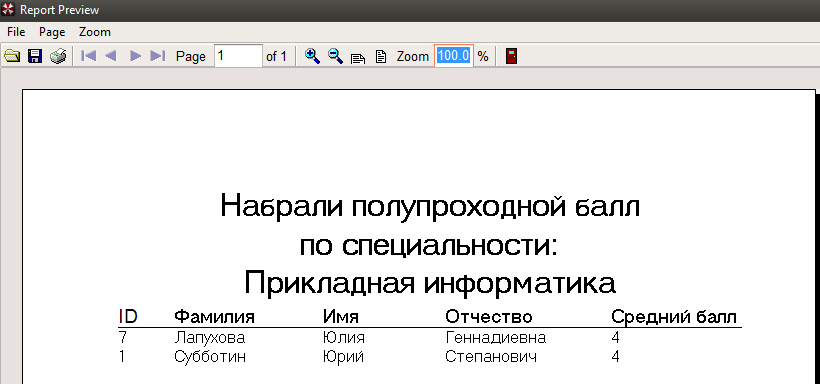
Рисунок 13. Набрали полупроходной балл на специальность. 

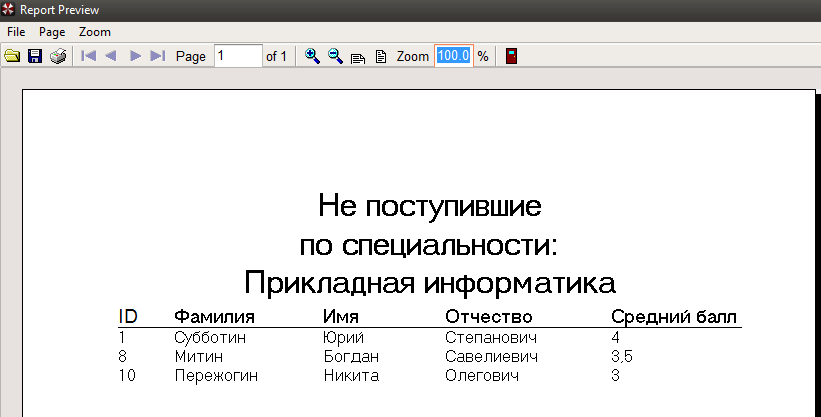
Рисунок 14. Не поступившие на специальность. 

Рисунок 15. Расписка в приеме документов.

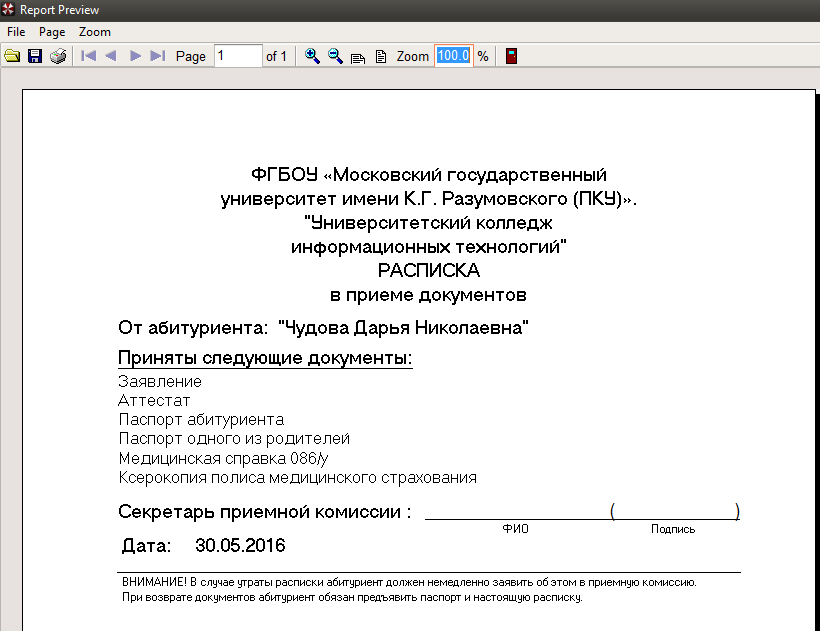


Рисунок 16. Заявление.



**Заключение**

В ходе выполнения проекта была создана информационная система базы данных приёмной комиссии колледжа, которая в себя включает скрипт создания БД, с созданием всех объектов базы данных, а также ролей и стандартных пользователей БД. Было разработано приложение позволяющее удаленное управление базой данных, имеющее широкий спектр возможностей, как для работы с базой данных, так и для обработки действий пользователей, такие как генерирование отчетов, сортирови отображаеммых данных, либо поиска по ним с запросом к базе данных.

Репозиторий проекта доступен на [github.com](https://github.com/Zaxelon/4revo.git) (кликабельно)  
также ниже приведен $ git log, который идентичен с логом на сайте [github.com](https://github.com/Zaxelon/4revo.git).

commit fdd343592843067e774f4f7f8fb7f10607ceabdd

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Tue May 31 01:50:26 2016 +0300

[dock] modified: CREATE\_BD.sql

commit fefb8c34e468ec1a7d831566505af24a76d20eb5

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Fri May 27 00:02:25 2016 +0300

added documentation for robodoc

commit 081aabd584a2bcb1af41f5942bc6d9a82c351aec

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Wed May 18 01:29:02 2016 +0300

added: error processing and installation manual

commit 389ac74a0ee980d4ccb26a4705643834709f146f

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Mon May 16 16:02:29 2016 +0300

RapidZ 0.4 Beta

commit 81f8a1ae7260e6550c6cdbc796e4f2a6d63e6ef2

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Mon May 16 10:18:00 2016 +0300

FIX ERROR into launge PROJ

commit f31dc787b65eb4b71dd5b984f97eb34a90f8587e

Author: Zaxelon <1@zaxelon.hol.es>

Date: Mon May 16 09:55:38 2016 +0300

Create README.md

commit a7babfc4df20b4176985158cb642923e29c40721

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Mon May 16 04:17:14 2016 +0300

[FIX], [INI], [DESIGN]

FIX BUG into deleting record from specialty and documents.

FIX search student table into secretar.

ADD save settings to .ini file.

Complete redesign project.

commit 479c59c8bdc13bca165488d610703c52f125a3f7

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Sat May 14 18:35:59 2016 +0300

The project design was changed,

it is possible to use the Skins.

commit f568fbb0558ce53c85fc5b7c4dd0e7d92680f3d0

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Wed May 11 12:12:42 2016 +0300

beta 0.3

commit c073088dfbc1a89bfc5642136129869483734f48

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Fri May 6 18:52:07 2016 +0300

beta 0.2

commit ba460a776250e178dcdd51a153b97ffcca26b970

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Thu May 5 16:40:34 2016 +0300

beta 0.1

commit d99b2f1a4c91261bbdd7f6c61f76071962ca6e98

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Tue May 3 01:25:40 2016 +0300

add all rave report

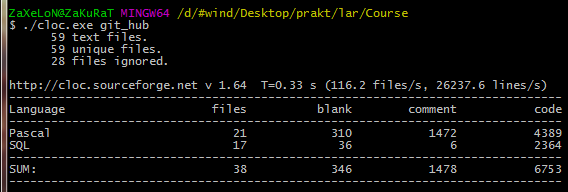
commit e72c1f17629df694173a3dbf32a1fe2b76ed7af1

Author: ZaXeLoN <1@zaxelon.hol.es>

Date: Sat Apr 30 13:49:47 2016 +0300

Frist commit

Полная статистика, полученная при помощи утилиты CLOC.



Начало разработки приложения - 30 апреля 2016.

Последний коммит на данный момент 27 мая 2016.

Из чего следует что на разработку приложения был затрачен один месяц, разбор предметной области и написание базы данных (включая скрипт создания) заняло приблизительно тоже время. Написани пояснительной записки заняло неделю.

**Список литературы**

Учебник по языку SQL/ М.: «Хабрахабр».

URL: <https://habrahabr.ru/post/255361>

Справочник по языкам SQL Server / М.: «Microsoft».

URL: <https://msdn.microsoft.com/ru-ru/library/dn205322(v=sql.120).aspx>

VCL components for GUI design development / М.: «AlphaControls».

URL: <http://www.alphaskins.com/index.php>

Проектирование баз данных / М.: «Citforum».

URL: <http://citforum.ru/database/dbguide/>

Работа с ADO в Delphi / М.: «Devdelphi».

URL: <http://devdelphi.ru/?p=16>

Rave report Delphi 7 / М.: «Delphimaster».

URL: <http://www.delphimaster.ru/articles/raverpt/>

Шифрование SQL Server / М.: «Microsoft».

URL: <https://msdn.microsoft.com/ru-ru/library/bb510663(v=sql.120).aspx>

**Приложение 1, SQL скрипт.**

SET NOCOUNT ON;

IF OBJECT\_ID('priemka', N'U') IS NOT NULL DROP DATABASE [priemka];

GO

IF OBJECT\_ID('priemka', N'U') IS NULL

BEGIN

CREATE DATABASE [priemka]

ON

( NAME = priemka,FILENAME = 'C:\priemka.mdf',SIZE = 10,MAXSIZE = 100,FILEGROWTH = 5)

LOG ON

(NAME = priemka\_log,FILENAME = 'C:\priemka\_log.ldf',SIZE = 5,MAXSIZE = 50,FILEGROWTH = 5);

END

GO

USE priemka

GO

CREATE TABLE abiture (

id\_statement int IDENTITY(1,1),

surname varchar(50) NOT NULL,

CHECK (LEN(surname)>2),

first\_name varchar(20) NOT NULL,

CHECK (LEN(first\_name)>2),

middle\_name varchar(30),

CHECK (LEN(middle\_name)>2),

is\_male bit NOT NULL,

birthday date NOT NULL,

CHECK ((birthday)<CONVERT(date,GETDATE())),

phone varchar(11) NOT NULL,

CHECK (LEN(phone)>5),

address varchar(50) NOT NULL,

CHECK (LEN(address)>2),

training\_courses bit NOT NULL,

date\_of\_application date NOT NULL,

CHECK ((date\_of\_application)<=CONVERT(date,GETDATE())),

PRIMARY KEY (id\_statement)

);

GO

CREATE TABLE benefits (

id\_statement int NOT NULL,

name varchar(50) NOT NULL,

CHECK (LEN(name)>2),

document\_number varchar(20) NOT NULL,

CHECK (LEN(document\_number)>2),

PRIMARY KEY (id\_statement),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE pasport (

id\_statement int NOT NULL,

series int NOT NULL,

CHECK (LEN(series)=4),

numb int NOT NULL,

CHECK (LEN(numb)=6),

place\_of\_birth varchar(70) NOT NULL,

CHECK (LEN(place\_of\_birth)>2),

issued\_by varchar(100) NOT NULL,

CHECK (LEN(issued\_by)>2),

date\_of\_issue date NOT NULL,

CHECK ((date\_of\_issue)<CONVERT(date,GETDATE())),

PRIMARY KEY (id\_statement),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE document (

numb\_doc int IDENTITY(1,1),

name varchar(50) NOT NULL,

CHECK (LEN(name)>2),

PRIMARY KEY (numb\_doc)

);

GO

CREATE TABLE documents (

id\_dc int IDENTITY(1,1),

id\_statement int NOT NULL,

numb\_doc int NOT NULL,

PRIMARY KEY (id\_dc),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (numb\_doc) REFERENCES document(numb\_doc) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE pasp\_rod (

id\_statement int NOT NULL,

series int NOT NULL,

CHECK (LEN(series)=4),

numb int NOT NULL,

CHECK (LEN(numb)=6),

surname varchar(30) NOT NULL,

CHECK (LEN(surname)>2),

first\_name varchar(20) NOT NULL,

CHECK (LEN(first\_name)>2),

middle\_name varchar(30),

CHECK (LEN(middle\_name)>2),

birthday date NOT NULL,

CHECK ((birthday)<CONVERT(date,GETDATE())),

place\_of\_birth varchar(70) NOT NULL,

CHECK (LEN(place\_of\_birth)>2),

PRIMARY KEY (id\_statement),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE specialty (

id\_spec int IDENTITY(1,1),

name\_spec varchar(150) NOT NULL,

code\_spec varchar(20) NOT NULL,

CHECK (LEN(code\_spec)>2),

places int,

CHECK (places>=0),

PRIMARY KEY (id\_spec)

);

GO

CREATE TABLE priem (

id\_pos int IDENTITY(1,1),

id\_statement int NOT NULL,

id\_spec int NOT NULL,

priority int NOT NULL,

CHECK (priority>0 and priority<4),

PRIMARY KEY (id\_pos),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (id\_spec) REFERENCES specialty(id\_spec) ON UPDATE CASCADE

);

GO

CREATE TABLE atestat (

id\_statement int NOT NULL,

numb\_at int NOT NULL,

CHECK (numb\_at>0),

CHECK (LEN(numb\_at)=6),

institution varchar(50) NOT NULL,

CHECK (LEN(institution)>2),

s\_otl bit NOT NULL,

avgn decimal(3,2) NOT NULL,

CHECK ((avgn>=2) and (avgn<=5)),

date\_of\_issue date NOT NULL,

CHECK ((date\_of\_issue)<CONVERT(date,GETDATE())),

PRIMARY KEY (id\_statement),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE oge (

id\_statement INT NOT NULL,

numb\_sv INT NOT NULL,

CHECK (numb\_sv>0),

avg\_oge decimal(3,2) NOT NULL,

CHECK ((avg\_oge>=2) and (avg\_oge<=5)),

date\_of\_issue date NOT NULL,

CHECK ((date\_of\_issue)<CONVERT(date,GETDATE())),

PRIMARY KEY (id\_statement),

FOREIGN KEY (id\_statement) REFERENCES abiture(id\_statement) ON DELETE CASCADE ON UPDATE CASCADE

);

GO

CREATE TABLE abit\_prow(

ID INT IDENTITY(1,1),

name\_spec varchar(50),

places INT,

prowed INT,

ne\_prowed INT,

proh\_ball varchar(20)

);

GO

CREATE PROCEDURE add\_doc

@id\_statement int,

@name varchar(50)

AS

INSERT INTO documents(id\_statement,numb\_doc) VALUES

(@id\_statement,(SELECT TOP 1 numb\_doc from document WHERE (name=@name)))

GO

CREATE PROCEDURE add\_priem

@id\_statement int,

@name\_spec nvarchar(100),

@priority int

AS

INSERT INTO priem(id\_statement,id\_spec,priority) VALUES

(@id\_statement,(SELECT TOP 1 id\_spec FROM specialty WHERE (name\_spec = @name\_spec)),@priority)

GO

CREATE PROCEDURE sel\_title

@surname varchar(50)

AS

DECLARE @bum varchar(51)

if @surname is not null

Begin

SET @bum = @surname + N'%'

end else

Begin

SET @bum = N'%'

end

SELECT id\_statement AS ID, surname AS [Фамилия], first\_name AS [Имя],

middle\_name AS [Отчество],birthday AS [День рождения], phone AS [Номер],

address AS [Адрес] FROM abiture

WHERE (surname like @bum)

GO

CREATE PROCEDURE priem\_doz

@kot INT

AS

Begin

DECLARE @port table

(

num int IDENTITY,

name\_spec varchar(100)

)

INSERT INTO @port SELECT name\_spec FROM specialty

SELECT num FROM @port

WHERE name\_spec in (SELECT name\_spec AS [TJK] FROM specialty INNER JOIN priem ON

priem.id\_spec = specialty.id\_spec WHERE id\_statement= @kot)

end

GO

CREATE PROCEDURE sel\_avg\_atest

@name\_spec varchar(100),

@prior bit

AS

DECLARE @nsmp varchar(100)

DECLARE @st\_count int

if (@prior = 1)

begin

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = 1)

ORDER BY dbo.atestat.avgn DESC

end else

begin

if @name\_spec is not null

begin

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end else

begin

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

ORDER BY dbo.atestat.avgn DESC

end

end

GO

CREATE PROCEDURE prowedwie

@name\_spec varchar(100),

@prior int,

@count int

AS

BEGIN

DELETE FROM abit\_prow

DECLARE @not\_avg decimal(4,2)

DECLARE @id\_s INT

DECLARE @sr\_nm varchar(50)

DECLARE @fr\_nm varchar(50)

DECLARE @md\_nm varchar(50)

DECLARE @avgn decimal(4,2)

DECLARE @docnb varchar(20)

DECLARE @zpr table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

if @prior <> 0

Begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior)

ORDER BY dbo.atestat.avgn DESC

end

else

begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end

OPEN zlp

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

WHILE @@FETCH\_STATUS = 0

BEGIN

DECLARE @iluwka decimal(4,2)

if @docnb is not null

SET @iluwka = @avgn + 10.00 else

SET @iluwka = @avgn

INSERT INTO @zpr (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @iluwka)

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

END

CLOSE zlp

DEALLOCATE zlp

DECLARE @zpr1 table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

DECLARE @zapas table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

DECLARE @cntzt INT

if @prior <> 0

Begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ FROM @zpr)

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zpr

ORDER BY avgn DESC

end

else

begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ FROM @zpr)

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zpr

ORDER BY avgn DESC

end

OPEN zlp

FETCH LAST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

if @cntzt>@count SET @not\_avg = @avgn else SET @not\_avg = 0

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0

BEGIN

if not(@avgn = @not\_avg)

Begin

DECLARE @bubl decimal(4,2)

if @avgn>10 SET @bubl = @avgn - 10.00

else SET @bubl = @avgn

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @bubl)

end else

Begin

DECLARE @pub decimal(4,2)

if @avgn>10 SET @pub = @avgn - 10.00

else SET @pub = @avgn

INSERT INTO @zapas(ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @pub)

end

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

CLOSE zlp

DEALLOCATE zlp

DECLARE @real\_count INT

SET @real\_count = (SELECT COUNT(\*) FROM @zpr1)

IF @count > @real\_count

Begin

DECLARE zlp SCROLL CURSOR FOR

SELECT ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zapas as [zp]

ORDER BY (SELECT bb.avg\_oge FROM oge as[bb] WHERE (bb.id\_statement = zp.ID)) DESC

OPEN zlp

DECLARE @old\_s INT

SET @old\_s = 0

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0 and @count > @real\_count

BEGIN

DECLARE @zip bit

SET @zip =(SELECT training\_courses FROM dbo.abiture

WHERE (id\_statement = @id\_s))

if @old\_s <> @avgn

BEGIN

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

END else

BEGIN

if @zip = 1

Begin

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

end

END

SET @old\_s = @avgn

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0 and @count > @real\_count

BEGIN

SET @zip =(SELECT training\_courses FROM dbo.abiture

WHERE (id\_statement = @id\_s))

if @old\_s = @avgn

BEGIN

if @zip = 0

Begin

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

end

END

SET @old\_s = @avgn

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

CLOSE zlp

DEALLOCATE zlp

END

SELECT ID as [id\_statement],sr\_nm as surname,fr\_nm as first\_name,md\_nm as middle\_name,avgn FROM @zpr1

END

GO

CREATE PROCEDURE [dbo].[proh\_ball]

@name\_spec varchar(100),

@prior int,

@count int

AS

BEGIN

DECLARE @not\_avg decimal(3,2)

DECLARE @id\_s INT

DECLARE @sr\_nm varchar(50)

DECLARE @fr\_nm varchar(50)

DECLARE @md\_nm varchar(50)

DECLARE @avgn decimal(3,2)

DECLARE @cntzt INT

DECLARE @skip bit

SET @skip = 0

if @prior <> 0

Begin

SET @cntzt = (SELECT TOP(@count) COUNT(\*) AS CNTZ

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior))

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count) dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior)

ORDER BY dbo.atestat.avgn DESC

end

else

begin

SET @cntzt = (SELECT TOP(@count) COUNT(\*) AS CNTZ

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec))

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count) dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end

OPEN zlp

if @count = 0 or @name\_spec = '' SELECT 'FALSE' AS APoL

if @cntzt >= @count

Begin

FETCH LAST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

if @cntzt>=@count SET @not\_avg = @avgn else SET @not\_avg = 0

FETCH LAST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0

BEGIN

if not(@avgn = @not\_avg)

Begin

SELECT @avgn AS APoL

SET @skip = 1

BREAK

end

FETCH PRIOR FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

END ELSE SELECT 3.00 AS APoL

SELECT 'FALSE' AS APoL

CLOSE zlp

DEALLOCATE zlp

END

GO

CREATE PROCEDURE doomsayer

@name\_spec varchar(100),

@prior int,

@count int

AS

BEGIN

DECLARE @not\_avg decimal(3,2)

DECLARE @id\_s INT

DECLARE @sr\_nm varchar(50)

DECLARE @fr\_nm varchar(50)

DECLARE @zpr table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(3,2)

)

DECLARE @md\_nm varchar(50)

DECLARE @avgn decimal(3,2)

DECLARE @cntzt INT

if @prior <> 0

Begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior))

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior)

ORDER BY dbo.atestat.avgn DESC

end

else

begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec))

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end

OPEN zlp

FETCH LAST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

if @cntzt>@count SET @not\_avg = @avgn else SET @not\_avg = 0

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0

BEGIN

if not(@avgn = @not\_avg)

INSERT INTO @zpr (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

CLOSE zlp

DEALLOCATE zlp

SELECT \* FROM @zpr

END

GO

CREATE PROCEDURE polu\_proh

@name\_spec varchar(100),

@prior int,

@count int

AS

BEGIN

DECLARE @not\_avg decimal(4,2)

DECLARE @id\_s INT

DECLARE @sr\_nm varchar(50)

DECLARE @fr\_nm varchar(50)

DECLARE @md\_nm varchar(50)

DECLARE @avgn decimal(4,2)

DECLARE @docnb varchar(20)

DECLARE @zpr table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

DECLARE @ideallocate table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

if @prior <> 0

Begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior)

ORDER BY dbo.atestat.avgn DESC

end

else

begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end

OPEN zlp

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

WHILE @@FETCH\_STATUS = 0

BEGIN

DECLARE @iluwka decimal(4,2)

if @docnb is not null

SET @iluwka = @avgn + 10.00 else

SET @iluwka = @avgn

INSERT INTO @zpr (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @iluwka)

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

END

CLOSE zlp

DEALLOCATE zlp

if ((SELECT COUNT(\*) FROM @zpr)>@count) and (@count <> 0)

BEGIN

DECLARE zlp SCROLL CURSOR FOR

SELECT \* FROM @zpr

ORDER BY avgn DESC

OPEN zlp

DECLARE @cnt INT = 1;

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE (@cnt < @count) and (@@FETCH\_STATUS = 0)

BEGIN

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

SET @cnt = @cnt + 1;

END;

DECLARE @GOD decimal(4,2)

SET @GOD = @avgn

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

if @GOD = @avgn

BEGIN

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE (@@FETCH\_STATUS = 0)

BEGIN

if (@avgn = @GOD)

INSERT INTO @ideallocate (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END;

END;

CLOSE zlp

DEALLOCATE zlp

END ELSE

INSERT INTO @ideallocate (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (NULL,NULL,NULL,NULL,NULL)

SELECT ID as [id\_statement],sr\_nm as surname,fr\_nm as first\_name,md\_nm as middle\_name,avgn FROM @ideallocate

END

GO

CREATE PROCEDURE sel\_count\_kurs

AS

SELECT COUNT(\*) FROM abiture

WHERE (training\_courses = 1)

GO

CREATE PROCEDURE ne\_prowedwie

@name\_spec varchar(100),

@prior int,

@count int

AS

BEGIN

if object\_id(N'tempdb..#t1',N'U') is not null drop table #t1

CREATE TABLE #t1(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

insert #t1

exec prowedwie @name\_spec, @prior, @count

if @prior = 0

BEGIN

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE ((dbo.specialty.name\_spec = @name\_spec) and (abiture.id\_statement not in (select twe.ID FROM #t1 as [twe])))

ORDER BY dbo.atestat.avgn DESC

END ELSE

BEGIN

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE ((dbo.specialty.name\_spec = @name\_spec) and dbo.priem.priority = @prior and (abiture.id\_statement not in (select twe.ID FROM #t1 as [twe])))

ORDER BY dbo.atestat.avgn DESC

END

END

GO

CREATE PROCEDURE [dbo].[gen\_count\_prow]

@name\_spec varchar(50),

@prior int,

@count int

AS

BEGIN

DECLARE @not\_avg decimal(4,2)

DECLARE @id\_s INT

DECLARE @sr\_nm varchar(50)

DECLARE @fr\_nm varchar(50)

DECLARE @md\_nm varchar(50)

DECLARE @avgn decimal(4,2)

DECLARE @docnb varchar(20)

DECLARE @zpr table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

if @prior <> 0

Begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and dbo.priem.priority = @prior)

ORDER BY dbo.atestat.avgn DESC

end

else

begin

DECLARE zlp SCROLL CURSOR FOR

SELECT dbo.abiture.id\_statement, dbo.abiture.surname, dbo.abiture.first\_name, dbo.abiture.middle\_name, dbo.atestat.avgn, benefits.document\_number

FROM dbo.abiture INNER JOIN dbo.priem ON dbo.abiture.id\_statement = dbo.priem.id\_statement

INNER JOIN dbo.specialty ON dbo.priem.id\_spec = dbo.specialty.id\_spec

INNER JOIN dbo.atestat ON dbo.abiture.id\_statement = dbo.atestat.id\_statement

FULL JOIN dbo.benefits ON dbo.benefits.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec)

ORDER BY dbo.atestat.avgn DESC

end

OPEN zlp

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

WHILE @@FETCH\_STATUS = 0

BEGIN

DECLARE @iluwka decimal(4,2)

if @docnb is not null

SET @iluwka = @avgn + 10.00 else

SET @iluwka = @avgn

INSERT INTO @zpr (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @iluwka)

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn, @docnb

END

CLOSE zlp

DEALLOCATE zlp

DECLARE @zpr1 table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

DECLARE @zapas table

(

ID int,

sr\_nm varchar(50),

fr\_nm varchar(50),

md\_nm varchar(50),

avgn decimal(4,2)

)

DECLARE @cntzt INT

if @prior <> 0

Begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ FROM @zpr)

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zpr

ORDER BY avgn DESC

end

else

begin

SET @cntzt = (SELECT TOP(@count+1) COUNT(\*) AS CNTZ FROM @zpr)

DECLARE zlp SCROLL CURSOR FOR

SELECT TOP(@count+1) ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zpr

ORDER BY avgn DESC

end

OPEN zlp

FETCH LAST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

if @cntzt>@count SET @not\_avg = @avgn else SET @not\_avg = 0

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0

BEGIN

if not(@avgn = @not\_avg)

Begin

DECLARE @bubl decimal(4,2)

if @avgn>10 SET @bubl = @avgn - 10.00

else SET @bubl = @avgn

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @bubl)

end else

Begin

DECLARE @pub decimal(4,2)

if @avgn>10 SET @pub = @avgn - 10.00

else SET @pub = @avgn

INSERT INTO @zapas(ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @pub)

end

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

CLOSE zlp

DEALLOCATE zlp

DECLARE @real\_count INT

SET @real\_count = (SELECT COUNT(\*) FROM @zpr1)

IF @count > @real\_count

Begin

DECLARE zlp SCROLL CURSOR FOR

SELECT ID, sr\_nm, fr\_nm, md\_nm, avgn

FROM @zapas as [zp]

ORDER BY (SELECT bb.avg\_oge FROM oge as[bb] WHERE (bb.id\_statement = zp.ID)) DESC

OPEN zlp

DECLARE @old\_s INT

SET @old\_s = 0

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0 and @count > @real\_count

BEGIN

DECLARE @zip bit

SET @zip =(SELECT training\_courses FROM dbo.abiture

WHERE (id\_statement = @id\_s))

if @old\_s <> @avgn

BEGIN

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

END else

BEGIN

if @zip = 1

Begin

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

end

END

SET @old\_s = @avgn

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

FETCH FIRST FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

WHILE @@FETCH\_STATUS = 0 and @count > @real\_count

BEGIN

SET @zip =(SELECT training\_courses FROM dbo.abiture

WHERE (id\_statement = @id\_s))

if @old\_s = @avgn

BEGIN

if @zip = 0

Begin

SET @real\_count = @real\_count + 1

INSERT INTO @zpr1 (ID,sr\_nm,fr\_nm,md\_nm,avgn)

VALUES (@id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn)

end

END

SET @old\_s = @avgn

FETCH NEXT FROM zlp INTO @id\_s, @sr\_nm, @fr\_nm, @md\_nm, @avgn

END

CLOSE zlp

DEALLOCATE zlp

END

DECLARE @pzkwd bit

if @prior = 1 SET @pzkwd = 1 else SET @pzkwd = 0

DECLARE @bgl INT

if @count<=(SELECT COUNT(\*)

FROM dbo.specialty

INNER JOIN dbo.priem ON dbo.specialty.id\_spec = dbo.priem.id\_spec

INNER JOIN dbo.abiture ON dbo.priem.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and ((@pzkwd = 0)

or (@pzkwd = 1 and not(dbo.priem.priority in (2,3))))))

SET @bgl = (SELECT COUNT(\*)

FROM dbo.specialty

INNER JOIN dbo.priem ON dbo.specialty.id\_spec = dbo.priem.id\_spec

INNER JOIN dbo.abiture ON dbo.priem.id\_statement = dbo.abiture.id\_statement

WHERE (dbo.specialty.name\_spec = @name\_spec and ((@pzkwd = 0)

or (@pzkwd = 1 and not(dbo.priem.priority in (2,3)))))) - @count else SET @bgl = 0

DECLARE @PHP Table

(zz varchar(20))

INSERT @PHP

EXEC proh\_ball @name\_spec,@prior,@count

INSERT INTO abit\_prow (name\_spec, places, prowed, ne\_prowed, proh\_ball)

VALUES (@name\_spec, @count, (SELECT COUNT(\*) FROM @zpr1),@bgl, (SELECT TOP 1 \* FROM @PHP))

END

GO

CREATE PROCEDURE [dbo].[gen\_all\_prow]

@prior int

AS

BEGIN

DELETE FROM abit\_prow

DECLARE @name\_spec varchar(50)

DECLARE @places INT

DECLARE zlpz SCROLL CURSOR FOR

SELECT name\_spec,places FROM specialty

OPEN zlpz

FETCH FIRST FROM zlpz INTO @name\_spec, @places

WHILE @@FETCH\_STATUS = 0

BEGIN

EXEC gen\_count\_prow @name\_spec, @prior, @places

FETCH NEXT FROM zlpz INTO @name\_spec, @places

END

CLOSE zlpz

DEALLOCATE zlpz

END

GO

CREATE PROCEDURE [dbo].[see\_all\_table]

AS

BEGIN

SELECT abiture.\*,benefits.name,benefits.document\_number, pasport.series, pasport.numb,pasport.place\_of\_birth,pasport.date\_of\_issue

FROM abiture

FULL JOIN benefits ON benefits.id\_statement = abiture.id\_statement

INNER JOIN pasport ON pasport.id\_statement = abiture.id\_statement

END

GO

CREATE PROCEDURE [dbo].[add\_users]

(

@username varchar(50),

@userrole varchar(50),

@pbable varchar(50)

)

AS

BEGIN

EXEC('CREATE LOGIN '+@username+' WITH PASSWORD = N'''+@pbable+''', DEFAULT\_DATABASE=[priemka], CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON '+

'CREATE USER '+@username+' FOR LOGIN '+@username)

EXEC sp\_addrolemember @userrole, @username

END

GO

CREATE PROCEDURE [dbo].[update\_users]

(

@oldname varchar(50),

@username varchar(50),

@userrole varchar(50),

@pbable varchar(50)

)

AS

BEGIN

exec dbo.drop\_users @oldname

exec dbo.add\_users @username, @userrole, @pbable

END

GO

CREATE PROCEDURE [dbo].[drop\_users]

(

@username varchar(50)

)

AS

BEGIN

exec sp\_dropuser @username

exec sp\_droplogin @username

END

GO

CREATE VIEW sel\_user\_deathwing AS

select DISTINCT(convert(char(25),noda1.name))as [Пользователи],

convert(char(25),noda3.name) as [Роль] from sys.database\_principals AS noda1

INNER JOIN sys.database\_role\_members AS noda2 ON

noda1.principal\_id =noda2.member\_principal\_id

INNER JOIN sys.database\_principals AS noda3 ON

noda2.role\_principal\_id = noda3.principal\_id

WHERE (noda1.type\_desc = 'SQL\_USER')and

(noda1.principal\_id between 6 and 16000)

GO

USE [priemka]

GO

CREATE ROLE [Secretar]

GO

GRANT INSERT ON [dbo].[benefits] TO [Secretar]

GO

GRANT SELECT ON [dbo].[benefits] TO [Secretar]

GO

GRANT ALTER ON [dbo].[benefits] TO [Secretar]

GO

GRANT DELETE ON [dbo].[benefits] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[prowedwie] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[proh\_ball] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[add\_priem] TO [Secretar]

GO

GRANT INSERT ON [dbo].[pasport] TO [Secretar]

GO

GRANT SELECT ON [dbo].[pasport] TO [Secretar]

GO

GRANT ALTER ON [dbo].[pasport] TO [Secretar]

GO

GRANT DELETE ON [dbo].[pasport] TO [Secretar]

GO

GRANT INSERT ON [dbo].[oge] TO [Secretar]

GO

GRANT SELECT ON [dbo].[oge] TO [Secretar]

GO

GRANT ALTER ON [dbo].[oge] TO [Secretar]

GO

GRANT DELETE ON [dbo].[oge] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[sel\_title] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[priem\_doz] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[sel\_count\_kurs] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[sel\_avg\_atest] TO [Secretar]

GO

GRANT INSERT ON [dbo].[pasp\_rod] TO [Secretar]

GO

GRANT SELECT ON [dbo].[pasp\_rod] TO [Secretar]

GO

GRANT ALTER ON [dbo].[pasp\_rod] TO [Secretar]

GO

GRANT DELETE ON [dbo].[pasp\_rod] TO [Secretar]

GO

GRANT INSERT ON [dbo].[document] TO [Secretar]

GO

GRANT SELECT ON [dbo].[document] TO [Secretar]

GO

GRANT ALTER ON [dbo].[document] TO [Secretar]

GO

GRANT DELETE ON [dbo].[document] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[add\_doc] TO [Secretar]

GO

GRANT INSERT ON [dbo].[documents] TO [Secretar]

GO

GRANT SELECT ON [dbo].[documents] TO [Secretar]

GO

GRANT ALTER ON [dbo].[documents] TO [Secretar]

GO

GRANT DELETE ON [dbo].[documents] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[doomsayer] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[see\_all\_table] TO [Secretar]

GO

GRANT INSERT ON [dbo].[abiture] TO [Secretar]

GO

GRANT SELECT ON [dbo].[abiture] TO [Secretar]

GO

GRANT ALTER ON [dbo].[abiture] TO [Secretar]

GO

GRANT DELETE ON [dbo].[abiture] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[ne\_prowedwie] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[gen\_count\_prow] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[gen\_all\_prow] TO [Secretar]

GO

GRANT INSERT ON [dbo].[abit\_prow] TO [Secretar]

GO

GRANT SELECT ON [dbo].[abit\_prow] TO [Secretar]

GO

GRANT ALTER ON [dbo].[abit\_prow] TO [Secretar]

GO

GRANT DELETE ON [dbo].[abit\_prow] TO [Secretar]

GO

GRANT INSERT ON [dbo].[atestat] TO [Secretar]

GO

GRANT SELECT ON [dbo].[atestat] TO [Secretar]

GO

GRANT ALTER ON [dbo].[atestat] TO [Secretar]

GO

GRANT DELETE ON [dbo].[atestat] TO [Secretar]

GO

GRANT EXECUTE ON [dbo].[polu\_proh] TO [Secretar]

GO

GRANT INSERT ON [dbo].[specialty] TO [Secretar]

GO

GRANT SELECT ON [dbo].[specialty] TO [Secretar]

GO

GRANT ALTER ON [dbo].[specialty] TO [Secretar]

GO

GRANT DELETE ON [dbo].[specialty] TO [Secretar]

GO

GRANT INSERT ON [dbo].[priem] TO [Secretar]

GO

GRANT SELECT ON [dbo].[priem] TO [Secretar]

GO

GRANT ALTER ON [dbo].[priem] TO [Secretar]

GO

GRANT DELETE ON [dbo].[priem] TO [Secretar]

GO

CREATE ROLE [Star\_Secretar]

GO

GRANT INSERT ON [dbo].[abit\_prow] TO [Star\_Secretar]

GO

GRANT SELECT ON [dbo].[abit\_prow] TO [Star\_Secretar]

GO

GRANT ALTER ON [dbo].[abit\_prow] TO [Star\_Secretar]

GO

GRANT DELETE ON [dbo].[abit\_prow] TO [Star\_Secretar]

GO

GRANT INSERT ON [dbo].[document] TO [Star\_Secretar]

GO

GRANT SELECT ON [dbo].[document] TO [Star\_Secretar]

GO

GRANT DELETE ON [dbo].[document] TO [Star\_Secretar]

GO

GRANT UPDATE ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT VIEW DEFINITION ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT DELETE ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT INSERT ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT SELECT ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT ALTER ON [dbo].[specialty] TO [Star\_Secretar]

GO

GRANT UPDATE ON [dbo].[abit\_prow] TO [Star\_Secretar]

GO

GRANT ALTER ON [dbo].[document] TO [Star\_Secretar]

GO

GRANT UPDATE ON [dbo].[document] TO [Star\_Secretar]

GO

CREATE ROLE [Admun]

GO

GRANT INSERT TO [Admun]

GO

GRANT SELECT TO [Admun]

GO

GRANT EXECUTE TO [Admun]

GO

GRANT ALTER TO [Admun]

GO

GRANT UPDATE TO [Admun]

GO

GRANT DELETE TO [Admun]

GO

GRANT CONTROL TO [Admun]

GO

GRANT ALTER ANY DATABASE AUDIT TO [Admun]

GO

GRANT ALTER ANY USER TO [Admun]

GO

CREATE LOGIN [secr\_user] WITH PASSWORD=N'bubl11', DEFAULT\_DATABASE=[priemka], CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON

GO

CREATE USER [secr\_user] FOR LOGIN [secr\_user]

GO

EXEC sp\_addrolemember N'Secretar', N'secr\_user'

GO

USE [master]

GO

CREATE LOGIN [star\_user] WITH PASSWORD=N'doom22', DEFAULT\_DATABASE=[priemka], CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON

GO

USE [priemka]

GO

CREATE USER [star\_user] FOR LOGIN [star\_user]

GO

EXEC sp\_addrolemember N'Star\_Secretar', N'star\_user'

GO

USE [master]

GO

CREATE LOGIN [admn\_user] WITH PASSWORD=N'slay33', DEFAULT\_DATABASE=[priemka], CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON

GO

USE [priemka]

GO

CREATE USER [admn\_user] FOR LOGIN [admn\_user]

GO

EXEC sp\_addrolemember N'Admun', N'admn\_user'

EXEC master..sp\_addsrvrolemember @loginame = N'admn\_user', @rolename = N'securityadmin'

GO

INSERT INTO abiture(surname,first\_name,middle\_name,is\_male,birthday,phone,address,training\_courses,date\_of\_application) VALUES

('Субботин','Юрий','Степанович','true','09.02.1996',79292406219,'г.Москва, пер.Никитников, д.3 кв.14','true','22.03.2016'),

('Смирнов','Святослав','Романович','true','07.04.1992',79259446959,'г.Москва, Покровский б-р,7 кв.88','false','22.03.2016'),

('Тарасов','Виталий','Сергеевич','true','26.06.1994',79263388721,'г.Москва, ул.Трудовая д.16 кв.69','false','22.03.2016'),

('Белозёрова','Галина','Анатольевна','false','19.07.1995',79263388721,'г.Москва, ул.Маршала Тухачевского 58 кв.9','false','23.03.2016'),

('Чудова','Дарья','Николаевна','false','12.03.1997',79264472111,'г.Химки, ул.Пожарского д.9 кв.195','true','23.03.2016'),

('Ядрихинский','Александр','Титович','true','17.11.1996',79252406219,'г.Москва, ул.Чертановская д.23 кв.9','false','23.03.2016'),

('Лапухова','Юлия','Геннадиевна','false','29.01.1996',79251234567,'г.Москва, ул.Октябрьская д.40 кв.7','false','23.03.2016'),

('Митин','Богдан','Савелиевич','true','13.02.1996',79269863452,'г.Москва, ул.Полковая д.34 кв.127','true','24.03.2016'),

('Неретин','Илья','Андроникович','true','18.12.1994',79295633462,'г.Москва, ул.Сумогатная д.3\8 кв.11','false','24.03.2016'),

('Пережогин','Никита','Олегович','true','11.03.1997',79269851653,'г.Люберцы, ул.Космонавтов 26 кв.71','false','24.03.2016')

GO

INSERT INTO benefits(id\_statement,name,document\_number) VALUES

(3,'Сирота',141322),

(9,'Инвалид',6565656)

GO

INSERT INTO pasport(id\_statement,series,numb,place\_of\_birth,issued\_by,date\_of\_issue) VALUES

(1,4617,322666,'пер.Никитников, д.3 кв.14','ТП №3 В Гор.Москва ОУФМС России по Москве','11.11.2011'),

(2,4655,777333,'Покровский б-р,7 кв.88','ТП №2 В Гор.Москва ОУФМС России по Москве','12.12.2012'),

(3,4692,230198,'ул.Трудовая д.16 кв.69','ТП №4 В Гор.Москва ОУФМС России по Москве','23.01.2009'),

(4,4677,980022,'ул.Маршала Тухачевского 58 кв.9','ТП №3 В Гор.Москва ОУФМС России по Москве','22.09.2011'),

(5,4614,143728,'ул.Пожарского д.9 кв.195','ТП №1 В Гор.Химки ОУФМС России по МО','29.11.2011'),

(6,1438,123456,'ул.Чертановская д.23 кв.9','ТП №1 В Гор.Москва ОУФМС России по Москве','19.12.2010'),

(7,4621,654321,'ул.Октябрьская д.40 кв.7','ТП №1 В Гор.Люберцы ОУФМС России по МО','23.01.2010'),

(8,4667,928374,'ул.Полковая д.34 кв.127','ТП №1 В Гор.Химки ОУФМС России по МО','17.08.2009'),

(9,4677,109822,'ул.Сумогатная д.3\8 кв.11','ТП №1 В Гор.Москва ОУФМС России по Москве','12.11.2015'),

(10,4611,148800,'Москва ул.Космонавтов 26 кв.71','ТП №1 В Гор.Люберцы ОУФМС России по МО','29.03.2011')

GO

INSERT INTO document(name) VALUES

('Заявление'),

('Аттестат'),

('Паспорт абитуриента'),

('Паспорт одного из родителей'),

('Медицинская справка 086/у'),

('Ксерокопия полиса медицинского страхования'),

('Выписка результатов ГИА\ЕГЭ')

GO

INSERT INTO documents(id\_statement,numb\_doc) VALUES

(1,1),(1,2),(1,3),(1,4),(1,5),(1,6),

(2,1),(2,2),(2,3),(2,4),(2,5),(2,6),

(3,1),(3,2),(3,3),(3,4),(3,5),(3,6),(3,7),

(4,1),(4,2),(4,3),(4,4),(4,5),(4,6),

(5,1),(5,2),(5,3),(5,4),(5,5),(5,6),

(6,1),(6,2),(6,3),(6,4),(6,5),(6,6),

(7,1),(7,2),(7,3),(7,4),(7,5),(7,6),(7,7),

(8,1),(8,2),(8,3),(8,4),(8,5),(8,6),

(9,1),(9,2),(9,3),(9,4),(9,5),(9,6),(9,7),

(10,1),(10,2),(10,3),(10,4),(10,5),(10,6)

GO

INSERT INTO pasp\_rod(id\_statement,series,numb,surname,first\_name,middle\_name,birthday,place\_of\_birth) VALUES

(1,4611,194307,'Алексеев','Ахмет','Кириллович','21.05.1965','г.Северодвинск, ул.Загородная д.25, кв.134'),

(2,4612,246827,'Осипов','Георгий','Богданович','14.03.1968','г.Москва, Покровский б-р д.7 кв.88'),

(3,4613,315075,'Дмитриев','Роман','Павлович','17.02.1969','г.Москва, ул.Айвазовского д.93 кв.119'),

(4,4614,614749,'Пономарёва','Ангелина','Егоровна','10.06.1972','г.Озера, ул.Бакунинская д.53, кв.9'),

(5,4615,195889,'Антонов','Демид','Максимович','05.10.1976','г.Москва, ул.Барклая д.40, кв.24'),

(6,4616,137431,'Тарский','Силантий','Вениаминович','23.06.1969','г.Москва, ул.Журавлёва д.5 кв.4'),

(7,4617,337121,'Быков','Валентин','Николаевич','15.01.1974','г.Москва, ул.Глинки д.8 кв.2'),

(8,4618,384997,'Жуков','Андоим','Павлович','12.02.1977','г.Москва, ул.Бабаевская д.38 кв.87'),

(9,4618,511364,'Мартынова','Клара','Ильинична','27.07.1980','г.Москва, ул.Базанова д.76 кв.94'),

(10,4619,204455,'Тюрина','Кларисса','Федоровна','11.09.1971','г.Химки, ул. Басовская д.1 кв.16')

GO

INSERT INTO specialty(name\_spec,code\_spec,places) VALUES

('Компьютерные сети','09.02.02',30),

('Программирование в компьютерных системах','09.02.03',40),

('Прикладная информатика','09.02.05',4),

('Информационная безопасность автоматизированных систем','10.02.03',25),

('Земельно-имущественные отношения','21.02.01',30),

('Реклама','42.02.01',20)

GO

INSERT INTO priem(id\_statement,id\_spec,priority) VALUES

(1,2,1),(1,4,2),(1,3,3),

(2,6,1),(2,5,2),(2,4,3),

(3,2,1),(3,3,2),(3,4,3),

(4,1,1),(4,2,2),(4,3,3),

(5,6,1),(5,5,2),(5,4,3),

(6,2,1),(6,4,2),(6,6,3),

(7,1,1),(7,3,2),(7,5,3),

(8,4,1),(8,3,2),(8,5,3),

(9,3,1),(9,6,2),(9,1,3),

(10,4,1),(10,3,2),(10,2,3)

GO

INSERT INTO atestat(id\_statement,numb\_at,institution,s\_otl,avgn,date\_of\_issue) VALUES

(1,124257,'Гимназия «Классика»','false',4,'23.11.2015'),

(2,235251,'Гимназия «Утро»','false',3,'25.05.2015'),

(3,235252,'Гимназия «Ясенево»','true',5,'21.05.2015'),

(4,876457,'Гимназия «Доверие»','true',5,'03.06.2015'),

(5,952421,'Гимназия «Утро»','false',4.5,'25.05.2015'),

(6,754782,'Гимназия №1409','false',3.5,'28.05.2015'),

(7,182653,'Гимназия №1505','false',4,'26.05.2015'),

(8,322666,'Гимназия №1515','false',3.5,'29.05.2015'),

(9,777666,'Гимназия «Алфавит»','false',3,'28.05.2015'),

(10,848521,'Гимназия №1781','false',3,'06.06.2015')

GO

INSERT INTO oge(id\_statement,numb\_sv,avg\_oge,date\_of\_issue) VALUES

(1,246262,4,'15.06.2015'),

(2,246262,3,'15.06.2015'),

(3,346543,3,'11.06.2015'),

(4,246262,5,'15.06.2015'),

(5,246262,5,'11.06.2015'),

(6,246262,5,'15.06.2015'),

(7,246262,5,'15.06.2015'),

(8,246262,5,'15.06.2015'),

(9,246262,5,'15.06.2015'),

(10,674712,5,'17.06.2015')

GO

**Приложение 2, код программы.**

program Rapid\_Z;

//Автор Орищенко Андрей П-304.

uses Forms,

AuthUnit in 'Units\AuthUnit.pas' {AuthForm},

MainUserUnit in 'Units\MainUserUnit.pas' {MainUserForm},

Add\_obwee in 'Units\Add\_obwee.pas' {AddObweeForm},

NetAPI in 'Units\netapi.pas',

ForMeMComp in 'Units\ForMeMComp.pas' {CoNSQL: TDataModule},

add\_abiture in 'Units\add\_abiture.pas' {AddStudForm},

pasport in 'Units\pasport.pas' {PasportForm},

Doc\_unit in 'Units\Doc\_unit.pas' {DocForm},

pasp\_rod in 'Units\pasp\_rod.pas' {PaspRodForm},

atestat in 'Units\atestat.pas' {Atestat},

OGE in 'Units\OGE.pas' {OGEG},

Priem in 'Units\Priem.pas' {Priemform},

Lgot in 'Units\Lgot.pas' {LgotForm},

SeeUnit in 'units\SeeUnit.pas' {SeeForm},

Prohball in 'units\Prohball.pas' {ProhForm},

StForm in 'units\StForm.pas' {StatistForm},

ATableForm in 'units\ATableForm.pas' {AllTableForm},

ChangePassUnit in 'units\ChangePassUnit.pas' {ChangePassForm},

UserUnit in 'units\UserUnit.pas' {UserForm};

{$R \*.res}

begin

Application.Initialize;

Application.Title := 'Áàçà ïðèåìíîé êîìèññèè êîëëåäæà';

Application.CreateForm(TAuthForm, AuthForm);

Application.CreateForm(TMainUserForm, MainUserForm);

Application.CreateForm(TAddObweeForm, AddObweeForm);

Application.CreateForm(TCoNSQL, CoNSQL);

Application.CreateForm(Taddstudform, addstudform);

Application.CreateForm(TPasportForm, PasportForm);

Application.CreateForm(TDoc\_form, Doc\_form);

Application.CreateForm(TPasp\_rod\_Form, Pasp\_rod\_Form);

Application.CreateForm(TAtest\_form, Atest\_form);

Application.CreateForm(TOGEG, OGEG);

Application.CreateForm(TPriem\_form, Priem\_form);

Application.CreateForm(TLgotForm, LgotForm);

Application.CreateForm(TSeeForm, SeeForm);

Application.CreateForm(TProhForm, ProhForm);

Application.CreateForm(TStatistForm, StatistForm);

Application.CreateForm(TAllTableForm, AllTableForm);

Application.CreateForm(TChangePassForm, ChangePassForm);

Application.CreateForm(TForm4, Form4);

Application.ShowMainForm:=false;

AuthForm.Show;

Application.Run;

end.

unit AuthUnit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Buttons, inifiles;

type

TAuthForm = class(TForm)

Label1: TLabel;

Label2: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Button1: TButton;

Label3: TLabel;

ComboBox1: TComboBox;

SpeedButton1: TSpeedButton;

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure Button1Click(Sender: TObject);

procedure Edit1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure FormCreate(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure SpeedButton1Click(Sender: TObject);

procedure ComboBox1Exit(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var AuthForm: TAuthForm;

implementation

uses ForMeMComp, MainUserUnit, ADODB, Netapi, atestat;

{$R \*.dfm}

procedure TAuthForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

application.Terminate;

end;

procedure TAuthForm.Button1Click(Sender: TObject);

var tmp:integer;

begin

MainUserForm.Npravka.Visible:=false;

MainUserForm.Nsee.Visible:=false;

MainUserForm.Nstar\_sekt.Visible:=false;

MainUserForm.Nstar\_admin.Visible:=false;

MainUserForm.Label2.Visible:=false;

MainUserForm.Edit1.Visible:=false;

Label3.Caption:='';

if Edit1.Text='' then Label3.Caption:='Ââåäèòå ëîãèí.' else

if Edit2.Text='' then Label3.Caption:='Ââåäèòå ïàðîëü.' else

Begin

CoNSQL.ADOConnection1.ConnectionString:='Provider=SQLOLEDB.1;Password='+Edit2.Text+

';Persist Security Info=True;User ID='+Edit1.Text+

';Initial Catalog=priemka;Data Source='+combobox1.Items[combobox1.ItemIndex];;

try

CoNSQL.adoconnection1.Connected:=true;

except

begin

Label3.Caption:='Îøèáêà àâòîðèçàöèè.';

end;

end;

end;

if CoNSQL.ADOConnection1.Connected then

Begin

if (Edit1.Text = 'sa') then

Begin

showmessage('Access denied');

halt;

end else

Begin

CoNSQL.CheckRole.SQL.Text:='declare @i varchar(50); set @i = SYSTEM\_USER; exec sp\_helpuser @i';

CoNSQL.CheckRole.Open;

CoNSQL.CheckRole.First;

if not(CoNSQL.CheckRole.IsEmpty) then

Begin

CoNSQL.UserName:=CoNSQL.CheckRole.FieldByName('UserName').AsString;

CoNSQL.UserRole:=CoNSQL.CheckRole.FieldByName('RoleName').AsString;

if (CoNSQL.CheckRole.FieldByName('RoleName').Asstring = 'Admun') then

Begin

MainUserForm.Nstar\_admin.Visible:=true;

MainUserForm.Caption:='RapidZ - Àäìèíèñòðàòîð';

end else

if (CoNSQL.CheckRole.FieldByName('RoleName').Asstring = 'Star\_Secretar') then

Begin

MainUserForm.Nstar\_sekt.Visible:=true;

MainUserForm.Caption:='RapidZ - Ñòàðøèé ñåêðåòàðü';

end else

if (CoNSQL.CheckRole.FieldByName('RoleName').Asstring = 'Secretar') then

Begin

MainUserForm.Nsee.Visible:=true;

MainUserForm.Npravka.Visible:=true;

MainUserForm.Label2.Visible:=true;

MainUserForm.Edit1.Visible:=true;

MainUserForm.Caption:='RapidZ - Cåêðåòàðü';

end;

end;

end;

IniFile.WriteString('Auth','UserLogin',Edit1.Text);

tmp:=IniFile.ReadInteger('StyleID',CoNSQL.UserName,0);

if (tmp < 0) or (tmp >= MainUserForm.ComBobox1.Items.Count) then tmp:=0;

MainUserForm.ComBobox1.ItemIndex:=tmp;

CoNSQL.SkinIsRough.SkinName:=MainUserForm.ComBobox1.Items[MainUserForm.ComBobox1.ItemIndex];

MainUserForm.ComboBox1Change(self);

Edit2.Text:='';

AuthForm.Hide;

MainUserForm.Show;

end;

end;

procedure TAuthForm.Edit1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

if key = 13 then AuthForm.Button1Click(self);

end;

procedure TAuthForm.FormCreate(Sender: TObject);

var deathwing:cardinal;

Begin

IniFile:=TIniFile.Create('./Config.ini');

combobox1.Items.Add(IniFile.ReadString('DataBase','IPDef','127.0.0.1'));

GetSQLServerNames(combobox1.Items,'',deathwing);

combobox1.ItemIndex:=0;

end;

procedure TAuthForm.FormShow(Sender: TObject);

var tmp:integer;

begin

Consql.ADOConnection1.Connected:=false;

Edit1.Text:=IniFile.ReadString('Auth','UserLogin','');

tmp:=IniFile.ReadInteger('Design','SkinID',0);

if (tmp < 0) or (tmp >= MainUserForm.ComBobox1.Items.Count) then tmp:=0;

MainUserForm.ComBobox1.ItemIndex:=tmp;

CoNSQL.SkinIsRough.SkinName:=MainUserForm.ComBobox1.Items[MainUserForm.ComBobox1.ItemIndex];

end;

procedure TAuthForm.SpeedButton1Click(Sender: TObject);

begin

IniFile.WriteString('DataBase','IPDef',Combobox1.Text);

end;

procedure TAuthForm.ComboBox1Exit(Sender: TObject);

begin

Combobox1.TabStop:=false;

end;

end.

unit ChangePassUnit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TChangePassForm = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Edit2: TEdit;

Edit3: TEdit;

Label4: TLabel;

Button1: TButton;

procedure FormShow(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var ChangePassForm: TChangePassForm;

implementation

uses ForMeMComp;

procedure TChangePassForm.FormShow(Sender: TObject);

begin

Label4.Caption:=CoNSQL.UserName;

Edit2.Text:='';

Edit3.Text:='';

end;

procedure TChangePassForm.Button1Click(Sender: TObject);

begin

CoNSQL.CheckRole.SQL.Text:='ALTER LOGIN '+Label4.Caption+' WITH PASSWORD = N'+#39+

Edit3.Text+#39+' OLD\_PASSWORD = N'+#39+Edit2.Text+#39;

try

CoNSQL.CheckRole.ExecSQL;

except

showmessage('Íå óäàëîñü ñìåíèòü ïàðîëü.');

end;

close;

end;

end.

unit Doc\_unit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, DBCtrls, StdCtrls;

type

TDoc\_form = class(TForm)

Button2: TButton;

Button1: TButton;

ListBox1: TListBox;

ListBox2: TListBox;

Button4: TButton;

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var Doc\_form: TDoc\_form;

implementation

uses ForMeMComp,add\_abiture;

procedure TDoc\_form.Button2Click(Sender: TObject);

begin

if ListBox2.ItemIndex <> -1 then

Begin

ListBox1.Items.Add(ListBox2.Items[ListBox2.ItemIndex]);

ListBox2.DeleteSelected;

if listbox2.Count>0 then ListBox2.Selected[0]:=true;

end;

end;

procedure TDoc\_form.Button1Click(Sender: TObject);

begin

if ListBox1.ItemIndex <> -1 then

Begin

ListBox2.Items.Add(ListBox1.Items[ListBox1.ItemIndex]);

ListBox1.DeleteSelected;

if listbox1.Count>0 then ListBox1.Selected[0]:=true;

end;

end;

procedure TDoc\_form.Button4Click(Sender: TObject);

begin

if (ListBox1.Count>0) then

Begin

close;

addstudform.Label3.Caption:='+';

addstudform.Label3.Font.Color:=clgreen;

end else

Begin

showmessage('Íåò äàííûõ');

end;

end;

procedure TDoc\_form.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

if not(ListBox1.Count>0) then

Begin

addstudform.Label3.Caption:='X';

addstudform.Label3.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end.

unit ForMeMComp;

interface

uses SysUtils, Classes, ADODB, DB, RpCon, RpDefine, RpRave, RpConDS, RpRender, RpRenderCanvas, RpRenderPreview, sSkinManager, RpBase, RpSystem, inifiles;

type

TCoNSQL = class(TDataModule)

ADOConnection1: TADOConnection;

ADOTable1: TADOTable;

DataSource1: TDataSource;

ADOQuery1: TADOQuery;

ADOQuery2: TADOQuery;

DataSource2: TDataSource;

ADOTable2: TADOTable;

ADOQuery3: TADOQuery;

DataSource3: TDataSource;

ADOQuery4: TADOQuery;

RvDataSetConnection1: TRvDataSetConnection;

MyRave: TRvProject;

CheckRole: TADOQuery;

TempQuery: TADOQuery;

RvDtStCoN2: TRvDataSetConnection;

SkinIsRough: TsSkinManager;

procedure ADOQuery3BeforeInsert(DataSet: TDataSet);

private

{ Private declarations }

public

UserName,UserRole:string;

end;

var CoNSQL: TCoNSQL; IniFile : TIniFile;

implementation

procedure TCoNSQL.ADOQuery3BeforeInsert(DataSet: TDataSet);

begin

if CoNSQL.UserRole = 'Star\_Secretar' then

Begin

end else

abort;

end;

end.

unit Lgot;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TLgotForm = class(TForm)

Label1: TLabel;

Edit1: TEdit;

Label2: TLabel;

Edit2: TEdit;

Button2: TButton;

Button1: TButton;

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var LgotForm: TLgotForm;

implementation

uses add\_abiture;

procedure TLgotForm.Button2Click(Sender: TObject);

begin

addstudform.t\_is\_lgt;

close;

end;

procedure TLgotForm.Button1Click(Sender: TObject);

begin

addstudform.f\_is\_lgt;

close;

end;

procedure TLgotForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

if addstudform.g\_is\_lgt then

Begin

if not((Edit1.Text<>'') and (Edit2.Text<>'')) then

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

end;

end.

unit MainUserUnit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Menus, Grids, DBGrids, acDBGrid;

type

TMainUserForm = class(TForm)

MainMenu1: TMainMenu;

Npravka: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

Nsee: TMenuItem;

N8: TMenuItem;

N9: TMenuItem;

N12: TMenuItem;

N13: TMenuItem;

N14: TMenuItem;

N22: TMenuItem;

N26: TMenuItem;

N27: TMenuItem;

N28: TMenuItem;

Nstar\_sekt: TMenuItem;

N40: TMenuItem;

N41: TMenuItem;

Nstar\_admin: TMenuItem;

Edit1: TEdit;

Label2: TLabel;

N4: TMenuItem;

N6: TMenuItem;

RapidZ1: TMenuItem;

N1: TMenuItem;

N5: TMenuItem;

N10: TMenuItem;

Ndoc: TMenuItem;

Nspec: TMenuItem;

ComboBox1: TComboBox;

DBGrid1: TsDBGrid;

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure FormShow(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure sDBGrid1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure Edit1Change(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure DBGrid1TitleClick(Column: TColumn);

procedure N9Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N27Click(Sender: TObject);

procedure N28Click(Sender: TObject);

procedure N24Click(Sender: TObject);

procedure N25Click(Sender: TObject);

procedure N12Click(Sender: TObject);

procedure N13Click(Sender: TObject);

procedure N14Click(Sender: TObject);

procedure N6Click(Sender: TObject);

procedure N1Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure N10Click(Sender: TObject);

procedure NdocClick(Sender: TObject);

procedure NspecClick(Sender: TObject);

procedure N40Click(Sender: TObject);

procedure N41Click(Sender: TObject);

procedure N22Click(Sender: TObject);

procedure ComboBox1Change(Sender: TObject);

procedure DBGrid1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

Private see\_abit\_atest:byte;

Public uwpdate:boolean; procedure aborted; function show\_your\_atest:byte;

end;

var MainUserForm: TMainUserForm; rd:boolean;

implementation

uses ForMeMComp, Add\_obwee, AuthUnit, add\_abiture, pasport, Doc\_unit,

pasp\_rod, Priem, SeeUnit, Prohball, StForm, ATableForm, ChangePassUnit, UserUnit,

OGE, Lgot, atestat;

procedure TMainUserForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

application.Terminate;

end;

procedure TMainUserForm.FormShow(Sender: TObject);

Begin

try

DBGrid1.ReadOnly:=true;

if (CoNSQL.UserRole='Secretar') then

Begin

CoNSQL.ADOQuery3.SQL.Text:='exec sel\_title :1';

CoNSQL.ADOQuery3.Parameters.ParseSQL(CoNSQL.ADOQuery3.SQL.Text,true);

CoNSQL.ADOQuery3.Parameters.ParamByName('1').Value:= '';

CoNSQL.ADOQuery3.Open;

DBGrid1.Columns[0].Width:=30;

DBGrid1.Columns[1].Width:=100;

DBGrid1.Columns[2].Width:=75;

DBGrid1.Columns[3].Width:=100;

DBGrid1.Columns[6].Width:=275;

end;

if (CoNSQL.UserRole='Star\_Secretar') then

Begin

CoNSQL.ADOQuery3.SQL.Text:='SELECT name\_spec AS [Íàçâàíèå], code\_spec AS [Êîä],'+

'places AS [Ìåñò] FROM specialty';

CoNSQL.ADOQuery3.Open;

DBGrid1.ReadOnly:=false;

DBGrid1.Columns[0].Width:=450;

DBGrid1.Columns[1].Width:=150;

DBGrid1.Columns[2].Width:=75;

end;

if (CoNSQL.UserRole='Admun') then

Begin

CoNSQL.ADOQuery3.SQL.Text:='SELECT \* FROM sel\_user\_deathwing';

CoNSQL.ADOQuery3.Open;

DBGrid1.ReadOnly:=false;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N2Click(Sender: TObject);

begin

try

uwpdate:=false;

with addstudform do

Begin

is\_insert:=true;

Label1.Caption:='X';

Label1.Font.Color:=clred;

Label2.Caption:='X';

Label2.Font.Color:=clred;

Label3.Caption:='X';

Label3.Font.Color:=clred;

Label4.Caption:='X';

Label4.Font.Color:=clred;

Label5.Caption:='X';

Label5.Font.Color:=clred;

Label6.Caption:='X';

Label6.Font.Color:=clred;

Label7.Caption:='X';

Label7.Font.Color:=clred;

f\_is\_lgt;

end;

with AddObweeForm do

Begin

edit1.Text:='';

edit2.Text:='';

edit3.Text:='';

Edit5.date:=Date;

edit6.Text:='';

edit7.Text:='';

Checkbox1.Checked:=false;

end;

with PasportForm do

Begin

edit1.Text:='';

edit2.Text:='';

edit3.Text:='';

edit4.Text:='';

Edit5.date:=Date;

end;

with Doc\_form do

Begin

ListBox1.Clear;

ListBox2.Clear;

CoNSQL.ADOQuery1.SQL.Text:='SELECT name FROM document';

CoNSQL.ADOQuery1.Open;

While not consql.ADOQuery1.Eof do

begin

ListBox2.Items.Add(consql.ADOQuery1.FieldByName('name').AsString);

consql.ADOQuery1.Next;

end;

end;

with Pasp\_rod\_Form do

Begin

edit1.Text:='';

edit2.Text:='';

edit3.Text:='';

Edit5.date:=Date;

edit6.Text:='';

edit7.Text:='';

edit8.Text:='';

end;

with Atest\_form do

Begin

edit1.Text:='';

edit2.Text:='';

edit3.Text:='';

CheckBox1.checked:=false;

edit5.date:=Date;

end;

with OGEG do

Begin

edit1.Text:='';

edit3.Text:='';

edit5.date:=Date;

end;

with Priem\_form do

Begin

CoNSQL.adoquery2.SQL.Clear;

ComboBox1.Clear;

ComboBox2.Clear;

ComboBox3.Clear;

CoNSQL.adoquery2.SQL.add('SELECT name\_spec AS VGA FROM specialty');

CoNSQL.adoquery2.open;

while not(CoNSQL.adoquery2.Eof) and (CoNSQL.adoquery2.RecordCount>0) do

Begin

ComboBox1.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

ComboBox2.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

ComboBox3.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

CoNSQL.adoquery2.Next;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

with LgotForm do

Begin

edit1.Text:='';

edit2.Text:='';

end;

addstudform.ShowModal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.sDBGrid1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

try

if (key = 13) then

Begin

CoNSQL.ADOQuery3.Edit;

CoNSQL.ADOQuery3.Post;

end;

if (key = 46) and (ssShift in shift) then

Begin

if not(CoNSQL.ADOQuery3.IsEmpty) then

Begin

if CoNSQL.UserRole = 'Admun' then

Begin

with CoNSQL do

Begin

if CoNSQL.UserName <> trim(adoquery3.Fields.Fields[0].asstring) then

Begin

ADOQuery4.SQL.Clear;

ADOQuery4.SQL.Text:='exec drop\_users :1';

ADOQuery4.Parameters.ParseSQL(ADOQuery4.SQL.Text,true);

ADOQuery4.Parameters.ParamByName('1').Value:=trim(adoquery3.Fields.Fields[0].asstring);

ADOQuery4.ExecSQL;

ADOQuery3.Active:=false;

ADOQuery3.Active:=true;

end else showmessage('Access denied');

end;

end else

CoNSQL.ADOQuery3.Delete;

End;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.Edit1Change(Sender: TObject);

begin

try

if Edit1.Text<>'' then

Begin

CoNSQL.ADOQuery3.SQL.Text:='exec sel\_title :1';

CoNSQL.ADOQuery3.Parameters.ParseSQL(CoNSQL.ADOQuery3.SQL.Text,true);

CoNSQL.ADOQuery3.Parameters.ParamByName('1').Value:= Edit1.Text;

CoNSQL.ADOQuery3.Open;

DBGrid1.Columns[0].Width:=30;

DBGrid1.Columns[1].Width:=100;

DBGrid1.Columns[2].Width:=75;

DBGrid1.Columns[3].Width:=100;

DBGrid1.Columns[6].Width:=275;

end else

Begin

CoNSQL.ADOQuery3.SQL.Text:='exec sel\_title :1';

CoNSQL.ADOQuery3.Parameters.ParseSQL(CoNSQL.ADOQuery3.SQL.Text,true);

CoNSQL.ADOQuery3.Parameters.ParamByName('1').Value:= '';

CoNSQL.ADOQuery3.Open;

DBGrid1.Columns[0].Width:=30;

DBGrid1.Columns[1].Width:=100;

DBGrid1.Columns[2].Width:=75;

DBGrid1.Columns[3].Width:=100;

DBGrid1.Columns[6].Width:=275;

End;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N3Click(Sender: TObject);

begin

try

CoNSQL.ADOQuery1.SQL.Text:='SELECT surname,first\_name,middle\_name,is\_male,'+

'CONVERT(varchar(30), birthday, 104) AS [DDate],phone,address,training\_courses FROM abiture WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

if CoNSQL.ADOQuery1.IsEmpty then abort;

uwpdate:=true;

with addstudform do

Begin

abit\_id:=Consql.ADOQuery3.FieldByName('ID').AsInteger;

is\_insert:=false;

Label1.Caption:='+';

Label1.Font.Color:=clred;

Label2.Caption:='+';

Label2.Font.Color:=clred;

Label3.Caption:='+';

Label3.Font.Color:=clred;

Label4.Caption:='+';

Label4.Font.Color:=clred;

Label5.Caption:='+';

Label5.Font.Color:=clred;

Label6.Caption:='+';

Label6.Font.Color:=clred;

Label7.Caption:='+';

Label7.Font.Color:=clred;

f\_is\_lgt;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT surname,first\_name,middle\_name,is\_male,'+

'CONVERT(varchar(30), birthday, 104) AS [DDate],phone,address,training\_courses FROM abiture WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with AddObweeForm do

Begin

edit1.Text:=Consql.ADOQuery1.FieldByName('surname').AsString;

edit2.Text:=Consql.ADOQuery1.FieldByName('first\_name').AsString;

edit3.Text:=Consql.ADOQuery1.FieldByName('middle\_name').AsString;

if (Consql.ADOQuery1.FieldByName('is\_male').AsBoolean) then

Begin RadioButton1.Checked:=true; end else

Begin RadioButton1.Checked:=false; RadioButton2.Checked:=true; end;

edit5.Date:=strtodate(Consql.ADOQuery1.FieldByName('DDate').AsString);

edit6.Text:=Consql.ADOQuery1.FieldByName('phone').AsString;

edit7.Text:=Consql.ADOQuery1.FieldByName('address').AsString;;

Checkbox1.Checked:=Consql.ADOQuery1.FieldByName('training\_courses').AsBoolean;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT series,numb,place\_of\_birth,issued\_by,'+

'CONVERT(varchar(30),date\_of\_issue, 104) AS [DDate] FROM pasport WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with PasportForm do

Begin

edit1.Text:=CoNSQL.ADOQuery1.FieldByName('series').AsString;

edit2.Text:=CoNSQL.ADOQuery1.FieldByName('numb').AsString;;

edit3.Text:=CoNSQL.ADOQuery1.FieldByName('place\_of\_birth').AsString;

edit4.Text:=CoNSQL.ADOQuery1.FieldByName('issued\_by').AsString;;

edit5.Date:=strtodate(Consql.ADOQuery1.FieldByName('DDate').AsString);

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT document.name FROM document INNER JOIN documents ON document.numb\_doc = documents.numb\_doc '+

'WHERE documents.id\_statement = :1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with Doc\_form do

Begin

ListBox1.Clear;

ListBox2.Clear;

While not consql.ADOQuery1.Eof do

begin

ListBox1.Items.Add(consql.ADOQuery1.FieldByName('name').AsString);

consql.ADOQuery1.Next;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT name FROM document WHERE name NOT IN '+

'(SELECT document.name FROM document INNER JOIN documents ON '+

'document.numb\_doc = documents.numb\_doc WHERE documents.id\_statement = :1)';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

While not consql.ADOQuery1.Eof do

begin

ListBox2.Items.Add(consql.ADOQuery1.FieldByName('name').AsString);

consql.ADOQuery1.Next;

end;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT series,numb,surname,first\_name,middle\_name,place\_of\_birth, '+

'CONVERT(varchar(30), birthday, 104) AS [DDate] FROM pasp\_rod WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with Pasp\_rod\_Form do

Begin

edit1.Text:=CoNSQL.ADOQuery1.FieldByName('series').AsString;

edit2.Text:=CoNSQL.ADOQuery1.FieldByName('numb').AsString;

edit3.Text:=CoNSQL.ADOQuery1.FieldByName('place\_of\_birth').AsString;

edit5.Date:=strtodate(Consql.ADOQuery1.FieldByName('DDate').AsString);

edit6.Text:=Consql.ADOQuery1.FieldByName('surname').AsString;

edit7.Text:=Consql.ADOQuery1.FieldByName('first\_name').AsString;

edit8.Text:=Consql.ADOQuery1.FieldByName('middle\_name').AsString;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT numb\_at,institution,s\_otl,avgn, '+

'CONVERT(varchar(30), date\_of\_issue, 104) AS [DDate] FROM atestat WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with Atest\_form do

Begin

edit1.Text:=CoNSQL.ADOQuery1.FieldByName('numb\_at').AsString;

edit2.Text:=CoNSQL.ADOQuery1.FieldByName('institution').AsString;

edit3.Text:=CoNSQL.ADOQuery1.FieldByName('avgn').AsString;;

Checkbox1.Checked:=Consql.ADOQuery1.FieldByName('s\_otl').AsBoolean;

edit5.Date:=strtodate(Consql.ADOQuery1.FieldByName('DDate').AsString);

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT numb\_sv,avg\_oge, '+

'CONVERT(varchar(30), date\_of\_issue, 104) AS [DDate] FROM oge WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with OGEG do

Begin

edit1.Text:=CoNSQL.ADOQuery1.FieldByName('numb\_sv').AsString;

edit3.Text:=CoNSQL.ADOQuery1.FieldByName('avg\_oge').AsString;

edit5.date:=strtodate(Consql.ADOQuery1.FieldByName('DDate').AsString);

end;

CoNSQL.ADOQuery1.SQL.Text:='EXEC priem\_doz :1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with Priem\_form do

Begin

ComboBox1.Clear;

ComboBox2.Clear;

ComboBox3.Clear;

CoNSQL.adoquery2.SQL.Clear;

CoNSQL.adoquery2.SQL.add('SELECT name\_spec AS VGA FROM specialty');

CoNSQL.adoquery2.open;

while not(CoNSQL.adoquery2.Eof) and (CoNSQL.adoquery2.RecordCount>0) do

Begin

ComboBox1.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

ComboBox2.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

ComboBox3.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

CoNSQL.adoquery2.Next;

end;

CoNSQL.ADOQuery2.SQL.Clear;

if Not(CoNSQL.ADOQuery1.IsEmpty) and not(CoNSQL.ADOQuery1.Eof) then

Begin

Combobox1.ItemIndex:=CoNSQL.ADOQuery1.FieldByName('num').AsInteger-1;

CoNSQL.ADOQuery1.Next;

end;

if Not(CoNSQL.ADOQuery1.IsEmpty) and not(CoNSQL.ADOQuery1.Eof) then

Begin

Combobox2.ItemIndex:=CoNSQL.ADOQuery1.FieldByName('num').AsInteger-1;

CoNSQL.ADOQuery1.Next;

end;

if Not(CoNSQL.ADOQuery1.IsEmpty) and not(CoNSQL.ADOQuery1.Eof) then

Begin

Combobox3.ItemIndex:=CoNSQL.ADOQuery1.FieldByName('num').AsInteger-1;

CoNSQL.ADOQuery1.Next;

end;

end;

CoNSQL.ADOQuery1.SQL.Text:='SELECT name,document\_number FROM benefits WHERE id\_statement=:1';

CoNSQL.ADOQuery1.Parameters.ParseSQL(CoNSQL.ADOQuery1.SQL.Text,true);

CoNSQL.ADOQuery1.Parameters.ParamByName('1').Value:= addstudform.abit\_id;

CoNSQL.ADOQuery1.Open;

with LgotForm do

Begin

edit1.Text:=CoNSQL.ADOQuery1.FieldByName('name').AsString;

edit2.Text:=CoNSQL.ADOQuery1.FieldByName('document\_number').AsString;

end;

addstudform.ShowModal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.DBGrid1TitleClick(Column: TColumn);

begin

try

if CoNSQL.ADOQuery3.Sort <> ('['+ Column.FieldName + '] ASC') then

CoNSQL.ADOQuery3.Sort:= '['+ Column.FieldName + '] ASC' else

CoNSQL.ADOQuery3.Sort:= '['+ Column.FieldName + '] DESC';

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N9Click(Sender: TObject);

begin

try

see\_abit\_atest:=1;

CoNSQL.ADOQuery4.SQL.Text:='exec sel\_avg\_atest NULL,:2';

SeeForm.DBGrid1.Visible:=true;

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= 0;

CoNSQL.ADOQuery4.Open;

if not(CoNSQL.ADOQuery4.IsEmpty) then

Begin

SeeForm.DBGrid1.Columns[0].Width:=30;

SeeForm.DBGrid1.Columns[1].Width:=100;

SeeForm.DBGrid1.Columns[2].Width:=75;

SeeForm.DBGrid1.Columns[3].Width:=100;

SeeForm.DBGrid1.Columns[4].Width:=30;

SeeForm.DBGrid1.Columns[0].Title.caption:='ID';

SeeForm.DBGrid1.Columns[1].Title.caption:='Ôàìèëèÿ';

SeeForm.DBGrid1.Columns[2].Title.caption:='Èìÿ';

SeeForm.DBGrid1.Columns[3].Title.caption:='Îò÷åñòâî';

SeeForm.DBGrid1.Columns[4].Title.caption:='Áàëë';

end else DBGrid1.Visible:=false;

SeeForm.showmodal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N4Click(Sender: TObject);

begin

ProhForm.showmodal;

end;

procedure TMainUserForm.N27Click(Sender: TObject);

var temp\_id:integer;

begin

try

temp\_id:=CoNSQL.ADOQuery3.FieldByName('ID').AsInteger;

CoNSQL.MyRave.SetParam('DTFAM',CoNSQL.ADOQuery3.FieldByName('Ôàìèëèÿ').AsString);

CoNSQL.MyRave.SetParam('DTNAM',CoNSQL.ADOQuery3.FieldByName('Èìÿ').AsString);

CoNSQL.MyRave.SetParam('DTOT4',CoNSQL.ADOQuery3.FieldByName('Îò÷åñòâî').AsString);

CoNSQL.MyRave.SetParam('DTRJD',CoNSQL.ADOQuery3.FieldByName('Äåíü ðîæäåíèÿ').AsString);

CoNSQL.MyRave.SetParam('DTADR',CoNSQL.ADOQuery3.FieldByName('Àäðåñ').AsString);

CoNSQL.TempQuery.SQL.Clear;

CoNSQL.TempQuery.SQL.Text:='SELECT series,numb,date\_of\_issue FROM pasport WHERE id\_statement = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

CoNSQL.MyRave.SetParam('DTSER',CoNSQL.TempQuery.FieldByName('series').AsString);

CoNSQL.MyRave.SetParam('DTNUM',CoNSQL.TempQuery.FieldByName('numb').AsString);

CoNSQL.MyRave.SetParam('DTDVI',CoNSQL.TempQuery.FieldByName('date\_of\_issue').AsString);

CoNSQL.TempQuery.SQL.Clear;

CoNSQL.TempQuery.SQL.Text:='SELECT phone FROM abiture WHERE id\_statement = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

CoNSQL.MyRave.SetParam('DTTEL',CoNSQL.TempQuery.FieldByName('phone').AsString);

CoNSQL.TempQuery.SQL.Text:='SELECT name\_spec,code\_spec FROM priem INNER JOIN specialty '+

'ON priem.id\_spec = specialty.id\_spec WHERE id\_statement = :1 ORDER BY priority ASC';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

CoNSQL.TempQuery.First;

CoNSQL.MyRave.SetParam('DTNS1',CoNSQL.TempQuery.FieldByName('name\_spec').AsString);

CoNSQL.MyRave.SetParam('DTKS1',CoNSQL.TempQuery.FieldByName('code\_spec').AsString);

CoNSQL.TempQuery.Next;

CoNSQL.MyRave.SetParam('DTNS2',CoNSQL.TempQuery.FieldByName('name\_spec').AsString);

CoNSQL.MyRave.SetParam('DTKS2',CoNSQL.TempQuery.FieldByName('code\_spec').AsString);

CoNSQL.TempQuery.Next;

CoNSQL.MyRave.SetParam('DTNS3',CoNSQL.TempQuery.FieldByName('name\_spec').AsString);

CoNSQL.MyRave.SetParam('DTKS3',CoNSQL.TempQuery.FieldByName('code\_spec').AsString);

CoNSQL.TempQuery.SQL.Text:='SELECT numb\_at,avgn,date\_of\_issue FROM atestat WHERE id\_statement = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

CoNSQL.MyRave.SetParam('DTANB',CoNSQL.TempQuery.FieldByName('numb\_at').AsString);

CoNSQL.MyRave.SetParam('DTASB',CoNSQL.TempQuery.FieldByName('avgn').AsString);

CoNSQL.MyRave.SetParam('DTADI',CoNSQL.TempQuery.FieldByName('date\_of\_issue').AsString);

CoNSQL.TempQuery.SQL.Text:='SELECT name FROM benefits WHERE id\_statement = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

if not(CoNSQL.TempQuery.IsEmpty) then

CoNSQL.MyRave.SetParam('DTNBN',CoNSQL.TempQuery.FieldByName('name').AsString) else

CoNSQL.MyRave.SetParam('DTNBN','-');

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\Za9vl.rav';

CoNSQL.MyRave.Execute;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N28Click(Sender: TObject);

var temp\_id:integer; temp\_s:string;

begin

try

CoNSQL.MyRave.SetParam('DTDAT',datetostr(Date));

temp\_s:='"'+CoNSQL.ADOQuery3.FieldByName('Ôàìèëèÿ').AsString+' '+

CoNSQL.ADOQuery3.FieldByName('Èìÿ').AsString+' '+

CoNSQL.ADOQuery3.FieldByName('Îò÷åñòâî').AsString+'"';

temp\_id:=CoNSQL.ADOQuery3.FieldByName('ID').AsInteger;

CoNSQL.MyRave.SetParam('DTSTD',temp\_s);

CoNSQL.TempQuery.SQL.Text:='SELECT name FROM document INNER JOIN documents '+

'ON document.numb\_doc = documents.numb\_doc WHERE id\_statement = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:=temp\_id;

CoNSQL.TempQuery.Open;

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\rasp.rav';

CoNSQL.MyRave.Execute;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N24Click(Sender: TObject);

begin

try

CoNSQL.TempQuery.SQL.Text:='SELECT COUNT(\*) FROM abiture WHERE training\_courses = 1';

CoNSQL.TempQuery.Open;

StatistForm.Label3.Caption:=CoNSQL.TempQuery.Fields.Fields[0].AsString;

CoNSQL.TempQuery.SQL.Text:='SELECT COUNT(\*) FROM abiture INNER JOIN benefits '+

'ON abiture.id\_statement = benefits.id\_statement';

CoNSQL.TempQuery.Open;

StatistForm.Label5.Caption:=CoNSQL.TempQuery.Fields.Fields[0].AsString;

CoNSQL.ADOQuery4.SQL.Text:='SELECT name\_spec AS [Ñïåöèàëüíîñòü], places AS [Ìåñò],'+

' prowed AS [Ïðîøåäøèå], ne\_prowed AS [Íå ïðîøåäøèå] FROM abit\_prow';

CoNSQL.ADOQuery4.Open;

if not(CoNSQL.ADOQuery4.IsEmpty) then

Begin

StatistForm.DBGrid1.Columns[0].Width:=350;

StatistForm.DBGrid1.Columns[1].Width:=75;

StatistForm.DBGrid1.Columns[2].Width:=75;

StatistForm.DBGrid1.Columns[3].Width:=85;

end;

StatistForm.showmodal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

TMainUserForm.N25Click(Sender: TObject);

begin

try

CoNSQL.ADOQuery4.SQL.Text:='exec sel\_count\_benf';

CoNSQL.ADOQuery4.Open;

showmessage('Êîëè÷åñòâî ñòóäåíòîâ èìåþùèõ ëüãîòû : ' + CoNSQL.ADOQuery4.Fields.Fields[0].AsString);

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N12Click(Sender: TObject);

begin

try

see\_abit\_atest:=1;

CoNSQL.ADOQuery4.SQL.Text:='exec prowedwie NULL,NULL,0';

CoNSQL.ADOQuery4.Open;

SeeForm.Label3.Visible:=true;

SeeForm.DBGrid1.Visible:=false;

SeeForm.SpinEdit1.Visible:=true;

SeeForm.N1.Visible:=true;

SeeForm.ShowModal;

SeeForm.SpinEdit1.Value:=0;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N13Click(Sender: TObject);

begin

try

see\_abit\_atest:=2;

CoNSQL.ADOQuery4.SQL.Text:='exec polu\_proh NULL,NULL,0';

CoNSQL.ADOQuery4.Open;

SeeForm.Label3.Visible:=true;

SeeForm.DBGrid1.Visible:=false;

SeeForm.SpinEdit1.Visible:=true;

SeeForm.N1.Visible:=true;

SeeForm.ShowModal;

SeeForm.SpinEdit1.Value:=0;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N14Click(Sender: TObject);

begin

try

see\_abit\_atest:=3;

CoNSQL.ADOQuery4.SQL.Text:='exec ne\_prowedwie NULL,NULL,0';

CoNSQL.ADOQuery4.Open;

SeeForm.Label3.Visible:=true;

SeeForm.DBGrid1.Visible:=false;

SeeForm.SpinEdit1.Visible:=true;

SeeForm.N1.Visible:=true;

SeeForm.ShowModal;

SeeForm.SpinEdit1.Value:=0;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

function TMainUserForm.show\_your\_atest: byte;

begin

show\_your\_atest:=see\_abit\_atest;

end;

procedure TMainUserForm.N6Click(Sender: TObject);

begin

try

CoNSQL.ADOQuery4.SQL.Text:='exec see\_all\_table';

CoNSQL.ADOQuery4.Open;

alltableform.showmodal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N1Click(Sender: TObject);

begin

halt;

end;

procedure TMainUserForm.N5Click(Sender: TObject);

begin

CoNSQL.ADOConnection1.Connected:=false;

MainUserForm.Hide;

AuthForm.Show;

end;

procedure TMainUserForm.N10Click(Sender: TObject);

begin

if CoNSQL.UserName = '' then

Begin

showmessage('Access denied');

end else

ChangePassForm.ShowModal;

end;

procedure TMainUserForm.NdocClick(Sender: TObject);

begin

try

Nspec.Checked:=false;

Ndoc.Checked:=true;

CoNSQL.ADOQuery3.SQL.Text:='SELECT numb\_doc AS [Íîìåð], name AS [Íàçâàíèå] '+

'FROM document';

CoNSQL.ADOQuery3.Open;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.NspecClick(Sender: TObject);

begin

try

Nspec.Checked:=true;

Ndoc.Checked:=false;

CoNSQL.ADOQuery3.SQL.Text:='SELECT name\_spec AS [Íàçâàíèå], code\_spec AS [Êîä],'+

'places AS [Ìåñò] FROM specialty';

CoNSQL.ADOQuery3.Open;

DBGrid1.Columns[0].Width:=450;

DBGrid1.Columns[1].Width:=150;

DBGrid1.Columns[2].Width:=75;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N40Click(Sender: TObject);

begin

rd:=false;

form4.ShowModal;

end;

procedure TMainUserForm.N41Click(Sender: TObject);

begin

try

if trim(CoNSQL.adoquery3.Fields.Fields[0].asstring) <> CoNSQL.UserName then

Begin

try

form4.Edit1.Text:=trim(ConSQL.adoquery3.Fields.Fields[0].asstring);

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

rd:=true;

form4.Edit3.Text:='thisishesh';

form4.ShowModal;

end else showmessage('Access denied!');

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.N22Click(Sender: TObject);

begin

try

CoNSQL.TempQuery.SQL.Text:='SELECT COUNT(\*) FROM abiture WHERE training\_courses = 1';

CoNSQL.TempQuery.Open;

StatistForm.Label3.Caption:=CoNSQL.TempQuery.Fields.Fields[0].AsString;

CoNSQL.TempQuery.SQL.Text:='SELECT COUNT(\*) FROM abiture INNER JOIN benefits '+

'ON abiture.id\_statement = benefits.id\_statement';

CoNSQL.TempQuery.Open;

StatistForm.Label5.Caption:=CoNSQL.TempQuery.Fields.Fields[0].AsString;

CoNSQL.ADOQuery4.SQL.Text:='SELECT name\_spec AS [Ñïåöèàëüíîñòü], places AS [Ìåñò],'+

' prowed AS [Ïðîøåäøèå], ne\_prowed AS [Íå ïðîøåäøèå],proh\_ball AS [Ïðîõîäíîé áàëë] FROM abit\_prow';

CoNSQL.ADOQuery4.Open;

if not(CoNSQL.ADOQuery4.IsEmpty) then

Begin

StatistForm.DBGrid1.Columns[0].Width:=325;

StatistForm.DBGrid1.Columns[1].Width:=30;

StatistForm.DBGrid1.Columns[2].Width:=70;

StatistForm.DBGrid1.Columns[3].Width:=80;

StatistForm.DBGrid1.Columns[4].Width:=90;

end;

StatistForm.showmodal;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.ComboBox1Change(Sender: TObject);

begin

try

CoNSQL.SkinIsRough.SkinName:=ComBobox1.Items[ComBobox1.ItemIndex];

IniFile.WriteInteger('Design','SkinID',ComBobox1.ItemIndex);

IniFile.WriteInteger('StyleID',CoNSQL.UserName,ComBobox1.ItemIndex);

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TMainUserForm.DBGrid1KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

try

if (key = 13) then

Begin

CoNSQL.ADOQuery3.Edit;

CoNSQL.ADOQuery3.Post;

end;

if (key = 46) and (ssShift in shift) then

Begin

if not(CoNSQL.ADOQuery3.IsEmpty) then

Begin

if CoNSQL.UserRole = 'Admun' then

Begin

with CoNSQL do

Begin

if CoNSQL.UserName <> trim(adoquery3.Fields.Fields[0].asstring) then

Begin

ADOQuery4.SQL.Clear;

ADOQuery4.SQL.Text:='exec drop\_users :1';

ADOQuery4.Parameters.ParseSQL(ADOQuery4.SQL.Text,true);

ADOQuery4.Parameters.ParamByName('1').Value:=trim

(adoquery3.Fields.Fields[0].asstring);

ADOQuery4.ExecSQL;

ADOQuery3.Active:=false;

ADOQuery3.Active:=true;

end else showmessage('Access denied');

end;

end else

try

CoNSQL.ADOQuery3.Delete;

except

showmessage('Âû íå ìîæåòå âûïîëíèòü ýòî óäàëåíèå íå íàðóøèâ öåëîñòíîñòè ÁÄ!');

end;

End;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

end.

unit NetAPI;

interface

uses Windows, Classes;

procedure GetSQLServerNames(Names: TStrings; const DomainName: string; const ServerType: DWORD);

function ScanDomainForSQLNames(List: TStrings; NetRes: PNetResource): Word;

implementation

const

NERR\_SUCCESS = 0;

MAX\_PREFERRED\_LENGTH = DWORD(-1);

SV\_TYPE\_WORKSTATION = $00000001;

SV\_TYPE\_SERVER = $00000002;

SV\_TYPE\_SQLSERVER = $00000004;

SV\_TYPE\_DOMAIN\_CTRL = $00000008;

SV\_TYPE\_DOMAIN\_BAKCTRL = $00000010;

SV\_TYPE\_TIME\_SOURCE = $00000020;

SV\_TYPE\_AFP = $00000040;

SV\_TYPE\_NOVELL = $00000080;

SV\_TYPE\_DOMAIN\_MEMBER = $00000100;

SV\_TYPE\_PRINTQ\_SERVER = $00000200;

SV\_TYPE\_DIALIN\_SERVER = $00000400;

SV\_TYPE\_XENIX\_SERVER = $00000800;

SV\_TYPE\_SERVER\_UNIX = SV\_TYPE\_XENIX\_SERVER;

SV\_TYPE\_NT = $00001000;

SV\_TYPE\_WFW = $00002000;

SV\_TYPE\_SERVER\_MFPN = $00004000;

SV\_TYPE\_SERVER\_NT = $00008000;

SV\_TYPE\_POTENTIAL\_BROWSER = $00010000;

SV\_TYPE\_BACKUP\_BROWSER = $00020000;

SV\_TYPE\_MASTER\_BROWSER = $00040000;

SV\_TYPE\_DOMAIN\_MASTER = $00080000;

SV\_TYPE\_SERVER\_OSF = $00100000;

SV\_TYPE\_SERVER\_VMS = $00200000;

SV\_TYPE\_WINDOWS = $00400000; // Windows95 and above

SV\_TYPE\_DFS = $00800000; // Root of a DFS tree

SV\_TYPE\_CLUSTER\_NT = $01000000; // NT Cluster

SV\_TYPE\_DCE = $10000000; // IBM DSS (Directory and Security Services) or equivalent

SV\_TYPE\_ALTERNATE\_XPORT = $20000000; // return list for alternate transport

SV\_TYPE\_LOCAL\_LIST\_ONLY = $40000000; // Return local list only

SV\_TYPE\_DOMAIN\_ENUM = $80000000;

SV\_TYPE\_ALL = $FFFFFFFF; //handy for NetServerEnum2

type

TServerInfo101 = record

platform\_id: DWORD;

name: PWideChar;

version\_major: DWORD;

version\_minor: DWORD;

server\_type: DWORD;

comment: PWideChar;

end;

PServerInfo101 = ^TServerInfo101;

TServerInfo100 = record

platform\_id: DWORD;

name: PWideChar;

end;

PServerInfo100 = ^TServerInfo100;

function NetServerEnum(const ServerName: PWideString; level: DWORD; var Buffer: pointer;

PrefMaxLen: DWORD; var EntriesRead: DWORD; var TotalEntries: DWORD; ServerType: DWORD;

const Domain: PWideChar; var ResumeHandle: DWORD): DWORD; stdcall;

external 'netapi32.dll';

function NetApiBufferFree(Buffer: pointer): DWORD; stdcall;

external 'netapi32.dll';

procedure GetSQLServerNames(Names: TStrings; const DomainName: string; const ServerType: DWORD);

var

Buffer: pointer;

EntriesRead, i, ErrCode, ResumeHandle, TotalEntries: DWORD;

DomainUnicode: array[0..100] of WideChar;

PDomainUnicode: PWideChar;

ServerInfo: PServerInfo101;

begin

ResumeHandle:= 0;

if (DomainName = '') then PDomainUnicode:= nil

else begin

StringToWideChar(DomainName, DomainUnicode, SizeOf(DomainUnicode));

PDomainUnicode:= DomainUnicode;

end;

errCode:= NetServerEnum(nil, 101, Buffer, MAX\_PREFERRED\_LENGTH, EntriesRead, TotalEntries, ServerType, PDomainUnicode, ResumeHandle);

if (errCode <> NERR\_SUCCESS) then Exit;

try

ServerInfo:= Buffer;

for i := 1 to EntriesRead do begin

Names.Add(ServerInfo^.name);

Inc(ServerInfo);

end;

finally

NetApiBufferFree(Buffer);

end;

end;

function ScanDomainForSQLNames(List: TStrings; NetRes: PNetResource): Word;

var

ArNetRes: array[0..59] of TNetResource;

i: integer;

EntrReq, SizeReq: DWORD;

hEnum: THandle;

DomenName: string;

begin

Result:= WNetOpenEnum(RESOURCE\_GLOBALNET, RESOURCETYPE\_ANY, RESOURCEUSAGE\_CONTAINER, NetRes, hEnum);

if Result = ERROR\_NO\_NETWORK then exit;

if Result = NO\_ERROR then begin

EntrReq:= 1;

SizeReq:= SizeOf(TNetResource) \* 59;

while (hEnum <> 0) and (WNetEnumResource(hEnum, EntrReq, @ArNetRes, SizeReq) <> ERROR\_NO\_MORE\_ITEMS) do begin

for i:= 0 To EntrReq - 1 do

with ArNetRes[i] do begin

case dwDisplayType of

RESOURCEDISPLAYTYPE\_DOMAIN: begin

if lpRemoteName <> '' then DomenName:= lpRemoteName

else DomenName:= lpComment;

GetSQLServerNames(List, DomenName, SV\_TYPE\_SQLSERVER);

end;

else ScanDomainForSQLNames(List, @ArNetRes[i]);

end;

end;

end;

WNetCloseEnum(hEnum);

end;

end;

end.

unit OGE;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls;

type

TOGEG = class(TForm)

Label1: TLabel;

Label5: TLabel;

Label4: TLabel;

Edit1: TEdit;

Edit5: TDateTimePicker;

Edit3: TEdit;

Button2: TButton;

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var OGEG: TOGEG;

implementation

uses add\_abiture, ForMeMComp;

procedure TOGEG.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',#08]) then Key := #0;

end;

procedure TOGEG.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',',',#08]) then Key := #0;

end;

procedure TOGEG.Button2Click(Sender: TObject);

begin

if ((length(Edit1.Text)=6) and (strtofloat(edit3.Text) >= 3) and

(strtofloat(edit3.Text) <= 5) and (Edit5.date<Date)) then

Begin

close;

addstudform.Label6.Caption:='+';

addstudform.Label6.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TOGEG.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

if not((length(Edit1.Text)=6) and (strtofloat(edit3.Text) >= 3) and

(strtofloat(edit3.Text) <= 5) and (Edit5.date<Date)) then

Begin

addstudform.Label6.Caption:='X';

addstudform.Label6.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end.

unit pasp\_rod;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls;

type

TPasp\_rod\_Form = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Edit6: TEdit;

Edit7: TEdit;

Edit8: TEdit;

Label9: TLabel;

Edit5: TDateTimePicker;

Button2: TButton;

procedure Edit6KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var Pasp\_rod\_Form: TPasp\_rod\_Form;

implementation

uses add\_abiture, ForMeMComp;

procedure TPasp\_rod\_Form.Edit6KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['a'..'z', 'A'..'Z', 'à'..'ÿ', 'À'..'ß',#08]) then Key := #0;

end;

procedure TPasp\_rod\_Form.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',#08]) then Key := #0;

end;

procedure TPasp\_rod\_Form.Button2Click(Sender: TObject);

begin

if (length(Edit1.Text) = 4) and (length(Edit2.Text) = 6) and (edit6.Text<>'') and

(Edit5.date<date) and (edit7.Text<>'') and (edit8.Text<>'') and (edit3.Text<>'') then

Begin

close;

addstudform.Label4.Caption:='+';

addstudform.Label4.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TPasp\_rod\_Form.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

if not((length(Edit1.Text) = 4) and (length(Edit2.Text) = 6) and (edit6.Text<>'') and

(Edit5.date<date) and (edit7.Text<>'') and (edit8.Text<>'') and (edit3.Text<>'')) then

Begin

addstudform.Label4.Caption:='X';

addstudform.Label4.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end.

unit pasport;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls;

type

TPasportForm = class(TForm)

Button2: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TDateTimePicker;

procedure Button2Click(Sender: TObject);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var PasportForm: TPasportForm;

implementation

uses ForMeMComp,add\_abiture;

procedure TPasportForm.Button2Click(Sender: TObject);

begin

if (length(Edit1.Text) = 4) and (length(Edit2.Text) = 6) and (edit3.Text<>'') and (Edit5.date<date) and

(edit4.Text<>'') then

Begin

close;

addstudform.Label2.Caption:='+';

addstudform.Label2.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TPasportForm.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',#08]) then Key := #0;

end;

procedure TPasportForm.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

if not((length(Edit1.Text) = 4) and (length(Edit2.Text) = 6) and (edit3.Text<>'')

and (Edit5.date<date) and (edit4.Text<>'')) then

Begin

addstudform.Label2.Caption:='X';

addstudform.Label2.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end.

unit Priem;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms;

type

TPriem\_form = class(TForm)

ComboBox1: TComboBox;

ComboBox2: TComboBox;

ComboBox3: TComboBox;

Button2: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

procedure Button2Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var Priem\_form: TPriem\_form;

implementation

uses add\_abiture, ForMeMComp;

procedure TPriem\_form.Button2Click(Sender: TObject);

begin

if ((ComboBox1.Items[ComboBox1.itemindex]<>'')and (ComboBox2.Items[ComboBox1.itemindex]<>'')and

(ComboBox1.Items[ComboBox1.itemindex]<>'')) then

Begin

close;

addstudform.Label7.Caption:='+';

addstudform.Label7.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TPriem\_form.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

if not((ComboBox1.Items[ComboBox1.itemindex]<>'')and (ComboBox2.Items[ComboBox1.itemindex]<>'')and

(ComboBox1.Items[ComboBox1.itemindex]<>'')) then

Begin

addstudform.Label7.Caption:='X';

addstudform.Label7.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end.

unit Prohball;

interface

usesWindows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Spin, sEdit, sSpinEdit;

type

TProhForm = class(TForm)

ComboBox1: TComboBox;

Label2: TLabel;

Label1: TLabel;

CheckBox1: TCheckBox;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Button1: TButton;

SpinEdit1: TsSpinEdit;

procedure FormShow(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure ComboBox1Change(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var ProhForm: TProhForm;

implementation

uses ForMeMComp;

procedure TProhForm.FormShow(Sender: TObject);

begin

checkbox1.Checked:=false;

Label5.Caption:='';

CoNSQL.adoquery2.SQL.Clear;

CoNSQL.adoquery2.SQL.add('SELECT name\_spec AS VGA FROM specialty');

try

CoNSQL.adoquery2.open;

ComboBox1.Clear;

while not(CoNSQL.adoquery2.Eof) and (CoNSQL.adoquery2.RecordCount>0) do

Begin

ComboBox1.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

CoNSQL.adoquery2.Next;

end;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

end;

procedure TProhForm.Button1Click(Sender: TObject);

var prior:integer;

begin

if checkbox1.Checked then prior:=1 else prior:=0;

CoNSQL.ADOQuery4.SQL.Text:='exec proh\_ball :1,:2,:3';

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:= ComboBox1.text;

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= prior;

CoNSQL.ADOQuery4.Parameters.ParamByName('3').Value:= SpinEdit1.Value;

try

CoNSQL.ADOQuery4.Open;

if (CoNSQL.ADOQuery4.FieldByName('APoL').AsString = 'FALSE') then

Label5.Caption:='Íåò' else

Label5.Caption:=CoNSQL.ADOQuery4.FieldByName('APoL').AsString;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

end;

TProhForm.ComboBox1Change(Sender: TObject);

begin

CoNSQL.TempQuery.SQL.Text:='select TOP 1 places from specialty WHERE name\_spec = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:= ComboBox1.text;

try

CoNSQL.TempQuery.Open;

SpinEdit1.Value:=CoNSQL.TempQuery.Fields.Fields[0].AsInteger;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

end;

end.

unit SeeUnit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Grids, DBGrids, StdCtrls, Menus, Spin, printers, acDBGrid, sEdit, sSpinEdit;

type

TSeeForm = class(TForm)

ComboBox1: TComboBox;

Label2: TLabel;

Label1: TLabel;

CheckBox1: TCheckBox;

Label3: TLabel;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

Button1: TButton;

DBGrid1: TsDBGrid;

SpinEdit1: TsSpinEdit;

procedure FormShow(Sender: TObject);

procedure ComboBox1Change(Sender: TObject);

procedure CheckBox1Click(Sender: TObject);

procedure SpinEdit1Change(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

procedure change\_exec;

end;

var SeeForm: TSeeForm;

implementation

uses ForMeMComp, MainUserUnit;

procedure TSeeForm.FormShow(Sender: TObject);

begin

checkbox1.Checked:=false;

CoNSQL.adoquery2.SQL.Clear;

CoNSQL.adoquery2.SQL.text:='SELECT name\_spec AS VGA FROM specialty';

try

CoNSQL.adoquery2.open;

ComboBox1.Clear;

while not(CoNSQL.adoquery2.Eof) and (CoNSQL.adoquery2.RecordCount>0) do

Begin

ComboBox1.Items.Add(CoNSQL.adoquery2.fieldbyname('VGA').asstring);

CoNSQL.adoquery2.Next;

end;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

procedure TSeeForm.change\_exec;

var prior:integer;

begin

DBGrid1.Visible:=true;

if Checkbox1.Checked then prior:=1 else prior:=0;

if (Combobox1.Text = '') and (SpinEdit1.Visible) then

Begin

CoNSQL.ADOQuery4.SQL.Text:='exec prowedwie NULL,NULL,0';

try

CoNSQL.ADOQuery4.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

SpinEdit1.Value:=0;

end else

if MainUserForm.show\_your\_atest = 3 then

Begin

Begin

CoNSQL.TempQuery.SQL.Text:='select TOP 1 places from specialty WHERE name\_spec = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:= ComboBox1.text;

try

CoNSQL.TempQuery.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

SpinEdit1.Value:=CoNSQL.TempQuery.Fields.Fields[0].AsInteger;

CoNSQL.ADOQuery4.SQL.Text:='exec ne\_prowedwie :1,:2,:3';

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:= ComboBox1.text;

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= prior;

CoNSQL.ADOQuery4.Parameters.ParamByName('3').Value:= SpinEdit1.Value;

end;

end else

if MainUserForm.show\_your\_atest = 2 then

Begin

Begin

CoNSQL.TempQuery.SQL.Text:='select TOP 1 places from specialty WHERE name\_spec = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:= ComboBox1.text;

try

CoNSQL.TempQuery.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

SpinEdit1.Value:=CoNSQL.TempQuery.Fields.Fields[0].AsInteger;

CoNSQL.ADOQuery4.SQL.Text:='exec polu\_proh :1,:2,:3';

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:= ComboBox1.text;

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= prior;

CoNSQL.ADOQuery4.Parameters.ParamByName('3').Value:= SpinEdit1.Value;

end;

end else

if not(SpinEdit1.Visible) then

Begin

if ComboBox1.text='' then CoNSQL.ADOQuery4.SQL.Text:='exec sel\_avg\_atest NULL,NULL' else

Begin

CoNSQL.ADOQuery4.SQL.Text:='exec sel\_avg\_atest :1,:2';

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:= ComboBox1.text;

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= prior;

end;

end else

Begin

Begin

CoNSQL.TempQuery.SQL.Text:='select TOP 1 places from specialty WHERE name\_spec = :1';

CoNSQL.TempQuery.Parameters.ParseSQL(CoNSQL.TempQuery.SQL.Text,true);

CoNSQL.TempQuery.Parameters.ParamByName('1').Value:= ComboBox1.text;

try

CoNSQL.TempQuery.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

SpinEdit1.Value:=CoNSQL.TempQuery.Fields.Fields[0].AsInteger;

CoNSQL.ADOQuery4.SQL.Text:='exec prowedwie :1,:2,:3';

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:= ComboBox1.text;

CoNSQL.ADOQuery4.Parameters.ParamByName('2').Value:= prior;

CoNSQL.ADOQuery4.Parameters.ParamByName('3').Value:= SpinEdit1.Value;

end;

end;

try

CoNSQL.ADOQuery4.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

if not(CoNSQL.ADOQuery4.IsEmpty) then

Begin

DBGrid1.Columns[0].Width:=30;

DBGrid1.Columns[1].Width:=100;

DBGrid1.Columns[2].Width:=75;

DBGrid1.Columns[3].Width:=100;

DBGrid1.Columns[4].Width:=30;

DBGrid1.Columns[0].Title.caption:='ID';

DBGrid1.Columns[1].Title.caption:='Ôàìèëèÿ';

DBGrid1.Columns[2].Title.caption:='Èìÿ';

DBGrid1.Columns[3].Title.caption:='Îò÷åñòâî';

DBGrid1.Columns[4].Title.caption:='Áàëë';

end else DBGrid1.Visible:=false;

if Combobox1.Text = '' then

Begin

CoNSQL.MyRave.SetParam('po\_spec','');

CoNSQL.MyRave.SetParam('spec','');

end else

Begin

CoNSQL.MyRave.SetParam('po\_spec','ïî ñïåöèàëüíîñòè:');

CoNSQL.MyRave.SetParam('spec',Combobox1.Text);

end;

end;

procedure TSeeForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

SeeForm.Label3.Visible:=false;

SeeForm.SpinEdit1.Visible:=false;

SeeForm.N1.Visible:=false;

end;

procedure TSeeForm.Button1Click(Sender: TObject);

Begin

if MainUserForm.show\_your\_atest = 3 then

Begin

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\Ne\_postup.rav';

CoNSQL.MyRave.Execute;

end else

if MainUserForm.show\_your\_atest = 2 then

Begin

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\pol\_proh.rav';

CoNSQL.MyRave.Execute;

end;

if MainUserForm.show\_your\_atest = 1 then

Begin

if SpinEdit1.Visible then

Begin

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\postup.rav';

CoNSQL.MyRave.Execute;

end else

Begin

CoNSQL.MyRave.Close;

CoNSQL.MyRave.ProjectFile:='.\ravki\AllAbb.rav';

CoNSQL.MyRave.Execute;

end;

end;

end;

unit StForm;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Grids, DBGrids, StdCtrls, acDBGrid;

type

TStatistForm = class(TForm)

Button1: TButton;

Label1: TLabel;

CheckBox1: TCheckBox;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

DBGrid1: TsDBGrid;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var StatistForm: TStatistForm;

implementation

uses ForMeMComp;

procedure TStatistForm.Button1Click(Sender: TObject);

begin

CoNSQL.ADOQuery4.SQL.Text:='exec gen\_all\_prow :1';

CoNSQL.ADOQuery4.Parameters.ParseSQL(CoNSQL.ADOQuery4.SQL.Text,true);

if Checkbox1.Checked then

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:=1 else

CoNSQL.ADOQuery4.Parameters.ParamByName('1').Value:=0;

try

CoNSQL.ADOQuery4.ExecSQL;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

CoNSQL.ADOQuery4.SQL.Text:='SELECT name\_spec AS [Ñïåöèàëüíîñòü], places AS [Ìåñò],'+

' prowed AS [Ïðîøåäøèå], ne\_prowed AS [Íå ïðîøåäøèå],proh\_ball AS [Ïðîõîäíîé áàëë] FROM abit\_prow';

try

CoNSQL.ADOQuery4.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

if not(CoNSQL.ADOQuery4.IsEmpty) then

Begin

StatistForm.DBGrid1.Columns[0].Width:=325;

StatistForm.DBGrid1.Columns[1].Width:=30;

StatistForm.DBGrid1.Columns[2].Width:=70;

StatistForm.DBGrid1.Columns[3].Width:=80;

StatistForm.DBGrid1.Columns[4].Width:=90;

end;

end;

end.

unit UserUnit;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms;

type

TForm4 = class(TForm)

Label2: TLabel;

Edit1: TEdit;

Label1: TLabel;

Label3: TLabel;

Edit2: TEdit;

Button1: TButton;

Label4: TLabel;

Edit3: TEdit;

ComboBox1: TComboBox;

procedure FormHide(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure Button1Click(Sender: TObject);

procedure clearform;

procedure ComboBox1KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure FormShow(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var Form4: TForm4;

implementation

uses ForMeMComp,MainUserUnit;

procedure TForm4.FormHide(Sender: TObject);

begin

form4.Label4.Visible:=false;

form4.Edit2.Visible:=false;

CoNSQL.ADOQuery3.SQL.Text:='SELECT \* FROM sel\_user\_deathwing';

try

CoNSQL.ADOQuery3.Open;

except

showmessage('Îøèáêà ïðè ïîëó÷åíèè äàííûõ ñ ñåðâåðà');

end;

end;

procedure TForm4.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

form4.Label4.Visible:=false;

form4.Edit2.Visible:=false;

end;

procedure TForm4.Button1Click(Sender: TObject);

begin

try

if not(rd) then

Begin

if (Edit1.Text<>'')and(Edit3.Text<>'')and(Combobox1.Items[Combobox1.ItemIndex]<>'')then

Begin

with CoNSQL do

Begin

ADOQuery3.Active:=false;

adoquery4.SQL.Clear;

adoquery4.SQL.Add('exec add\_users :1,:2,:3');

ADOQuery4.Parameters.ParseSQL(ADOQuery4.SQL.Text,true);

Adoquery4.Parameters.ParamByName('1').Value:=Edit1.Text;

Adoquery4.Parameters.ParamByName('2').Value:= Combobox1.Items[Combobox1.ItemIndex];

Adoquery4.Parameters.ParamByName('3').Value:=Edit3.Text;

adoquery4.ExecSQL;

ADOQuery3.Active:=true;

clearform;

close;

end;

end else showmessage('Íå âñå ïîëÿ çàïîëíåíû');

CoNSQL.adoquery4.SQL.Clear;

end else

Begin

if (Edit1.Text<>'')and(Combobox1.Items[Combobox1.ItemIndex]<>'')then

Begin

with CoNSQL do

Begin

adoquery4.SQL.Clear;

adoquery4.SQL.Text:='exec update\_users :1,:2,:3,:4';

ADOQuery4.Parameters.ParseSQL(ADOQuery4.SQL.Text,true) Adoquery4.Parameters.ParamByName('1').Value:= ADOQuery3.Fields.Fields[0].asstring;

Adoquery4.Parameters.ParamByName('2').Value:=Edit1.Text;

Adoquery4.Parameters.ParamByName('3').Value:= Combobox1.Items[Combobox1.ItemIndex];

Adoquery4.Parameters.ParamByName('4').Value:=Edit3.Text;

adoquery4.ExecSQL;

clearform;

close;

end;

end else showmessage('Íå âñå ïîëÿ çàïîëíåíû');

CoNSQL.adoquery4.SQL.Clear;

End;

form4.Label4.Visible:=false;

form4.Edit2.Visible:=false;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure TForm4.ComboBox1KeyPress(Sender: TObject; var Key: Char);

begin

Key:=#0;

end;

procedure TForm4.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['a'..'z', 'A'..'Z', 'à'..'ÿ', 'À'..'ß']) then Key := #0;

end;

procedure TForm4.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

if (Key =' ') then Key := #0;

end;

end.

unit add\_abiture;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms;

type

Taddstudform = class(TForm)

Button2: TButton;

Button1: TButton;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Button6: TButton;

Button7: TButton;

Button8: TButton;

Button9: TButton;

Button10: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure Button8Click(Sender: TObject);

procedure Button9Click(Sender: TObject);

procedure Button7Click(Sender: TObject);

procedure Button10Click(Sender: TObject);

private

is\_benfit:boolean;

procedure insert\_obw;

procedure insert\_pasp;

procedure insert\_doc;

procedure insert\_pasp\_rod;

procedure insert\_atest;

procedure insert\_oge;

procedure insert\_priem;

procedure insert\_benfit;

public

is\_insert:boolean;

abit\_id:integer;

procedure t\_is\_lgt;

procedure f\_is\_lgt;

function g\_is\_lgt:boolean;

end;

var addstudform: Taddstudform;

implementation

uses Add\_obwee, pasport, ForMeMComp, doc\_unit, pasp\_rod, atestat, OGE,

Priem, Lgot, MainUserUnit;

procedure Taddstudform.f\_is\_lgt;

begin

is\_benfit:=false;

end;

procedure Taddstudform.t\_is\_lgt;

begin

is\_benfit:=true;

end;

function Taddstudform.g\_is\_lgt: boolean;

begin

g\_is\_lgt:=is\_benfit;

end;

procedure Taddstudform.Button2Click(Sender: TObject);

begin

AddObweeForm.Showmodal;

end;

procedure Taddstudform.Button3Click(Sender: TObject);

begin

if (label1.Caption='+') and (label2.Caption='+') and (label3.Caption='+')

and (label4.Caption='+') and (label5.Caption='+') and (label6.Caption='+')

and (label7.Caption='+') then

Begin

if MainUserForm.uwpdate then

Begin

CoNSQL.ADOQuery2.SQL.Text:='DELETE FROM abiture WHERE id\_statement = :1';

CoNSQL.ADOQuery2.Parameters.ParseSQL(CoNSQL.ADOQuery2.SQL.Text,true);

CoNSQL.ADOQuery2.Parameters.ParamByName('1').Value:= abit\_id;

try

CoNSQL.ADOQuery2.ExecSQL;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

insert\_obw;

insert\_pasp;

insert\_doc;

insert\_pasp\_rod;

insert\_atest;

insert\_oge;

insert\_priem;

if is\_benfit then insert\_benfit;

showmessage('Àáèòóðèåíò óñïåøíî äîáàâëåí');

CoNSQL.ADOQuery3.SQL.Text:='exec sel\_title :1';

CoNSQL.ADOQuery3.Parameters.ParseSQL(CoNSQL.ADOQuery3.SQL.Text,true);

CoNSQL.ADOQuery3.Parameters.ParamByName('1').Value:= '';

try

CoNSQL.ADOQuery3.Open;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

if not(CoNSQL.ADOQuery3.IsEmpty) then

Begin

MainUserForm.DBGrid1.Columns[0].Width:=30;

MainUserForm.DBGrid1.Columns[1].Width:=100;

MainUserForm.DBGrid1.Columns[2].Width:=75;

MainUserForm.DBGrid1.Columns[3].Width:=100;

MainUserForm.DBGrid1.Columns[6].Width:=275;

end;

close;

end else

Begin

showmessage('Âû çàïîëíèëè íå âñå òðåáóåìûå äàííûå');

end;

end;

procedure Taddstudform.Button4Click(Sender: TObject);

begin

PasportForm.ShowModal;

end;

procedure Taddstudform.Button5Click(Sender: TObject);

begin

doc\_form.ShowModal;

end;

procedure Taddstudform.insert\_obw;

begin

try

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO abiture(surname,first\_name,middle\_name,is\_male,birthday,phone,address,training\_courses,date\_of\_application)'+

' VALUES (:1,:2,:3,:4,:5,:6,:7,:8,CONVERT(date,GETDATE()))';

ADOQuery2.Parameters.ParseSQL(CoNSQL.ADOQuery2.SQL.Text,true);

with AddObweeForm do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit2.text;

ADOQuery2.Parameters.ParamByName('3').Value:=Edit3.text;

ADOQuery2.Parameters.ParamByName('4').Value:=RadioButton1.Checked;

ADOQuery2.Parameters.ParamByName('5').Value:=datetostr(Edit5.date);

ADOQuery2.Parameters.ParamByName('6').Value:=Edit6.text;

ADOQuery2.Parameters.ParamByName('7').Value:=Edit7.text;

ADOQuery2.Parameters.ParamByName('8').Value:=CheckBox1.checked;

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

ADOQuery2.SQL.Add('SELECT IDENT\_CURRENT('+#39+'abiture'+#39+')');

ADOQuery2.Open;

abit\_id:=ADOQuery2.Fields[0].AsInteger;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.insert\_pasp;

begin

try

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO pasport(id\_statement,series,numb,place\_of\_birth,issued\_by,date\_of\_issue)'+

' VALUES (:1,:2,:3,:4,:5,:6)';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

with PasportForm do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('3').Value:=Edit2.text;

ADOQuery2.Parameters.ParamByName('4').Value:=Edit3.Text;

ADOQuery2.Parameters.ParamByName('5').Value:=Edit4.Text;

ADOQuery2.Parameters.ParamByName('6').Value:=datetostr(Edit5.date);

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.insert\_doc;

begin

try

with Doc\_form do

Begin

while (listbox1.Count>0) do

Begin

ListBox1.Selected[0]:=true;

CoNSQL.ADOQuery2.SQL.text:='EXEC add\_doc @id\_statement='+inttostr(abit\_id)+',@name=N'+#39+

ListBox1.Items[ListBox1.ItemIndex]+#39;

CoNSQL.ADOQuery2.ExecSQL;

CoNSQL.ADOQuery2.SQL.Clear;

ListBox1.DeleteSelected;

end;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.Button6Click(Sender: TObject);

begin

Pasp\_rod\_Form.ShowModal;

end;

procedure Taddstudform.insert\_pasp\_rod;

begin

try

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO pasp\_rod(id\_statement,series,numb,surname,first\_name,middle\_name,birthday,place\_of\_birth)'+

' VALUES (:1,:2,:3,:4,:5,:6,:7,:8)';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

with Pasp\_rod\_Form do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('3').Value:=Edit2.text;

ADOQuery2.Parameters.ParamByName('4').Value:=Edit6.Text;

ADOQuery2.Parameters.ParamByName('5').Value:=Edit7.Text;

ADOQuery2.Parameters.ParamByName('6').Value:=Edit8.Text;

ADOQuery2.Parameters.ParamByName('7').Value:=datetostr(Edit5.date);

ADOQuery2.Parameters.ParamByName('8').Value:=Edit3.text;

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.insert\_atest;

var temp\_s:string;

zar:integer;

begin

try

temp\_s:=Atest\_form.Edit3.Text;

for zar:=0 to 4 do

Begin if (temp\_s[zar] = ',') then temp\_s[zar]:='.'; end;

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO atestat(id\_statement,numb\_at,institution,s\_otl,avgn,date\_of\_issue)'+

' VALUES (:1,:2,:3,:4,:5,:6)';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

with Atest\_form do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('3').Value:=Edit2.text;

ADOQuery2.Parameters.ParamByName('4').Value:=Checkbox1.Checked;

ADOQuery2.Parameters.ParamByName('5').Value:=temp\_s;

ADOQuery2.Parameters.ParamByName('6').Value:=datetostr(Edit5.Date);

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.Button9Click(Sender: TObject);

begin

OGEG.ShowModal;

end;

procedure Taddstudform.insert\_oge;

var temp\_s:string;

zar:integer;

begin

try

temp\_s:=OGEG.Edit3.Text;

for zar:=0 to 4 do

Begin if(temp\_s[zar] = ',') then temp\_s[zar]:='.'; end;

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO oge(id\_statement,numb\_sv,avg\_oge,date\_of\_issue) VALUES (:1,:2,:3,:4)';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

with OGEG do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('3').Value:=temp\_s;

ADOQuery2.Parameters.ParamByName('4').Value:=datetostr(Edit5.Date);

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.Button7Click(Sender: TObject);

begin

Priem\_form.showmodal;

end;

procedure Taddstudform.insert\_priem;

begin

try

with Priem\_form do

Begin

with CoNSQL do

Begin

ADOQuery2.SQL.Clear;

if ComboBox1.Items[ComboBox1.itemindex]<>'' then

Begin

ADOQuery2.SQL.Text:='EXEC add\_priem :1,:2,:7';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=ComboBox1.Items[ComboBox1.itemindex];

ADOQuery2.Parameters.ParamByName('7').Value:=1;

ADOQuery2.ExecSQL;

end else Begin showmessage('Îøèáêà ïðèåìà íà ñïåöèàëüíîñòè'); abort; end;

if ComboBox2.Items[ComboBox2.itemindex]<>'' then

Begin

ADOQuery2.SQL.Text:='EXEC add\_priem :3,:4,:8';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

ADOQuery2.Parameters.ParamByName('3').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('4').Value:=ComboBox2.Items[ComboBox2.itemindex];

ADOQuery2.Parameters.ParamByName('8').Value:=2;

ADOQuery2.ExecSQL;

if ComboBox3.Items[ComboBox3.itemindex]<>'' then

Begin

ADOQuery2.SQL.Text:='EXEC add\_priem :5,:6,:9';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

ADOQuery2.Parameters.ParamByName('5').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('6').Value:=ComboBox3.Items[ComboBox3.itemindex];

ADOQuery2.Parameters.ParamByName('9').Value:=3;

ADOQuery2.ExecSQL;

end;

end;

ADOQuery2.SQL.Clear;

end;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.insert\_benfit;

begin

try

with CoNSQL do

Begin

ADOQuery2.SQL.text:=

'INSERT INTO benefits(id\_statement,name,document\_number) VALUES (:1,:2,:3)';

ADOQuery2.Parameters.ParseSQL(ADOQuery2.SQL.Text,true);

with LgotForm do

Begin

ADOQuery2.Parameters.ParamByName('1').Value:=abit\_id;

ADOQuery2.Parameters.ParamByName('2').Value:=Edit1.text;

ADOQuery2.Parameters.ParamByName('3').Value:=Edit2.text;

end;

ADOQuery2.ExecSQL;

ADOQuery2.SQL.Clear;

end;

except

on E: Exception do

Showmessage('ERRoR: ' + E.Message);

end;

end;

procedure Taddstudform.Button10Click(Sender: TObject);

begin

LgotForm.ShowModal;

end;

end.

unit Add\_obwee;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls, Mask, sMaskEdit, sCustomComboEdit, sToolEdit;

type

TAddObweeForm = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit6: TEdit;

Edit7: TEdit;

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

CheckBox1: TCheckBox;

Button2: TButton;

Edit5: TDateTimePicker;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit6KeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

public

{ Public declarations }

end;

var AddObweeForm: TAddObweeForm;

implementation

uses ForMeMComp,add\_abiture;

procedure TAddObweeForm.Button2Click(Sender: TObject);

begin

if (length(Edit6.Text) < 6) then showmessage('Íîìåð òåëåôîíà ââåäåí íå âåðíî') else

if (edit1.Text<>'') and (edit2.Text<>'') and (edit3.Text<>'') and (Edit5.date<Date) and

(edit6.Text<>'') and (edit7.Text<>'') then

Begin

close;

addstudform.Label1.Caption:='+';

addstudform.Label1.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TAddObweeForm.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

if not((edit1.Text<>'') and (edit2.Text<>'') and (edit3.Text<>'') and (Edit5.date<date) and

(length(Edit6.Text) > 5) and (edit7.Text<>'')) then

Begin

addstudform.Label1.Caption:='X';

addstudform.Label1.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

procedure TAddObweeForm.Edit6KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',#08]) then Key := #0;

end;

end.

unit ATableForm;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Grids, DBGrids, acDBGrid;

type

TAllTableForm = class(TForm)

sDBGrid1: TsDBGrid;

private

{ Private declarations }

public

{ Public declarations }

end;

var AllTableForm: TAllTableForm;

implementation

end.

unit atestat;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Grids, StdCtrls, ComCtrls, Mask;

type

TAtest\_form = class(TForm)

Button2: TButton;

Label1: TLabel;

Label2: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Label8: TLabel;

CheckBox1: TCheckBox;

Label5: TLabel;

Label4: TLabel;

Edit5: TDateTimePicker;

Edit3: TEdit;

procedure Button2Click(Sender: TObject);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

{ Public declarations }

end;

var Atest\_form: TAtest\_form;

implementation

uses ForMeMComp, add\_abiture;

procedure TAtest\_form.Button2Click(Sender: TObject);

begin

if ((length(Edit1.Text)=6) and (edit2.Text<>'') and (Edit5.date<Date) and

(strtofloat(edit3.Text) >= 3) and (strtofloat(edit3.Text) <= 5) ) then

Begin

close;

addstudform.Label5.Caption:='+';

addstudform.Label5.Font.Color:=clgreen;

end else

Begin

showmessage('Íå âñå ïîëÿ âåðíî çàïîëíåíû');

end;

end;

procedure TAtest\_form.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',#08]) then Key := #0;

end;

procedure TAtest\_form.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9',',',#08]) then Key := #0;

end;

procedure TAtest\_form.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

if not((length(Edit1.Text)=6) and (edit2.Text<>'') and (Edit5.date<Date) and

(strtofloat(edit3.Text) >= 3) and (strtofloat(edit3.Text) <= 5) ) then

Begin

addstudform.Label5.Caption:='X';

addstudform.Label5.Font.Color:=clred;

end;

CoNSQL.ADOQuery2.SQL.Clear;

end;

end