МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Информационно-вычислительные системы»

Курсовой проект

по дисциплине «Проектирвоание автоматизированных системы специального назначения»

на тему «Разработка автоматизированной системы кафедры»

Специальность 09.05.01 «Применение и эксплуатация автоматизированных систем специального назначения»

Специализация №12 «Автоматизированные системы обработки информации и управления специального назначения»

ПГУ 1.090501.02

Выполнил: ст-т группы 20ВОА1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Басов М. В.

Руководитель: д.т.н., профессор кафедры ИВС

­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Чувыкин Б. В.

**Работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_**

**Преподаватели \_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_**

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

2024

Реферат

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

ПГУ 1.090501.02

Разраб.

Басов М. В.

Провер.

Чувыкин Б. В.

Н. Контр.

.

Утв.

Кузьмин А. В.

Разработка автоматизированной системы кафедры

Пояснительная записка

Лит.

Листов

гр. 20ВОА1

Пояснительная записка содержит листов, рисунков, таблицы,  приложение.

Объектом исследования является процесс учета закрепления за студентами тем дипломных проектов.

Цель работы является разработка автоматизированной системы кафедры, состоящей из базы данных и приложения для работы с ней.

При разработке модели бизнес-процессов, концептуальной, логической, а также реляцеонной моделей было использовано CASE-средство OpenModelSphere 3.2.

Для разработки самой базы данных, на основе полученных моделей, использовалась система управления базы данных ЛИНТЕР. При разработке приложения для работы с базой данных использовалась среда разработки Embarcadero Delphi 10 Tokyo.

Содержание

Введение…............................................................................................................... 6 1 Анализ предметной области................................................................................ 8

2 Техническое задание…........................................................................................ 12

2.1 Основания для разработки ….......................................................................... 12

2.2 Назначение разработки …................................................................................ 12

2.3 Требования к программной документации …................................................ 13

2.4 Порядок контроля и приемки........................................................................... 13

2.5 Стадии и этапы разработки............................................................................. 13

2.6 Требования к программе …............................................................................ 14

2.6.1 Требования к функциональным характеристикам…................................ 14

2.6.2 Требования к надежности …....................................................................... 14

2.6.3 Требования к составу и параметрам технических средств ..................... 15

2.6.4 Требования к информационной и программной совместимости........... 15

3 Обоснование выбора средств разработки.......................................................... 16

4 Проектирование базы данных............................................................................. 22

4.1 Физическое проектирование базы данных.................................................... 22

4.2 Разработка мандатной защиты базы данных................................................ 41

5 Проектирование приложения............................................................................. 44

5.1 Анализ функций приложения......................................................................... 44

5.2 Определение описания функций.................................................................... 44

5.3 Отображение функций в модули.................................................................... 46

6 Описание программы........................................................................................... 49

6.1 Общие сведения................................................................................................ 49

6.2 Функциональное назначение........................................................................... 49

6.3 Логическая структура программы.................................................................. 49

6.4 Использование технических средств............................................................. 49

6.5 Вызов и загрузка.............................................................................................. 70

6.6 Входные данные.............................................................................................. 70

6.7 Выходные данные............................................................................................ 70

7 Программа и методика испытаний................................................................... 71

7.1 Объект испытания.......................................................................................... 71

7.2 Цель испытания.............................................................................................. 71

7.3 Требования к программе................................................................................ 71

7.4 Требования к программной документации.................................................. 71

7.5 Средство и порядок испытаний..................................................................... 72

7.6 Методика испытаний...................................................................................... 73

8 Описание применения........................................................................................ 77

8.1 Назначение программы.................................................................................. 77

8.2 Условия применения....................................................................................... 77

8.3 Описания задач.......................................................................................... 77 Заключение............................................................................................................. 81 Список использованных источников............................................................. 82 Приложение А. SQL- скрипт баз данных.............................................. 84 Приложение Б. Результат выполнения тестовых примеров........................... 88 Приложение В. Листинг программы................................................................ 97 Приложение Г. Текст dfm- файлов......................................................................... 107

Введение

Проектирование информационных систем – это процесс разработки технической документации, связанной с организационной системой получения и преобразования исходных данных в результаты.

Документ, полученный, в результате проектирования называется проектом, а сама документация проектной документацией. /1/

Целью проектирования является подбор технического и формирование информационного, математического, программного и организационно-правового обеспечения.

Подбор технического обеспечения должен быть таким, чтобы обеспечить своевременный сбор, регистрацию, передачу, хранение, наполнение и обработку информации.

Информационное обеспечение должно предусматривать создание и функционирование единого информационного фонда системы, представленного множеством информационных массивов, набором данных или базой данных.

Формирование математического обеспечения систем включает комплектацию методов и алгоритмов решения функциональных задач. При формировании программного обеспечения систем особое внимание обращается на создание комплекса программ и инструкций пользователя и выбор эффективных программных продуктов. /2/

Для хранения всего объема информации АИС используются специальные базы данных. База данных – это организованная структура, в которой совокупность однородных данных, преимущественно больших объемов, размещается в таблицах и отображает состояние объектов и их отношений в определенной предметной области. Управленческая функция по отношению к информационным процессам в базах данных осуществляется посредством систем управления базами данных. /3/

СУБД ЛИНТЕР – это реляционная система управления базами данных. Возможности системы ЛИНТЕР позволяют использовать ее для решения практически любых задач – от приложений для микроконтроллеров до корпоративных информационных систем. /4/

ЛИНТЕР - открытая система, предназначенная для использования в многозадачных операционных средах, поэтому алгоритмам распараллеливания обработки запросов уделялось пристальное внимание. При этом максимально используется то распараллеливание, которое дает операционная система, и, в дополнение к этому, ядро СУБД проводит собственное распараллеливание обработки запросов. Алгоритм обработки запросов ведется несколькими задачами (ядро, транслятор запросов, транслятор хранимых процедур, сортировщик ответов), распараллеливанием работы которых занимается операционная система.

Данные в таблицах физически хранятся построчно. В одну строку могут входить данные разных типов (символы, целые и вещественные числа, строки символов различной длины, и так далее).

В ЛИНТЕР реализован Search engine предназначенный для выполнения задач полнотекстового поиска. Индексируются следующие типы данных: char, varchar, nchar, nvarchar, BLOB, внешний файл данных.

ЛИНТЕР работает на многих программно-аппаратных платформах: Linux (различные версии и аппаратные платформы включая Embedded Linux), МСВС, Solaris, Mac OS X, BSD (OpenBSD, FreeBSD, BSDI, NetBSD ), UnixWare, IRIX, AIX, SINIX, QNX, USIX, VxWorks, OS-9, OS-9000, OC2000, ИНТРОС, VMS, Windows (95,Me,2000,XP,NT4), Win CE ( различные версии, начиная с 3), причем, практически на всех платформах базовый вариант СУБД работает внешне одинаково. /5/

Разработанная в данном курсовом проекте автоматизированная система, предназначена для хранения сведений о кафедре, которой может пользоваться бухгалтер и обеспечивать её информацией об факультетах, о группах, о студантах, о руководителях и о темах проекта.

При разработке приложения была использована среда Delphi 10 Tokyo, база данных спроектирована с помощью CASE-средства OpenModulSphere 3.2 и реализована с помощью СУБД Линтер.

# 1 Анализ предметной области

Кафедра — это структурное подразделение образовательного учреждения, которое занимается обучением студентов в определенной предметной области. Обычно на кафедре работают преподаватели, специализирующиеся в определенных областях знаний, проводят научно-исследовательскую работу, участвуют в публикациях, научных конференциях и семинарах.

Сотрудник кафедры осуществляет учет о студентах и преподователях. Каждый студент может выбрать одного рукуводителя. Каждый руководитель может выбрать несколько студентов. Над одной темой проекта может работать несколько студентов, но одна тема может быть выбрана только один раз. Каждая группа может иметь несколько студентов. Один студент может быть только в одной группе. Каждый факультет может иметь несколько групп. При учете студентов указывается следующая информация: фамилия, имя, отчество, место предипломной практики, номер группы, название проекта, руководитель. Также указываются сведения о группах, а именно название группы. В информации о руководителе указывается фамилия, имя, отчество, ученое звание, место работы, должность. Информация о проектах содержит название проекта. Также в информации о факультете указывается название и фамилия, имя, отчество декана.

Процесс автоматизации кафедры – неотъемлемая часть успешного проведения всех этапов работ. Современная автоматизация учета осуществляется с помощью пакета Microsoft Office, который требует ручного ввода данных и не может обеспечить быстрый поиск по информации. Чтобы упростить управление кафедрой и вывести ее на новый уровень, необходима оптимизация текущих бизнес-процессов и внедрение специализированной IT-системы.

Таким образом, актуальной является задача автоматизации системы учета сведения о студентах и преподователях путем разработки автоматизированной системы или использования готовой автоматизированной системы.

Известная система автоматизации учета сведения о студентах и преподователей является «ATutor» — это система дистанционого обучения с открытым исходным кодом, используемая для управления онлайн-курсами, а также для создания интерактивных электронных учебных материалов и предоставления доступа к ним.

Система управления обучением с открытым исходным кодом ATutor (рус. АТьютор) используется для разработки онлайн-курсов и проведения обучения в образовательных учреждениях и в частных компаниях. Система может быть установлена на сервер и использоваться с применением веб-браузера.

Рассматривая преимущества программы «ATutor», стоит отметить основные пункты:

-не требует предварительного обучения и специальных знаний;

-программа может использоваться на всех устройствах;

-программа строго соответствует законодательству;

-программа имеет поддержку нескольких языков;

-бесплатная лицензия;

-имеет дружественный графический пользовательский интерфейс.

Также недостатками являются:

-малый функционал;

-нет функции по огранизайции занятий в формате онлайн видео-концеренций;

-нет возможности вести журнал успеваемости учащихся. /6/

Еще одной системой, которая является не менее распространенной является «Moodle» — это гибкая настраиваемая система дистанционного обучения с открытым исходным кодом, помогающая удовлетворить широкие потребности в организации обучения для образовательных учреждений и для бизнеса.

Программный продукт Moodle (рус. Мудл) от одноимённой австралийской компании является одной из наиболее популярных в настоящий момент платных систем управления обучением и систем дистанционного обучения. Стоимость данного продукта составляет 230 тыс. рублей.

Программное обеспечение moodle предоставляет преподавателям во всём мире решение для электронного дистанционного обучения, масштабируемое, настраиваемое и обеспечивающее безопасность с одним из самых больших наборов функций. /7/

Данные аналоги автоматизированных систем имею малый функционал и дорогую стоимость лицензии, что значительно затрудняет их использование в полном объеме.

Автоматизированная система кафедры будет значительно превосходить альтернативные решения благодаря следующим премуществам: система сможет охватывать многие аспекты учебного процесса, студенты получат удобный доступ к материалам курса и будет соответствие высоким стандартам безопасноти и конфиденциальности, а также недорогая стоимость лицензии.

Исходя из этого, можно сделать вывод, что разработка правильно работающей, но не дорогой системы учета кафедры является важной и актуальной задачей. Такая система позволит значительно улучшить качество учебного процесса. Для работы такой системы необходимо создать хорошо структурированную и надежную базу данных.

# 2 Техническое задание

### 2.1 Основание для разработки

Автоматизированная система разрабатывается на основании задания на курсовое проектирование, выданном д.т.н., профессор кафедры ИВС Чувыкин Б.В. и утвержденого заведующим кафедрой ИВС к.т.н., профессором Кузьминым А. В.

### 2.2 Назначение разработки

Разрабатываемое приложение должно быть предназначена для учета информации о студентах, о руководителях, о группе, о проектах и факультете.

### 2.3 Требования к функциональным характеристикам

Программа должна сопровождаться пояснительной запиской, содержащей следующие программные документы:

* анализ предметной области;
* техническое задание;
* обоснование выбора средств разработки;
* физическое проектирование БД;
* проектирование приложения;
* описание программы;
* программа и методика испытания;
* описание применения;
* текст программы.

### 2.4 Порядок контроля и приемки

Для проверки выполнения программой всех её функций в соответствии с техническим заданием следует разработать тестовые примеры и протестировать реакцию программы на возможные действия пользователя: запуск программы, заполнение полей ввода, работа с базой данных, выбор пунктов меню, нажатие кнопок. Приложение соответствует требованиям технического задания, если оно выполняет все заданные функции при различных наборах входных данных из тестовых примеров.

Приемка программы должна осуществляться при выполнении всех её функций, а также при наличии полной документации на программу (пояснительная записка с приложениями: текстом программы и результатами тестовых примеров).

### 2.5 Стадии и этапы разработки

Стадии и этапы разработки автоматизированной системы приведены в таблице 1.

Таблица 2.1 – Стадии и этапы разработки автоматизированной системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Стадии разработки | Этап работ | Сроки исполнения | Исполнитель |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Расчетная часть | Разработка исходных данных (ИД) | 17.09.24г. | Басов М. В. |
| Разработка программы с использованием среды Delphi | 26.11.24г. | Басов М. В. |
| Написание пояснительной записки | 17.12.24г. | Басов М. В. |
| 2 | Графическая часть | Разработка диаграммы физической модели данных | 08.10.24г. | Басов М. В. |

Продолжение таблицы 2.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 3 | Экспериментальная часть | Создание демонстрационной БД для СУБД Линтер | 22.10.24г. | Басов М. В. |
| Отладка программы приложения и испытания системы | 03.12.24г. | Басов М. В. |

### 2.6 Порядок контроля и приемки

2.6.1 Требования к функциональным характеристикам

Разрабатываемая программа должна выполнять следующие функции:

﻿﻿ - вести учет информации о студентах;

﻿﻿ - вести учет информации о руководителях;

﻿﻿ - вести учет информации о группах;

﻿﻿ - вести учет информации о факультете;

- вести учет информации о проектах;

﻿﻿ - выводить информацию о разработчике программы;

﻿﻿ - осуществлять проверку правильности ввода данных.

Входными данными должна быть информация, вводимая пользователем.

Выходными данными должна быть информация, выводимая из базы данных по запросам пользователя.

2.6.2 Требования к надежности

Программа должна работать без прерываний, поддерживать диалоговый

режим в рамках предоставляемых возможностей.

2.6.3 Требования к составу и параметрам технических средств

Программа должна быть предназначена для использования, как на локальном компьютере, так и в сети. Локальный компьютер или рабочая станция в сети должны иметь следующие технические характеристики:

﻿﻿ - тактовая частота процессора - не менее 2,1 МГц;

﻿﻿ - оперативная память - не менее 4096 Мбайт;

﻿﻿ - на жестком диске при установке используется около 250 Мбайт;

﻿﻿ - объем жесткого диска должен быть не менее 1024 Мбайт.

Локальный компьютер или сервер в сети должены иметь следующие технические характеристики:

- тактовая частота процессора - не менее 2,1 МГц;

- оперативная память - не менее 4096 Мбайт;

- на жестком диске при установке используется около 250 Мбайт;

- объем жесткого диска должен быть не менее 1024 Мбайт.

2.6.4 Требования к информационной и программной совместимости

Приложение должно работать как на локальном компьютере, так и в сети.

Для работы на сервере на нем должна быть установлена СУБД Линтер и файл с

БД, а на рабочей станции сети - приложение для работы с БД. Для работы на локальном компьютере должны быть установлены СУБД Линтер, приложение для работы с БД, файл с БД.

Автоматизированная система должна работать в ОС Windows 10.

3 Обоснование выбора средств разработки

Автоматизированная система учета информации о кафедры состоит из базы данных и приложения для работы с ней.

Для разработки базы данных используются специальные CASE-средства для разработки концептуальных, логических и реляционных моделей.

PowerBuilder 5.0 - это профессиональный объектно-ориентированный графический инструментарий построения приложений, работающих с базами данных в различных архитектурах как через прямые драйверы, так и через драйверы стандарта ODBC.

Основные функции данного CASE – средства:

– поддержка всех классов Windows;

– наличие специального объекта, называемого DataWindow, для всех манипуляций с базами данных, а также генерации отчетов;

– компилируемый 32-разрядный код, позволяющий увеличить производительность приложений;

– особые классы для создания распределенных приложений;

– полная переносимость кода между платформами Windows (Windows 3.x, Windows 95, Windows NT (Intel), Windows NT (Alpha)), основными платформами Unix (Sun Solaris, AIX, Digital Unix) и MacOS;

– поддержка динамических библиотек (DLL);

– поддержка импорта и экспорта файлов наиболее популярных форматов.

Недостатками CASE – средства Vantage Team Builder, являются:

– уступает по быстродействию Delphi и Microsoft SQL Server 6.0;

– он не позволяет во время проектирования одновременно подсоединяться ко многим источникам данных для просмотра объектов баз данных различных серверов;

– высокая стоимость лицензионного продукта (139 000 руб). /8/

Другим популярным CASE — является большим инструментом моделирования для проектов матобеспечения любого размера. Проектирование модели данных, модели бизнес-процесса (BPM) и модели UML. Open ModelSphere предлагает функции, подходящие для всех уровней пользователей: от опытных баз данных и архитекторов систем до начинающих студентов.

Open ModelSphere 3.2 предлагает большой набор функций:

– моделирование баз данных;

– моделирование процессов бизнеса;

– моделирование системных архитектур;

– моделирование сетевых архитектур;

– функциональность редактирования и создания диаграмм;

– генерация SQL-кода;

– многоязычный интерфейс, включая русский;

– поддержка различных операционных систем: Windows, Linux и macOS.

Как и любая программа, Open ModelSphere 3.2 имеет свои недостатки:

– периодические сбои при работе с большим количеством данных;

– отсутствие поддержки некоторых наиболее популярных баз данных, таких как Microsoft SQL Server и SQLite. /9/

Таким образом, для проектирования БД было выбрано CASE — средство Open ModelSphere 3.2, так как оно не требовательно к ресурсам системы, имеет большой функционал, является свободно распространяемой и понятно не опытному пользователю.

Для функционирования базы данных необходимо выбрать систему управления базами данных.

Oracle DB  - это культовая СУБД, разработанная ещё в семидесятые программистами Ларри Эллисоном и Бобом Майнером (которые, к слову, до этого работали на ЦРУ). Культовой её считают потому, что это первая коммерчески успешная СУБД, работающая по реляционной модели.

Преимущества Oracle DB перед иными СУБД вытекают из задач, под которые данный программный продукт «заточен». Рассмотрим основные из них.

1. **Масштабируемость и производительность.** Oracle обеспечивает высокую производительность даже при работе с очень большими объемами данных.

2. **Отказоустойчивость.** Oracle обеспечивает высокую надежность и защиту данных.

3. **Безопасность.** СУБД даёт все возможности для информационной безопасности.

4. **Многофункциональность.** Oracle предлагает множество дополнительных модулей и инструментов, позволяющих адаптировать систему под конкретные нужды бизнеса.

5. **Кроссплатформенность.** СУБД поддерживает все операционные системы и аппаратные платформы, что делает её универсальной. /5/

Недостатки Oracle DB:

1. Сложность. Одним из крупных недостатков Oracle Database является ее сложность. Oracle не рекомендуется для использования без достаточных технических знаний и навыков работы с Oracle Database. Oracle также не подойдет для тех, кто ищет простую в использовании и основных функциях базу данных. Установка Oracle и начало работы с ним требуют специализированных навыков, ведь это чрезвычайно сложный движок.

2. Oracle может быть в десять раз дороже, чем среднерыночное решение MS [SQL](https://blog.uniwex.io/luchshie-yazyki-dlya-veb-razrabotki-top-10/) Server Database.

3. **Сложность управления**. Oracle обычно требует больше усилий для управления некоторыми операциями. Совет: начните с установки базовой версии и минимальной настройки. Oracle Database оправдана только при работе с большими базами данных. Для малых и средних компаний, где требуются небольшие базы данных, Oracle не рекомендуется. В таком случае лучшим вариантом будет MySQL, который является более экономически выгодным. /10/

Система управления базами данных ЛИНТЕР предназначена для работы с реляционной моделью данных автоматизированных систем различного назначения.

СУБД ЛИНТЕР обеспечивает:

- возможности поддержку стандарта SQL:2008, за исключением не реляционных возможностей;

- возможности организации полнотекстового поиска в текстовых данных БД;

- полное или выборочное сохранение БД в архив и последующее восстановление ее из архива;

- возможности по горячему резервированию БД;

- поддержку программных интерфейсов для доступа к БД (интерфейсы нижнего и верхнего уровней, встроенный SQL, JDBC, ODBC, ADO.NET, PHP, Perl, Ruby, Qt, Python, TCL/TK).

Недостатки:

- падение эффективности в случае высокой динамики изменений.

- временное рассогласование данных на серверах, которое практически исключает применение систем асинхронной репликации в приложениях, требующих абсолютной синхронности данных, получаемых разными клиентами. /11/

Таким образом, для разработки базы данных, в соответствии с техническим заданием, была выбрана СУБД Линтер, обеспечивающая высокий уровень защиты, способность горячего резервирования, поддержание большого количества программных интерфейсов для доступа к БД.

Далее необходимо выбрать среду разработки приложения. Существуют различные среды разработки, такие как: Borland C++ Bilder, Delphi 10 Tokyo, Qt creator, Eclipse и др./4/

Qt creator - это одна из самых популярных кросс платформенных сред разработок на языках С, С++, Python, на основе фреймворка QT, поддерживает Windows, Linux и macOS и позволяет разрабатывать десктопные и мобильные приложения.

Преимущества Qt creator:

1. Удобное межпроцессное взаимодействие (сигналы/слоты);

2. Кросс-платформенность;

3. Низкий порог вхождения;

4. Удобный интерфейс;

5. Многофункциональность.

Недостатки Qt creator:

1. Слабая поддержка некоторых языков;

2. Ограниченные возможности отладки для не-Qt кода;

3. Высокий порог вхождения;

4. Проблемы с производительностью на больших проекта;. /12/

Delphi - является мощным и универсальным средством разработки приложений, RAD-оболочкой. Ее вместе с библиотекой VCL, на которой оболочка основана и написана, можно назвать действительно революционной. Сравнение с C++ Builder 4 показывает, что производительность Pascal-кода, сгенерированного Delphi, всего на 4-5% меньше, чем кода C++.

Delphi — это язык программирования, основанный на [Pascal](https://blog.skillfactory.ru/glossary/pascal/" \t "_blank). Имеет один из самых быстрых компиляторов, порождающий, тем не менее, весьма и весьма неплохой объектный код.

Преимущества Delphi:

– быстрота разработки приложения (RAD);

– **эффективность.** При создании Delphi пожертвовал частью эффективности своей работы ради понятности и очевидности действий.;

– **надежность.** Сейчас язык применяют там, где от программы нужны надежность и стабильность.

**Недостатки Delphi:**

**- несовременность.** Из-за своих особенностей Delphi считается многими «морально устаревшим».

**- малая популярность.** По причине выше, а еще из-за своеобразного подхода Delphi не слишком популярен на рынке, особенно если сравнивать с такими языками, как [JavaScript](https://blog.skillfactory.ru/glossary/javascript/) или Python.

**- узкая сфера использования.** Delphi сейчас применяют мало где, и в основном это компании, где работа ПО ориентирована на стабильность. /13/

Таким образом, для разработки приложения была выбрана программа Delphi 10 Tokyo.

Для создания автоматизированной системы кафедры, были выбраны следующие программные продукты: Open ModelSphere 3.2, СУБД Линтер и среда разработки Delphi 10 Tokyo.

# 4 Физическое проектирование базы данных

4.1 Физическое проектирование базы данных

На основе логической модели данных была построена физическая модель данных, для этого была выполнена команда меню «Сервис 🡪 Модель данных 🡪 Добавить/Удалить целевую систему», чтобы добавить к проекту новую целевую СУБД (используемую для создания базы данных).

В открывшемся окне, представленном на рисунке 4.1, была нажата кнопка «Добавить».

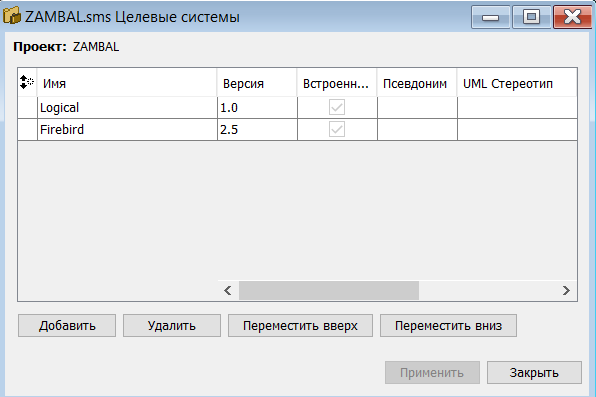


Рисунок 4.1 — Добавление к проекту целевой СУБД

Из списка доступных целевых систем была выбрана система Linter 6.0, как представлено на рисунке 4.2.

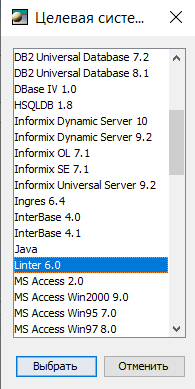


Рисунок 4.2 — Выбор целевой СУБД

Затем была нажата кнопка «Выбрать», а далее - кнопка «Закрыть», чтобы закрыть окно со списком целевых систем, представленных на рисунке 4.3.

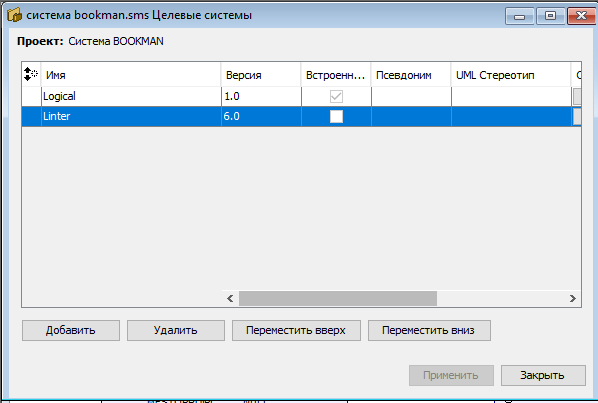


Рисунок 4.3 — Список целевых систем проекта

После назначения проекту целевой системы, могут быть использованы встроенные типы данных, поддерживаемые этой целевой системой, для определения типов данных в столбцах таблиц. В окне «Навигатор» в узле «Встроенные типы» выводится список встроенных типов данных, поддерживаемых СУБД Linter (Рисунок 4.4).

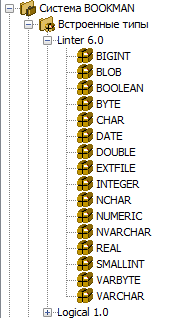


Рисунок 4.4 — Типы данных Linter

В окне «Навигатор» был выбран узел «Встроенные типы» и добавлена новая модель данных для СУБД Linter. Для этого нужно нажать правой кнопкой по узлу «Встроенные типы» и из всплывающего меню выбрать команду «Добавить 🡪 Модель данных 🡪 Linter 6.0». Затем необходимо переименовать созданную модель в «Физическая модель данных <Linter 6.0>».

Далее раскрыт узел «Реляционная модель данных» и выделены все таблицы этой модели в узле «Таблицы», а затем скопированы в окно физической модели. Для этого нужно щелкнуть правой кнопкой по выделенным строкам и из всплывающего меню выбрать команду «Копировать», как представлено на рисунке 4.5.

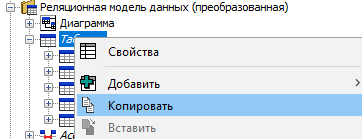


Рисунок 4.5 — Копирование таблиц

Модель данных представляет собой контейнер таблиц: для этого выбран узел «Физическая модель данных <Linter 6.0>» и вставлены скопированные таблицы. Для этого было нажато правой кнопкой по выбранному узлу и из всплывающего меню выбрана команда «Вставить».

После этого модель данных Linter содержит таблицы. Их столбцы имеют типы, встроенные в СУБД Linter.

Далее был выбран узел «Диаграмма» для модели Физическая модель данных <Linter 6.0> в окне «Навигатор», затем было нажато по нему правой кнопкой мыши и из всплывающего меню выбрана команда «Показать диаграмму».

По умолчанию, таблицы при этом не были отображены в графическом представлении. Далее была выбрана из всплывающего меню команда «Добавить отсутствующие объекты диаграммы».

Графические представления таблиц перекрывают друг друга. Поэтому было необходимо из всплывающего меню выбрать команду «Упорядочить всю диаграмму». После этого все таблицы расположены в одну линию. Поэтому их необходимо было переупорядочить и создать между ними связи.

Результат представлен на рисунке 4.6.

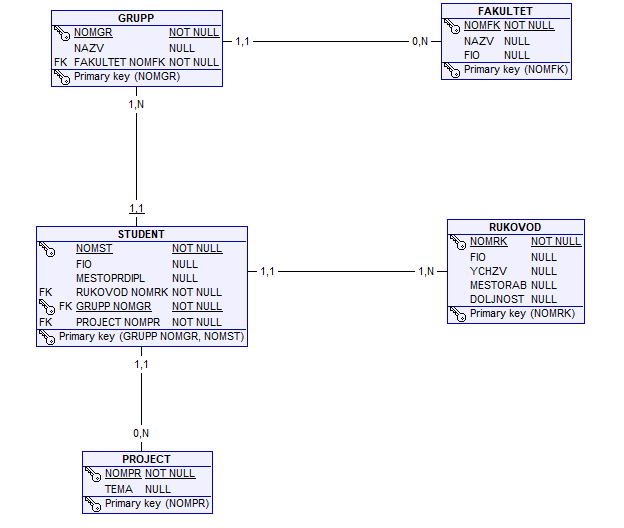


Рисунок 4.6 — Упорядоченная реляционная модель данных

Для определения типов данных необходимо было по очереди выделить в таблицах реляционной модели имена атрибутов, а затем в окне «Редактор свойств» выбрать в столбце «Значение» свойства «Тип». В результате было открыто окно «Выбрать новое значение», как представлено на рисунке 4.7.

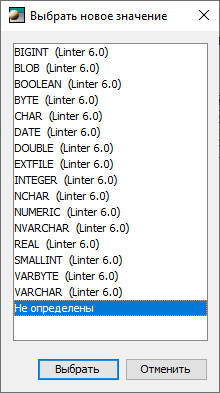


Рисунок 4.7 — Окно «Выбрать новое значение»

Из списка были выбраны необходимые типы данных, в соответствии с таблицами. После этого была нажата кнопка «Select» выпадающего списка. На диаграмме реляционной модели в соответствующих строках были отображены выбранные типы данных Linter (Рисунок 4.8).

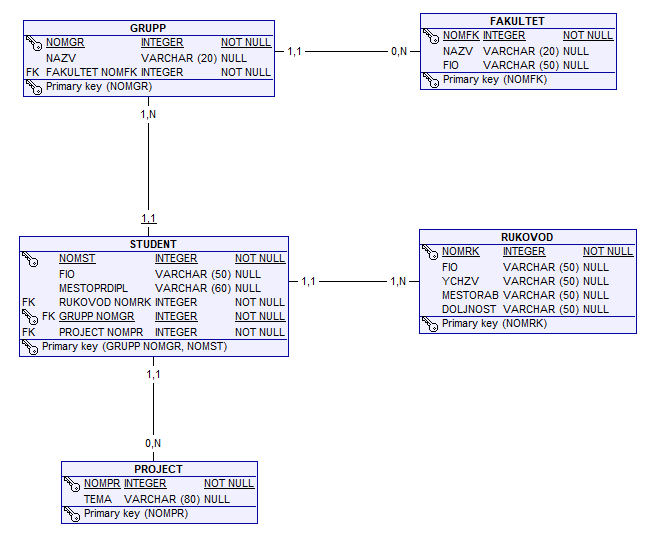


Рисунок 4.8 — Типы столбцов созданных таблиц

Далее были сгенерированы внешние ключи. Для этого использовалась команда меню «Сервис 🡪 Модель данных 🡪 Сгенерировать внешние ключи».

В открывшемся окне, представленном на рисунке 4.9, была выбрана кнопка «Генерация внешних ключей».

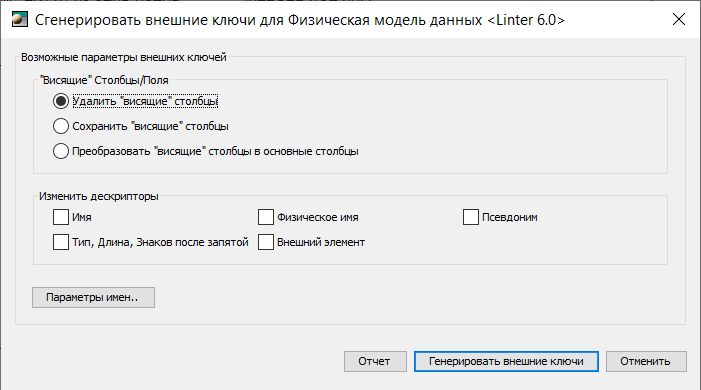


Рисунок 4.9 — Генерация внешних ключей

Как видно из отчета о генерации внешних ключей, представленном на рисунке 4.10, сгенерированы три новых внешних столбца и три новых внешних ключа.

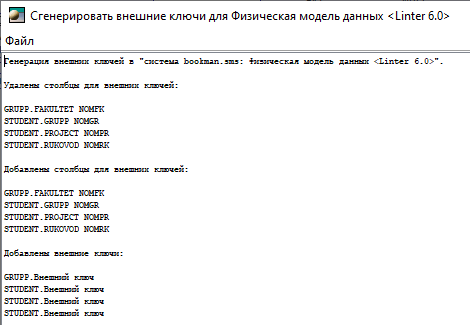


Рисунок 4.10 — Отчет о генерации внешних ключей

Модель после генерации внешних ключей показана на рисунке 4.11.

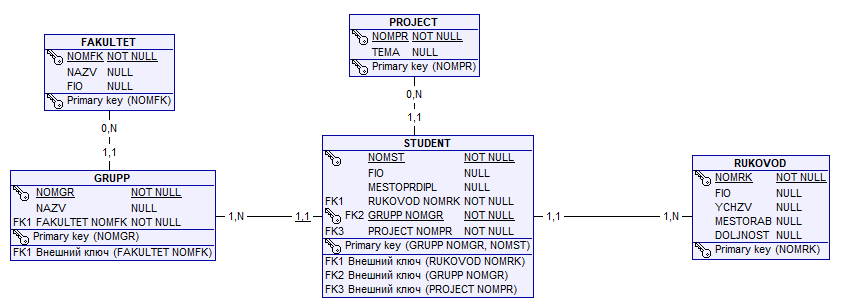


Рисунок 4.11 — Модель после генерации внешних ключей

После ввода вручную физических имен всех таблиц и их столбцов, остаются без физических имен первичные и внешние ключи всех таблиц физической модели.

Создать физические имена первичных и внешних ключей можно вручную или с помощью генератора физических имен системы Open ModelSphere. Однако, возможности автоматической генерации физических имен довольно ограничены, генерируемые таким образом имена далеки от совершенства и имеют большую длину. Поэтому необходимо воспользоваться ручным способом.

Для заданного варианта физические имена первичных и внешних ключей приведены в таблицах 4.1-4.2.

Таблица 4.1 - Первичные ключи

|  |  |  |
| --- | --- | --- |
| Имя таблицы | Имя ключа | Нулевое значение |
| GRUPP | NOMGR | Недопустимо |
| STUDENT | NOMST | Недопустимо |
| PROJECT | NOMPR | Недопустимо |
| RUKOVOD | NOMRK | Недопустимо |
| FAKULTET | NOMFK | Недопустимо |

Таблица 4.2 - Внешние ключи

|  |  |  |
| --- | --- | --- |
| Имя таблицы | Имя ключа | Нулевое значение |
| STUDENT | RUKOVOD NOMRK | Недопустимо |
| GRUPP NOMGR | Недопустимо |
| PROJECT NOMPR | Недопустимо |
| GRUPP | FAKULTET NOMFK | Недопустимо |

Для выполнения генерации указанных физических имен с помощью средств ModelSphere был выбран в окне «Навигатор» узел «Физическая модель данных <Linter 6.0>» и была выполнена команда меню «Сервис 🡪 Генерировать 🡪 Физические имена». В результате было открыто окно «Генерация физических имен», как представлено на рисунке 4.12.

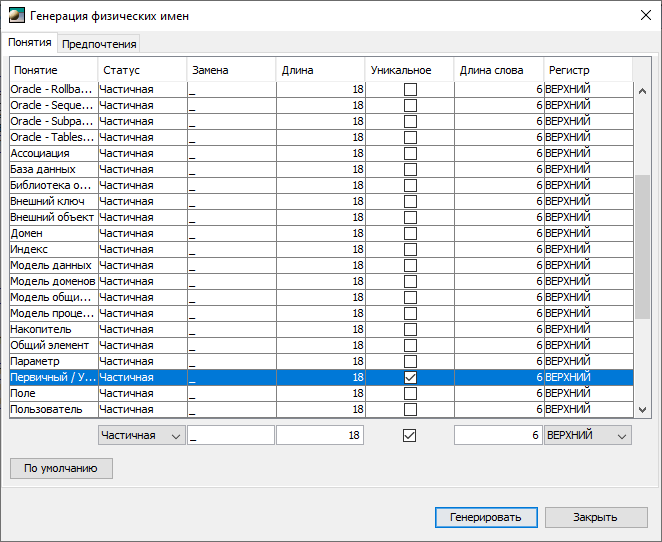


Рисунок 4.12 — Генерирование физических имен первичных и внешних ключей

В открывшемся окне в поле «Статус» можно было увидеть слово «Частичная», которое означает, что только отсутствующие физические имена были модифицированы, так как ранее физические имена всех атрибутов таблиц уже были определены. В поле «Замена» расположен символ замены недопустимых символов, таких как пробел. В открывшемся окне выделены строки с именами «Внешний ключ» и «Первичный/Альтернативный Ключ». Затем были установлены в столбце «Уникальное» в нужных строках флажки, чтобы физические имена были уникальными. Остальные настройки не изменялись. Затем была нажата кнопка «Генерировать».

В результате сгенерированы физические имена всех первичных и внешних ключей физической модели данных. Кроме того, были сгенерированы физические имена для таблиц и ролей, которые ранее не были определены вручную (Рисунок 4.13).

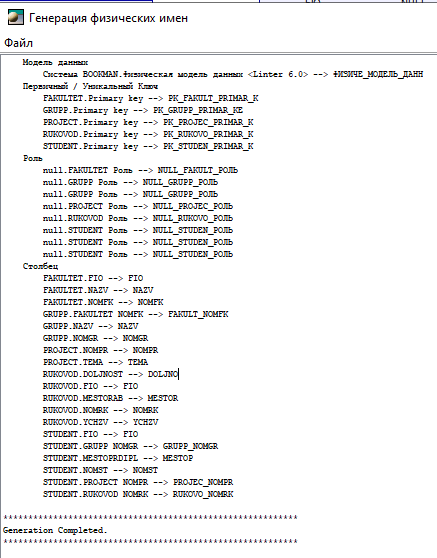


Рисунок 4.13 — Отчет о генерировании физических имен первичных и внешних ключей

В отчете о генерации физических имен можно увидеть, что сгенерированные физические имена имеют значительно большую длину, чем те имена, которые были введены вручную. Кроме того, имена внешних ключей не содержат названий таблиц, в которых они используются.

Индексы в базе данных используются в следующих основных случаях:

1) ускорение выполнения запросов;

2) обеспечение уникальности значений в полях. Ограничение первичного ключа требует, чтобы во всей таблице не нашлось двух одинаковых значений полей, входящих в первичный ключ. Чтобы выполнить это условие, необходимо при каждой вставке новой записи производить поиск такого же значения, которое будет вставлено. Для поиска записей используется уникальный индекс;

3) обеспечение ссылочной целостности. Ограничения внешних ключей используются для проверки того, чтобы вставляемые в таблицу значения обязательно существовали в другой таблице. При создании внешнего ключа индекс применяется для проверки условий внешнего ключа.

Для создания уникального индекса для первичного ключа в таблице было щелкнуто по первичному ключу (для таблицы FAKULTET по строке Primary key (NOMFK)) правой кнопкой. Из контекстного меню была выбрана команда «Добавить 🡪 Индекс». В нижней части изображения таблицы появилась новая строка, в которой была расположена запись о добавленном индексе (для таблицы FAKULTET это будет X1 Primary key (NOMFK) UNIQUE). Далее была выбрана строка «Свойства». В результате было открыто окно со свойствами добавленного индекса, изображенного на рисунке 4.14.

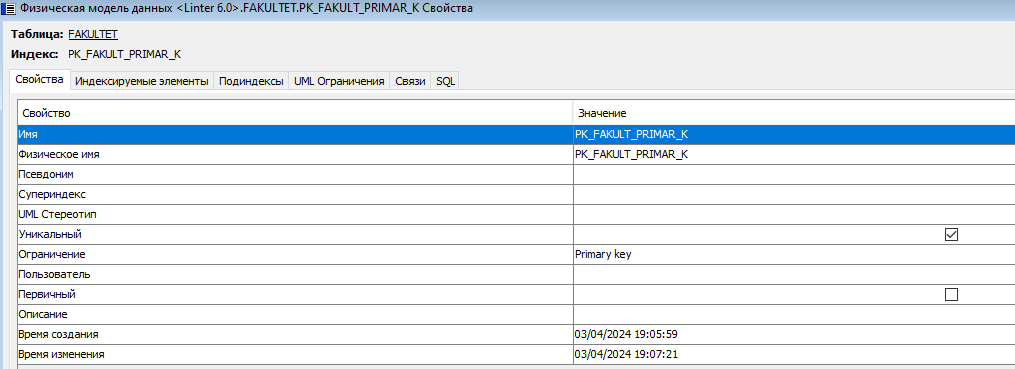


Рисунок 4.14 — Свойства уникального индекса для первичного ключа

На вкладке «Свойства» этого окна было изменено «Имя» этого индекса и вместо значения Primary key введено PK\_FAKULTET\_PRIMAR\_K (значение физического имени) и нажата кнопка «Применить» (Рисунок 4.14). Затем открыта вкладка «Индексируемые элементы», представленная на рисунке 4.15, и в столбце «Сортировка» в строке с именем элемента необходимо было щелкнуть два раза и выбрать из контекстного меню строку ASC (сортировка по возрастанию). После этого была нажата кнопка «Применить», а затем кнопка «Закрыть».

На изображении будут видны внесенные изменения.

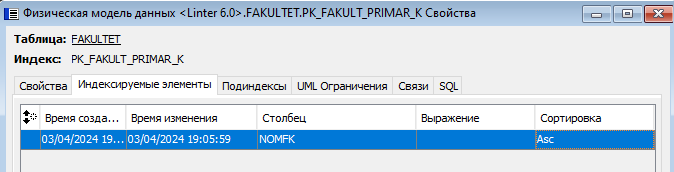


Рисунок 4.15 — Определение направления сортировки в индексе

Для создания индекса для внешнего ключа в таблице было нажато по внешнему ключу (для таблицы GRUPP по строке FK1 Foreign key (FAKULTET NOMFK)) правой кнопкой. Из контекстного меню была выбрана команда «Добавить 🡪 Индекс». В нижней части изображения таблицы появилась новая строка, в которой была расположена запись о добавленном индексе (для таблицы GRUPP это будет X1 Foreign key (FAKULTET NOMFK) NOT UNIQUE).

Далее было нажато правой кнопкой по этой записи и из контекстного меню выбрана строка «Свойства». В результате было открыто окно со свойствами добавленного индекса, представленное на рисунке 4.16.

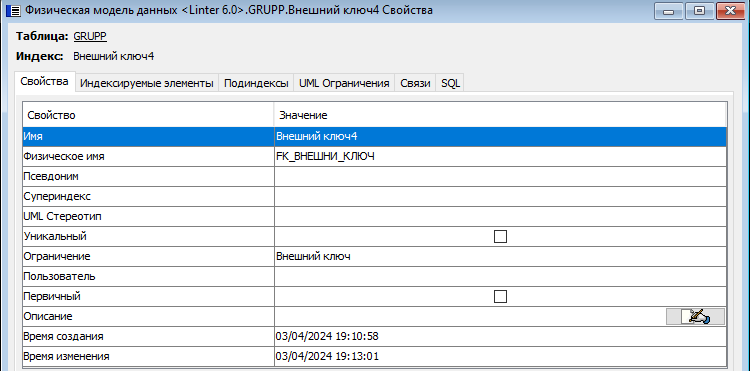


Рисунок 4.16 — Свойства индекса для внешнего ключа

На вкладке «Свойства» этого окна было изменено «Имя» этого индекса и вместо значения Foreign key введено FK\_ВНЕШНИ\_КЛЮЧ4 (значение физического имени) и нажата кнопка «Применить». Затем открыта вкладка «Индексируемые элементы», представленная на рисунке 4.17, и в столбце «Sort» в строке с именем элемента выбрана из контекстного меню строку ASC (сортировка по возрастанию). После этого была нажата кнопка «Применить», а затем на кнопку «Закрыть».

На изображении таблицы станут видны внесенные изменения.

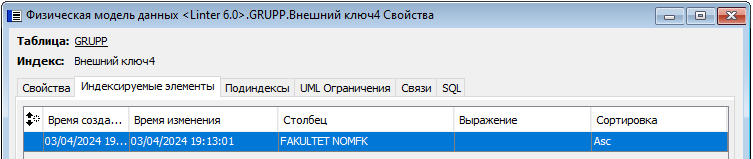


Рисунок 4.17 — Определение направления сортировки в индексе

Аналогичным образом добавлены ко всем таблицам физической модели индексы для первичных и внешних ключей. В результате физическая модель данных имеет вид, приведенный на рисунке 4.18.

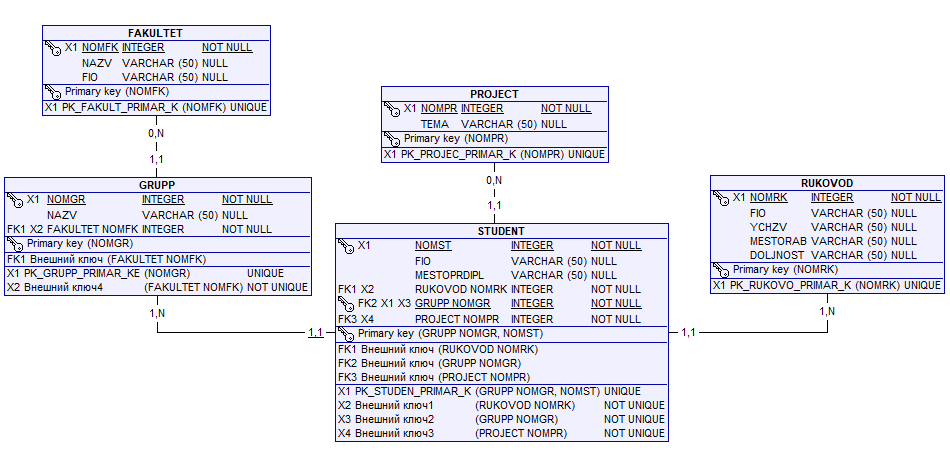


Рисунок 4.18 — Окончательный вид физической модели данных

Были определены домены для каждого атрибута всех таблиц в соответствие с СУБД Линтер. Описание типов данных атрибутов приведено в таблицах 4.3 — 4.7.

Таблица 4.3 - Описание атрибутов таблицы GRUPP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя атрибута | Физическое имя | Ключевое поле | Нулевое значение | Тип данных |
| NOMGR | NOMGR | PK | Недопустимо | INTEGER |
| NAZV | NAZV | - | Допустимо | VARCHAR(50) |

Таблица 4.4 — Описание атрибутов таблицы STUDENT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя атрибута | Физическое имя | Ключевое поле | Нулевое значение | Тип данных |
| 1 | 2 | 3 | 4 | 5 |
| NOMST | NOMST | PK | Недопустимо | INTEGER |
| FIO | FIO | - | Допустимо | VARCHAR(50) |

Продолжение таблицы 4.4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| MESTOPRDIPL | MESTOPRDIPL | - | Допустимо | VARCHAR(50) |
| NOMGR | NOMGR | PK, FK | Недопустимо | INTEGER |
| NOMRUK | NOMRUK | FK | Недопустимо | INTEGER |
| NOMPR | NOMPR | FK | Недопустимо | INTEGER |

Таблица 4.5 — Описание атрибутов таблицы RUKOVOD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя атрибута | Физическое имя | Ключевое поле | Нулевое значение | Тип данных |
| NOMRUK | NOMRUK | PK | Недопустимо | INTEGER |
| FIO | FIO | - | Допустимо | VARCHAR(50) |
| YCHZV | YCHZV | - | Допустимо | VARCHAR(50) |
| MESTORAB | MESTORAB | - | Допустимо | VARCHAR(50) |
| DOLJNOST | DOLJNOST | - | Допустимо | VARCHAR(50) |

Таблица 4.6 — Описание атрибутов таблицы PROJECT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя атрибута | Физическое имя | Ключевое поле | Нулевое значение | Тип данных |
| NOMPR | NOMPR | PK | Недопустимо | INTEGER |
| TEMA | TEMA | - | Допустимо | VARCHAR(50) |

Таблица 4.7 — Описание атрибутов таблицы FAKULTET

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя атрибута | Физическое имя | Ключевое поле | Нулевое значение | Тип данных |
| NOMFK | NOMFK | PK | Недопустимо | INTEGER |
| NAZV | NAZV | - | Допустимо | VARCHAR(20) |
| FIO | FIO | - | Допустимо | VARCHAR(50) |

Затем необходимо было проверить целостность физической модели с

помощью команды меню «Сервис → Проверить целостность».

В отчете о проверке целостности физической модели били выведены сообщения о том, что для всех связей модели не определены параметры ссылочной целостности модели. Поэтому, затем необходимо было выполнить генерацию правил для ссылочной целостности. Для этого была выполнена команда главного меню «Сервис → Модель данных→ Сгенерировать правила ссылочной целостности», изображенном на рисунке 19. В открывшемся окне была нажата кнопка ОК, изображенном на рисунке 20.

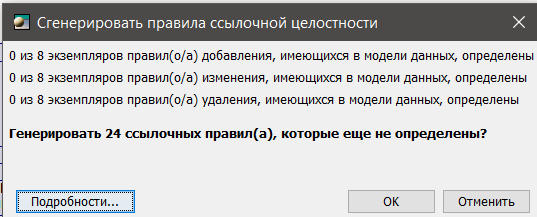


Рисунок 4.19 – Генерация правил ссылочной целостности

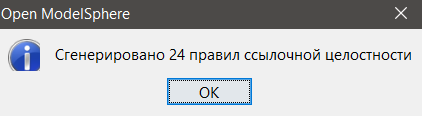


Рисунок 4.20 – Отчет о генерации

После этого необходимо снова проверить целостность скорректированной физической модели, после чего в отчете о проверке не должно быть никаких сообщений об ошибках. Результат проверки целостности представлен на рисунке 21.

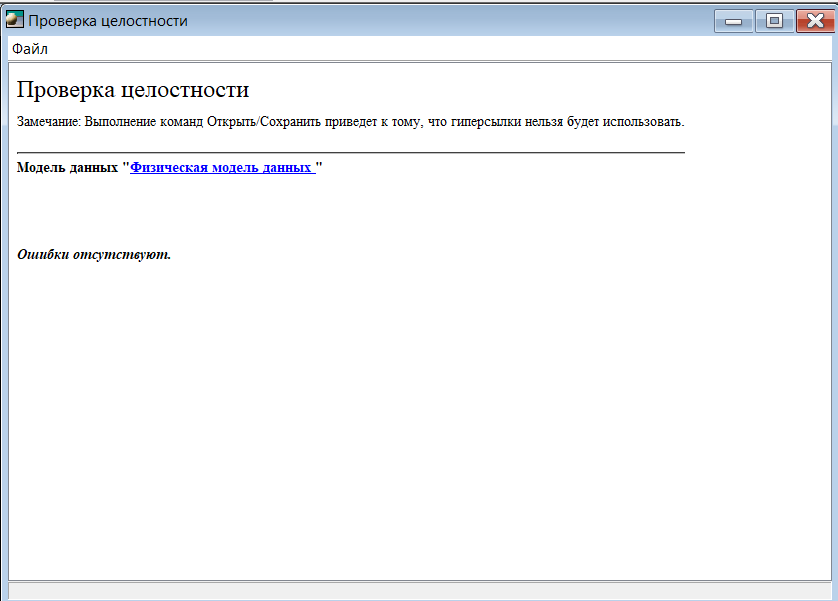


Рисунок 4.21 - Отчет о проверке целостности модели

Затем было необходимо сгенерировать DDL (Data Definition Language) скрипт для создания из текущей физической модели базы данных Linter. Для этого необходимо предварительно в дереве проекта создать узел используемой базы данных (в нашем случае Linter). Однако, ModelSphere позволяет это сделать автоматически в процессе генерации DDL скрипта.

Для этого необходимо выделить узел «Физическая модель данных <Linter 6.0>» в окне «Навигатор» и выполнить команду главного меню «Сервис →База данных → Прямое проектирование». Перед генерацией скрипта, было выведено окно с предупреждениями о том, что в процессе работы будет создан узел используемой базы данных и что эта база данных будет автоматически связана с физической моделью данных, как изображено на рисунке 22.

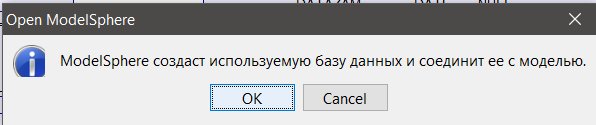


Рисунок 4.22 – Окно с предупреждением

При этом в дереве проекта появился новый узел с именем «База данных». Кроме того, откроется окно «Ansi Forward Engineering», изображенное на рисунке 23, в котором необходимо настроить свойства прямого проектирования базы данных.

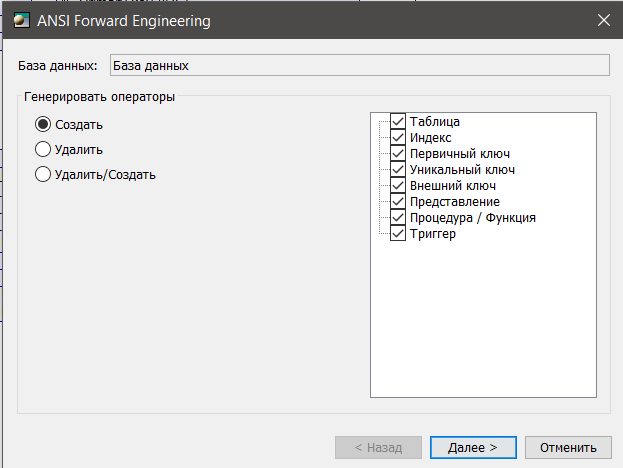


Рисунок 4.23 – Настройка прямого проектирования БД

Необходимо было выбрать переключатель «Создать» для всех существующих в базе данных объектов. Для продолжения необходимо нажать кнопку «Далее» (рисунок 24).

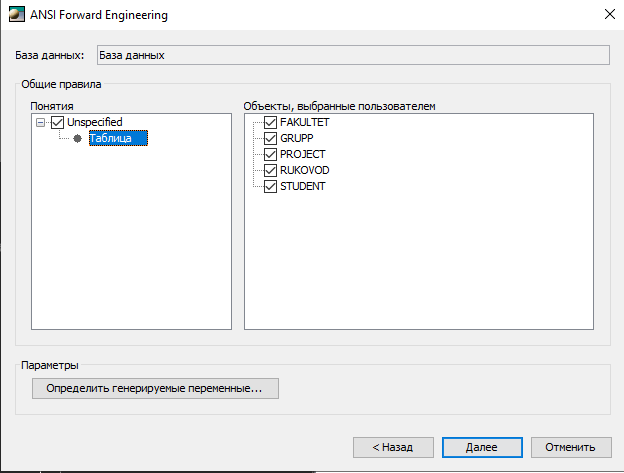


Рисунок 4.24 – Выбор создаваемых объектов

Затем необходимо установить переменные для генерации скрипта, для чего нужно щелкнуть по кнопке «Определить генерируемые переменные». В открывшемся окне необходимо установить флажки, как показано на рисунке 25.

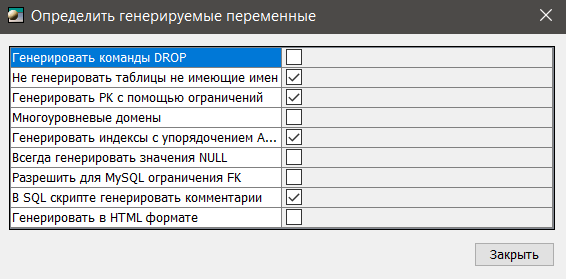


Рисунок 4.25 - Установка переменных для генерации скрипта

Затем необходимо нажать кнопку «Закрыть», а затем на кнопку «Далее». В результате было открыто окно «Сохранить SQL скрипт как», представленное на рисунке 26.

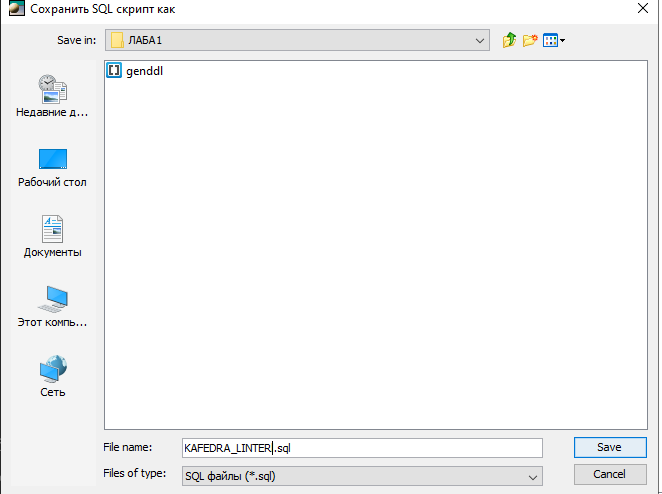


Рисунок 4.26 – Сохранение скрипта для создания БД

После чего необходимо определить имя выходного файла, в котором будет храниться SQL скрипт для создания объектов базы данных. Результат генерации скрипта представлен на рисунке 27.

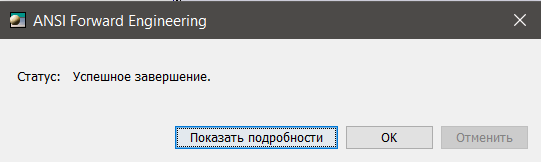


Рисунок 4.27 – Сообщение о генерации скрипта

Текст скрипта приведен в приложении А.

Таким образом, была разработана физическая модель базы данных для СУБД Линтер.

4.2 Разработка мандатной защиты данных

Мандатная защита информации обеспечивает высокую безопасность информационных систем путем введения многоуровневой системы доступа к данным. Эта система основана на принципе мандатного доступа, который определяет уровень секретности для каждого элемента информации.

Уровни доступа в СУБД представляют собой числовые значения от 1 до 10, где более высокое значение указывает на более высокий уровень секретности. Эти уровни назначаются пользователям, объектам БД и даже отдельным данным в строках таблиц.

Уровни доступа пользователей включают:

- уровень чтения (RAL): Пользователи могут читать информацию, уровень секретности которой не превышает их собственного уровня доступа.

- уровень доверия (WAL): Ограничивает пользователей от понижения уровня секретности информации. Они не могут записывать информацию с уровнем секретности ниже данного WAL.

Уровни доступа для информации включают:

- уровень чтения (RAL): Определяет минимальный уровень доступа, необходимый для чтения данных.

- уровень записи (WAL): Ограничивает уровень доступа для записи данных. Информация может быть изменена или удалена только пользователями с уровнем доступа не ниже WAL модифицируемой информации.

Для таблиц и их столбцов уровни доступа имеют несколько иное значение:

- уровень чтения (RAL): Минимальный уровень доступа, необходимый для получения данных из таблицы.

- уровень записи (WAL): Ограничивает минимальный уровень чтения данных, которые могут быть вставлены в таблицу.

СУБД сравнивает уровни доступа пользователей и объектов БД при попытке доступа. Это гарантирует, что пользователям предоставляется доступ к информации в соответствии с их уровнем секретности.

# 5 Проектирование приложения

5.1 Анализ функций приложения

Анализ функций приложения приведен на основе анализа предметной области и требований к функциональным характеристикам. Результат анализа функций представлен на рисунке 5.1.

Командой «Пуск > Все программы > СУБД ЛИНТЕР > Администратор СУБД ЛИНТЕР» был запущен на выполнение «Администратор СУБД ЛИНТЕР». В дереве баз данных имеется список БД установленных сервере HOME-PC (Рисунок 5.1).

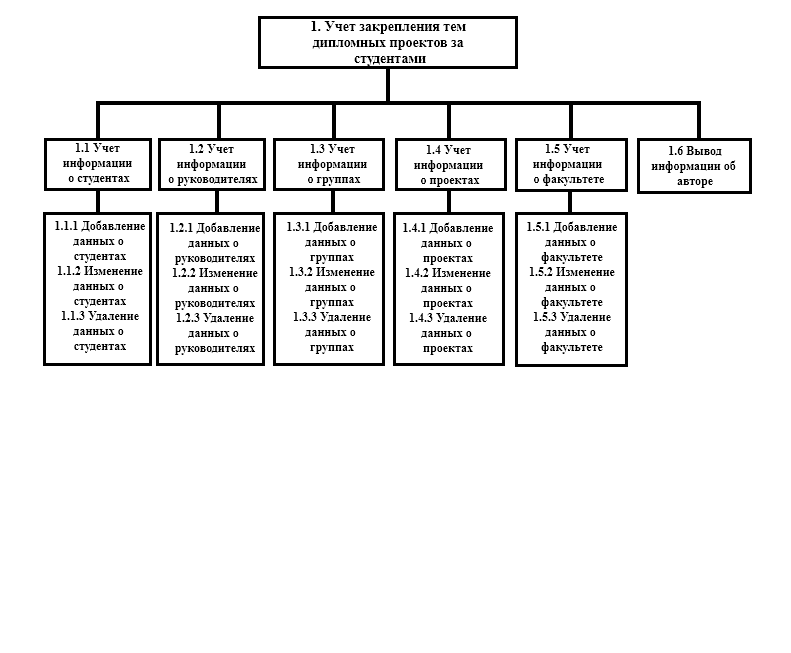


Рисунок 5.1 - Иерархия функций приложения

5.2 Определение описания функций

Функция «Учет информации о факультете» имеет 3 подфункции:

1) Добавление данных о факультете.

Для добавления информации о факультете необходимо открыть набор данных «FAKULTET» ввести на форму необходимые сведения о новом факультете, подтвердить добавление и закрыть набор данных.

2) Изменение данных о факультете.

Для изменения информации о факультете необходимо открыть набор данных «FAKULTET» ввести на форму изменение сведения, подтвердить изменения и закрыть набор данных.

3) Удаление данных о факультете.

Для удаления информации о факультете необходимо открыть набор данных «FAKULTET» выбрать данные, которые нужно удалить, подтвердить удаление и закрыть набор данных.

5.3 Отображение функций в модули

Соотношение функций и моделей представлено в таблице 5.1 и на рисунке 5.2.

Таблица 5.1 - Соотношение функций и моделей

|  |  |
| --- | --- |
| Функция | Модули |
| Добавление информации о факультете | Учет информации о факультете |
| Изменение информации о факультете |
| Удаление информации о факультете |
| Добавление информации о группах | Учет информации о группах |
| Изменение информации о группах |
| Удаление информации о группах |
| Добавление информации о студентах | Учет информации о студентах |
| Изменение информации о студентах |
| Удаление информации о студентах |
| Добавление информации о руководителях | Учет информации о руководителях |
| Изменение информации о руководителях |
| Удаление информации о руководителях |
| Добавление информации о проектах | Учет информации о проектах |
| Изменение информации о проектах |
| Удаление информации о проектах |
| Вывод информации о разработчике | Главный модуль |

Отображение функций в модуле представлен на рисунке 5.2.

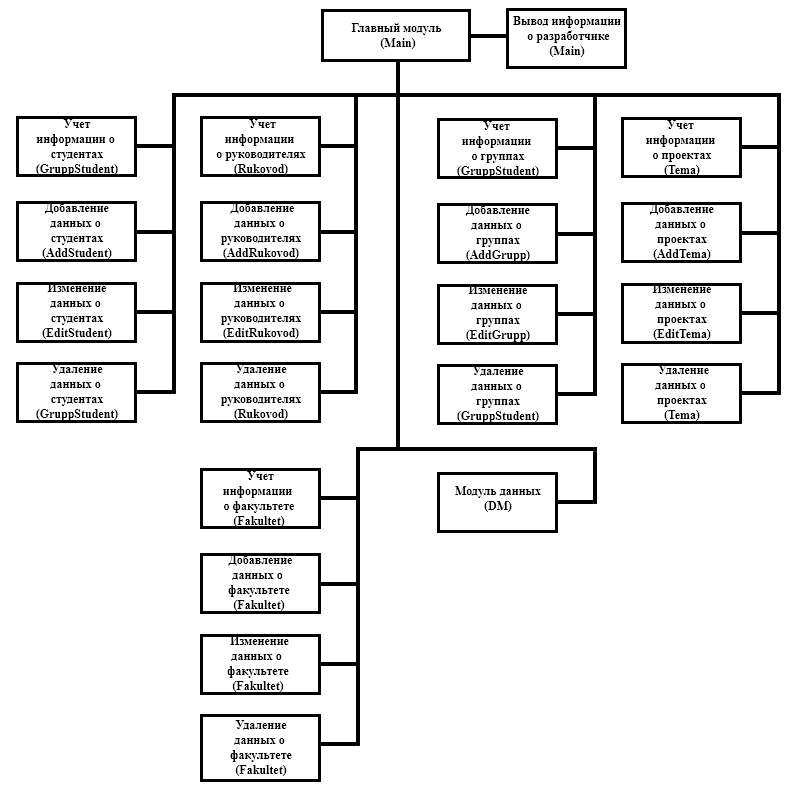


Рисунок 5.2 — Соотношение функций и модулей

6 Описание программы

6.1 Общие сведения

Разработанная программа имеет наименование системы «Автоматизированная система кафедры». Текст программы находится в файле «Predp.exe».

Программа написана на языке программирования Object Pascal, в среде визуального программирования Delphi 10 Tokyo, c исполнение компонентов dbExpress. Для функционирования программы должна быть установлена СУБД Линтер. Листинг программы приведен в приложении В.

6.2 Функциональное назначение

Программа предназначена для работы с базой данных, содержащей сведения о кафедре.

6.3 Логическая структура программы

Программа является событийно-управляемой. При запуске программы открывается главное окно. Приложение для работы с БД имеет модульную структуру. Схема взаимодействия модулей приведена на рисунке 6.1.

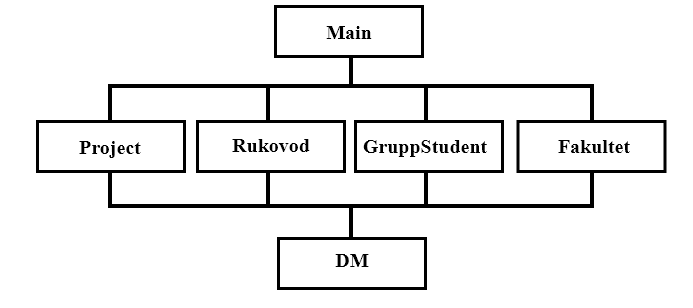


Рисунок 6.1 — Схема взаимодействия модулей приложения

Вызов модулей приложения производится в соответствии с рисунком 6.1.

При запуске программы на выполнение запускается главный модуль Main. С помощью главного модуля можно вызвать модули:

- FAKULTET – для просмотра, добавления, изменения и удаления данных об отделах из таблицы FAKULTET;

- GRUPP - для просмотра, добавления, изменения и удаления данных о накладная и строки накладной из таблиц GRUPP и STUDENT;

- RUKOVOD - для просмотра, добавления, изменения и удаления данных о товарах из таблицы RUKOVOD;

- PROJECT - для просмотра, добавления, изменения и удаления данных о товарах из таблицы PROJECT;

- DM – не визуальный модуль данных. С помощью компонентов установленных на модуле DM осуществляется связь приложения с базой данных. Данный модуль является не визуальным и не доступным для пользователя.

Модули Main, FAKULTET, GRUPP, RUKOVOD, PROJECT связаны с модулем данных DM.

Описание модулей и соответствующих им форм приведено в таблице 6.1

Таблица 6.1 — Описание модулей

|  |  |  |
| --- | --- | --- |
| Наименование модуля | Наименование форм | Примечание |
| Main | fmMain | Главный модуль программы |
| DM | fmDM | Модуль данных |
| FAKULTET | fmFAKULTET | Для просмотра и изменения данных о факультете |
| GRUPP | fmGRUPP | Для просмотра и изменения данных о группах |
| RUKOVOD | fmRUKOVOD | Для просмотра и изменения данных о руководителях |
| PROJECT | fmPROJECT | Для просмотра и изменения данных о проектах |

Главная форма fm\_Main представлена на рисунке 6.2

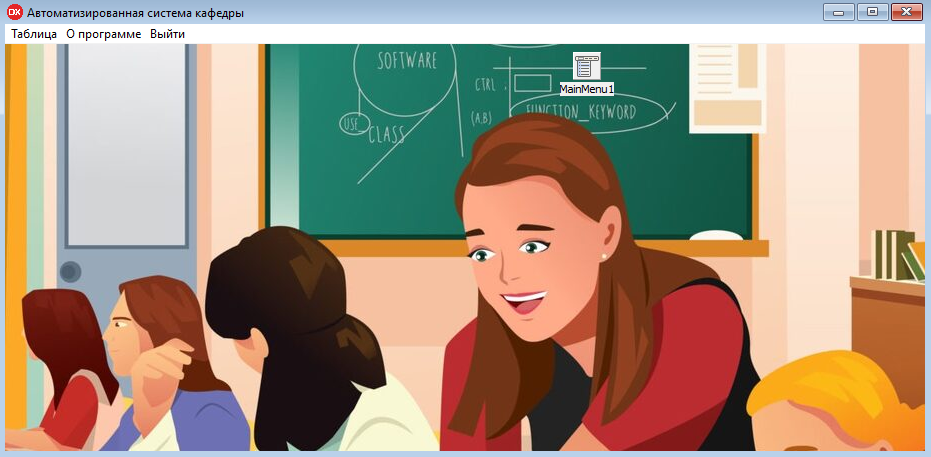


Рисунок 6.2 — Вид формы fmMain

На главной форме приложения размещены следующие элементы:

- компонент меню MainMenu (Menu) с вкладки Standart;

- компонент Image (Image1) для вывода фонового изображения главной формы с вкладки Additional.

Данные компоненты отображены в дереве объектов на рисунке 6.3.

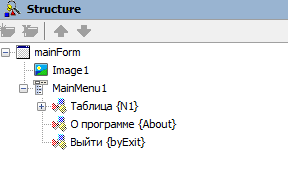


Рисунок 6.3 — Дерево объектов для формы fmMain

Компонент Menu используется для создания главного меню программы. Главное меню состоит из трех пунктов. Пункт «Выход» осуществляет завершение работы программы. Пункт меню «О программе», при выборе которого выводится информация о программе и о ее разработчике. При выборе пункта меню «Таблицы» открывается подменю, в котором расположены пункты «Факультет», при клике на который открывается форма fmFAKULTET, пункт «Группа-Студенты», при клике на который открывается форма fmGRUPP, пункт «Руководитель», при клике на который открывается форма fmRUKOVOD и пункт «Проекты», при клике на который открывается форма fmPROJECT.

Компонент Image позволил разместить на форме fmMain фоновое изображение склада предприятия.

Форма fmFAKULTET представлена на рисунке 6.4

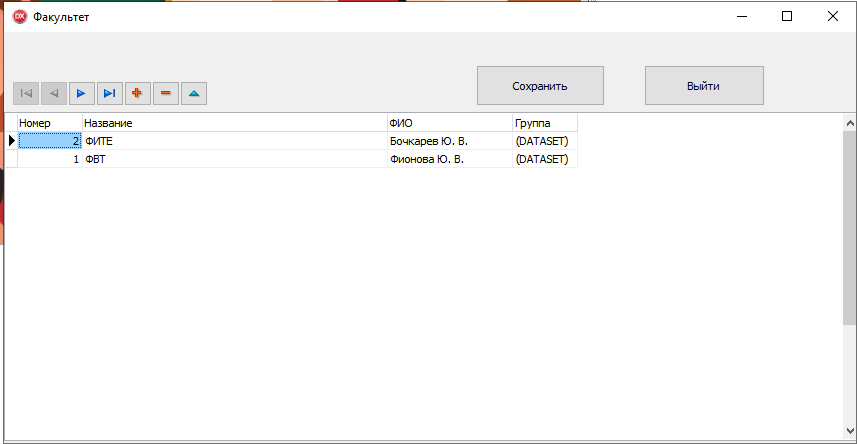


Рисунок 6.4 — Вид формы fmFAKULTET

На форме fmFAKULTET размещены следующие элементы:

- таблица DBGrid (DBGrid1) с вкладки Data Controls;

- компонент навигатор TDBNavigator (TDBNavigator1) c вкладки Data Control;

- два компонента Button (buSave, buClose) с вкладки Standart.

Компонент DBGrid1 используется для отображения данных таблицы FAKULTET базы данных BASOV.lhb. Свойства компонента DBGrid1 представлены на рисунке 6.5.

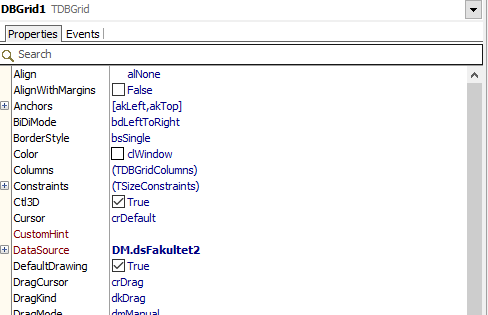


Рисунок 6.5 — Свойства компонента DBGrid1 таблицы FAKULTET

Для добавления, изменения и удаления данных из таблицы FAKULTET используется компонент TDBNavigator1. Свойство компонента TDBNavigator1 представлены на рисунке 6.6. При нажатии на кнопку buSave все изменения отправляются на сервер, после чего они вступают в силу. При нажатии на кнопку buClose происходит закрытие формы fmFAKULTET.

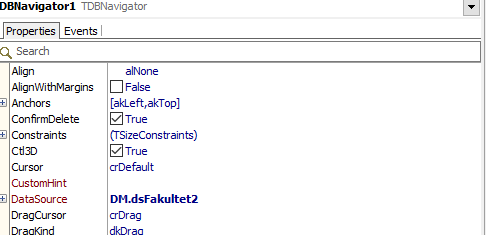


Рисунок 6.6 — Свойства компонента TDBNavigator1 таблицы FAKULTET

Форма fmRUKOVOD представлена на рисунке 6.7.

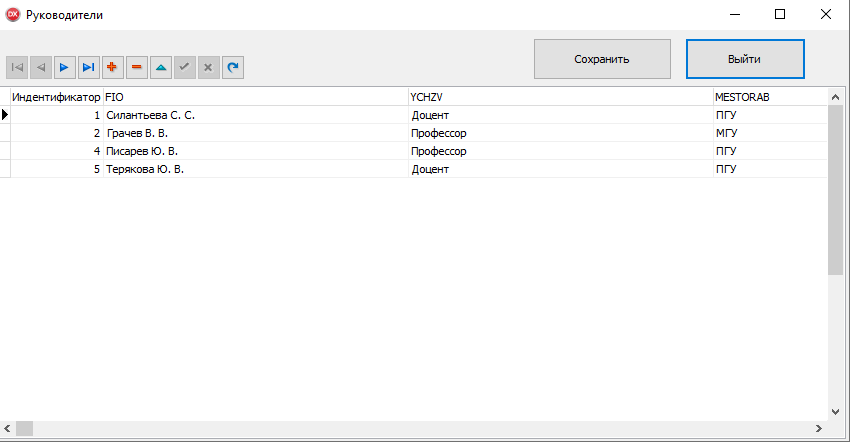


Рисунок 6.7 — Вид формы fmRUKOVOD

На форме fmRUKOVOD размещены следующие элементы:

- таблица DBGrid (DBGrid1) с вкладки Data Controls;

- компонент навигатор TDBNavigator (TDBNavigator1) c вкладки Data Control;

- два компонента Button (buSave, buClose) с вкладки Standart.

Компонент DBGrid1 используется для отображения данных таблицы RUKOVOD базы данных BASOV.lhb. Свойства компонента DBGrid1 представлены на рисунке 6.8.

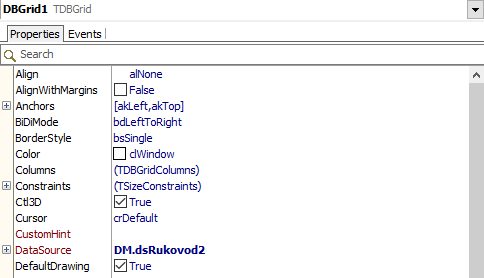


Рисунок 6.8 — Свойства компонента DBGrid1 таблицы RUKOVOD

Для добавления, изменения и удаления данных из таблицы RUKOVOD используется компонент TDBNavigator1. Свойство компонента TDBNavigator1 представлены на рисунке 6.9. При нажатии на кнопку buSave все изменения отправляются на сервер, после чего они вступают в силу. При нажатии на кнопку buClose происходит закрытие формы fmRUKOVOD.

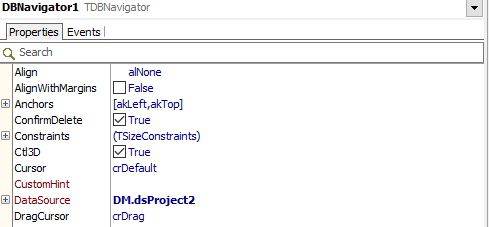


Рисунок 6.9 — Свойства компонента TDBNavigator1 таблицы RUKOVOD

Форма fmGRUPP представлена на рисунке 6.10.

Рисунок 6.10 — Вид формы fmGRUPP

На форме fmGRUPP размещены следующие элементы:

- таблица DBGrid (DBGrid1, DBGrid2) с вкладки DataControls;

- компонент навигатор TDBNavigator (TDBNavigator1, TDBNavigator2) c вкладки Data Control;

- два компонента Button (buSave, buClose) с вкладки Standart.

- два компонент DBLookupComboBox с вкладки DataControls;

Компонент DBGrid1 используется для отображения данных таблицы GRUPP базы данных BASOV.lhb. Свойства компонента DBGrid1 представлены на рисунке 6.11.

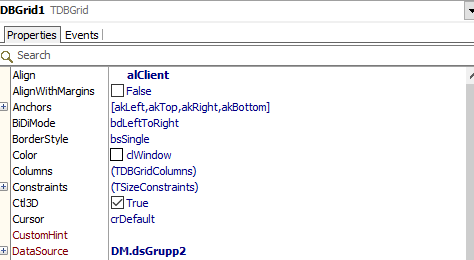


Рисунок 6.11 — Свойства компонента DBGrid1 таблицы GRUPP

Компонент DBGrid2 используется для отображения данных таблицы STUDENT базы данных BASOV.lhb. Свойства компонента DBGrid2 представлены на рисунке 6.12.

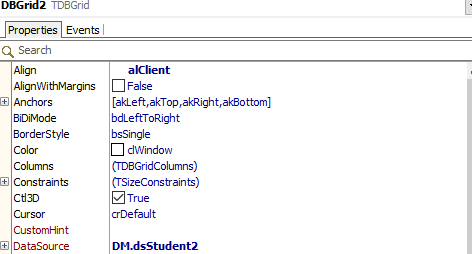


Рисунок 6.12 — Свойства компонента DBGrid2 таблицы STUDENT

Для добавления, изменения и удаления данных из таблицы GRUPP используется компонент TDBNavigator1. Свойство компонента TDBNavigator1 представлены на рисунке 6.13. При нажатии на кнопку buSave все изменения отправляются на сервер, после чего они вступают в силу. При нажатии на кнопку buClose происходит закрытие формы fmGRUPP.

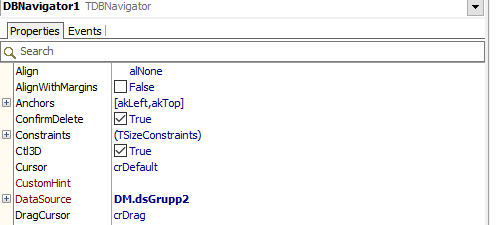


Рисунок 6.13 — Свойства компонента TDBNavigator1 таблицы GRUPP

Для добавления, изменения и удаления данных из таблицы STUDENT используется компонент TDBNavigator2. Свойство компонента TDBNavigator2 представлены на рисунке 6.14. При нажатии на кнопку buSave все изменения отправляются на сервер, после чего они вступают в силу. При нажатии на кнопку buClose происходит закрытие формы fmSTUDENT.

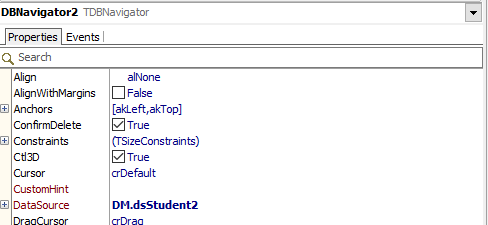


Рисунок 6.14 — Свойства компонента TDBNavigator1 таблицы STUDENT

Для выбора отдела из таблицы OTDEL используется компонент DBLookupComboBox1. Свойство компонента DBLookupComboBox1 представлены на рисунке 6.15.



Рисунок 6.15 — Свойства компонента DBLookupComboBox1 таблицы FAKULTET

Форма fmPROJECT представлена на рисунке 6.16.

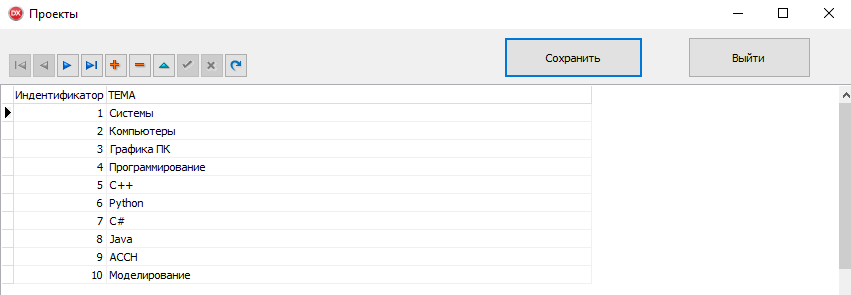


Рисунок 6.16 — Вид формы fmPROJECT

На форме fmPROJECT размещены следующие элементы:

- таблица DBGrid (DBGrid1) с вкладки Data Controls;

- компонент навигатор TDBNavigator (TDBNavigator1) c вкладки Data Control;

- два компонента Button (buSave, buClose) с вкладки Standart.

Компонент DBGrid1 используется для отображения данных таблицы PROJECT базы данных BASOV.lhb. Свойства компонента DBGrid1 представлены на рисунке 6.17.

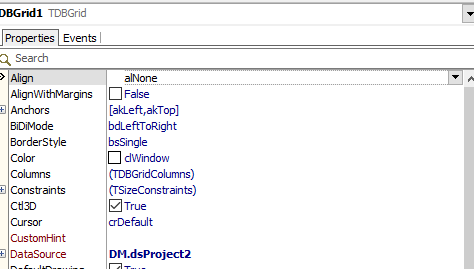


Рисунок 6.17 — Свойства компонента DBGrid1 таблицы PROJECT

Для добавления, изменения и удаления данных из таблицы PROJECT используется компонент TDBNavigator1. Свойство компонента TDBNavigator1 представлены на рисунке 6.18. При нажатии на кнопку buSave все изменения отправляются на сервер, после чего они вступают в силу. При нажатии на кнопку buClose происходит закрытие формы fmPROJECT.

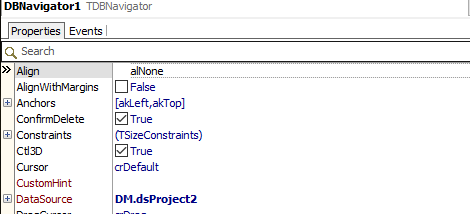


Рисунок 6.18 — Свойства компонента TDBNavigator1 таблицы PROJECT

Форма модуля данных DM представлена на рисунке 6.19.

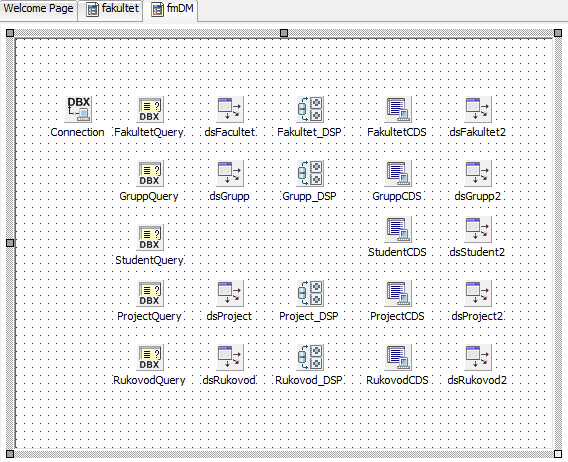


Рисунок 6.19 — Модуль данных DM

На модуль DM помещены следующие компоненты:

- компонент TSQLConnection с вкладки dbExpress;

- пять компонентов SQLQuery с вкладки dbExpress;

- четыре компонента DataSetProvider с вкладки Data Access;

- пять компонентов ClientDataSet с вкладки Data Access;

- девять компонентов DataSource с вкладки Data Access.

Компонент SQLСonnection устанавливает соединение с базой данных. Свойства этого компонента представлены на рисунке 6.20.

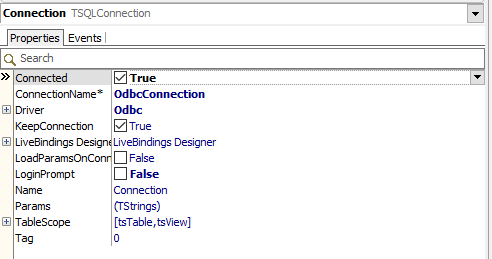


Рисунок 6.20 — Свойства компонента SQLСonnection

Редактор свойств Params этого компонента представлен на рисунке 6.19.

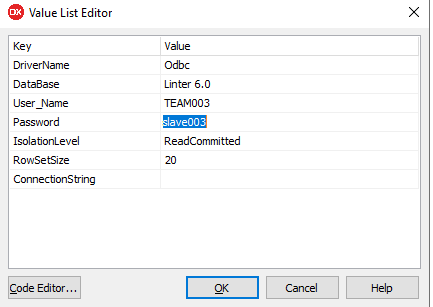


Рисунок 6.21 — Редактор свойств Params

Компонент SQLQuery возвращает однонаправленный набор записей. На форму помещены компоненты FakultetQuery, GruppQuery, StudentQuery, ProjectQuery, RukovodQuery для получения набора данных таблиц FAKULTET, GRUPP, STUDENT, PROJECT и RUKOVOD соответственно. Свойства этих компонентов представлены на рисунках 6.22 — 6.26.

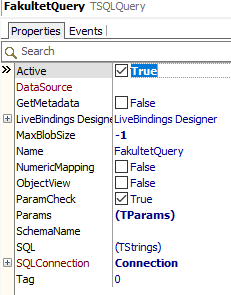
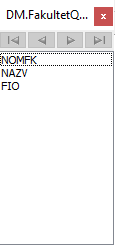
 

Рисунок 6.22 — Свойства компонента FakultetQuery

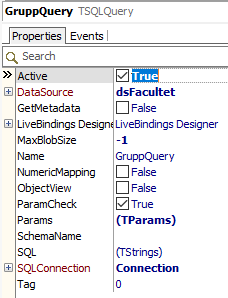
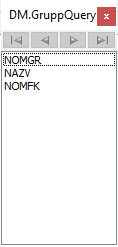
 

Рисунок 6.23 — Свойства компонента GruppQuery

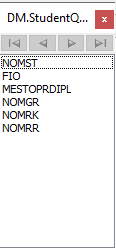
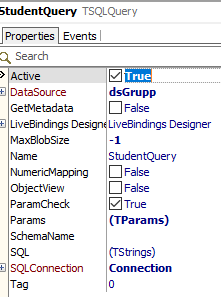


Рисунок 6.24 — Свойства компонента StudentQuery

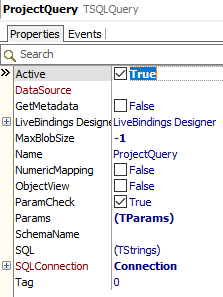
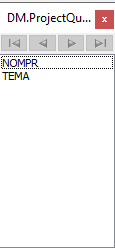
 

Рисунок 6.25 — Свойства компонента ProjectQuery

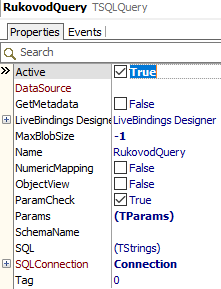
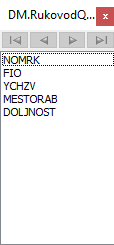
 

Рисунок 6.26 — Свойства компонента RukovodQuery

Компонент DataSetProvider связывается с компонентами DataSet через одноименное свойство. DataSetProvider поставляет данные компоненту ClientDataSet и генерирует операторы SQL DML для обновления базы данных, используя журнал изменений, накапливаемый ClientDataSet. Свойства компонентов Fakultet\_DSP, Grupp\_DSP, Project\_DSP, Rukovod\_DSP представлены на рисунках 6.27 — 6.30.

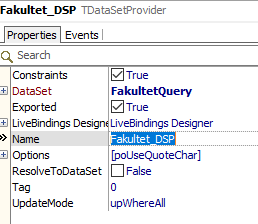


Рисунок 6.27 — Свойства компонента Fakultet\_DSP

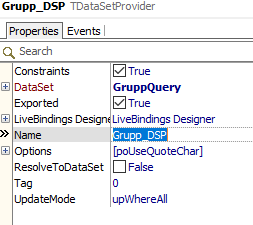


Рисунок 6.28 — Свойства компонента Grupp\_DSP

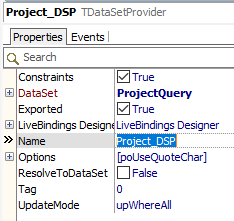


Рисунок 6.29 — Свойства компонента Project\_DSP

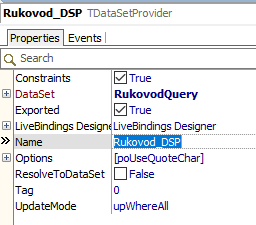


Рисунок 6.30 — Свойства компонента Rukovod\_DSP

ClientDataSet подсоединяется к DataSetProvider через свойство ProviderName. Он получает данные из DataSetProvider, буферизует данные памяти, регистрирует все изменения данных, и отправляет список изменений в DataSetProvider при вызове ClientDataSet.ApplyUpdates. Свойства этих компонентов представлены на рисунках 6.31 - 6.35.

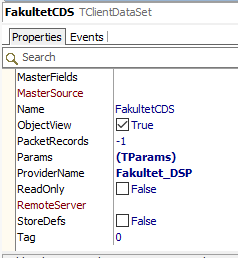


Рисунок 6.31 — Свойства компонента FakultetCDS

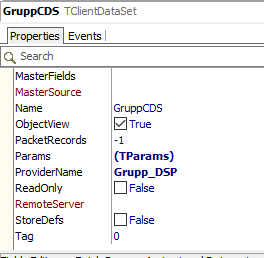


Рисунок 6.32 — Свойства компонента GruppCDS

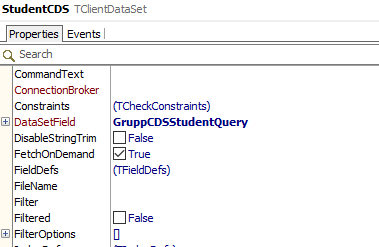


Рисунок 6.33 — Свойства компонента StudentCDS

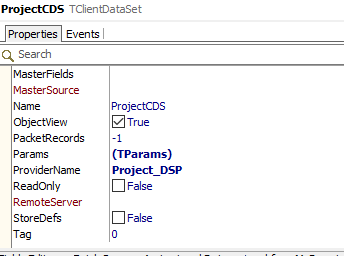


Рисунок 6.34 — Свойства компонента ProjectCDS

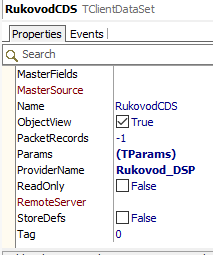


Рисунок 6.35 — Свойства компонента RukovodCDS

Вложенные наборы данных — многофункциональный инструмент, поскольку они исключают непонимание порядка обновления данных таблиц со связью главная — подчиненная. Например, если добавлены несколько записей в Master-таблицу и несколько записей в Detail-таблицу, то INSERT в Master-таблицу должен быть отправлен на сервер БД раньше, чем INSERT Detail-таблицу. И наоборот, если удаляются записи в подчиненной таблице, а затем записи в главной таблице, то оператор DELETE должен выполняться именно в таком порядке. При вложенных наборах данных DataSetProvider обеспечивает правильный порядок следования запросов на обновление, что исключает ошибки на сервере БД. Компоненты DataSource выполняют связь не визуальных источников данных с визуальными компонентами формы для отображения данных. Настройка всех компонентов DataSource представлена в таблице 6.2.

Таблица 6.2 — Настройка компонентов DataSource

|  |  |
| --- | --- |
| Имя компонента | Значение свойтва DataSet |
| dsFakultet | FakultetQuery |
| dsGrupp | GruppQuery |
| dsProject | ProjectQuery |
| dsRukovod | RukovodQuery |
| dsFakultet2 | FakultetCDS |
| dsGrupp2 | GruppCDS |
| dsProject2 | ProjectCDS |
| dsRukovod2 | RukovodCDS |

Текст dhm-файлов, используемых модулей представлены в приложении Г. Описание созданных обработчиков событий приложения приведено в таблице 6.3.

Таблица 6.3 - Описание обработчиков событий

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование модуля | Наименование формы | Наименование процедуры или функции | Описание |
| 1 | 2 | 3 | 4 |
| Main | fmMain | procedure TmainForm.AboutClick(Sender: TObject); | Процедура выполняет вывод информации о разработчике |
| procedure TmainForm.byExitClick(Sender: TObject); | Обработчик события закрытия формы fmMain приложения |
| procedure TmainForm.fakultetClick(Sender: TObject); | Оброботчик события осуществляет вызов формы fmFakultet |
| procedure TmainForm.projectClick(Sender: TObject); | Оброботчик события осуществляет вызов формы fmProject |
| procedure TmainForm.rukovodClick(Sender: TObject); | Оброботчик события осуществляет вызов формы fmRukovod |
| procedure TmainForm.gruppstudentClick(Sender: TObject); | Оброботчик события осуществляет вызов формы fmGruppStudent |

Продолжение таблицы 6.3

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| DM | fmDM | procedure TDM.FakultetCDSReconcileError(DataSet: TCustomClientDataSet;  E: EReconcileError; UpdateKind: TUpdateKind; var Action: TReconcileAction); | Вывод ошибки при некорректном вводе данных. |
| Fakultet | fmFakultet | procedure TtbFakultet.FormKeyPress(Sender: TObject; var Key: Char); | Оброботчик событий контроля введенных данных. |
| procedure TtbFakultet.bySaveClick(Sender: TObject); | Обработчик сохранения данных в таблице Fakultet |
| procedure TtbFakultet.byExitClick(Sender: TObject); | Обработчик собития осуществляет вызов сохранения данных в таблицах закрытия формы fmFakultet |
| procedure TtbFakultet.FormCreate(Sender: TObject); | Обработчик события отображения формы fmFakultet приложения программы. Процедура обработки и создания формы. |
| Project | fmProject | procedure TtbProject.bySaveClick(Sender: TObject); | Обработчик сохранения данных в таблице Fakultet |
| procedure TtbProject.byExitClick(Sender: TObject); | Обработчик собития осуществляет вызов сохранения данных в таблицах закрытия формы fmProject |
| procedure TtbProject.FormCreate(Sender: TObject); | Обработчик события отображения формы fmProject приложения программы. Процедура обработки и создания формы. |
| Rukovod | fmRukovod | procedure TtbRukovod.Button1Click(Sender: TObject); | Обработчик сохранения данных в таблице Rukovod |
| procedure TtbRukovod.Button2Click(Sender: TObject); | Обработчик собития осуществляет вызов сохранения данных в таблицах закрытия формы fmRukovod |
| procedure TtbRukovod.FormShow(Sender: TObject); | Обработчик события отображения формы fmRukovod приложения программы. Процедура обработки и создания формы. |
| GruppStudent | fmGruppStudent | procedure TGrupp\_Student.bySaveClick(Sender: TObject) | Обработчик сохранения данных в таблице GruppStudent |
| procedure TGrupp\_Student.byExitClick(Sender: TObject); | Обработчик собития осуществляет вызов сохранения данных в таблицах закрытия формы fmGruppStudent |

Продолжение таблицы 6.3

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
|  |  | procedure TGrupp\_Student.FormCreate(Sender: TObject); | Обработчик события отображения формы fmGruppStudent приложения программы. Процедура обработки и создания формы. |
| procedure TGrupp\_Student.DBLookupComboBox1Click(Sender: TObject); | Обработчик события отображения факультета из таблицы Fakultet. |
| procedure TGrupp\_Student.DBNavigator1Click(Sender: TObject; Button: TNavigateBtn); | Добавление или редактирование ззаписи в таблице GruppStudent. |

6.4 Используемые технические средства

Приложение предназначено для работы на персональных компьютерах, имеющих следующие функциональные характеристики:

- процессор с тактовой частотой 2.4 ГГц или более;

- оперативная память - 2 Гб или более;

- свободное место на жестком диске - 120 Мбайт.

В качестве сервера используется компьютер, имеющий следующие функциональные характеристики:

- процессор – 2.5 ГГц или более;

- оперативная память;

- 2 Гб или более;

- свободное место на жестком диске - 800 Мбайт.

6.5 Вызов и загрузка

Запуск приложения осуществляется запуском исполняемого файла KAFEDRA.exe, он расположен в папке E:\Kyrsovoy\.

Если приложение работает на локальном компьютере, то на нем должны быть размещены файл KAFEDRA.exe, файл с базой данных BASOV.lhb и СУБД Линтер. Приложение разработано для операционной системы Windows 10.

6.6 Входные данные

Входными данными является информация из базы данных и из запросов пользователя.

6.7 Выходные данные

Выходными данными является информация, сформированная по запросам пользователя из таблиц БД.

7 Программа и методика испытаний

7.1 Объект испытаний

Объектом испытаний является приложение KAFEDRA.exe, предназначенное для учета сведений кафедры института.

7.2 Цель испытаний

Испытания проводятся с целью проверки работоспособности и надежности приложения «KAFEDRA.exe». Для проверки правильности работы необходимо испытать его на тестовых примерах, сравнив полученные результаты можно определить правильность работы программы.

7.3 Требования к программе

Приложение «KAFEDRA.exe» должно выполнять все функции, указанные в техническом задании. Работа приложения не должна приводить к сбоям и ошибкам. Программа должна иметь дружественный для пользователя интерфейс.

7.4 Требования к программной документации

Программная документация к разработанному приложению должна содержать следующие разделы:

- введение;

- анализ предметной области;

- техническое задание;

- обоснования выбора средств для разработки;

- проектирование базы данных;

- проектирование приложения;

- описание программы;

- программа и методика испытаний;

- описание применения;

- заключение;

- список использованных источников;

- приложение А (SQL - скрипт базы данных);

- приложение Б (Результаты выполнения тестовых примеров);

- приложение В (Листинг программы);

- приложение Г (Текст dfm — файлов).

7.5 Средство и порядок испытаний

Во время испытаний приложения «Predpria.exe» необходимо использовать сервер — компьютер со следующими функциональными характеристиками: тактовая частота процессора — 2.4 ГГц или более, оперативная память — 2 Гб или более, свободное место на жестком диске — 800 Мбайт, на котором установлены СУБД Линтер и файл с базой данных; и рабочую станцию сети — компьютер со следующими функциональными характеристиками: тактовая частота процессора — 2.4 ГГц или более, оперативная память — 2 Гб или более, свободное место на жестком диске — 120 Мбайт, с установленной операционной системой Windows 7 и выше, на котором установлены файл с приложением Predpria.exe.

Для проверки правильности работы программы необходимо провести тестирование всех режимов ее работы, следует протестировать реакцию программы на возможные действия пользователя: запуск программы, выбор пунктов меню, нажатие кнопок, работа с базой данных. Приложение работает верно, если результаты его работы совпадают с результатами выполненных тестовых примеров.

7.6 Методика испытаний

Для проверки правильности работы программы необходимо запустить программу на выполнение, используя откомпилированный файл KAFEDRA.exe. В результате запуска должна открыться главная форма проекта. Затем необходимо проверить работоспособность программы на данных тестовых примерах. Для проверки работоспособности разработан следующие тестовые примеры:

Тестовый пример 1. Проверка выполнения функции учета информации о факультетах.

Для учета данных об факультетах необходимо проверить функцию добавления записей в таблицу «Факультет». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню пункт «Таблицы» главной формы программы, далее выбрать в выпадающем подменю «Факультеты» (Рисунок Б.2). В результате должна открыться форма «Факультеты» (Рисунок Б.3). На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на верхнем навигаторе, после чего ввести данные об отделе: Название - «ФВТ», ФИО — «Фионова В. В.», после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Факультеты» добавится новая запись (Рисунок Б.4). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения: Название - «ФВТ» на Название - «ФИТЕ», после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.5). Затем необходимо выбрать запись под номером «3» и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.6).

Тестовый пример 2. Проверка выполнения функции учета информации о проектах.

Для учета данных о товарах необходимо проверить функцию добавления записей в таблицу «Проекты».

Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Проекты» (Рисунок Б.2). В результате должна открыться форма «Проекты» (Рисунок Б.7). На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на навигаторе, после чего ввести данные о проекте: Название - «С++» после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Проекты» добавится новая запись (Рисунок Б.8). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения: Название - «С++» на название - «С#», после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.9). Затем необходимо выбрать товар под номером «3» и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.10).

Тестовый пример 3. Проверка выполнения функции учета информации о группах.

Для учета данных о накладных необходимо проверить функцию добавления записей в таблицу «Группы». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Группа-студенты» (Рисунок Б.2). В результате должна открыться форма «Группа-студенты» (Рисунок Б.11). На открывшейся форме необходимо кликнуть левой кнопкой мыши на выпадающий список с меткой «Выберите факультет», выбрать факультет – «ФВТ». Затем левой кнопкой мыши кликнуть на кнопку «Добавить», находящуюся на верхнем навигаторе, после чего ввести данные: Название — «20во1» после чего нажать на кнопку «Применить», находящуюся на вехнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Накладная» добавится новая запись (Рисунок Б.12). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся на навигаторе, после чего внести изменения: Название - «20во1» заменить на «20воа1», после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.13). Затем необходимо выбрать группу под номером «8» и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.14).

Тестовый пример 4. Проверка выполнения функции учета информации о студентах.

Для учета данных о строках накладной необходимо проверить функцию добавления записей в таблицу «Студент». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Группа-студенты» (Рисунок Б.2). В результате должна открыться форма «Группа-студенты» (Рисунок Б.11). На открывшейся форме необходимо кликнуть левой кнопкой мыши на выпадающий список с меткой «Выберите руководителя», выбрать руководителя – «Терякова Ю. В.». Затем выбрать товар с выпадающего списка с меткой «Выберите проект», выбрать «С++». Затем левой кнопкой мыши кликнуть на кнопку «Добавить», находящуюся на нижнем навигаторе, после чего ввести данные: ФИО — «Басов М. В.», Место предипломной практики — «ПГУ», после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Студент» добавится новая запись (Рисунок Б.15).

Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся на навигаторе, после чего внести изменения: Место предипломной практики — «ПГУ» заменить на «МГУ», после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.16).

Затем необходимо выбрать строку под номером «1» и нажать на кнопку «Удалить», находящуюся на нижнем навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.17).

Тестовый пример 5. Проверка выполнения функции учета информации об руководителе.

Для учета данных об руководителе необходимо проверить функцию добавления записей в таблицу «Руководители». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню пункт «Таблицы» главной формы программы, далее выбрать в выпадающем подменю «Руководители» (Рисунок Б.2). В результате должна открыться форма «Руководители» (Рисунок Б.18). На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на верхнем навигаторе, после чего ввести данные об отделе: ФИО - «Терякова Ю. В.», Должность — «ст. Преподователь» , Место работы — «ПГУ», Ученая звание — «ст. Преподователь», после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Руководители» добавится новая запись (Рисунок Б.19). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения: ФИО - «Терякова Ю. В.», Должность — «ст. Преподователь» , Место работы — «МГУ», Ученая звание — «ст. Преподователь», после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.20). Затем необходимо выбрать запись под номером «3» и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.21).

Тестовый пример 6. Проверка функции вывода информации об авторе.

Для проверки вывода информации об авторе необходимо кликнуть левой кнопкой мыши на пункт меню «О программе» на главной форме приложения (Рисунок Б.1). В результате на форме появится надпись о приложении и его разработчике (Рисунок Б.22).

Тестовый пример 7. Проверка выполнения функции правильности ввода данных в поля форм приложения.

Для проверки выполнения функции контроля правильности вводимых данных необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Факультеты» (Рисунок Б.2). На открывшейся форме необходимо заполнить поле ФИО – «12345678». Поле не будет заполнено, так как введенное значение не может использоваться для поля «ФИО» (Рисунок Б.23).

8 Описание применения

8.1 Назначение программы

Приложение предназначено для учета сведений отдела предприятия, а именно информацию об отделах, о накладных, о товарах.

8.2 Условия применения

Программа может использоваться как на локальных компьютерах, так и в сети. На локальном компьютере необходимо установить СУБД Линтер и разместить файл с базой данных, а также файл, содержащий приложение для работы с базой данных KAFEDRA.exe. При использовании на рабочей станции сети, на сервере базы данных необходимо установить СУБД Линтер и разместить файл с базой данных. На рабочей станции сети нужно разместить файл, содержащий клиент-серверное приложение для работы с базой данных KAFEDRA.exe.

8.3 Описание задач

Разработанное приложение предназначено для учета сведений кафедры. Программа является событийно — управляемой. Для начала работы необходимо запустить приложение на выполнение, используя откомпилированный файл KAFEDRA.exe.

Для учета данных об отделах необходимо проверить функцию добавления записей в таблицу «Факультеты». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню пункт «Таблицы» главной формы программы, далее выбрать в выпадающем подменю «Факультеты» (Рисунок Б.2). В результате должна открыться форма «Факультеты» (Рисунок Б.3).

На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на верхнем навигаторе, после чего ввести данные, после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Факультеты» добавится новая запись (Рисунок Б.4). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения и после нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.5). Затем необходимо выбрать клиента и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.6).

Для учета данных о проектах необходимо проверить функцию добавления записей в таблицу «Проекты».

Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Проекты» (Рисунок Б.2). В результате должна открыться форма «Проекты» (Рисунок Б.7). На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на навигаторе, после чего ввести данные о товаре, после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Проекты» добавится новая запись (Рисунок Б.8). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения и после нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.9). Затем необходимо выбрать тему проекта и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.10).

Для учета данных о группах необходимо проверить функцию добавления записей в таблицу «Группы».

Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Группа-студенты» (Рисунок Б.2). В результате должна открыться форма «Группа-студенты» (Рисунок Б.11). На открывшейся форме необходимо кликнуть левой кнопкой мыши на выпадающий список с меткой «Выберите факультет», выбрать факультет. Затем левой кнопкой мыши кликнуть на кнопку «Добавить», находящуюся на верхнем навигаторе, после чего ввести данные, после чего нажать на кнопку «Применить», находящуюся на вехнем навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Группы» добавится новая запись (Рисунок Б.12).

Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся на навигаторе, после чего внести изменения и после чего нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.13). Затем необходимо выбрать накладную и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.14).

Для учета данных о студентах необходимо проверить функцию добавления записей в таблицу «Студенты». Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Группа-студенты» (Рисунок Б.2).

В результате должна открыться форма «Группа-студенты» (Рисунок Б.11). На открывшейся форме необходимо кликнуть левой кнопкой мыши кликнуть на кнопку «Добавить», находящуюся на нижнем навигаторе, после чего ввести данные и после нажать на кнопку «Применить», находящуюся на верхнем навигаторе, а потом на кнопку «Сохранить».

В результате в таблицу «Студенты» добавится новая запись (Рисунок Б.15).

Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся на навигаторе, после чего внести изменения и нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.16).

Затем необходимо выбрать запист и нажать на кнопку «Удалить», находящуюся на нижнем навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.17).

Для учета данных о руководителях необходимо проверить функцию добавления записей в таблицу «Руководители».

Для этого необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Руководители» (Рисунок Б.2). В результате должна открыться форма «Руководители» (Рисунок Б.18). На открывшейся форме необходимо кликнуть левой кнопкой мыши на кнопку «Добавить», находящуюся на навигаторе, после чего ввести данные о руководителе, после чего нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». В результате в таблицу «Руководители» добавится новая запись (Рисунок Б.19). Затем необходимо выбрать добавленную запись и нажать на кнопку «Изменить», находящуюся в навигаторе, после чего внести изменения и после нажать на кнопку «Применить», находящуюся на навигаторе, а потом на кнопку «Сохранить». Изменения окажутся в таблице (Рисунок Б.20). Затем необходимо выбрать тему проекта и нажать на кнопку «Удалить», находящуюся на навигаторе, после чего подтвердить удаление, нажав на кнопку «ОК». В результате будет удалена выбранная запись (Рисунок Б.21).

Для вывода информации об авторе необходимо кликнуть левой кнопкой мыши на пункт меню «О программе» на главной форме приложения (Рисунок Б.1). В результате на форме появится надпись о приложении и его разработчике (Рисунок Б.22).

Для выполнения функции контроля правильности вводимых данных необходимо кликнуть левой кнопкой мыши на пункт меню «Таблицы» главной формы программы, далее выбрать в выпадающем подменю пункт «Факультет» (Рисунок Б.2). На открывшейся форме необходимо заполнить поле ФИО – цифрами. Поле не будет заполнено, так как введенное значение не может использоваться для поля «ФИО» (Рисунок Б.23).

Для выхода из программы необходимо нажать на пункт меню «Выход» на главной форме программы.

Заключение

В ходе выполнения курсового проекта была разработана автоматизированная система отдела предприятия. База данных была создана для СУБД Линтер с именем BASOV для пользователя с именем TEAM003 и паролем slave003. Проведено концептуальное, логическое и физическое проектирование базы данных. Построены концептуальная, логическая и физическая модели базы данных. Проектирование базы данных проведено с использованием CASEсредства Open ModelSphere 3.2 и инструментального средства СУБД Линтер «Рабочий стол Линтера». Получены навыки использования приложений «Администратор СУБД Линтер», «Рабочий стол Линтера», «Резервное копирование» для создания, удаления, регистрации, подключения, извлечения метаданных, резервного копирования и восстановления базы данных СУБД ЛИНТЕР. Были созданы следующие таблицы: FAKULTET – для хранения данных о факультете; RUKOVOD — для хранения данных о руководителях; STUDENT – для хранения данных о студенте; GRUPP – для хранения данных о группе; PROJECT – для хранения данных о проекте; SEQUENCES - для хранения значений первичных ключей таблиц. Были созданы ограничения целостности, и триггеры для таблиц FAKULTET, RUKOVOD, STUDENT, GRUPP, PROJECT. Проведено тестирование базы данных. Полученные результаты показали, что все требования технического задания выполнены в полном объеме, что свидетельствует о правильности работы базы данных, разработанной для автоматизированной системы «Учет сведения кафедры».

# Список используемых источников

1. Методология и технология проектирования АИС. [Электронный ресурс] - Режим доступа URL: <https://studfile.net/preview/7433236/page:7/> (дата обращения: 17.09.2024).

2. Проектирование АИС. [Лекции по Автоматизированным информационным системам и СУБД](https://studizba.com/lectures/10-informatika-i-programmirovanie/320-lekcii-po-avtomatizirovannym-informacionnym-sistemam-i-subd/). [Электронный ресурс] - Режим доступа URL: https://studizba.com/lectures/informatika-i-programmirovanie/lekcii-po-avtomatizirovannym-informacionnym-sistemam-i-subd/4275-proektirovanie-ais.html (дата обращения: 17.09.2024).

3. Виды АИС. Создание и использование. [Электронный ресурс] - Режим доступа URL: https://www.napishem.ru/spravochnik/informatika/setevye-informaczionnye-sistemy/avtomatizirovannye-informacionnye-sistemy-ais.html (дата обращения: 17.09.2024).

4. Российская СУБД ЛИНТЕР. [Электронный ресурс] - Режим доступа URL: <http://bourabai.bladeweb.org/dbt/servers/linter.htm> (дата обращения: 15.09.2024).

5. СУБД ЛИНТЕР® – сделано в России. [Электронный ресурс] - Режим доступа URL: <http://citforum.ru/products/relex/made_in_russia/> (дата обращения: 15.09.2024).

6. ATutor: Вопросы и Ответы – 2024. [Электронный ресурс] - Режим доступа URL: https://soware.ru/products/atutor (дата обращения: 25.10.2024).

7. Moodle: Описание, Функции и Интерфейс – 2024. [Электронный ресурс] - Режим доступа URL: <https://sow>are.ru/products/moodle (дата обращения: 20.10.2024).

8. PowerBuilder 5.0 - открытый инструментарий для создания сложных распределенных клиент-серверных приложений. [Электронный ресурс] - Режим доступа URL: <https://www.osp.ru/dbms/1996/05/13031520> (дата обращения: 25. 10.2024).

9. Использование case-средства open modelsphere для решения задач анализа и проектирования информационных систем. [Электронный ресурс] - Режим доступа URL: <https://www.agrorisk.ru/pub/202003/2> (дата обращения: 25. 10.2024).

10. Плюсы и минусы Oracle Database. [Электронный ресурс] - Режим доступа URL: <https://blog.uniwex.io/plyusy-i-minusy-oracle-database/> (дата обращения: 28. 10.2024).

11. Обзор СУБД ЛИНТЕР СТАНДАРТ. [Электронный ресурс] - Режим доступа URL: <https://dzen.ru/a/Y5xm-QhnDUh4CbvL> (дата обращения: 26. 10.2024).

12. Qt Creator - Лучшая кросс платформенная среда разработки. [Электронный ресурс] - Режим доступа URL: <https://dzen.ru/a/Y30vgUbkywd_ITQa> (дата обращения: 26. 10.2024).

13. Delphi. [Электронный ресурс] - Режим доступа URL: <https://blog.skillfactory.ru/glossary/delphi/> (дата обращения: 26. 10.2024).

Приложение А

(Обязательное)

SQL-скрипт базы данных

CREATE OR REPLACE TABLE STUDENT(

NOMST INTEGER NOT NULL PRIMARY KEY AUTOINC,

FIO VARCHAR(50) NULL,

MESTOPRDIPL VARCHAR(60) NULL,

NAME VARCHAR(30) GENERATED BY DEFAULT AS (

SUBSTRING(FIO FROM 1 FOR 5))

);

CREATE OR REPLACE TABLE GRUPP(

NOMGR INTEGER NOT NULL PRIMARY KEY AUTOINC,

NAZV VARCHAR(20) NULL

);

CREATE OR REPLACE TABLE PROJECT(

NOMPR INTEGER NOT NULL PRIMARY KEY AUTOINC,

TEMA VARCHAR(80) NULL

);

CREATE OR REPLACE TABLE RUKOVOD(

NOMRK INTEGER NOT NULL PRIMARY KEY AUTOINC,

FIO VARCHAR(50) NOT NULL,

YCHZV VARCHAR(50) NOT NULL,

MESTORAB VARCHAR(50) NOT NULL,

DOLJNOST VARCHAR(50) NOT NULL

);

CREATE OR REPLACE TABLE FAKULTET(

NOMFK INTEGER NOT NULL PRIMARY KEY AUTOINC,

NAZV VARCHAR(50) NOT NULL,

FIO VARCHAR(20) NOT NULL

);

CREATE INDEX index\_GRUPP\_NAZV ON GRUPP(NAZV);

CREATE INDEX index\_FAKULTET\_NAZV ON FAKULTET(NAZV);

CREATE INDEX index\_RUKOVOD\_FIO ON RUKOVOD(FIO);

CREATE INDEX index\_PROJECT\_TEMA ON PROJECT(TEMA);

CREATE INDEX index\_STUDENT\_FIO ON STUDENT(FIO);

CREATE OR REPLACE TABLE STUDENT(

NOMST INTEGER NOT NULL PRIMARY KEY AUTOINC,

FIO VARCHAR(50) NULL,

MESTOPRDIPL VARCHAR(60) NULL,

NOMGR int, foreign key(NOMGR) references GRUPP(NOMGR)

ON DELETE CASCADE

ON UPDATE CASCADE,

NOMRK int, foreign key(NOMRK) references RUKOVOD(NOMRK)

ON DELETE CASCADE

ON UPDATE CASCADE,

NOMRR int, foreign key(NOMRR) references PROJECT(NOMPR)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE OR REPLACE TABLE GRUPP(

NOMGR INTEGER NOT NULL PRIMARY KEY AUTOINC,

NAZV VARCHAR(20) NULL,

NOMFK int, foreign key(NOMFK) references FAKULTET(NOMFK)

ON DELETE CASCADE

ON UPDATE CASCADE

);

ALTER TABLE STUDENT ADD UNIQUE(FIO);

ALTER TABLE GRUPP ALTER NAZV DISABLE NULL;

ALTER TABLE STUDENT ADD UNIQUE(FIO);

ALTER TABLE STUDENT ALTER COLUMN FIO

ADD CHECK (FIO NOT LIKE '%[0-9]%');

CREATE OR REPLACE TABLE SEQUENCES(

TAB\_NAME VARCHAR(64) PRIMARY KEY NOT NULL,

CUR\_VALUE INT NOT NULL

);

CREATE PROCEDURE GENERATEPK(IN TABNAME CHAR(64)) RESULT INT FOR DEBUG

DECLARE

VAR V CURSOR(VALUE INT);

CODE

EXECUTE DIRECT "LOCK TABLE SEQUENCES WAIT;";

EXECUTE DIRECT "UPDATE SEQUENCES SET COR\_VALUE = CUR\_VALUE + 1 WHERE TAB\_NAME =" ' + TABNAME+ ' ";";

IF ROWCOUNT() > O THEN

OPEN V FOR DIRECT "SELECT CUR\_VALUE FROM SEQUENCES WHERE TAB\_NAME =" ' + TABNAME+ ' ";

ELSE.

EXECUTE DIRECT "INSERT INTO SEQUENCES VALUES("'+ TABNAME +'"1);";

V.VALUE := 1;

ENDIF

EXECUTE DIRECT "UNLOCK TABLE SEQUENCES;";

COMMIT;

RETURN V.VALUE;

END;

CREATE TRIGGER INS\_STUDENT BEFORE INSERT ON STUDENT

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL GENERATEPK("STUDENT") INTO NEW.NOMST;

END;

CREATE TRIGGER INS\_GRUPP BEFORE INSERT ON GRUPP

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL GENERATEPK("GRUPP") INTO NEW.NOMGR;

END;

CREATE TRIGGER INS\_PROJECT BEFORE INSERT ON PROJECT

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL GENERATEPK("PROJECT") INTO NEW.NOMPR;

END;

CREATE TRIGGER INS\_RUKOVOD BEFORE INSERT ON RUKOVOD

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL GENERATEPK("RUKOVOD") INTO NEW.NOMRK;

END;

CREATE TRIGGER INS\_FAKULTET BEFORE INSERT ON FAKULTET

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL GENERATEPK("FAKULTET") INTO NEW.NOMFK;

END;

CREATE OR REPLACE TABLE STUDENT\_HISTORY(

ID INT,

DT DATE,

VALUE VARCHAR(40),

STATUS CHAR(1)

);

ALTER TABLE STUDENT\_HISTORY ADD PRIMARY KEY(ID, DT);

CREATE TRIGGER INS\_STUDENT\_HIST AFTER INSERT ON STUDENT

FOR EACH ROW EXECUTE FOR DEBUG

CODE

EXECUTE DIRECT "INSERT INTO STUDENT\_HISTORY(NOMST,DT,FIO,STATUS) VALUES (" +ITOA(NEW.NOMST) + ", SYSDATE, ' "+ NEW.FIO +" ', '|');";

END;

INSERT INTO STUDENT(FIO, MESTOPRDIPL, NAME, NOMGR, NOMRK, NOMPR) VALUES ('Карташов И. В.','ПГУ','Илья',4,2,5);

CREATE OR REPLACE PROCEDURE CHECK\_NAZV(IN NAZV VARCHAR(50)) RESULT VARCHAR(50) FOR DEBUG

CODE

IF (LENGTH(NAZV) = 0) THEN RETURN "Не указана";

else RETURN NAZV;

ENDIF;

END;

CREATE TRIGGER INS\_CHECK\_NAZV BEFORE INSERT ON GRUPP

FOR EACH ROW EXECUTE FOR DEBUG

CODE

CALL CHECK\_NAZV (NEW.NAZV) INTO NEW.NAZV;

END;

CREATE OR REPLACE PROCEDURE CHECK\_MESTOPRDIPL(IN MESTOPRDIPL VARCHAR(50)) RESULT VARCHAR(50) FOR DEBUG

CODE

IF (LENGTH(MESTOPRDIPL) = 0) THEN RETURN "Кафедра института";

else RETURN MESTOPRDIPL;

ENDIF;

END;

SELECT \* FROM STUDENT, GRUPP WHERE STUDENT.NOMGR = GRUPP.NOMGR;

SELECT \* FROM STUDENT WHERE NOMGR = :NOMGR;

Приложение Б

(Обязательное)

Результаты выполнения тестовых примеров

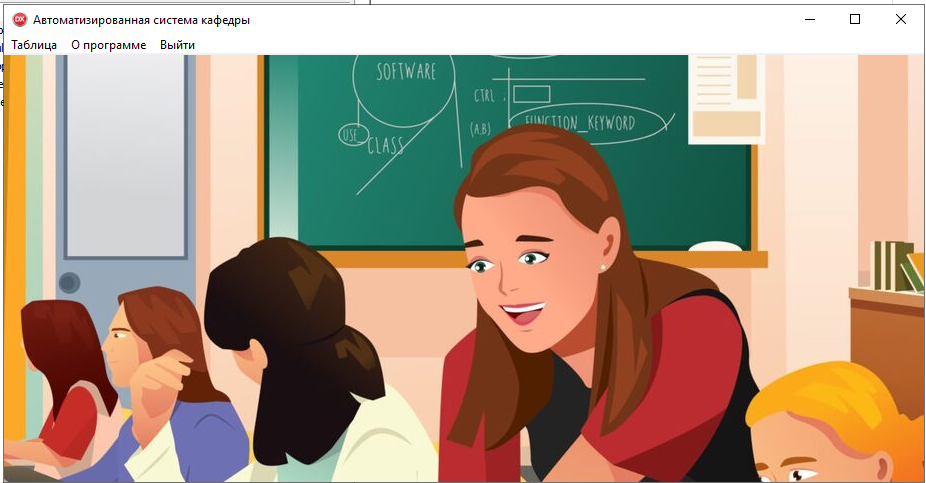


Рисунок Б.1 – Главная форма программы

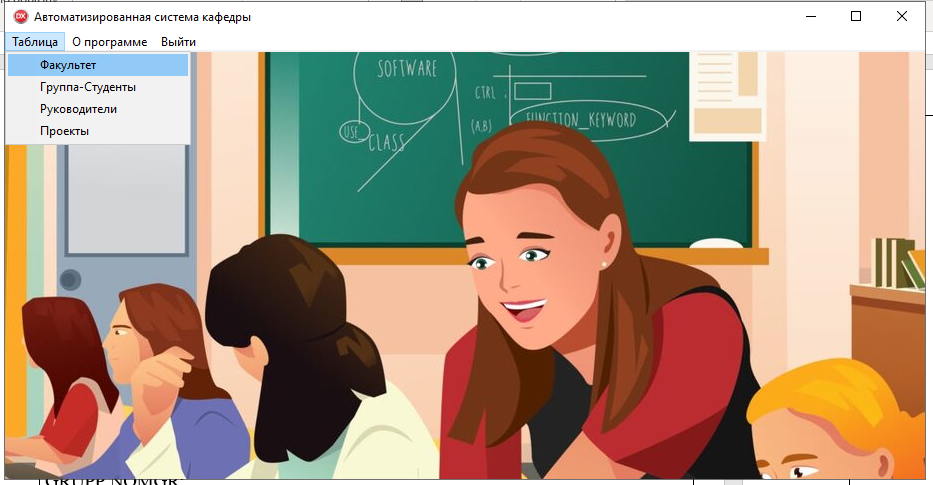


Рисунок Б.2 – Выбор пункта подменю работы с таблицами

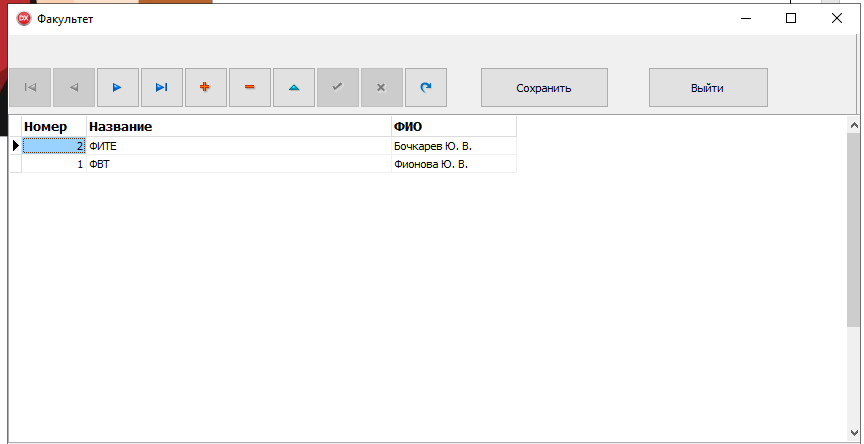


Рисунок Б.3 – Форма «Факультет»

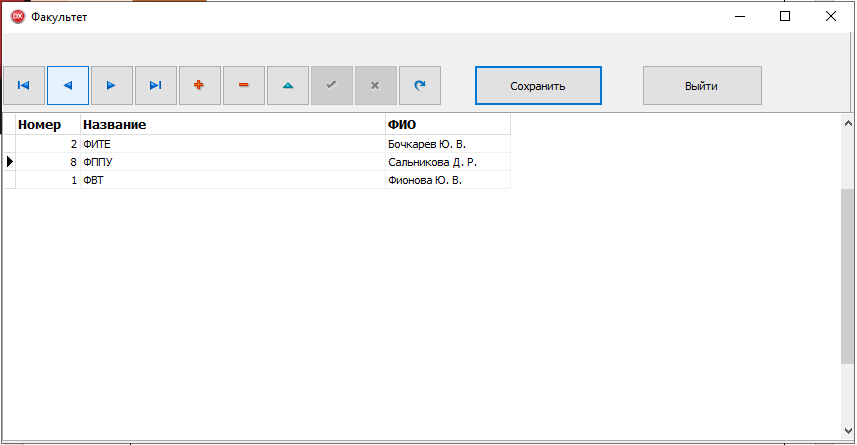


Рисунок Б.4 – Добавление записи в таблицу «Факультет»

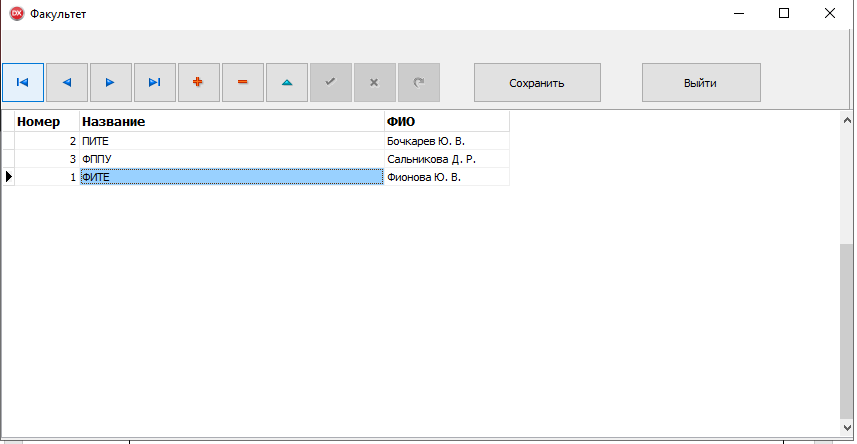


Рисунок Б.5 – Редактирование записи в таблицу «Факультет»

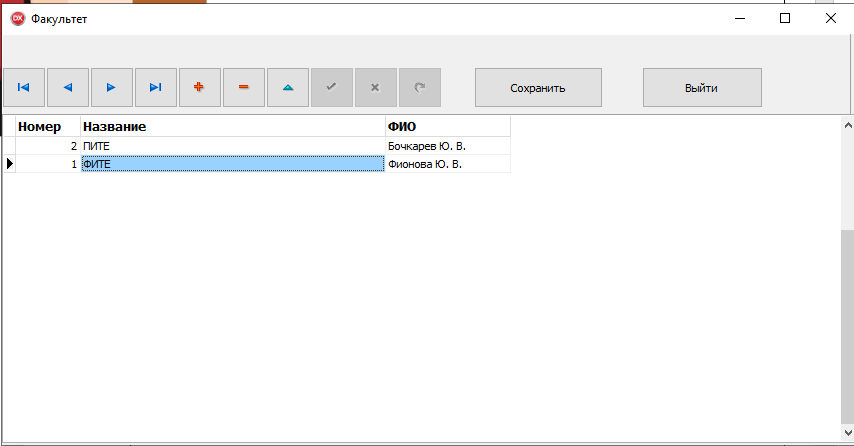


Рисунок Б.6 – Удаление записи в таблицу «Факультет»

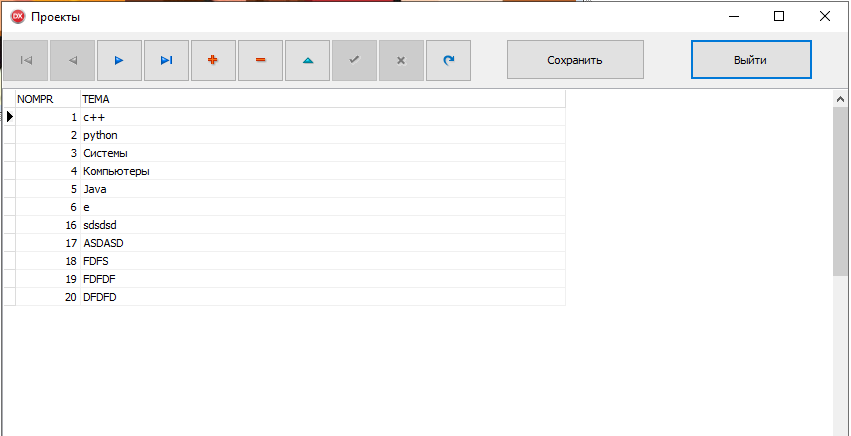


Рисунок Б.7 – Форма «Проекты»

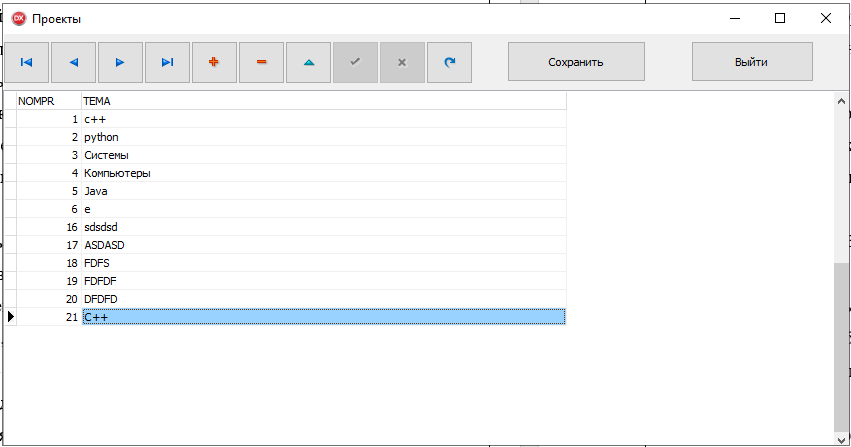


Рисунок Б.8 – Добавление записи в таблицу «Проекты»

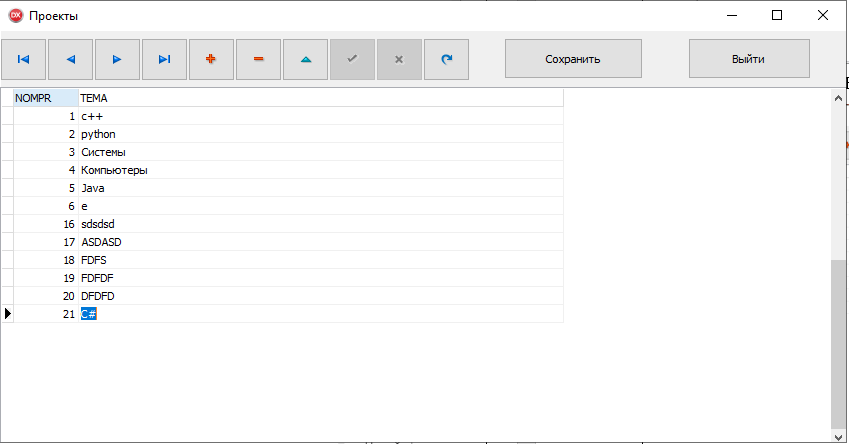


Рисунок Б.9 – Редактирование записи в таблицу «Проекты»

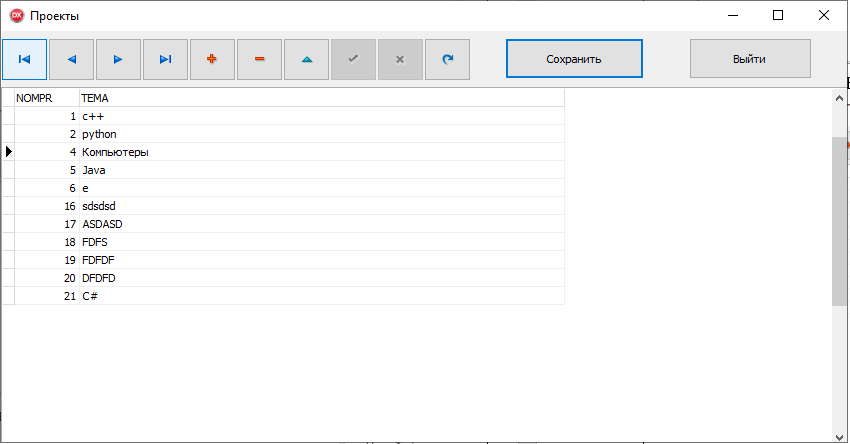


Рисунок Б.10 – Удаление записи в таблицу «Факультет»

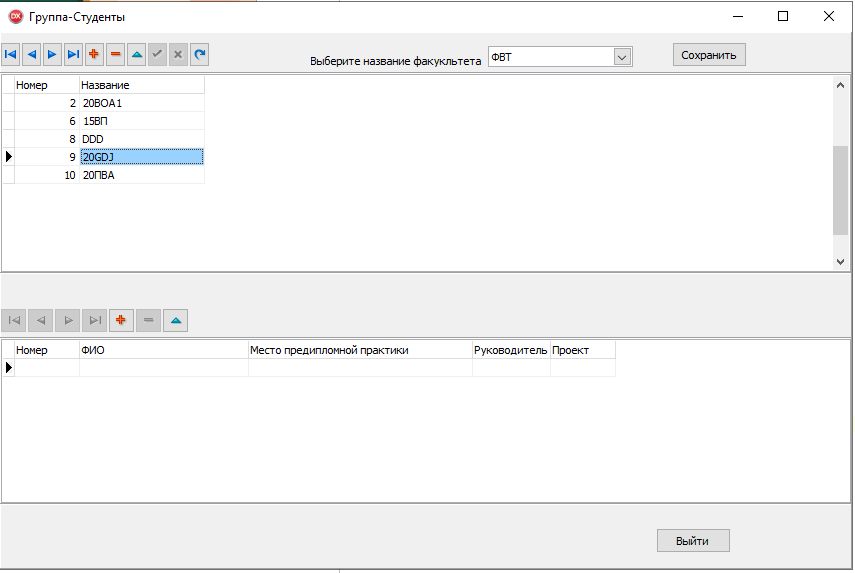


Рисунок Б.11 – Форма «Группа-Студенты»

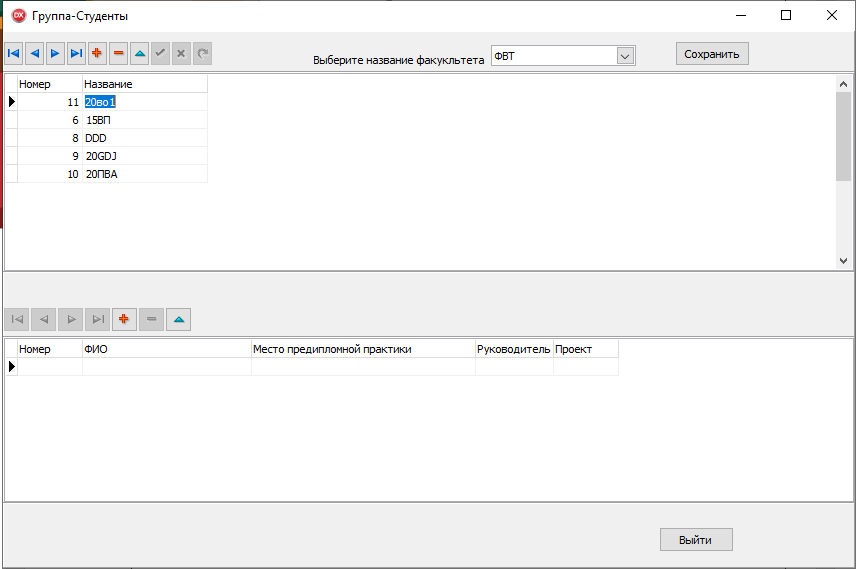


Рисунок Б.12 – Добавление записи в таблицу «Группа»

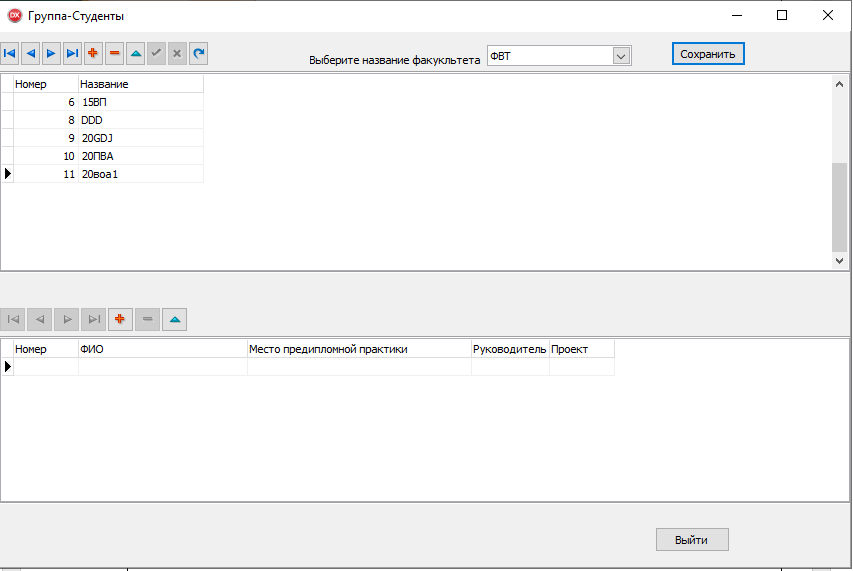


Рисунок Б.13 – Редактирование записи в таблицу «Группа»

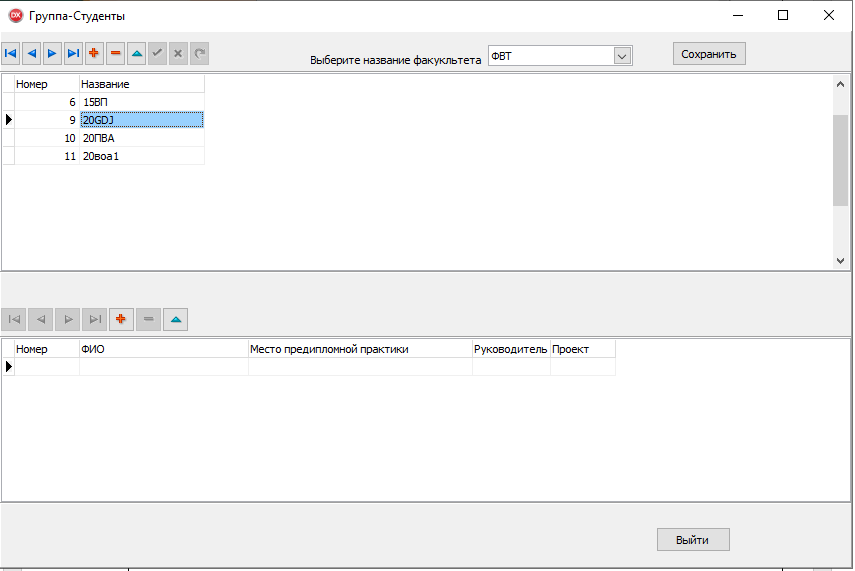


Рисунок Б.14 – Удаление записи в таблицу «Группа»

Рисунок Б.15 – Добавление записи в таблицу «Студенты»

Рисунок Б.16 – Редактирование записи в таблицу «Студенты»

Рисунок Б.17 – Удаление записи в таблицу «Студенты»

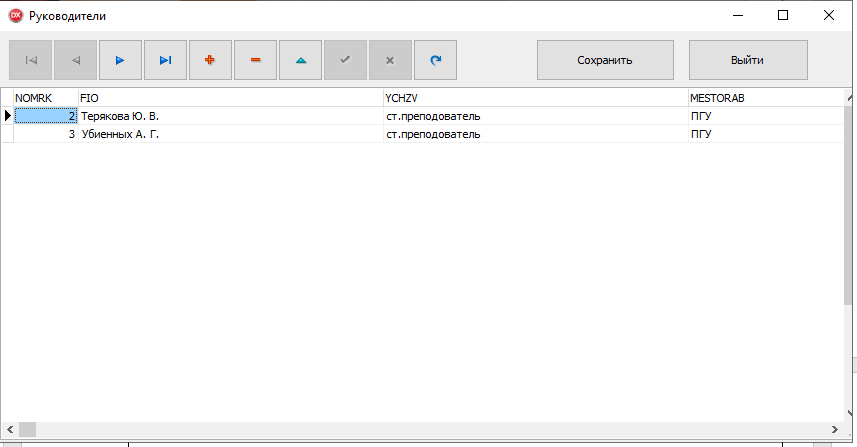


Рисунок Б.18 – Форма «Руководители»

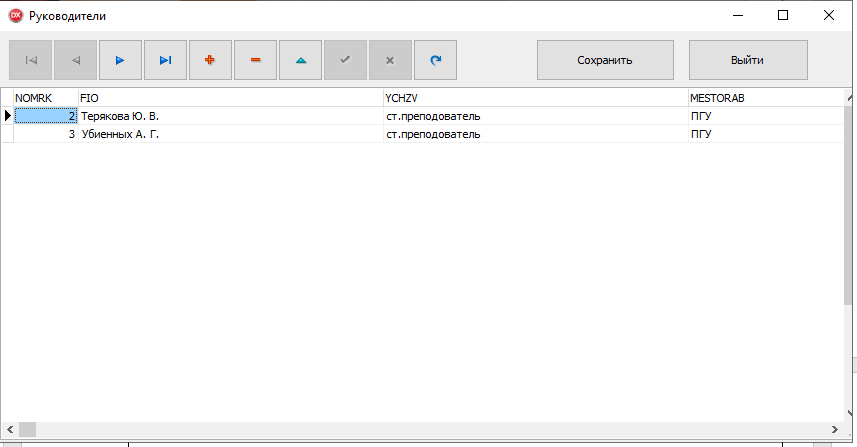


Рисунок Б.19 – Добавление записи в таблицу «Руководители»

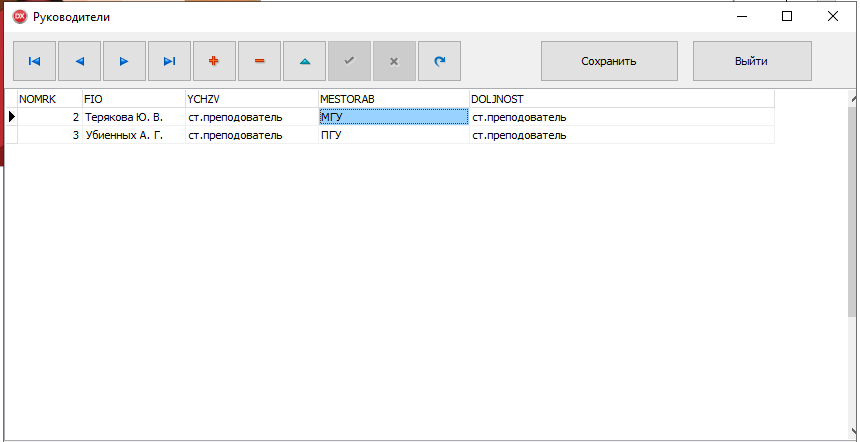


Рисунок Б.20 – Редактирование записи в таблицу «Руководители»

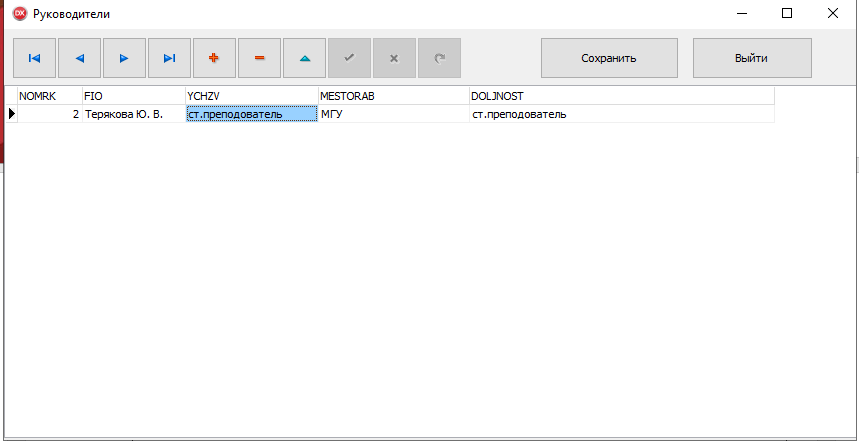


Рисунок Б.21 – Удаление записи в таблицу «Руководители»

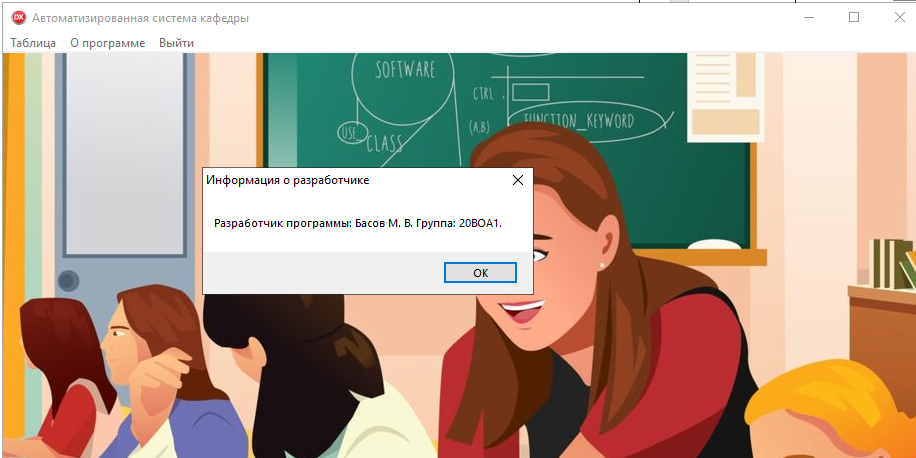


Рисунок Б.22 – Вывод информации о разработчике

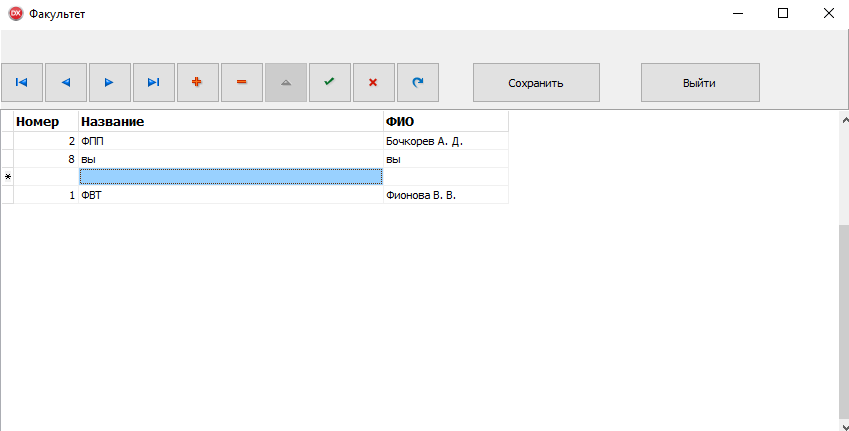


Рисунок Б.23 – Контроль рпавильности ввода

Приложение В

(Обязательное)

Листинг программы

unit linMain;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.Imaging.jpeg,

Vcl.ExtCtrls;

type

TmainForm = class(TForm)

MainMenu1: TMainMenu;

Image1: TImage;

N1: TMenuItem;

About: TMenuItem;

byExit: TMenuItem;

fakultet: TMenuItem;

gruppstudent: TMenuItem;

rukovod: TMenuItem;

project: TMenuItem;

procedure gruppstudentClick(Sender: TObject);

procedure AboutClick(Sender: TObject);

procedure byExitClick(Sender: TObject);

procedure fakultetClick(Sender: TObject);

procedure rukovodClick(Sender: TObject);

procedure projectClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

mainForm: TmainForm;

implementation

{$R \*.dfm}

uses gruppstudent, fakultet, project, rukovod, fmDM;

procedure TmainForm.AboutClick(Sender: TObject);

begin

Application.MessageBox('Разработчик программы: Басов М. В. Группа: 20ВОА1.', 'Информация о разработчике');

end;

procedure TmainForm.byExitClick(Sender: TObject);

begin

Close();

end;

procedure TmainForm.fakultetClick(Sender: TObject);

begin

tbFakultet.Show();

end;

procedure TmainForm.gruppstudentClick(Sender: TObject);

begin

Grupp\_Student.Show();

end;

procedure TmainForm.projectClick(Sender: TObject);

begin

tbProject.Show();

end;

procedure TmainForm.rukovodClick(Sender: TObject);

begin

tbRukovod.Show();

end;

end.

DM

unit fmDM;

interface

uses

System.SysUtils, System.Classes, Data.DBXOdbc, Data.DB, Data.SqlExpr,

Data.FMTBcd, RecError, Datasnap.DBClient, Datasnap.Provider, DBXDynalink;

type

TDM = class(TDataModule)

Connection: TSQLConnection;

FakultetQuery: TSQLQuery;

dsFacultet: TDataSource;

Fakultet\_DSP: TDataSetProvider;

FakultetCDS: TClientDataSet;

dsFakultet2: TDataSource;

dsGrupp2: TDataSource;

dsStudent2: TDataSource;

GruppCDS: TClientDataSet;

StudentCDS: TClientDataSet;

Grupp\_DSP: TDataSetProvider;

dsGrupp: TDataSource;

GruppQuery: TSQLQuery;

StudentQuery: TSQLQuery;

FakultetQueryNOMFK: TIntegerField;

FakultetQueryNAZV: TStringField;

FakultetQueryFIO: TStringField;

GruppQueryNOMGR: TIntegerField;

GruppQueryNAZV: TStringField;

GruppQueryNOMFK: TIntegerField;

StudentQueryNOMST: TIntegerField;

StudentQueryFIO: TStringField;

StudentQueryMESTOPRDIPL: TStringField;

StudentQueryNOMGR: TIntegerField;

StudentQueryNOMRK: TIntegerField;

StudentQueryNOMRR: TIntegerField;

FakultetCDSNOMFK: TIntegerField;

FakultetCDSNAZV: TStringField;

FakultetCDSFIO: TStringField;

FakultetCDSGruppQuery: TDataSetField;

GruppCDSNOMGR: TIntegerField;

GruppCDSNAZV: TStringField;

GruppCDSNOMFK: TIntegerField;

GruppCDSStudentQuery: TDataSetField;

StudentCDSNOMST: TIntegerField;

StudentCDSFIO: TStringField;

StudentCDSMESTOPRDIPL: TStringField;

StudentCDSNOMGR: TIntegerField;

StudentCDSNOMRK: TIntegerField;

StudentCDSNOMRR: TIntegerField;

ProjectQuery: TSQLQuery;

dtProject: TDataSource;

Project\_DSP: TDataSetProvider;

ProjectCDS: TClientDataSet;

dsProject2: TDataSource;

ProjectQueryNOMPR: TIntegerField;

ProjectQueryTEMA: TStringField;

ProjectCDSNOMPR: TIntegerField;

ProjectCDSTEMA: TStringField;

procedure FakultetCDSReconcileError(DataSet: TCustomClientDataSet;

E: EReconcileError; UpdateKind: TUpdateKind;

var Action: TReconcileAction);

private

{ Private declarations }

public

{ Public declarations }

end;

var

DM: TDM;

implementation

{%CLASSGROUP 'Vcl.Controls.TControl'}

{$R \*.dfm}

procedure TDM.FakultetCDSReconcileError(DataSet: TCustomClientDataSet;

E: EReconcileError; UpdateKind: TUpdateKind; var Action: TReconcileAction);

begin

Action := HandleReconcileError(DataSet, UpdateKind, E);

end;

end.

TABLE FAKULTET

unit fakultet;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Vcl.StdCtrls, Vcl.ExtCtrls,

Vcl.DBCtrls, Vcl.Grids, Vcl.DBGrids, fmDM;

type

TtbFakultet = class(TForm)

DBGrid1: TDBGrid;

bySave: TButton;

byExit: TButton;

Panel1: TPanel;

DBNavigator1: TDBNavigator;

procedure bySaveClick(Sender: TObject);

procedure byExitClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

public

{ Public declarations }

end;

var

tbFakultet: TtbFakultet;

implementation

{$R \*.dfm}

procedure TtbFakultet.bySaveClick(Sender: TObject);

begin

with DM do

begin

if FakultetCDS.ChangeCount > 0 then

begin

FakultetCDS.ApplyUpdates(0);

FakultetCDS.Refresh;

end; //ir

end; //with

DBGrid1.Refresh;

end;

procedure TtbFakultet.byExitClick(Sender: TObject);

begin

bySaveClick(Sender);

Close;

end;

procedure TtbFakultet.FormCreate(Sender: TObject);

begin

DM.FakultetCDS.Open;

end;

procedure TtbFakultet.FormKeyPress(Sender: TObject; var Key: Char);

begin

if not ((Key = #8) or (ord(Key) >= 1040) and (ord(Key) <= 1103)) then

Key:=#0;

end;

end.

TABLE RUKOVOD

unit rukovod;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Vcl.StdCtrls, Vcl.ExtCtrls,

Vcl.DBCtrls, Vcl.Grids, Vcl.DBGrids;

type

TtbRukovod = class(TForm)

DBGrid1: TDBGrid;

DBNavigator1: TDBNavigator;

Button1: TButton;

Button2: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormShow(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

tbRukovod: TtbRukovod;

implementation

{$R \*.dfm}

uses fmDM;

procedure TtbRukovod.Button1Click(Sender: TObject);

begin

with DM do

begin

if RukovodCDS.ChangeCount > 0 then

begin

RukovodCDS.ApplyUpdates(0);

RukovodCDS.Refresh;

end; //ir

end; //with

DBGrid1.Refresh;

end;

procedure TtbRukovod.Button2Click(Sender: TObject);

begin

Button1Click(Sender);

Close;

end;

procedure TtbRukovod.FormShow(Sender: TObject);

begin

DM.RukovodCDS.Open;

end;

end.

TABLE PROJECT

unit project;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Vcl.ExtCtrls, Vcl.DBCtrls,

Vcl.Grids, Vcl.DBGrids, Vcl.StdCtrls, fmDm;

type

TtbProject = class(TForm)

bySave: TButton;

byExit: TButton;

Panel1: TPanel;

DBGrid1: TDBGrid;

DBNavigator1: TDBNavigator;

procedure bySaveClick(Sender: TObject);

procedure byExitClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

tbProject: TtbProject;

implementation

{$R \*.dfm}

procedure TtbProject.bySaveClick(Sender: TObject);

begin

with DM do

begin

if ProjectCDS.ChangeCount > 0 then

begin

ProjectCDS.ApplyUpdates(0);

ProjectCDS.Refresh;

end; //ir

end; //with

DBGrid1.Refresh;

end;

procedure TtbProject.byExitClick(Sender: TObject);

begin

bySaveClick(Sender);

Close;

end;

procedure TtbProject.FormCreate(Sender: TObject);

begin

DM.ProjectCDS.Open;

//DM.ProjectCDS.FieldDefs.Add('NOMPR', ftInteger, 0, True); // True означает, что это ключ

end;

end.

TABLE GRYPPSTUDENT

unit gruppstudent;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, fmDM, Data.DB, Vcl.DBCtrls,

Vcl.Grids, Vcl.DBGrids, Vcl.StdCtrls;

type

TGrupp\_Student = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

Panel3: TPanel;

Panel4: TPanel;

Panel5: TPanel;

Splitter1: TSplitter;

DBGrid1: TDBGrid;

DBGrid2: TDBGrid;

DBNavigator1: TDBNavigator;

DBNavigator2: TDBNavigator;

DBLookupComboBox1: TDBLookupComboBox;

byExit: TButton;

Label1: TLabel;

bySave: TButton;

procedure bySaveClick(Sender: TObject);

procedure byExitClick(Sender: TObject);

procedure DBLookupComboBox1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Grupp\_Student: TGrupp\_Student;

implementation

{$R \*.dfm}

procedure TGrupp\_Student.byExitClick(Sender: TObject);

begin

bySaveClick(Sender);

Close;

end;

procedure TGrupp\_Student.bySaveClick(Sender: TObject);

begin

with DM do

begin

if GruppCDS.ChangeCount > 0 then

begin

GruppCDS.ApplyUpdates(0);

GruppCDS.Refresh;

end;

end;

DBGrid1.Refresh;

DBGrid2.Refresh;

end;

procedure TGrupp\_Student.DBLookupComboBox1Click(Sender: TObject);

begin

with DM.GruppCDS DO

begin

Close;

Params.ParamByName('NOMFK').Value:=DBLookupComboBox1.KeyValue;

Open;

end;

end;

procedure TGrupp\_Student.DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);

begin

if (Button = nbPost) then

begin

DBGrid1.DataSource.DataSet.Edit;

DBGrid1.DataSource.DataSet.FieldByName('NOMFK').AsInteger:=dblookupComboBox1.KeyValue;

DBGrid1.DataSource.DataSet.Post;

end;

end;

procedure TGrupp\_Student.FormCreate(Sender: TObject);

begin

DM.FakultetCDS.Open;

end;

end.

Приложение Г

(обязательное)

Текс dfm-файлов

object DM: TDM

OldCreateOrder = False

Height = 364

Width = 525

object Connection: TSQLConnection

ConnectionName = 'OdbcConnection'

DriverName = 'Odbc'

LoginPrompt = False

Params.Strings = (

'DriverName=Odbc'

'DataBase=Linter 6.0'

'User\_Name=TEAM003'

'Password=slave003'

'IsolationLevel=ReadCommitted'

'RowSetSize=20'

'ConnectionString=')

Connected = True

Left = 48

Top = 56

end

object FakultetQuery: TSQLQuery

MaxBlobSize = -1

Params = <

item

DataType = ftInteger

Name = 'NOMFK'

ParamType = ptInput

Value = 0

end>

SQL.Strings = (

'select \* from FAKULTET order by FIO')

SQLConnection = Connection

Left = 120

Top = 56

object FakultetQueryNOMFK: TIntegerField

FieldName = 'NOMFK'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

end

object FakultetQueryNAZV: TStringField

FieldName = 'NAZV'

Size = 50

end

object FakultetQueryFIO: TStringField

FieldName = 'FIO'

end

end

object dsFacultet: TDataSource

DataSet = FakultetQuery

Left = 200

Top = 56

end

object Fakultet\_DSP: TDataSetProvider

DataSet = FakultetQuery

Options = [poCascadeDeletes, poCascadeUpdates, poUseQuoteChar]

Left = 280

Top = 56

end

object FakultetCDS: TClientDataSet

Active = True

Aggregates = <>

Params = <

item

DataType = ftInteger

Name = 'NOMFK'

ParamType = ptInput

Value = 0

end>

ProviderName = 'Fakultet\_DSP'

OnReconcileError = FakultetCDSReconcileError

Left = 368

Top = 56

object FakultetCDSNOMFK: TIntegerField

FieldName = 'NOMFK'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

end

object FakultetCDSNAZV: TStringField

FieldName = 'NAZV'

Size = 50

end

object FakultetCDSFIO: TStringField

FieldName = 'FIO'

end

object FakultetCDSGruppQuery: TDataSetField

FieldName = 'GruppQuery'

Visible = False

end

end

object dsFakultet2: TDataSource

DataSet = FakultetCDS

Left = 448

Top = 56

end

object dsGrupp2: TDataSource

DataSet = GruppCDS

Left = 448

Top = 120

end

object dsStudent2: TDataSource

DataSet = StudentCDS

Left = 448

Top = 176

end

object GruppCDS: TClientDataSet

Active = True

Aggregates = <>

Params = <

item

DataType = ftInteger

Name = 'NOMFK'

ParamType = ptInput

Value = 0

end

item

DataType = ftInteger

Name = 'NOMGR'

ParamType = ptInput

Value = 0

end>

ProviderName = 'Grupp\_DSP'

Left = 368

Top = 120

object GruppCDSNOMGR: TIntegerField

FieldName = 'NOMGR'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object GruppCDSNAZV: TStringField

FieldName = 'NAZV'

Required = True

end

object GruppCDSNOMFK: TIntegerField

FieldName = 'NOMFK'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object GruppCDSStudentQuery: TDataSetField

FieldName = 'StudentQuery'

Visible = False

end

end

object StudentCDS: TClientDataSet

Active = True

Aggregates = <>

DataSetField = GruppCDSStudentQuery

Params = <

item

DataType = ftInteger

Name = 'NOMST'

ParamType = ptUnknown

Value = 0

end>

Left = 368

Top = 176

object StudentCDSNOMST: TIntegerField

FieldName = 'NOMST'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object StudentCDSFIO: TStringField

FieldName = 'FIO'

Required = True

Size = 50

end

object StudentCDSMESTOPRDIPL: TStringField

FieldName = 'MESTOPRDIPL'

Required = True

Size = 60

end

object StudentCDSNOMGR: TIntegerField

FieldName = 'NOMGR'

Required = True

end

object StudentCDSNOMRK: TIntegerField

FieldName = 'NOMRK'

Required = True

end

object StudentCDSNOMRR: TIntegerField

FieldName = 'NOMRR'

Required = True

end

end

object Grupp\_DSP: TDataSetProvider

DataSet = GruppQuery

Options = [poCascadeDeletes, poCascadeUpdates, poUseQuoteChar]

Left = 280

Top = 120

end

object dsGrupp: TDataSource

DataSet = GruppQuery

Left = 200

Top = 120

end

object GruppQuery: TSQLQuery

DataSource = dsFacultet

MaxBlobSize = -1

Params = <

item

DataType = ftInteger

Name = 'NOMFK'

ParamType = ptInput

Value = 0

end

item

DataType = ftInteger

Name = 'NOMGR'

ParamType = ptInput

Value = 0

end>

SQL.Strings = (

'SELECT \* FROM GRUPP '

'where NOMFK=:NOMFK')

SQLConnection = Connection

Left = 120

Top = 120

object GruppQueryNOMGR: TIntegerField

FieldName = 'NOMGR'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object GruppQueryNAZV: TStringField

FieldName = 'NAZV'

Required = True

end

object GruppQueryNOMFK: TIntegerField

FieldName = 'NOMFK'

Required = True

end

end

object StudentQuery: TSQLQuery

DataSource = dsGrupp

MaxBlobSize = -1

Params = <

item

DataType = ftInteger

Name = 'NOMGR'

ParamType = ptInput

end>

SQL.Strings = (

'SELECT \* FROM STUDENT where ((NOMGR=:NOMGR))')

SQLConnection = Connection

Left = 120

Top = 184

object StudentQueryNOMST: TIntegerField

FieldName = 'NOMST'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object StudentQueryFIO: TStringField

FieldName = 'FIO'

Required = True

Size = 50

end

object StudentQueryMESTOPRDIPL: TStringField

FieldName = 'MESTOPRDIPL'

Required = True

Size = 60

end

object StudentQueryNOMGR: TIntegerField

FieldName = 'NOMGR'

Required = True

end

object StudentQueryNOMRK: TIntegerField

FieldName = 'NOMRK'

Required = True

end

object StudentQueryNOMRR: TIntegerField

FieldName = 'NOMRR'

Required = True

end

end

object ProjectQuery: TSQLQuery

MaxBlobSize = -1

Params = <

item

DataType = ftInteger

Name = 'NOMPR'

ParamType = ptInput

Value = 0

end>

SQL.Strings = (

'SELECT \* FROM PROJECT')

SQLConnection = Connection

Left = 120

Top = 248

object ProjectQueryNOMPR: TIntegerField

FieldName = 'NOMPR'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object ProjectQueryTEMA: TStringField

FieldName = 'TEMA'

Required = True

Size = 80

end

end

object dtProject: TDataSource

DataSet = ProjectQuery

Left = 192

Top = 248

end

object Project\_DSP: TDataSetProvider

DataSet = ProjectQuery

Options = [poCascadeDeletes, poCascadeUpdates, poUseQuoteChar]

Left = 280

Top = 248

end

object ProjectCDS: TClientDataSet

Active = True

Aggregates = <>

Params = <

item

DataType = ftInteger

Name = 'NOMPR'

ParamType = ptInput

Value = 0

end>

ProviderName = 'Project\_DSP'

Left = 368

Top = 248

object ProjectCDSNOMPR: TIntegerField

FieldName = 'NOMPR'

ProviderFlags = [pfInUpdate, pfInWhere, pfInKey]

Required = True

end

object ProjectCDSTEMA: TStringField

FieldName = 'TEMA'

Required = True

Size = 80

end

end

object dsProject2: TDataSource

DataSet = ProjectCDS

Left = 448

Top = 248

end

end

object mainForm: TmainForm

Left = 0

Top = 0

Caption = #1040#1074#1090#1086#1084#1072#1090#1080#1079#1080#1088#1086#1074#1072#1085#1085#1072#1103' '#1089#1080#1089#1090#1077#1084#1072' '#1082#1072#1092#1077#1076#1088#1099

ClientHeight = 407

ClientWidth = 920

Color = clBtnFace

Font.Charset = DEFAULT\_CHARSET

Font.Color = clWindowText

Font.Height = -11

Font.Name = 'Tahoma'

Font.Style = []

Menu = MainMenu1

OldCreateOrder = False

PixelsPerInch = 96

TextHeight = 13

object Image1: TImage

Left = -224

Top = -168

Width = 1680

Height = 1119

AutoSize = True

ParentShowHint = False

Picture.Data = {

ShowHint = False

end

object MainMenu1: TMainMenu

Left = 568

Top = 8

object N1: TMenuItem

Caption = #1058#1072#1073#1083#1080#1094#1072

object fakultet: TMenuItem

Caption = #1060#1072#1082#1091#1083#1100#1090#1077#1090

OnClick = fakultetClick

end

object gruppstudent: TMenuItem

Caption = #1043#1088#1091#1087#1087#1072'-'#1057#1090#1091#1076#1077#1085#1090#1099

OnClick = gruppstudentClick

end

object rukovod: TMenuItem

Caption = #1056#1091#1082#1086#1074#1086#1076#1080#1090#1077#1083#1080

OnClick = rukovodClick

end

object project: TMenuItem

Caption = #1055#1088#1086#1077#1082#1090#1099

OnClick = projectClick

end

end

object About: TMenuItem

Caption = #1054' '#1087#1088#1086#1075#1088#1072#1084#1084#1077

OnClick = AboutClick

end

object byExit: TMenuItem

Caption = #1042#1099#1081#1090#1080

OnClick = byExitClick

end

end

end

object tbFakultet: TtbFakultet

Left = 0

Top = 0

Caption = #1060#1072#1082#1091#1083#1100#1090#1077#1090

ClientHeight = 411

ClientWidth = 852

Color = clBtnFace

Font.Charset = DEFAULT\_CHARSET

Font.Color = clWindowText

Font.Height = -11

Font.Name = 'Tahoma'

Font.Style = []

OldCreateOrder = False

Position = poScreenCenter

OnCreate = FormCreate

OnKeyPress = FormKeyPress

PixelsPerInch = 96

TextHeight = 13

object DBGrid1: TDBGrid

Left = 0

Top = 80

Width = 857

Height = 329

DataSource = DM.dsFakultet2

TabOrder = 0

TitleFont.Charset = DEFAULT\_CHARSET

TitleFont.Color = clWindowText

TitleFont.Height = -11

TitleFont.Name = 'Tahoma'

TitleFont.Style = []

Columns = <

item

Expanded = False

FieldName = 'NOMFK'

Title.Caption = #1053#1086#1084#1077#1088

Title.Font.Charset = DEFAULT\_CHARSET

Title.Font.Color = clWindowText

Title.Font.Height = -13

Title.Font.Name = 'Tahoma'

Title.Font.Style = [fsBold]

Visible = True

end

item

Expanded = False

FieldName = 'NAZV'

Title.Caption = #1053#1072#1079#1074#1072#1085#1080#1077

Title.Font.Charset = DEFAULT\_CHARSET

Title.Font.Color = clWindowText

Title.Font.Height = -13

Title.Font.Name = 'Tahoma'

Title.Font.Style = [fsBold]

Visible = True

end

item

Expanded = False

FieldName = 'FIO'

Title.Caption = #1060#1048#1054

Title.Font.Charset = DEFAULT\_CHARSET

Title.Font.Color = clWindowText

Title.Font.Height = -13

Title.Font.Name = 'Tahoma'

Title.Font.Style = [fsBold]

Visible = True

end

item

Expanded = False

FieldName = 'GruppQuery'

Title.Caption = #1043#1088#1091#1087#1087#1072

Visible = False

end>

end

object Panel1: TPanel

Left = 0

Top = 0

Width = 849

Height = 81

TabOrder = 1

object bySave: TButton

Left = 472

Top = 33

Width = 129

Height = 41

Caption = #1057#1086#1093#1088#1072#1085#1080#1090#1100

TabOrder = 0

OnClick = bySaveClick

end

object byExit: TButton

Left = 640

Top = 33

Width = 121

Height = 41

Caption = #1042#1099#1081#1090#1080

TabOrder = 1

OnClick = byExitClick

end

object DBNavigator1: TDBNavigator

Left = 0

Top = 33

Width = 440

Height = 41

DataSource = DM.dsFakultet2

TabOrder = 2

end

end

end

object Grupp\_Student: TGrupp\_Student

Left = 0

Top = 0

Caption = #1043#1088#1091#1087#1087#1072'-'#1057#1090#1091#1076#1077#1085#1090#1099

ClientHeight = 537

ClientWidth = 852

Color = clBtnFace

Font.Charset = DEFAULT\_CHARSET

Font.Color = clWindowText

Font.Height = -11

Font.Name = 'Tahoma'

Font.Style = []

OldCreateOrder = False

Position = poScreenCenter

OnCreate = FormCreate

PixelsPerInch = 96

TextHeight = 13

object Panel1: TPanel

Left = 0

Top = 0

Width = 852

Height = 41

Align = alTop

TabOrder = 0

object Label1: TLabel

Left = 311

Top = 22

Width = 171

Height = 13

Caption = #1042#1099#1073#1077#1088#1080#1090#1077' '#1085#1072#1079#1074#1072#1085#1080#1077' '#1092#1072#1082#1091#1082#1083#1100#1090#1077#1090#1072

end

object DBNavigator1: TDBNavigator

Left = 0

Top = 10

Width = 210

Height = 25

DataSource = DM.dsGrupp2

TabOrder = 0

OnClick = DBNavigator1Click

end

object bySave: TButton

Left = 672

Top = 10

Width = 75

Height = 25

Caption = #1057#1086#1093#1088#1072#1085#1080#1090#1100

TabOrder = 1

OnClick = bySaveClick

end

object DBLookupComboBox1: TDBLookupComboBox

Left = 488

Top = 14

Width = 145

Height = 21

KeyField = 'NOMFK'

ListField = 'NAZV'

ListSource = DM.dsFakultet2

TabOrder = 2

OnClick = DBLookupComboBox1Click

end

end

object Panel2: TPanel

Left = 0

Top = 41

Width = 852

Height = 200

Align = alTop

TabOrder = 1

object DBGrid1: TDBGrid

Left = 1

Top = 1

Width = 850

Height = 198

Align = alClient

DataSource = DM.dsGrupp2

TabOrder = 0

TitleFont.Charset = DEFAULT\_CHARSET

TitleFont.Color = clWindowText

TitleFont.Height = -11

TitleFont.Name = 'Tahoma'

TitleFont.Style = []

Columns = <

item

Expanded = False

FieldName = 'NOMGR'

Title.Caption = #1053#1086#1084#1077#1088

Visible = True

end

item

Expanded = False

FieldName = 'NAZV'

Title.Caption = #1053#1072#1079#1074#1072#1085#1080#1077

Visible = True

end

item

Expanded = False

FieldName = 'NOMFK'

Visible = False

end

item

Expanded = False

FieldName = 'StudentQuery'

Visible = False

end>

end

end

object Panel3: TPanel

Left = 0

Top = 241

Width = 852

Height = 65

Align = alTop

TabOrder = 2

object Splitter1: TSplitter

Left = 1

Top = 1

Width = 850

Height = 3

Cursor = crVSplit

Align = alTop

ExplicitLeft = 0

ExplicitTop = 82

ExplicitWidth = 329

end

object DBNavigator2: TDBNavigator

Left = 0

Top = 35

Width = 189

Height = 25

DataSource = DM.dsStudent2

VisibleButtons = [nbFirst, nbPrior, nbNext, nbLast, nbInsert, nbDelete, nbEdit]

TabOrder = 0

end

end

object Panel4: TPanel

Left = 0

Top = 472

Width = 852

Height = 65

Align = alBottom

TabOrder = 3

object byExit: TButton

Left = 656

Top = 24

Width = 75

Height = 25

Caption = #1042#1099#1081#1090#1080

TabOrder = 0

OnClick = byExitClick

end

end

object Panel5: TPanel

Left = 0

Top = 306

Width = 852

Height = 166

Align = alClient

TabOrder = 4

object DBGrid2: TDBGrid

Left = 1

Top = 1

Width = 850

Height = 164

Align = alClient

DataSource = DM.dsStudent2

TabOrder = 0

TitleFont.Charset = DEFAULT\_CHARSET

TitleFont.Color = clWindowText

TitleFont.Height = -11

TitleFont.Name = 'Tahoma'

TitleFont.Style = []

Columns = <

item

Expanded = False

FieldName = 'NOMST'

Title.Caption = #1053#1086#1084#1077#1088

Visible = True

end

item

Expanded = False

FieldName = 'FIO'

Title.Caption = #1060#1048#1054

Width = 168

Visible = True

end

item

Expanded = False

FieldName = 'MESTOPRDIPL'

Title.Caption = #1052#1077#1089#1090#1086' '#1087#1088#1077#1076#1080#1087#1083#1086#1084#1085#1086#1081' '#1087#1088#1072#1082#1090#1080#1082#1080

Width = 223

Visible = True

end

item

Expanded = False

FieldName = 'NOMGR'

Title.Caption = #1043#1088#1091#1087#1087#1072

Visible = False

end

item

Expanded = False

FieldName = 'NOMRK'

Title.Caption = #1056#1091#1082#1086#1074#1086#1076#1080#1090#1077#1083#1100

Visible = True

end

item

Expanded = False

FieldName = 'NOMRR'

Title.Caption = #1055#1088#1086#1077#1082#1090

Visible = True

end>

end

end

end

object tbProject: TtbProject

Left = 0

Top = 0

Caption = #1055#1088#1086#1077#1082#1090#1099

ClientHeight = 411

ClientWidth = 846

Color = clBtnFace

Font.Charset = DEFAULT\_CHARSET

Font.Color = clWindowText

Font.Height = -11

Font.Name = 'Tahoma'

Font.Style = []

OldCreateOrder = False

Position = poScreenCenter

OnCreate = FormCreate

PixelsPerInch = 96

TextHeight = 13

object Panel1: TPanel

Left = 0

Top = -1

Width = 849

Height = 58

TabOrder = 0

object byExit: TButton

Left = 688

Top = 8

Width = 123

Height = 41

Caption = #1042#1099#1081#1090#1080

TabOrder = 0

OnClick = byExitClick

end

object bySave: TButton

Left = 504

Top = 8

Width = 139

Height = 41

Caption = #1057#1086#1093#1088#1072#1085#1080#1090#1100

TabOrder = 1

OnClick = bySaveClick

end

object DBNavigator1: TDBNavigator

Left = 0

Top = 8

Width = 470

Height = 43

DataSource = DM.dsProject2

TabOrder = 2

end

end

object DBGrid1: TDBGrid

Left = 0

Top = 56

Width = 849

Height = 361

DataSource = DM.dsProject2

TabOrder = 1

TitleFont.Charset = DEFAULT\_CHARSET

TitleFont.Color = clWindowText

TitleFont.Height = -11

TitleFont.Name = 'Tahoma'

TitleFont.Style = []

Columns = <

item

Expanded = False

FieldName = 'NOMPR'

Visible = True

end

item

Expanded = False

FieldName = 'TEMA'

Visible = True

end>

end

end

object tbRukovod: TtbRukovod

Left = 0

Top = 0

Caption = #1056#1091#1082#1086#1074#1086#1076#1080#1090#1077#1083#1080

ClientHeight = 411

ClientWidth = 852

Color = clBtnFace

Font.Charset = DEFAULT\_CHARSET

Font.Color = clWindowText

Font.Height = -11

Font.Name = 'Tahoma'

Font.Style = []

OldCreateOrder = False

Position = poScreenCenter

PixelsPerInch = 96

TextHeight = 13

object DBGrid1: TDBGrid

Left = 0

Top = 56

Width = 862

Height = 353

TabOrder = 0

TitleFont.Charset = DEFAULT\_CHARSET

TitleFont.Color = clWindowText

TitleFont.Height = -11

TitleFont.Name = 'Tahoma'

TitleFont.Style = []

Columns = <

item

Expanded = False

FieldName = 'NOMRK'

Title.Caption = #1053#1086#1084#1077#1088

Title.Font.Charset = DEFAULT\_CHARSET

Title.Font.Color = clWindowText

Title.Font.Height = -13

Title.Font.Name = 'Tahoma'

Title.Font.Style = [fsBold]

Visible = True

end

item

Expanded = False

FieldName = 'FIO'

Width = 146

Visible = True

end

item

Expanded = False

FieldName = 'DOLJNOST'

Width = 148

Visible = True

end

item

Expanded = False

FieldName = 'MESTORAB'

Width = 132

Visible = True

end

item

Expanded = False

FieldName = 'YCHZV'

Visible = True

end>

end

object DBNavigator1: TDBNavigator

Left = 8

Top = 8

Width = 450

Height = 42

TabOrder = 1

end

object Button1: TButton

Left = 536

Top = 8

Width = 139

Height = 42

Caption = #1057#1086#1093#1088#1072#1085#1080#1090#1100

TabOrder = 2

end

object Button2: TButton

Left = 688

Top = 8

Width = 121

Height = 42

Caption = #1042#1099#1081#1090#1080

TabOrder = 3

end

end