ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»**

**( Н И У « Б е л Г У » )**

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНАЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И РОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

# ИНФОРМАЦИОННО-СПРАВОЧНАЯ СИСТЕМА ПО ИНТЕПРИТАЦИИ РЕЗУЛЬТАТОВ ФАРМАКОГЕНЕТИЧЕСКОГО ТЕСТИРОВАНИЯ

Выпускная квалификационная работа

обучающееся по направлению подготовки

02.03.03 Математическое обеспечение и администрирование информационных систем

очной формы обучения,

группы 07001302

Гладыщевой Анастасии Сергеевны

Научный руководитель

доц. Румбешт В. В.

**БЕЛГОРОД 2017**

# СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

Лидирующую позицию среди всех причин смерти в большинстве стран мира занимают болезни системы кровообращения. Эпидемиологическая ситуация в России характеризуется термином «сверхсмертность» от сердечнососудистых заболеваний по сравнению с экономически развитыми странами

Среди всех болезней системы кровообращения наиболееираспространённой является ишемическая болезнь сердца. В нашей стране общее количество больных стенокардией составляет 30-40 тыс. на1 млн. населения. В настоящее время наблюдается процесс «омоложения» ишемической болезни сердца, больничные койки все больше заполняют пациенты трудоспособного возраста, занимающие важные позиции в обществе, потому данное заболевание следует считать одним из наиболее социально значимых [10].

В настоящее время в НИУ «БелГУ» проводится фармакогенетическое тестирование. На основе тестирования врачи кардиологи получат результаты влияние генотипа на эффективность применения лекарственных средств. Поэтому актуально автоматизировать результаты фармакогенетического тестирования, для удобной работы с результатами.

Цель: разработка информационно-справочной системы по интепритации результатов формакогенетического тестирования.

Задачи:

1. Обзор и анализ применения формакогенетического тестирования;
2. Формулировка требований;
3. Проектирвание информационно-справочной системы;

4. Программная реализация информационно-справочной системы;

5. Тестирование системы.

Дипломная работа содержит: 62 листа, 8 таблиц, 26 рисунков.

# 1. ПОСТАНОВКА ЗАДАЧИ

## 1.1 Проблемы применения фармакогенетического тестирования для индивидуального назначения лекарственных препаратов

Фармакогенетика — раздел медицинской генетики и фармакологии, изучающий зависимость реакций организма на лекарственные средства от наследственных факторов. Основной задачей является изучение этих реакций, разработка методов их диагностики, коррекции и профилактики[7].

Установлено, что причинами атипичных реакций организма на лекарственные средства обычно являются наследственные изменения ферментных систем, т.е. генетически обусловленные энзимопатии, а также некоторые наследственные болезни обмена веществ и иногда передающиеся по наследству пороки развития отдельных органов.

К наиболее распространенным фармакогенетическим реакциям на лекарственные средства относят гемолиз у лиц с врожденной недостаточностью глюкозо-6-фосфат-дегидрогеназы (Г-6-ФД) эритроцитов, синтез которой контролируется х-хромосомой. Способностью вызывать гемолиз при недостаточности Г-6-ФД обладают многие лекарственные препараты и промышленные вещества, проявляющие свойства окислителей, например производные нитрофурана (фурадонин, фуразолидон и др.), сульфаниламиды (стрептоцид, сульфапиридазин и др.), диафенилсульфон и прочие сульфоны, а также хинин, хинидин, примахин, толуидиновый синий, тринитротолуол и т.д. Имеются также потенциальные гемолитические вещества (фенацетин, ацетилсалициловая кислота, левомицетин, нитриты, метиленовый синий, аскорбиновая кислота, хингамин, акрихин и др.), кото-

рые вызывают гемолиз при определенных сопутствующих факторах, например при инфекционных заболеваниях. Возникновению гемолитических кризов в ответ на некоторые даже малоопасные в этом отношении вещества может способствовать их накопление в крови в высоких концентрациях, например при поражениях печени и почек, сопровождающихся снижением их функции. Чувствительность эритроцитов к действию веществ повышается при диабетическом ацидозе и других нарушениях электролитного баланса. Степень гемолиза обычно зависит от дозы. Гемолитическое действие веществ может проявиться не только при приеме их внутрь, но и при попадании на кожу или вдыхании паров (например, паров нафталина).

Резкое увеличение продолжительности миопаралитического эффекта дитилина наблюдается у лиц с недостаточностью сывороточной псевдохолинэстеразы, наследующейся по аутосомно-рецессивному принципу. Для прекращения действия дитилина в этих случаях (как и при его передозировке) прибегают к переливанию свежей донорской крови или внутривенному введению очищенных препаратов псевдохолинэстеразы.

Неодинаковая переносимость противотуберкулезного препарата изониазида также обусловлена фармакогенетическими причинами. Установлено, что индивидуальная чувствительность организма к нему связана с различиями в его метаболизме в печени под влиянием фермента М-ацетилтрансферазы. Наследование активности и содержания этого фермента носит аутосомно-рецессивный характер. У больных с низкой активностью данного фермента ацетилирование изониазида происходит медленно, в связи с чем в этих случаях чаще и в более выраженной степени проявляются признаки его побочного действия (головокружение, головная боль, тошнота, рвота, раздражительность, бессонница, снижение аппетита, тахикардия, боли за грудиной, периферические неврит ы и др.). Полиморфизм реакции ацетилирования в организме характерен не только для

изониазида, но и для ряда других лекарственных средств, например апрессина, сульфадимезина и диафенилсульфона.

Имеются сведения о генетических различиях в процессах окислительного метаболизма ряда лекарственных средств, что, очевидно, связано с наследственной неоднородностью зависимых от цитохрома Р-450 оксидаз.

По этой причине основные параметры фармакокинетики антипирина, бутадиона, сибазона, феназепама, дифенина, индометацина, карбамазепина, нортриптилина, пармидина, бутамида и некоторых других препаратов, метаболизирующихся под влиянием этой ферментной системы, могут иметь выраженные индивидуальные колебания.

К наследственно обусловленным реакциям на лекарства относятся также необычные изменения внутриглазного давления в ответ на введение глюкокортикоидов, повышенная токсичность меркаптопурина и азатиоприна у лиц с врожденной недостаточностью тиопуринметилтрансферазы в эритроцитах, изменения эффектов перекиси водорода при акаталазии и гипокаталазии, возникновение цианоза после приема фенацетина или сульфаниламидов при наследственной метгемоглобинемии, гипертермию, возникающую при применении дитилина и некоторых средств для наркоза (фторотана, метоксифлурана), и др.

Важной задачей Ф. является исследование влияния лекарственных средств на генетически обусловленное патологическое состояния организма. Например, наследственные нарушения пуринового обмена при подагре могут усиливаться под влиянием лекарственных средств, стимулирующих образование мочевой кислоты (меркаптопурина, азатиоприна и др.) или нарушающих ее выделение из организма (дихлотиазида и др.). При наследственной порфирии признаки ее обострения (приступы кишечной колики, полиневриты, параличи мышц, психические нарушения, судороги и т.п.) вызывают барбитураты, амидопирин, гризеофульвин, сульфаниламиды,

эстрогены и содержащие их пероральные противозачаточные средства, некоторые транквилизаторы и противоэпилептические средства, которые стимулируют активность синтетазы b-аминолевулиновой кислоты. Повышению уровня билирубина в крови при синдромах Жильбера и Дубина — Джонсона способствуют эстрогены и содержащие их пероральные противозачаточные средства[7].

Наследственные заболевания системы крови (гемофилия А, болезнь Виллебранда) являются противопоказанием к применению лекарственных средств (антиагрегантов, антикоагулянтов), нарушающих процесс тромбообразования и усиливающих вследствие этого кровоточивость тканей при указанных заболеваниях.

Условия для применения фармакогенетического тестирования в клинической практике

Фармакогенетическое тестирование особенно показано:

• пациентам с высоким риском развития НЛР;

• пациентам с наследственным анамнезом по НЛР.

Требования к ЛС, для персонализации применения которого планируется использование фармакогенетического теста:

• ЛС не имеет альтернатив в той или иной клинической ситуации;

• ЛС с большим спектром и выраженностью НЛР;

• ЛС должно применяться длительно / пожизненно;

• ЛС имеет узкий терапевтический диапазон;

• ЛС эффективно у ограниченного числа пациентов, что особенно актуально для дорогостоящих ЛС.

Требования к фармакогенетическому тесту для использования в клинической практике:

• наличие выраженной ассоциации выявляемого аллельного

варианта того или иного гена с изменением фармакологического ответа (развитием НЛР, недостаточной эффективностью или высокой эффективностью);

• фармакогенетический тест должен с высокой чувствительностью и специфичностью прогнозировать фармакологический ответ (развитие НЛР, недостаточная эффективность или высокая эффективность);

• должен быть разработан алгоритм применения ЛС в зависимости от результатов фармакогенетического тестирования (выбор ЛС, его режима дозирования);

• частота выявляемого аллельного варианта должна быть известна в популяции, в которой планируется применять фармакогенетическое тестирование;

• должны быть доказаны преимущества (в т.ч. и экономические) применения ЛС с использованием результатов фармакогенетического теста, по сравнению с традиционным подходом (повышение эффективности, безопасности фармакотерапии и экономическая рентабельность подобного подхода);

• фармакогенетический тест должен быть доступен для врачей и пациентов[6].

В настоящее время этим требованиям удовлетворяет ограниченное число фармакогенетических тестов. Принципы включения фармакогенетических тестов в рекомендации. В рекомендации включались фармакогенетические тесты (включая необходимые для определения аллельные варианты и лекаственные средсва для персонализации которых используется), для которых известно следующее:

Фармакогенетические тесты для персонализации применения лекарственных средств, для которых имеется «генетическая» информация или рекомендации по применению фармакогенетического тестирования в российских инструкциях по медицинскому применению и / или в типовых

клинико-фармакологических статьях Государственного реестра лекарственных средств1, одобренных и зарегистрированных Минздравсоцразвития РФ. Фармакогенетические тесты для персонализации применения ЛС, для которых имеется «генетическая» информация или рекомендации по применению фармакогенетического тестирования в инструкциях, утвержденных FDA и / или EMA. Фармакогенетические тесты включены в Рекомендации международных и национальных профессиональных научных общественных организаций:

Информация с официального сайта «Обращение лекарственных средств»:

Рекомендации экспертов Европейского научного фонда (ESF), обсужденные и одобренные участниками Европейской Конференции по фармакогенетике и фармакогеномике в Барселоне в июне 2010 года (опубликовано в марте 2011 года).

Рекомендации экспертов Рабочей группы фармакогенетики Королевской голландской ассоциации фармацевтов (опубликовано в марте 2011 года).

Рекомендации экспертов Консорциума по внедрению клинической фармакогенетики. Необходимость внедрения фармакогенетического теста в клиническую практику регламентировано канадской организаций, проводящей оценку медицинских технологий

Консультативным комитетом по медицинским технологиям Онтарио, Канада[6].

На данный момент в медицинском институте НИУ БелГУ, проводятся фармакогенитические исследования. Ромащенко Олеси Викторовне актуально автоматизировать результаты тестирования.

## 1.2 Требования информационной системы

Информационно-справочная система должна обеспечивать выполнения перечисленных функций:

- ведения справочника лекарственных средств для лечения ишемической болезни сердца;
- ведения справочника генов кандидатов и их аллельных вариантов;
- поддержка описания влияние генотипа на эффективность применения лекарственных средств;
- хранение личных данных пациента;
- ведение справочника анализов пациента;
- выдача рекомендаций врачу кардиологу.

*Требования к организации входных данных.*

Входными данными в программе являются данные, которые пользователи вводят в процессе заполнения таблиц – такие как:

В справочнике «группы лекарственных средств» записывается наименование группы лекарственных средств, максимальное количество символов 100.

В справочнике «лекарственные средства» записывается наименование лекарственных средств, максимальное количество символов 100, средняя дозировка препарата, будет заноситься в виде процентных соотношений.

В справочнике «гены кандидаты» записывается наименование гена кандидата, максимальное количество символов 100.

В справочнике «полиморфизм» записывается наименование полиморфизма, максимальное количество символов 100.

В справочнике «особенности применения» записывается влияние лекарственных средств, максимальное количество символов 10000, дозировка.

Общие требования к входным данным, вводимым пользователем в программу: данные должны вводиться в специальные поля если это текстовые данные. Те поля, в которые пользователь вводит информацию вручную, должны проверяться на наличие данных.

*Требования к организации выходных данных*.

Вывод данных в программы должен осуществляться путем вывода информации. Эти страницы отличаются и делятся на несколько групп, по назначению информации.

Из справочника «группы лекарственных средств» и «лекарсвенные средства» должны выводиться данные, которые представляют собой список, состоящий из:

- наименование группы лекарственных средств;
- наименование лекарственных средств;
- Средняя дозировка препарата.

Из справочника «гены кандидаты» и «полиморфизм» должны выводиться данные, которые представляют собой список, состоящий из:

- наименование гена кандидата;
- наименование полиморфизма.

Из справочника «особенности применения» должны выводиться данные, которые представляют собой список, состоящий из:

- влияние лекарственных средств;
- дозировка.

Также общими являются следующие требования к выводимым данным:

- доступ к таблицам должен осуществляться непосредственно врачам кардиологам.

*Требования к временным характеристикам.*

Требования к временным характеристикам зависит от количества запросов и пропускной способности сервера. Информационно-справочная система должна выполнять работу, осуществляя как можно меньше запросов к серверу.

*Требования к надежности.*

Надежная работа программы должна обеспечиваться выполнением организационно-технических мероприятий, приведенных ниже:

- организацией бесперебойного питания сервера;
- исполнением всех рекомендаций Министерства труда и социального развития РФ, записанных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
- исполнением требований ГОСТ 51188-98. Защита информации;
- Испытания программных средств на наличие компьютерных вирусов;

*Отказы из-за некорректных действий оператора.*

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий.

*Климатические условия эксплуатации.*

Климатические эксплуатационные условия, в которых обязаны обеспечиваться заданные характеристики, должны соответствовать требо-

ваниям, которые предъявляются к техническим средствам в части условий их эксплуатации.

*Требования к численности и квалификации персонала*.

Для управления работы приложения требуется минимум один человек, который должен наполнять его актуальной информацией. Так же требуется персонал, который обслуживает сервер.

*Требование к помещению*.

Рабочее помещение должно быть просторным. Каждый день его нужно проветривать, чтобы в воздухе не накапливались электромагнитные волны. Возле компьютера должны стоять кактусы и аквариум, создавая влажность помещению, или стакан с водой.

Для ряда технологических процессов важное значение имеет внутренняя отделка помещений, в частности, использование материалов, которые не собирают вредные вещества и не превращаются в дальнейшем в источник их выделения. В этих помещениях полы, стены, потолки должны быть плотными, гладкими, с закругленными углами, должны допускать влажную уборку. В случае значительных тепло- и влаговыделений важное значение имеет теплоизоляция стен и перекрытий с целью предупреждения образования на них конденсата в холодный период года.

*Требования к составу и параметрам технических средств*.

Для работы автоматизированной системы на стороне клиента требуется персональный компьютер.

*Требования к защите информации и программ*.

В Системе должен быть обеспечен надлежащий уровень защиты информации в соответствии с законом о защите персональной информации и программного комплекса в целом от несанкционированного доступа - “ Об информации, информатизации и защите информации” РФ N 24-ФЗ от 20.02.95.

*Специальные требования.*

Программа должна обеспечивать взаимодействие с пользователем (оператором) посредством графического пользовательского интерфейса. Программа должна обеспечивать высокую защиту данных и удобный просмотр необходимой информации.

# 2. ПРОЕКТИРОВАНИЕ И ПРОГРАММАНАЯ РЕАЛИЗАЦИЯ

Проектирование информационно-справочной системы - это один из важнейших этапов ее существования то, с чего, собственно, должна начинаться её жизнь. Любая правильная информационно-справочная система базируется на тщательно спроектированной базе данных, в которой учтены не только все особенности ведения бизнеса, но и заложена возможность будущего развития путем добавления функциональности в информационную систему.

## 2.1 Архитерктура информационно-справочной системы

Исходя из требований к информационно-справочной системы мы построили архитектуру приложения, которая представлена на рисунке 2.1.

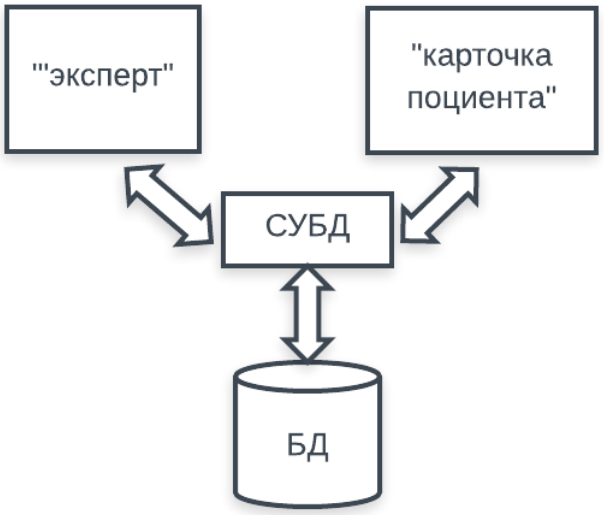


Рис 2.1 Схема архитектуры информационно справочной системы

Из архитектуры видно, что информационно справочная система состоит из двух приложений:

1. Эксперт. В редакторе пользователь сможет вносить, изменять и удалять данные таблиц;

2. Карточка для пациента. С этим приложением работает не посредственно сам врач кардиолог.

В данной работе у нас будет разработано только первое приложение из-за нехватки времени.

Обычно в понятие архитектуры входят решения об базовых аппаратных и программных составляющих системы, их функциональном назначении и организации связей между ними. Отметим, что в значительной степени архитектуру ИС определяет способ доступа к БД и возможностями СУБД, которая ее поддерживает.

Выбор архитектуры ИС влияет на следующие характеристики:

1. производительность ИС – количество работ, выполняемых в ИС за единицу времени;

2. время реакции системы на запросы пользователя (время отклика системы);

3. надёжность – способность к безотказному функционированию в течение определенного периода времени.

Локальные ИС, которые располагаются целиком на одном компьютере и предназначены для работы только одного пользователя, сейчас встречаются крайне редко. В дальнейшем речь пойдет о распределенных ИС, которые функционируют в сети и предназначены для многопользовательской (коллективной) работы.

Обычно БД целиком хранится в одном узле сети, поддерживается одним сервером и доступна для всех пользователей локальной сети, называемых клиентами. Такая база данных принято называть централизованной. Распределенные базы данных, в которых БД распределена по нескольким узлам сети, обычно используются в организациях, содержащих территориально удаленные подразделения[16].

Сервер, как правило, — самый мощный и самый надежный компьютер.

Он обязательно подключается через источник бесперебойного питания, в нем предусматриваются системы двойного или даже тройного дублирования. Учитывая зависимость от распределения функций обработки данных между сервером и клиентами различают три базовых архитектуры – «файл-сервер», «клиент-сервер» и «локальная»[16].

### 2.1.1 Архитектура файл-сервер

Самой простой архитектурой для реализации является архитектура "файл-сервер" (см рис. 2.1), но она же обладает и самым большим количеством недостатков, ограничивающих спектр решаемых ею задач. Простейшим случаем является случай, когда данные располагаются физически на том же компьютере, что и само приложение.

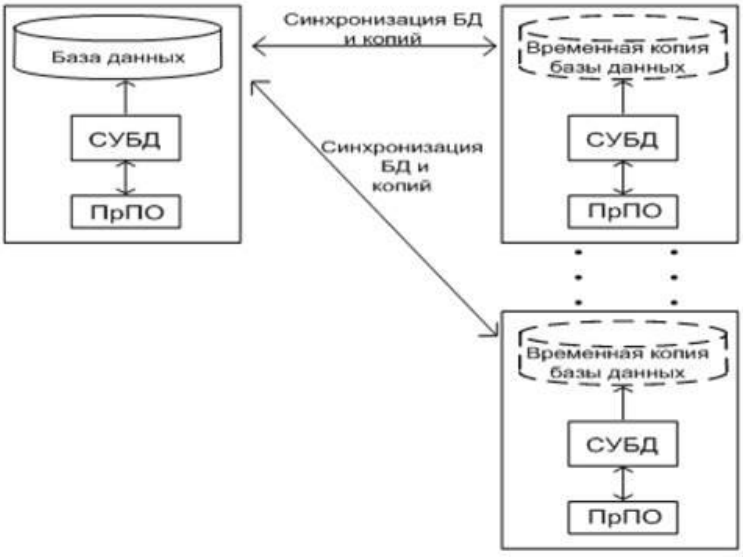


Рис. 2.2 Структура информационной системы с файл-сервером

К существенным неудобствам, возникающим при работе с системой, построенной по такой архитектуре, можно отнести следующее:

- в целом, невысокая скорость обработки и представления информации;

- высокие требования к ресурсам компьютеров. При этом возникают следующие ограничения;

- невозможность организации равноправного одновременного доступа; пользователей к одному и тому же участку базы данных.

В среде файлового сервера программа управления данными, которая выполняется на машине-клиенте, должна осуществить запрос каждой записи базы, после чего она может определить, удовлетворяет ли запись поисковым условиям, лишь после этого передать для функциональной обработки. Очевидно, что для этой схемы характерно наибольшее суммарное время обработки информации.

При всем этом система обладает одним очень важным преимуществом - низкой стоимостью[16].

### 2.1.2 Архитектура клиент-сервер

Сервером определенного ресурса в компьютерной сети называется компьютер (программа), управляющая этим ресурсом, клиентом - компьютер (программа), использующий этот ресурс. В качестве ресурса компьютерной сети могут выступать, базы данных, файловые системы, службы печати, почтовые службы. Тип сервера определяется видом ресурса, которым он управляет. Например, если управляемым ресурсом является база данных, то соответствующий сервер называется сервером базы данных.

Структура распределенной ИС, построенной по архитектуре клиент-сервер с использованием сервера баз данных, рассматривается на рисунке 2. При такой архитектуре сервер базы данных обеспечивает выполнение основного объема обработки данных. Формируемые пользователем

или приложением запросы поступают к серверу базы данных в виде инструкции языка SQL. Сервер базы данных выполняет поиск и извлечение нужных данных, которые затем передаются на компьютер пользователя. Достоинством такого подхода в сравнении с файл-сервером является заметно меньший объем передаваемых данных.

Для создания и управления персональными базами данных и приложений, работающих с ними, используются СУБД, такие как Access и Visual FoxPro фирмы Microsoft, Paradox фирмы Borland.

Корпоративная база данных создается, поддерживается и функционирует под управлением сервера баз данных, например Microsoft SQL Server. В зависимости от размеров организации и особенностей решаемых задач ИС может иметь одну из следующих конфигураций: компьютер-сервер, содержащий корпоративную и персональную базы; компьютер-сервер и персональные компьютеры с ПБД; несколько компьютеров-серверов и персональных компьютеров с ПБД.

Использование архитектуры клиент-сервер дает возможность постепенного наращивания ИС предприятия, во-первых, по мере развития предприятия; во-вторых, по мере развития самой ИС[16].

Разделение общей базы данных на корпоративную и персональные позволяет уменьшить сложность проектирования баз данных по сравнению с централизованным вариантом, а значит снизить вероятность ошибок при проектировании и стоимость проектирования.

Важнейшим достоинством применения базы данных в ИС является обеспечение независимости данных от прикладных программ, это дает возможность пользователям не заниматься проблемами представления данных на физическом уровне: размещение данных в памяти, методов доступа к ним.

Такая независимость достигается поддерживаемым СУБД многоуровневым представлением данных в базе данных на логическом (пользо-

вательском) и физическом уровнях. Благодаря СУБД и наличию логического уровня представления данных обеспечивается отделение концептуальной (понятийной) модели базы данных от ее физического представления в памяти ЭВМ. Важнейшим параметром крупной информационной системы является быстродействие при значительном количестве пользователей, а также надежность, масштабируемость и безопасность. Всё это обеспечивает архитектура "клиент-сервер". Такая архитектура позволяет оптимально распределить работу между клиентскими и серверной частями системы: теперь приложение, работающее на рабочей станции, не читает записи базы данных "напрямую", а посылает запросы на сервер, где они принимаются и последовательно отрабатываются специальными программами. В результате на рабочую станцию поступают только обработанные данные, что радикально сокращает информационные потоки в ЛВС.

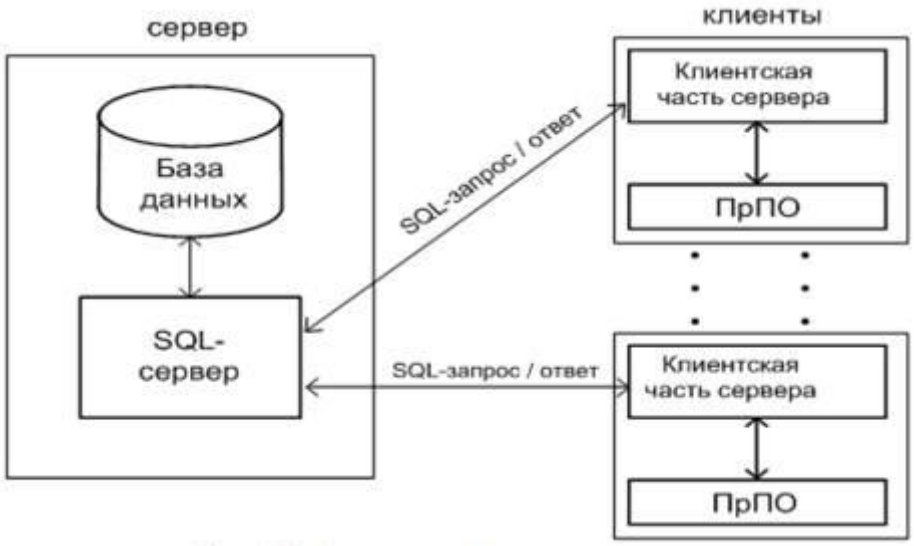


Рис. 2.3 Структура информационной системы с клиент-сервером

### 2.1.3 Локальная архитектура

Если информационная система имеет локальную архитектуру рис. 2.4, то работа с базой данных происходит, как правило, в однопользова-

тельском режиме. При необходимости можно запустить на компьютере другое приложение, одновременно осуществляющее доступ к этим данным. Для управления совместным доступом к базе данных необходимы специальные средства контроля и защиты.


Рис. 2.4 Локальная архитектура

Ромащенко Олеся Викторовна, врач кардиолог, преподаватель в медицинском институте университета НИУ «БелГУ». Проводит исследования, которыми очень дорожит, поэтому в данной работе архитектура локальная, но в дальнейшем, как Олеся Викторовна запантетует свои исследования, приложение будет разработано под архитектуру клиент-сервер.

## 2.2 Выбор системы управления базы данных

Сбор информации, ее обработка (поиск требуемых данных, сортировка и т.п.), создание форм для просмотра и распечатки данных - все эти функции обеспечивает система управления базами.

### 2.2.1 Microsoft Access

Access - это система управления базами данных (СУБД), которая позволяет не только хранить большие массивы данных в определенном формате, но и обрабатывать их, представляя в удобном для пользователей виде[17].

Также Access дает возможность автоматизировать часто выполняемые операции, разрабатывать удобные формы ввода и просмотра данных, но и составлять сложные отчеты. Access является приложением Windows, поэтому ей доступны все преимущества Windows. СУБД Access полностью совместима с другими компонентами пакета Microsoft Office, такими как электронная таблица Excel и текстовый процессор Word.

При помощи Access можно выполнить следующие задачи:

- организовать данные в легко управляемые связанные элементы;
- вводить, модифицировать и находить данные;
- извлекать данные по определенным критериям;
- создавать собственные формы и отчеты;
- автоматизировать часто используемые задачи по управлению базами данных;
- создать графики и диаграммы;
- оформлять отчеты и формы с помощью графических объектов;
- создавать собственные приложения по управлению базами данных, полностью оснащенные меню, диалоговыми окнами и управляющими кнопками.

Access - это реляционная СУБД (см рис. 2.4), поэтому с ее помощью можно работать одновременно с несколькими таблицами базы данных. Применение реляционной СУБД помогает упростить структуру данных и таким образом облегчить выполнение работы.


Рис. 2.5 Структура СУБД

В базе данных обеспечиваются все возможности динамического обмена данными DDE (Dynamic Data Exchange), также механизм OLE (Object Linking and Embedding) – связь и внедрение объектов, обеспечивающий установление связи с объектами другого приложения или внедрение объекта в базу данных. Внедряемыми или связываемыми объектами могут быть документы различных приложений Windows – рисунки, графики, электронные таблицы или звуковые файлы. Access распространил широко используемый в Windows 9x метод drag-and-drop (перетащить и отпустить) на работу с формами и отчетами, также может использовать данные других СУБД. Непосредственно может обрабатывать файлы Paradox, dBase, FoxPro, а также файлы СУБД, поддерживающих стандарт открытого доступа к данным ODBC (Open Database Connectivity) Oracle, Microsoft SQL Server, DB2, Sybase SQL Server.

В СУБД Access предусмотрено много дополнительных сервисных возможностей. Мастера для создания таблиц, форм или отчетов из имеющихся заготовок; выражения для проверки допустимости введенного значения; макросы позволяют автоматизировать многие процессы без программирования и создавать такие же мощные, ориентированные на пользователя приложения, как и приложения, созданные с помощью "полно-

ценных" языков программирования, дополнять их кнопками, меню и диалоговыми окнами; язык VBA (Visual Basic for Applications) для использования в приложениях Microsoft Office - дает возможность программировать сложные процедуры обработки данных, можно создавать программы, по мощности не уступающие самой Access. Многие инструментальные средства Access (например, мастера и конструкторы) написаны именно на VBA.

В Microsoft Access добавлено множество новых средств, разработанных для облегчения работы в Интернет и создания приложений для Web. Для доступа к сети Интернет и использования преимуществ новых средств необходимы средства просмотра Web, например Microsoft Internet Explorer[17].

### 2.2.2 Firebird

СУБД FireBird является одной из самых популярных в мире бесплатных, кросплатформенных систем управления базами данных с открытым исходным кодом. Она была разработана на основе исходного кода СУБД Interbase и развивается сегодня независимым международным сообществом. По надёжности, производительности и функциональным возможностям эта система мало в чём уступает признанным лидерам своего класса - Oracle и Microsoft SQL Server[18].

Соответствие требованиям ACID: Firebird сделан специально, чтобы удовлетворять требованиям “атомарности, целостности, изоляции и надёжности” транзакций (“Atomicity, Consistency, Isolation and Durability”).

Версионная архитектура: Основная особенность Firebird — версионная архитектура, позволяющая серверу обрабатывать различные версии одной и той же записи в любое время таким образом, что каждая транзакция видит свою версию данных, не мешая соседним (“читающие транзак-

ции не блокируют пишущие, а пишущие не блокируют читающих”). Это позволяет использовать одновременно OLTP и OLAP запросы.

Хранимые процедуры: Используя язык PSQL (процедурный SQL) Firebird, возможно создавать сложные хранимые процедуры для обработки данных полностью на стороне сервера. Для генерации отчётов особенно удобны хранимые процедуры с возможностью выборки, возвращающие данные в виде набора записей. Такие процедуры можно использовать в запросах точно так же как и обычные таблицы.

События: Хранимые процедуры и триггеры могут генерировать события, на которые может подписаться клиент. После успешного завершения транзакции (COMMIT) он будет извещён о произошедших событиях и их количестве.

Генераторы: Идея генераторов (последовательностей) делает возможной простую реализацию автоинкрементных полей, и не только их. Генераторы являются 64-битными хранимыми в базе данных счётчиками, работающими независимо от транзакций. Они могут быть использованы для различных целей, таких как генерация первичных ключей, управление длительными запросами в соседних транзакциях, и т. д.

Базы данных только для чтения: позволяют распространять базы данных, к примеру, на CD-ROM. Особенно упрощает распространение данных их использование в комбинации с встраиваемой версией сервера Firebird (Firebird Embedded)[18].

Полный контроль за транзакциями: Одно клиентское приложение может выполнять множество одновременных транзакций. В разных транзакциях могут быть использованы разные уровни изоляции. Протокол двухфазного подтверждения транзакций обеспечивает гарантированную устойчивость при работе с несколькими базами данных. Так же доступны оптимистическое блокирование данных и точки сохранения транзакций.

Резервное копирование на лету: Для резервного копирования нет надобности останавливать сервер. Процесс резервного копирования сохраняет состояние базы данных на момент своего старта, не мешая при этом работе с базой. Кроме того, существует возможность производить инкрементальное резервное копирование БД.

Триггеры: Для каждой таблицы возможно назначение нескольких триггеров, срабатывающих до или после вставки, обновления или удаления записей. Для триггеров используется язык PSQL, позволяя вносить начальные значения, проверять целостность данных, вызывать исключения, и т. д. В Firebird 1.5 появились “универсальные” триггеры, позволяющие в одном триггере обрабатывать вставки, обновления и удаления записей таблицы.

Внешние функции: библиотеки с UDF (User Defined Function) могут быть написаны на любом языке и легко подключены к серверу в виде DLL/SO, позволяя расширять возможности сервера “изнутри”.

Декларативное описание ссылочной целостности: Обеспечивает непротиворечивость и целостность многоуровневых отношений “master-detail” между таблицами.

Наборы символов: Firebird поддерживает множество международных наборов символов (включая Unicode) с множеством вариантов сортировки[18].

Исходя из архитектуры мы остановим свой выбор на Microsoft Access, так как:

- Данная СУБД легка в распространении;
- Устойчива к сбоям;
- Не требует установки сервера.

## 2.3 Проектирование базы данных

Перед созданием базы данных разработчик должен определить, из каких таблиц должна состоять база данных, какие данные нужно поместить в каждую таблицу, как связать таблицы. Эти вопросы решаются на этапе проектирования базы данных.

Так как мы будем проектировать часть информационно-справочной системы, на моделях базы данных представлена только часть.

### 2.3.1 Логическая структура базы данных

Исходя из входной и выходной информации, мы будем строить логическую модель.

Справочник «группа лекарственных средств» будет состоять из двух полей:

- код группы. Поле с первичным ключом;
- наименование группы. Поле с текстовым типом.

Справочник «лекарственные средства» будет состоять из четырех полей:

- код лекарства. Поле с первичным ключом;
- код группы. Поле с вторичным ключом;
- наименование лекарства. Поле с текстовым типом;
- дозировка. Поле числовым типом.

Справочник «гены кандидаты» будет состоять из двух полей:

- код гена. Поле с первичным ключом;
- наименование гена. Поле с текстовым типом.

Справочник «полиморфизм» будет состоять из тех полей:

- код полиморфизма. Поле с первичным ключом;

- код гена. Поле с вторичным ключом;
- наименование полиморфизма. Поле с текстовым типом.

Справочник «особенности применения» будет состоять из пяти полей:

- код применения. Поле с первичным ключом;
- код полиморфизма. Поле с вторичным ключом;
- код лекарства. Поле с вторичным ключом;
- влияние. Поле с текстовым типом;
- дозировка. Поле числовым типом.

Таблица «гены кандидаты» связана с таблицей «полиморфизм», связь осуществляется один ко многим. Таблица «группа лекарственных средств» связана с таблицей «лекарственные средства», связь осуществляется один ко многим. Таблицы «лекарственные средства» и таблица «полиморфизм» связаны с таблицей «особенности применения» связь осуществляется один ко многим.

Исходя из всего выше перечисленного мы составили логическую модель базы данных. На рисунке  рис. 2.5 представлена логическая модель базы данных.


Рис. 2.5 Логическая модель базы данных

Очевидно, что все сущности состоят в третьей нормальной форме.

### 2.3.2 Физическая структура базы данных

Физические модели баз данных определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. Первыми системами хранения и доступа были файловые структуры и системы управления файлами, которые фактически являлись частью операционных систем. СУБД создавала над этими файловыми моделями свою надстройку, которая позволяла организовать всю совокупность файлов таким образом, чтобы она работала как единое целое и получала централизованное управление от СУБД. Однако непосредственный доступ осуществлялся на уровне файловых команд, которые СУБД использовала при манипулировании всеми файлами, составляющими хранимые данные одной или нескольких баз данных.

Этап физического проектирования заключается в определении схемы хранения, т.е. физической структуры БД. Схема хранения зависит от той физической структуры, которую поддерживает выбранная СУБД. Физическая структура БД, с одной стороны, должна адекватно отражать логическую структуруБД, а с другой стороны, должна обеспечивать эффективное размещение данных и быстрый доступ к ним. Результаты этого этапа документируются в форме схемы хранения на языке определения данных выбранной СУБД. Принятые на этом этапе решения оказывают огромное влияние на производительность системы.

Одной из важнейших составляющих проекта базы данных является разработка средств защиты БД. Защита данных имеет два аспекта:

- защита от сбоев;
- защита от несанкционированного доступа.

Для защиты от сбоев на этапе физического проектирования разрабатывается стратегия резервного копирования.

Для защиты от несанкционированного доступа каждому пользователю доступ к данным предоставляется только в соответствии с его правами доступа, набор которых также является составной частью проекта БД.

Каждое реляционное отношение соответствует одной сущности и в него вносятся все атрибуты этой сущности. Для каждого отношения определяются первичный ключ и внешние ключи. В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записей.

Исходя из проектирования логической модели базы данных, были созданы таблицы. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной.

Для сущности «группа лекарственных средств» соответствует таблица 2.1

Таблица 2.1

Группа лекарственных средств

| Атрибуты | Тип | Ключ | Описание |
|---|---|---|---|
| id_group | Счетчик | primary | Код группы |
| name_g | Текстовый | | Наименование группы лекарственных средств |

- код группы. Поле с первичным ключом;

- наименование группы. Поле с текстовым типом, в это поле вводится наименование группы лекарственных средств, максимальная длина поля составляет 100 символов

Для сущности «лекарственные средства» соответствует таблица 2.2

Таблица 2.2

Лекарственные средства

| Атрибуты | Тип | Ключ | Описание |
|---|---|---|---|
| id_med | Счетчик | primary | Код лекарственных средств |
| id_group | Числовой | Foreign | Код группы |
| name_m | Текстовый | | Наименование лекарственных средств |
| c_dose | Числовой | | Средняя дозировка препарата |

- код лекарства. Поле с первичным ключом;
- код группы. Поле с вторичным ключом, по это полю происходит связь с таблицей группы лекарственных средств, связь один ко многим;
- наименование лекарства. Поле с текстовым типом, в это поле вводится наименование лекарственных средств, максимальная длина поля составляет 100 символов;
- дозировка. Поле числовым типом, в это поле вводится средняя дозировка лекарственных средств.

Для сущности «гены кандидаты» соответствует таблица 2. 3

Таблица 2.3

Гены кандидаты

| Атрибуты | Тип | Ключ | Описание |
|---|---|---|---|
| id_den | Счетчик | primary | Код гена |
| name_gen | Текстовый | | Наименование гена кандидата |

- код гена. Поле с первичным ключом;
- наименование гена. Поле с текстовым типом, в это поле вводится наименование генов кандидатов, максимальная длина поля составляет 100 символов;

Для сущности «полиморфизм» соответствует таблица 2.4

Таблица 2.4

Полиморфизм

| Атрибуты | Тип | Ключ | Описание |
|---|---|---|---|
| id_pol | Счетчик | primary | Код полиморфизма |
| id_gen | Числовой | Foreign | Код гена |
| name_p | Текстовый | | Наименование полиморфизма |

- код полиморфизма. Поле с первичным ключом;
- код гена. Поле с вторичным ключом, по это полю происходит связь с таблицей гены кандидаты, связь один ко многим;
- наименование полиморфизма. Поле с текстовым типом, в это поле вводится наименование полиморфизма, максимальная длина поля составляет 100 символов.

Для сущности «особенности применения» соответствует таблица 2.5

Таблица 2.5

Особенности применения

| Атрибуты | Тип | Ключ | Описание |
|---|---|---|---|
| id_prim | Счетчик | primary | Код применения |
| id_pol | Числовой | Foreign | Код полиморфизма |
| id_med | Числовой | Foreign | Код лекарственных средств |
| vliyanie | Текстовый | | Влияние препарата |
| dose | Числовой | | дозировка |

- код применения. Поле с первичным ключом;
- код полиморфизма. Поле с вторичным ключом, по это полю происходит связь с таблицей полиморфизм, связь один ко многим;
- код лекарства. Поле с вторичным ключом, по это полю происходит связь с таблицей лекарственные средства, связь один ко многим;
- влияние. Поле с текстовым типом, в этом поле описывается влияние генотипа на эффективность применения лекарственных средств;
- дозировка. Поле числовым типом, вносится дозировка по результатам влияния.

Данные таблицы были сделаны в MS Access.

Таблице «группы лекарственных средств» соответствует созданная таблица «group of medicines», которая представлена на рис 2.7

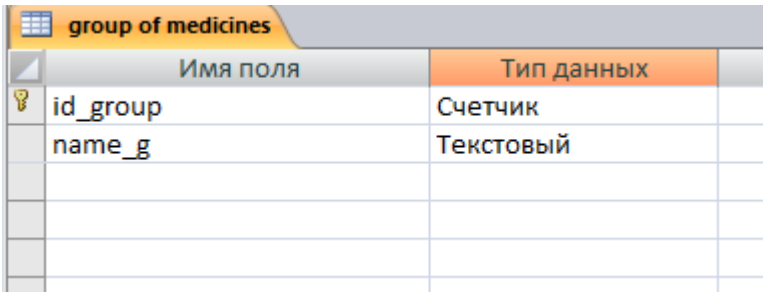


Рис 2.6 Создание таблицы «группа лекарственных средств»

Поле «id_group»:

- размер поля: длинное целое;
- новые значения: последовательные;
- индексированное поле: да (совпадения не допускаются).

Поле «name_g»:

- размер поля: 100;
- обязательное поле: да;
- пустые строки: нет.

Таблице «лекарственные средства» соответствует созданная таблица «medicinal products», которая представлена на рис 2.8

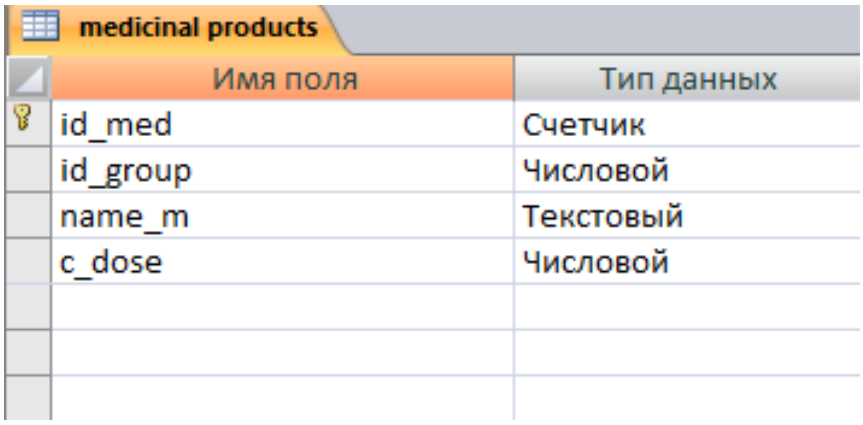


Рис 2.7 Создание таблицы «лекарственные средства»

Поле «id_med»:

- размер поля: длинное целое;
- новые значения: последовательные;

- индексированное поле: да (совпадения не допускаются).

Поле «id_group»:

- размер поля: длинное целое;
- обязательное поле: да.

Поле «name_m»:

- размер поля: 100;
- обязательное поле: да;
- пустые строки: нет.

Поле «c_dose»:

- размер поля: двойное с плавающей точкой;
- обязательное поле: да;
- пустые строки: нет.

Таблице «гена кандидаты» соответствует созданная таблица «genes candidates», которая представлена на рис 2.9

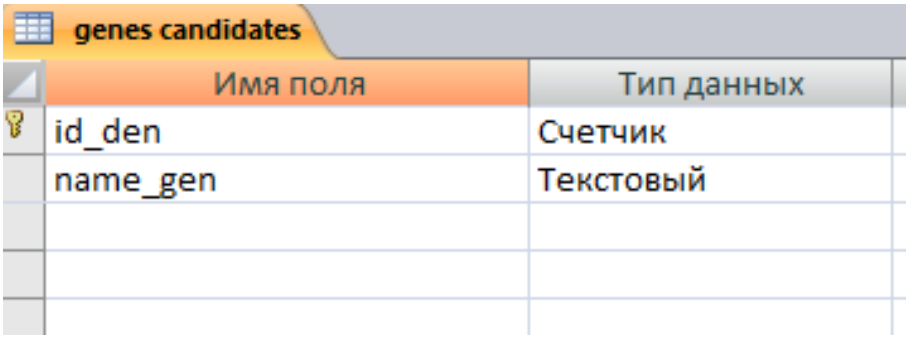


Рис 2.8 Создание таблицы «гены кандидаты»

Поле «id_den»:

- размер поля: длинное целое;
- новые значения: последовательные;
- индексированное поле: да (совпадения не допускаются).

Поле «name_gen»:

- размер поля: 100;
- обязательное поле: да;

- пустые строки: нет.

Таблице «полиморфизм» соответствует созданная таблица «polymorphism», которая представлена на рис 2.10

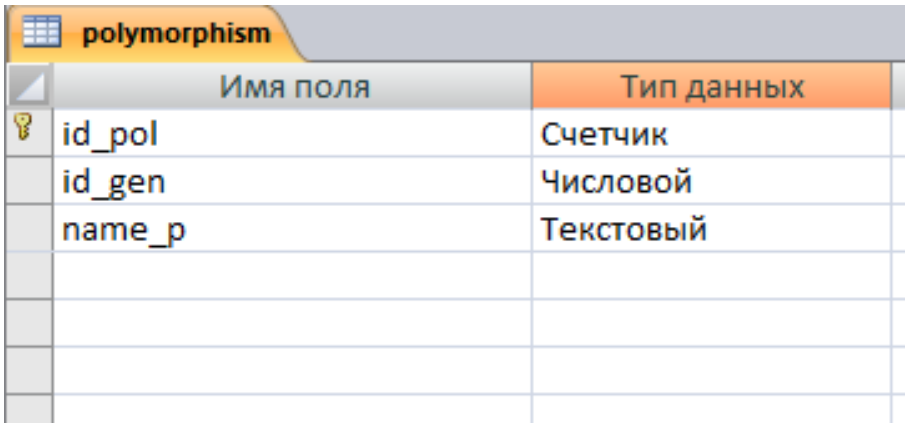


Рис 2.9 Создание таблицы «Полиморфизм»

Поле «id_pol»:

- размер поля: длинное целое;
- новые значения: последовательные;
- индексированное поле: да (совпадения не допускаются).

Поле «id_gen»:

- размер поля: длинное целое;
- обязательное поле: да.

Поле «name_p»:

- размер поля: 100;
- обязательное поле: да;
- рустые строки: нет.

Таблице «особенности применения» соответствует созданная таблица «application features», которая представлена на рис 2.11

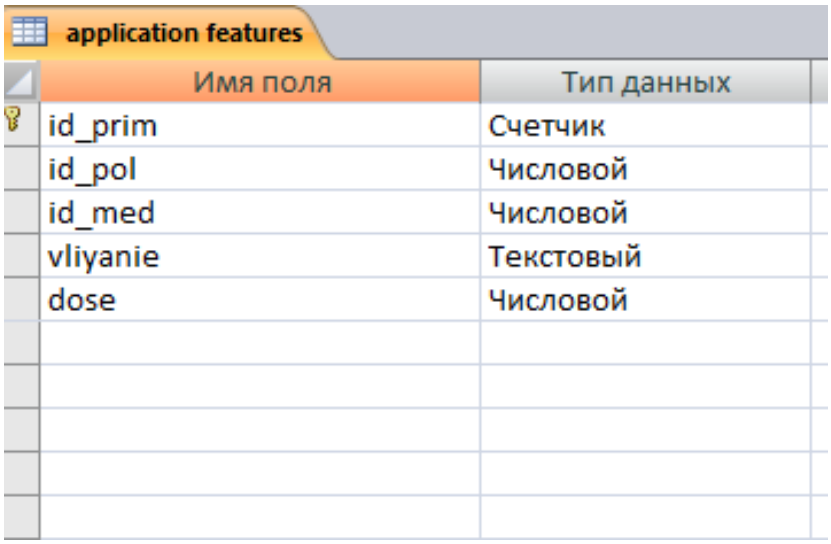


Рис 2.10 Создание таблицы «Особенности применения»

Поле «id_prim»:

- размер поля: длинное целое;
- новые значения: последовательные;
- индексированное поле: да (совпадения не допускаются).

Поле «id_pol»:

- размер поля: длинное целое;
- обязательное поле: да.

Поле «id_med»:

- размер поля: длинное целое;
- обязательное поле: да.

Поле «vliyanie»:

- размер поля: 1000;
- обязательное поле: да;
- пустые строки: нет.

Поле «dose»:

- размер поля: двойное с плавающей точкой;
- обязательное поле: да.

После создания всех таблиц, мы можем составить физическую модель рис 2.11

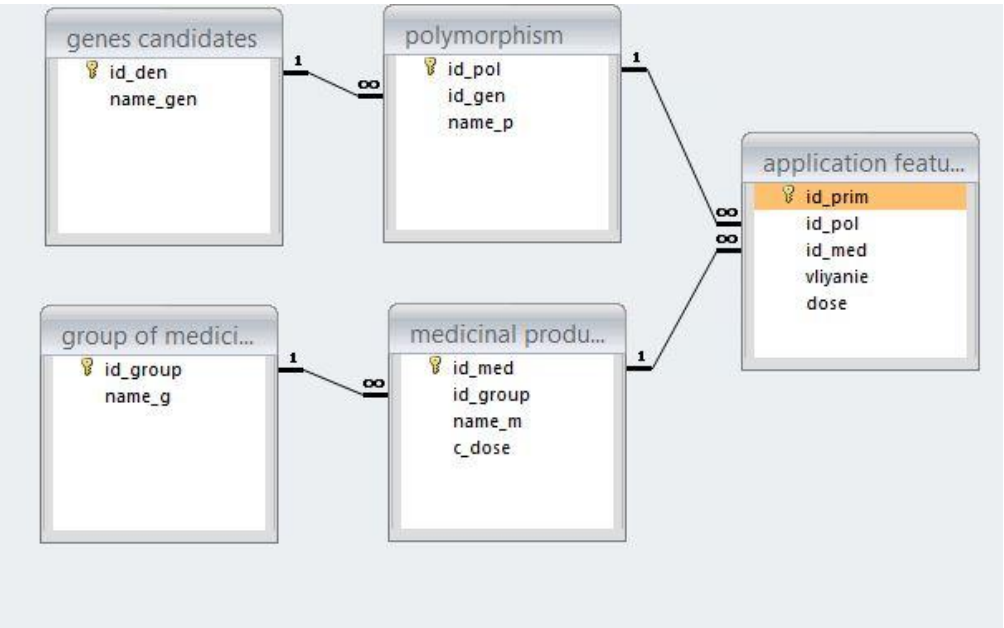


Рис 2.11 Физическая модель

## 2.4 Проектирование модульной структуры приложения

Исходя из требований можно выделить головной модуль, который реализует работу с модулями:

- работа с лекарственными средствами;
- работа с полиморфизмом;
- работа с особенностями применения.

Модуль работа с лекарственными средствами осуществляет работу с модулями

- группа лекарственных средств. Осуществляет добавление, удаление, изменение данных из области данных группа лекарственных средств;
- лекарственные средства Осуществляет добавление, удаление, изменение данных из области данных лекарственные средства.

Модуль работа с полиморфизмом осуществляет работу с модулями

- гены кандидаты. Осуществляет добавление, удаление, изменение данных из области данных гены кандидаты;
- полиморфизм. Осуществляет добавление, удаление, изменение данных из области данных полиморфизм.

Модуль работа с особенностями применения осуществляет работу с модулем

- особенности применения. Осуществляет добавление, удаление, изменение данных из области данных особенности применения .

Из всего выше перечисленного выходит модульная структура, которая представлена на рис 2.12

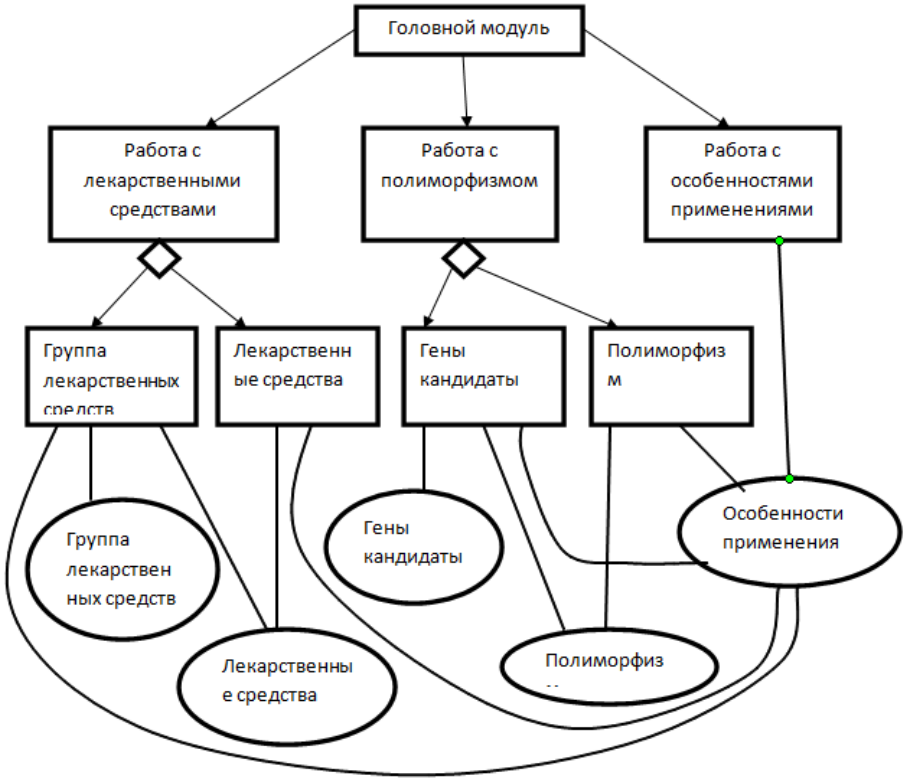


Рис 2.12 модульная структура приложения

## 2.5 Программаная реализация модулей

Головной модуль реализует главное окно, которое представлено на рисунке. Для этого на форму были расположены следующие компоненты:

- PageControl - является многостраничной панелью, которая позволяет экономить пространство окна приложения, размещая на одном и том же месте страницы разного содержания;
- ADOTable – служит для управления таблицами в базе данных;
- ADOConnection –служит для подключения базы данных;
- DBGrid – просмотр и редактирования базы данных;
- DataSourse - представляет собой источник данных, который; обеспечивает связь между набором данных и компонентами отображения и редактирования данных.
- Button – взывает форму «редактирование».

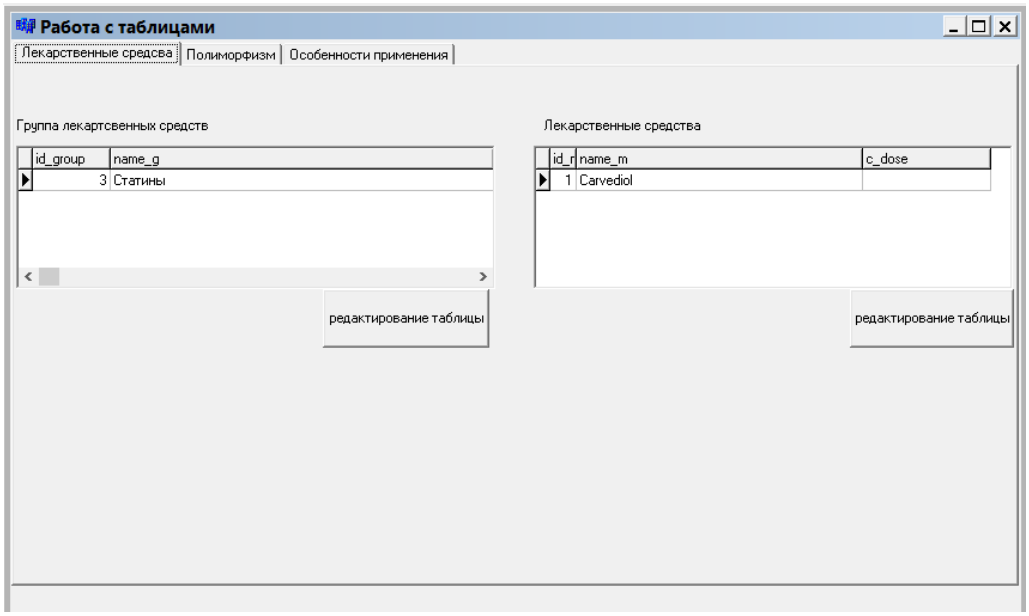


Рис. 2.13 Главное окно приложения

На форме «редактирование» пользователь может добавить, удалить, изменить данные (см рис), для этого на форме были расположены следующие компоненты:

- PageControl - является многостраничной панелью, которая позволяет экономить пространство окна приложения, размещая на одном и том же месте страницы разного содержания;
- Edit – вводятся данные для заполнения полей базы данных;

- Button – отправляет данные в таблицу.

После тог, как мы разместили компоненты начинаем их настройку.

Чтобы создать несколько вкладок на форме нажимаем, провой кнопкой мыши на компонент PageControl и в открывшемся меню нажимаем на «New Page», создаем определенное количество страниц.

Затем мы подключаемся к базе данных. Для этого нажимаем на компонент DataSource1 и в свойстве DataSet выбираем «ADOTable1», затем нажимаем на компонент ADOTable1 и в свойстве Connection выбираем «ADOConnection1». Дважды нажимаем на компанент ADOConnection1 и нажимаем кнопку Build... .Выбираем как показано на рис и нажимаем "Далее". И нажимаем два раза "Ок".

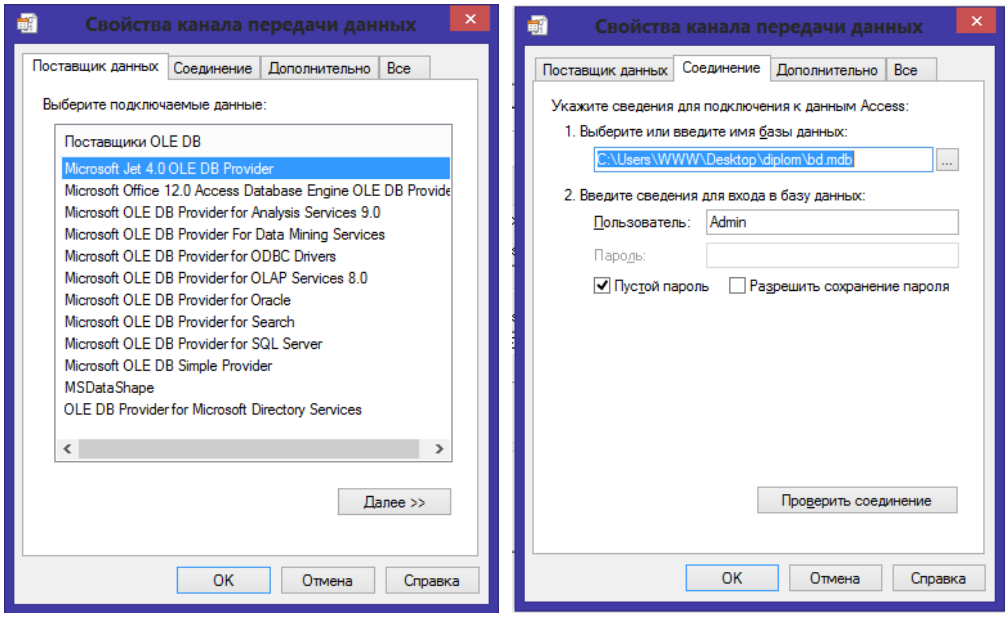


Рис. 2.15 подключенеие к базе данных

Теперь, в Object Inspector ставим свойство Connected в «true», в User Name напишем «Admin», поле пароля оставим пустым. Чтобы больше не повторять эту процедуру, установим в свойстве LoginPrompt в «false». Нажимаем на компонент DBGrid и в свойстве DataSource выбираем «DataSource1». Нажимаем на компанент ADOTable1 в свойстве TableName, выбираем нашу таблицу. Устанавливаем в свойстве Active в «true». Нажи-

маем дважды по DBGrid, в появившемся окошечке нажимаем на кнопку "Add All Fields". Тоже самое проделываем и с другими таблицами. Затем на форме размещаем компонент  Button, он будет вызывать форму для редактирования. В свойстве caption переименовываем на «редактирование таблицы». Затем в обработчике событий пишем код, который представлен в приложении 1. Создаем вторую форму и переименовываем в «редактирование» рис 2.14.

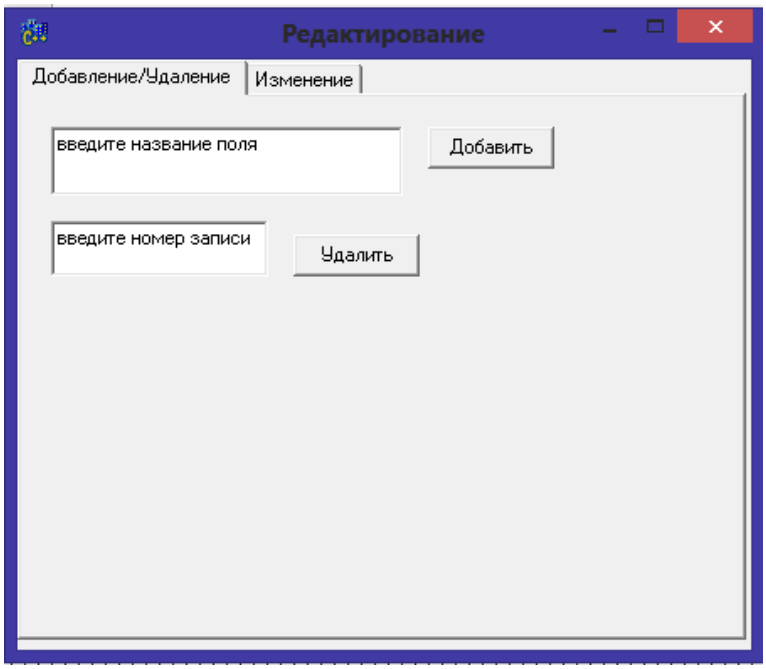


Рис. 2.14 Форма редактирования

На форме размещаем компаненты:

- PageControl;
- Edit;
- Button.

Компоненты Button предназначены для добавления, удаления, изменения данных в базе данных.

Для кнопки «добавить» в обработчике событий пишем код, который представлен в приложении 1.

# 3. ИСПЫТАНИЕ

## 3.1 Программы и методы испытаний

В соответствии с требованиями к информационно-справочной системе мы будем осуществлять проверку нашего приложения. Для это нам необходимо подготовить тестовые данные.

1. Ведение справочника лекарственных средств. Для проверки этого требования мы подготовили тестовые данные, которые представлены в таблице 3.1

Таблица 3.1

Тестовые данные

| Группа лекарственных средств | Лекарственные средства | Средняя дозировка |
|---|---|---|
| Бета-адреноблокаторы | Carvedilol | 25 мг в сутки |
| Антикоагуаны | Warfarin | 2,5-7,5 м |
| Статины | Simvastatin | 30 м |

Для того что бы просмотреть справочник лекарственных средств нам нужно запустить приложение и открыть вкладку «лекарственные средства».

2. Ведение справочника генов кандидатов и их аллельных вариантов. Для проверки этого требования мы подготовили тестовые данные, которые представлены в таблице 3.2

Таблица 3.2

Тестовые данные

| Гены кандидаты | Полиморфизм |
|---|---|
| Кодирующие белки | Однонуклеотидный полиморфизм |

Таблица 3.2 продолжение

| Гены-ингибиторы | Вариации числа копий |
|---|---|

Для того что бы просмотреть справочник лекарственных средств нам нужно запустить приложение и открыть вкладку «полиморфизм».

3. Поддержка описания влияния генотипа на эффективность применения лекарственных средств. Для проверки этого требования мы подготовили тестовые данные, которые представлены в таблице 3.3

Таблица 3.3

Тестовые данные

| Влияние | Дозировка |
|---|---|
| артериальная гипертензия взрослым | 12,5 мг |
| стабильная стенокардия | 25 мг |

Для того что бы просмотреть справочник лекарственных средств нам нужно запустить приложение и открыть вкладку «особенности применения».

## 3.2 Результаты тестирования

Выполняем тестирование согласно методам испытаний. Для того чтобы посмотреть справочник по лекарственным средствам надо запустить приложение и нажать вкладку «лекарственные средства». На рисунке 3.1 показана вкладка «лекарственные средства»

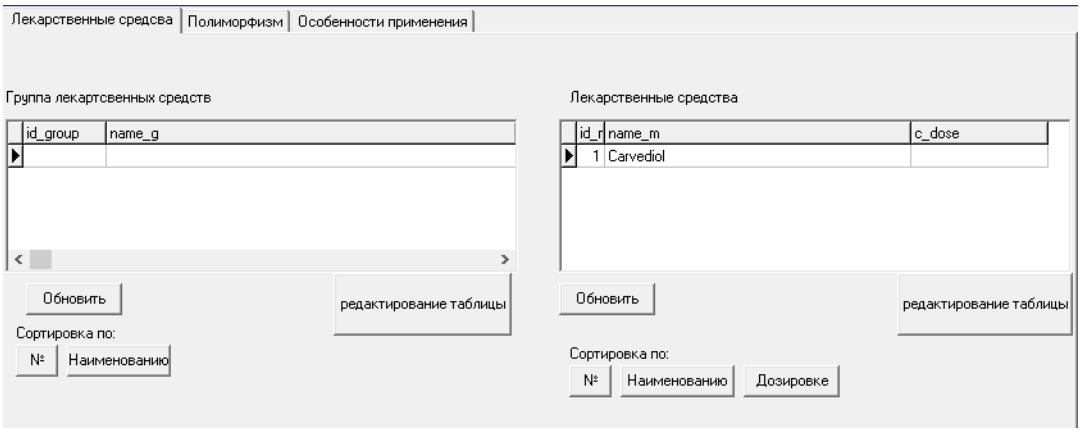


Рис 3.1 вкладка «лекарственные средства»

Для того, что бы добавить запись в таблицы нужно нажать кнопку «редактирование таблицы». После это появится оно редактор, как показано на рис 3.2

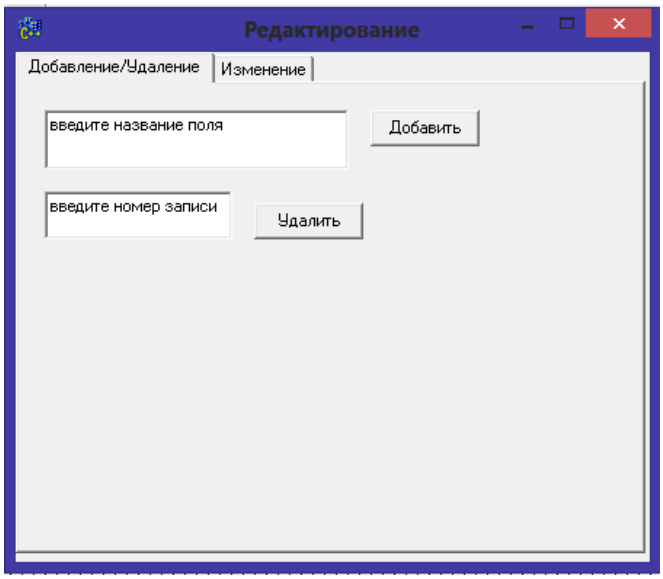


Рис 3.2 Окно редактирования

Для добавления или удаления записи мы нажимаем вкладку «добавление/удаление». Затем вводим наименования поля и нажимаем кнопку «добавить». На рисунке 3.3 продемонстрировано добавление.

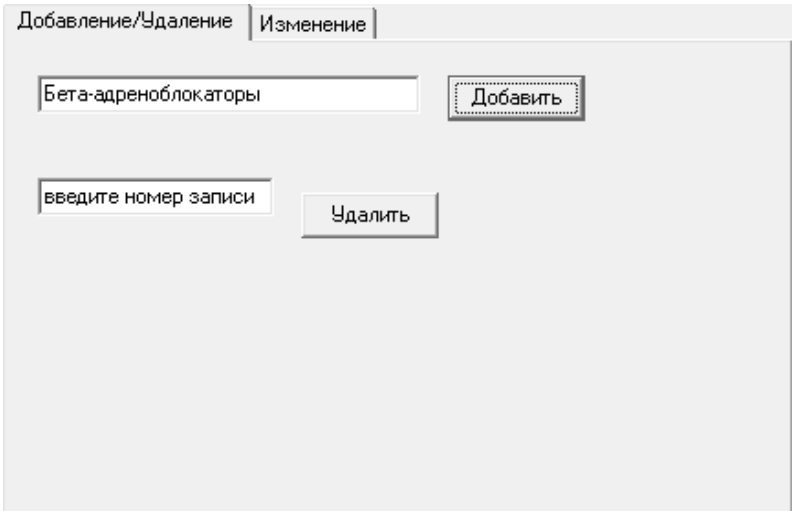


Рис 3.3 Добавление записи

Используем тестовые данные и добавляем записи в нашу таблицу. После того как мы добавили нужное количество записей, мы переходим к главному окну приложения. При необходимости можно сделать обновление таблицы. Для этого нужно нажать кнопку «обновить». На рис 3.4 показаны таблицы после добавлений записей.

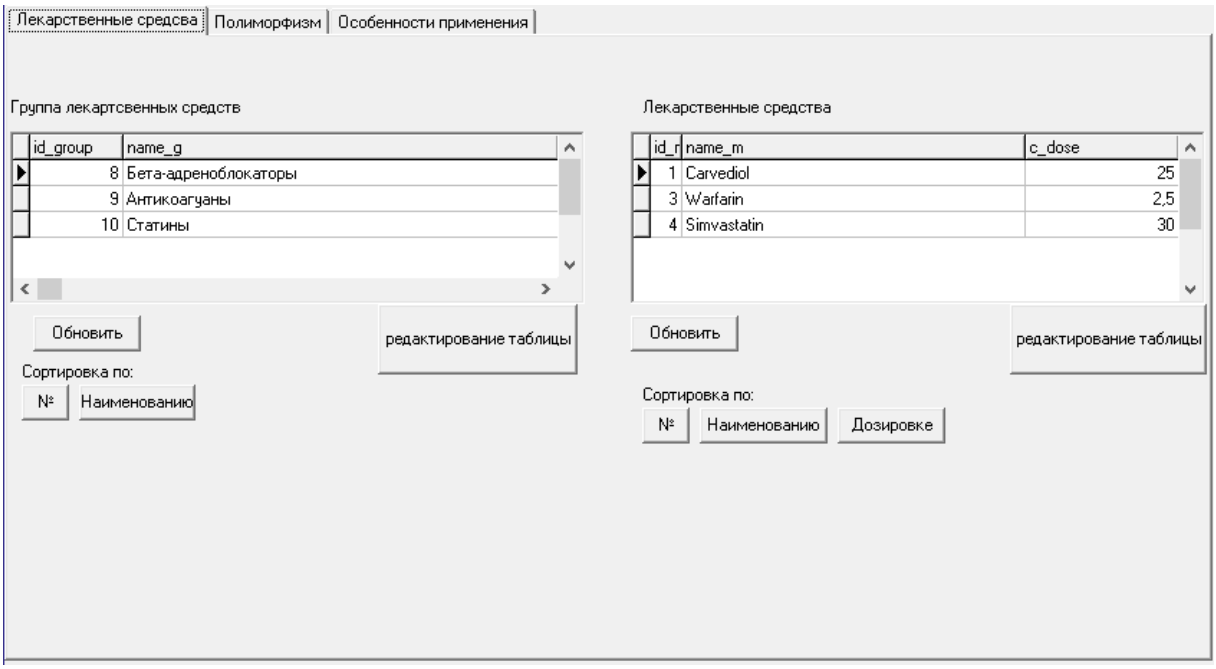


Рис 3.4 Просмотр таблиц

Так же по желанию можно отсортировать. Для этого нужно нажать ту кнопку, по которой будет происходить сортировка данных. Для удаления записи, нажимаем кнопку «редактирование таблиц». После того как появится окно редактора, в нужном поле вводим номер записи, которую хотим удалить, как показано на рисунке 3.5


Рис 3.5 Удаление записи

После удаления записи можно закрыть окно редактора и вернуться к главному окну. При необходимости нажать кнопку «обновить». На рис 3.6 показана таблица после удаления записей.

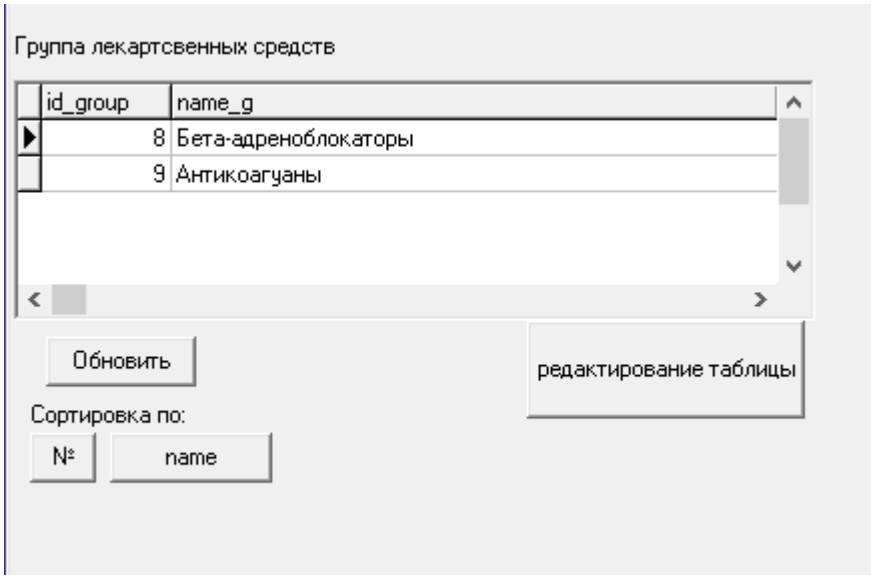


Рис 3.6 Таблица после удаления

Для того чтобы посмотреть справочник полиморфизм надо запустить приложение и нажать вкладку «лекарственные средства». На рисунке 3.7 показана вкладка «Полиморфизм»

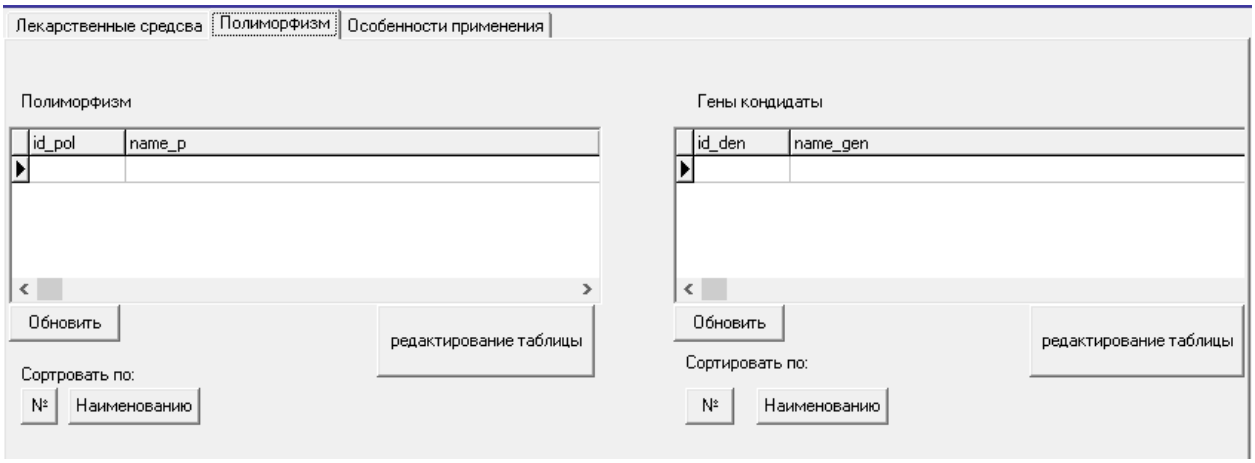


Рис 3.7 вкладка «полиморфизм»

Для того, что бы добавить запись в таблицы нужно нажать кнопку «редактирование таблицы». После это появится оно редактор, как показано на рис 3.8

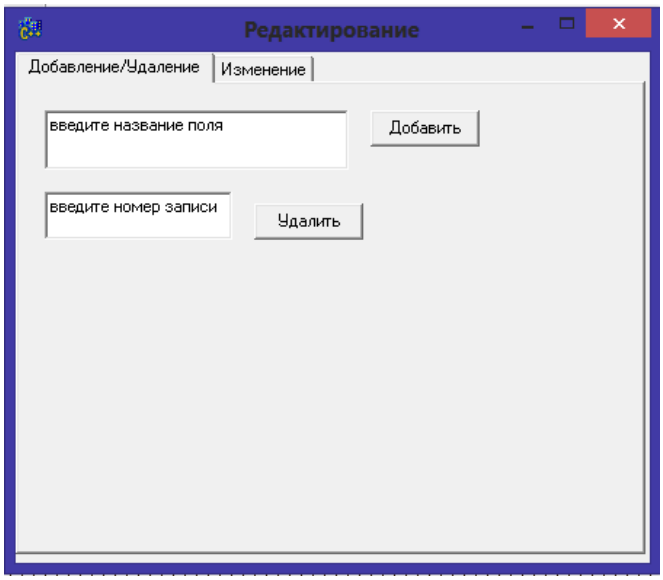


Рис 3.8 Окно редактирования

Для добавления или удаления записи мы нажимаем вкладку «добавление/удаление». Затем вводим наименования поля и нажимаем кнопку «добавить». На рисунке 3.9 продемонстрировано добавление.

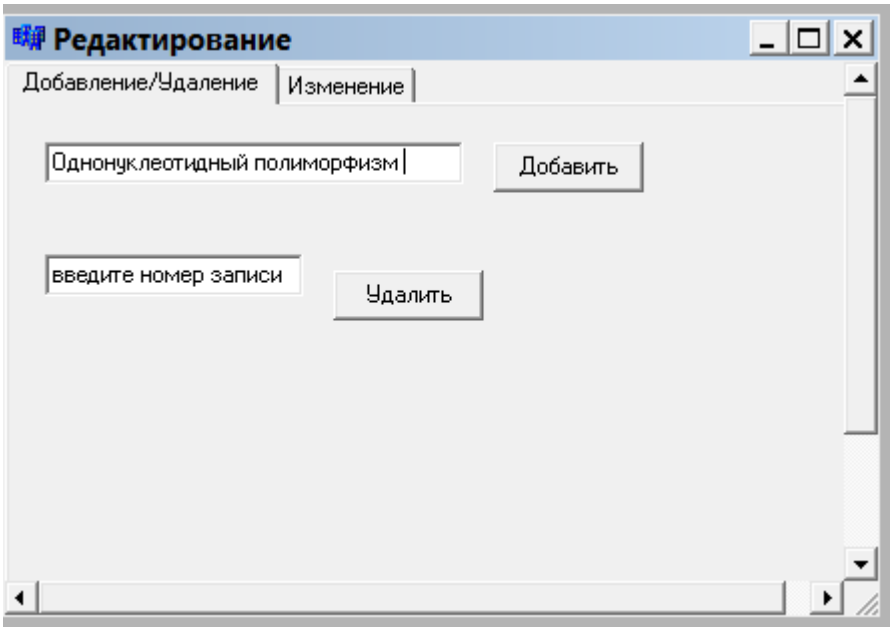


Рис 3.9 Добавление записи

Используем тестовые данные и добавляем записи в нашу таблицу. После того как мы добавили нужное количество записей, мы переходим к главному окну приложения. При необходимости можно сделать обновление таблицы. Для этого нужно нажать кнопку «обновить». На рис 3.10 показаны таблицы после добавлений записей.

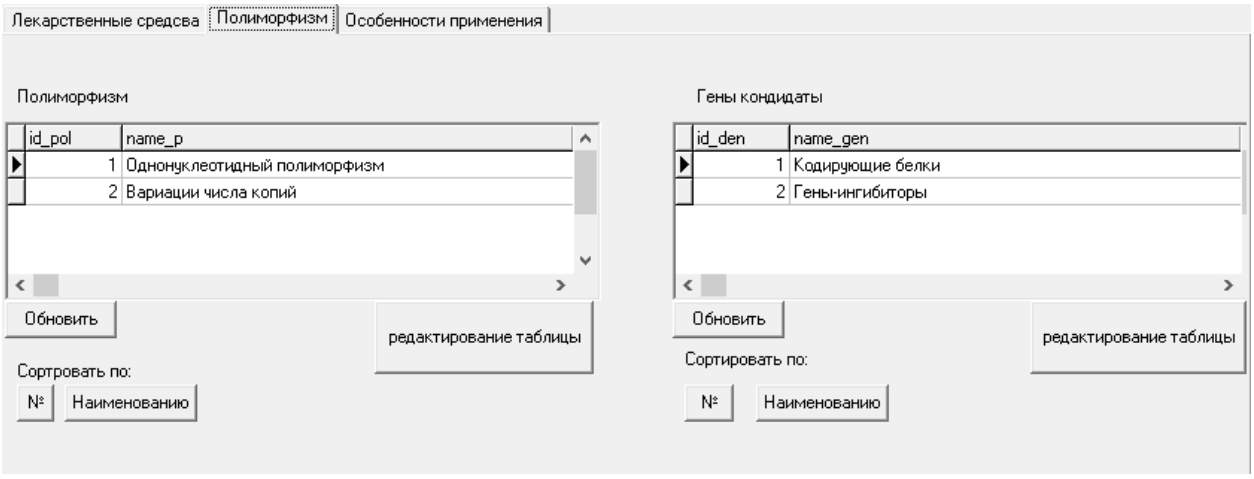


Рис 3.10 Просмотр таблиц

Так же по желанию можно отсортировать. Для этого нужно нажать ту кнопку, по которой будет происходить сортировка данных. Для удаления записи, нажимаем кнопку «редактирование таблиц». После того как появится окно редактора, в нужном поле вводим номер записи, которую хотим удалить, как показано на рисунке 3.11

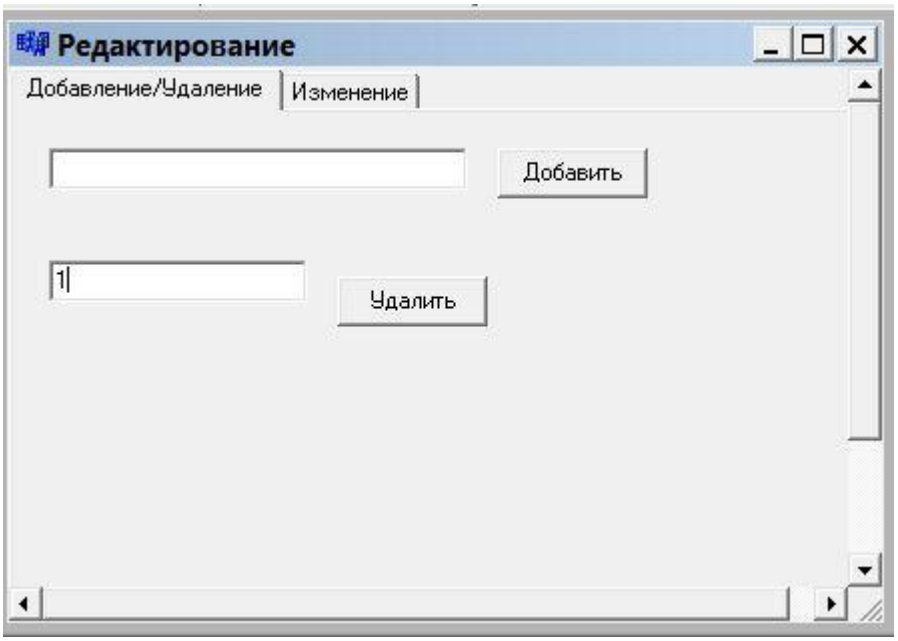


Рис 3.11 Удаление записи

После удаления записи можно закрыть окно редактора и вернуться к главному окну. При необходимости нажать кнопку «обновить». На рис 3.12 показана таблица после удаления записей.

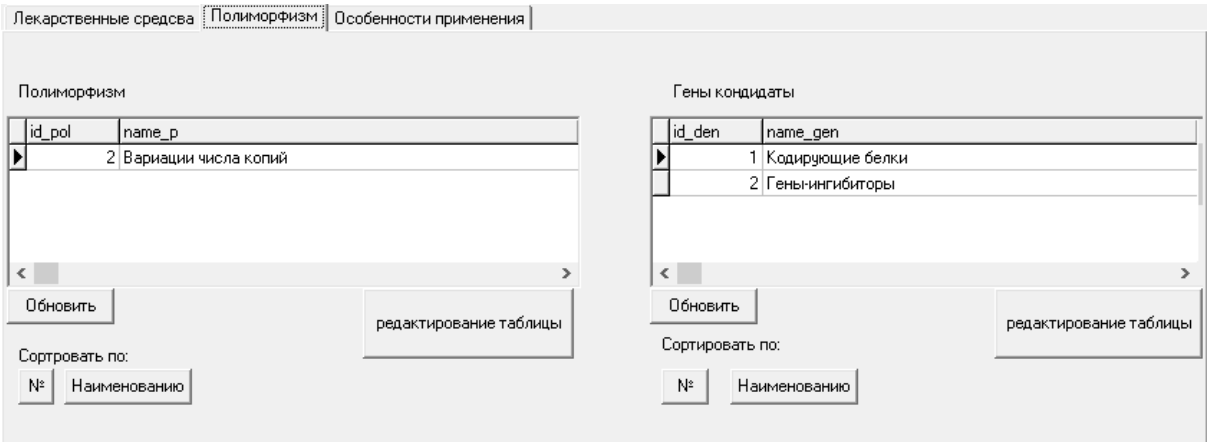


Рис 3.12 Таблица после удаления

# ЗАКЛЮЧЕНИЕ

В данной работе была поставлена цель разработка информационно-справочной системы по интепритации результатов формакогенетического тестирования. Для достижения этой целы нам были поставлены задачи, а именно:

1. Обзор и анализ применения формакогенетического тестирования;
2. Формулировка требований;
3. Проектирвание информационно-справочной системы;
4. Программная реализация информационно-справочной системы;
5. Тестирование системы.

Таким образом, поставленая цель была достигнута

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. А. Хоменко, С. Ададуров Работа с базами данных в C++ BUILDER, Санкт-Петербург, 2006 г

2. А.Д. Хомоненко, В.М. Цыганков, Базы данных учебник, Санкт-Петербург, Корона, 2003

3. Б. Карпов, MS Access, 2000. Справочник, Питер, 2001

4. Боуман Джудит, Эмерсон Сандра, Дарновски Марси. «Практическое руководство по SQL. 3-е издание». Пер с англ. – Киев, Диалектика. 1997.

5. В.И. Загвязинский, Теория обучения современная интерпретация, Москва, 2001

6. Д. А. Сычев Фармакогенетическое тестирование: клиническая интерпритация результатов, Москва, 2011

7. Е.Т. Лильин, В.И. Трубников и М.М. Ванюков, Введение в современную фармакогенетику, М., 1984;

8. И. Соради Основы и педиатрические аспекты фармакогенетики, пер. с венгер., Будапешт, 1984.

9. Научно-методический журнал. Классный руководитель, Москва, 2001

10. О. В. Ромащенко, клинико-фармакологические подходы к персонализации назначения препаратов метаболического ряда при лечении пациентов с ишемической болезнью сердца, Москва, 2015 г.

11. О.Л. Голицына, Н.В. Максимов,Базы данных , Москва, Форум - Инфра-М, 2003

12. Р.Ахаян и др. «Эффективная работа с СУБД», Санкт-Петербург, «Питер», 1997г.

13. С. Бобровский Технологии C++ BUILDER. Разработка приложений для бизнеса. Учебный курс, Питер, 2007

14. С. Симонович, Специальная информатика, Москва, АСТпресс, 2002

15. Т. Карпова, Базы данных: модели, разработка, реализация, Питер, 2001

16. Разработка приложений архитектуры клиент-сервер при помощи SQL. Среды программирования на языке SQL. Эффективность функционирования информационной системы. Структура информационной системы с файл-сервером. Достоинство применения базы данных в ИС.[Электронный ресурс].- http://otherreferats.allbest.ru/programming/c00231654.html

17. Система управления базами данными accsess [Электронный ресурс].- https://infourok.ru/sistema-upravleniya-bazami-dannih-access-1342351.html

18. Свободная СУБД Firebird [Электронный ресурс].- http://bourabai.ru/dbt/servers/firebird.htm

**Приложение 1**

```
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button1Click(TObject *Sender)
{
 Form2->Show();
 }
//---------------------------------------------------------------------------
void __fastcall TForm1::Button6Click(TObject *Sender)
{
ADOTable1->Close();
ADOTable1->Open();
}
//---------------------------------------------------------------------------


void __fastcall TForm1::Button8Click(TObject *Sender)
{
ADOTable1->IndexFieldNames = "Name";
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::Button13Click(TObject *Sender)
{
ADOTable3->Close();
ADOTable3->Open();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button11Click(TObject *Sender)
{
ADOTable2->Close();
ADOTable2->Open();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button14Click(TObject *Sender)
{
ADOTable4->Close();
ADOTable4->Open();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button19Click(TObject *Sender)
{
ADOTable5->Close();
ADOTable5->Open();
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::Button10Click(TObject *Sender)
{
ADOTable2->IndexFieldNames = "Name";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button16Click(TObject *Sender)
{
ADOTable3->IndexFieldNames = "Name";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button18Click(TObject *Sender)
{
ADOTable4->IndexFieldNames = "Name";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button21Click(TObject *Sender)
{
ADOTable5->IndexFieldNames = "Name";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button7Click(TObject *Sender)
{
ADOTable1->IndexFieldNames = "№";
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::Button9Click(TObject *Sender)
{
ADOTable2->IndexFieldNames = "№";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button15Click(TObject *Sender)
{
ADOTable3->IndexFieldNames = "№";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button17Click(TObject *Sender)
{
ADOTable4->IndexFieldNames = "№";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button20Click(TObject *Sender)
{
ADOTable5->IndexFieldNames = "№";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button12Click(TObject *Sender)
{
ADOTable2->IndexFieldNames = "doze";
}
```

```
//---------------------------------------------------------------------------

void __fastcall TForm1::Button22Click(TObject *Sender)
{
ADOTable5->IndexFieldNames = "doze";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button2Click(TObject *Sender)
{
Form3->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button3Click(TObject *Sender)
{
Form4->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button4Click(TObject *Sender)
{
Form5->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::PageControl1Change(TObject *Sender)
{
```

```
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button5Click(TObject *Sender)
{
Form6->Show();
}
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm2 *Form2;
//---------------------------------------------------------------------------
__fastcall TForm2::TForm2(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm2::Button1Click(TObject *Sender)
{
Form1->ADOTable1->Close();
Form1->ADOTable1->Open();
Form1->ADOTable1->Insert();
Form1->ADOTable1->FieldByName("name_g")->AsString = Edit1->Text; //
???: "P_NAME" - ???????? ??????? ? ??.
Form1->ADOTable1->Post();
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm2::Button2Click(TObject *Sender)
{
Form1->ADOTable1->Close();
Form1->ADOTable1->Delete();
Form1->ADOTable1->Parameters->ParamByName("id_group")->AsString =
Edit1->Text;
Form1->ADOTable1->Post();
}
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm3 *Form3;
//---------------------------------------------------------------------------
__fastcall TForm3::TForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm3::Button1Click(TObject *Sender)
{
Form1->ADOTable2->Close();
Form1->ADOTable2->Open();
Form1->ADOTable2->Insert();
Form1->ADOTable2->FieldByName("name_m")->AsString = Edit1->Text; //
???: "P_NAME" - ???????? ??????? ? ??.
Form1->ADOTable2->Post();
}
```

```
//---------------------------------------------------------------------------
void __fastcall TForm3::Button2Click(TObject *Sender)
{
Form1->ADOTable2->Close();
Form1->ADOTable2->Delete();
Form1->ADOTable2->Parameters->ParamByName("id_med")->AsString =
Edit1->Text;
Form1->ADOTable2->Post();
}
//---------------------------------------------------------------------------
```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

_____ _____

*(подпись)* *(Ф.И.О.)*