МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
**«Сибирский государственный аэрокосмический университет
имени академика М. Ф. Решетнева»
(СибГАУ)**

ИНСТИТУТ Информатики и телекоммуникаций

НАПРАВЛЕНИЕ 230200 «Информационные системы»

КАФЕДРА Информатики и вычислительной техники

БАКАЛАВРСКАЯ РАБОТА

Проектирование информационной системы

для медико-социальной экспертизы

Студент М. С. Алехина
ФИО

Руководитель Р. Р. Кагиров
ФИО

Ответственный
за нормоконтроль А. В. Гуменникова
ФИО

Допускается к защите
Зав. кафедрой М. Н. Фаворская
ФИО

«___» _____ 2012 г.

Красноярск 2012 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ....4
1 *WEB*-ТЕХНОЛОГИИ И СУБД....6
1.1 Требования, предъявляемые к интеграции СУБД в среду *Web*....7
1.2 Преимущества и недостатки интеграции СУБД в среду *Web*....8
1.3 Скриптовый язык программирования *PHP*....14
1.4 Анализ существующего программного обеспечения....16
1.5 Выводы по главе....17
2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ....19
2.1 Концептуальное проектирование базы данных....19
2.2 Логическое проектирование базы данных....20
2.3 Физическое проектирование базы данных....23
2.4 Выводы по главе....27
3 РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА....28
3.1 Структура программного продукта....28
3.2 Руководство программиста....29
3.3 Руководство пользователя....31
3.4 Отладка и тестирование....38
3.5 Выводы по главе....38
ЗАКЛЮЧЕНИЕ....39
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ....40

ВВЕДЕНИЕ

В настоящее время ценнейшим ресурсом становится информация. Для успешного выполнения задач необходимо владеть актуальной информацией и быть способным быстро и качественно ее обработать. Это в полной мере относится ко всем отраслям деятельности, в том числе и к медицине.

Федеральный закон «О социальной защите инвалидов Российской Федерации» [6] определяет государственную политику в области социальной защиты инвалидов, целью которой является обеспечение инвалидам равных с другими гражданами возможностей в реализации гражданских, экономических, политических и других прав и свобод, предусмотренных Конституцией РФ. Порядок и условия признания лица инвалидом осуществляется учреждением государственной службы медико-социальной экспертизы (МСЭ) по направлению врачебной комиссии учреждения здравоохранения.

Анализ информации о структуре заболеваний, приводящих к выходу на инвалидность, позволит планировать и своевременно проводить мероприятия по профилактике заболеваний, оздоровлению населения и реабилитации инвалидов. Для учета данных о пациентах стоит использовать базы данных, так как они представляют очень широкие возможности автоматизированной обработки и хранения необходимой информации.

Целью выпускной квалификационной работы является разработка информационной системы учета выхода на инвалидность, предназначенной для оптимизации работы сотрудников бюро МСЭ и повышения качества медико-социальной услуги. Для достижения цели планируется решение следующих **задач**:

- анализ технологии проектирования клиент-серверных приложений на основе *Web*;
- анализ существующего программного обеспечения;
- проектирование БД;

- разработка приложения для работы с информацией в базе данных.

Система должна обладать максимально упрощенным интерфейсом, доступным для освоения даже неподготовленному пользователю и в то же время должна обеспечивать необходимую функциональность.

Работа изложена на 41 странице и состоит из введения, трех глав, заключения и списка использованных источников из 15 наименований. Первая глава содержит обзор технологии СУБД в среде *Web*, анализ существующего программного обеспечения на рынке. Вторая глава описывает процесс разработки и структуру базы данных. В третьей главе изложена структура программного продукта, приведены руководства программиста и пользователя.

1 *WEB*-ТЕХНОЛОГИИ И СУБД

В настоящее время комбинирование технологий *World Wide Web* и технологий баз данных открывает множество новых возможностей создания все более совершенных приложений баз данных. *Web*-среда представляет собой очень привлекательную платформу для разработки и распространения информационно-центрированных (*data-centric*) интерактивных приложений. Благодаря повсеместному распространению *Web*-технологий, созданные для этой среды приложения обладают глобальной доступностью для пользователей и организаций. Поскольку *Web*-архитектура была спроектирована как платформенно-независимая, она обладает значительным потенциалом в отношении существенного сокращения расходов на развертывание приложений и обучение персонала. В настоящее время многие организации быстрыми темпами создают новые и усовершенствуют старые приложения баз данных с целью использования всех преимуществ, достигаемых при выборе *Web*-технологии в качестве стратегической платформы воплощения инновационных решений. Данный подход по существу превращает подобные компании в *Web*-центрированные организации.

Сегодня многие *Web*-сайты построены на основе файловых систем, в которых каждый документ хранится в отдельном файле. Для небольших *Web*-сайтов подобная структура вполне приемлема, но в случае крупных *Web*-сайтов подобное решение существенно усложняет процессы управления данными. Например, достаточно сложно организовать своевременное обновление содержимого сотен и тысяч разных документов, хранящихся в отдельных файлах. Однако еще более трудной задачей является поддержание актуальности связей между этими файлами, особенно если документы создаются разными авторами.

Вторая проблема является следствием того факта, что в настоящее время многие *Web*-сайты содержат в основном динамичную информацию, например, сведения об имеющихся товарах и ценах на них. Управление потоком подобной информации одновременно в базе данных и в отдельных *HTML*-файлах может оказаться невыполнимой задачей, особенно в отношении синхронизации вносимых изменений. По этим и многим другим причинам непосредственный доступ к базам данных из *Web*-среды явился именно тем способом, который получил наиболее широкое распространение при организации управления *Web*-информацией или динамически изменяющимся содержимым *Web*-сайта. Хранение *Web*-информации в базе данных может либо заменять, либо дополнять сохранение ее в обычных файловых структурах.

1.1 Требования, предъявляемые к интеграции СУБД в среду *Web*

Требования, предъявляемые к интеграции СУБД в среду *Web*, отчасти идеалистичны и в настоящее время в полной мере недостижимы:

- возможность защищенного доступа к ценным корпоративным данным;
- способ подключения, не зависящий от данных и разработчика программного обеспечения, предоставляющий необходимую свободу выбора типа СУБД;
- возможность взаимодействия с базой данных независимо от конкретного типа используемого *Web*-браузера или *Web*-сервера;
- наличие такого подключения, которое позволяет извлечь все преимущества всех компонентов СУБД, используемой в организации;
- экономически эффективное решение, допускающее масштабируемость, рост и изменение стратегических направлений, а также способствующее сокращению расходов на разработку и сопровождение приложений;

- поддержка сеансов на основе идентификации пользователей средствами СУБД и приложений;

- приемлемая производительность;

- минимальный уровень администрирования;

- набор высокоуровневых инструментов разработки, позволяющих относительно просто и быстро создавать, внедрять в эксплуатацию и сопровождать новые приложения [12].

1.2 Преимущества и недостатки интеграции СУБД в среду *Web*

Среда *Web*, используемая в качестве платформы для систем с базами данных, может стать основой для инновационных решений в области внутри- и межкорпоративных бизнес-задач. К сожалению, этот подход обладает также определенными недостатками. Но вначале рассмотрим преимущества интеграции СУБД в среду *Web*:

- *Преимущества использования функций СУБД*. *World Wide Web* разрабатывалась с очень малым или вообще без какого-либо использования достижений технологии баз данных. Однако СУБД обладает преимуществами перед традиционной файловой системой. Например, полностью устраняется проблема синхронизации информации в базе данных и *HTML*-файлах, поскольку *HTML*-страницы динамически генерируются на основе информации, извлекаемой из базы данных. В результате существенно упрощается сопровождение системы, а к *HTML*-содержимому сайта могут быть применены все функциональные возможности СУБД (например, средства защиты или поддержки целостности данных).

- *Простота реализации*. В исходном виде язык разметки *HTML* был очень прост для освоения как профессиональными разработчиками, так и обычными конечными пользователями. В определенной степени это остается в силе, если сама *HTML*-страница не перегружена функциональными компонентами. Однако с течением времени язык *HTML* постоянно расширяется за

счет ввода новых или модернизации уже существующих элементов, а потому его исходная простота постепенно теряется.

- *Независимость от платформы*. Привлекательным аспектом создания приложений баз данных на основе *Web*-среды является тот факт, что *Web*-клиенты (или браузеры) обладают независимостью от платформы. Поскольку браузеры имеются практически для всех существующих вычислительных платформ при условии поддержки ими стандартов *HTML*/*Java*, разработчикам не потребуется вносить в приложения изменения для того, чтобы они могли работать с разными операционными системами или различными оконными пользовательскими интерфейсами. В случае же использования традиционных баз данных для переноса приложений на другие платформы потребуется выполнить существенную модификацию (если не полную модернизацию) их клиентских частей. К сожалению, поставщики *Web*-браузеров стали включать в состав своих продуктов специфические компоненты собственной разработки, что приводит к постепенному исчезновению упомянутых выше преимуществ.

- *Графический интерфейс пользователя*. Главной целью использования базы данных является обеспечение доступа к данным, который можно осуществлять с помощью командного интерфейса на основе текстовых меню или с помощью программного интерфейса, подобного тому, который определен в стандарте *SQL*2. Однако эти интерфейсы могут быть достаточно сложными и вызывать затруднения в использовании. С другой стороны, качественный графический пользовательский интерфейс (*Graphic User Interface – GUI*) может существенно упростить и расширить возможности доступа к базе данных. К сожалению, *GUI*-интерфейс сложнее программируется, в большей степени зависит от платформы, а зачастую и от конкретной фирмы-разработчика. В то же время *Web*-браузеры предоставляют широко распространенный и простой в использовании графический пользовательский интерфейс, который можно применять для доступа ко многим типам объектов, включая и базы данных.

Помимо этого, использование широко распространенного типового интерфейса позволяет сократить расходы на обучение конечных пользователей.

- *Стандартизация*. *HTML* фактически является стандартом, который поддерживается всеми существующими *Web*-браузерами, что позволяет читать *HTML*-документы, находящиеся на одном компьютере, с помощью другого компьютера, расположенного в любой точке земного шара, при условии, что он имеет подключение к *Internet* и установленный *Web*-браузер.

- *Межплатформенная поддержка*. *Web*-браузеры теперь доступны практически для любого типа вычислительной платформы. Подобная межплатформенная поддержка позволяет пользователям большинства типов компьютеров осуществлять доступ к одной и той же базе данных из любой точки планеты. Таким образом, информация может распространяться с минимальными затратами времени и сил, а также без необходимости разрешения проблем, связанных с несовместимостью различных типов оборудования, операционных систем и программного обеспечения.

- *Прозрачный сетевой доступ*. Важнейшим достоинством среды *Web* является прозрачность сетевого доступа для пользователя, за исключением необходимости указания *URL*-адреса, что полностью обеспечивается *Web*-браузером и *Web*-сервером. Эта встроенная поддержка сетевого доступа существенно упрощает доступ к базе данных, исключая необходимость приобретения дорогого сетевого программного обеспечения, а также дополнительные сложности согласования различных взаимодействующих платформ.

- *Масштабируемость развертывания*. Традиционная двухуровневая структура клиент/сервер требует создания «толстых» клиентов, недостаточно эффективно реализующих как функции интерфейса пользователя, так и логику самого приложения. И наоборот, решения на основе *Web*-технологии позволяют создать более естественную трехуровневую архитектуру, обеспечивающую масштабируемость системы. Размещая всю функциональность приложения на отдельном сервере и удаляя ее из программы-

клиента, *Web*-технология позволяет сэкономить время и деньги, затрачиваемые на развертывание приложений. В то же время упрощается модернизация и администрирование системы при работе с различными вычислительными платформами, расположенными в нескольких офисах. При наличии сервера приложений доступ к функциям приложения легко осуществить из любого *Web*-сайта, расположенного в любой точке планеты. Возможность глобального доступа к серверной части приложений существенно упрощает создание новых услуг и открытие новых пунктов обслуживания клиентов.

- *Инновационность*. Среда *Web* как *Internet*-платформа позволяет организациям предоставлять новые услуги и находить новых клиентов посредством создания глобально доступных приложений. Такие преимущества ранее были совершенно недоступны для любых решений на основе традиционных централизованных приложений архитектуры клиент/сервер или локальных групповых приложений.

К недостаткам интеграции СУБД в среду *Web* относятся:

- *Недостаточная надежность*. В настоящее время *Internet* является недостаточно надежной и очень медленной коммуникационной средой, т. е. при передаче запроса через *Internet* нет никаких реальных гарантий его доставки (например, просто потому, что сервер может быть выключен). Трудности особенно возрастают в часы пиковой нагрузки, когда сервер очень перегружен или для передачи данных используются перегруженные фрагменты сети. Ненадежность *Internet* – это проблема, для решения которой потребуется достаточно много времени. Наряду с проблемами обеспечения безопасности, ненадежность является еще одной причиной, по которой организации для наиболее критичных приложений предпочитают использовать внутренние сети собственной разработки, а не общедоступную сеть *Internet*.

- *Слабая защищенность*. Безопасность – это один из основных вопросов, которые следует учитывать организации, решившейся сделать свои базы

данных доступными в среде *Web*. В данном случае из-за большого количества потенциально анонимных пользователей становятся очень критичными вопросы идентификации пользователей и безопасной передачи данных.

- *Высокая стоимость*. Вопреки очень распространенному заблуждению, сопровождение нетривиального *Web*-сайта в *Internet* может оказаться очень дорогим, особенно с ростом требований и ожиданий пользователей. Например, согласно отчету фирмы *Forrester Research*, стоимость коммерческого *Web*-сайта может варьироваться от $300 000 до $3 400 000 в зависимости от его предназначения. К тому же предполагается, что эта цена возрастет на 50-200% в течение ближайших двух лет. На самом верху этой шкалы располагаются *Web*-сайты, которые «занимаются» продажей товаров или обработкой транзакций. Причем, 20% затрат уходит на аппаратное и программное обеспечение, 24% – на маркетинговые расходы, а оставшиеся 56% – на разработку содержимого *Web*-сайта. Однако за счет использования улучшенных инструментов и промежуточного коммуникационного уровня можно существенно сократить расходы на его техническую разработку.

- *Трудности с определением масштаба*. *Web*-приложения могут столкнуться с непредсказуемым и потенциально огромным уровнем пиковой нагрузки. Это потребует разработки высокопроизводительной архитектуры серверной части, способной обеспечить необходимый уровень производительности.

- *Ограниченная функциональность языка HTML*. Хотя язык *HTML* предоставляет широко распространенный и простой в использовании интерфейс, его простота означает, что некоторые приложения баз данных с высоким уровнем интерактивности будет очень непросто преобразовать в *Web*-приложения с тем же уровнем дружественности к пользователю. Дополнительная функциональность *Web*-страницы реализуется с помощью языков сценариев (*JavaScript*, *VBScript*), посредством использования компонентов *ActiveX* или средствами языка *Java*. Однако в большинстве случаев подоб-

ный подход оказывается слишком сложным для неопытных конечных пользователей. Кроме того, эти решения связаны с дополнительным снижением производительности (за счет пересылки по сети и выполнения соответствующего кода).

- *Отсутствие запоминания состояния*. Отсутствие функций запоминания состояния в среде *Web* затрудняет управление соединениями с базой данных и выполнение пользовательских транзакций, вызывая необходимость хранения в приложениях дополнительной информации.

- *Недостаточная производительность*. Многие элементы клиентских *Web*-программ сложных приложений баз данных построены с использованием интерпретирующих языков. В результате клиентская часть такой базы данных работает медленнее, чем клиентская часть приложения, использующего обычную базу данных. В частности, все *HTML*-содержимое должно интерпретироваться и отображаться *Web*-браузером; *JavaScript* и *VBScript* – это интерпретируемые языки сценариев, предназначенные для расширения языка *HTML* некоторыми программными конструкциями; любой *Java*-аплет компилируется в байт-код, который пересылается по сети, а затем интерпретируется браузером. Для критических по времени выполнения приложений накладные расходы на интерпретацию многочисленных элементов могут оказаться неприемлемыми. Тем не менее, существует достаточно много приложений, для которых скорость их выполнения не так важна.

- *Несовершенство инструментов разработки*. Разработчики приложений баз данных для среды *Web* быстро осознали тот факт, что имеющиеся инструменты разработки весьма несовершенны. Еще совсем недавно разработчики *Internet*-приложений использовали языки программирования первого поколения со средой разработки, незначительно отличающейся от текстового редактора. Это очень серьезный недостаток *Internet*-программирования остается актуальным и в наши дни. К тому же не накоплен необходимый

опыт в области интеграции баз данных в среду *Web*, сравнимый с опытом работы с традиционными приложениями баз данных, не использующих *Web*-технологий. Однако со временем этот недостаток будет устранен.

Многие перечисленные выше преимущества и недостатки являются временными. Некоторые преимущества когда-нибудь исчезнут (например, язык *HTML* постепенно становится все более сложным). Исчезнут и некоторые недостатки (например, *Web*-технологии станут более совершенными и понятными). Это лишь подчеркивает изменчивость той среды, в которой приходится работать при создании приложений баз данных на основе *Web*-технологии [12].

1.3 Скриптовый язык программирования *PHP*

Для интеграции СУБД в среду *Web* можно использовать язык *PHP* (*Hypertext Preprocessor* – «препроцессор гипертекста») – скриптовый язык программирования общего назначения, интенсивно применяемый для разработки *Web*-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков программирования, применяющихся для создания динамических *Web*-сайтов.

В области программирования для сети Интернет *PHP* – один из популярных скриптовых языков (наряду с *JSP, Perl* и языками, используемыми в *ASP.NET*) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии *PHP*.

Основные области, где используется *PHP*:

- создание скриптов для выполнения на стороне сервера. *PHP* наиболее широко используется именно таким образом. Все, что вам понадобится,

это парсер *PHP* (в виде программы *CGI* или серверного модуля), *Web*-сервер и браузер. Чтобы просматривать результаты выполнения *PHP*-скриптов в браузере, нужен работающий *Web*-сервер и установленный *PHP*;

- создание скриптов для выполнения в командной строке. Можно создать *PHP*-скрипт, способный запускаться вне зависимости от *Web*-сервера и браузера. Все, что для этого потребуется – парсер *PHP*. Такой способ использования *PHP* идеально подходит для скриптов, которые должны выполняться регулярно, например, с помощью *cron* (на платформах **nix* или *Linux*) или с помощью планировщика задач (*Task Scheduler*) на платформах *Windows*. Эти скрипты также могут быть использованы в задачах простой обработки текстов.

Популярность в области построения *Web*-сайтов определяется наличием большого набора встроенных средств для разработки *Web*-приложений. Основные из них:

- автоматическое извлечение *POST* и *GET*-параметров, а также переменных окружения *Web*-сервера в предопределённые массивы;
- взаимодействие с большим количеством различных систем управления базами данных (*MySQL, PostgreSQL, Oracle, Microsoft SQL Server, Sybase, ODBC, IBM DB2, dBase, Firebird /InterBase, Paradox File Access* и др.);
- автоматизированная отправка *HTTP*-заголовков;
- работа с *HTTP*-авторизацией;
- работа с *cookies* и сессиями;
- работа с локальными и удалёнными файлами, сокетами,
- обработка файлов, загружаемых на сервер.

PHP доступен для большинства операционных систем, включая *Linux*, многие модификации *Unix* (такие, как *HP-UX, Solaris* и *OpenBSD*), *Microsoft Windows*, *Mac OS X, RISC OS*, и многих других. Также в *PHP* включена поддержка большинства современных *Web*-серверов, таких, как *Apache, Microsoft Internet Information Server, Personal Web Server,* серверов *Netscape* и

iPlanet, сервера *Oreilly Website Pro*, *Caudium*, *Xitami*, *OmniHTTPd* и многих других. Для большинства серверов *PHP* поставляется в качестве модуля, для других, поддерживающих стандарт *CGI*, *PHP* может функционировать в качестве процессора *CGI*.

В настоящее время *PHP* используется сотнями тысяч разработчиков. Согласно рейтингу корпорации *TIOBE*, базирующемся на данных поисковых систем, в апреле 2011 года *PHP* находился на 5 месте среди языков программирования. К крупнейшим сайтам, использующим *PHP*, относятся *Facebook*, ВКонтакте, *Wikipedia* и др. [12].

1.4 Анализ существующего программного обеспечения

Компания СП.АРМ – это ведущая российская компания, основанная в 1988 году, специализирующаяся на разработке и внедрении информационных систем, создании корпоративных приложений, интеграции программного обеспечения.

В числе приоритетных направлений деятельности компании – разработка, развитие и внедрение информационных систем в медико-социальной сфере [9].

АИС «МСЭ» (Автоматизированная информационная система Медико-социальной экспертизы) компании СП.АРМ создана для поддержки работы федеральной службы МСЭ, формирования необходимой информации в области МСЭ в управляющих структурах государства, для специалистов, работающих с инвалидами и для самих инвалидов.

АИС МСЭ – это единая инфраструктура, обеспечивающая санкционированный доступ ко всем информационным ресурсам по проблемам инвалидности.

АИС «МСЭ» обеспечивает:

- создание и ведение базы данных на прошедших освидетельствование в автоматизированных составах службы МСЭ;

- проверку правильности ввода данных;
- подготовку основных документов в процессе освидетельствования и по его результатам в автоматизированных составах службы МСЭ;
- приглашение на освидетельствование;
- поиск информации по имеющимся признакам;
- выполнение произвольных статистических запросов;
- формирование Актов освидетельствования.

Информация о всех инвалидах, проходящих освидетельствование в составе МСЭ хранится в единой базе данных и доступна на любом рабочем месте специалистов в соответствии с установленным для них уровнем доступа [8].

Программный продукт «*Специалист по реабилитации*» – является подсистемой автоматизированной информационной системы «Медико-социальная экспертиза и реабилитация инвалидов» («ФРИ АС») и обеспечивает информационную поддержку работы специалиста по реабилитации состава работы бюро МСЭ. В процессе работы на граждан, прошедших освидетельствование в данном бюро, в БД добавляются данные, которые находятся в компетенции специалиста по реабилитации. Введенная информация накапливается в БД и позволяет формировать статистические отчеты.

Система позволяет:

1) хранить следующие данные об освидетельствуемом:
   - анкетные данные (фамилия, имя, отчество, дата рождения, пол);
   - общие сведения (адреса прописки и регистрации, социальные сведения, сведения об образовании, обучении, месте работы, семейном положении, жилищных условиях, источниках дохода);
   - данные об инвалидности до освидетельствования в данном бюро МСЭ;
   - данные акта освидетельствования;
   - данные акта на приобретение транспорта.
2) производить просмотр и редактирование данных;

3) формировать произвольные запросы по поиску граждан по произвольным параметрам;

4) печатать, сформированные по запросам списки;

5) формировать и печатать статистические отчеты [8].

1.5 Выводы по главе

Комбинирование технологий *World Wide Web* и технологий баз данных открывает множество новых возможностей создания все более совершенных приложений баз данных. Благодаря повсеместному распространению *Web*-технологий, созданные для этой среды приложения обладают глобальной доступностью для пользователей и организаций. Поскольку *Web*-архитектура была спроектирована как платформенно-независимая, она обладает значительным потенциалом в отношении существенного сокращения расходов на развертывание приложений и обучение персонала.

Непосредственный доступ к базам данных из *Web*-среды получил наиболее широкое распространение при организации управления *Web*-информацией или динамически изменяющимся содержимым *Web*-сайта. Хранение *Web*-информации в базе данных может либо заменять, либо дополнять сохранение ее в обычных файловых структурах.

Одна из технологий интеграции СУБД в среду *Web* – язык *PHP* (*Hypertext Preprocessor* – «препроцессор гипертекста») – скриптовый язык программирования общего назначения, интенсивно применяемый для разработки *Web*-приложений благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии *PHP.*

2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

2.1 Концептуальное проектирование базы данных

Концептуальное проектирование базы данных – это построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

Цель концептуального проектирования – создание концептуальной модели данных на основе представлений о предметной области каждого отдельного типа пользователей. Концептуальная модель представляет собой описание основных сущностей (таблиц) и связей между ними без учета принятой модели БД и синтаксиса целевой СУБД.

Концептуальная модель данных включает определение типов сущностей, определение типов связей, определение атрибутов и доменов атрибутов, определение атрибутов, являющихся потенциальными и первичными ключами, предварительное создание диаграммы «сущность-связь».

Сведения о типах сущностей разработанной системы сведены в таблицу 1.

Таблица 1 – Характеристика типов сущностей

Имя сущности	**Псевдонимы**	**Описание**	**Особенности использования**
Пациенты	*pacient*	Содержит общую информацию о пациентах	Операционная таблица
Направления	*dispatch*	Содержит информацию о направлениях на МСЭ	Операционная таблица
Результаты	*result*	Содержит информацию о результатах МСЭ	Операционная таблица
Программы реабилитации	*ipr*	Содержит информацию о программах реабилитации пациентов	Операционная таблица
Пользователи	*users*	Содержит информацию о пользователя	Операционная таблица

Продолжение таблицы 1

Категории населения	*categ*	Содержит список категорий населения	Справочник
Заболевания	*diag*	Содержит список заболеваний	Справочник
Группы инвалидности	*group*	Содержит список групп инвалидности	Справочник
Степени нетрудоспо-собности	*ratio_srtict*	Содержит список степеней нетрудоспособности	Справочник
Населенные пункты	*list_city*	Содержит список населенных пунктов	Справочник
Улицы	*list_street*	Содержит список улиц	Справочник
Медицинские средства реабилитации	*msr*	Содержит список МСР	Справочник
Технические средства реабилитации	*tsr*	Содержит список ТСР	Справочник
Проф. средства реабилитации	*psr*	Содержит список ПСР	Справочник
Соц. средства реабилитации	*ssr*	Содержит список ССР	Справочник
Население	*population*	Содержит информацию о численности населения	Справочник

2.2 Логическое проектирование базы данных

Логическое проектирование базы данных представляет собой процесс конструирования модели информационной структуры организации, выполняемый в соответствии с выбранной схемой организации информации (например, реляционной). Однако создаваемая логическая модель не зависит от особенностей конкретной СУБД и физических условий реализации.

При проектировании базы данных в реляционной СУБД основной целью разработки логической модели данных является создание точного представления данных, связей между ними и требуемых ограничений. Это осуществляется группировкой атрибутов в отношения так, чтобы минимизировать избыточность данных и таким образом сократить объем памяти, необходимый для физического хранения отношений, представленных в виде таблиц.

Сведения об атрибутах сведены в таблицу 2.

Таблица 2 – Атрибуты и домены атрибутов

Сущность	**Атрибуты**	**Домены**
Пациенты	*id* пациента	числовой
	Имя	символьный
	Отчество	символьный
	Фамилия	символьный
	Пол	символьный
	Дата рождения	дата
	Населенный пункт	символьный
	Улица	символьный
	Дом	числовой
	Корпус	символьный
	Квартира	числовой
Направления	*id* направления	числовой
	id пациента	
	Категория населения	числовой
	Диагноз	числовой
	Дата направления	дата
	Примечание	символьный
Результаты	*id* направления	числовой
	Дата результата	дата
	Группа инвалидности	числовой
	Степень нетрудоспособности	числовой
	Срок действия группы	числовой
Программы реабилитации	*id* пациента	числовой
	МСР	числовой
	МСР	числовой
	МСР	числовой
	ПСР	числовой
	ПСР	числовой
	ТСР	числовой
	ТСР	числовой
	ССР	числовой
	ССР	числовой
Пользователи	*id* пользователя	числовой
	Имя пользователя	символьный
	Логин	символьный
	Пароль	символьный
	Уровень доступа	числовой
Населенные пункты	*id* населенного пункта	числовой
	Населенный пункт	символьный
Улицы	*id* улицы	числовой
	Тип улицы	символьный
	Название улицы	символьный
Категории населения	*id* категории	числовой
	Категория	символьный
Заболевания	*id* диагноза	числовой
	Заболевание	символьный

Продолжение таблицы 2

Степени нетрудоспособности	*id* степени	числовой
	Степень	символьный
Группы инвалидности	*id* группы	числовой
	Группа	символьный
Медицинские средства реабилитации	*id* МСР	числовой
	МСР	символьный
Профессиональные средства реабилитации	*id* ПСР	числовой
	ПСР	символьный
Технические средства реабилитации	*id* ТСР	числовой
	ТСР	символьный
Социальные средства реабилитации	*id* ССР	числовой
	ССР	символьный
Население	*id*	числовой
	Год переписи	числовой
	Население района	числовой
	Население района трудоспособное	числовой
	Население Богучан	числовой
	Население Богучан трудоспособное	числовой

На рисунке 1 представлена логическая структура базы данных.

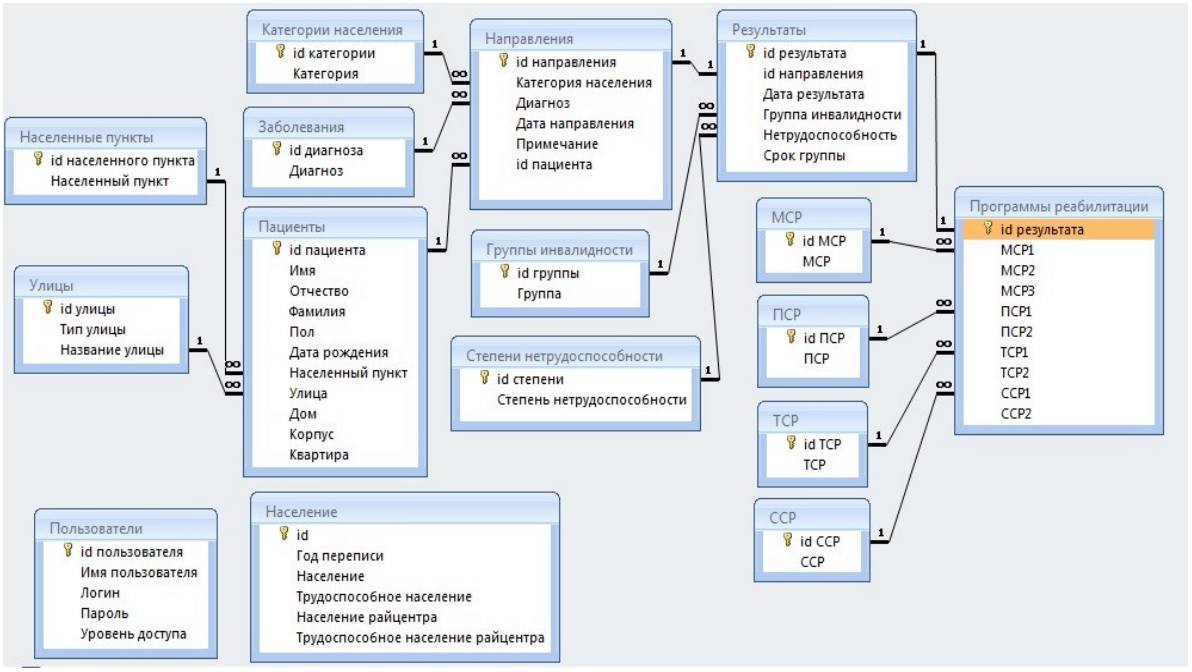

Рисунок 1 – Логическая структура базы данных

2.3 Физическое проектирование базы данных

Физическое проектирование базы данных – это процесс создания описания конкретной реализации базы данных, размещаемой во вторичной памяти. Предусматривает описание структуры хранения данных и методов доступа, предназначенных для осуществления наиболее эффективного доступа к информации.

Фаза физического проектирования базы данных предусматривает принятие разработчиком окончательного решения о способах реализации создаваемой базы. Поэтому физическое проектирование обязательно производится с учетом всех особенностей используемой СУБД.

Для создания базы данных выбрана СУБД *PostgreSQL* – свободная объектно-реляционная система управления базами данных. Существует в реализациях для следующих платформ: *Linux*, *Solaris*/*OpenSolaris*, *Win*32, *Win x*86-64, *Mac OS X*, *FreeBSD*, *QNX* 4.25, *QNX* 6. *PostgreSQL* базируется на языке SQL и поддерживает многие из возможностей стандарта *SQL*: 2003 (*ISO*/*IEC* 9075).

Сильными сторонами PostgreSQL считаются:

- поддержка БД практически неограниченного размера;
- мощные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются *PL*/*pgSQL*, *PL*/*Perl*, *PL*/*Python* и *PL*/*Tcl*; дополнительно можно использовать *PL*/*Java*, *PL*/*PHP*, *PL*/*Py*, *PL*/*R*, *PL*/*Ruby*, *PL*/*Scheme* и *PL*/*sh*, а также имеется поддержка загрузки *C*-совместимых модулей;
- наследование;
- легкая расширяемость [6].

Структура полей таблиц спроектированной базы данных подробно рассмотрена в таблице 3.

Таблица 3 – Структура таблиц базы

Имя поля	Содержание поля	Тип данных	Тип данных *PostgreSQL*	Примечание
Населенные пункты (*list_city*)				
id_city	*id* населенного пункта	числовой	*integer*	первичный ключ
n_city	населенный пункт	символьный	*varchar*	
Названия улиц (*list_street*)				
id_str	*id* улицы	числовой	*integer*	первичный ключ
str_type	тип улицы	символьный	*varchar*	
street	название улицы	символьный	*varchar*	
Заболевания (*diag*)				
id_diag	*id* диагноза	числовой	*integer*	первичный ключ
diagnosis	диагноз	символьный	*varchar*	
Группы инвалидности (*groups*)				
id_res	*id* группы	числовой	*integer*	первичный ключ
grp	группа инвалидности	символьный	*varchar*	
Категории населения (*categ*)				
id_cat	*id* категории	числовой	*integer*	первичный ключ
category	категория населения	символьный	*varchar*	
Степени нетрудоспособности (*ratio_strict*)				
ratio	*id* степени	числовой	*integer*	первичный ключ
ratio_descr	степень нетрудоспособности	символьный	*varchar*	
Медицинские средства реабилитации (*msr*)				
id_msr	*id*	числовой	*integer*	первичный ключ
msr	МСР	символьный	*varchar*	
Профессиональные средства реабилитации (*psr*)				
id_psr	*id*	числовой	*integer*	первичный ключ
psr	ПСР	символьный	*varchar*	
Технические средства реабилитации (*tsr*)				
id_tsr	*id*	числовой	*integer*	первичный ключ
tsr	ТСР	символьный	*varchar*	
Социальные средства реабилитации (*ssr*)				
id_ssr	*id*	числовой	*integer*	первичный ключ
ssr	ССР	символьный	*varchar*	

Продолжение таблицы 3

Население (*population*)				
id_pop	*id*	числовой	*integer*	первичный ключ
years	год переписи	числовой	*integer*	
pop_rn	население района	числовой	*integer*	
pop_rn_t	население района трудоспособное	числовой	*integer*	
pop_bg	население Богучан	числовой	*integer*	
pop_bg_t	население Богучан трудоспособное	числовой	*integer*	
Пациенты (*pacient*)				
id_p	*id* пациента	числовой	*integer*	первичный ключ
s_name	имя	символьный	*varchar*	
f_name	отчество	символьный	*varchar*	
l_name	фамилия	символьный	*varchar*	
sex	пол	символьный	*varchar*	
birth_day	дата рождения	дата	*date*	
city	населенный пункт	символьный	*varchar*	
street	улица	символьный	*varchar*	
build	дом	числовой	*integer*	
korp	корпус	символьный	*char*	
flat	квартира	числовой	*integer*	
Направления (*dispatch*)				
id_p	*id* направления	числовой	*integer*	первичный ключ
id_kat	категория населения	числовой	*integer*	внешний ключ
id_diag	диагноз	числовой	*integer*	внешний ключ
date_disp	дата направления	дата	*date*	
rem	примечание	символьный	*char*	
Результаты (*result*)				
id_p	*id* пациента	числовой	*integer*	первичный ключ
date_res	дата результата	дата	*date*	
id_res	группа инвалидности	числовой	*integer*	внешний ключ
ratio	степень нетрудоспособности	числовой	*integer*	внешний ключ
date_exrip	срок действия группы	числовой	*integer*	

Продолжение таблицы 3

Программы реабилитации (*ipr*)				
id_p	*id* пациента	числовой	*integer*	первичный ключ
*msr*1	МСР	числовой	*integer*	
*msr*2	МСР	числовой	*integer*	
*msr*3	МСР	числовой	*integer*	
*psr*1	ПСР	числовой	*integer*	
*psr*2	ПСР	числовой	*integer*	
*ssr*1	ССР	числовой	*integer*	
*ssr*2	ССР	числовой	*integer*	
*tsr*1	ТСР	числовой	*integer*	
*tsr*2	ТСР	числовой	*integer*	
*tsr*3	ТСР	числовой	*integer*	
Пользователи (*users*)				
id	*id* пользователя	числовой	*integer*	первичный ключ
u_name	логин	символьный	*varchar*	
u_pass	пароль	символьный	*varchar*	
u_level	уровень доступа	числовой	*integer*	
sname	имя	символьный	*varchar*	
fname	отчество	символьный	*varchar*	
lname	фамилия	символьный	*varchar*	

Кроме того, для удобства обработки созданы три вспомогательные таблицы: *list_disp*, *list_r* и *list_ipr*, отображающие информацию о направленных на МСЭ, результаты МСЭ и индивидуальные программы реабилитации соответственно. Эти таблицы заполняются данными из нескольких таблиц (рабочих и справочных) при помощи триггеров.

Физическая структура базы представлена на рисунке 2.

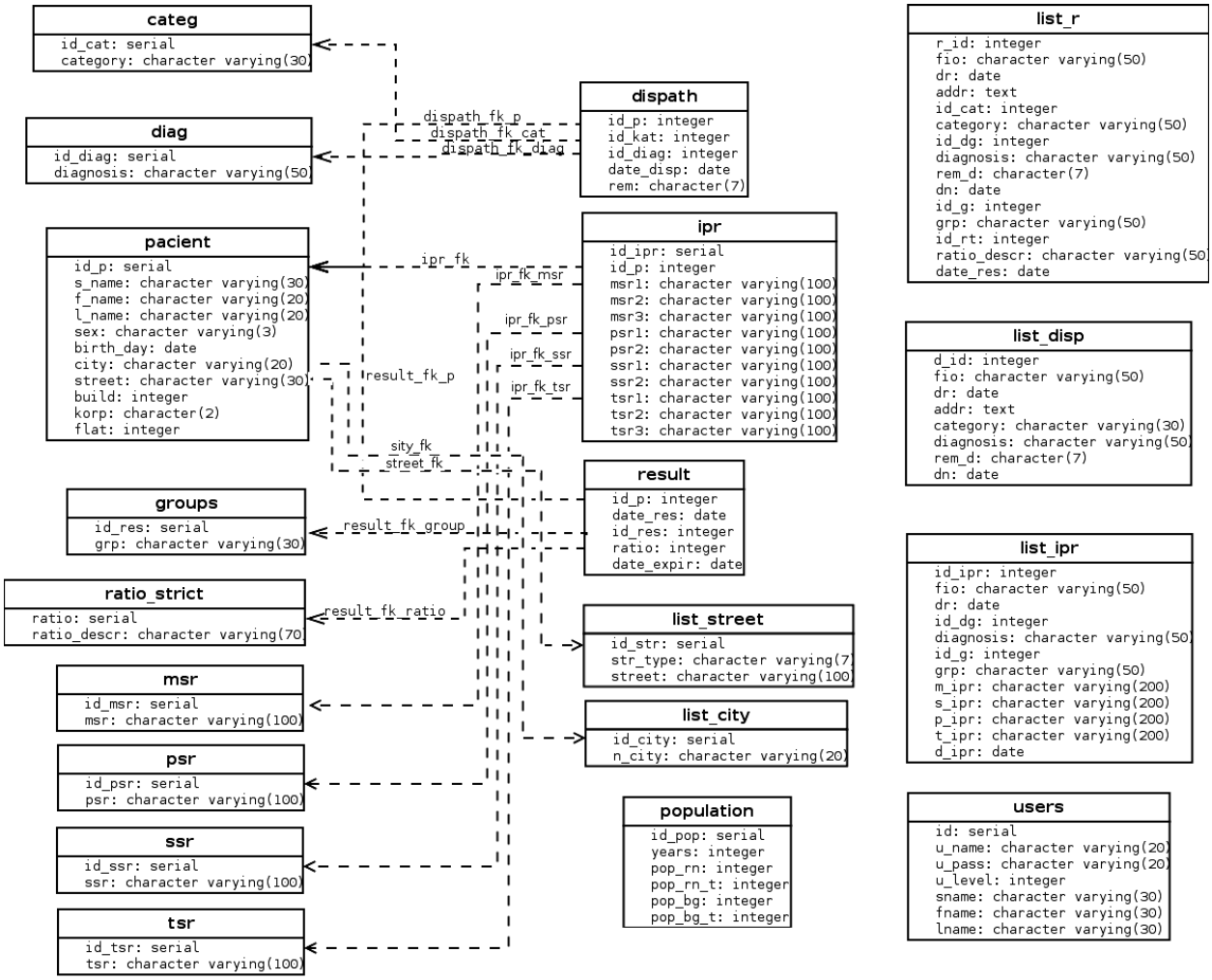

Рисунок 2 – Физическая структура базы данных

2.4 Выводы по главе

Во второй главе был описан процесс проектирования базы данных, концептуальное, логическое и физическое проектирование, обоснован выбор СУБД. Приведена структурная схема базы данных.

3 РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

3.1 Структура программного продукта

Программный продукт в технологическом плане представляет собой информационную систему, построенную на основе СУБД и *Web*-технологий. Это позволит в качестве клиентского места использовать *Web*-браузер, а при необходимости – организовать удаленный доступ к базе данных через *Internet*/*Intranet*.

Основная функция программы – возможность создания и последующего сопровождения базы данных лиц, прошедших медико-социальную экспертизу с учетом установленной группы инвалидности с возможностью отслеживания как индивидуальных показателей по каждому пациенту, так и показателей статистической отчетности в целом.

В содержательном плане программный продукт должен выполнять следующие функции:

- просмотр информации: выводимая на экран информация должна иметь вид таблицы;
- добавление новой информации к уже имеющейся в базе данных;
- редактирование введенной информации;
- удаление ошибочно введенной информации;
- поиск информации по имени, адресу, диагнозу, группе инвалидности, категории населения;
- анализ информации и создание отчетов за любой период времени,
- составление индивидуальных программ реабилитации.

Предусмотрено два вида пользователей: администратор и пользователь (врач). На рисунке 3 представлена диаграмма вариантов использования программы.

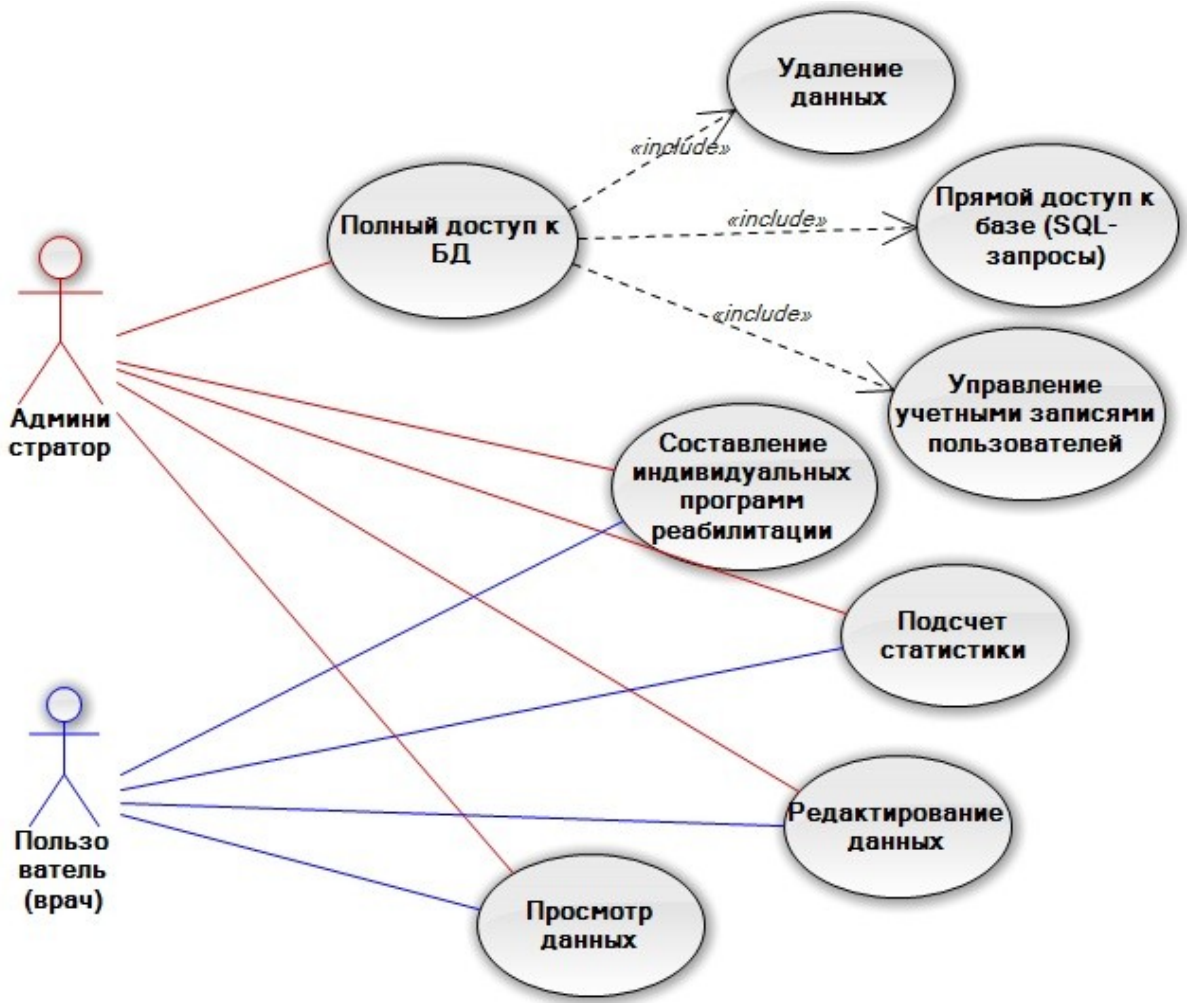

Рисунок 3 – Диаграмма вариантов использования

3.2 Руководство программиста

Программный продукт представляет собой информационную систему, построенную на основе СУБД и программных модулей, написанных на языке *PHP*. Для работы системы необходимы установленные в операционной системе СУБД *PostgreSQL*, *PHP*5 и *Web*-сервер *Apache*. Наличие *PHP* именно пятой версии критично – поддержка полноценной работы с классами появилась только в *PHP*5.

Системные требования:

- *Оперативная память*: объем, достаточный для нормальной работы *Web*-сервера и сервера баз данных. Для работы в среде *Linux* достаточен объем ОЗУ 256 Мб.

- *Объем дискового пространства*: приблизительно 100-150 Мб с учетом СУБД *PostgreSQL* (дополнительный требуемый объем зависит от объема электронных документов).

- *Операционная система*: хотя информационная система разрабатывалась и писалась в среде *Linux*, она является кроссплатформенной и с одинаковым успехом может работать в различных ОС, таких как: *Windows*, *MacOS*, *FreeBSD*, *Solaris*, *Qnx* и др.

Автоматизированная информационная система состоит из базы данных и программных модулей, реализованных в виде файлов **.php*, расположенных в рабочем каталоге *Web*-сервера. Ниже приведено краткое описание модулей программы:

1) *index.php* – основной управляющий файл, «сердце» системы. Именно из него вызываются почти все функции и процедуры, он перенаправляет неавторизованных пользователей на страницу аутентификации, проверяет уровень доступа (привилегии) пользователей.

2) *auth.php* – в этом файле реализован механизм аутентификации и распределения прав пользователей. Он содержит функцию аутентификации, проверяющую правильность введенных имени и пароля и устанавливающую *cookies* браузера, на основе которых определяется уровень доступа пользователей. Так же он содержит функции закрытия сессии, и извлечения значения уровня доступа из *cookies*.

3) *display.php* – шаблоны *HTML*. Файл содержит набор функций, формирующих *HTML*-страницы, а также обеспечивающих отображение данных, полученных из БД в табличной форме.

4) *dbsub.php* – здесь находятся функции, обеспечивающие соединение с БД.

5) *sub.php* – содержит функции для просмотра БД и поиска. Также здесь располагаются некоторые вспомогательные функции.

6) *dbedit.php* – содержит функции для редактирования из занесения данных в БД.

7) *reports.php* – содержит функции формирования статистических отчетов.

8) *classes.php* – содержит объявления классов.

9) *config.php* – содержит некоторые конфигурационные параметры для соединения с БД.

Наиболее часто используемые операции: извлечение, вставка, удаление и модификация информации в базе данных – реализованы с помощью классов, объявленных в модуле *classes.php*. Все многообразие запросов к БД осуществляется с помощью четырех классов:

- Класс *DB_view* реализует запросы к БД на извлечение информации, обрабатывает полученные от *PostgreSQL* данные и выводит их пользователю в виде таблиц. Экземпляры класса вызываются из модулей *dbedit.php*, *sub.php* *и reports.php*.

- Класс *DB_list* реализует запросы к справочным таблицам БД и заполняет полученными от сервера *SQL* данными поля тега *HTML* <*select*> в пользовательских формах на вставку данных в БД. Экземпляры класса вызываются из модуля *display.php*.

- Класс *DB_insert* реализует вставку данных в БД. Вызывается из модуля *dbedit.php*.

- Класс *DB_insert_p* является расширением класса *DB_insert* (дочерний класс) и реализует *UPDATE*-запросы к БД на модификацию данных. Вызывается из модуля *dbedit.php*.

Дистрибутив программы представляет собой архив, содержащий *php*-файлы, копируемые в рабочий каталог *Web*-сервера и скрипт *install_db.sql*, обеспечивающий создание структуры БД на *PostgreSQL*.

3.3 Руководство пользователя

Программный продукт предназначен для использования в клинико-экспертной службе медицинских учреждений для ввода, хранения и обработки

информации о пациентах, направленных на МСЭ с целью определения группы инвалидности.

Интерфейс программы максимально упрощен, что позволяет быстро освоить работу с программой пользователями любой квалификации.

Для работы с программой необходим любой *Web*-браузер, например *Opera*, *Firefox* или *Internet Explorer.*

В начале работы на запрос программы нужно ввести логин и пароль для входа, полученные у администратора (Рисунок 4).



Рисунок 4 – Запрос авторизации

После успешной авторизации отобразится главное меню программы (Рисунок 5).

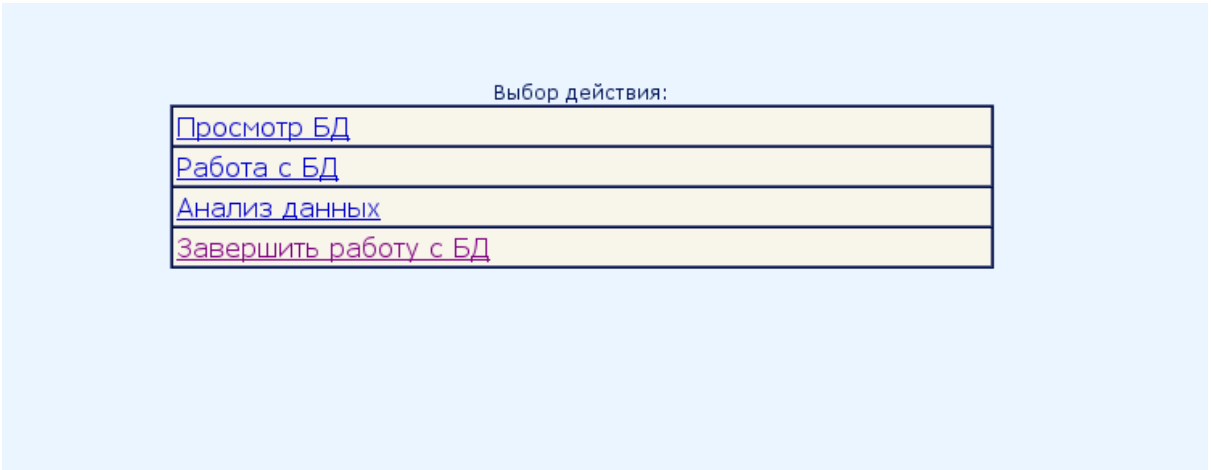

Рисунок 5 – Главное меню

При щелчке на пункте меню «Просмотр БД» откроется страница просмотра информации о пациентах, хранящихся в БД.

На этой странице отображается информация о всех пациентах, направленных на МСЭ, кнопки расположенные в правом верхнем углу позволяют переключиться на отображение информации о результатах МСЭ, о пациентах направленных на МСЭ, но не прошедших ее, либо вернуться в главное меню. Здесь же можно осуществлять поиск по различным критериям и управлять сортировкой найденной информации. Обеспечивается «неточный» поиск записей: запись считается удовлетворяющей условию поиска, если она содержит часть поискового контекста, например, по запросу на поиск «Ив» будут выданы записи со значениями «Иванов», «Ивашкин» и т.п.

Кнопки «Результаты МСЭ» и «Без результатов МСЭ» включают отображение информации о пациентах, прошедших МСЭ и получивших группу инвалидности и о пациентах, направленных, но не прошедших МСЭ (Рисунок 6).



Рисунок 6 – Просмотр БД

Кнопка «Работа с БД» позволяет переключиться в режим ввода данных в БД. В этот режим можно войти также по щелчку на пункте «Работа с БД» в главном меню. При этом отображается подменю работы с БД (Рисунок 7).

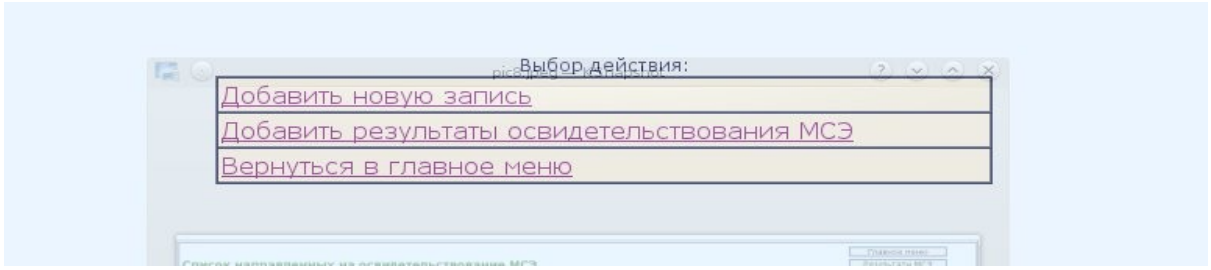

Рисунок 7 – Подменю работы с БД

Пункт «Добавить новую запись» открывает страницу для ввода информации о новых пациентах (Рисунок 8).

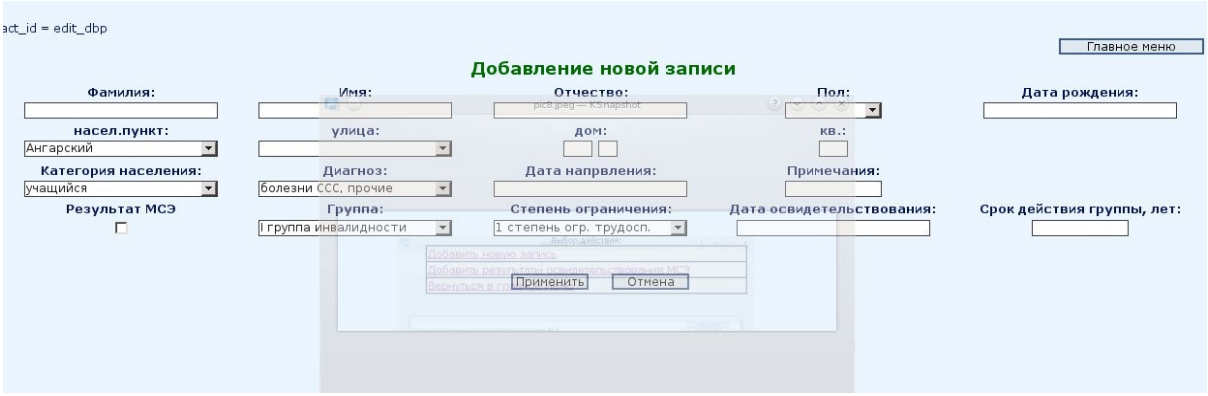

Рисунок 8 – Добавление новой записи о пациенте

В поля формы вводятся данные о пациенте, в выпадающих списках можно выбрать необходимые данные из справочника. Программа не позволяет дважды ввести одного и того же пациента, в этом случае выведется предупреждение.

Если на момент ввода данных известны результаты МСЭ, для их ввода нужно отметить галочку «Результат МСЭ» и ввести необходимые данные.

Щелчок на кнопке «Применить» заносит информацию в БД, при этом в нижней части окна отображается информация о только что введенном пациенте.

Кнопка «Отмена» очищает все поля формы.

Пункт «Добавить результаты освидетельствования МСЭ» открывает страницу для ввода информации о результатах МСЭ для пациентов, уже существующих в БД (Рисунок 9).

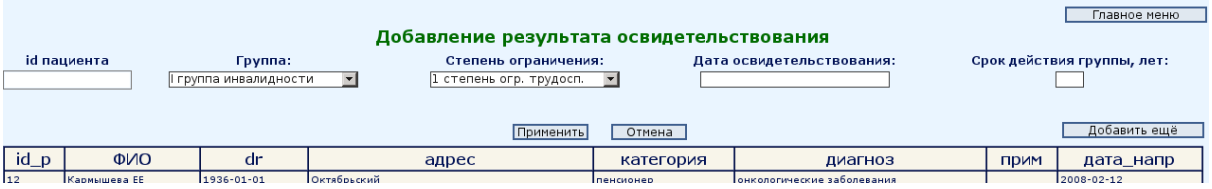

Рисунок 9 – Добавление результата освидетельствования

В нижней части окна отображаются записи о пациентах, не имеющих результата МСЭ. Для добавления информации о результатах необходимо определить *id* пациента (первая колонка в таблице) и ввести его в поле «*id* пациента», ввести данные о группе инвалидности, степени ограничения трудоспособности, дате заключения МСЭ, сроке действия группы и щелкнуть на кнопке «Применить». Данные запишутся в БД, и в нижней части окна отобразится только что введенная запись. Для ввода следующей записи нужно щелкнуть по кнопке «Добавить еще».

Пункт главного меню «Анализ данных» открывает страницу, на которой можно сформировать различные сводные отчеты на основе анализа информации из БД за произвольные календарные периоды (Рисунок 10).

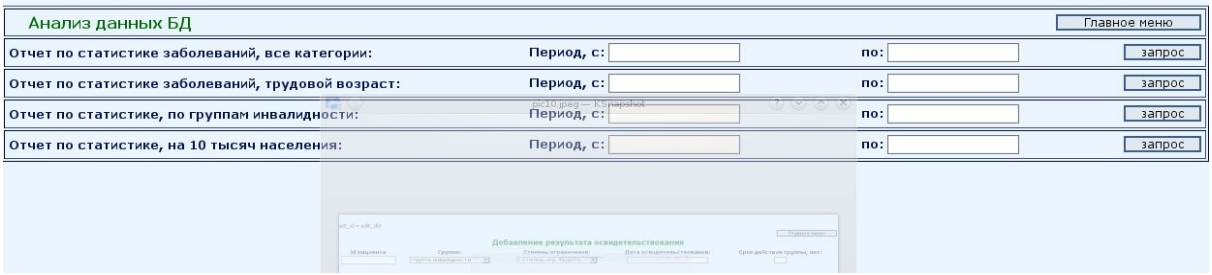

Рисунок 10 – Страница анализа данных

Для получения отчета в соответствующих полях нужно определить начальную и конечную даты периода, за который будут анализироваться данные, и щелкнуть по кнопке «Запрос».

Помимо описанных выше элементов интерфейса, для пользователя с уровнем доступа «администратор», в главном меню добавляется пункт «Полный доступ к БД» (Рисунок 11).

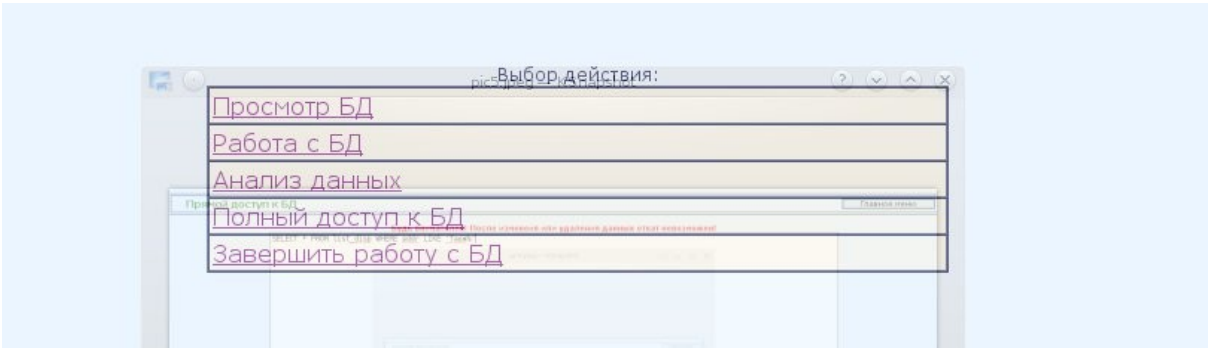

Рисунок 11 – Главное меню пользователя с уровнем доступа «администратор»

При выборе полного доступа к базе данных открывается подменю (Рисунок 12).

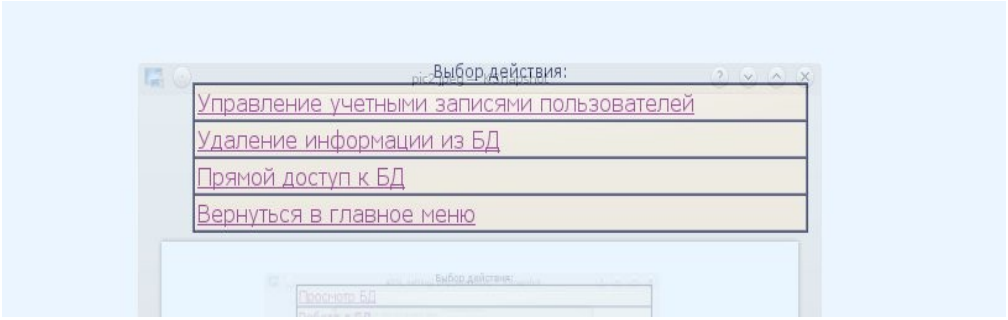

Рисунок 12 – Административные функции системы

На страничке управления учетными записями пользователей можно добавить нового пользователя, удалить ученую запись и изменить пароль пользователя (Рисунок 13).

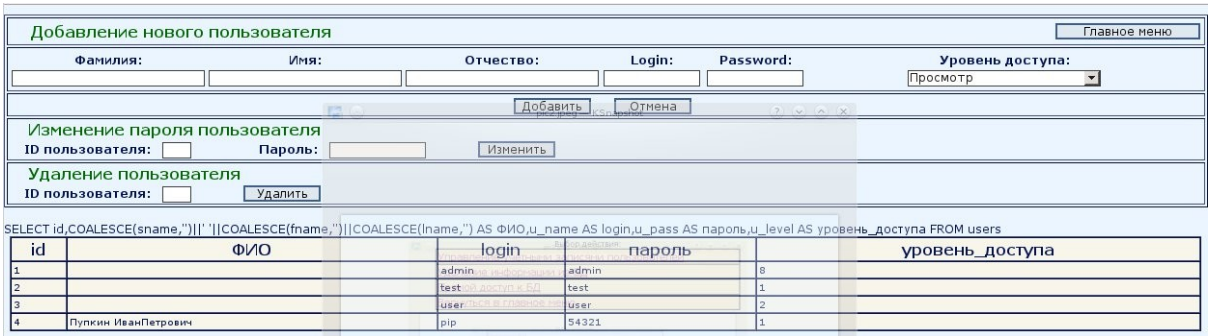

Рисунок 13 – Управление пользователями системы

На странице «Удаление информации из БД» администратор может удалить ошибочно введенную пользователями информацию (Рисунок 14).



Рисунок 14 – Удаление записей из БД

Здесь же можно осуществлять поиск с различными критериями по пациентам.

И, наконец, самый мощный инструмент администратора – «Прямой доступ к БД» (Рисунок 15).

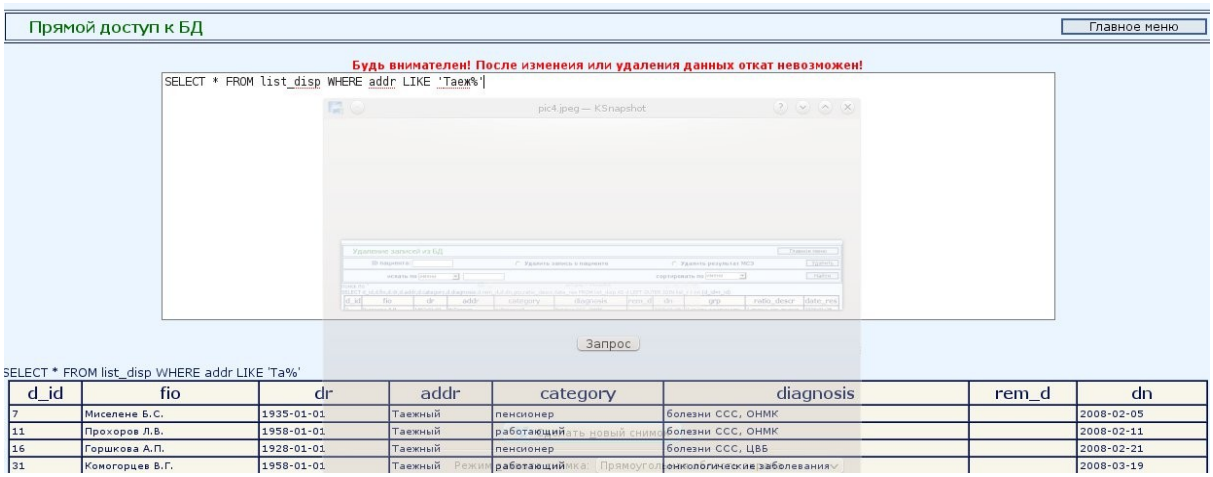

Рисунок 15 – Прямой доступ к базе данных

На этой страничке в текстовом поле можно выполнять любые запросы к БД: выборку данных, удаление, редактирование создание и удаление таблиц и т. д., используя стандартный язык *SQL*.

Также для отслеживания возможных ошибок, в режиме администратора текст всех запросов, переданных в БД, отображаются в окне браузера в том виде, в каком их получает *PostgreSQL* (Рисунок 16).

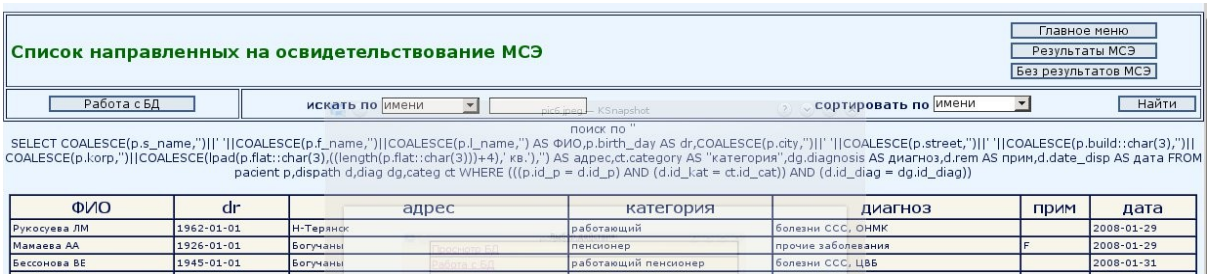

Рисунок 16 – Отображение администратору строки запроса к *PostgreSQL*

3.4 Отладка и тестирование

Все возможные ошибки программы были выявлены в ходе отладки автоматизированной системы.

В качестве тестирования программы были созданы учетные записи для пользователей с различными уровнями прав доступа. По каждому уровню были проведены различные проверки на работоспособность всех функций программного продукта (просмотр, поиск информации о пациентах по различным критериям с разными способами сортировки, добавление новых записей о пациентах, добавление информации о результатах МСЭ для уже существующих записей в БД, генерация отчетов). В режиме администратора дополнительно протестировано создание, удаление и модификация учетных записей пользователей системы, удаление информации из БД, механизм прямого доступа к БД. Кроме того, протестирован механизм разграничения прав доступа к системе. Все тесты показали полную работоспособность разработанной информационной системы.

3.5 Выводы по главе

В третьей главе была рассмотрена структура программного продукта, *UML*-диаграмма вариантов использования, приведены руководства программиста и пользователя.

ЗАКЛЮЧЕНИЕ

Цель бакалаврской работы заключалась в разработке программного обеспечения для учета выхода на инвалидность, предназначенного для применения в учреждениях здравоохранения.

Было разработано приложение, в котором реализованы: максимально упрощенный, в то же время обладающий необходимой функциональностью интерфейс, ориентированный на неподготовленного пользователя; разграничение уровней доступа; поиск и сортировка информации по различным критериям; возможность анализа накопленных данных, просмотр динамики выхода на инвалидность.

Так же в ходе работы были достигнуты следующие результаты:

- проанализирована технология проектирования *Web*-приложений;
- рассмотрено существующее программное обеспечение для работы бюро МСЭ;
- спроектирована БД для хранения информации о пациентах;
- спроектировано и реализовано приложение для учета выхода на инвалидность.

Вся информация хранится в базе данных, обеспечивающей ее целостность и надежность хранения. Система может быть установлена как на компьютер конечного пользователя, так и на выделенный сервер, в последнем случае рабочим местом пользователя может быть любой компьютер, подключенный к локальной сети.

Модульная структура системы позволяет легко расширять функциональность программного продукта путем подключения новых модулей с необходимой функциональностью.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Гутманс, Э. *PHP* 5. Профессиональное программирование [Текст] : пер. с англ. А. Киселев / Э. Гутманс, С. Баккен, Д. Ретанс. – СПб. : Символ-Плюс, 2006. – 704 с.

2. Клайн, К. *SQL*. Справочник. 2-е издание [Текст] : пер. с англ. С.М. Лунин / К. Клайн. – М. : КУДИЦ-ОБРАЗ, 2006. – 832 с.

3. Коггзолл, Дж. *PHP* 5. Полное руководство [Текст] / Дж. Коггзолл. – М. : Вильямс, 2006. – 752 с.

4. Олищук, А.В. Разработка *WEB*-приложений на *PHP* 5. Профессиональная работа [Текст] / А. В. Олищук, А. Н. Чаплыгин. – М. : Вильямс, 2006. – 352 с.

5. Уорсли, Дж. *PostgreSQL*. Для профессионалов [Текст] : пер. с англ. Е. Матвеев / Дж. Уорсли, Дж. Дрейк. – СПб. : Питер, 2003. — 496 с.

6. Федеральный закон от 20.07.1995 № 181-ФЗ (ред. от 30.11.2011) «О социальной защите инвалидов в Российской Федерации» // Официальный интернет-портал правовой информации *www.pravo.gov.ru*, 01.12.2011.

7. Документация | *PostgreSQL* [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://postgresql.ru.net/docs.html*. – Загл. с экрана.

8. Интеграция Internet-технологий и СУБД [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://citforum.ru/internet/intranet_app/interintr_11.shtml*. – Загл. с экрана.

9. Информационные системы по проблемам инвалидности и инвалидов [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://www.sparm.com/files/sparm_invalidnost.pdf*. – Загл. с экрана.

10. Компания СП.АРМ «Информационные технологии в медицине и социальной защите населения» [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://www.sparm.com/.* – Загл. с экрана.

11. Технологии баз данных для *World Wide Web*: обзор [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://www.hardline.ru/2/22/526*/. – Загл. с экрана.

12. Фаворская, М.Н. Базы Данных [Электронный ресурс] : учебно-методический электронный комплекс / М. Н. Фаворская. – Красноярск, СибГАУ.

13. *Apache Software* [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://projects.apache.org*/. – Загл. с экрана.

14. *PHP, MySQL* и другие веб-технологии [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://php.su/*. – Загл. с экрана.

15. *PostgreSQL* 8.4.4 *Documentation* [Электронный ресурс]. – Электрон. дан. – Режим доступа: *http://www.postgresql.org/docs/8.4/static /index.html*. – Загл. с экрана.