Содержание

Введение

Цель курсовой работы - разработка и реализация информационной системы, прикладные программы которой используют данные, хранящиеся в базе данных, для автоматизации деятельности некоторого предприятия. В данном контексте слово «предприятие» используется как общий термин для достаточно независимой, коммерческой, научной, технической или другой организации. Предприятие может состоять из одного человека (с небольшой частной базой данных), быть крупной организацией (с очень большой базой данных) или представлять собой нечто среднее, например, промышленное предприятие, коммерческую организацию, учебное заведение, медицинское учреждение, библиотеку, банк или правительственное ведомство.

Список рекомендуемых тем курсовых работ, а также ссылки на соответствующие литературные источники приведены в Приложениях 1, 2 и 3.

В рамках курсовой работы должны быть решены две основные задачи:

a) разработка *базы данных* и
b) разработка *приложения* (или *приложений*) доступа к базе данных.

Основными видами деятельности процесса разработки информационной системы являются: *формулировка требований*, *анализ*, *проектирование*, *реализация*, *тестирование*, *развертывание* и *настройка*. Эти виды деятельности могут быть разбиты на три части, соответствующие трем основным частям РПЗ на курсовую работу: аналитическую, конструкторскую и технологическую.

2. Общие вопросы проектирования информационных систем и программного обеспечения

При проектировании информационных систем и программного обеспечения используются различные методологии, но двумя основными являются:

a) *структурный анализ и проектирование* и
b) *объектно-ориентированный анализ и проектирование*.

Основная идея структурного подхода заключается в декомпозиции исходной задачи на функции или процессы, которая приводит к созданию иерархии процессов и подпроцессов. Основные методы структурного подхода представлены стандартами семейства IDEF (http://www.idef.com). В настоящий момент к семейству IDEF относятся следующие стандарты:

- IDEF0 - методология функционального моделирования. С помощью наглядного графического языка IDEF0, проектируемая система представляется в виде набора взаимосвязанных функций (функциональных блоков). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы.
- IDEF1 – методология моделирования информационных потоков внутри системы, позволяющая отображать и анализировать их структуру и взаимосвязи.
- IDEF1X (IDEF1 Extended) – методология построения реляционных структур. IDEF1X относится к типу методологий “Сущность-связь” (ER - Entity-Relationship) и, как правило, используется для моделирования реляционных баз данных.
- IDEF2 – методология динамического моделирования развития систем. В настоящее время используются алгоритмы и их компьютерные реализации, позволяющие превращать набор статических диаграмм IDEF0 в динамические модели, построенные на базе “раскрашенных сетей Петри” (CPN – Color Petri Nets).
- IDEF3 – методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях. С помощью IDEF3 описываются сценарий и последовательность операций для каждого процесса. IDEF3 имеет прямую взаимосвязь с методологией IDEF0 – каждая функция (функциональный блок) может быть представлена в виде отдельного процесса средствами IDEF3.
- IDEF4 – методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия, тем самым, позволяя анализировать и оптимизировать сложные объектно-ориентированные системы.
- IDEF5 – методология онтологического исследования сложных систем. С помощью методологии IDEF5 онтология системы может быть описана при помощи определенного словаря терминов и правил, на основании которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы, и производится её оптимизация.

В дополнение к IDEF0 для описания движения документов и обработки информации используются диаграммы потоков данных (DFD - Data Flow Diagrams). В отличие от IDEF0, где система рассматривается как

взаимосвязанные работы, связи в DFD показывают лишь то, как объекты (включая данные) движутся от одной работы к другой. DFD отражает функциональные зависимости значений, вычисляемых в системе, включая входные значения, выходные значения и внутренние хранилища данных. DFD содержит пять компонентов.

1) *Процессы*, которые преобразуют данные. Процессы самого нижнего уровня представляют собой функции без побочных эффектов (примером такой функции является вычисление комиссионного сбора за выполнение проводки с помощью банковской карточки). Весь граф потока данных тоже представляет собой процесс (высокого уровня). Процесс может иметь побочные эффекты, если он содержит нефункциональные компоненты, такие как хранилища данных или внешние объекты.
2) *Потоки данных*, которые переносят данные. Поток данных соединяет выход объекта (или процесса) с входом другого объекта (или процесса). Он представляет промежуточные данные вычислений.
3) *Активные объекты*, которые производят и потребляют данные. Активным называется объект, который обеспечивает движение данных, поставляя или потребляя их. Активные объекты обычно бывают присоединены к входам и выходам DFD.
4) *Хранилища данных*, которые пассивно хранят данные. Хранилище данных - это пассивный объект в составе DFD, в котором данные сохраняются для последующего доступа. Хранилище данных допускает доступ к хранимым в нем данным в порядке, отличном от того, в котором они были туда помещены. Агрегатные хранилища данных, как, например, списки и таблицы, обеспечивают доступ к данным в порядке их поступления, либо по ключам.
5) *Потоки управления*. DFD показывает все пути вычисления значений, но не показывает, в каком порядке значения вычисляются. Решения о порядке вычислений связаны с управлением программой, которое отражается в динамической модели. Эти решения, вырабатываемые специальными функциями, или предикатами, определяют, будет ли выполнен тот или иной процесс, но при этом не передают процессу никаких данных, так что их включение в функциональную модель необязательно. Тем не менее, иногда бывает полезно включать указанные предикаты в функциональную модель, чтобы в ней были отражены условия выполнения соответствующего процесса. Функция, принимающая решение о запуске процесса, будучи включенной в DFD, порождает в DFD поток управления.

Первым шагом при построении диаграмм потока данных является построение *контекстных диаграмм*. Обычно при проектировании относительно простых информационных систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы. Для сложных информационных систем строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не главный единственный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем. Пример контекстной диаграммы приведен в Приложении 4.

Основная идея объектно-ориентированного подхода состоит в рассмотрении предметной области и логического решения задачи с точки зрения объектов (понятий и сущностей). В процессе объектно-ориентированного анализа основное внимание уделяется определению и описанию объектов (или понятий) в терминах предметной области. В процессе объектно-ориентированного проектирования определяются логические программные объекты, которые будут реализованы средствами объектно-ориентированного языка программирования. Эти программные объекты включают в себя атрибуты и методы. И, наконец, в процессе конструирования или объектно-ориентированного программирования обеспечивается реализация разработанных компонентов и классов. Основные принципы объектно-ориентированного анализа и проектирования сводятся к следующему:

- *Анализ требований*, во время которого выделяются основные процессы, происходящие в моделируемой системе и их формулировка в виде прецедентов. Прецедент – это текстовое описание процессов, происходящих в предметной области.
- *Объектно-ориентированный анализ предметной области*. Задача этого шага в определении видов деятельности участников процесса и составлении концептуальной модели, которая отражает различные категории элементов предметной области. Причем не только виды деятельности участников, но и все относящиеся к делу понятия.
- *Объектно-ориентированное проектирование*, при котором основное внимание сосредоточено на распределении обязанностей. Распределение обязанностей означает выделение задач и обязанностей различных программных объектов в приложении.

Наиболее важным моментом объектно-ориентированного анализа и проектирования является квалифицированное распределение обязанностей между компонентами программной системы. Обязанности объектов и их взаимодействия изображаются с использованием диаграмм классов и диаграмм взаимодействий. Основным инструментальным средством объектно-ориентированного анализа и проектирования является унифицированный язык моделирования UML (Unified Modeling Language). Технология UML, одобренная

консорциумом OMG (Object Management Group), является мощным средством описания бизнес-процессов и представления их в той форме, которая устраивает как разработчиков, так и пользователей. В распоряжении проектировщика баз данных имеется множество UML-диаграмм (http://www.uml.org), которые часто используются на этапе определения требований, развертывания и на всех промежуточных этапах проектирования. Примеры различных UML-диаграмм приведены в Приложениях 5 и 6.

Хотя в основном UML применяется в объектно-ориентированном анализе и проектировании, однако этот язык моделирования может использоваться и во многих других типах анализа и моделирования систем, не требующих создания объектно-ориентированных приложений.

При выполнении проекта необходимо выбрать одну из методологий, и следовать ей на протяжении всей работы.

3. Содержание аналитического раздела

Приступая к работе нужно добиться простого и ясного видения проблемы, при котором задачи и приоритеты проекта стали бы очевидными. Атрибутом хорошего видения проблемы является центральная идея (лейтмотив проекта).

Когда требования сформулированы необходимо проанализировать их в целом и каждое по отдельности. Каждому требованию должен быть назначен тот или иной приоритет. Детализация приоритетов может быть любой, в зависимости от потребностей. Уровни приоритета можно определить следующим образом:

- *Необходимые*. Эти требования обязательно должны быть воплощены в программе, без этого ее нельзя выпускать на защиту. Необходимые функции должны быть реализованы и испытаны как можно раньше.
- *Желательные*. Присутствие этих требований крайне желательно. При достаточном обосновании от их реализации можно отказаться, но это не уменьшает важности их наличия в программе. Реализация и испытания желательных требований начинается сразу после обязательных.
- *Возможные*. Эти требования также желательны, но реализуются в последнюю очередь и являются первыми кандидатурами на удаление, если вдруг возникнут проблемы со сроками.

Обычно проект начинается с создания документа, содержащего описание его основных целей. В описании формулируются очень абстрактные (и зачастую нетехнические) требования к системе, необходимые для понимания того, что же собой представляет система в целом. Пример такого документа приведен в Приложении 7.

В рамках любой предметной области действует широкий набор политик, законов и промышленных стандартов. Все эти контролирующие принципы в целом называются правилами делового регламентами или деловыми правилами. Правила делового регламента - один из основных источников функциональных требований, поскольку они определяют возможности, которыми должна обладать система для выполнения правил.

Для создания и документирования правил делового регламента и их применения разработаны различные методики, а для организации правила делового регламента предложено множество разных таксономий (схем классификаций). Простейшая из них включает пять типов правил.

1) *Факты* (или *инварианты*) – это неизменные истины о сущности данных и их атрибутах. Зачастую они описывают связи и отношения между важными бизнес-терминами. Примеры фактов:
   - стоимость билетов не возвращается, если покупатель изменяет маршрут;
   - со стоимости доставки налог с продаж не берется.
2) *Ограничения* определяют, какие операции может выполнять система и ее пользователи. Вот некоторые слова и фразы, которые часто применяются при описании ограничивающего правила: должен, не должен, не может и только. Примеры ограничений:
   - постоянный посетитель библиотеки может отложить для себя до 10 книг;
   - экипажи коммерческих авиарейсов должны каждые 24 часа отдыхать не менее 8 часов;
3) *Активаторы операций* – это правила, при определенных условиях приводящие к выполнению каких-либо действий. Выразительным способом документирования активирующих операции являются таблицы решений. Примеры активаторов операций:
   - если срок хранения контейнера с химикатом истек, об этом необходимо уведомить лицо, у которого в данный момент находится контейнер;
   - если клиент заказал книгу автора, написавшего несколько книг, клиенту следует предложить другие книги этого автора, прежде чем принять заказ,

4) *Вывод* - это правило, устанавливающее новые реалии на основе достоверности определенных условий. Вывод создает новый факт на основе других фактов или вычислений. Выводы зачастую записывают в формате «если — то», применяемом также при записи правил, активирующих операции; тем не менее, раздел «то» вывода заключает в себе факт или предположение, а не действие. Примеры выводов:
   - если платеж не поступил в течение 30 календарных дней с момента отправки счета, счет считается просроченным;
   - если поставщик не может поставить заказанный товар в течение пяти дней с момента получения заказа, заказ считается невыполненным;
5) *Вычисления* – это один из классов правил, которые определяют вычисления, выполняемые с использованием математических формул и алгоритмов. В отличие от активирующих операций правила вычислений можно рассматривать в качестве требований к программному обеспечению:
   - цена единицы товара снижается на 10% при заказе от 6 до 10 единиц, на 20% - при заказе от 11 до 20 единиц и на 30% - при заказе свыше 20 единиц;
   - общая стоимость заказа вычисляется как сумма стоимостей всех заказанных товаров, за вычетом скидок на количество, плюс государственные и местные налоги, действующие в округе, куда будет доставлен товар, плюс стоимость доставки и плюс необязательный страховой сбор.

Итогом разработки требований является техническое задание. При оформлении технического задания следует придерживаться ГОСТ 19.201-78 «ТЕХНИЧЕСКОЕ ЗАДАНИЕ. Требования к содержанию и оформлению».

Подробные функциональные и нефункциональные требования к разрабатываемой системе должны быть записаны в спецификации к требованиям. Существует несколько способов представления требований:
- *документация*, в которой используется четко структурированный и аккуратно используемый естественный язык;
- *графические модели*, иллюстрирующие процессы трансформации, состояния системы и их изменения, взаимодействия данных, а также логические потоки, классы объектов и отношения между ними;
- *формальные спецификации*, где требования определены с помощью математически точных, формальных логических языков.

После формулирования требований проводят их анализ и осуществляют построение концептуальной и логической модели информационной системы, в соответствии с выбранной методологией. Подготавливается набор диаграмм, одной из которых, вероятно, будет ER-диаграмма (в случае структурного подхода). При необходимости осуществляют построение ER-диаграммы с рациональной структурой, используя концепцию нормальных форм.

4. Содержание конструкторского раздела

В конструкторской части проекта выполняется проектирование базы данных с учетом выбранной СУБД. В этой части осуществляются следующие мероприятия.
- Реализуются все объекты базы данных: пользовательские типы данных или домены, таблицы, представления, индексы, умолчания, правила, ограничения целостности, хранимые процедуры, триггеры и функции. Создание перечисленных объектов необходимо оформить в виде сценария создания базы данных и разместить в приложении. Следует заметить, что триггеры как процедурные средства обеспечения целостности, хранимые процедуры, реализующие правила делового регламента, и функции, возвращающие табличные значения, трактуются как «объекты» базы данных наряду с таблицами, утверждениями, индексами и прочими структурами.
- Реализуются запросы к базе данных, как на выборку, так и на обновление данных (часть запросов может быть запрограммирована в хранимых процедурах). Демонстрируется весь спектр запросов на выборку: простые, многотабличные, итоговые запросы с использованием статистических функций, запросы с группировкой, подчиненные и составные запросы.
- Для обеспечения бесконфликтной работы в многопользовательском режиме при выполнении операций обновления составляются транзакции. В Приложении 8 приведен ряд типичных примеров транзакций для учебной базы данных.
- Описываются меры безопасности, находящиеся в компетенции СУБД. Принято различать шесть уровней системы безопасности БД: 1) физическую систему безопасности; 2) безопасность сетевого протокола; 3) доменную безопасность; 4) безопасность локального компьютера; 5) систему безопасности сервера БД; 6) безопасность приложений. За реализацию первых четырех уровней системы безопасности БД отвечают сетевые и системные администраторы, за пятый уровень отвечают администраторы и разработчики БД, а за шестой - разработчики приложений. В конструкторской части проекта основное внимание должно быть уделено системе безопасности сервера БД, занимающей пятый уровень модели безопасности. Средства безопасности сервера БД можно разделить на четыре категории: *аутентификация*, *авторизация*, *аудит* и *шифрование*. Процесс предоставления доступа к

БД состоит из двух фаз: сначала выполняется подключение к серверу БД (аутентификация), а затем открывается доступ к БД с ее объектами (авторизация). Разрешения на работу с объектами позволяют или запрещают пользователям выполнять действия над объектами БД, например таблицами и представлениями. Разрешения на выполнение SQL-операторов позволяют или запрещают пользователям создавать объекты, делать копии БД и файлов журнала. Действия, выполняемые в БД, отслеживаются с помощью аудита сервера БД. Некоторые объекты БД, например, хранимые процедуры, разрешается зашифровать, чтобы защитить их содержимое. Для планирования системы безопасности необходимо определить требования к системе безопасности из списка системных требований, сформулированных в аналитической части проекта. Для реализации системы безопасности следует определить пользователей, группы и роли, которым назначаются привилегии, и связать с ними требования, создав список соответствий «пользователь - действие». По возможности следует оптимизировать структуру системы безопасности с помощью вложенных групп и ролей, а также цепочек владения. Созданную систему безопасности необходимо испытать, подключаясь к БД с различными учетными именами, проверяя работу разрешений и выполняя аудит.

5. Содержание технологического раздела

В технологической части работы выполняется разработка приложения и дается описание презентационного уровня приложения. Презентационный уровень - это часть приложения, на котором происходит взаимодействие пользователя с уровнем функций делового регламента. В простейших реализациях презентационного уровня используются *элементы пользовательского интерфейса* из состава графического пользовательского интерфейса (GUI), встроенного в ОС, например Microsoft Windows Forms и Microsoft ASP.NET Web Forms.

Презентационный уровень в определенной степени это основная часть приложения, поскольку для большинства пользовательский интерфейс - это и есть само приложение. Компоненты пользовательского интерфейса применяются для отображения информации и получения данных от пользователя, интерпретации событий, обусловленных действиями пользователя, изменения состояния интерфейса и информирования о степени выполнения задач. Кроме того, пользовательский интерфейс должен фильтровать операции, разрешая только те, на которые у данного пользователя есть право. Основными характеристиками удачно спроектированного и эффективного интерфейса являются:

- интуитивно понятный дизайн;
- продуманное использование площади экрана;
- внешний вид (можно быстро научитесь создавать удобный пользовательский интерфейс, если следовать общепринятым принципам и стандартам, описанным в библиотеке MSDN);
- простота навигации;
- управляемая навигация;
- заполнение полей данными по умолчанию;
- проверка входных данных на корректность;
- меню, панели инструментов и справочная система;
- эффективная обработка событий.

При разработке Web-приложений необходимо решить ряд дополнительных задач. По вопросам разработки Web-приложений имеется обширная литература. Существует также множество книг по отдельным технологиям, таким как HTML, XML, EJB, SSL, CGI, TCP/IP, ASP, ASP.NET, JSP и многим другим. Поэтому ограничимся рассмотрением двух вопросов: выбором архитектурного шаблона Web-приложений и выбором программного интерфейса для доступа к данным.

На достаточно высоком уровне абстракции выделяют следующие три шаблона Web-приложений:

- *Thin Web Client* (на основе «тонкого» Web-клиента) используется в большинстве приложений Интернета и предоставляет ограниченные возможности по управлению конфигурацией клиента. В распоряжении клиента должен быть только стандартный браузер, поддерживающий формы. Все операции, связанные с бизнес-логикой, выполняются на сервере.
- *Thick Web Client* (на основе «толстого» Web-клиента) предполагает, что значительная часть бизнес-логики выполняется на клиентской машине. Обычно для выполнения бизнес-логики клиентом используется динамический HTML, аплеты Java или управляющие элементы ActiveX. Взаимодействие с сервером, по-прежнему происходит через протокол HTTP.
- *Web Delivery* (на основе механизма Web-доставки). При взаимодействии клиента и сервера, кроме протокола HTTP, используются и другие протоколы, например, такие как IOOP и DCOM, которые могут применяться для поддержки системы распределенных объектов. В данном случае браузер функционирует как контейнерный модуль системы распределенных объектов.

Приведенный список, нельзя считать исчерпывающим, особенно учитывая тот факт, что технологические изменения происходят ежегодно. Различные шаблоны можно одновременно применять в одной архитектуре. Например, в системе электронной коммерции шаблон Thin Web Client может использоваться при продаже товаров потребителю, а шаблон Thick Web Client или Web Delivery можно применять для поддержки офиса филиала торговой компании. Выбор такой архитектуры вполне оправдан, поскольку уровни управляемости конфигурацией клиента и администратора должны различаться

Программные интерфейсы для доступа к данным в значительной степени связаны с особенностями конкретной СУБД, тем не менее, в основе всех интерфейсов, применяемых в коммерческих СУБД, лежат одни и те же принципы. Например, СУБД MS SQL Server поддерживает такие интерфейсы доступа к данным: Embedded SQL for C (ESQL/C), SQL-DMO, DB-Library, ODBC, OLE DB, ADO, ADO.NET. Среди интерфейсов, поддерживаемых другими СУБД, следует отметить BDE - универсальный стандарт компании Inprise (SQL Links), OCI - интерфейс вызовов Oracle компании Oracle, JDBC - API для выполнения SQL запросов к базам данных из программ, написанных на языке Java компанией Sun Microsystems. При разработке приложения необходимо выполнить мотивированный выбор соответствующего интерфейса. Пример реализации Web-приложения приведен в Приложении 9.

6. Содержание исследовательской части

Исследовательская часть является факультативной, т. е. необязательной. В качестве примеров исследований могут быть предложены следующие темы:

- Методика аудита и оптимизация производительности сервера БД.
- Оптимизация производительности сервера БД для полнотекстового поиска.
- Сравнительный анализ методов доступа к метаданным: с помощью системных таблиц, системных хранимых процедур и представлений информационной схемы.
- Исследование проблемы сеанса, остающегося открытым на стороне сервера после отключения пользователя.
- Сравнительный анализ работы со строками: в хранимых процедурах, в компонентах промежуточного уровня или на клиенте.
- Автоматическая проверка хранимых процедур на наличие нарушений стандартов кодирования.
- Определение набора простых и составных индексов для базовых таблиц и материализованных представлений.
- Исследование проблемы защиты строк подключения баз данных, паролей и других секретных параметров приложения.

7. Требования к содержанию структурных элементов РПЗ

Структурными элементами РПЗ являются:

- титульный лист;
- задание на курсовую работу;
- реферат;
- содержание;
- введение;
- основная часть;
- заключение;
- список использованных источников;
- приложения.

Титульный лист является первой страницей РПЗ и при его оформлении следует придерживаться ГОСТ 7.32-2001 «Отчет о научно-исследовательской работе. Структура и правила оформления».

Задание на курсовую работу оформляется на специальном бланке. Образец бланка задания приведен в Приложении 10.

При оформлении реферата следует придерживаться ГОСТ 7.9-95 «РЕФЕРАТ И АННОТАЦИЯ. Общие требования». Реферат должен содержать:

1) сведения об объеме РПЗ, количестве иллюстраций, таблиц, приложений, количестве использованных источников;
2) перечень ключевых слов;
3) текст реферата.

Перечень ключевых слов должен включать от 5 до 15 слов или словосочетаний из текста РПЗ, которые в наибольшей мере характеризуют ее содержание и обеспечивают возможность информационного поиска.

Ключевые слова приводятся в именительном падеже и печатаются строчными буквами в строку через запятые. Текст реферата должен отражать:

1) объект разработки (или исследования);
2) цель работы;
3) метод или методологию проведения работы;
4) результаты работы;
5) основные конструкторские и технологические характеристики разработанной информационной системы и программного обеспечения;
6) область применения;
7) прогнозные предположения о развитии объекта разработки.

Пример составления реферата приведен в ГОСТе 7.9-95.

Содержание включает введение, наименование всех разделов, подразделов, пунктов (если они имеют наименование), заключение, список использованных источников и наименование приложений с указанием номеров страниц, с которых начинаются эти элементы РПЗ.

Во введении должны быть сформулированы цели и задачи проекта. Введение должно содержать оценку современного состояния решаемой проблемы, основание и исходные данные для разработки темы, обоснование необходимости проведения исследований, актуальность темы, связь данной работы с другими аналогичными работами.

В основной части РПЗ приводят данные, отражающие сущность, методику и основные результаты выполненной работы. Основная часть в обязательном порядке должна содержать три раздела: аналитический, конструкторский и технологический. При необходимости допускается исследовательский раздел. В конце каждого раздела приводятся краткие выводы.

Заключение должно содержать:

1) краткие выводы по результатам выполненной работы;
2) оценку полноты решений поставленных задач;
3) рекомендации по конкретному использованию результатов работы;
4) оценку научно-технического уровня выполненной работы в сравнении с лучшими достижениями в данной области.

Список использованных источников должен содержать сведения об источниках, использованных при составлении РПЗ. Сведения об источниках приводятся в соответствии с требованиями ГОСТ 7.1–2003 «БИБЛИОГРАФИЧЕСКАЯ ЗАПИСЬ. БИБЛИОГРАФИЧЕСКОЕ ОПИСАНИЕ. Общие требования и правила составления». Библиографическое описание кратких ссылок выполняют в соответствии с требованиями ГОСТ Р 7.0.5 - 2008 «БИБЛИОГРАФИЧЕСКАЯ ССЫЛКА. Общие требования и правила составления». Особое внимание следует обратить на составление библиографических ссылок на электронные ресурсы.

В приложения рекомендуется включать материалы, связанные с выполненным проектом, которые по каким-либо причинам не могут быть включены в основную часть. В качестве приложения рекомендуется включать сценарий создания базы данных на языке SQL. Примеры сценариев создания баз данных приведены в Приложении 11.

8. Правила оформления расчетно-пояснительной записки

При оформлении расчетно-пояснительной записки следует придерживаться ГОСТ 7.32-2001 «Отчет о научно-исследовательской работе. Структура и правила оформления». Краткая сводка правил оформлении РПЗ приведена в Приложении 12.

Литература

Список рекомендуемой литературы для выполнения курсовой работы приведен в Приложении 13.

Незаконченный пример оформления РПЗ приведен в Приложении 14.