

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Колесникова Екатерина Дмитриевна

Должность: Ректор СГТИ

Дата подписания: 13.10.2025 16:04:23

Уникальный идентификатор:

5791137b901af6f58fa81bc87176652f0e29200737d9e2c40df6a70c9c69444d



**ЧАСТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СРЕДНЕРУССКИЙ ГУМАНИТАРНО-ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»**

ПРИНЯТО:

Решение Ученого Совета СГТИ

от «10» октября 2025 г.

Протокол № 3

УТВЕРЖДАЮ:

Ректор СГТИ

Е.Д. Колесникова

«10» октября 2025 г.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ
ПО ПОДГОТОВКЕ И ЗАЩИТЕ КУРСОВЫХ РАБОТ ПО ДИСЦИПЛИНЕ
«ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ»**

для студентов, обучающихся по направлению подготовки
09.03.03 Прикладная информатика
(уровень бакалавриата)
направленность (профиль): «Прикладная информатика в экономике»

1. СТРУКТУРА И СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

Вне зависимости от решаемой задачи и подхода при проектировании курсовая работа должна включать в себя следующие основные разделы:

- -введение;
- -основная содержательная часть;
- -заключение;
- -список использованных источников и литературы;
- -приложения.

Во *введении* к письменной работе содержится:

- научное и практическое обоснование *актуальности* выбранной темы и вытекающие из этого *цели и задачи* работы;
- описание структуры пояснительной записки, состав и содержание глав и разделов, наличие приложений, схем, графиков и таблиц.
- краткий обзор-аннотацию нескольких основных, наиболее активно
- использованных в работе источников и литературы.

По объему введение не должно превышать 2-3 страниц.

Основная содержательная часть письменной работы строится в соответствии с разработанным планом, позволяющим последовательно, логично и доказательно изложить материал и сделать вытекающие из него теоретические и практические выводы.

Глава состоит из нескольких пунктов и подпунктов. Подпункты, как правило, не должны содержать более мелких делений. Главы, пункты и подпункты должны иметь заголовки, отражающие их содержание и нумероваться. Внутри располагается текст, таблицы, схемы и графики.

Примерная структура основной части курсовой работы:

1. Аналитическая часть
2. Проектная часть
3. Реализация с проектированной ИС

Обязательным структурным элементом основной части курсовой работы является *аналитический обзор* по выбранной теме, который входит в первую главу.

Аналитический обзор представляет собой результат аналитико-синтетической обработки совокупности документов по определенной теме, содержащей обобщенные и критически проанализированные сведения об истории, современном состоянии, тенденциях и перспективах развития предмета обзора.

К тексту аналитического обзора предъявляются основные требования:

- Полнота и достоверность информации;
- Наличие критической оценки использованной информации;
- Логичность структуры;
- Композиционная целостность;
- Аргументированность выводов;
- Ясность, четкость и лаконичность изложения.

Работа делится на *главы, пункты и подпункты*, представляющие собой законченные в смысловом отношении фрагменты работы.

1. СОДЕРЖАНИЕ ПЕРВОЙ ГЛАВЫ

Целью аналитической части является рассмотрение существующего состояния предметной области, характеристики объекта и системы управления и обоснование предложений по устранению выявленных недостатков, внедрению новых подходов, новых технологий и т. д.

Ниже, в зависимости от поставленной задачи предлагается содержание первой главы курсовой работы.

1. Аналитическая часть

1.1. Анализ предметной области

1.1.1. Техничко-экономическая характеристика предметной области

1.1.2. Системный анализ объекта исследования

1.2. Обоснование необходимости и цели использования вычислительной техники для решения задачи.

1.3. Постановка задачи

1.3.1. Цель и назначение автоматизированного варианта решения задачи

1.3.2. Общая характеристика организации решения задачи на ЭВМ

1.4. Анализ существующих разработок и обоснование выбора технологии проектирования

1.1. Анализ предметной области

Содержит анализ источников и литературы, связанный с исследованием современных технологий и средств разработки могущих иметь отношение к курсовой работе, определение проблемы связанной с разработкой ИС, на основе поставленной цели, определение класса проектируемой ИС, ее жизненного цикла, выбор методологии и технологии проектирования.

При проектировании и разработке ИС обычно используется два подхода:

- *функционально-модульный* или *структурный* подход,
- *объектно-ориентированный* подход.

При использовании этих подходов студент должен руководствоваться различными группами требований, которые найдут отражение ниже. Объектно -ориентированный подход может применяться при проектировании всех классов задач, поэтому не следует заведомо ограничивать “область допустимых значений” методики проектирования. Использование новейших методик проектирования и разработки является неотъемлемым условием жизнеспособности ЭИС в условиях современной технологической революции.

Далее следует дать краткую характеристику современных **технологий проектирования**, их положительные черты и недостатки, перечислить основные факторы выбора.

1.1.1. Техничко-экономическая характеристика предметной области

В качестве **предметной области** может выступать **подразделение предприятия**, фирмы, объединения и т.д., или **отдельный вид деятельности**, протекающий в нем, поэтому в начале данного раздела необходимо отразить цель функционирования предприятия, его организационную структуру и основные параметры его функционирования.

1.1.2. Системный анализ объекта исследования

При разработке любой информационной системы необходимо провести системный анализ предметной области, который позволит определить круг решаемых задач, подлежащих автоматизации, позволит обосновать необходимость процессов автоматизации, определить функциональную часть ИС, выявить функциональные подсистемы разрабатываемой ИС, информационные связи между ними. Это даст возможность создать функциональную и информационную модель системы, которые в дальнейшем будут использоваться на этапе проектирования.

Для проведения системного анализа предметной области необходимо выбрать метод его проведения. Для разрабатываемых в курсовой работе систем наиболее приемлемым является метод структурного анализа.

Структурным анализом принято называть метод исследования системы, начинающий с ее общего обзора, который затем детализируется, приобретая иерархическую структуру со все большим числом уровней.

В структурном анализе основным методом разбиения на уровни абстракции является функциональная декомпозиция, заключающаяся в декомпозиции системы на функциональные подсистемы, которые, в свою очередь, делятся на подфункции, т.е. - на задачи и так далее до конкретных процедур. При этом система сохраняет целостное представление, в котором все составляющие компоненты взаимосвязаны.

Для выполнения **структурно-функционального анализа** объекта управления и решаемой задачи рекомендуется разработать структурно-функциональную диаграмму по методологии SADT(IDEFO). Для их разработки целесообразно использовать CASE средства, например Design/IDEF, CASE - аналитик, BPwin, Silverrun-BMP, Natural Engeneering Workbentch. При наличии в курсовой работе таких диаграмм на их графическое содержание не будут накладываться условия соответствия ГОСТ.

Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой -либо предметной области. Функциональная модель SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями. Основные элементы этой методологии основываются на следующих концепциях:

- графическое представление блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описываются посредством интерфейсных дуг, выражающих «ограничения», которые в свою очередь определяют, когда и каким образом функции выполняются и управляются;
- строгость и точность. Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. Правила SADT включают:
 - ограничение количества блоков на каждом уровне декомпозиции (правило 3 -6 блоков);
 - связность диаграмм (номера блоков);
 - уникальность меток и наименований (отсутствие повторяющихся имен);
 - синтаксические правила для графики (блоков и дуг);
 - разделение входов и управлений (правило определения роли данных).
 - отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

Методология SADT может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих систем, SADT может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Функциональную часть разрабатываемой ИС можно представить в виде бизнес - процессов, которые предполагается реализовать в проектируемой системе. Модель бизнес - процессов разрабатывается помощью CASE-средства проектирования BWin.

В основе функционального анализа лежит принцип структурного проектирования, что требует разработки системы методом «сверху-вниз». Этот метод заключается в декомпозиции системы, т.е. разбиение ее на отдельные более мелкие части. На рис.1 представлен пример контекстной диаграммы для ИС учета материальных ценностей библиотеки.

На рис. 2 она детализирована на отдельные подсистемы, т.е. проведена декомпозиция системы первого уровня. Декомпозицию можно продолжить до необходимого количества уровней, которые дают полное представление о функциях разрабатываемой системы.

При разработке бизнес-процессов необходимо полно описать входные потоки информации, выходные потоки информации, нормативы и механизмы, использующиеся при работе ИС.

1.2.Обоснование необходимости и цели использования вычислительной техники для решения задачи

В этом разделе требуется обосновать экономическую целесообразность и сформулировать цели использования вычислительной техники для рассматриваемой задачи. Здесь необходимо:

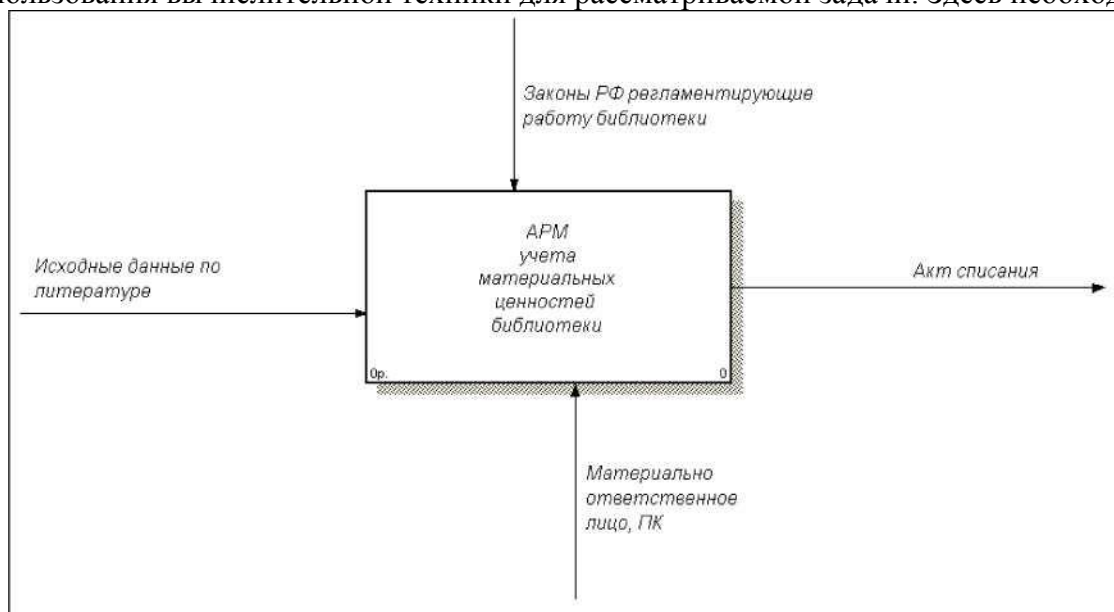


Рис.1. Контекстная диаграмма

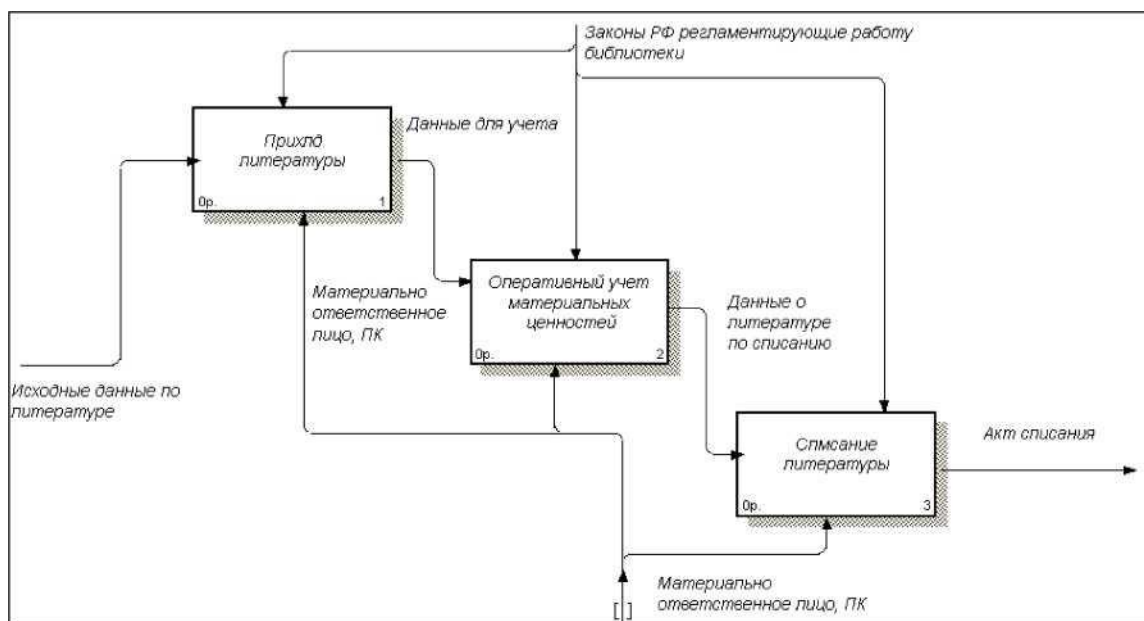


Рис.2. Функциональная диаграмма первого уровня

- описать **существующую (предметную) технологию** выполнения выбранной для рассмотрения функции управления (или комплекса функций), т.е. указать на особенности расчета показателей, указать перечни и источники используемых входных документов, перечни и адресаты результатных документов, места их обработки, методы и технические средства, применяемые для их обработки;
- привести **схемы документооборота** для каждого документа и таблицы, содержащие прагматические оценки потоков информации (объемы в документах, показателях и символах за год, трудовые затраты на их обработку за год, частоту возникновения и др.);
- выявить **основные недостатки**, присущие существующей практике управления и обработки экономической информации.

При этом следует сделать акцент на те недостатки, устранение которых предполагается осуществить в проекте, например:

- наличие опозданий в поставках сырья и материалов;
- наличие выплат штрафных санкций и неустоек;
- простои оборудования;
- низкая производительность труда в производственной сфере;
- невозможность расчета показателей, необходимых для управления объектом из-за сложности вычислений или большого объема информации;
- высокая трудоемкость обработки информации (привести объемно-временные параметры);
- низкая оперативность, снижающая качество управления объектом;
- невысокая достоверность результатов решения задачи из-за дублирования потоков информации;
- несовершенство организации сбора и регистрации исходной информации;
- несовершенство процессов сбора, передачи, обработки, хранения, защиты целостности и секретности информации и процессов выдачи результатов расчетов конечному пользователю и т.д.

1.3. Постановка задачи

В этом пункте необходимо сформулировать задачу разработки проекта и выделить основные требования к проектируемой системе обработки данных. Стоит определить тип проектируемой системы: это может быть диалоговая система решения задачи или обработки транзакций, система поддержки принятия решений или комбинированная система.

Постановка задачи может сводиться к разработке технического задания на проектируемую систему, выполненную согласно требований стандартов.

Можно описать данный пункт согласно схеме приведенной ниже.

1.3.1. Цель и назначение автоматизированного варианта решения задачи

Цель решения задачи должна сводиться к устранению тех недостатков, которые были отмечены автором в предыдущем разделе, поэтому ее можно разделить на две группы подцелей:

- достижения **улучшения** ряда **показателей выполнения** выбранной **функции управления** или работы рассматриваемого подразделения, или всего предприятия в целом (например, увеличение выпуска продукции, или увеличение числа обслуживаемых клиентов, сокращение простоев на . . .число часов и т. д);
- **улучшения значений показателей качества обработки информации** (например, сокращение времени обработки и получения оперативных данных для принятия управленческих решений; повышение степени достоверности обработки информации, степени ее защищенности , повышение степени автоматизации получения первичной информации; увеличение количества аналитических показателей, получаемых на базе исходных и т.д).

При описании **назначения** решения задачи студенту следует сделать акцент на перечень тех функций управления, которые будут автоматизированы, при внедрении предлагаемого проекта.

Пример. Назначением реализации проекта «.....» может служить:

- автоматизация получения по электронной почте входных документов;
- автоматизация ввода, контроля и загрузки данных первичных документов в базу данных с использованием экранных форм (дать перечень);
- ведение файлов с условно-постоянной информацией в базе данных;
- выполнение расчетов и выдача результатных документов;
- выдача справочной информации:
 - а) по регламентированным запросам;
 - б) по нерегламентированным запросам.

1.3.2. Общая характеристика организации решения задачи на ЭВМ

В данном пункте автору следует раскрыть требования к будущему проекту через ответ на следующие вопросы:

- **изменения в функциях** подразделения, связанных со сбором, обработкой и выдачей информации;
- **источники** поступления оперативной и условно-постоянной информацией и периодичность ее поступления;
- **этапы** решения задачи, **последовательность** и временной регламент их выполнения, выявленные на основе рассмотренной в п.1.3. декомпозиции задачи (при этом следует рассмотреть целесообразность автоматизации этапов и операций решения задачи, оценивая возможность формализации связей между ними);
- **порядок ввода** первичной информации (названия документов) и перечень используемых экранных форм;
- краткая **характеристика результатов** (названия результатных документов, экранных форм выдачи результатов, перечень результатных файлов, способов их выдачи: на экран, печать или в канал связи) и мест их использования;
- краткая **характеристика системы ведения** файлов в базе данных (перечень файлов с условно-постоянной и оперативной информацией, периодичность обновления, требования защиты целостности и секретности);
- **режим решения задачи** (пакетный, диалоговый, с использованием методов телеобработки или смешанный);
- периодичность решения задачи.

1.4. Анализ существующих разработок и обоснование выбора технологии проектирования

В этом разделе следует отметить, используются ли при существующей технологии решения задачи какие-либо программные средства и, если используются, то каким образом. Если на рынке программных средств существуют готовые программные решения, желательно дать краткое описание и провести анализ хотя бы одной такой разработки, указав основные характеристики и функциональные возможности.

Обзор рынка программных средств удобно проводить с помощью Internet. Адреса используемых при обзоре ресурсов следует добавить в список литературы курсовой работы.

Затем следует отметить, чем, с точки зрения программной реализации, должна и будет отличаться проектируемая технология решения задачи от существующей, а также, почему необходимо разрабатывать новое программное средство, и чем оно должно отличаться от существующих.

Далее следует дать краткую характеристику современных **технологий проектирования**, используемых в данной работе, обосновать их выбор.

2. СОДЕРЖАНИЕ ВТОРОЙ ГЛАВЫ

Проектная часть курсовой работы является описанием решений, принятых по всей вертикали проектирования. Глава должна быть основана на информации, представленной в аналитической части, обобщать ее. По сути, проектная часть является решением проблематики, изложенной в аналитической части, на языке информационных технологий. Поэтому недопустимо, если при проектировании используется информация об объекте управления, не описанная в первой главе.

2. Проектирование ИС

2.1. Разработка информационной модели

2.2. Проектирование информационного хранилища

2.2.1. Разработка концептуальной модели БД

2.2.2. Разработка логической модели БД

2.2.3. Разработка физической модели БД

2.3. Разработка сценария диалога системы с пользователем

2. Проектирование ИС

2.1. Разработка информационной модели

Проектирование информационной модели производится на основе данных полученных в результате анализа предметной области. Информационную модель целесообразно представить в виде диаграммы потоков данных.

Диаграмма потоков данных разрабатывается по методологии Гейна/Сарсона, Йодана/ДеМарко. Для ее разработки целесообразно использовать CASE - средства.

При разработке диаграммы потоков данных наибольшее внимание следует уделить выделению:

- внешних сущностей;
- систем/подсистем;
- процессов;
- накопителей данных;
- потоков данных.

Внешние сущности

Внешняя сущность представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой ИС. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой ИС, если это необходимо, или, наоборот, часть процессов ИС может быть вынесена за пределы диаграммы и представлена как внешняя сущность.

Процессы

Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство и т.д.

Накопители данных

Накопитель данных представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми.

Потоки данных

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т.д.

Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока (рис. 7). Каждый поток данных имеет имя, отражающее его содержание.

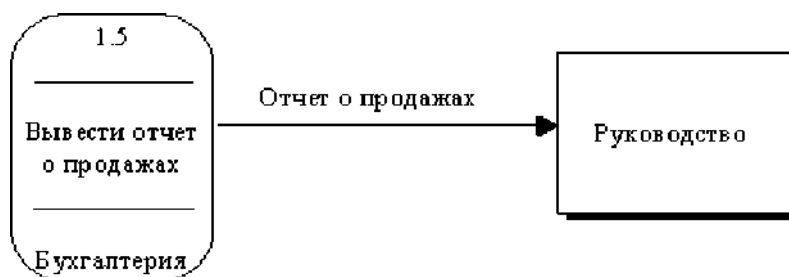


Рис. 7. Поток данных

Построение иерархии диаграмм потоков данных

Первым шагом при построении иерархии ДПД является построение контекстных диаграмм. Обычно при проектировании относительно простых ИС 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 ИС строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не единственный главный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

Пример диаграммы потоков данных разработанной средствами VPwin, представлен на рис.8.

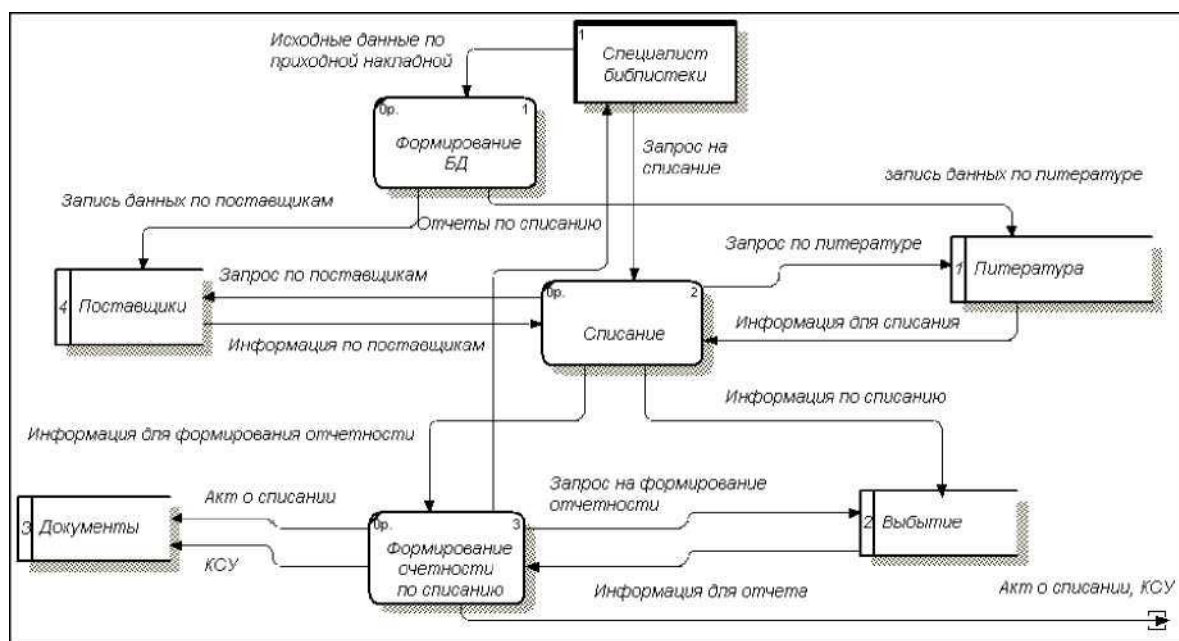


Рис.8. ДПД процесса учета материальных ценностей библиотеки

2.2. Проектирование информационного хранилища

Традиционно процедуру проектирования базы данных разбивают на три этапа, каждый из которых завершается созданием соответствующей информационной модели.

Этап 1-й. Концептуальное проектирование - создание представления (схемы, модели) БД, включающего определение важнейших сущностей (таблиц) и связей между ними, но не зависящего от модели БД (иерархической, сетевой, реляционной и т. д.) и физической реализации (целевой СУБД).

Этап 2-й. Логическое проектирование - развитие концептуального представления БД с учетом принимаемой модели (иерархической, сетевой, реляционной и т.д.).

Этап 3-й. Физическое проектирование - развитие логической модели БД с учетом выбранной целевой СУБД.

Концептуальное и логическое проектирование вместе называют также *инфологическим* или *семантическим проектированием*.

В настоящее время для проектирования БД активно используются CASE-средства, в основном ориентированные на использование *ERD (Entity - Relationship Diagrams, диаграммы «сущность-связь»)*. С их помощью определяются важные для предметной области объекты (сущности), отношения друг с другом (связи) и их свойства (атрибуты). Следует отметить, что средства проектирования ERD в основном ориентированы на реляционные базы данных (РБД), и если существует необходимость проектирования другой системы, скажем объектно-ориентированной, то лучше избрать другие методы проектирования.

Основные элементы ERD перечислены ниже.

Сущность (таблица, в РБД - отношение) - реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области, информация о котором подлежит хранению. Если выразиться точнее, то это не объект, а набор объектов (класс) с одинаковыми свойствами. Примеры сущностей: работник, деталь, ведомость, результаты сдачи экзамена и т. д.

Экземпляр сущности (запись, строка, в РБД - кортеж) - уникально идентифицируемый объект.

Связь - некоторая ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. Примерами связей могут являться родственные отношения «отец-сын», производственные - «начальник-подчиненный» или произвольные - «иметь в собственности», «обладать свойством».

Атрибут (столбец, поле) - свойство сущности или связи.

Большинство современных CASE-средств моделирования данных, как правило, поддерживает несколько графических нотаций построения информационных моделей. В частности система ERwin фирмы Computer Associates поддерживает две нотации: IDEF1X и IE (англ. Information Engineering - информационное проектирование). Данные нотации являются взаимно-однозначными, т. е. переход от одной нотации к другой и обратно выполняется без потери качества модели. Отличие между ними заключается лишь в форме отображения элементов модели.

При использовании любого CASE-средства вначале строится логическая модель БД в виде диаграммы с указанием сущностей и связей между ними. *Логической моделью* называется универсальное представление структуры данных, независимое от конечной реализации базы данных и аппаратной платформы. На основании полученной логической модели переходят к физической модели данных. *Физическая модель* представляет собой диаграмму, содержащую всю необходимую информацию для генерации БД для конкретной СУБД или даже конкретной версии СУБД. Перечисленный выше порядок действий называется *прямым проектированием БД (Forward Engineering DB)*. CASE- средства позволяют выполнять также *обратное проектирование БД (Reverse Engineering DB)*, т.е. на основании системного каталога БД построить физическую и, далее, логическую модель данных.

Далее рассматривается процедура прямого проектирования с использованием методологии IDEF1X.

2.2.1. Разработка концептуальной модели БД

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. Часто на такой модели отображаются только имена сущностей (таблиц) без указания их атрибутов. *Представление пользователя* включает в себя данные, необходимые конкретному пользователю для принятия решений или выполнения некоторого задания.

Ниже рассматривается последовательность шагов при концептуальном проектировании.

1. Выделение сущностей.

Первый шаг в построении концептуальной модели данных состоит в определении основных объектов (сущностей), которые могут интересовать пользователя и, следовательно, должны храниться в БД. При наличии функциональной модели IDEF0 прообразами таких объектов являются входы, управления и выходы. Еще лучше для этих целей использовать DFD. Прообразами объектов в этом случае будут накопители данных. Как было отмечено выше, накопитель данных является совокупностью таблиц (набором объектов) или непосредственно таблицей (объектом). Для более детального определения набора основных объектов необходимо также проанализировать потоки данных и весь методический материал, требуемый для решения задачи. Например, для задачи определения допускаемых скоростей основными объектами (наборами объектов) являются: нормативно-справочная информация, информация об участках дороги, задания на расчет, ведомости допускаемых скоростей и т. д. В ходе анализа и проектирования информационной модели наборы объектов должны быть детализированы. Например, составной объект «информация об участках дороги» с учетом специфики решаемой задачи требует разбиения на отдельные составляющие: участки, пути, отдельные пункты, километраж, план, верхнее строение пути и т. д.

Возможные трудности в определении объектов связаны с использованием постановщиками задачи:

- примеров и аналогий при описании объектов (например, вместо обобщающего понятия «работник» они могут упоминать его функции или занимаемую должность: «руководитель», «ответственный», «контролер», «заместитель»);
- синонимов (например, «допускаемая скорость» и «установленная скорость», «разработка» и «проект», «барьерное место» и «ограничение скорости»);
- омонимов (например, «программа» может обозначать компьютерную программу, план предстоящей работы или программу телепередач).

Далеко не всегда очевидно то, чем является определенный объект - сущностью, связью или атрибутом. Например, как следует классифицировать «семейный брак»? На практике это понятие можно вполне обоснованно отнести к любой из упомянутых категорий. Анализ является субъективным процессом, поэтому различные разработчики могут создавать разные, но вполне допустимые интерпретации одного и того же факта. Выбор варианта в значительной степени зависит от здравого смысла и опыта проектировщика.

Каждая сущность должна обладать некоторыми свойствами:

должна иметь уникальное имя, и к одному и тому же имени должна всегда применяться одна и та же интерпретация;

- обладать одним или несколькими атрибутами, которые либо принадлежат сущности, либо наследуются через связь;
 - обладать одним или несколькими атрибутами (первичным ключом), которые однозначно идентифицируют каждый экземпляр сущности, т. е. делают уникальной каждую строку таблицы;
- может обладать любым количеством связей с другими сущностями.

В графической нотации IDEF1X для отображения сущности используются обозначения, изображенные на рис. 9.

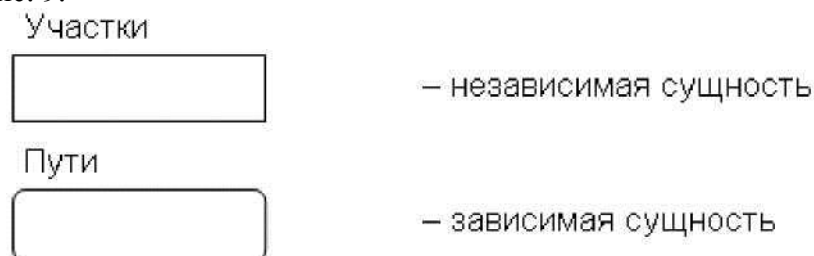


Рис. 9. Сущности

Сущность в методологии IDEF1X является *независимой (сильной, родительской, доминантной, владельцем)*, если сущность не зависит от существования другой сущности (другими словами, каждый экземпляр сущности может быть однозначно идентифицирован без определения его связей с другими сущностями, или уникальность экземпляра определяется только собственными атрибутами). Сущность называется *зависимой (слабой, дочерней, подчиненной)*, если ее существование зависит от существования других сущностей. Терминология «родительская» - «дочерняя» и «владелец» - «подчиненный» также может использоваться в отношении двух зависимых сущностей, если экземпляры одной из них (дочерней, подчиненной) могут быть однозначно определены с использованием экземпляров другой (родительской, владельца), несмотря на то, что вторая сущность в свою очередь зависит от третьей сущности.

2. Определение атрибутов.

Как правило, атрибуты указываются только для сущностей. Если у связи имеются атрибуты, то это указывает на тот факт, что связь является сущностью. Самый простой способ определения атрибутов - после идентификации сущности или связи, задать себе вопрос «Какую информацию требуется хранить о ...?». Выявленные атрибуты могут быть следующих видов:

- простой (атомарный, неделимый) - состоит из одного компонента с независимым существованием (например, «должность работника», «зарплата», «норма непогашенного ускорения», «радиус кривой» и т. д.);
- составной (псевдоатомарный) - состоит из нескольких компонентов (например, «ФИО», «адрес», и т. д.). Степень атомарности атрибутов, закладываемая в модель, определяется разработчиком. Если от системы не требуется выборки всех клиентов с фамилией Иванов или проживающих на улице Комсомольской, то составные атрибуты можно не разбивать на атомарные;
- однозначный - содержит только одно значение для одного экземпляра сущности (например, у кривой в плане может быть только одно значение радиуса, угла поворота, возвышения наружного рельса и т. д.);
- многозначный - содержит несколько значений (например, у одного отделения компании может быть несколько контактных телефонов);
- производный (вычисляемый) - значение атрибута может быть определено по значениям других атрибутов (например, «возраст» может быть определен по «дате рождения» и текущей дате, установленной на компьютере);
- ключевой - служит для уникальной идентификации экземпляра сущности (входит в состав первичного ключа);
- неключевой (описательный) - не входит в первичный ключ;
- обязательный - при вводе нового экземпляра в сущность или редактировании обязательно указывается допустимое значение атрибута, т. е. оно после редактирования не может быть неопределенным (NOT NULL).

После определения атрибутов задаются их *домены (области допустимых значений)*.

Задание доменов определяет набор допустимых значений для атрибута (нескольких атрибутов), а также тип, размер и формат атрибута (атрибутов).

На основании выделенного множества атрибутов для сущности определяется набор ключей. *Ключ* - один или несколько атрибутов сущности, служащих для однозначной идентификации ее экземпляров или для их быстрого поиска. Выделяют следующие типы ключей:

- суперключ (superkey) - атрибут или множество атрибутов, которое единственным образом идентифицирует экземпляр сущности. Суперключ может содержать «лишние» атрибуты, которые необязательны для уникальной идентификации экземпляра. При правильном проектировании структуры БД суперключом в каждой сущности (таблице) будет являться полный набор ее атрибутов;
- потенциальный ключ (potential key) - суперключ, который не содержит подмножества, также являющегося суперключом данной сущности, т. е. суперключ, содержащий минимально необходимый набор атрибутов, единственным образом идентифицирующих экземпляр сущности. Сущность может иметь несколько потенциальных ключей. Если ключ состоит из нескольких атрибутов, то он называется составным ключом. Среди всего множества потенциальных ключей для однозначной идентификации экземпляров выбирают один, так называемый первичный ключ, используемый в дальнейшем для установления связей с другими сущностями;

- первичный ключ (primary key) - потенциальный ключ, который выбран для уникальной идентификации экземпляров внутри сущности;
- альтернативные ключи (alternative key) - потенциальные ключи, которые не выбраны в качестве первичного ключа.

Если некий атрибут (набор атрибутов) присутствует в нескольких сущностях, то его наличие обычно отражает наличие связи между экземплярами этих сущностей. В каждой связи одна сущность выступает как родительская, а другая - в роли дочерней. Это означает, что один экземпляр родительской сущности может быть связан с несколькими экземплярами дочерней. Для поддержки этих связей обе сущности должны содержать наборы атрибутов, по которым они связаны. В родительской сущности это первичный ключ. В дочерней сущности для моделирования связи должен присутствовать набор атрибутов, соответствующий первичному ключу родительской. Однако здесь этот набор атрибутов уже является вторичным ключом. Данный набор атрибутов в дочерней сущности принято называть *внешним ключом (foreign key)*.

Рассмотрим пример. Пусть имеется таблица, содержащая сведения о студенте, со следующими столбцами:

- фамилия;
- имя;
- отчество;
- дата рождения;
- место рождения;
- номер группы;
- ИНН;
- номер пенсионного страхового свидетельства (НПСС);
- номер паспорта;
- дата выдачи паспорта;
- организация, выдавшая паспорт.

Для каждого экземпляра (записи) в качестве суперключа может быть выбран весь набор атрибутов. Потенциальными ключами (уникальными идентификаторами) могут быть:

- ИНН;
- номер пенсионного страхового свидетельства;
- номер паспорта.

В качестве уникального идентификатора можно было бы выбрать совокупность атрибутов «Фамилия»+«Имя»+«Отчество», если вероятность учебы в вузе двух полных тезок была бы равна нулю.

Если в сущности нет ни одной комбинации атрибутов, подходящей на роль потенциального ключа, то в сущность добавляют отдельный атрибут - *суррогатный ключ (искусственный ключ, surrogate key)*. Как правило, тип такого атрибута выбирают символьный или числовой. В некоторых СУБД имеются встроенные средства генерации и поддержания значений суррогатных ключей (например, MS Лсsez). Также стоит отметить, что некоторые разработчики вместо поиска потенциальных ключей и выбора из них первичного в каждую сущность добавляют искусственный атрибут, который в дальнейшем и используют в качестве первичного ключа.

3. Определение связей.

Наиболее характерными типами связей между сущностями являются:

- связи типа «часть-целое», определяемые обычно глаголами «состоит из», «включает» и т.п.;
- классифицирующие связи (например, «тип - подтип», «множество - элемент», «общее - частное» и т. п.);
- производственные связи (например, «начальник-подчиненный»);
- функциональные связи, определяемые обычно глаголами «производит», «влияет», «зависит от», «вычисляется по» и т. п.

Среди них выделяются только те связи, которые необходимы для удовлетворения требований к разработке БД.

Связь характеризуется следующим набором параметров:

- именем - указывается в виде глагола и определяет семантику (смысловую подоплеку)связи;

- кратностью (кардинальность, мощность): один-к-одному (1:1), один-ко-многим (1:N) и многие-ко-многим (N:M, $N = M$ или $N < M$). Кратность показывает, какое количество экземпляров одной сущности определяется экземпляром другой. Например, на одном участке (описывается строкой таблицы «Участки») может быть один, два и более путей (каждый путь описывается отдельной строкой в таблице «Пути»). В данном случае связь 1:N. Другой пример: один путь проходит через несколько отдельных пунктов и через один отдельный пункт может проходить несколько путей - связь N:M;
- типом: идентифицирующая (атрибуты одной сущности, называемые внешним ключом, входят в состав дочерней и служат для идентификации ее экземпляров, т.е. входят в ее первичный ключ) и неидентифицирующая (внешний ключ имеется в дочерней сущности, но не входит в состав первичного ключа);
- обязательностью: обязательная (при вводе нового экземпляра в дочернюю сущность заполнение атрибутов внешнего ключа обязательно и для введенных значений должен существовать экземпляр в родительской сущности) и необязательная (заполнение атрибутов внешнего ключа в экземпляре дочерней сущности необязательно или введенным значениям не соответствует экземпляр в родительской сущности);
- степенью участия - количеством сущностей, участвующих в связи. В основном между сущностями существуют бинарные связи, т. е. ассоциации, связывающие две сущности (степень участия равна 2). Например, «Участок» состоит из «Пути». В то же время по степени участия возможны следующие типы связей:
 - унарная (рекурсивная) - сущность может быть связана сама с собой. Например, в таблице «Работники» могут быть записи и по подчиненным, и по их начальникам. Тогда возможна связь «начальник» - «подчиненный», определенная на одной таблице;
 - тернарная - связывает три сущности. Например, «Студент» на «Сессии» получил «Оценку по дисциплине»;
 - кватернарная и т.д.

2.2.2. Разработка логической модели БД

Цель логического проектирования - развить концептуальное представление БД с учетом принимаемой модели БД (иерархической, сетевой, реляционной и т. д.).

Примем в качестве модели реляционную БД в третьей нормальной форме (набор нормализованных отношений с кратностью связей 1:N). Поэтому необходимо будет проверить концептуальную модель с помощью методов нормализации и контроля выполнения транзакций.

Транзакция - одно действие или их последовательность, выполняемых как единое целое одним или несколькими пользователями (прикладными программами) с целью осуществления доступа к БД и изменению ее содержимого.

1. Удаление и проверка элементов, не отвечающих принятой модели данных.

1.1. Удаление связей N:M.

Удаление связей с атрибутами.

Связи с атрибутами должны быть преобразованы в сущности.

В разработанной концептуальной модели существовала связь N:M («Раздельные пункты» : «Пути»), которая имела собственные атрибуты. После устранения связь N:M, ее атрибуты перешли в сущность «Раздельные пункты на пути» (см. рис. 7.9, неключевые атрибуты).

В графической нотации IDEF1X не предусмотрено отображение связей с атрибутами, хотя некоторые методологии ERD допускают их наличие и отображение.

1.2. Удаление сложных связей (со степенью участия более 2).

Сложную связь заменяют необходимым количеством бинарных связей 1:N со вновь созданной сущностью, которая и показывает эту связь.

1.3. Удаление рекурсивных связей (со степенью участия 1).

Рекурсивную связь заменяют, определив дополнительную сущность и необходимое количество связей (рис. 16).



Рис. 16. Замена рекурсивной связи

В данной схеме можно принять следующее правило: если в сущности «Сотрудник» атрибуты «Табельный номер» и «Табельный номер руков.» совпадают, то набор атрибутов в сущности «Сотрудник» характеризует руководителя.

1.5. Удаление многозначных атрибутов (атрибутов имеющих несколько значений). Многозначность устраняется путем введения новой сущности и связи 1:N (рис. 17).



Рис. 17. Удаление многозначных атрибутов

Данное преобразование, помимо соответствия реляционной модели данных, также позволяет хранить любое количество телефонов по одному филиалу.

1.6. Удаление избыточных связей.

Связь является избыточной, если одна и та же информация может быть получена не только через нее, но и с помощью другой связи (рис. 7.12).



Рис. 18. Избыточные связи

В приведенном примере одну из связей «Руководит» можно смело удалить (лучше между «Руководителем филиала» и «Сотрудником»).

1.7. Перепроверка связей 1:1.

В процессе определения сущностей могли быть созданы сущности, которые на самом деле являются одной. В этом случае их следует объединить. Например, в приведенном выше примере (рис. 18) сущности «Филиал» и «Руководитель филиала» лучше объединить.

В то же время не всегда можно выполнить такое объединение (рис. 19).



Рис. 19. Связи 1:1

Офисный пакет состоит из строго определенного набора компонентов, причем каждый из них характеризуется большим количеством атрибутов. Кроме этого, некоторые компоненты могут отсутствовать в офисном пакете (например, почтовый клиент) или они не входят в его состав (выступают в качестве самостоятельного продукта). В описанных случаях рекомендуется не объединять сущности.

2. Проверка модели с помощью правил нормализации.

Основная идея нормализации заключается в том, чтобы каждый факт хранился в одном месте, т. е. чтобы не было дублирования данных. Многие из требований нормализации, как правило, уже учитываются при выполнении предыдущих шагов проектирования.

Ниже приводятся краткие сведения из теории нормализации.

Проектирование реляционной БД представляет собой пошаговый процесс создания набора отношений (таблиц, сущностей), в которых отсутствуют нежелательные функциональные зависимости.

Функциональная зависимость определяется следующим образом. Пусть A и B - произвольные наборы атрибутов отношения. Тогда B функционально зависит от A ($A \rightarrow B$), в том и только в том случае, если каждому значению A соответствует в точности одно значение B . Левая часть функциональной зависимости (A) называется *детерминантом*, а правая (B) - *зависимой частью*. В частности, в отношении A может быть первичным ключом, а B - набором неключевых атрибутов, так как одному значению первичного ключа в точности соответствует одно значение набора неключевых атрибутов.

Если в БД отсутствуют нежелательные функциональные зависимости, то это обеспечивает минимальную избыточность данных, что в свою очередь ведет к уменьшению объема памяти, необходимой для хранения данных. Процесс устранения таких зависимостей получил название *нормализация*. Она выполняется в виде последовательности тестов для некоторого отношения (таблицы, сущности) с целью проверки его соответствия (или несоответствия) набору ограничений для заданной нормальной формы.

На практике нормальные формы более высоких порядков используются крайне редко. При проектировании БД, как правило, ограничиваются третьей нормальной формой, что позволяет предотвратить возможное возникновение избыточности данных и аномалии обновлений.

1NF. Отношение находится в 1NF, если на пересечении каждого столбца и строки находятся только элементарные (атомарные, неделимые) значения атрибутов.

Степень неделимости (атомарности), т. е. решение о том, следует разбивать неатомарный атрибут на атомарные или оставить его псевдоатомарным, определяется проектировщиком БД исходя из конкретных условий. Если при обработке таблиц нет необходимости различать атомарные составляющие псевдоатомарного атрибута, то его можно не делить (например, атрибуты «Фамилия, имя, отчество», «Адрес» и т. д.).

2NF. Отношение находится во 2NF, если оно находится в 1NF, и каждый неключевой атрибут характеризуется полной функциональной зависимостью от первичного ключа.

Полная функциональная зависимость определяется следующим образом. В некотором отношении атрибут B полностью зависит от атрибута A , если атрибут B функционально зависит от полного значения атрибута A и не зависит от какого-либо подмножества полного значения атрибута A .

Например, таблица «Оценки по экзаменам» характеризуется следующим набором атрибутов {Номер зачетной книжки, Дисциплина, Дата сдачи, ФИО студента, № группы, Оценка}. Очевидно, что первичным ключом является набор {Номер зачетной книжки, Дисциплина, Дата сдачи}. Полной функциональной зависимостью обладает только один неключевой атрибут «Оценка». Атрибуты «ФИО студента» и «№ группы» могут быть однозначно определены по части первичного ключа - «Номер зачетной книжки». Таким образом, требуется разбиение исходной таблицы на две (рис. 20).



Рис. 20. Обеспечение полной функциональной зависимости

3NF. Отношение находится в 3NF, если оно находится во 2NF и никакой неключевой атрибут функционально не зависит от другого неключевого атрибута, т. е. нет транзитивных зависимостей.

Транзитивная зависимость. Если для атрибутов А, В и С некоторого отношения существуют зависимости вида $A \rightarrow B$ и $B \rightarrow C$, то атрибут С транзитивно зависит от атрибута А через атрибут В.

Например, таблица «Работник» характеризуется набором атрибутов {Табельный номер, Фамилия, Имя, Отчество, Должность, Зарплата, ...}, первичный ключ - {Табельный номер}. В этой таблице от первичного ключа («Табельный номер») зависит неключевой атрибут «Должность», а от «Должности» другой неключевой атрибут «Зарплата». Для приведения к 3NF необходимо добавить новую таблицу (рис. 21).

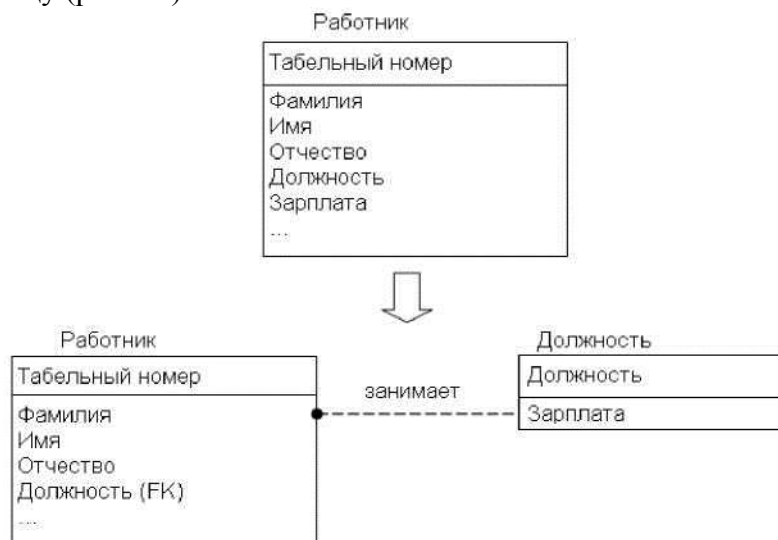


Рис. 20. Устранение транзитивной зависимости

3. Проверка выполнения транзакций.

Наиболее частыми операциями по работе с БД являются ввод, удаление и модификация записей, а также выборка данных. На основе текущей ERD необходимо проверить выполнимость и, далее, корректность выполнения всех требуемых операций по работе с БД. Примеры транзакций:

- изменение наименования отдельного пункта;
- удаление задания на расчет;
- выборка данных по отдельным пунктам расчетного пути участка и т.д.

4. Определение требований поддержки целостности данных.

Ограничения целостности данных представляют собой ограничения, которые вводятся с целью предотвращения помещения в базу противоречивых данных.

К этим ограничениям относятся:

- обязательные данные - атрибуты, которые всегда должны содержать одно из допустимых значений (NOT NULL). Например, поворот кривой (влево или вправо) должен быть обязательно задан. Обязательными также являются все атрибуты, входящие в первичный ключ сущности;

- домены - наборы допустимых значений для атрибута. Например, радиус кривой должен быть положительным числом не более 4 цифр или поворот кривой может принимать одно из двух допустимых значений - «Л» (влево) или «П» (вправо);
- бизнес-правила (бизнес-ограничения) - ограничения, принятые в рассматриваемой предметной области. Например, сумма длин переходных кривых не должна быть более длины всей кривой, километраж начала или конца кривой должен быть в пределах общего километража пути и т. д.;
- ссылочная целостность - набор ограничений, определяющих действия при вставке, обновлении и удалении записей (экземпляров сущности). Например:
- при наличии обязательной связи вставка записи в дочернюю сущность требует обязательного заполнения атрибутов внешнего ключа, и введенному значению должна соответствовать запись родительской сущности;
- аналогичное требование выдвигается при обновлении внешнего ключа в дочерней сущности;
- удаление записи из дочерней сущности или вставка записи в родительскую не вызывают нарушения ссылочной целостности;
- удаление записи в родительской сущности может требовать удаления всех связанных записей в дочерней сущности.

Автоматическая поддержка всех видов ограничений целостности возможна за счет использования операторов SQL.

Ссылочная целостность может быть обеспечена за счет использования триггеров. *Триггер* - это хранимая в БД процедура, вызываемая автоматически при выполнении удаления (DELETE), вставки (INSERT) или обновления (UPDATE) записи. Набор команд, входящих в триггер, зависит от принятой стратегии (типа триггера) поддержания целостности. Рассмотрим *типы триггеров* на примере удаления записи:

- RESTRICT (ограничение действия). Удаление записи из родительской таблицы запрещается, если в дочерней таблице существует хотя бы одна зависимая запись;
- CASCADE (каскадное удаление или обновление). При удалении записи из родительской таблицы автоматически удаляются все связанные с ней записи дочерней таблицы. Если удаляемая запись из дочерней таблицы выступает в качестве родительской стороны в некоторой другой связи, то операция удаления применяется ко всем записям дочерней таблицы этой связи и т.д.;
- SET NULL (установка неопределенного значения). При удалении записи из родительской таблицы во всех связанных с ней записях дочерней таблицы в атрибуты внешнего ключа записываются неопределенные значения (NULL). Такой тип триггера возможен только для необязательных связей;
- SET DEFAULT (установка значения по умолчанию). При удалении записи из родительской таблицы во всех связанных с ней записях дочерней таблицы в атрибуты внешнего ключа записываются заранее определенные значения по умолчанию. Такой тип триггера возможен только для необязательных связей;
- NO CHECK или NONE или IGNORE (без проверки). При удалении записи из родительской таблицы никаких действий по сохранению ссылочной целостности данных не предпринимается.

Назначение типа триггера на действия с записями является ответственной операцией. Выбор неверного типа может привести к нарушению ссылочной целостности в БД. Особой осторожности требует выбор каскадного удаления, ведь при таком удалении по цепочке могут быть удалены сотни и тысячи записей из разных таблиц.

Пример построения логической модели

На рис. 21 приведен блок «Информация об участках дороги» логической информационной модели. Данная модель соответствует третьей нормальной форме.

На рис. 21 также показаны триггеры на действия, выполняемые как со стороны родительской сущности, так и со стороны дочерней. Триггеры показаны в следующем формате «Действие : Тип триггера». Действие может быть одного из трех типов: D (DELETE), I (INSERT) и U (UPDATE). Тип триггера обозначается: C (CASCADE) и R (RESTRICT).

2.2.3. Разработка физической модели БД

Цель физического проектирования - преобразование логической модели с учетом синтаксиса, семантики и возможностей выбранной целевой СУБД.

В связи с тем, что методология физического проектирования существенно зависит от выбранной целевой СУБД, ограничимся лишь общими рекомендациями.

1. Анализ необходимости введения контролируемой избыточности.

При реализации проекта часто для достижения большей эффективности системы требуется снизить требования к уровню нормализации отношений, т.е. внести некоторую избыточность данных. Процесс внесения таких изменений в БД называется *денормализацией*.

Рассмотрим некоторые виды денормализации, которые в определенных случаях могут существенно повысить производительность системы.

1.1. Использование производных данных.

С точки зрения физического проектирования любой производный атрибут либо может сохраняться в БД, либо при каждом обращении к нему его значение будет вычисляться заново. Например, длина пути может каждый раз вычисляться по таблицам «Действительный километраж» и «Неправильные пикеты» либо храниться как атрибут в таблице «Пути».

Проектировщик при использовании производных данных должен оценить:

- дополнительную стоимость хранения производных данных и поддержки согласованности с текущими значениями тех данных, на основании которых они вычисляются, т.е. минусы хранения производных данных;
- издержки на выполнение вычислений значений производных атрибутов при каждом обращении к ним - плюсы.

1.2. Дублирование атрибутов.

1.2.1. Объединение отношений, связанных 1:1.

Даже в тех случаях, когда связь между двумя сущностями необязательная, стоит подумать об их объединении, с учетом того, что часть полей в записях не будет заполняться. Руководствоваться в таких случаях надо из тех же соображений, что и при использовании производных данных. Несмотря на избыточность, такая замена с точки зрения обработки данных и эксплуатации более выгодна.

1.2.2. Дублирование атрибутов в связях типа 1:M.

Например, при запросе к таблице «Раздельные пункты на пути» очень часто будет требоваться наименование самих раздельных пунктов. С целью уменьшения нагрузки на БД следует рассмотреть возможность включения атрибута в эту таблицу.

1.2.3. Использование служебных таблиц (справочных таблиц, классификаторов, типовых списков значений).

Служебные таблицы, как правило, создаются для атрибутов символьного типа, значения которых могут выбираться из строго определенного и ограниченного списка. Например, значениями атрибута «Род балласта» могут быть только «Щебеночный», «Песчаный», «Гравийный» и «Асбестовый».

Обычно служебные таблицы содержат два атрибута: идентификатор (код, шифр) и описание (наименование). Например, в БД можно предусмотреть служебные таблицы «Вид раздельного пункта», «Род балласта» и т. д. с атрибутами {ID, Наименование}. Эти таблицы связываются неидентифицирующей обязательной связью с исходной, при этом в ней вместо наименования параметра будет содержаться идентификатор этого наименования.

Использование служебных таблиц дает следующие преимущества:

- значительно снижается вероятность ошибки при указании значений для этих атрибутов. Если не использовать служебные таблицы, то разные пользователи могут вносить рассогласованные значения, в том числе и с ошибками.
- уменьшается размер исходной таблицы.
- если описание параметра может измениться, то значительно проще изменить одно значение в

1.2.4. Введение повторяющихся (многозначных) атрибутов.

Для достижения большей производительности при выполнении часто вызываемых запросов может быть целесообразным подход сохранения многозначных атрибутов, чем вынесение их в отдельную таблицу. Например, если количество контактных телефонов у филиала компании невелико (до 10), эта величина постоянная и не увеличится со временем, то в таблице «Филиал» можно предусмотреть атрибуты «Номер телефона 1», ..., «Номер телефона 10».

2. Перенос логической модели данных в среду целевой СУБД.

Данная стадия включает в себя проектирование таблиц и связей между ними с учетом возможностей целевой СУБД. При этом проектировщик должен хорошо ориентироваться в функциональных возможностях СУБД, а именно поддерживает ли СУБД задание:

- доменов;
- первичных, альтернативных и внешних ключей;
- неопределенных (NULL) и обязательных (NOT NULL) значений;
- значений по умолчанию (DEFAULT);
- правил контроля целостности;
- хранимых процедур и триггеров.

Кроме этого, целевая СУБД должна поддерживать требуемые типы данных или иметь возможность адекватного их хранения. Стадия переноса также включает в себя модификацию логической модели с учетом семантики и синтаксиса, принятой в целевой СУБД, а именно, соблюдение правил наименования таблиц, атрибутов, типов данных, описания триггеров, хранимых процедур и т. д.

3. Реализация бизнес-правил и анализ транзакций.

Реализацию бизнес-правил (сумма длин переходных кривых не должна быть более длины всей кривой) можно включить в SQL-операторы создания таблиц (CREATE TABLE опция CHECK для полей или таблицы в целом) или в триггеры (CREATE TRIGGER).

После реализации бизнес-правил необходимо проверить выполнимость и эффективность (время отклика, скорость выборки, объем задействованных данных) выполнения всех транзакций.

4. Разработка механизмов защиты.

Ввиду того, что работают с системой, как правило, несколько пользователей, необходимо продумать механизмы защиты данных от несанкционированного просмотра и модификации..

Ниже рассматриваются два наиболее популярных способа обеспечения защиты данных.

4.1. Разработка пользовательских представлений (видов).

Представление пользователя включает в себя данные, необходимые конкретному пользователю для принятия решений или выполнения некоторого задания.

Представление в БД - динамический результат одной или более операций, выполненных над таблицами БД с целью получения новой сводной таблицы. Представление является виртуальной таблицей, которая реально в БД не существует, но создается по запросу (SELECT) определенного пользователя в результате выполнения этого запроса.

В БД представления создаются для упрощения запросов и для организации защиты. Например, с помощью представления можно ограничить доступ к отдельным атрибутам или записям некоторых типов пользователей.

4.2. Определение прав доступа (привилегий).

В СУБД, поддерживающих SQL, возможно выполнение запросов от имени определенного пользователя, которое задается администратором БД. Каждый пользователь обладает строго определенным набором прав (привилегий) в отношении конкретной таблицы или представления. Наделение правами выполняется с помощью оператора GRANT, отмена - REVOKE. Операции, на которые можно назначить права: SELECT, INSERT, DELETE и UPDATE. Кроме того, возможно задание передачи прав от одного пользователя к другому.

5. Организация мониторинга и настройка функционирования системы.

Мониторинг функционирования и достигнутого уровня производительности системы необходим с целью устранения ошибочных проектных решений или изменения требований к системе.

2.3. Разработка сценария диалога системы с пользователем

Выбор структуры диалога

Выбор структуры диалога — это первый из этапов, который должен быть выполнен при разработке интерфейса. Рассмотренные ниже четыре варианта структуры диалога являются разновидностями структуры типа «вопрос — ответ», тем не менее каждая из них имеет свои особенности и наиболее удобна для определенного класса задач.

Диалог типа «вопрос-ответ»

Структура диалога типа «вопрос-ответ» (Q&A) основана на аналогии с обычным интервью. Система берет на себя роль интервьюера и получает информацию от пользователя в виде ответов на вопросы. Это наиболее известная структура диалога; все диалоги, управляемые компьютером, в той или иной степени состоят из вопросов, на которые пользователь отвечает. Однако в структуре Q&A этот процесс выражен явно. В каждой точке диалога система выводит в качестве подсказки один вопрос, на который пользователь дает один ответ. В зависимости от полученного ответа система может решить, какой следующий вопрос задавать. Структура Q&A предоставляет естественный механизм ввода как управляющих сообщений (команд), так и данных. Никаких ограничений на диапазон или тип входных данных, которые могут обрабатываться, не накладывается.

Диалог на основе меню

Меню является, пожалуй, наиболее популярным вариантом организации запросов на ввод данных во время диалога, управляемого компьютером

Существует несколько основных форматов представления меню на экране:

список объектов, выбираемых прямым указанием, либо указанием номера (или мнемонического кода);

- меню в виде блока данных;
- меню в виде строки данных;
- меню в виде пиктограмм.

Меню можно с равным успехом применять для ввода как управляющих сообщений, так и данных. Приемлемая структура меню зависит от его размера и организации, от способа выбора пунктов меню и реальной потребности пользователя в поддержке со стороны меню.

Структура типа меню является наиболее естественным механизмом для работы с устройствами указания и выбора: меню представляет собой изображение тех объектов, которые выбираются пользователем. Если диалог состоит исключительно из меню, можно реализовать последовательный интерфейс, при котором пользователь применяет только устройства для указания; однако такое постоянство редко достижимо на практике. Следует также заметить, что, хотя работа с этими устройствами не требует профессионального владения клавиатурой, для подготовленного пользователя это не самый быстрый способ выбора из меню. Вместо указания пользователь может сообщить о своем выборе вводом соответствующего идентификатора.

Меню — это наиболее удобная структура диалога для неподготовленных пользователей; жесткая очередность открытия и иерархическая вложенность меню может вызывать раздражение профессионала, замедлять его работу. Традиционная структура меню недостаточно гибка и не в полной мере согласуется с методами адаптации диалога, такими, например, как опережающий ввод, с помощью которого можно ускорить темп работы подготовленного пользователя. Более подробно вопросы организации и визуального представления меню рассмотрены в разделе «Проектирование элементов управления».

Диалог на основе экранных форм

Как структура типа «вопрос — ответ», так и структура типа меню предполагают обработку на каждом шаге диалога единственного ответа. Диалог на основе экранных форм допускает обработку на одном шаге диалога нескольких ответов. На практике формы используются в основном там, где учет какой-либо деятельности требует ввода достаточно стандартного набора данных. Человек, заполняющий форму, может выбирать последовательность ответов, временно пропускать некоторый вопрос, возвращаться назад для коррекции предыдущего ответа и даже «порвать бланк» и начать заполнять новый. Он работает с формой до тех пор, пока не заполнит ее полностью и не передаст системе. Если встретилась какая-либо ошибка, приложение не должно заново выводить пустую форму; выводится форма с предыдущими ответами и допущенными ошибками. Новый «бланк» выдается лишь в случае соответствующего запроса пользователя.

Таким образом, эту структуру уместно применять там, где источником данных служит существующая входная («бумажная») форма документа.

Не обязательно, чтобы внешний вид этих форм совпадал (это даже может ухудшить восприятие данных на экране), но все вводимые элементы данных должны располагаться в том же относительном порядке и иметь такой же формат, что и в исходном документе.

Часто все необходимые единицы ввода нельзя отобразить одновременно в пределах одного экрана (или окна), и их необходимо разделить на группы, которые отображаются на последовательности экранов (окон). Важно, чтобы это разбиение сохраняло логические связи и не приводило к разделению связанных частей документа.

Разработка сценария диалога

Развитие диалога во времени можно рассматривать как последовательность переходов системы из одного состояния в другое. Очевидно, что ни одно из этих состояний не должно быть «тупиковым», т.е. пользователь должен иметь возможность перейти из любого текущего состояния диалога в требуемое (за один или несколько шагов). Для этого в ходе разработки интерфейса необходимо определить все возможные состояния диалога и пути перехода из одного состояния в другое. Другими словами, необходимо разработать *сценарий диалога*.

Целями разработки сценария диалога являются:

- выявление и устранение возможных тупиковых ситуаций в ходе развития диалога;
- выбор рациональных путей перехода из одного состояния диалога в другое (из текущего в требуемое);
- выявление неоднозначных ситуаций, требующих оказания дополнительной помощи пользователю.

Сложность разработки сценария определяется в основном двумя факторами: функциональными возможностями создаваемого приложения (т.е. числом и сложностью реализуемых функций обработки информации) и степенью неопределенности возможных действий пользователя.

В свою очередь, степень неопределенности действий пользователя зависит от выбранной структуры диалога. Наибольшей детерминированностью обладает диалог на основе меню, наименьшей — диалог типа «вопрос-ответ», управляемый пользователем.

Из сказанного следует, что сценарий диалога можно упростить, снизив степень неопределенности действий пользователя. Возможными способами решения этой задачи являются:

- использование смешанной структуры диалога (применение меню с целью «ограничения свободы» пользователя там, где это возможно);
- применение входного контроля вводимой информации (команд и данных).

Дополнительные возможности по снижению неопределенности действий пользователя предоставляет объектно-ориентированный подход к разработке интерфейса, при котором для каждого объекта заранее устанавливается перечень свойств и допустимых операций. Наиболее эффективен такой подход при создании графического интерфейса; более подробно эти вопросы обсуждаются в разделе «Особенности графического интерфейса».

Сокращая число возможных состояний диалога, разработчик вместе с тем должен помнить о необходимости отражения в его сценарии работы средств поддержки пользователя, что, несомненно, делает сценарий более сложным.

Способ описания сценария диалога зависит от степени его сложности. Существующие методы описания сценариев можно разделить на две большие группы:

- неформальные методы
- формальные методы.

Главное достоинство формальных методов состоит в том, что они позволяют автоматизировать как проектирование диалога, так и его модификацию (адаптацию) в соответствии с характеристиками пользователя.

В настоящее время наиболее широко используются формальные методы описания сценариев на основе сетей Петри и их расширений, а также на основе систем представления знаний (фрейм-модели и продукционные системы).

Сценарий диалога позволяет описать процесс взаимодействия пользователя с приложением на уровне решаемой им прикладной задачи. Однако для программной реализации интерфейса такое описание носит слишком общий характер. Поэтому на этапе реализации необходимо перейти на уровень описания соответствующих процессов с помощью используемых инструментальных средств разработки приложения.

В данном пункте следует привести иерархию функций управления и обработки данных, которые призван автоматизировать разрабатываемый программный продукт. При этом можно выделить и детализировать два подмножества функций: реализующих служебные функции (например, проверки пароля, ведения календаря, архивации баз данных, тьютора и др.) и реализующих основные функции ввода первичной информации, обработки, ведения справочников, ответов на запросы и др.

При разработке структуры диалога необходимо предусмотреть возможность работы с входными документами, формирование выходных документов, корректировки вводимых данных, просмотра введенной информации, работу с файлами нормативносправочной информации, протоколирования действий пользователя, а также помощь на всех этапах работы.

Диалог в ЭИС не всегда можно формализовать в структурной форме. Как правило, диалог в явном виде реализован в тех ЭИС, которые жестко привязаны к исполнению предметной технологии. В некоторых сложных ЭИС (например, в экспертных системах) диалог не формализуется в структурной форме и тогда данный пункт может не содержать описанных схем. Описание диалога, реализованного с использованием контекстнозависимого меню не требует нестандартного подхода. Необходимо лишь однозначно определить все уровни, на которых пользователь принимает решение относительно следующего действия, а также обосновать решение об использовании именно этой технологии (описать дополнительные функции, контекстные подсказки и т.д.)

3. СОДЕРЖАНИЕ ТРЕТЬЕЙ ГЛАВЫ

В этой главе обосновывается выбор среды разработки, осуществляется разработка физической структуры (БД, сайта, АРМ, ИС и т.д.). Реализация логики(алгоритма) работы приложений в программной среде. Тестирование и наполнение требуемой реальной информацией полностью или частично. Подготавливается документация по использованию: рабочие инструкции пользователей, а также по установке и настройке.

3. Реализация

3.1. Выбор среды разработки

3.2. Реализация проекта БД

3.3. Разработка пользовательского интерфейса

3.3.1. Характеристика нормативно-справочной и входной оперативной информации

3.3.2. Характеристика результатной информации

3.4. Программное обеспечение задачи

3.5. Технология обработки информации

3.Реализация

3.1. Выбор среды разработки

Необходимо дать обоснование выбора среды реализации. Здесь же определяется технология доступа данных. Описываются не только достоинства средства разработки, но и указываются преимущества для проектируемой ИС.

3.2. Реализация БД

Необходимо разработать для выбранной СУБД даталогическую диаграмму. Для этого для каждого атрибута создается таблица вида:

Имя поля	Тип поля	Ограничения	Примечание
<Имя атрибута допустимое в выбранной СУБД>	<Тип в терминах СУБД>	<Для числовых значений на величину, для строковых количество символов>	<например, «ключевое поле», «значение по умолчанию 5» и т.д.>

В данном пункте приводятся реализация данных таблиц в СУБД, т.е. макеты таблиц в конкретной СУБД.

Здесь приводится сема данных полученная в СУБД. Определяются структуры запросов и их алгоритмы.

3.3.1. Характеристика нормативно-справочной и входной оперативной информации

Представляет собой описание состава входных документов и справочников, соответствующих им экранных форм размещения данных и структуры файлов. При этом следует уделять внимание следующим вопросам:

- при описании входных документов необходимо привести в приложении формы документов; перечень содержащихся в них первичных показателей; источник получения документа; в каком файле используется информация этого документа, описывается структура документа, число строк, объемные данные, частоту возникновения документа;
- описание экранной формы входного документа должно содержать макет экранной формы в приложении, особенностей организации рабочей и служебной зон макета, состав и содержание подсказок, необходимых пользователю для заполнения макета, перечень справочников, автоматически подключаемых при заполнении этого макета;
- описание структур входных файлов с оперативной информацией должно включать таблицу с описанием наименований полей, идентификатором каждого поля и его шаблона; по каждому файлу должна быть информация о ключевом поле, длине одной записи, числе записей в файле, частоте создания файла, длительности хранения, способе обращения (последовательный, выборочный или смешанный), способе логической и физической организации, объеме файла в байтах;
- описание структур файлов с условно-постоянной информацией содержит те же сведения, что и для файлов с оперативной информацией, но добавляются сведения о частоте актуализации файла и объеме актуализации (в процентах).

Необходимо отметить соответствие проектируемых файлов входным документам или справочникам. Описывается структура записи каждого информационного файла.

Если информационная база организована в форме базы данных, то приводится описание и других её элементов (ключей, бизнес-правил, триггеров).

3.3.2. Характеристика результатной информации

Характеристика результатной информации, один из важнейших пунктов всей проектной части, представляет собой обзор результатов решения поставленных в аналитической части задач с точки зрения предметной технологии. Если решение представляет собой формирование ведомостей (в виде экранных или печатных форм), каждую ведомость необходимо описать отдельно (в приложении следует привести заполненные экземпляры ведомостей и экранных форм документов).

В частности, какое место занимает ведомость в информационных потоках предприятия (служит для оперативного управления или для отчетности), является уточняющей или обобщающей и т. д. Каждая ведомость должна иметь итоги, не включать избыточной информации, быть универсальной. Далее приводится описание печатных форм, экранных макетов с перечислением и краткой характеристикой содержащихся показателей (см. описание входных документов и их экранных форм), для каждого документа указывается, на основе каких файлов получается этот документ.

Если результатная информация предоставляется не в виде ведомостей (например, при проектировании подсистемы распределенной обработки данных), необходимо подробно описать ее дальнейший путь, основываясь на имеющейся организации многопользовательской ЭИС.

Файлы с результатной и промежуточной информацией описываются по той же схеме, что и файлы с первичной информацией.

3.4. Программное обеспечение задачи

Пункт **программного обеспечения** включает общие положения, отражающие стандарты, а также требования к аппаратным и программным ресурсам для успешной эксплуатации программного средства. Здесь же приводится описание использованных средств разработки. Затем производится характеристика архитектуры проектируемого программного средства и представляется структурной схемой пакета (деревом вызова процедур и программ). После чего производится описание программных модулей и файлов.

3.4.1. Структурная схема пакета (дерево вызова процедур и программ)

На основе результатов, полученных в предыдущем пункте, строится дерево программных модулей, отражающих структурную схему пакета, содержащей программные модули различных классов:

- выполняющие служебные функции;
- управляющие модули, предназначенные для загрузки меню и передачи управления другому модулю;
- модули, связанные с вводом, хранением, обработкой и выдачей информации.

В данном пункте необходимо для каждого модуля указать идентификатор и выполняемые функции.

В случае проектирования программного обеспечения АРМ для корпоративной ЭИС следует дополнительно рассмотреть состав транзакций и типовых процедур ведения корпоративных баз данных.

3.4.2. Описание программных модулей

Описание программных модулей должно включать блок - схемы и описание блок- схем алгоритмов основных расчетных модулей (объемом не менее 500 операторов).

3.4.3. Схема взаимосвязи программных модулей и информационных файлов

Схема взаимосвязи программных модулей и информационных файлов отражает взаимосвязь программного и информационного обеспечения комплекса задач, и может быть представлен несколькими схемами, каждая из которых соответствует определенному режиму. Главная же часть, представляется одним блоком с указателями схем режимов.

В Заключении к курсовой работе даются общие итоги проведенного исследования/разработки, обобщаются результаты и выводы, указываются конкретные достоинства разработки, ее практическая ценность. Могут быть указаны перспективы и направления дальнейшей разработки темы.

Заключение должно иметь объем 1-2 листа машинописного текста.

Приложения к курсовой работе формируются автором работы и служат для иллюстрации отдельных положений исследуемой темы или являются практическим результатом проектирования: исходные данные (входные и выходные формы документов, результаты моделирования бизнес-процессов, коды, скрипты, листинги, результаты документирования процессов автоматизации, разработанные автором формы документов и т.п.).

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОФОРМЛЕНИЮ КУРСОВОЙ РАБОТЫ

1. Курсовая работа представляется руководителю в распечатанном сброшюрованном виде (в папке со скоросшивателем).

2. Курсовая работа оформляется на стандартных листах белой бумаги формата А4 (210*297 мм).

3. Текст курсовой работы должен быть отпечатан на компьютере с использованием редактора Microsoft Word, шрифт «Times New Roman», размер шрифта – 14, межстрочный интервал – полуторный, интервал между абзацами – 0.

4. Текст курсовой работы, таблицы и иллюстрации следует располагать на листах, соблюдая следующие размеры полей: левое поле – 30 мм, правое поле – 10 мм, верхнее поле – 20 мм, нижнее поле – 20 мм, выравнивание текста «по ширине» (двухстороннее выравнивание).

5. Нумерация страниц – сквозная, начиная с титульного листа. Непосредственно на титульном листе (приложение 3) и оглавлении номер страницы не ставится, номера последующих страниц проставляются в низу по центру арабскими цифрами (размер шрифта – 10) без точки в конце. В приложениях страницы не нумеруются.

6. Названия структурных элементов работы и глав основной части располагаются на отдельных строках и выполняются жирным шрифтом, прописными (заглавными) буквами (ОГЛАВЛЕНИЕ, ВВЕДЕНИЕ и т.д.), без переносов и с выравниванием по центру. Подчеркивать заголовки не следует. Точку в конце заголовка ставить не нужно. Заголовки параграфов печатают строчными буквами (первая – прописная), располагая по всей ширине страницы и без точки в конце. Заголовки параграфов выделяются жирным шрифтом. Заголовок не должен состоять из нескольких предложений, переносы слов в заголовках не допускаются. Расстояние между заголовками глав и параграфов не допускается.

7. Пример оформления оглавления представлен в приложении 4.

8. Каждую главу основной части следует начинать с новой страницы. Параграфы продолжают на текущей странице. Расстояние между параграфами – 2 строки.

9. Структурным элементам работы номер не присваивается, т.е. части работы «ОГЛАВЛЕНИЕ», «ВВЕДЕНИЕ», «ЗАКЛЮЧЕНИЕ» и т.п. порядкового номера не имеют. Нумерации подлежат только главы и параграфы в рамках основной части работы.

10. Главы основной части нумеруются арабскими цифрами в пределах всей работы. Параграфы должны иметь нумерацию в пределах каждой главы. Номер параграфа состоит из номера главы и номера параграфа, разделенных точкой (например, 1.1). В конце номера параграфа точка не ставится.

11. Абзацный отступ (отступ в начальной строке текста абзаца) должен составлять 1,25 см.

12. Текст работы должен быть четким, законченным, понятным. Орфография и пунктуация текста должны соответствовать ныне действующим правилам.

В тексте работ не следует:

- применять для одного и того же понятия различные термины, даже близкие по смыслу, а также иностранные термины при наличии равнозначных по смыслу терминов в русском языке;
- применять сокращения слов, не расшифрованные в перечне сокращений, условных обозначений, символов, единиц и терминов, кроме установленных правилами орфографии и пунктуации, а также соответствующими нормативными документами (стандартами и т.п.);
- использовать сокращенные обозначения единиц измерения величин, если они в тексте употребляются без цифр, за исключением единиц измерения в таблицах и в расшифровках буквенных обозначений, входящих в формулы;
- употреблять математические знаки без цифр (например, $<$, $>$, $=$, $/$, №, %);
- использовать в тексте математические знаки $-$ (минус) перед отрицательными величинами и $+$ (плюс) перед положительными величинами. Вместо этих знаков необходимо писать соответственно слова «минус», «плюс»;
- употреблять аббревиатуры стандартов, методических указаний, руководящих документов и т.п. без регистрационных номеров.
- использовать жирный шрифт и курсив, кроме выделения структурных элементов работы.

13. Числовые значения величин в тексте следует указывать с необходимой степенью точности. При этом числа с размерностью необходимо писать цифрами, а без размерности – словами (например, цена 10 руб., цена повысилась в сто раз).

14. Иллюстрации (чертежи, графики, схемы, диаграммы, фотоснимки, рисунки) объединяются единым названием «рисунок». Характер иллюстрации может быть указан в ее названии (например, Рис.1. Блок-схема алгоритма...). При необходимости перед названием рисунка помещают поясняющие данные. Иллюстрации следует нумеровать арабскими цифрами порядковой нумерацией в пределах всей работы (сквозная нумерация в рамках курсовой работы). Нумерация рисунков в рамках глав, а тем более параграфов не допускается. Пример оформления рисунка приведен ниже.

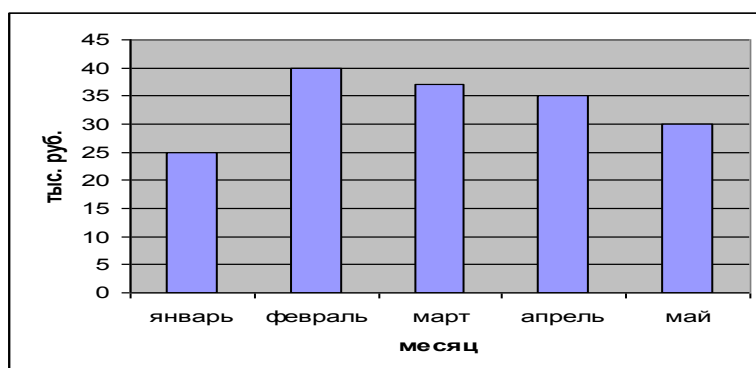


Рис. 1. Динамика стоимости запасов фирмы за пять месяцев 20__ г.

15. Таблицы следует нумеровать арабскими цифрами сквозной нумерацией в пределах всей работы. Нумерация таблиц в рамках глав, а тем более параграфов не допускается. Номер следует размещать в правом верхнем углу над заголовком таблицы после слова «Таблица».

Каждая таблица должна иметь заголовок, который помещается ниже слова «Таблица» и располагается по центру. Слово «Таблица» и заголовок начинаются с прописной буквы, точка в конце заголовка не ставится.

Заголовки граф таблицы должны начинаться с прописных букв, подзаголовки – со строчных, если последние подчиняются заголовку. Заголовки граф указываются в единственном числе, точки в конце заголовков не ставятся.

Таблицу следует размещать так, чтобы читать её без поворота работы. Если такое размещение невозможно, таблицу располагают так, чтобы ее можно было читать, поворачивая работу по часовой стрелке.

При переносе таблицы «шапку» таблицы следует повторить, над ней размещают слова «Продолжение таблицы» с указанием ее номера. Если «шапка» таблицы велика, допускается ее не повторять; в этом случае следует пронумеровать графы и повторить их нумерацию на следующей странице. Заголовок граф таблицы не повторяют.

Если все показатели, приведенные в таблице, выражены в одной и той же единице измерения, то её обозначение помещается над таблицей, например, в конце заголовка.

Если цифровые или иные данные в какой-либо строке таблицы отсутствуют, то ставится прочерк. Повторяющийся в строках графы текст можно заменять кавычками (если текст – из одного слова) или при первом повторении словами «То же», а далее кавычками. Ставить кавычки при повторении цифровых данных, математических и иных символов не допускается.

Пример оформления таблицы приведен ниже (см. табл.2).

Таблица 2

Сведения о заемщиках ПАО «Банк»

Наименование заемщика	Сумма выданного кредита (тыс. руб.)	Сумма обеспечения (тыс. руб.)	Коэффициент покрытия	Коэффициент ликвидности
ООО «Миф»	100	140	1,5	3,4
АО «Удача»	80	200	2,3	5,6

Если в работе только одна иллюстрация либо только одна таблица, их нумеровать не следует.

Рисунки и таблицы следует располагать в работе непосредственно после текста, в котором они упоминаются впервые, или на следующей странице, если в указанном месте они не помещаются.

Иллюстрации вместе с их названиями, а также таблицы вместе с их реквизитами должны быть отделены от основного текста снизу и сверху пробелами с одинарным межстрочным интервалом.

В поле иллюстраций и в таблице допускается более мелкий шрифт текста, чем основной текст, но не менее шрифта №10, а также меньший межстрочный интервал.

На все иллюстрации и таблицы должны быть ссылки в тексте работы (например: «на рис.5 показано...», «в соответствии с данными табл.2» и т.п.).

После любых иллюстраций или таблиц обязательно должен быть сделан вывод по данным иллюстрации или таблицы.

16. Уравнения и формулы следует выделять из текста в отдельную строку с отделением от текста пробелами в один межстрочный интервал сверху и снизу. Если уравнение не умещается в одну строку, оно должно быть перенесено после знака равенства (=), или после знака плюс (+), или после других математических знаков с их обязательным повторением в новой строке.

Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, как и в формуле. Значение каждого символа и числового коэффициента следует давать с новой строки, первую строку пояснения начинают со слова «где» без двоеточия.

Формулы и уравнения в работе следует нумеровать порядковой нумерацией в пределах всей работы или текущей главы арабскими цифрами в круглых скобках с правой стороны напротив формулы. Допускается нумерация только тех формул, на которые есть ссылки в тексте.

Ссылки в тексте на порядковые номера формул даются в круглых скобках, например, «... в формуле (1)».

Если в работе только одна формула или уравнение, то их не нумеруют.

Формулы должны быть выполнены с помощью редактора формул WORD.

17. В работе обязательно должны быть ссылки на используемые источники информации. В целях упрощения работы по формированию текста курсовой работы рекомендуется сноски на источники информации помещать в тексте в квадратных скобках. В таких сносках указываются номер источника информации, присвоенный ему в списке литературы, а также страница в источнике, на которой находится цитата или цифра (например - [5, с. 26], где 5 – номер источника в списке источников информации; с. 26 – страница, на которой находится цитата или цифровой материал).

18. При использовании цитат автор работы обязан сверить их с первоисточниками. Цитаты необходимо приводить с соблюдением правил правописания, пунктуации и выделений (курсив, подчеркивание и т.п.) источника. Можно использовать не прямое цитирование, то есть пересказывать мысли авторов своими словами. Однако и в этом случае, кроме точного и корректного изложения чужих мыслей, также необходимо дать ссылку на источник.

19. Список источников информации должен формироваться в алфавитном порядке по фамилии авторов. Литература обычно группируется в списке в такой последовательности:

- законодательные и нормативно-методические документы и материалы;
- специальная научная отечественная и зарубежная литература (монографии, учебники, научные статьи и т.п.);
- статистические, инструктивные и отчетные материалы предприятий, организаций и учреждений;
- электронные ресурсы.

Включенная в список литература нумеруется сплошным порядком от первого до последнего названия.

По каждому литературному источнику указываются: автор (или группа авторов), полное название книги или статьи, место и наименование издательства (для книг и брошюр), год издания; для журнальных статей указывается название журнала, год выпуска и номер. По сборникам трудов (статей) указывается автор статьи, ее название и далее название книги (сборника) и ее выходные данные.

Пример оформления списка источников информации представлен в приложении 4.

20. Приложения следует оформлять как продолжение курсовой работы. Каждое приложение должно начинаться с новой страницы. Вверху страницы справа указывается слово «ПРИЛОЖЕНИЕ» и его номер. Приложение должно иметь заголовок, который располагается по центру листа отдельной строкой и печатается прописными буквами. Приложения следует нумеровать порядковой нумерацией арабскими цифрами. На все приложения в тексте работы должны быть ссылки. Располагать приложения следует в порядке появления ссылок на них в тексте. Если в качестве приложения используется конкретный документ или бланк формы документа, имеющий самостоятельное значение, его вкладывают в работу без изменений по сравнению с оригиналом, проставив на титульном листе в правом верхнем углу слово «Приложение» и его номер. Страницы приложений не нумеруются.

3. КОНСУЛЬТИРОВАНИЕ ПРИ ПОДГОТОВКЕ КУРСОВОЙ РАБОТЫ

В процессе подготовки курсовой работы студенту необходимы консультации (советы и разъяснения) руководителя - преподавателя дисциплины.

Первая консультация посвящается выбору темы и определению плана (оглавления) курсовой работы, она должна произойти после выбора и утверждения темы курсовой работы. Последующие консультации осуществляются по мере необходимости.

К консультации студенту надо готовиться. Задаваемые вопросы должны быть конкретными, их следует готовить заранее, фиксировать в письменном виде и за несколько дней до консультации передавать преподавателю. Это существенно повышает эффективность консультаций, так как у преподавателя появляется возможность дать более обдуманные и развернутые советы и разъяснения, особенно, когда дело касается дискуссионных или недостаточно исследованных вопросов.

В процессе консультаций тема и план курсовой работы могут быть уточнены или изменены.

4. ЗАЩИТА КУРСОВОЙ РАБОТЫ

Подведение итогов написания курсовой работы состоит из следующих этапов:

- сдача курсовой работы на проверку руководителю;

- доработка курсовой работы с учетом замечаний руководителя;
- сдача в электронном виде готовой курсовой работы для проверки в системе «Антиплагиат»;
- сдача готовой курсовой работы на защиту;
- защита курсовой работы.

Защиту курсовой работы принимает научный руководитель - преподаватель дисциплины «Финансы предприятий (организаций)», который оценивает, качество выполнения и оформления работы, а также содержательность доклада и ответов на вопросы. В случае необходимости, по просьбе студента, на защите может присутствовать (с правом оценки) дополнительный преподаватель (заведующий профильной кафедрой).

На защите курсовой работы студент кратко, в течение 5 минут (максимум), докладывает об актуальности выбранной темы, объекте, предмете, цели и задачах исследования, основных выводах, полученных в результате проведенного исследования.

Краткий доклад целесообразно подготовить в письменном виде, но выступать на защите следует свободно, не зачитывая, а лишь пользуясь при необходимости подготовленным текстом.

По окончании доклада студенту задаются вопросы по содержанию курсовой работы. Отвечая на вопросы, студент должен давать короткие и исчерпывающие ответы.

Курсовая работа оценивается по следующей системе.

Оценка «отлично» выставляется за курсовую работу, которая носит исследовательский характер, содержит грамотно изложенный материал, с соответствующими выводами и обоснованными предложениями. При его защите студент показывает глубокие знания темы, свободно оперирует данными исследования, легко отвечает на поставленные вопросы.

Оценка «хорошо» выставляется за грамотно выполненную во всех отношениях курсовую работу при наличии небольших недочетов в его содержании, оформлении или защите. Например, выдвигаемые студентом предложения носят не вполне обоснованный характер, или он не очень уверенно (хотя и верно) отвечает на поставленные вопросы.

Оценка «удовлетворительно» выставляется за курсовую работу, которая удовлетворяет всем предъявляемым требованиям, но отличается поверхностностью, в ней просматривается непоследовательность изложения материала, представлены необоснованные выводы и предложения. При ее защите студент проявляет неуверенность, показывает слабое знание темы, не дает полного аргументированного ответа на заданные вопросы.

Оценка «неудовлетворительно» выставляется за курсовую работу, которая не носит исследовательского характера, не содержит анализа и практического исследования деятельности объекта, нет выводов и предложений. При защите работы студент затрудняется отвечать на поставленные вопросы по его теме, не знает теории вопроса, при ответе допускает существенные ошибки.

Типичные ошибки, допускаемые студентами при написании курсовых работ:

- работа представляет собой фактически конспекты найденных студентом публикаций по теме курсовой работы. Отсутствует как связь между различными частями работы, так и попытка самостоятельного критического анализа и сопоставления точек зрения различных исследователей по данной проблеме;

- в работе не выделяются теоретические положения, подлежащие последующей верификации; она носит чисто описательный характер; следует отметить, что формулировка подходов, позволяющих эмпирически подтвердить или опровергнуть выдвигаемые гипотезы, является весьма важной частью курсовых работ; хотя в силу объективных трудностей студентам часто бывает не под силу осуществить проверку своих гипотез, тем не менее, без определения того, *каким образом это необходимо сделать*, курсовую работу нельзя считать выполненной;

- при изучении научных публикаций по данной теме студенты пользуются устаревшей литературой.

Студент, не представивший в установленный срок готовую курсовую работу по дисциплине учебного плана или не защитивший ее, считается имеющим академическую задолженность и не допускается к сдаче зачета или экзамена по данной дисциплине.

**ПРИМЕРНАЯ ТЕМАТИКА КУРСОВЫХ РАБОТ ПО ДИСЦИПЛИНЕ
«ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ»**

Тема курсовой работы должна быть сформулирована как проектирование информационной системы в конкретной предметной области. Примерный перечень предметных областей представлен ниже.

1. Учет работников предприятия.
2. Расчет заработной платы работников предприятия.
3. Учет готовой продукции на складе предприятия.
4. Учет рабочего времени на промышленном предприятии.
5. Учет электронных закупок.
6. Электронная коммерция (Интернет-магазин).
7. Учет товаров на складе для оптовой торговли.
8. Работа автосалона.
9. Учет договоров на предприятии.
10. Учет торговых операций.
11. Поставка товаров в торговую фирму.
12. Учет материальных ценностей библиотеки.
13. Работа рекламного агентства.
14. Учет командировочных удостоверений на предприятии.
15. Учет коммунальных платежей для управляющей компании.
16. Учет успеваемости студентами ВУЗа.
17. Учет абонентов телефонной связи.
18. Учет компьютеров на предприятии.
19. Учет актов выполненных работ.
20. Учет кассовых операций.

Тема может быть предложена самостоятельно студентом по согласованию с научным руководителем работы.

Приложение 2
Образец заявления на выбор темы курсовой работы
Зав. кафедрой _____ СГТИ

_____ от студента ___ курса
направления подготовки: 09.03.03 Прикладная информатика
направленность (профиль): «Прикладная информатика в экономике»
очной формы обучения

заявление.

Прошу утвердить следующую тему курсовой работы по дисциплине «Проектирование информационных систем»: _____

и назначить научным руководителем _____
(Ф.И.О., ученая степень, ученое звание, должность)

« _____ » _____ 20__ г.

_____/_____/_____
(подпись)



**ЧАСТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СРЕДНЕРУССКИЙ ГУМАНИТАРНО-ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»**

Кафедра _____

КУРСОВАЯ РАБОТА

по дисциплине: «Проектирование информационных систем»

на тему: _____

Выполнил (а) студент (ка) _____ курса
Направление подготовки: 09.03.03 Прикладная
информатика
Направленность (профиль): «Прикладная
информатика в экономике»
Очной формы обучения

Преподаватель:

(Ф.И.О., ученая степень, ученое звание, должность)

ПРИМЕРНЫЕ ТЕМЫ КУРСОВЫХ РАБОТ

Тема курсовой работы должна быть сформулирована как проектирование информационной системы в конкретной предметной области. Примерный перечень предметных областей представлен ниже.

21. Учет работников предприятия.
22. Расчет заработной платы работников предприятия.
23. Учет готовой продукции на складе предприятия.
24. Учет рабочего времени на промышленном предприятии.
25. Учет электронных закупок.
26. Электронная коммерция (Интернет-магазин).
27. Учет товаров на складе для оптовой торговли.
28. Работа автосалона.
29. Учет договоров на предприятии.
30. Учет торговых операций.
31. Поставка товаров в торговую фирму.
32. Учет материальных ценностей библиотеки.
33. Работа рекламного агентства.
34. Учет командировочных удостоверений на предприятии.
35. Учет коммунальных платежей для управляющей компании.
36. Учет успеваемости студентами ВУЗа.
37. Учет абонентов телефонной связи.
38. Учет компьютеров на предприятии.
39. Учет актов выполненных работ.
40. Учет кассовых операций.

Тема может быть предложена самостоятельно студентом по согласованию с научным руководителем работы.