

1. СИСТЕМЫ БАЗ ДАННЫХ. ОСНОВНЫЕ ПОНЯТИЯ

1.1. Система баз данных. Определения

Предшественниками систем баз данных являются *файловые системы* – наборы программ, выполняющие для пользователей некоторые операции, например, ввод данных, их обработку и формирование некоторых отчетов. Файловым системам присущи следующие ограничения: 1) разделение и изоляция данных, содержащихся в отдельных файлах; 2) дублирование данных в файлах; 3) зависимость структуры программ от данных; 4) несовместимость форматов файлов; 5) слабая адаптация приложения к изменению требований(фиксированные запросы); :6) быстрое увеличение объема приложения. Названные ограничения файловых систем являются следствиями двух факторов: определение данных содержится внутри приложений, а не хранятся отдельно и независимо от них; помимо приложений не предусмотрено никаких других инструментов доступа к данным и их обработки.

К.Дейт [1] определяет *систему баз данных* как «компьютеризированную систему, основное назначение которой – хранить информацию, предоставляя пользователям средства ее извлечения и модификации. К информации может относиться все, что заслуживает отдельного пользователя или организации, использующей систему».

База данных – совместно используемый набор структурированных, логически связанных данных, предназначенный для удовлетворения информационных потребностей пользователей организации.

База данных – это единое, интегрированное, большое хранилище данных, однократно определяемое и затем используемое одновременно многими пользователями. Подход, основанный на применении баз данных, при котором определение данных отделено от приложений (*абстрагирование данных*) позволяет пользователям видеть только внешнее определение объекта, не заботясь о его определении и функционировании

СУБД (*система управления базами данных*) – программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, осуществлять контролируемый доступ к ней.

Система баз данных включает сами по себе данные, сохраняемые в базе данных, аппаратное обеспечение, программное обеспечение, в том числе и СУБД, пользователей различных категорий (прикладные программисты, конечные пользователи, администратор базы данных).

Система баз данных имеет ряд преимуществ, наиболее важным из которых является физическая независимость данных от пользовательских приложений (иммунитет прикладных программ к изменениям способа хранения данных и используемых методов доступа). К другим преимуществам систем баз данных относятся:

- возможность совместного доступа к данным нескольких существующих приложений базы данных;
- сокращение избыточности данных вследствие интеграции данных;
- устранение противоречивости данных;
- обеспечение целостности (корректности) данных в базе;
- возможность поддержки транзакций (транзакция – логическая единица работы, включающая несколько операций базы данных, являющаяся важным механизмом поддержания целостности базы данных);
- организация защиты данных;
- возможность балансирования противоречивых требований;
- возможность введения стандартизации данных.

1.2. Архитектура систем баз данных

Рассмотрим теперь *архитектуру* систем баз данных, называемую *ANSI/SPARC*, с достаточной точностью описывающую практически все реальные системы баз данных (Рис. 1). Архитектура ANSI/SPARC включает три уровня: *внутренний*, *внешний* и *концептуальный*. Внешний уровень (пользовательский) наиболее близок к пользователям, т.е. связан со способами представления данных для отдельных пользователей. Внутренний уровень (физический) наиболее близок к физическому хранению информации, т.е. связан со способами хранения данных на физических носителях. Концептуальный уровень (логический) является промежуточным между двумя первыми.

Если внешний уровень связан с индивидуальными представлениями пользователей, то концептуальный уровень связан с обобщенным представлением пользователей. То есть, может существовать несколько внешних представлений, каждое из которых состоит из более или менее абстрактного представления определенной части базы данных и только одно концептуальное представление, состоящее из абстрактного представления базы данных в целом. Также существует единственное внутренне представление, отражающее способ физического хранения всей базы данных.

У каждого пользователя есть свой язык общения: это может быть язык программирования (C++, C#, Java и т.д.), язык запросов или язык специального назначения. Все эти языки включают подъязык данных, т.е. такое подмножество операторов языка, которое связано с объектами базы данных и операциями с ними. Подъязык данных встроен в базовый язык,

дополнительно обеспечивающий различные возможности: вычислительные, логические операции, обработка переменных и т.д.

Подъязык данных является комбинацией двух языков – **языка определения данных (ЯОД)**, который поддерживает определения и объявления объектов базы данных, и **языка манипулирования данными (ЯМД)**, который поддерживает операции над объектами базы или их обработку.

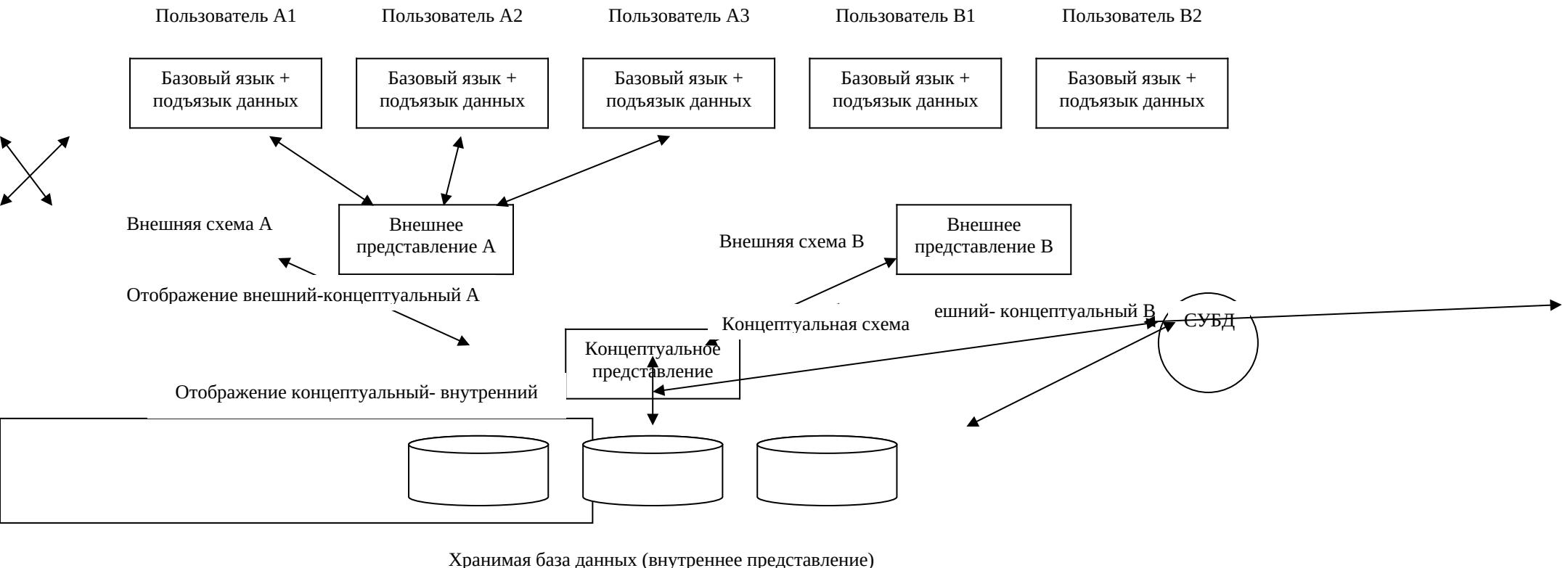


Рис.1. Схема архитектуры системы баз данных (ANSI/SPARC)

Отдельного пользователя интересует лишь некоторая часть всей базы данных. Представление пользователя об этой части будет абстрактным по сравнению с выбранным способом физического хранения данных. В соответствии с терминологией ANSI/SPARC представление отдельного пользователя называется ***внешним представлением***. Таким образом, внешнее представление – это содержимое базы данных таким, каким его видит определенный пользователь. В общем случае внешнее представление состоит из некоторого множества экземпляров каждого типа внешних записей. Подъязык данных всегда определяется в терминах внешних записей. Каждое внешнее представление определяется посредством ***внешней схемы***, которая записывается с помощью внешнего языка определения данных.

Концептуальное представление – это представление всей информации базы данных в более абстрактной форме, представление данных такими, какими они являются на самом деле, а не такими, какими их видит пользователь. Концептуальное представление состоит из некоторого множества экземпляров каждого из существующих типов концептуальных записей. Концептуальная запись не обязательно совпадает с внешней или хранимой записями. Концептуальное представление определяется с помощью ***концептуальной схемы***, записанной на концептуальном языке определения данных. Концептуальное представление – это представление всего содержимого базы данных, а концептуальная схема – это определение такого представления.

Внутреннее представление – это низкоуровневое представление всей базы данных как базы, состоящей из некоторого множества экземпляров каждого из существующих типов внутренних записей. Внутренне представление отделено от физического уровня, поскольку в нем не рассматриваются физические записи, называемые блоками или страницами, и физические области устройства хранения, называемые цилиндрами и дорожками. Внутреннее представление описывается с помощью внутренней схемы, которая определяет не только различные типы хранимых записей, но также существующие индексы, способы представления хранимых полей, физическую упорядоченность хранимых записей и т.д. ***Внутренняя схема*** записывается с использованием внутреннего языка определения данных.

Приведенная архитектура кроме элементов самих трех уровней, включает ***отображения***: отображение концептуального уровня на внутренний и несколько отображений внешних уровней на концептуальный. Отображение ***«концептуальный-внутренний»*** устанавливает соответствие между концептуальным представлением и хранимой базой данных, т.е. описывает как концептуальные записи и поля представлены на внутреннем уровне. При изменении структуры хранимой базы отображение ***«концептуальный-внутренний»*** также

изменится таким образом, чтобы концептуальная схема осталась неизменной. Т.е. чтобы обеспечить независимость данных результаты внесения любых изменений в схему хранения не должны быть видны на концептуальном уровне.

Отображение «внешний-концептуальный» определяет соответствие между некоторым внешним представлением и концептуальным представлением. Отображение «внешний-концептуальный» является основой достижения логической независимости данных.

1.3. Роль СУБД и АБД в архитектуре

Система баз данных содержит корпоративные ресурсы, которыми следует управлять и контролировать. **Администратор данных** (АД) отвечает за управление данными, включая планирование базы данных, разработку и сопровождение стандартов, бизнес-правил и деловых процедур, а также за концептуальное и логическое проектирование базы данных. **Администратор базы данных** (АБД) отвечает за физическую реализацию базы данных, включая физическое проектирование и воплощение проекта, за обеспечение безопасности и целостности данных, за сопровождение системы, обеспечение максимальной производительности приложений. По сравнению с АД, обязанности АБД носят технический характер, ему необходимы знания конкретной СУБД и системного окружения.

Рассмотрим основные функции АД и АБД.

- *Определение концептуальной схемы.* АД решает, какая именно информация должна храниться в базе данных, т.е. осуществляет логическое проектирование базы данных. После того, как содержимое базы данных на абстрактном уровне определено АД, АБД создает соответствующую концептуальную схему, используя концептуальный язык определения данных. Объектная форма этой схемы используется СУБД для получения ответов на запросы, связанные с доступом к данным. Исходная форма будет храниться в Словаре данных системы в роли справочника для пользователей.
- *Определение внутренней схемы.* АБД должен решить, как данные будут представлены в хранимой базе данных, т.е. осуществить физическое проектирование базы данных. После завершения физического проектирования АБД создает соответствующие структуры хранения, используя внутренний язык определения данных. Кроме того он определяет соответствующее отображение между внутренней и концептуальной схемами. Как и в предыдущем случае, концептуальная схема, внутренняя схема и отображение хранятся в исходной и откомпилированной формах.
- *Взаимодействие с пользователями.* В обязанности АБД входит взаимодействие с пользователями для предоставления им необходимых данных и написания требуемых внешних схем с

применением внешнего языка определения данных. Кроме того, необходимо определить отображение между каждой созданной внешней и концептуальной схемами. Другие аспекты взаимодействия с пользователями включают консультации по разработке приложений, обеспечение требуемого технического образования, предоставление помощи в выявлении и устранении возникающих проблем и прочие виды профессионального обслуживания.

- *Определение требований защиты и обеспечение целостности данных.* Задача обеспечения целостности заключается в гарантированной поддержке корректности данных в базе. Требования защиты и поддержания целостности данных рассматриваются как часть концептуальной схемы.
- *Определение процедур резервного копирования и восстановления.* При использовании на предприятии системы баз данных в случае повреждения базы данных вследствие ошибки человека, отказа оборудования, сбоя программного обеспечения важно иметь возможность восстановить утраченные данные с минимальной задержкой и с наименьшими последствиями.
- *Управление производительностью и реагирование на изменяющиеся требования.* АБД отвечает за такую организацию системы, при которой можно получить производительность, оптимальную для всего предприятия в целом, а также за коррекцию работы системы в соответствии с изменяющимися требованиями.

1.4. СУБД и ее функции

Система управления базой данных (СУБД) представляет собой программное обеспечение, управляющее доступом ко всей базе данных. В целом назначением СУБД является предоставление пользовательского интерфейса с системой баз данных. Основные функции СУБД:

- *Обработка запросов пользователей к базе данных.* Обработка запросов осуществляется следующим образом:
 - пользователь выдает запрос на доступ к данным, применяя определенный подъязык данных (например, SQL);
 - СУБД перехватывает запрос и анализирует его;
 - СУБД последовательно просматривает внешнюю схему, отображение «внешний-концептуальный», концептуальную схему, отображение «концептуальный-внутренний», определение структуры хранения, определяет местоположение требуемых данных;
 - СУБД, используя методы доступа операционной системы, выполняет необходимые операции в хранимой базе данных.

При выполнении операций используется механизм защиты данных от несанкционированного доступа.

- *Определение данных.* СУБД должна предоставлять средства определения данных (внешних схем, концептуальной схемы, внутренней схемы, отображений) в исходной и объектной

(откомпилированной) формах. Иначе говоря, СУБД должна включать в себя компоненты процессора (компилятора) ЯОД (языка определения данных) для каждого из поддерживаемых ею языков определения данных.

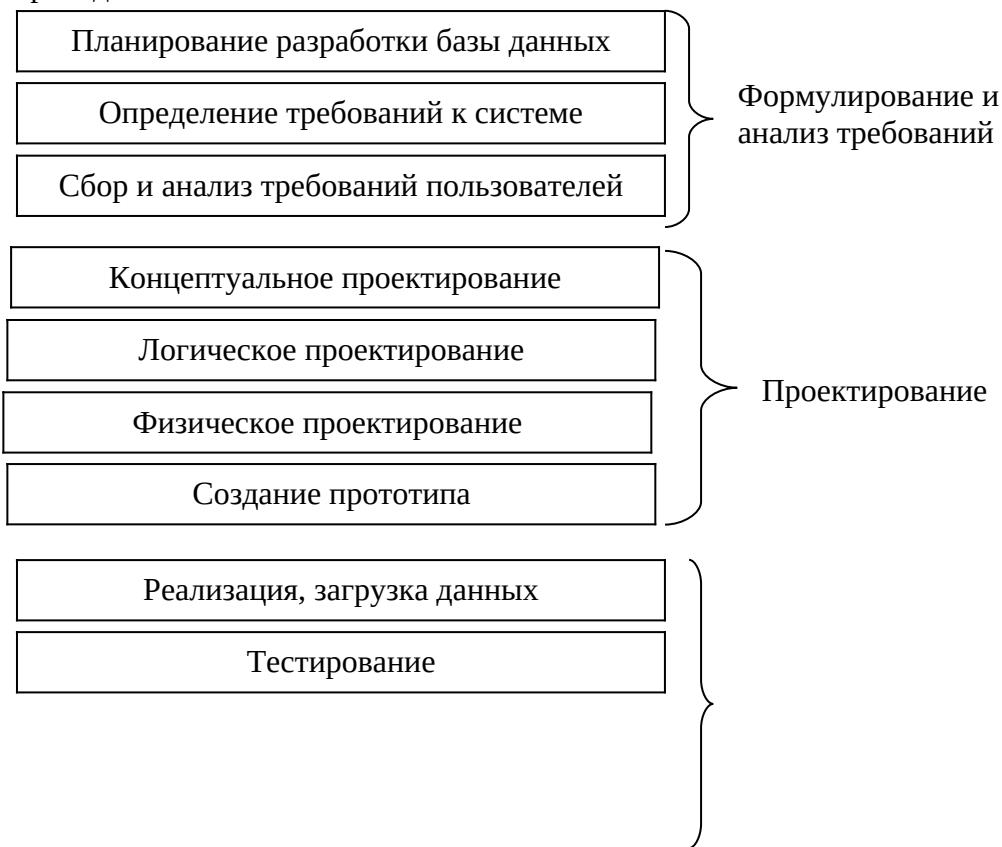
- *Обработка данных.* СУБД должна уметь обрабатывать запросы пользователя на выборку, изменение, добавление или удаление данных. Т.е., СУБД должна включать в себя компонент процессора (компилятора) ЯМЛ, обеспечивающего поддержку языка манипулирования данными.
- *Оптимизация запросов.* Запросы ЯМЛ должны быть обработаны оптимизатором, назначение которого состоит в поиске достаточно эффективного способа выполнения каждого из запросов.
- *Защита и сохранение целостности данных.* СУБД должна контролировать пользовательские запросы и пресекать любые попытки нарушения ограничений защиты и сохранения целостности данных, определенные АБД.
- *Восстановление данных и поддержка параллельности.* СУБД содержит компонент, называемый менеджером транзакций, который должен обеспечивать необходимый контроль над восстановлением данных и управлением параллельностью обработки (при распределении данных).
- *Ведения Словаря данных.* СУБД должна обеспечивать функцию ведения Словаря данных. Словарь данных содержит «данные о данных» (метаданные), т.е. определение объектов системы. Например, в Словаре данных сохраняются исходные и объектные формы всех существующих схем, отображений, установленные ограничения защиты и целостности данных, описания существующих приложений.
- *Обеспечение производительности.* СУБД должна выполнять все указанные функции с максимально возможной эффективностью.

2. ПРОЕКТИРОВАНИЕ СИСТЕМ БАЗ ДАННЫХ

2.1. Жизненный цикл системы баз данных

Проектирование информационных систем является скорее искусством, чем наукой, хотя существуют методики проектирования систем. *Методология проектирования* может рассматриваться как совокупность методов и средств, последовательное применение которых обеспечивает разработку проекта базы данных, удовлетворяющего целям проектирования. *Задачей процесса проектирования* является построение структуры базы данных, адекватно отражающей описываемую проблемную среду, реализуемой с помощью технических и программных средств.

Организационную основу проектирования системы баз данных составляет *жизненный цикл системы*. Можно выделить три основные фазы проектирования: 1) *формулирование и анализ требований*; 2) *проектирование* и 3) *реализация, эксплуатация и сопровождение*. Каждая фаза, в свою очередь, делится на этапы. Этапы не являются строго последовательными, а включают некоторое число шагов в виде циклов обратной связи. Этапы жизненного цикла приложения баз данных приведены на Рис.3.



Реализация,

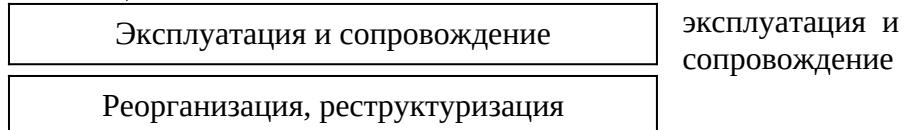


Рис.3. Жизненный цикл системы баз данных

2.2. Этапы проектирования системы БД

Перечислим основные действия, выполняемые на каждом этапе жизненного цикла базы данных:

- ◆ *Планирование разработки БД* – планирование самого эффективного способа реализации этапов жизненного цикла системы. Планирование разработки БД связано с общей стратегией построения ИС организации. Суть стратегии заключается в решении основных задач: а) определение бизнес-планов и целей организации; б) оценка показателей уже существующих ИС с целью выявления их сильных и слабых сторон; в) оценка возможностей использования ИТ для достижения конкурентноспособного преимущества. Для поддержки планирования разработки БД может быть создана корпоративная модель данных, отображающая наиболее важные данные и связи между ними, а также их отношение к различным функциональным сферам организации. Обычно корпоративная модель данных имеет вид *упрощенной ER-диаграммы*.
- ◆ *Определение требований к системе* – определение диапазона действия и границ системы, состава пользователей и областей применения.
- ◆ *Сбор и анализ требований пользователей* – сбор и анализ информации о той части организации, работа которой будет поддерживаться с помощью создаваемого приложения БД, а также использование этой информации для определения требований пользователей к системе. Необходимая информация может быть собрана следующими способами: а) посредством опроса сотрудников предприятия, особенно специалистов; б) с помощью наблюдения за деятельностью предприятия; в) посредством изучения документов; г) с помощью анкетирования пользователей; д) за счет использования опыта проектирования других подобных систем. Собранныя информация может быть плохо структурирована и включать неформальные заявления пользователей, которые впоследствии потребуется преобразовать и представить в виде более четко структурированных требований. Это достигается путем использования методов составления спецификаций требований, например, с помощью *диаграмм потоков данных* (DFD).
- ◆ *Концептуальное проектирование* имеет целью построение независимой от СУБД информационной структуры путем объединения информационных требований пользователей. Результат

проектирования часто представляется в виде *диаграммы «сущность-связь»* или *ER-диаграммы*.

- ◆ *Логическое проектирование* обеспечивает построение СУБД-ориентированной модели данных. Первоначально осуществляется выбор СУБД, далее строится соответствующая модель данных. В качестве такой модели может использоваться *реляционная модель данных*. Также на этом этапе осуществляется проектирование интерфейса пользователя, транзакций, прикладных программ, предназначенных для работы с БД.
- ◆ *Физическое проектирование* заключается в выборе физической структуры базы данных. Результатом физического проектирования является полностью готовая к реализации структура БД. На этом этапе также производится окончательная отладка программных модулей, определенных ранее.
- ◆ *Создание прототипов* – создание рабочей модели приложения базы данных. Прототип – это рабочая модель, обладающая лишь частью требуемых возможностей и не обеспечивающая всей функциональности готовой системы. Таким образом, прототип представляет собой инструмент, позволяющий в значительной степени прояснить требования пользователей и для них самих и для разработчиков системы, а также оценить гибкость разработанного проекта БД.
- ◆ *Реализация, загрузка данных* – физическая реализация базы данных и разработанных приложений. Реализация базы данных и написанных приложений осуществляется посредством создания ее описания на языке определения данных (ЯОД) целевой СУБД. Загрузка данных – заполнение данными БД, ввод их с помощью оператора. При замене новой базы данных старой осуществляется конвертирование и загрузка данных – перенос любых существующих данных в новую базу данных и модификация любых существующих приложений с целью организации совместной работы с новой базой данных.
- ◆ *Тестирование* – процесс выполнения прикладных программ с целью поиска ошибок. Прежде чем использовать новую систему на практике, ее следует тщательно протестировать. Это можно сделать путем выработки последовательной и методически продуманной стратегии тестирования с использованием реальных данных. При успешном проведении тестирования вскрываются имеющиеся ошибки, подтверждается соответствие системы БД спецификациям и требованиям пользователей. Кроме того, осуществляется сбор статистики, позволяющий установить показатели надежности и качества созданного программного обеспечения.
- ◆ *Эксплуатация и сопровождение* – наблюдение за системой и поддержка ее нормального функционирования по окончании развертывания. Этот этап включает выполнение таких действий как,

контроль производительности системы, сбор статистических данных о функционировании системы.

- ◆ *Реорганизация, реструктуризация* предусматривает внесение в реализованный проект изменений, возникающих вследствие новых требований, анализа функционирования системы и анализа мнений пользователей о работе системы. Целью этого этапа является оптимизация функционирования существующей системы путем реорганизации базы данных и/или внесения изменений в программное обеспечение. Программная адаптация подразумевает процесс модификации прикладных программ в случае проведения реструктуризации базы данных.

Существует два основных подхода к проектированию систем баз данных: «нисходящий» и «восходящий». При *восходящем* подходе работа начинается с самого нижнего уровня – уровня определения атрибутов (свойств сущностей), которые на основе анализа существующих между ними связей группируются в отношения, представляющие типы сущностей и связи между ними. Восходящий подход лучше всего подходит для проектирования простых баз данных с небольшим числом атрибутов. Более подходящей стратегией проектирования сложных баз данных является использование *нисходящего* подхода. Начинается этот подход с разработки моделей данных, содержащих несколько высокоуровневых сущностей и связей, а затем работа продолжается в виде серии нисходящих уточнений низкоуровневых сущностей, связей и относящихся к ним атрибутов.

Рассмотрим один из методов концептуального проектирования систем баз данных: семантическое моделирование.

3. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

3.1. Семантическое моделирование

Общий подход к проблеме семантического моделирования характеризуется четырьмя основными этапами:

1. Прежде всего выявляется некоторое множество семантических концепций, понятий, называемых «сущностями» которые могут быть полезны при неформальном обсуждении реального мира. Сущности могут быть классифицированы по разным типам сущностей и тогда все сущности определенного типа обладают некоторыми общими свойствами. Каждая сущность обладает некоторым особым свойством, предназначенным для ее идентификации. Каждая сущность может быть связана с другими сущностями посредством некоторых связей.

2. Далее определяется набор соответствующих символических, формальных объектов, которые могут использоваться для представления описанных семантических концепций.

3. Определяется набор формальных общих правил целостности, предназначенных для работы с такими формальными объектами.

4. Также определяется набор формальных операторов, предназначенных для манипулирования формальными объектами.

Необходимо подчеркнуть, что правила целостности и операторы являются такой же частью модели данных, как и объекты.

На первом этапе пытаются выделить множество семантических концепций, полезных для описания предметной области (части реального мира, имеющей интерес для проектировщика ИС). Некоторые из этих концепций, а именно – сущности, свойства, связи и подтипы представлены в Табл. 1.

Понятие	Неформальное определение	Примеры
СУЩНОСТЬ (Entity)	Некоторый отличимый объект, явление, процесс	Поставщик, деталь, поставка Работник, отдел, человек Произведение, концерт, оркестр Заказ на поставку, серия заказов
СВОЙСТВО (Property)	Элемент данных, описывающий характеристику сущности	Номер поставщика, количество поставки Отдел работника, оклад работника Тип концерта Дата заказа
СВЯЗЬ (Relationship)	Связь, служащая для обеспечения взаимодействия между двумя или более другими сущностями	Поставка (поставщик-деталь) Должность (работник-отдел) Запись(произведение–оркестр–дирижер)
ПОДТИП	Сущность типа Y является	«Работник» является подтипом

(Subtype)	подтипов сущности типа X тогда, когда каждый экземпляр сущности типа Y обязательно является экземпляром сущности типа X	сущности «Человек» «Концерт» является подтипов сущности «Произведение»
-----------	---	--

Табл. 1. Некоторые семантические концепции

3.2. Модель «сущность-связь»

Цель моделирования данных состоит в обеспечении разработчика ИС концептуальной схемой базы данных в форме одной или нескольких моделей, которые относительно легко могут быть отображены в любую систему баз данных. Наиболее распространенным средством моделирования данных являются модели "сущность-связь" (ER-модели), нотация которых впервые введена П. Ченом в 1976 г. ER- модель включает аналоги всех семантических объектов, представленных в Табл.1. Ченом была предложена не только ER-модель, но и соответствующая ей технология построения диаграмм, получивших

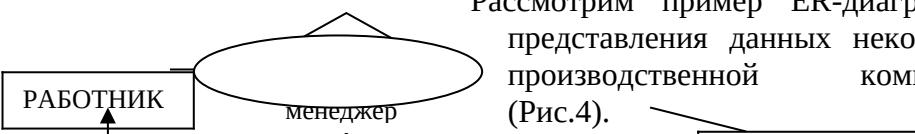


название «ER-диаграммы».

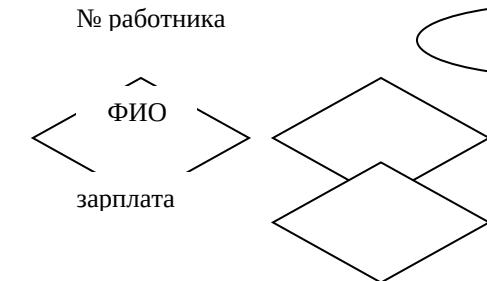
работает

работает в

работает над



Рассмотрим пример ER-диаграммы представления данных некоторой производственной компании (Рис.4).

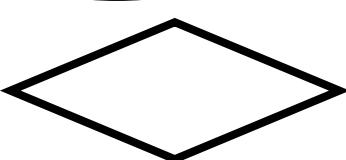


ПОСТАВЩИК

Наименование

№ детали

ДЕТАЛЬ



имеет
подчиненног

ПОДЧИНЕННЫЙ

Рис.4. Пример диаграммы «сущность-связь»

Базовыми понятиями ER-диаграмм являются:

Сущность (Entity) — реальный или абстрактный объект, явление или процесс, имеющий существенное значение для рассматриваемой предметной области.

Каждая сущность должна обладать уникальным идентификатором (ами), благодаря чему каждый экземпляр сущности однозначно идентифицируется и отличается от всех других экземпляров данного *типа* сущности. Каждая сущность должна обладать свойствами:

- ◆ иметь уникальное имя;
- ◆ обладать одним или несколькими атрибутами, которые принадлежат сущности;
- ◆ обладать одним или несколькими идентифицирующими атрибутами, однозначно идентифицирующими каждый экземпляр сущности.

Сущности могут подразделяться на *сильные* и *слабые*. *Слабой* называется такая сущность, существование которой зависит от другой сущности, т.е. она не может существовать, если этой другой сущности не существует. Например, на Рис.4 сущность «ПОДЧИНЕННЫЙ» (ПОДЧИНЕННЫЙ РАБОТНИК) является слабой, поскольку она не может существовать, если не существует соответствующей сущности «РАБОТНИК». В частности, если сведения о некотором работнике (сущность «РАБОТНИК») будут удалены, то и сведения обо всех зависящих от него работниках (сущность «ПОДЧИНЕННЫЙ») также будут удалены. *Сильной* называется сущность, которая не является слабой.

Сущности обладают некоторыми свойствами. Все сущности одного и того же типа обладают некоторыми общими свойствами. **Атрибут** — любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для классификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Атрибут представляет тип характеристик или свойств, ассоциированных со множеством реальных или абстрактных объектов (людей, мест, событий, состояний, идей, предметов и т.д.). Экземпляр атрибута — это определенная характеристика отдельного элемента множества. Экземпляр атрибута определяется типом характеристики и ее значением, называемым значением атрибута. В ER-модели атрибуты ассоциируются

с конкретными сущностями. Таким образом, экземпляр сущности должен обладать единственным определенным значением для ассоциированного атрибута.

Каждая сущность может обладать любым количеством связей с другими сущностями модели.

Связь (Relationship) — поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. Связь — это ассоциация между сущностями, при которой каждый экземпляр одной сущности ассоциирован с произвольным количеством экземпляров второй сущности, и наоборот. Тип связи рассматривается между типами сущностей, а конкретный экземпляр связи рассматриваемого типа существует между конкретными экземплярами рассматриваемых типов сущностей. Связь может быть обязательной, возможной, условной. Чаще всего встречаются связи между двумя типами сущностей (бинарные связи), хотя в модели могут быть выделены связи между любым количеством сущностей (тернарные, ... n-арные).

Мощность связи представляет собой отношение количества экземпляров одной сущности к соответствующему количеству экземпляров другой сущности. Связи в модели «сущность-связь» могут иметь мощность: «один-к-одному» (1:1), «один-ко-многим» (1:M) и «многие-ко-многим» (M:N).

Связь 1:1 (**«один-к-одному»**) — это связь между двумя типами сущностей А и В, при которой каждому экземпляру сущности А соответствует один и только один экземпляр сущности В.

Связь 1:M (**«один-ко-многим»**) — это связь между двумя типами сущностей А и В, при которой одному экземпляру сущности А может соответствовать ноль, один или несколько экземпляров сущности В, но каждому экземпляру сущности В соответствует только один экземпляр сущности А.

Связь M:N (**«многие-ко-многим»**) — это связь между двумя типами сущностей А и В, при которой каждому экземпляру сущности А может соответствовать ноль, один или несколько экземпляров сущности В и наоборот.

Информацию о концептуальной модели оформляют составлением спецификаций по сущностям, атрибутам и отношениям между ними и представляют графическую диаграмму «сущность-связь». При моделировании используются следующие правила:

- ◆ Можно использовать только следующие типы конструктивных элементов:

КНИГА

сущность;

Авторы

атрибут;



- ◆ Соединяются отдельные блоки ребрами и дугами. Неориентированные ребра используются для обозначения связей между 3 и более сущностями, а также для фиксирования принадлежащих сущностям атрибутов. Для связей между двумя типами сущностей используют направленные дуги.
- ◆ В отдельном проектном представлении каждый компонент информации моделируется только одним конструктивным элементом (элемент информации не может быть одновременно, например, сущностью и атрибутом или связью).
- ◆ Не допускается дублирование, повторное вхождение атрибутов во многие сущности.
- ◆ На ER-диаграмме отмечаются только связи между сущностями и не указываются связи типа «сущность-атрибут», «атрибут-атрибут».
- ◆ У каждой сущности выделяется идентифицирующий атрибут.
- ◆ Атрибуты могут быть не только у сущностей, но и у связей.

Каждая сущность имеет по крайней мере один тип, однако у некоторой сущности может быть несколько типов. Например, если некоторые работники являются программистами (и все программисты являются работниками), то можно сказать, что тип сущности «ПРОГРАММИСТ» является *подтипом* типа сущности «РАБОТНИК», или тип сущности «РАБОТНИК» является *супертипов* типа сущности «ПРОГРАММИСТ». Программисты автоматически обладают всеми свойствами работников, однако обратное не верно. Аналогично сущность «ПРОГРАММИСТ» автоматически участвует во всех связях, в которых участвует сущность «РАБОТНИК», но обратное также неверно. Поэтому говорят, что подтип *наследует* свойства и связи супертипа. Для некоторых типов сущностей можно образовать целую иерархию типов сущностей, представленную, например, на Рис.5.



Рис. 5. Пример иерархии типов сущностей

Следует отметить, что ER-диаграммы используются в качестве основы в любой методике проектирования, а не только в семантическом моделировании.

3.3. Пример проектирования ИС «Факультет» на основе семантического моделирования

Рассмотрим пример разработки учебного проекта информационной системы «Факультет». Создадим систему баз данных, содержащую информацию о личных данных студентов, изучаемых студентами предметах, результатах сдачи экзаменов и зачетов, преподавателях и их учебной нагрузке. Отметим, что проект носит учебный характер, а это значит, что реальные процессы могут быть идеализированы или упрощены.

Учебный процесс в образовательном учреждении (университет, институт и пр.) по определенной специальности строится в соответствии с государственным образовательным стандартом. На его основе вузом разрабатывается учебный план по специальности, в котором учебные дисциплины группируются в блоки и определяется трудоемкость каждой дисциплины, их распределение по семестрам, объем аудиторных занятий и самостоятельной работы, формы итогового контроля.

Факультет вуза организационно включает два взаимодействующих контингента: студентов и преподавателей. Студенты обучаются в течение определенного срока обучения на разных курсах, объединяясь в учебные группы. Процесс обучения учебной группы (курса) определенному предмету протекает под руководством преподавателя. Преподаватели работают на кафедрах вуза и в соответствии с их учебно-научным направлением в каждом году имеют определенную учебную нагрузку.

Учебный процесс организуется деканатом факультета путем составления расписания учебных занятий. Деканат осуществляет также контроль за проведением учебных занятий преподавателями, посещением студентами занятий, степенью усвоения студентами учебной программы по предметам, регулирует процесс сдачи студентами зачетов и экзаменов в сессию. После зачисления студента в вуз формируется «Личное дело», хранимое в студенческом отделе кадров, в деканате также известна информация анкетного характера о студентах и преподавателях.

Основной документооборот деканата осуществляется методистом факультета, который ведет учет успеваемости студентов, составляет расписание учебных занятий, выдает разнообразные справки и т.д. Декан составляет учебный план по специальности, утверждает учебную нагрузку преподавателей кафедр, принимает управленические решения по организации учебного процесса на факультете. Управление факультетом осуществляется деканом путем выдачи распоряжений, приказов и т.д. Напоминаем, что рассматривается упрощенный вариант, модель реального факультета вуза.

Рассмотрим далее образцы форм, документов, используемых в документообороте деканата факультета вуза.

Форма 1. Анкета студента

Анкета студента

Вуз _____ Факультет _____ Год поступления_____

ФИО студента _____ Дата рождения _____

Домашний адрес _____

Сведения о родителях _____

Примечания

Фото

Дата заполнения_____

Подпись_____

Форма 2. Анкета преподавателя

Анкета преподавателя

Вуз _____ Факультет _____ Кафедра _____

ФИО _____ Дата рождения _____

Домашний адрес _____

Ученая степень, звание _____

Примечания

Фото

Дата заполнения_____

Подпись_____

Форма 3. Список учебной группы

Вуз _____ Факультет_____
 Учебный год _____ Семестр _____
 Специальность _____ Курс ____ Группа _____

Список учебной группы

№	Фамилия И.О.	№ зачетной книжки

Форма 4. Состав кафедры

Вуз _____ Факультет_____
 Учебный год _____ Семестр _____
 Название кафедры _____ Специализация кафедры _____

Список преподавателей кафедры

№	Фамилия И.О.	Должность

Форма 5. Учебный план

Вуз _____ Факультет_____
 Учебный год _____ Специальность_____

Учебный план

Предмет	Описание предмета	Вид предмета (лекция, практич.)	Объем в часах

Форма 6. Расписание учебных занятий

Вуз _____ Факультет_____
 Учебный год _____ Семестр_____
 Специальность_____ Курс _____ Группа _____

Расписание занятий

День недели	№ пары	Предмет, ФИО преподавателя, № аудитории

Форма 7. Учебная нагрузка преподавателя

Вуз _____ Факультет_____ Кафедра _____
 Учебный год _____ Семестр_____
ФИО _____ Должность _____ Уч.степнь, звание_____

Учебная нагрузка преподавателя

Предмет	Специальность	Курс	Вид предмета	Объем в час.

Форма 8. Зачетная/ экзаменационная ведомость

Вуз _____	Факультет _____																				
Учебный год _____	Семестр _____																				
Специальность _____	Курс _____ Группа _____																				
Предмет _____	ФИО преподавателя _____																				
Дата экзамена/зачета _____																					
Экзаменационная ведомость																					
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>№</th> <th>ФИО студента</th> <th>№ зачетной книжки</th> <th>Оценка</th> <th>Подпись</th> </tr> </thead> <tbody> <tr><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td></tr> </tbody> </table>		№	ФИО студента	№ зачетной книжки	Оценка	Подпись															
№	ФИО студента	№ зачетной книжки	Оценка	Подпись																	

Форма 9. Журнал успеваемости студентов

Вуз _____	Факультет _____		
Учебный год _____	Семестр _____		
Специальность _____	Курс _____ Группа _____		
Журнал успеваемости учебной группы			
ФИО студента	Предметы		
	Предмет 1	Предмет 2	Предмет 3
ФИО	Оценка	Оценка	Оценка

Рис.6. Формы документов ИС «Факультет»

Цель разработки ИС: автоматизация работы методиста деканата факультета, актуальное и достоверное предоставление информации декану для принятия решений. Осуществление компьютеризированного хранения и обработки информации: ввод данных в формы документов, формирование регламентных запросов, вывод фиксированных отчетов.

Определим требования к системе:

Производственные задачи:

1) представление данных о структурном составе факультета (контингенты студентов и преподавателей);

2) ввод, обработка и выдача информации о студентах факультета и их успеваемости;

3) –// – // – о преподавателях и их учебной нагрузке;

4) – // – // – о расписании учебных занятий;

5) – // – // – об итоговой аттестации (данные о зачетах и экзаменах);

Управленческие задачи:

1) формирование приказов и распоряжений, необходимых для управления учебным процессом факультета. В рамках учебного проекта рассматриваться не будет.

Приведем упрощенную информационную схему системы «Факультет».

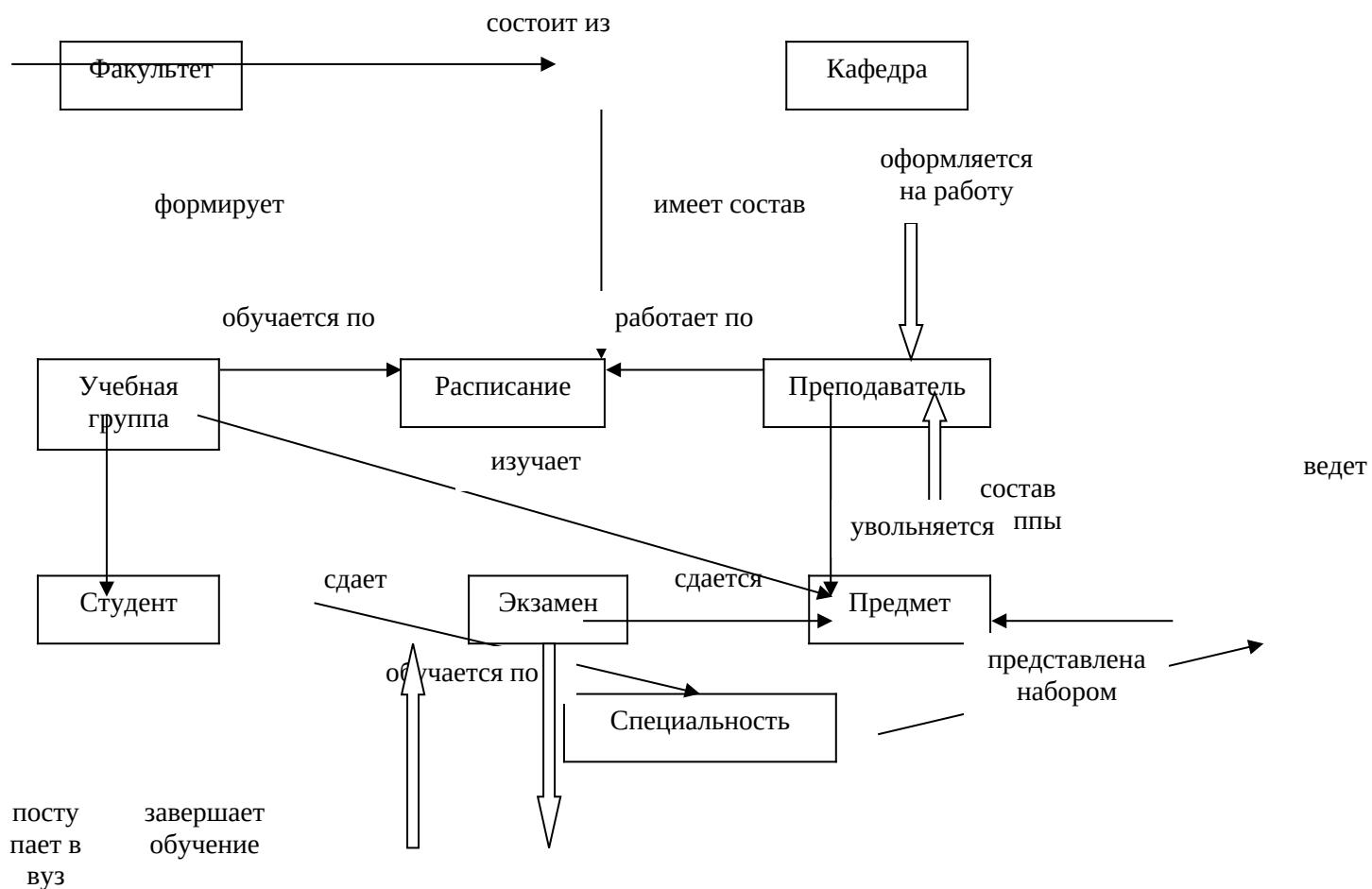


Рис. 7. Информационная схема учебного проекта «Факультет»

Необходимая для проектирования информация может быть собрана обычными способами:

- а) путем опроса потенциальных пользователей системы: декана факультета, методиста деканата, заведующих кафедрами;
- б) путем анализа, изучения, наблюдений за деятельностью факультета;
- в) за счет имеющегося опыта проектирования.

Собранныю информацию можно представить в виде диаграммы потоков данных (DFD).

На следующем этапе производим проектирование базы данных. Основными целями проектирования базы данных являются:

- ◆ представление данных и связей между ними, необходимых для основных областей применения данной системы и основных пользователей;
- ◆ создание модели данных, способной поддерживать выполнение требуемых транзакций обработки данных;
- ◆ разработка предварительного варианта проекта, структура которого позволит удовлетворить основные требования, предъявляемые к системе.

Для проектирования системы будем использовать восходящий метод проектирования.

Первоначально составим список информационных единиц (атрибутов или элементов данных) проектируемой ИС. Это необходимо для того, чтобы:

- а) иметь единый перечень информационных единиц;
- б) исключить синонимы;
- в) определить важные, возможно, ключевые элементы данных;
- г) определить атрибуты, реально отсутствующие в документах, но необходимые с точки зрения практического применения в базах данных;
- д) определить спецификации элементов данных (тип, диапазон возможных значений, ограничения на принимаемые значения).

Составленный список элементов данных приведен в Табл. 2, в которой указаны наименования атрибутов и отмечены «+» те формы документов, в которых каждый из атрибутов встречается.

№	Наименование	Ф1	Ф2	Ф3	Ф4	Ф5	Ф6	Ф7	Ф8	Ф9
---	--------------	----	----	----	----	----	----	----	----	----

	элемента данных								
1	ВУЗ	+	+	+	+	+	+	+	+
2	Название факультета	+	+	+	+	+	+	+	+
3	Название кафедры		+				+		
4	Специализация кафедры			+					
5	Учебный год			+	+	+	+	+	+
6	Семестр			+			+	+	+
7	Шифр специальности			+		+	+	+	+
8	Курс			+			+	+	+
9	Группа			+			+	+	+
10	ФИО студента	+		+				+	+
11	Фото студента	+							
12	Дата рождения студента	+							
13	Родители	+							
14	Домашний адрес студ.	+							
15	Дата поступления	+							
16	Примечания студента	+							
17	Дата анкеты студента	+							
18	№ зачетной книжки		+					+	
19	Наименование предмета				+	+	+	+	+
20	Вид предмета				+		+		
21	Описание предмета				+				
22	Объем в часах по виду				+				
23	Объем в часах преподав.						+		
24	ФИО преподавателя	+		+		+	+	+	
25	Фото преподавателя	+							
26	Адрес преподавателя	+							
27	Дата рождения препод.	+							
28	Ученая степень, звание	+					+		
29	Дата анкеты преподав.	+							
30	Примечания преподав.	+							
31	День недели					+			
32	Номер пары					+			
33	Номер аудитории					+			
34	Должность			+			+		
35	Оценка						+	+	
36	Дата экзамена						+		
37	Код студента								
38	Код преподавателя								
39	Код предмета								
40	Код группы								
41	Код кафедры								
42	Код факультета								

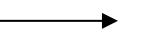
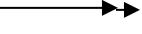
Табл. 2. Список атрибутов ИС «Факультет»

Анализ Табл.2 показывает, что атрибуты «вуз», «факультет», «учебный год», «семестр», «шифр специальности», «курс», «группа» встречаются практически в каждой форме документа. Это свидетельствует о том, что данные атрибуты являются определяющими, важными, основными, т.е. войдут в состав ключа. Для того, чтобы избежать длинных составных ключей удобно ввести дополнительные атрибуты, такие как «код студента». Дополнительные атрибуты приведены в строках 37-42 Табл. 2. Далее в Табл. 3 приведем описание некоторых из элементов данных.

№	Наименование элемента данных	Идентификатор	Описание	Спецификация	Примечание
1	ФИО студента	ФИО_студ	Фамилия, имя, отчество студента	строка длиной до 30 символов	возможный ключ
2	Код студента	Код_студ	Уникальный номер студента	длинное целое число	ключ
3	Фото студента	Фото_студ	Фотография студента	объект OLE (картина, граф. образ)	
4	День недели	День_недели	Название дня недели	строка длиной 12 символов	варианты значений: понед, вторник, ...
5	Наименование предмета	Предмет	Название предмета учебного плана	строка длиной до 50 символов	возможный ключ

Табл. 3. Описания некоторых элементов данных ИС «Факультет»

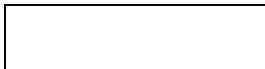
Определим далее взаимосвязи, существующие между группами элементов данных. В качестве основы для формирования взаимосвязей используем: а) информационную схему, приведенную на Рис. 7; б) основные формы документов Ф1–Ф9, приведенные на Рис.6; в) выявленные знания о предметной области (факультет вуза). Применяем следующие виды связей:

- а) 1:1  ;
- б)  1:M ;
- в)  M:N .

- 1) Код  студента ФИО студента, Фото студента, Дата рождения студента, Родители, Домашний адрес студента, Дата поступления, Примечания студента, № зачетной книжки, Дата анкеты студента

- 2) ВУЗ ↔ Название факультета
- 3) Код ↔ факультета ВУЗ, Название
факультета
- 4) Код факультета, Учебный год ↔
Код кафедры
- 5) Код ↔ группы Код факультета, Учебный год,
Семестр, Группа, Курс, Шифр специальности
- 6) Код группы ↔ Код студента
- 7) Код ↔ преподавателя ФИО
преподавателя, Адрес преподавателя, Дата рождения преподавателя,
Ученая степень и звание, Дата анкеты преподавателя, Примечания
преподавателя
- 8) Код ↔ кафедры Название кафедры,
Специализация кафедры, Учебный год
- 9) Код кафедры ↔ Код преподавателя
- 10) Код кафедры, Код ↔ преподавателя
Должность
- 11) Код предмета ↔ Наименование предмета, Вид
предмета, Описание предмета, Объем в часах по виду
- 12) Код студента, Код предмета, Код преподавателя, Дата экзамена
↔ Оценка
- 13) Код преподавателя, Код предмета, Код факультета, Учебный год,
↔ Семестр Объем в часах
преподавателя
- 14) Код группы, День недели, Номер пары ↔ Код
предмета, Код преподавателя, Номер аудитории
- 15) Код преподавателя, День недели, Номер ↔ пары
Код группы, Код предмета, Номер аудитории

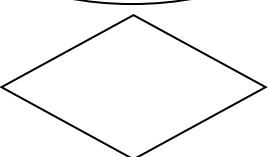
Далее выделим сущности, имеющие место, на наш взгляд, в представлении предметной области: «СТУДЕНТ», «ПРЕПОДАВАТЕЛЬ», «ПРЕДМЕТ», «ГРУППА», «КАФЕДРА», «ФАКУЛЬТЕТ». Построим диаграмму «сущность-связь» (ER-диаграмму), используя блоки:



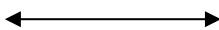
для обозначения сущностей,



для обозначения характеристик сущностей (атрибутов или элементов данных)

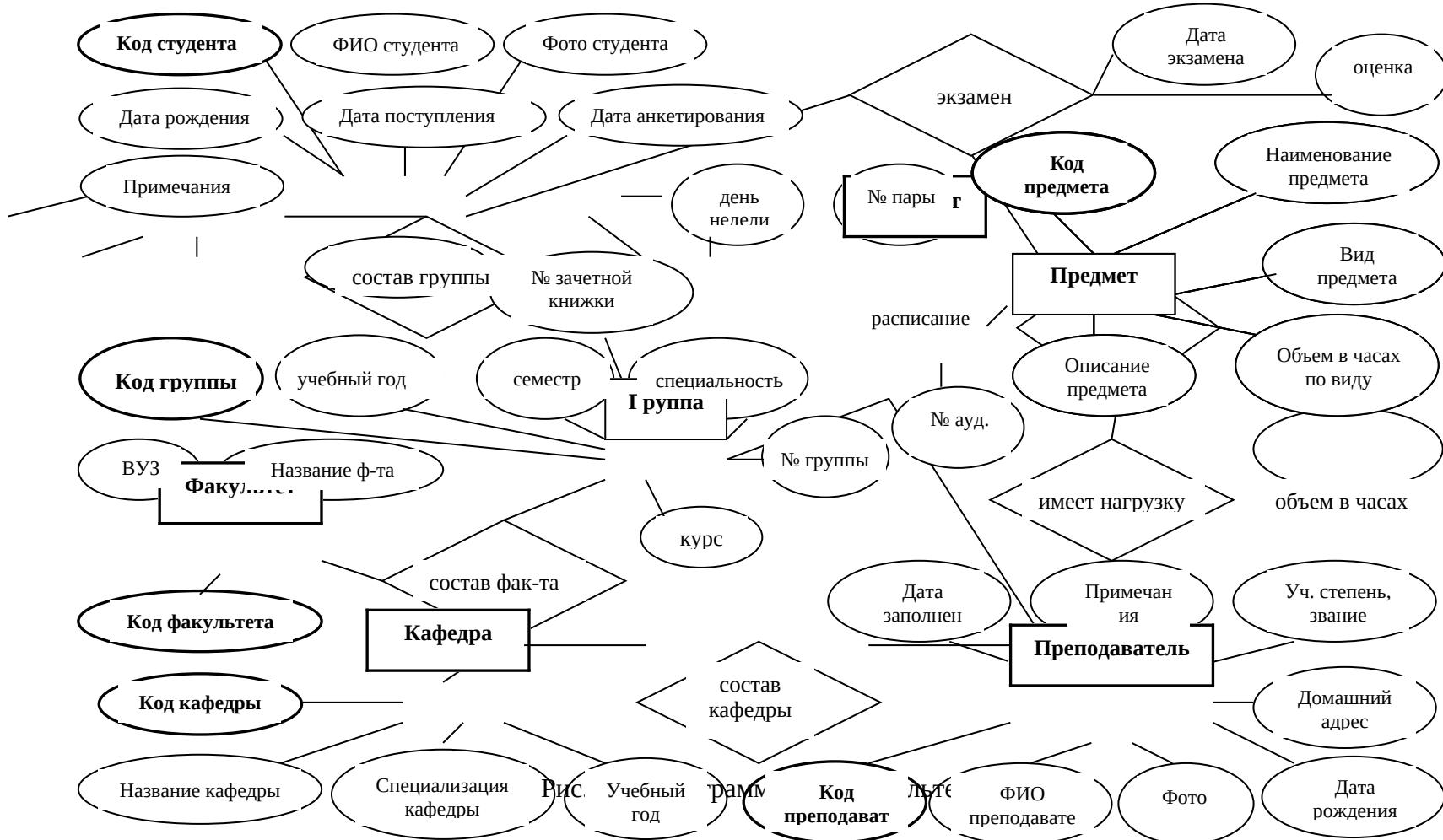


для представления связей между сущностями





При построении диаграммы «сущность-связь» (Рис. 8) придерживаемся правил, приведенных на стр. . У каждой сущности выделены идентифицирующие (ключевые) атрибуты.



3.4. Задания для аудиторной работы

В каждом варианте задания описаны требования, предъявляемые к проектируемой базе данных. Постройте диаграмму "сущность-связь".

1. В базе данных должны записываться даты начала и завершения каждого восхождения, имена и адреса участвовавших в нем альпинистов, название и высота горы, страна и район, где эта гора расположена. Дайте выразительные имена таблицам и полям, в которые могла бы заноситься указанная информация.

2. Базу данных использует для работы коллектив врачей. В таблицы должны быть занесены имя, пол, дата рождения и домашний адрес каждого их пациента. Всякий раз, когда врач осматривает больного, явившегося к нему на прием, или сам приходит к нему на дом, он записывает дату и место, где проводится осмотр, симптомы, диагноз и предписания больному, прописывает имя пациента, а также свое имя. Если врач прописывает больному какое-либо лекарство, в таблицу заносится название лекарства, способ его приема, словесное описание предполагаемого действия и возможных побочных эффектов.

3. В базе хранятся имена, адреса, домашние и служебные телефоны всех членов Думы. В Думе работает порядка сорока комиссий, все участники которых являются членами Думы. Каждая комиссия имеет свой профиль, например, вопросы образования, проблемы, связанные с жильем и так далее. Данные по каждой из комиссий включают: ее нынешний состав и председатель, прежние председатели и члены этой комиссии, участвовавшие в ее работе за прошедшие 10 лет, даты включения и выхода из состава комиссии, избрания ее председателей. Члены Думы могут заседать в нескольких комиссиях. В базу заносятся время и место проведения каждого заседания комиссии с указанием депутатов и служащих Думы, которые участвуют в его организации.

4. Фирме принадлежит небольшая флотилия рыболовных катеров. Каждый катер имеет «паспорт», куда занесены его название, тип, водоизмещение и дата постройки. Фирма регистрирует каждый выход на лов, записывая название катера, имена и адреса членов команды с указанием их должностей (капитан, боцман и т.д.), даты выхода и возвращения, а также вес пойманной рыбы отдельно по сортам (например, трески). За время одного рейса катер может посетить несколько банок. Фиксируется дата прихода на каждую банку и дата отплытия, качество выловленной рыбы (отличное, хорошее, плохое). На борту улов не взвешивается.

5. Разработать информационную систему обслуживания библиотеки, которая содержит следующую информацию: названия книг, ФИО авторов, наименования издательств, год издания, количество страниц, количество иллюстраций, стоимость, название филиала библиотеки или книгохранилища, в которых находится книга, количество

имеющихся в библиотеке экземпляров конкретной книги, количество студентов, которым выдавалась конкретная книга, названия факультетов, в учебном процессе которых используется указанная книга.