

Тема 5. ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ САПР

Содержание

5.1. Характеристика информации, используемой в САПР	1
5.2. Основные понятия	6
5.3. Эволюция методов организации и обработки данных в САПР	9
5.4. Банки и базы данных САПР	12
5.5. Модели данных	15
5.6. Реляционный подход	17
5.6.1. Операции над отношениями	20
5.6.2. Нормализация отношений	23
5.7. Иерархический и сетевой подходы	27
5.8. Организация базы данных на физическом уровне. Особенности баз данных САПР	29
5.9. Жизненный цикл информационной системы САПР и проектирование баз данных	31
5.10. Экспертные системы и базы знаний САПР	Ошибка! Закладка не определена.
5.10.1. Структура экспертной системы	38
5.10.2. Формы представления знаний	40

5.1. Характеристика информации, используемой в САПР

Проектирование реализуется комплексом задач, связанных с переработкой многочисленных массивов информации различного вида. Поэтому информационное обеспечение является одной из важнейших составных частей САПР, а затраты на его разработку составляют более половины стоимости системы в целом.

Время, затрачиваемое современной ЭВМ на вычисление, составляет 10%, а остальные 90% времени отводится на переработку информации. К переработке информации относятся запоминание, поиск необходимой информации в больших информационных массивах, передача информации от одного к другому, моделирование процессов и др. Следует отметить, что процесс вычисления также может рассматриваться как частный случай переработки информации.

Термин "информация" происходит от латинского *informatio*, что означает изложение, разъяснение. В обыденной жизни под этим словом понимают сведения, передаваемые людьми устным, письменным или другим образом. В научных и официальных источниках этот термин трактуется по-разному.

Информация¹, (*information* – англ.) – совокупность фактов, явлений, событий, представляющих интерес, подлежащих регистрации и обработке.

Это понятие объединяет двух партнеров: источник и приемник (потребитель) информации. В роли каждого из них может выступать объект науки и техники, общества и природы, животные и люди. Именно при их взаимодействии рождается информация.

В теории информации под этим термином понимается такое сообщение, которое содержит факты, неизвестные ранее потребителю и дополняющие его представление об изучаемом или анализируемом объекте (процессе, явлении). Другими словами, информация –

¹ Информация - сведения о лицах, предметах, фактах, событиях, явлениях и процессах [Закон Республики Беларусь Об информатизации //Ведомости ВС РБ.- 1995.- №33.-Ст. 428].

сведения, которые должны снять в той или иной степени существующую у потребителя до их получения неопределенность, расширить его понимание объекта полезными (для потребителя) сведениями. По Шеннону², *информация* – это снятая неопределенность.

В процессе обработки информация может менять структуру и форму. Признаками структуры являются элементы информации и их взаимосвязь. Различают *содержательную* и *формальную* структуры. *Содержательная структура* естественно ориентирована на содержание информации (эмпирические, научные знания, гипотезы, теории законы), а *формальная* – на форму представления информации. Формы представления информации также различны. Основные из них – *символьная* (основанная на использовании символов – букв, цифр, знаков), *текстовая* (использует тексты – символы, расположенные в определенном порядке), *графическая* (различные виды изображений), *звуковая*.

В зависимости от области знаний различают научную, техническую, производственную, правовую, патентную и другую информацию. Каждый из видов информации имеет свои особые смысловые нагрузки и ценность, требования к точности и достоверности, преимущественные технологии обработки, формы представления и носители (бумажные, магнитные и др.).

Информация – это единственный неубывающий ресурс³ жизнеобеспечения, более того, ее объем с течением времени возрастает. Особенно ярко это стало проявляться с середины XX в. В 70-е гг. объем информации удваивался каждые 5–7 лет. В 80-е гг. удвоение происходило уже за 20 месяцев, а в 90-е – ежегодно. Такой хлынувший лавинообразный поток не дает человеку воспринять информацию в полной мере.

В повседневной практике такие понятия, как информация, данные, знания, часто рассматриваются как синонимы. Однако это не верно. *Данными*⁴ называется информация, представленная в формализованном, удобном для обработки виде, т.е. в виде последовательности символов, букв, цифр, графиков, таблиц, чертежей, текстов и т.п. Взаимосвязанные данные часто называют *системой данных*, и хранимые данные называют *информационным фондом*.

Знание – это проверенный практикой результат познания действительности, ее верное отражение в сознании человека. Знания рассматривают как констатацию фактов и их описание. Научное знание заключается в понимании действительности в ее прошлом, настоящем и будущем, в достоверном обобщении фактов, в том, что за случайным оно находит необходимое, закономерное, за единичным – общее, а на основе этого осуществляет предвидение. Научное знание может быть как эмпирическим, так и теоретическим.

² Клод Шеннон - американский инженер и математик, родоначальник теории информации.

³ Ресурс - запасы, источники чего-нибудь. Информационный ресурс - организованная совокупность документированной информации, включающая базы данных и знаний, другие массивы информации в информационных системах (библиотеках, архивах, фондах и пр.).

⁴ Данные - документированная информация, циркулирующая в процессе ее обработки на электронно-вычислительных машинах.

В области систем искусственного интеллекта знания связываются с логическим выводом: *знания – это информация, на основании которой реализуется процесс логического вывода*. Другими словами, на основании этой информации можно делать различные заключения по имеющимся в системе данным с помощью логического вывода.

В системах обработки информации под **знаниями** понимают сложноорганизованные данные, содержащие одновременно как *фактографическую* (регистрация некоторого факта), так и *семантическую* (смысловое описание зарегистрированного факта) информацию, которая может потребоваться пользователю при работе с данными.

Информационное обеспечение (ИО) САПР — это совокупность сведений (данных), представленных в определенном виде и используемых при выполнении автоматизированного проектирования.

Информационное обеспечение САПР состоит из информационного фонда и средств управления этим фондом. *Информационный фонд* включает в себя информацию, необходимую для выполнения автоматизированного проектирования, и представляется в виде печатных документов, чертежей, файлов на машинных носителях, микрофиш и т.п. В информационный фонд входят данные о комплектующих деталях, узлах, материалах, технологической оснастке и оборудовании, типовых проектных решениях, текущем состоянии выполняемых проектов, сведения из ГОСТов, описания типовых проектных процедур и др.

Система управления информационным фондом организует хранение и доступ к информации. Значительная часть информационного фонда предназначена для многоразового использования различными лицами из коллектива проектировщиков и различными прикладными программами в маршрутах проектирования.

Информацию, используемую в САПР, исходя из различия информации по содержанию, форме представления, моменту возникновения, стабильности во времени, единицам измерения, отношению к процессу ее переработки и т.д. условно можно разделить на *исходную, промежуточную и выходную*.

Исходная информация, существующая до начала машинного проектирования, делится на *переменную и условно постоянную*.

Переменная информация – это разнообразные сведения о деталях основного производства, вводимые в систему в роли запроса с целью получения на них ответа в виде результатов конструкторско–технологического проектирования; они выступают в качестве исходных параметров при принятии тех или иных конструкторско–технологических решений. Переменная информация вводится при выполнении программы в оперативное запоминающее устройство каждый раз при проектировании нового объекта.

Например, переменной информацией при проектировании детали могут быть нагрузки на нее, внешние ограничения; для технологического проектирования – информация,

содержащаяся в конструкторских чертежах деталей основного производства, дополненная сведениями о размерах партий деталей, одновременно запускаемых в производство.

Для переменной информации является обязательным ее кодирование, т.е. описание на входном языке проектирования с занесением результатов в таблицы кодированных сведений. Продолжительность нахождения этой информации в системе кратковременное, ограничивающееся завершением полного цикла проектирования для закодированной партии деталей, после чего вводится очередная совокупность этой информации для следующей партии деталей.

Условно–постоянная информация – это информация длительное время хранящаяся на внешних запоминающих устройствах и многократно используемая при решении задач определенного класса. Например, к условно–постоянной информации при технологическом проектировании относятся сведения о металлорежущем оборудовании, станочных приспособлениях, режущих, вспомогательных и измерительных инструментах, используемых в системе. Сюда же относятся также справочные нормативы для расчета припусков, режимов резания, норм времени и другого рода сведения, необходимые для проектирования.

Условно–постоянная информация полностью или частично заменяется только в случаях перехода к новому классу задач или к другим производственным условиям. При этом не исключается возможность корректировки массивов условно–постоянной информации, отражающей некоторые изменения производственной обстановки, например, связанные с приобретением новых станков, приспособлений и других средств производства или с аннулированием устаревших и вышедших из строя и т. п.

Особую разновидность условно–постоянной информации представляет так называемая *критериальная информация*, которая является постоянной для конкретного класса задач, независимо от изменяющихся производственных условий, и отражает совокупность логически связанных между собой критериев, принятых при выборе проектных решений.

Совершенно очевидно, что как переменная, так и условно–постоянная информация поступают в систему извне по каналам ввода, в связи с чем оба эти вида информации относятся к входной информации. При этом завершение одноразового ввода полного объема условно–постоянной информации должно означать состояние рабочей готовности системы к приему непрерывного обновляемого потока переменной информации.

Промежуточной информацией называется информация, производимая в системе автоматизированного проектирования на различных этапах проектирования и используемая в качестве исходной для решения задач последующих этапов. Наличие массивов промежуточной информации является неизбежным явлением при конструкторском и технологическом проектировании, реализующем в определенной последовательности целую цепочку взаимосвязанных задач. С помощью массивов промежуточной информации в различных САПР

осуществляется стыковка задач, этапов и подсистем. Промежуточная информация применительно к САПР ТП содержит сведения о маршруте обработки заготовки, технологических операциях и переходах, режимах резания, графических изображениях операционных эскизов и инструментальных наладок и др.

Выходной информацией в САПР называется информация, формируемая по результатам проектирования. К ней относится информация, отражающая содержание выходных документов системы, а также массивы, формируемые для стыковки САПР с другими автоматизированными системами (АСУП, АСУ ТП) или службами и производственными подразделениями предприятия. В отношении характера ее воспроизведения информация выходных документов системы подразделяется на печатную и графическую. Информация же стыковочных массивов с помощью различных систем декодирования может быть представлена в виде звуковых или световых сигналов, команд на перемещение рабочих органов станка и т. п.

Неотъемлемым дополнением к конструкторско–технологической информации является прикладная часть информационного обеспечения – комплекс средств описания и классификации информации, перерабатываемой в системе. Это своего рода нормативно–руководящая система в рамках системы проектирования, включающая ряд ГОСТов, ОСТов, методик, положений и других руководств, регламентирующих правила кодирования, описания и представления всех видов технологической информации, используемой в системе. Можно выделить как минимум три категории документов этой системы:

- * документы конструкторского характера, отражающие закономерности основных положений конструкторского проектирования и правила конструкторской подготовки производства,

- * документы технологического характера, отражающие закономерности основных положений технологии машиностроения и правила технологической подготовки производства,

- * документы кибернетического характера, появление которых обусловлено спецификой самого метода автоматического проектирования.

Например, в нормативных документах технологической категории должны быть рассмотрены все виды технологических процессов, предусмотрено наличие широкого комплекса рекомендаций по выбору отдельных технологических решений, а также должны быть представлены различного рода многоуровневые классификаторы изготавливаемых изделий и их составных частей, элементов технологических процессов и средств, обеспечивающих их выполнение. Наличие всех документов этой категории является необходимым для успешного начала и развития работ в области автоматизированного проектирования. При создании информационного обеспечения САПР ТП из действующей в машиностроении нормативной документации используются отраслевые системы классификаций и условных обозначений оборудования, станочных приспособлений, режущего, вспомогательного и измерительного

инструмента, должна производиться унификация и разрабатываться классификация элементов деталей и технологических процессов, установления связей между ними, а также рекомендации по принятию технологических решений.

К нормативной документации специфического кибернетического характера в САПР ТП относятся методика описания и кодирования вводимой в систему переменной информации, а также правила и положения, устанавливающие логическую зависимость данных в массивах условно–постоянной информации. Документация этой категории является следствием принятого в системе метода автоматизированного проектирования.

В заключение заметим, что если о технологической информации, используемой в системах автоматизации технологической подготовки производства, можно говорить как о некоторой объективной реальности, существующей вне зависимости от методов организации проектирования, субъективного подхода разработчиков систем и т.д., то средства описания и представления одной и той же информации в разных системах могут отличаться. Поэтому правильный выбор имеющихся методов классификации, кодирования и описания информации является важным и ответственным моментом, особенно при создании сложных комплексных автоматизированных систем конструкторско–технологического проектирования.

5.2. Основные понятия

Итак, основу информационного обеспечения подсистем САПР составляет совокупность данных, которые необходимы для выполнения процесса проектирования. Основное назначение информационного обеспечения – предоставлять пользователям САПР достоверную информацию в определенном виде. Совокупность компонентов ИО образует *информационную базу*, называемую базой данных (БД) САПР.

Объектом считают любой предмет, событие, понятие и т.п., о которых приводятся данные. Все объекты характеризуются *атрибутами*. Например, объект ЭВМ можно характеризовать такими атрибутами: тактовой частотой процессора, объемом оперативной памяти, емкостью винчестера, числом процессоров, габаритами, типом видеоадаптера и т.п. Сведения, содержащиеся в каждом атрибуте, называют *значениями данных*. Ниже приведены значения данных для ПЭВМ IBM PC:

тип процессора	Pentium
тактовая частота процессора	133
объем оперативной памяти	32 Мбайт
емкость винчестера	1,3 Гбайт
заводской номер	785
и т.д.	

Среди атрибутов имеются такие, по значениям которых возможна идентификация объекта. Например, если известен заводской номер ЭВМ 785, то можно определить, что это IBM PC, тактовую частоту ее процессора и т. д.

Атрибуты, по значениям которых определяют значения других атрибутов, называют

идентификаторами объекта или ключевыми элементами данных. Отметим, что один и тот же объект могут идентифицировать несколько элементов данных. Их тогда считают кандидатами в идентификаторы. Проблему выбора идентификатора из нескольких кандидатов решает пользователь САПР.

Объединение значений связанных атрибутов называют *записью данных*. Например, ЭВМ 785 IBM PC – запись данных. Упорядоченную совокупность записей данных называют *файлом данных* или *набором данных*.

При обработке данных с расположением одних и тех же элементов данных в нескольких файлах возникают трудности, связанные с избыточностью данных, что требует нескольких процедур ввода, обновления и формирования; опасностью нарушения непротиворечивости данных, что связано с хранением одной и той же информации в нескольких местах; негибкостью к изменениям; сложностью управления процессом решения задачи.

Существует много адекватных и в то же время неформальных определений базы данных. Приведем те из них, которые получили наибольшее распространение.

База данных – совокупность данных и метаданных, т.е. описаний свойств данных, интерпретируемая в среде специальной программной системы – системы управления базами данных (СУБД) – как многоуровневая организация, обеспечивающая независимость представления данных на различных уровнях. *База данных* – это совокупность данных, отображающая состояние объектов и их отношений в рассматриваемой предметной области.

Приведем основное различие между БД и файлом данных. Файл данных имеет несколько назначений, но соответствует одному представлению хранимых данных. База данных имеет также несколько назначений, но соответствует различным представлениям о хранимых данных.

Сформулируем основные требования к БД:

Целостность данных – их непротиворечивость и достоверность.

Организация БД должна обеспечивать *согласование времени выборки данных* прикладными программами с частотами их использования прикладными программами САПР.

Организация сквозного автоматизированного проектирования характеризуется необходимостью автоматизировать не только выполнение различных проектных процедур, но и обмена информацией между различными частями САПР. Эти обмены составляют информационный интерфейс между подсистемами САПР, прикладными программами на маршрутах проектирования, пользователями и техническими средствами. Информационный интерфейс возлагается в САПР на банк данных.

Универсальность, т. е. наличие в БД всех необходимых данных и возможности доступа к ним в процессе решения проектной задачи.

Открытость БД для внесения в нее новой информации.

Наличие языков высокого уровня взаимодействия пользователей с БД.

Секретность, т.е. невозможность несанкционированного доступа к информации и ее изменений.

Минимизация *избыточности* данных.

Обеспечение одновременного использования БД многими пользователями.

Одним из принципов построения САПР является информационная согласованность частей ее программного обеспечения, т.е. пригодность результатов выполнения одной проектной процедуры для использования другой проектной процедурой без их трудоемкого ручного преобразования пользователем. Отсюда вытекают следующее условие информационной согласованности:

* использование программами одной и той же подсистемы САПР единой БД.

Комплексная автоматизация процесса проектирования объекта предполагает информационную согласованность не только отдельных программ подсистем САПР, но и самих подсистем между собой. Способом достижения этой согласованности является *единство информационного обеспечения*.

Основные способы информационного согласования подсистем САПР достигаются либо созданием единой БД, либо сопряжением нескольких БД с помощью специальных программ, которые перекодируют информацию, приводя ее к требуемому виду.

Части программного обеспечения и методы, осуществляющие управление базой данных, составляют *систему управления базами данных* (СУБД). На рис.5.1 показан пример использования СУБД и БД для проектирования изделий машиностроения.

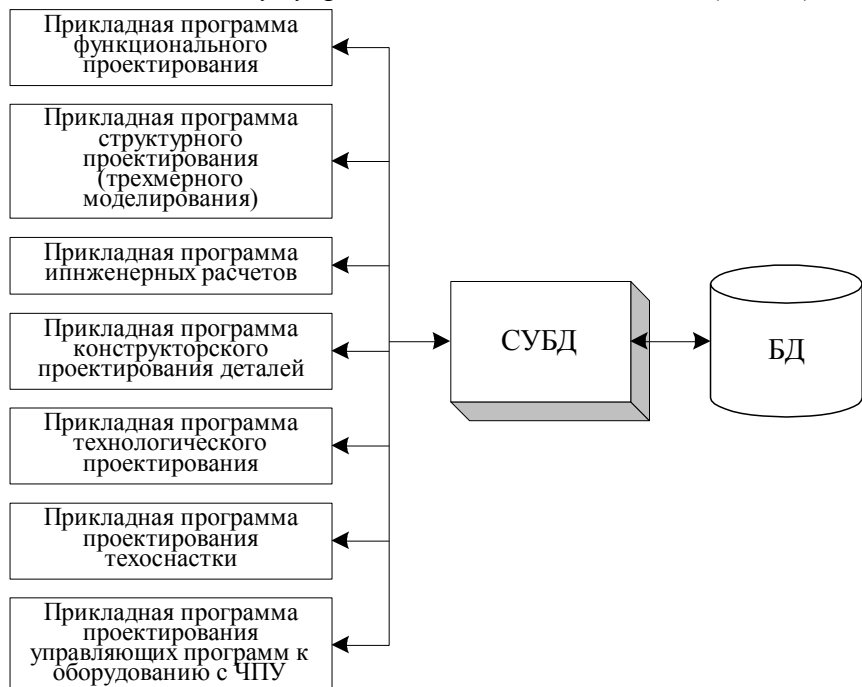


Рис. 5.1. Пример использования СУБД и БД для проектирования изделий

машиностроения. Опишем последовательность работы СУБД: 1) программа запрашивает возможность чтения данных у СУБД, она передает необходимую информацию о пользователе, типе записи и т.п.; 2) программа осуществляет поиск описания данных, на которые выдан запрос; 3) определяет, какого типа логические и физические записи

необходимы; 4) выдает операционной системе запрос на чтение требуемой записи; 5) операционная система взаимодействует с физической памятью; 6) записывает запрошенные

данные в системные буферы; 7) выделяет требуемую логическую запись, выполняя необходимые преобразования; 8) передает данные из системных буферов в программу пользователя, а затем программе пользователя информацию о результатах выполнения запроса; 9) прикладная программа обрабатывает полученные данные. Система управления БД позволяет получить доступ к интегрированным данным и допускает множество различных представлений о хранимых данных.

Приведем основные определения СУБД, сформулируем требования, которым они должны удовлетворять, и опишем функции, выполняемые СУБД.

Программное обеспечение, которое позволяет прикладным программам работать с БД без знания конкретного способа размещения данных в памяти ЭВМ, называют **системой управления базами данных**. Система управления БД выступает как совокупность программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями.

Система управления базами данных *должна обеспечивать*:

- * простоту физической реализации БД;
- * возможность централизованного и децентрализованного управления БД;
- * минимизацию избыточности хранимых данных;
- * предоставление пользователю по запросам непротиворечивой информации;
- * простоту разработки, ведение и совершенствование прикладных программ;
- * выполнение различных функций.

5.3. Эволюция методов организации и обработки данных в САПР

Любое предприятие, учреждение, организацию можно рассматривать как *информационную систему*, состоящую из элементов, связей между ними, по которым циркулирует некоторая информация, определенным образом представленная, перерабатываемая, передаваемая. Говорят, что ИС функционирует на базе некоторой *информационной технологии*. В понятие информационной технологии входят все устройства, носители, методы хранения, переработки, принципы обмена информацией. Например, информационная технология предприятия 30–40-х гг. строилась на базе телефона, почты, устных сообщений, отчетов, различных форм и бланков, бумаги и т.п.

Задачей проектировщиков автоматизированных систем является включение в существующие информационные системы со своей информационной технологией элементов автоматизации на всех уровнях и создание на базе автоматизации *новой информационной технологии*. В эти уровни включается и уровень автоматизации конструкторско-технологической подготовки производства.

Первые автоматизированные системы проектирования разрабатывались на основе так называемого *позадачного метода*. При этом решались вопросы автоматизации задач

оперативного уровня. Выбирались самые очевидные, формализуемые (имеющие алгоритмы решения) задачи, автоматизация которых сразу давала максимальную эффективность. Это были задачи расчета припусков и режимов резания, выбора заготовки и оборудования и т.п.

Позадачный метод автоматизации крайне прост, очевиден и заключается в следующем. Для каждой задачи создавался свой блок данных D_i и своя прикладная программа P_i , которая решала одну эту задачу с максимальной эффективностью:

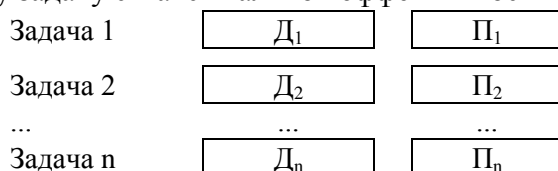


Рис.5.2.

Но при создании сложных информационных систем САПР разработчики столкнулись со следующими трудностями и проблемами, которые заложены в позадачном методе:

1. Возникла *проблема контроля избыточности данных*, поскольку данные в задачах могли дублироваться, т.е. пересечение $D_i \cap D_j \neq 0$. К примеру, и задача расчета режимов резания, и выбора оборудования содержат одинаковые данные о характеристиках станков. Сложность состоит в том, что любое изменение исходных данных в одной задаче влечет за собой необходимость одновременной корректировки данных во всех задачах.

2. Возникла *взаимосвязь между данными и прикладными программами*, которые писались на стандартных языках программирования типа Паскаль, Фортран, Бейсик и т.п. Прикладные программы содержали как описание данных, так и алгоритмы манипулирования данными (операции доступа, выбора, вставки, замены, удаления, реорганизации данных). В результате любое изменение в организации данных приводило к необходимости изменения программы. Кроме того, практически идентичные алгоритмы манипулирования данными должны были содержаться во всех прикладных программах.

3. *Невозможно выйти на качественно новый уровень информационных технологий*, поскольку

* не реализуется "принцип новых задач", автоматизация по-старому, исходя из существующих в учреждении задач;

* выборочная автоматизация информационных процессов нарушает естественные для предприятия взаимосвязи, информационная система работает в разнорой.

Эти и другие трудности привели разработчиков автоматизированных систем к пониманию того, что при переходе от автоматизации отдельных задач к созданию сложных автоматизированных информационных систем требуется не только взаимоувязка задач, но и качественно новый подход к организации данных. Были сформулированы *стандартные требования к организации данных в автоматизированной системе*. Два основных требования заключаются в следующем:

* *интеграция данных*, когда все данные накапливаются и хранятся централизованно, создавая динамически (в реальном масштабе времени) обновляемую модель предметной области. Предметная область – часть реального мира, подлежащая автоматизации;

* *максимально возможная независимость прикладных программ от данных*, т.е. отделение логической модели предметной области от физического представления в памяти ЭВМ или, как говорят, обеспечение логической и физической независимости данных.

Выполнение этих требований привело к созданию единого для всех задач блока данных (базы данных – БД) и разработке одной управляющей программы для манипулирования данными на физическом уровне (системы управления базой данных – СУБД) (рис.5.1).

Из прикладных программ позадачного метода в СУБД были выделены средства манипулирования данными на физическом уровне. Именно СУБД обеспечивает независимость данных, а прикладные программы поддерживают логику каждой конкретной задачи. Изменения физической организации воспринимаются СУБД и не влияют на прикладную программу. Изменение логики прикладной программы не требует реорганизации и изменения механизма доступа к физическим данным.

Таким образом, введение СУБД отделяет логическую структуру данных (т.е. представление об организации данных с точки зрения пользователя) от физической структуры данных в памяти ЭВМ.

Впервые термин "база данных" появился в 1962 г. База данных самый важный элемент автоматизированной системы. Но не всякий блок данных является базой данных. База данных – это совокупность данных, обладающих следующими качествами:

- * интегрированностью, направленной на решение общих задач;
- * модульностью (т.е. структурированностью, отражающей некоторую часть реального мира);
- * взаимосвязанностью;
- * коллективным использованием несколькими прикладными программами;
- * независимостью описания данных от прикладных программ, поскольку данные и их описания хранятся совместно в БД.

Различают термины "фактографическая БД" и "документальная БД". *Фактографическая БД* хранит множество сведений об объектах предметной области, их свойствах, связях между ними. *Документальная БД* накапливает и обрабатывает произвольные текстовые документы. Нас в дальнейшем будут интересовать фактографические, а также смешанные, документально–фактографические БД.

Аналогично не всякая управляющая программа работы с БД является СУБД. *СУБД* – это пакет программ, позволяющий:

- * обеспечить пользователей (прикладные программы) языковыми средствами

описания и манипулирования данными;

- * обеспечить поддержку логических моделей данных. Модель данных определяет логическое представление физических данных;

- * обеспечить операции создания и манипулирования логическими данными (выбор, вставка, обновление, удаление и т.п.) и одновременное отображение (выполнение) этих операций над физическими данными;

- * обеспечить защиту и целостность (согласованность) данных, поскольку при коллективном режиме работы многих пользователей возможно использование общих физических данных. Это означает, что необходимо обеспечивать защиту от некорректных обновлений пользователями, защиту от несанкционированного доступа, защиту данных от разрушений при сбоях оборудования.

Возможны два варианта построения СУБД. В первом случае средства описания и манипулирования данными СУБД включаются в универсальный язык программирования типа Си, Паскаль, на котором пишутся прикладные программы. Во втором случае СУБД представляет собой интерпретатор (или компилятор) специального "языка запросов" к данным, способный в диалоговом или программном режимах выполнять команды управления БД этого языка. Прикладная программа может быть полностью написана на этом языке. Тем не менее такие СУБД обычно допускают расширения на языках Си, Паскаль, Ассемблер, Бейсик. Изучение языка программирования СУБД второго типа не менее сложно, чем изучение стандартного языка программирования.

Система управления БД реализует два интерфейса:

- 1) между логическими структурами данных в программах и в БД;
- 2) между логической и физической структурами БД.

В заключение перечислим *преимущества* использования *концепции БД* в автоматизированных системах:

Централизованное управление информационными ресурсами, синхронное поддержание данных для всех приложений.

Отсутствие проблемы контроля избыточности данных вследствие их интеграции.

Однократный ввод и многократное использование данных благодаря устранению дублирования.

Унификация средств организации данных и независимость прикладных программ от организации данных.

5.4. Банки и базы данных САПР

Автоматизированный банк данных представляют собой человеко–машинную систему, включающую специальные структуры организации информации, алгоритмы, специализированные языки и административный персонал.

Базы данных, СУБД, аппаратные средства, обслуживающие службы и некоторые другие компоненты вместе составляют *банк данных* (БНД). В совокупности они обеспечивают создание и эксплуатацию эффективных систем накопления информации, поступающей от нескольких источников, ее обновление, корректировку, многоцелевое использование в различных подсистемах проектирования, а также прямую связь с пользователем для получения ответов на произвольные, в том числе заранее незапланированные вопросы.

Создание открытых САПР подразумевает реализацию в них возможностей взаимно независимого внесения изменений в информационное и программное обеспечение. Степень открытости будет высокой, если замена какой-либо одной программы не потребует внесения изменений в другие программы или в структуры данных и если изменения в информационном фонде или технических средствах хранения фонда не потребуют изменений в программном обеспечении. Такая развязка информационного и программного обеспечения реализуется в банках данных (БНД) и поддерживается введением *трех уровней описания данных*.

Концептуальный уровень, называемый логической схемой БД, служит для описания структуры информационного фонда (всей совокупности баз данных в рассматриваемом БНД) без привязки к физическим способам его хранения в запоминающих устройствах ЭВМ. *Внешний уровень* связан с описанием отдельных баз данных без учета способов хранения. Такие описания называются *логическими подсхемами*. *Физический уровень* касается размещения данных в памяти ЭВМ. Концептуальный и внешний уровни иногда объединяются в *логический уровень*.

Разработчики программного обеспечения САПР могут использовать лишь представления логического уровня, что существенно упрощает их работу по информационному согласованию частей разрабатываемого программного обеспечения. Эти представления выражаются средствами специальных языков БНД – *языков описания данных* (ЯОД) и *языков манипулирования данными* (ЯМД). С помощью ЯОД задаются структуры БД, т.е. типы и форматы данных, способы их объединения в те или иные структурные единицы и агрегаты, способы связи этих единиц между собой. По своему характеру предложения ЯОД близки к объявлениям типов данных в таких высокоразвитых алгоритмических языках, как ПАСКАЛЬ или АДА. С помощью ЯМД реализуется интерфейс пользователей и прикладных программ с БД. В большинстве случаев ЯМД представляют собой расширения известных языков программирования.

Банки данных и их составные части *классифицируют* по ряду признаков.

По **степени универсальности** различают СУБД универсальные и специализированные, а БД – проектно-зависимые и проектно-независимые. *Универсальные СУБД* можно использовать в различных приложениях, специализация соответствующего БНД при этом будет определяться конкретным наполнением БД. Возможности построения универсальных СУБД

вытекают из идентичности структур данных, имеющих различную проблемную принадлежность. *Специализированные СУБД* позволяют за счет ориентации на определенную предметную область с характерными структурами данных и процедурами их обработки добиться большей эффективности использования вычислительных ресурсов.

Проектно-зависимые БД содержат информацию о текущих проектах, эта информация претерпевает частые изменения. *Проектно-независимые БД*, называемые *архивами*, хранят данные, применяемые во многих проектах, выборка данных из архивов производится гораздо чаще, чем запись новых данных.

По **масштабам использования** различают БД интегрированные (общие), локальные и отдельных пакетов прикладных программ (ППП). *Интегрированная БД* относится ко всей САПР, в ней содержится информация, являющаяся предметом обработки в более чем одной подсистеме. Через интегрированную БД реализуются информационные связи между подсистемами САПР. *Локальные БД* вместе с соответствующими СУБД обслуживают одну из подсистем САПР и реализуют информационные связи между пакетами программ и программно-методическими комплексами внутри подсистемы. *Базы данных отдельных ППП* организуются для унификации информационных связей между отдельными программами пакета. Такие БД обычно появляются в случаях включения в САПР независимо разработанного пакета, имеющего свои средства информационного интерфейса.

По **месту хранения** БД делятся на централизованные и распределенные. *Централизованные БД* хранятся в запоминающих устройствах центрального вычислительного комплекса или в специально выделенном узле вычислительной сети. *Распределенные БД* состоят из нескольких частей, распределенных по узлам вычислительной системы или сети (например, по различным АРМ), в которых данные размещены по месту возникновения или наиболее эффективного использования. Она предполагает, что на каждой ЭВМ данные управляются локальными СУБД.

По **степени связности** (структурированности) данных различают БД и СУБД документальные и фактографические. Структура данных задается указанием множества составных частей информации и способов их взаимосвязи. При описании структур данных чаще всего оперируют записями как основными частями информации. Записи состоят из полей, поля – из элементов – наименьших неделимых без потери смысла единиц информации. Записи могут объединяться в более крупные структурные единицы, называемые массивами, списками, файлами, отношениями, базами данных. *Документальные* (дескрипторные) *БД*, называемые также *информационно-поисковыми системами* (ИПС), характеризуются тем, что информация представляется в виде слабоструктурированных записей. Слабоструктурированные записи состоят из элементов символьного типа переменной длины, чаще всего это предложения из слов естественного языка. *Фактографические БД* характеризуются тем, что информация

хранится в виде сильно структурированных записей, для которых характерны фиксированные количество и форматы полей. Примеры подобных записей – строки таблиц с числовыми значениями элементов.

По типу принятой модели данных различают БД реляционные, сетевые, иерархические. *Моделью данных* называется представление о предметной области в виде структуры данных – обозначений данных и связей между ними.

5.5. Модели данных

Пользователя БД интересует ее информационное и смысловое содержание. Подробности организации физического хранения данных его не интересуют. Поэтому в представлении данных в БД можно выделить два уровня абстракции:

- * информационную модель;
- * физическую модель данных.

Информационная модель должна отображать предметную область в терминах, понятных и привычных для пользователя. Обычно это информация об интересующих его фактах, явлениях, событиях, предметах, об их свойствах и связях между ними.

Проектировщики информационной модели термином *сущность* называют объект любой природы, о котором надо хранить информацию в БД. Например, множество студентов можно назвать сущностью "студент". Свойства, характеризующие сущность, называют *атрибутами*. Примерами атрибутов сущности "студент" являются Фамилия, Возраст, Курс, Дата поступления и т.п. Между различными сущностями предметной области и их атрибутами могут существовать межсущностные и межатрибутные связи, информационно важные для пользователя БД.

Абстрактная информационная модель предметной области, т.е. выделенные в ней сущности, атрибуты, связи должны быть каким-то образом описаны для представления в ЭВМ. Это описание делается средствами *модели данных*, которую поддерживает СУБД, и называется *внутренней схемой* информационной модели.

Таким образом, получаем три уровня описания данных в БД, упомянутых выше (рис.5.3).

Как видно, СУБД поддерживает некоторую модель данных и отображает ее в соответствующие структуры физической базы данных. Средствами модели данных СУБД строится логическая внутренняя схема информационной модели предметной области. Прикладная программа, которая пишется в терминах модели данных СУБД, поддерживает логику внешней информационной модели предметной области для пользователя, основываясь на внутренней схеме предметной области.

Таким образом осуществляется информационное моделирование предметной области в памяти ЭВМ. Язык программирования СУБД содержит средства описания как внутренней схемы, так и создания прикладных программ.

Модель данных, которую поддерживает СУБД на логическом уровне, определяется тремя

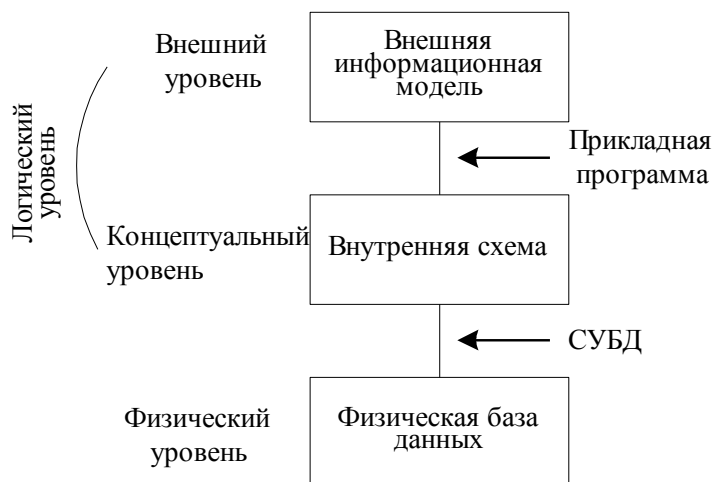


Рис.5.3. Уровни описания данных в БД

компонентами:

Допустимой структурой данных, разнообразием и количеством типов объектов, которые можно описать с помощью модели.

Множеством допустимых операций над данными.

Ограничениями для контроля целостности; это логические ограничения модели, накладываемые на данные, для сохранения

непротиворечивости данных и обеспечения адекватного (достоверного) отображения предметной области в БД.

Модели данных, поддерживаемые СУБД, делят на *сетевые, иерархические и реляционные*. Соответственно различают сетевые, иерархические и реляционные СУБД.

Сетевые и иерархические СУБД получили наибольшее распространение на больших и мини-ЭВМ. Основной вклад в разработку теории сетевых и, как частный случай, иерархических моделей данных, языков описания и манипулирования СУБД внесла КОДАСИЛ (CODASYL) – Ассоциация по языкам и системам обработки данных. Практически все СУБД персональных компьютеров поддерживают реляционную модель данных.

Реляционная модель данных представляет собой совокупность таблиц, называемых отношениями. Строки таблиц соответствуют записям, называемым кортежами, а столбцы – атрибутам, являющимся характеристиками объектов описываемой предметной области.

Сетевая модель данных представляет структуру данных в виде произвольного графа, вершины которого соответствуют записям, а ребра – связям между записями.

Иерархическая модель данных отличается от сетевой тем, что граф, представляющий структуру, является деревом.

Для каждой модели данных характерен свой набор операций, выполняемых над данными. Совокупность модели данных и соответствующего набора операций называется **подходом**. Иерархический подход обеспечивает наиболее быстрый доступ к данным. Реляционный подход характеризуется широкими возможностями манипулирования данными и удобством ЯМД для пользователей, хотя и приводит к значительным затратам времени на

поиск. Сетевой подход по этим характеристикам занимает промежуточное положение.

5.6. Реляционный подход

Концепция реляционной модели данных была предложена Е.Ф.Коддом в 1970 г. для обеспечения независимости представления и описания данных от прикладных программ.

В основе реляционной модели данных лежит понятие *отношения* (англ. relation). Отношение удобно представляется в виде двумерной таблицы при соблюдении определенных ограничивающих условий. Таблица понятна, обозрима и привычна для человека.

Кодд показал, что набор отношений (таблиц) может быть использован для хранения данных об объектах реального мира и моделирования связей между ними. Например, для хранения сущности "студент" используют отношение *СТУДЕНТ*, в котором свойства сущности располагаются в столбцах таблицы:

СТУДЕНТ

<i>Фамилия_И.О.</i>	<i>Дата_рождения</i>	<i>Курс</i>	<i>Специальность</i>
Иванов И.И.	12.03.1978	4	Технология машиностроения
Петров П.П.	30.10.79	3	История
Владимиров В.В.	1.01.1980	3	Технология машиностроения
Петровский П.С.	20.12.77	5	Бухучет
Сидоров С.С.	2.04.77	5	Экономика

Столбцы отношения называют *атрибутами* и присваивают им имена. Список имен атрибутов отношения называется *схемой отношения*. Схема нашего отношения *СТУДЕНТ* запишется так:

СТУДЕНТ(Фамилия_И.О., Дата_рождения, Курс, Специальность)

Реляционная база данных – это набор взаимосвязанных отношений. Каждое отношение (таблица) в ЭВМ представляется в виде отдельного файла либо части его. Между ними существуют следующие соответствия:

Файл	Таблица	Отношение	Сущность
запись	строка	кортеж	экземпляр сущности
поле	столбец	атрибут	атрибут

Таблицы всем хороши, но не учитывают необходимости согласования трех способов манипулирования данными: упорядочение, группировку по значению индексов, доступ по дереву параметров. В таблицах все три способа манипулирования жестко связаны между собой: упорядоченные по одному параметру данные не упорядочены по другому, значит, облегчается доступ к нужным данным по одной цепочке, а по другой поиск усложняется.

Оригинальность подхода Кодда состояла в том, что он предложил применять к отношениям (таблицам) стройную систему операций, позволяющую получать (выводить, вычислять подобно арифметическим операциям) одни отношения из других. Это дает возможность делить информацию на хранимую и вычисляемую части, экономить память, при необходимости вычисляя нехранимую часть информации из хранимой.

Основными операциями над отношениями в реляционной базе данных являются следующие восемь:

- * традиционные операции над множествами, такие, как объединение, пересечение, разность (вычитание), декартово произведение, деление;

- * специальные реляционные операции проекции, соединения и выбора (селекции, ограничения).

Эффективность конкретной СУБД определяется наличием и удобством использования средств выполнения этих операций.

Языки для выполнения операций над отношениями в реляционной СУБД можно разделить на два главных класса:

- * языки реляционной алгебры;
- * языки реляционного исчисления.

Языки реляционной алгебры основаны на реляционной алгебре. Записывая последовательности операций над отношениями в соответствующем порядке, можно получить желаемый результат. Поэтому языки реляционной алгебры являются процедурными.

Языки реляционного исчисления основаны на классическом исчислении предикатов. Они предоставляют пользователю набор правил для записи запросов к БД, содержащем только информацию о желаемом результате. На основании запроса СУБД автоматически, путем формирования новых отношений, выдаст желаемый результат. Языки реляционного исчисления являются непроцедурными. В настоящее время в реляционных СУБД широкое распространение получает язык запросов SQL (Structured Query Language), разработанный фирмой IBM.

Отношения реляционной БД в зависимости от содержания подразделяют на два класса: объектные отношения и связные отношения.

Объектное отношение хранит данные об объектах (экземплярах сущности). Приведенное выше отношение *СТУДЕНТ* является объектным отношением. В объектном отношении один из атрибутов однозначно идентифицирует отдельный объект. Такой ключевой атрибут называют *ключом отношения* или *первичным атрибутом*. В отношении *СТУДЕНТ* на роль ключа претендует атрибут "*Фамилия_И.О.*". Для удобства ключ записывают в первом столбце таблицы. Остальные атрибуты функционально зависят от этого ключа.

Ключ может включать несколько атрибутов (составной ключ) или быть частью значения атрибута (частичный ключ).

В объектном отношении не должно быть строк с одинаковыми ключами, т.е. не должно быть дублирования объектов. Это основное ограничение реляционной модели данных для обеспечения целостности данных.

Связное отношение хранит ключи двух или более объектных отношений, т.е. по ключам устанавливаются связи между объектами отношений. Рассмотрим связное отношение *ИЗУЧАЕТ* (*Студент, Предмет*), означающее, что студент изучает определенный предмет. При этом пусть

в БД имеются объектные отношения *СТУДЕНТ*(*Фамилия_И.О.*, *Дата_рождения*, *Курс*, *Специальность*), рассмотренное выше, и *ПРЕДМЕТ*(*Название*, *Семестр*) со следующими данными:

ПРЕДМЕТ

<i>Название</i>	<i>Семестр</i>
Высшая математика	1
Информатика	1
САПР	8
САПР ТП	9
Детали машин	6

Связное отношение *ИЗУЧАЕТ* может содержать такие данные:

ИЗУЧАЕТ

<i>Фамилия_И.О.</i>	<i>Название</i>
Иванов И.И.	Высшая математика
Петров П.П.	Информатика
Владимиров В.В.	САПР
Петровский П.С.	САПР ТП
Сидоров С.С.	Детали машин

Связное отношение кроме связываемых ключей может иметь и другие атрибуты, которые функционально зависят от этой связи. Примером может быть связное отношение *ИЗУЧАЕТ*(*Студент*, *Предмет*, *Оценка*):

ИЗУЧАЕТ

<i>Фамилия_И.О.</i>	<i>Название</i>	<i>Оценка</i>
Иванов И.И.	Высшая математика	4
Петров П.П.	Информатика	5
Владимиров В.В.	САПР	4
Петровский П.С.	САПР ТП	3
Сидоров С.С.	Детали машин	4

Ключи в связных отношениях называются *внешними ключами*, поскольку они являются первичными ключами других отношений.

Реляционная модель накладывает на внешние ключи ограничение для обеспечения целостности данных, называемое ссылочной *целостностью*. Это означает, что каждому внешнему ключу должна соответствовать строка какого-либо объектного отношения. Без такого ограничения может случиться, что внешний ключ ссылается на объект, о котором ничего не известно.

В реляционной базе данных на каждое отношение накладывается и другое ограничение – они должны быть *нормализованы*. Это означает, что каждый атрибут должен быть простым – содержать атомарные, неделимые значения. К примеру, приведенное ниже отношение *СТУДЕНТ* ненормализовано, поскольку содержит сложный атрибут "*Спорт*":

СТУДЕНТ

<i>Фамилия_И.О.</i>	<i>Курс</i>	<i>Специальность</i>	<i>Спорт</i>	
			<i>Вид</i>	<i>Разряд</i>
Иванов И.И.	4	Технология машиностроения	плавание	мс
Петров П.П.	3	История	футбол	1 р.
Владимиров В.В.	3	Технология машиностроения	хоккей	кмс
Петровский П.С.	5	Бухучет	шахматы	1 р.
Сидоров С.С.	5	Экономика	футбол	1 р.

Здесь ключом является атрибут "*Фамилия_И.О.*". Приведем это отношение к нормализованному виду, т.е. избавимся от сложного атрибута "*Спорт*":

СТУДЕНТ

Фамилия_И.О.	Вид_спорта	Курс	Специальность	Спорт_разряд
Иванов И.И.	плавание	4	Технология машиностроения	мс
Петров П.П.	футбол	3	История	1 р.
Владимиров В.В.	хоккей	3	Технология машиностроения	кмс
Петровский П.С.	шахматы	5	Бухучет	1 р.
Сидоров С.С.	футбол	5	Экономика	1 р.

В полученном отношении *СТУДЕНТ* ключ является составным, состоящим из атрибутов "Фамилия_И.О." и "Вид_спорта".

Отношение, у которого все атрибуты простые, называется приведенным к *первой нормальной форме* (1НФ).

Далее перечислим условия и ограничения, накладываемые на отношения реляционной моделью данных, которые позволяют таблицы считать отношениями:

Не может быть одинаковых первичных ключей, т.е. все строки (записи) таблицы должны быть уникальны.

Все строки таблицы должны иметь одну и ту же структуру, т.е. одно и то же количество атрибутов с соответственно совпадающими именами.

Имена столбцов таблицы должны быть различны, а значения столбцов однородными (однотипными).

Значения атрибутов должны быть атомарными, следовательно, отношения не могут иметь в качестве компонент другие отношения.

Должна соблюдаться ссылочная целостность для внешних ключей.

Порядок следования строк в таблице несуществен, так как влияет лишь на скорость доступа к строке.

5.6.1. Операции над отношениями

Известно, что основной единицей обработки в операциях реляционной модели данных является отношение, а не отдельные ее записи, как это принято в традиционных языках обработки данных. Эффективность реляционной СУБД определяется способностью выполнять над отношениями восемь операций:

- * объединение,
- * пересечение,
- * разность,
- * декартово произведение,
- * деление,
- * проекцию,
- * соединение,
- * выбор.

Рассмотрим их выполнение на примерах.

Введем некоторые понятия. *Степенью* отношения называется число входящих в него атрибутов. *Мощностью* (кардинальным числом) называется число кортежей отношения. При

выполнении некоторых операций отношения должны иметь *совместимые схемы*, т.е. иметь одинаковую степень и одинаковые типы основных атрибутов

Объединение. Операция выполняется над двумя совместимыми отношениями:

<i>ПРЕДМЕТ1</i>		<i>ПРЕДМЕТ2</i>	
<i>Название</i>	<i>Семестр</i>	<i>Название</i>	<i>Семестр</i>
Высшая математика	1	Физика	2
Информатика	1	Информатика	1
САПР	8	ТМС	9
САПР ТП	9	САПР	8
Детали машин	6		

Результат объединения все кортежи первого отношения и недостающие кортежи из второго отношения:

<i>ПРЕДМЕТ</i>	
<i>Название</i>	<i>Семестр</i>
Высшая математика	1
Информатика	1
САПР	8
САПР ТП	9
Детали машин	6
Физика	2
ТМС	9

Пересечение. Результат пересечения содержит только те кортежи первого отношения, которые есть во втором:

<i>ПРЕДМЕТ</i>	
<i>Название</i>	<i>Семестр</i>
Информатика	1
САПР	8

Разность. Результат вычитания включает только те кортежи первого отношения, которых нет во втором:

<i>ПРЕДМЕТ</i>	
<i>Название</i>	<i>Семестр</i>
Высшая математика	1
САПР ТП	9
Детали машин	6

Декартово произведение. Здесь отношения–операнды могут иметь разные схемы:

<i>СТУДЕНТЫ</i>	
<i>Фамилия_И.О.</i>	
Иванов И.И.	
Петров П.П.	
Владимиров В.В.	

<i>ЭКЗАМЕНЫ</i>	
<i>Предмет</i>	<i>Дата</i>
Высшая математика	5.01.99
Детали машин	15.01.99

Степень результирующего отношения равна сумме степеней отношений операндов, а мощность – произведению их мощностей:

<i>ЭКЗ ВЕДОМОСТЬ</i>		
<i>Фамилия_И.О.</i>	<i>Предмет</i>	<i>Дата</i>
Иванов И.И.	Высшая математика	5.01.99
Иванов И.И.	Детали машин	15.01.99
Петров П.П.	Высшая математика	5.01.99
Петров П.П.	Детали машин	15.01.99
Владимиров В.В.	Высшая математика	5.01.99
Владимиров В.В.	Детали машин	15.01.99

Деление. Отношение–делитель должно содержать подмножество атрибутов отношения–делителя. Результирующее отношение содержит только те атрибуты делимого, которых нет в

делителе. В него включают только те кортежи, декартовы произведения которых с делителем содержатся в делимом:

ЭКЗ_ВЕДОМОСТЬ

Фамилия_И.О.	Предмет	Оценка
Иванов И.И.	Высшая математика	5
Иванов И.И.	Детали машин	5
Петров П.П.	Высшая математика	4
Петров П.П.	Детали машин	4
Владимиров В.В.	Высшая математика	4
Владимиров В.В.	Детали машин	5

:

РЕЗУЛЬТАТЫ

Предмет	Оценка
Высшая математика	5
Детали машин	4

=

СТУДЕНТЫ

Фамилия_И.О.
Иванов И.И.
Петров П.П.

Проекция. Эта операция выполняется над одним отношением на некоторые атрибуты. Результирующее отношение включает часть атрибутов исходного, на которые выполняется проекция, например "Курс" и "Специальность". Кортежи-дубликаты отсутствуют:

СТУДЕНТ

Фамилия_И.О.	Курс	Специальность
Иванов И.И.	4	Технология машиностроения
Петров П.П.	3	История
Владимиров В.В.	4	Технология машиностроения
Петровский П.С.	5	Технология машиностроения
Сидоров С.С.	5	Экономика

→

СПЕЦИАЛЬНОСТИ

Курс	Специальность
4	Технология машиностроения
5	Технология машиностроения
3	История
5	Экономика

Соединение. Операция соединения выполняется над двумя отношениями. В каждом отношении выделяется атрибут, по которому будет производиться соединение.

ГРУППЫ

Специальность	Код студента	Код студента	Фамилия_И.О.	Курс
Технология машиностроения	1	1	Иванов И.И.	4
Бухучет	5	2	Петров П.П.	3
История	3	3	Владимиров В.В.	4
Экономика	6	4	Петровский П.С.	5
		5	Сидоров С.С.	5
		6	Семенов С.С.	2

В качестве атрибута для соединения выберем "Код студента". Результирующее отношение включает все атрибуты первого отношения и второго отношения:

СТАРОСТЫ ГРУПП

Специальность	Код студента	Фамилия_И.О.	Курс
Технология машиностроения	1	Иванов И.И.	4
Бухучет	5	Сидоров С.С.	5
История	3	Владимиров В.В.	4
Экономика	6	Семенов С.С.	2

Выбор. Операция выполняется над одним отношением. Результирующее отношение содержит подмножества кортежей, выбранных по некоторому условию, например "Курс" > 3:

СТУДЕНТ

Фамилия_И.О.	Курс	Специальность	Фамилия_И.О.	Курс
Иванов И.И.	4	Технология машиностроения	Иванов И.И.	4
Петров П.П.	3	История	Петровский П.С.	5
Владимиров В.В.	3	Технология машиностроения	Сидоров С.С.	5
Петровский П.С.	5	Бухучет		
Сидоров С.С.	5	Экономика		

→

Как видно, рассмотренные операции позволяют выделять из отношений их подмножества, при необходимости снова объединять эти подмножества в более "крупные" отношения, обновлять содержимое отношений и представлять их в требуемом виде.

Однако при выполнении операций обновления над отношениями базы данных могут возникать побочные эффекты, если между атрибутами отношений существуют нежелательные функциональные зависимости. Устранение побочных эффектов достигается *нормализацией отношений базы данных*, т.е. рациональной группировкой атрибутов в отношениях.

5.6.2. Нормализация отношений

Отношения реляционной базы данных содержат как структурную, так и семантическую (смысловую) информацию. Структурная информация задается схемой отношения, а семантическая выражается функциональными связями между атрибутами, известными и учитываемыми в схеме.

Состав атрибутов отношений базы данных должен удовлетворять двум основным требованиям:

- * между атрибутами не должно быть нежелательных функциональных зависимостей;
- * группировка атрибутов должна обеспечивать минимальное дублирование данных, обеспечивать их обработку и обновление без трудностей.

Удовлетворение этих требований достигается нормализацией отношений БД. *Нормализация отношений* – это пошаговый обратимый процесс декомпозиции (разложения) исходных отношений БД на другие, более мелкие и простые отношения. При этом устанавливаются либо выясняются все возможные функциональные зависимости.

В аппарате нормализации отношений определены различные *нормальные формы*. Каждая нормальная форма ограничивает *типы* допустимых *функциональных зависимостей* отношения. Для разработки реляционных баз данных выделяются в основном три нормальные формы (сокращенные названия: 1НФ, 2НФ, 3НФ). Самая совершенная из них – третья. Сегодня уже определены 4НФ и 5НФ.

Известно, что *отношение приведено к 1НФ*, если все его атрибуты простые (атомарные). Рассмотрим типы функциональных зависимостей и остальные нормальные формы.

В качестве примера для иллюстрации используем следующее отношение с составным ключом, состоящим из двух атрибутов: "Личный_номер" и "Название_предмета":

ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ(Личный_номер, Название_предмета, Количество_часов, Фамилия, Должность, Оклад, Кафедра, Телефон).

Введем понятие *функциональной зависимости*. Пусть имеются два атрибута: А и В. Если в любой момент времени каждому значению А соответствует не более чем одно значение атрибута В, говорят, что В функционально зависит от А. Функциональная зависимость обозначается так: $A \rightarrow B$.

В нашем примере:

Должность \rightarrow *Оклад*; *Личный_номер* \rightarrow *Фамилия*; *Личный_номер* \rightarrow *Фамилия* и т.д.

Если отношение находится в 1НФ, то все неключевые атрибуты функционально зависят от ключа. Но степень зависимости может быть различной.

Если неключевой атрибут зависит только от части ключа, то говорят о *частичной зависимости*. В нашем примере неключевой атрибут "*Количество_часов*" зависит от части ключа, т.е. только от атрибута "*Название_предмета*".

Если неключевой атрибут зависит от всего составного ключа и не находится в частичной зависимости от его частей, то говорят о его *полной функциональной зависимости* от составного ключа. В нашем примере нет атрибутов, находящихся в полной функциональной зависимости от составного ключа.

Если для атрибутов А, В, С выполняются условия $A \rightarrow B$ и $B \rightarrow C$, но обратная зависимость отсутствует, то говорят, что С зависит от А транзитивно. Пример *транзитивной зависимости*:

$$\text{Фамилия} \rightarrow \text{Кафедра} \rightarrow \text{Телефон}$$

В отношениях между атрибутами может существовать еще один тип зависимости – *многозначная зависимость*. В отношении R атрибут В многозначно зависит от А ($A \twoheadrightarrow B$), если каждому значению А соответствует множество значений В, никак не связанных с другими атрибутами из R.

Многозначная зависимость возможна при наличии в отношении хотя бы трех атрибутов: ключа и не менее двух независимых друг от друга атрибутов.

Для иллюстрации многозначной зависимости рассмотрим отношение с ключом "*Фамилия*":

$$\text{ПРЕПОДАВАТЕЛЬ}(\text{Фамилия}, \text{Группа}, \text{Предмет})$$

Между преподавателем и группами студентов имеется связь типа "один–ко–многим" (1:M), поскольку преподаватель может читать лекции в одной и более группах, однако каждой группе соответствует один преподаватель:

$$\text{Фамилия} \overset{1}{\longleftrightarrow} \overset{M}{\text{Группа}}$$

Между преподавателем и предметами имеется связь типа "многие–ко–многим" (M:N), поскольку преподаватель может читать один и более предметов, и наоборот, один предмет могут читать несколько преподавателей:

$$\text{Фамилия} \overset{M}{\longleftrightarrow} \overset{N}{\text{Группа}}$$

В рассматриваемом отношении существуют независимые многозначные зависимости $\text{Фамилия} \twoheadrightarrow \text{Группа}$ и $\text{Фамилия} \twoheadrightarrow \text{Предмет}$, так как значения многозначных атрибутов "*Группа*" и "*Предмет*" никак не связаны между собой и возможно изменение их значений в любой строке отношения.

На практике могут встречаться самые необычные случаи функциональных зависимостей,

отыскание которых может быть сложным. Кроме того, следует иметь в виду, что некоторые функциональные зависимости могут изменяться со временем. Например, в отношении *ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ* преподаватель закреплен за одной кафедрой. Но со временем он может начать работать параллельно и на другой кафедре. Тогда ему уже нельзя будет однозначно определить кафедру.

Каждая нормальная форма:

- * ограничивает определенный тип функциональной зависимости и
- * устраняет соответствующие аномалии при выполнении операций над отношениями БД.

При рассмотрении нормальных форм будем отмечать все свойственные им аномалии и нежелательные эффекты. Для иллюстрации используем отношение с составным ключом *ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ* из предыдущего параграфа. Оно находится в 1НФ, поскольку не содержит составных атрибутов.

В этом отношении можно отметить частичную функциональную зависимость атрибутов "Фамилия", "Должность", "Оклад", "Кафедра", "Телефон" от части "Личный_номер" составного ключа. Такая частичная зависимость приводит к следующим аномалиям:

Имеет место дублирование данных о преподавателе, поскольку преподаватель может читать несколько предметов.

Существует проблема контроля избыточности данных, так как изменение, например, оклада влечет за собой необходимость поиска и изменения значений окладов во всех кортежах с данным преподавателем.

Возникает проблема с преподавателями, которые в данное время не ведут предметы (находятся на повышении квалификации), и с лаборантами, которые вообще не ведут предметов. А именно, преподавателя без предмета невозможно включить в отношение. И наоборот, если преподаватель увольняется и удаляется из отношения, то будет удален и предмет, хотя предмет должен продолжать читаться.

Таким образом, отношение в 1НФ требует дальнейших преобразований.

Вторая нормальная форма. Отношение находится в 2НФ, если оно находится в 1НФ и каждый неключевой атрибут функционально полно зависит от составного ключа.

Чтобы устранить частичную зависимость и привести рассматриваемое отношение к 2НФ, необходимо разложить его на два отношения следующим образом:

- * построить проекцию без атрибутов, которые находятся в частичной функциональной зависимости от составного ключа;
- * построить проекцию на часть составного ключа и атрибуты, зависящие от этой части.

В итоге получим два отношения, *ПРЕДМЕТ* и *ПРЕПОДАВАТЕЛЬ*, находящиеся в 2НФ:

ПРЕДМЕТ(Личный_номер, Название_предмета, Количество_часов)

ПРЕПОДАВАТЕЛЬ(Личный_номер, Фамилия, Должность, Оклад, Кафедра, Телефон).

В полученном отношении *ПРЕПОДАВАТЕЛЬ* имеются транзитивные функциональные зависимости, например:

Личный_номер → Кафедра → Телефон

Личный_номер → Должность → Оклад

Наличие транзитивных зависимостей порождает неудобства и аномалии следующего характера (на примере атрибута "*Телефон*"):

Имеет место дублирование информации о телефоне для преподавателей одной кафедры.

Существует проблема контроля избыточности, поскольку изменение номера телефона кафедры влечет за собой необходимость поиска и изменения номеров всех преподавателей этой кафедры.

Нельзя включить данные о новой кафедре (название и номер телефона), если на данный момент еще отсутствуют преподаватели. И наоборот, при увольнении всех преподавателей с кафедры данные о ней нельзя сохранить.

Таким образом, отношение в 2НФ также может требовать дальнейших преобразований.

Третья нормальная форма. Отношение находится в 3НФ, если оно находится в 2НФ и в нем отсутствуют транзитивные зависимости неключевых атрибутов от ключа. В нашем примере получим три отношения:

ПРЕПОДАВАТЕЛЬ(Личный_номер, Фамилия, Должность)

ДОЛЖНОСТЬ(Должность, Оклад)

КАФЕДРА(Кафедра, Телефон)

3НФ освобождает от избыточности и аномалий выполнения операций включения, удаления и обновления (изменения), если отношение имеет один ключ и другие зависимости, в том числе многозначные, в нем отсутствуют. Но если при этом имеются другие зависимости, кроме зависимости от ключа, то 3НФ не обеспечивает отсутствия аномалий операций. В этом случае применяют усиленную 3НФ, 4НФ, 5НФ.

Как видно, нормализация отношений выполняется декомпозицией (разложением) их схем. Декомпозиция должна гарантировать *обратимость*, т.е. обеспечивать получение исходных отношений путем выполнения операций соединения над их проекциями. Обратимость предполагает, что

- * отсутствуют потери кортежей,
- * не появляются ранее отсутствовавшие кортежи,
- * сохраняются функциональные зависимости.

Еще раз отметим, что процесс нормализации отношений последовательно устраняет следующие типы функциональных зависимостей:

- * частичные зависимости неключевых атрибутов от ключа;
- * транзитивные зависимости неключевых атрибутов от ключа;
- * зависимости ключей от неключевых атрибутов;
- * независимые многозначные зависимости и т.д.

При устранении этих зависимостей исключается дублирование данных и аномалии операций включения, удаления, обновления. Уровень нормализации отношения зависит от его семантики, заданной функциональными связями. На практике достаточно знания и учета рассмотренных нормальных форм.

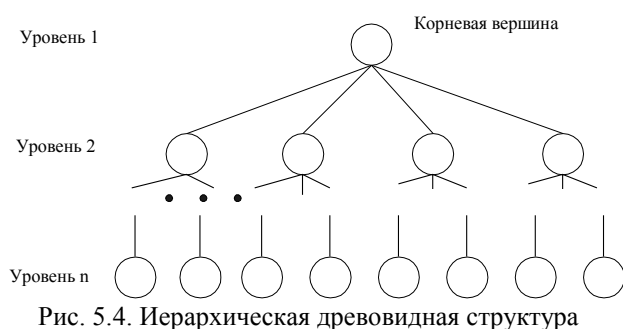
В заключение определим *место процесса нормализации* в проектировании БД:

- * вначале составляются исходные отношения проекта БД с использованием объектно–связной модели для отображения объектов предметной области и связей между ними;
- * затем производится нормализация, т.е. композиция–декомпозиция исходных отношений и назначение ключей новых отношений в соответствии с правилами нормализации;
- * далее схемы нормализованных отношений описываются средствами СУБД и вводятся в ЭВМ.

Отметим, что нормализация увеличивает число отношений в БД и тем самым может возрастет время обработки. Но в то же время благодаря корректности и устранению дублирования данных ускоряется выполнение операций доступа к данным.

5.7. Иерархический и сетевой подходы

Иерархические СУБД, частично реализующие и элементы сетевого подхода, широко распространены в связи со сравнительно малым временем поиска данных. Реляционные БД можно перестроить в иерархические и наоборот.



Иерархическая модель данных основана на понятии деревьев, состоящих из вершин и ребер. Вершина дерева ставится в соответствие совокупности атрибутов данных, характеризующих некоторый объект. Вершины и ребра дерева как бы образуют иерархическую древовидную структуру, состоящую из n уровней (рис. 5.4).

Первую вершину в дереве называют корневой вершиной иерархической древовидной структуры. Она удовлетворяет семи условиям:

1. Иерархия начинается с корневой вершины.
2. Каждая вершина соответствует одному или нескольким атрибутам.
3. На уровнях с большим номером находятся зависимые вершины. Вершина предшествующего уровня является начальной для новых зависимых вершин.
4. Каждая вершина, находящаяся на уровне i , соединена с одной и только одной

вершиной уровня $i-1$, за исключением корневой вершины.

5. Корневая вершина может быть связана с одной или несколькими зависимыми вершинами.

6. Доступ к каждой вершине происходит через корневую по единственному пути.

7. Существует произвольное количество вершин каждого уровня.

Иерархическая модель данных состоит из нескольких деревьев, т.е. является лесом. Каждая корневая вершина образует начало записи логической базы данных.

В иерархических БД обеспечивается удобный и быстрый поиск, если запрос строится по первичным ключам, в соответствии с которыми выбрана структура дерева. Если в запросе фигурируют другие атрибуты, то поиск может свестись к полному просмотру БД и, следовательно, резко замедлиться.

В *сетевой модели данных* элементарные данные и отношения между ними представляются в виде ориентированной сети (вершины – данные, дуги – отношения). База данных, описываемая сетевой моделью, состоит из нескольких областей. Область содержит записи. Одна запись состоит из нескольких полей. Набор, состоящий из записей, может размещаться в одной или нескольких областях (рис. 5.5). В сетевой модели данных объекты предметной области объединяются в сеть. Графически сетевая модель описывается прямоугольниками и стрелками. Каждый тип записи может содержать множество атрибутов.

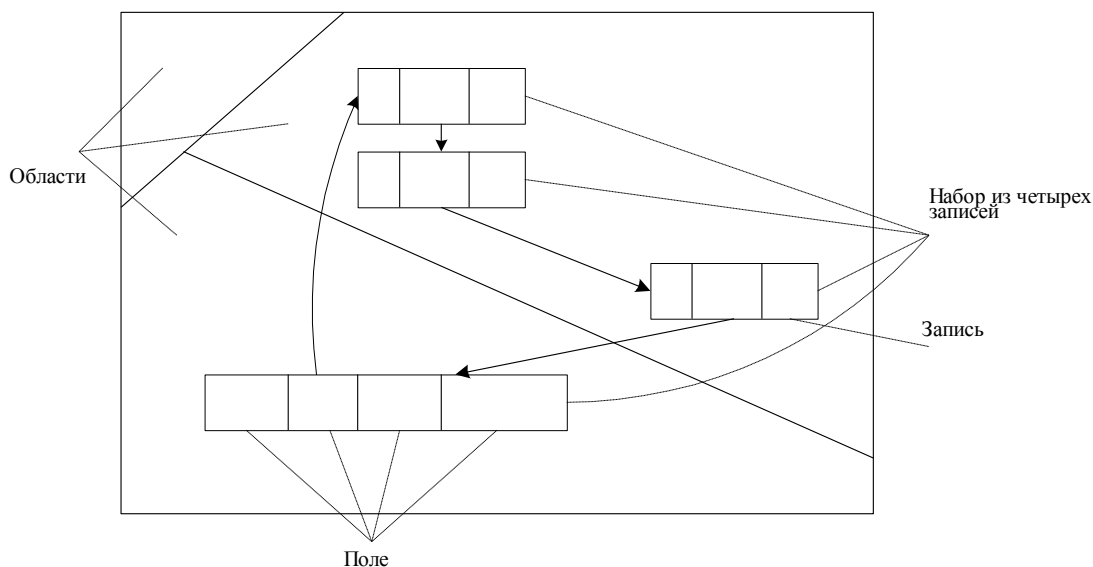


Рис. 5.5. База данных, описываемая сетевой моделью

Важное отличие сетевой модели данных от иерархической состоит в том, что в сетевой модели каждая запись может быть в любом числе наборов и может находиться как на верхнем, так и на нижнем иерархическом уровне. Следовательно, любая запись может быть задана как точка входа. Основные достоинства сетевой модели данных – наличие реализованных СУБД, обеспечивающих эту модель, простота реализации отношений "многие ко многим". Основным недостатком – ее сложность. При реорганизации БД возможна потеря независимости данных.

5.8. Организация базы данных на физическом уровне. Особенности баз данных САПР

В реляционных БД организуется обычное хранение информации в виде двумерных таблиц. Нужные пользователю таблицы, ключи и операции (или предикаты) содержатся в самом запросе.

В иерархических БД структура данных – список (цепное представление), в котором система указателей позволяет просматривать дерево сверху вниз и слева. Если значения ключа не совпадают, происходит переход к записи этого же уровня, если совпадают – переход к записям соседнего нижнего уровня.

Поиск нужной информации по заданному ключу выполняется с помощью индексного файла, в который вынесены значения ключей и соответствующих им адресов памяти, по которым хранятся записи. *Индексный файл* – это отношение между значениями первичного ключа и адресами. Наличие индексного файла позволяет ускорить поиск, так как просмотр всех полей записи производится лишь при совпадении значений ключа. Кроме того, можно вести ускоренный поиск по нескольким различным ключам, если ввести для каждого ключа свой индексный файл.

Разнохарактерность проектных процедур в САПР обуславливает разнообразие типов и структур данных, которыми обмениваются пользователи и прикладные программы через БД. База данных САПР должна быть приспособлена для хранения:

- * сведений справочного характера об используемых материалах, комплектующих деталях и приборах, инструменте, оборудовании, оснастке, которые первоначально имеют форму таблиц;
- * информации о чертежах и схемах, требующих для своего представления в ЭВМ специального кодирования;
- * текстовых документов типа пояснительных записок, инструкций по проектированию, описаний программ и др.

По мере развития проекта информация о нем существенно изменяется – увеличивается объем проектной документации, обновляются массивы данных из-за итерационного характера процесса проектирования, появляются альтернативные варианты и т. п.

Названные особенности отличают БД САПР от БД большинства АСУ. Их учет при проектировании баз данных в САПР обычно приводит к выделению интегрированной и локальных БД, предпочтительности выбора реляционной модели данных.

При создании конкретных САПР возникает *проблема выбора и оценки автоматизированного банка данных*. Правильность выбора автоматизированного банка данных во многом определяет эффективность создания и удобство функционирования САПР. Таким образом, эта проблема затрагивает не только разработчиков, но и пользователей САПР.

В настоящее время информационное обеспечение САПР строится, как правило, по принципам, выработанным и апробированным при использовании автоматизированных банков данных в информационных системах общего назначения. Автоматизированный банк данных относится к инвариантным средствам САПР. Поэтому создатели САПР обычно не разрабатывают специальные автоматизированные банки данных, а выбирают из готовых общего назначения и адаптируют их к целям конкретной САПР.

Выбор СУБД для разрабатываемых САПР должен быть направлен главным образом на удовлетворение требований и потребностей пользователей–проектировщиков.

Факторы, определяющие СУБД, разбиваются на две группы:

- * факторы, характеризующие предполагаемое применение СУБД;
- * факторы, характеризующие функциональные возможности СУБД.

В *характеристику применения* обычно включают:

- * классификацию и характеристику конечных пользователей и обслуживающего персонала СУБД (системные аналитики, администраторы автоматизированного банка данных, системные и прикладные программисты);
- * сложность и форму запросов;
- * режим работы (многопользовательский, пакетный, интерактивный и т. д.);
- * эволюционные характеристики (характеристики пользователей до и после внедрения СУБД, рост оперативности пользователей, появление новых задач);
- * характеристики данных;
- * характеристики ресурсов (конфигурация ЭВМ, программное обеспечение, квалификация обслуживающего персонала).

Функциональные возможности можно характеризовать следующими факторами:

- * назначение СУБД (обобщенная или специализированная на определенный круг задач);
- * поддерживаемые структуры данных (иерархические, сетевые, реляционные);
- * используемое оборудование (производительность процессора, объем главной памяти, типы и емкость внешней памяти);
- * используемое программное обеспечение;
- * трудности освоения и эксплуатации, наличие документации.

Выбор конкретного автоматизированного банка данных для САПР затруднен тем, что его эффективность зависит от многих факторов. И важно не только знать значения этих факторов для различных автоматизированных банков данных, но и выбирать наиболее значимые из них.

При разработке концепции САПР следует учитывать не только положительные стороны автоматизированного банка данных, но и “плату” за реализацию возможностей,

предоставляемых им. К главным компонентам “платы” за использование автоматизированного банка данных в САПР следует отнести:

- * затраты на управление автоматизированным банком данных: требуются значительные ресурсы (память, машинное время) для функционирования СУБД и операционной системы ЭВМ;
- * трудности соблюдения разработчиками и пользователями прикладных программ тех ограничений, которые положены в основу СУБД;
- * необходимость создания специального административного подразделения, ответственного за эксплуатацию автоматизированный банк данных (администратор базы данных, администратор по управлению системой и т. д.);
- * угроза безопасности данных, связанная с большой их централизацией: неисправности в технических средствах и ошибки в прикладных программах могут испортить данные для многих программ и многих пользователей.

Таким образом, в арсенале специалистов имеется большое количество СУБД, которые могут быть использованы в практике проектирования САПР.

5.9. Жизненный цикл информационной системы САПР и проектирование баз данных

Любой объект, создаваемый человеком, проходит три основных периода:

- * проектирование, когда составляются схема, чертежи объекта, выполняются расчеты и пр.;
- * реализацию, когда проект находит свое материальное воплощение;
- * эксплуатацию, когда построенный объект используется, ремонтируется, перестраивается.

Жизненный цикл информационной системы также можно разбить на 3 основные стадии:

- * "бумажное" проектирование;
- * программная реализация;
- * эксплуатация.

На *стадии проектирования* информационной системы проектировщик должен проделать следующую работу:

Обследовать предметную область автоматизации.

Определить объекты и перечень их атрибутов, для каждого объекта выделить первичные ключи и провести нормализацию.

Установить все структурные, иерархические связи между объектами и все запросные связи, обеспечивающие обработку всех запросов пользователей к БД. Начертить схему проекта со всеми объектами и связями.

Выработать технологию обслуживания информационной системы, т.е. определить

порядок сбора, хранения данных в БД, частоту и форматы ввода–вывода данных, правила работы всех групп пользователей. Проект должен обеспечивать простоту и удобство будущей эксплуатации информационной системы, защиту данных от некорректных обновлений пользователями и от разрушений при сбоях компьютера.

Выбрать компьютер и инструментальные средства (конкретную СУБД) для реализации. При этом надо оценить требуемые объемы памяти и трудоемкость разработки программ. В свою очередь проект должен учитывать ограничения, накладываемые компьютером на объем и время доступа к внешней и оперативной памяти, на быстродействие, и обеспечивать эффективное функционирование информационной системы на выбранной технике.

Проверить корректность проекта. Проект должен адекватно, на требуемом уровне детальности, отображать предметную область, т.е. всем выделенным объектам и процессам предметной области должны соответствовать данные и процедуры обработки в ЭВМ.

Определить сроки реализации информационной системы.

На *стадии программной реализации* необходимо выполнить следующие пункты:

Описать средствами СУБД и ввести в ЭВМ схемы всех отношений.

Разработать интерфейсы пользователей с БД. Сюда входят разработка экранных форм для ввода и отображения данных, удобных экранных способов обращения и доступа к данным в БД, порядка ввода и обновления данных; определение размеров и состава порций одновременно отображаемых на экране данных, порядка их размещения. Каждая картинка на экране должна обеспечивать максимальную информативность и удобство восприятия, создавать привычную для пользователя среду.

Разработать программное обеспечение информационной системы для всех приложений.

Заполнить информационную систему отладочными данными и отладить ее.

Провести тестирование системы и скорректировать технологию ее обслуживания.

Составить необходимые инструкции по системе и обучить пользователей.

Стадия эксплуатации начинается с наполнения системы реальными данными, после чего происходит непосредственно использование информационной системы, поддержание ее функционирования. Стадия эксплуатации может включать совершенствование системы и разработку новых приложений в случае развития и изменения предметной области автоматизации.

Проектирование баз данных информационной системы целесообразно выполнять по методике, узловые моменты которой изложены ниже.

Обследование предметной области выполняется путем бесед и интервью с Заказчиком. В роли Заказчика выступают как руководитель предметной области, так и все будущие

пользователи системы.

Первая беседа должна дать проектировщику начальное представление о предметной области (без деталей). Поэтому ему следует выслушать подробное сообщение Заказчика. К последующим встречам проектировщик должен заранее готовить перечень вопросов Заказчику. Результаты каждого интервью должны быть написаны на бумаге и подписаны Заказчиком, поскольку он несет ответственность за точность передаваемой информации. Неточные ответы могут выявиться на последующих этапах, привести к трудоемкой переделке системы и соответствующему повышению ее разработки.

После первых встреч с Заказчиком следует получить ответы на следующие вопросы:

⇒ Каковы границы предметной области, возможности ее изменения и развития?

⇒ Каков перечень фрагментов предметной области? Получить представление о каждом фрагменте.

⇒ Какая информация и с какой степенью детальности нужна пользователям каждого фрагмента? Определить перечень пользователей и их информационные потребности.

⇒ Какие процессы передачи и обработки данных происходят в каждом фрагменте, с какой интенсивностью?

⇒ Какова существующая технология накопления и обработки информации в предметной области? Каковы ее "узкие места", требующие введения автоматизации?

⇒ На каком компьютере планируется реализовать систему?

⇒ Каковы требования к технологии функционирования информационной системы? Как часто поступает и корректируется информация? Кто несет ответственность за ее достоверность? Нужны ли адаптация и настройка информационной системы в случае изменений в предметной области ?

Результатом обследования предметной области может быть документ, называемый "Техническим заданием", который подписывается первым лицом со стороны Заказчика. В техническом задании должны быть указаны требования к разрабатываемой информационной системе, отражены все приведенные выше характеристики.

Определение объектов и атрибутов также требует встреч с Заказчиком. При этом проектировщик непосредственно знакомится со всеми входными и выходными документами, формами, справками, другими сообщениями, циркулирующими в каждом фрагменте предметной области.

Здесь проектировщик должен уяснить тип создаваемой им БД – документальная, фактографическая или смешанная БД. Тип БД влияет на порядок выявления объектов и их атрибутов.

В документальной БД сущностями являются собственно текстовые документы. Перечень атрибутов выявить несложно, важно только определить структуру объектов.

В фактографической БД сущностями могут быть любые представляющие интерес процессы или объекты. И сведения о них могут поступать из различных сообщений и документов. Поэтому здесь выявление нового атрибута объекта происходит путем анализа каждого нового сообщения или документа, содержащего сведения об объекте.

Для каждого атрибута устанавливают и выписывают в виде таблицы следующие основные характеристики:

- тип атрибута (цифровой, буквенный, логический и пр.), длину и допустимый диапазон значений;
- процент наличия значений атрибута в экземплярах объекта. Например, значения атрибута "спортивный_разряд" будут присутствовать не во всех кортежах;
- вычислимость значений атрибута из другой информации. Например, значения атрибута "годовая_программа_выпуска" можно подсчитать, зная месячную программу, и не тратить память для их хранения;
- частота использования атрибута. В случае нехватки внешней памяти редко используемые данные, например архивные, могут размещаться на других носителях.

Для каждого объекта также определяются:

- * ключ или возможные ключи, по которым будет осуществляться обращение к экземплярам (кортежам) объекта;
- * количество экземпляров объекта и процент ежемесячного роста. Этим определяется предельное количество кортежей в отношении. Большие объемы (более 1000 записей) замедляют процедуры поиска и делают систему неповоротливой. В этом случае следует разбивать отношения на более мелкие, классифицируя объекты по некоторым признакам, задаваемых атрибутами;
- * частота использования и обновления экземпляров объекта. Информация в БД может быть редко обновляемой, например нормативно–справочные данные, и часто обновляемой, имеющей короткий срок существования.

На этом же шаге производят так называемое *внешнее кодирование*. Оно заключается в замене длинных текстовых значений атрибутов короткими кодами. Например, значения атрибута "факультет" можно закодировать так: МС – машиностроительный, ИС – инженерно–строительный, Ю – юридический, ИФ – историко–филологический, ФЭ – финансово–экономический и т.п. Внешнее кодирование требует создания некоторого "справочного объекта" с двумя атрибутами, например

ФАКУЛЬТЕТ(Код_факультета, Название).

Пользователю БД теперь достаточно ввести код – и он будет заменяться соответствующим значением. Кроме того, внешнее кодирование сокращает расход памяти. В БД может быть несколько справочных объектов для разных атрибутов. После определения всех

объектов и их атрибутов в каждом отношении выделяют первичный ключ и проводят их нормализацию, приводя отношения к 3НФ.

Определение всех запросов пользователей к БД также выполняется в контакте с Заказчиком. Цель встреч – составление полного перечня запросов для каждого фрагмента предметной области. За каждым запросом должны стоять определенные процессы обработки данных и запросные связи, по которым должна происходить навигация от объекта к объекту для удовлетворения запроса.

Примеры запросов могут быть такими: "Для данного факультета выдать список преподавателей с указанием ученой степени" или "Для всех факультетов выдать список студентов–отличников с указанием курса и специальности".

Некоторой запросной связи может не соответствовать структурная связь. Структурная связь между двумя объектами задается иерархической подчиненностью одного объекта другому. Если такой подчиненности нет, то структурная связь отсутствует.

Определение структурных связей выполняется после анализа всех запросных связей. Существующие структурные связи могут не удовлетворять всех запросов. Если для какой–либо запросной связи отсутствует структурная связь, то она устанавливается. Это обычно приводит к созданию новых объектов–связок (связных отношений).

Например, пусть имеются два структурно не связанных объекта (отношения) *СПЕЦИАЛЬНОСТЬ* и *ПРЕДМЕТ*. Тогда запрос "Какие предметы проходят на некоторой специальности на первом курсе" не может быть удовлетворен. Структурная связь может быть определена посредством введения объекта–связки

ИЗУЧАЮТ(Код_специальности, Код_предмета, Курс),

в котором каждой специальности будут соответствовать предметы и номера курсов.

После установления всех структурных связей производится при необходимости дальнейшая нормализация отношений. Далее следует начертить схему проекта со всеми объектами и структурными связями.

Проектирование БД завершается *проверкой корректности и полноты* полученного проекта. Она состоит в проверке возможности выполнения всех запросов к БД.