САНКТ-ПЕТЕРБУРГСКИЙ УНИВЕРСИТЕТ
УПРАВЛЕНИЯ И ЭКОНОМИКИ

Т. **А. Черняк, С. В. Удахина,**
М. **А. Косухина**

ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ СУБЪЕКТОВ ЭКОНОМИЧЕСКОЙ ДЕЯТЕЛЬНОСТИ: БАЗЫ ДАННЫХ И ЗНАНИЙ

Учебное пособие

Санкт-Петербург
2015

УДК 004
ББК 32.81
Ч 49

Утверждено на заседании кафедры информационных технологий и математики в качестве учебного пособия по направлению 230700.62 «Прикладная информатика» протокол № 7 от 25.02.2015 г.

Рецензенты:

профессор кафедры информационных технологий и математики Санкт-Петербургского университета управления и экономики, д. т. н. ***Костин Г. А.***

заведующая кафедрой информационных систем в экономике филиала Санкт-Петербургского государственного экономического университета в г. Кизляре, к. э. н. ***Абдулаева З. Л.***

Черняк Т. А., Удахина С. В., Косухина М. А.

Ч 49 Информационное обеспечение субъектов экономической деятельности: Базы данных и знаний: учебное пособие. — СПб.: Издательство Санкт-Петербургского университета управления и экономики, 2015. — 200 с.: ил.

ISBN 978-5-94047-732-7

Учебное пособие разработано в соответствии со стандартом ФГОС ВПО 230700 (09.03.03) «Прикладная информатика» и представляет курс лекционного материала, упорядоченного в соответствии с рабочей программой дисциплины «Базы данных и знаний». К каждой теме приводится список необходимых терминов. Наиболее сложные темы снабжены самостоятельными заданиями для закрепления пройденного материала.

Издание предназначено для студентов направления 230700 «Прикладная информатика в экономике» всех форм обучения.

УДК 004
ББК 32.81

ISBN 978-5-94047-732-7

ВВЕДЕНИЕ

Учебное пособие предназначено для студентов направления 230700 «Прикладная информатика в экономике» всех форм обучения. Разработанное учебное пособие соответствует стандарту ФГОС ВПО.

В пособии даются основные термины необходимые для усвоения материала по данной дисциплине. Прослеживается связь с дисциплинами направления «Интеллектуальные информационные системы», «Информационные системы».

Пособие представляет собой набор лекционного материала упорядоченного в соответствии с рабочей программой дисциплины. К каждой теме дается список терминов необходимых для усвоения. К наиболее сложным темам даются самостоятельные задания для закрепления пройденного материала.

В конце пособия представлен тестовый материал, задачи, а также материал для самостоятельной работы после изучения дополнительных источников.

Тема 1

ПОНЯТИЕ ИНФОРМАЦИИ

1. Понятие информации, теории информации
2. Свойства информации
3. Экономическая информация как основа базы данных.

Информация является фундаментом для любой автоматизированной системы. Она — один из основных ресурсов на котором базируется экономика постиндустриального общества. Обладая своеобразными свойствами не присущими другим ресурсам, ее ценность увеличивается с увеличением объема, из-за уменьшения энтропии (неопределенности).

Приведем основные термины понятия ИНФОРМАЦИЯ.

По определению Мишенина, информация — совокупность документированных данных, хранящихся на электронных или бумажных носителях, соответствующая определенной предметной области.

В формулировке Федерального закона Российской Федерации от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации» информация — сведения (сообщения, данные) независимо от формы их представления.

Информация — это знания, сведения, данные о каком-либо объекте, событии, процессе или явлении, передаваемые прямо или косвенно от источника к потребителю, в результате чего снимает-ся или уменьшается неопределенность сложившейся ситуации.

Все эти термины объединяет общее:

- информация кем-то или чем-то передается;
- она кому-то адресована;
- она может храниться на каком-либо носителе;
- имеет различные формы представления.

В основе информации лежат ДАННЫЕ.

Данные представляют собой набор утверждений, фактов и/ или цифр, лексически и синтаксически взаимосвязанных между собой.

Данные — это сведения о состоянии любого объекта (процесса, явления), представленные в формализованном виде и предназначенные для обработки или уже обработанные.

Любая информация в процессе обработки становится ЗНАНИЯМИ.

Знания — это результат индивидуальной познавательной деятельности. Оно жестко связано с собственником, с личностью его создателя и (или) обладателя.

Три основные теории информации:

1. *Статистическая теория информации* (основатель К. Шеннон). Информация — коммуникации и связь, в процессе которых устраняется неопределенность. В теории принимается два равновероятных исхода одного события. Используется единица информации, получившая название «бит». Эта теория получила свое распространение в период развития «линий передач» и позволила расширить их пропускную способность.

Для выбора одного сообщения из множества равновероятных событий используется формула Хартли:

$$I(N) = \log 2N,$$

где N — количество равновероятных

событий; I — количество бит в сообщении.

Для выбора наиболее вероятного сообщения при неравной вероятности событий используется формула Шеннона:

$$I \sum_{i=1}^{N} p_i \log_2 p_i,$$

где p_i — вероятность того, что именно i-е сообщение выделено в наборе из N сообщений.

Рассмотрим лишь некоторые примеры:

Задача 1

Световое табло состоит из лампочек. Каждая лампочка может находиться в одном из трех состояний («включено», «выключено» или «мигает»). Какое наименьшее количество лампочек должно находиться на табло, чтобы с его помощью можно было передать 18 различных сигналов?

Решение

В данной задаче мощность алфавита равна трем («включено», «выключено» или «мигает»). Количество необходимых сигналов 18, следовательно, $18 = 3^N$, $N = 3$ бита.

Ответ: количество лампочек равно 3.

Задача 2

В результате работы над проектом необходимо было ответить на 64 вопроса, ответ получили лишь на 4 из них. Какой объем информации был получен?

Решение: $64 = 4^N$, $N = 3$ бита.

Ответ: 3 бита.

Задача 3

На продажу выставлены акции двух предприятий. Среди них 18 тыс. первого предприятия. Сообщение о том, что продана ак-ция второго предприятия, несет 2 бита информации. Сколько всего тысяч акций выставлено на продажу?

Решение

Найдем по формуле Шеннона вероятность продажи акции второго предприятия: $\log_2 N = 2$, $N = 4$, следовательно, вероятность продажи акции второго предприятия равна 1/4 (25%), а вероятность продажи акции первого предприятия соответственно 3/4 (75%). Если 75% всех акций принадлежат первому предприятию, их количество 18 тыс., тогда 25% всех акций принадлежат второму предприятию, их количество (18*25) / 75 = 6 тыс.

Осталось найти количество всех акций выставленных на продажу 18 тыс. + 6 тыс. = 24 тыс.

Ответ: 24 тыс.

Задача 4

В код товара включает 5 символов составляется из заглавных букв (всего используется 30 букв) и десятичных цифр в любом по-рядке. Каждый символ кодируется одинаковым и минимально возможным количеством бит, а каждый код — одинаковым и ми-нимально возможным количеством байт. Определите объем памяти, необходимый для хранения 50 наименований товаров.

Решение

Количество символов используемых для кодирования товара составляет: 30 букв + 10 цифр = 40 символов. Количество информации несущий один символ равен 6 бит ($2^I = 40$, но количество информации не может быть дробным числом, поэтому берем ближайшую степень двойки большую количества символов $2^6 = 64$).

Мы нашли количество информации заложенное в каждом символе, количество символов в номере равно 5, следовательно, 5 * 6 = 30 бит. Каждый номер равен 30 битам информации, но по условию

задачи каждое наименование кодируется одинаковым и минимально возможным количеством байт, следовательно, нам необходимо узнать, сколько байт в 30 битах. Если разделить 30 на 8, получится дробное число, а нам необходимо найти целое количество байт на каждый номер, поэтому находим ближайший множитель восьмерки, который превысит количество бит, это 4 (8 * 4 = 32). Каждый номер кодируется 4 байтами.

Для хранения 50 наименований потребуется: 4 * 50 = 200 байт. ***Ответ:*** 200 байт.

2. *Кибернетическая теория* (основатель Н. Винер). Кибернетика придерживается принципа единства информации и управления. Процесс управления является процессом переработки информа-ции, что происходит если информация вращается в системе. Ин- формация — это обозначение содержания, полученного от внеш-него мира в процессе приспособления к нему (Винер) кибернети-ческая концепция подводит нас к необходимости оценить инфор-мацию как некоторое знание, имеющее одну ценностную меру по отношению к внешнему миру (семантический аспект) и другую по отношению к получателю, накопленному им знанию, познаватель-ным целям и задачам (прагматический аспект).

3. *Логико-семантическая теория* (основатели Р. Карнап, И. Бар-Хиллел и др.). Данная теория в отличие от теории Шенона постро-ена на тезаурусе. Если Шенон уделял значение количеству инфор-мации и совершенно различные сообщения, но схожие по количе-ству символов трактуются машиной одинаковым объемом, то в семантической теории большое значение имеют предыдущие знания об объекте или сообщении, получившие названия тезаурус.

Информация рассматривается в трех аспектах:

1. Синтаксическом (связан только со способом передачи информации). Если информация передается от человека к человеку в разговоре, то используются звуки, жесты мимика и т. д. для ее передачи, для передачи с помощью ЭВМ различные кодировки, для отображения используется набор символов алфавита.

2. Семантическом (отражает смысловое содержание, сведения должны соответствовать тезаурусу); известное выражение: «нельзя казнить помиловать» очень ярко отражает данный аспект.

3. Прагматическом (отражает потребительские свойства информации).

Три фазы существования информации:

1. *Ассимилированная информация* — представление сообщений в сознании человека, наложенное на систему его понятий и оценок.

2. *Документированная информация* — сведения, зафиксированные в знаковой форме на каком-то физическом носителе.

3. *Передаваемая информация* — сведения, рассматриваемые в момент передачи информации от источника к потребителю.

Процесс передачи и восприятия информации представлен на рис. 1.1.

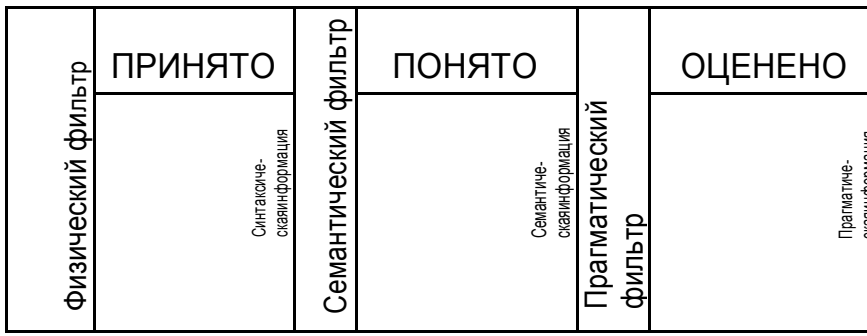


Рис. 1.1. Процесс передачи информации

Информация обладает свойствами. Одни из них атрибутивные (без которых информация не существует), показатели качества, прагматические.

Атрибутивные свойства:

- *неотрывность* информации от физического носителя и языковая природа информации. Одно из важнейших направлений информатики как науки является изучение особенно-стей различных носителей и языков информации, разработка новых, более совершенных и современных. Необходимо отметить, что хотя информация и неотрывна от физического носителя и имеет языковую природу, она не связана жестко ни с конкретным языком, ни с конкретным носителем;
- *дискретность*. Содержащиеся в информации сведения, знания дискретны, т. е. характеризуют отдельные фактические данные, закономерности и свойства изучаемых объектов, которые распространяются в виде различных сообщений, со-

стоящих из линий, составных цветов, букв, цифр, символов, знаков;

- *непрерывность*. Информация имеет свойство сливаться с уже зафиксированной и накопленной ранее, тем самым способствуя поступательному развитию и накоплению. К

показателям качества информации относят:

Репрезентативность информации, одно из свойств, используемых при статистическом анализе, которое связано с правильностью ее отбора и формирования в целях адекватного отражения свойств объекта. Важнейшее значение здесь имеют правильность концепции, на базе которой сформулировано исходное понятие, обоснованность отбора существенных признаков и связей отображаемого явления. Нарушение репрезентативности информации приводит нередко к существенным ее погрешностям. Это свойство учитывается в поисковых интернет системах.

Содержательность информации отражает семантическую емкость, находится по формуле:

$$C \frac{K}{V},$$

где K — количество семантической информации в

сообщении; V — объем обрабатываемых данных;

С — коэффициент содержательности.

С увеличением содержательности информации растет семантическая пропускная способность информационной системы, так как для получения одних и тех же сведений требуется преобразо-вать меньший объем данных. Для отражения синтаксической ем-кости используют коэффициент информативности, характеризу-ющийся отношением количества синтаксической информации (по Шеннону) к объему данных.

Достаточность (полнота) информации означает, что она содержит минимальный, но достаточный для принятия правильного решения набор показателей. Понятие полноты информации связано с ее смысловым содержанием (семантикой) и прагматикой. Как неполная, т. е. недостаточная для принятия правильного решения, так и избыточная информация снижает эффективность принимаемых пользователем решений.

Доступность отражает, как информация воспринимается пользователем. В связи с этим особое значение имеют процессы получения и преобразования информации.

Актуальность информации определяется степенью сохранения ее ценности для управления в момент использования и зависит от динамики изменения ее характеристик и от интервала времени, прошедшего с момента возникновения данной информации.

Своевременность информации означает ее поступление не позже заранее назначенного момента времени, согласованного со временем решения поставленной задачи.

Точность информации определяется степенью близости получаемой информации к реальному состоянию объекта, процесса, явления и т. п. Для информации, отображаемой цифровым кодом, известны четыре классификационных понятия точности:

- формальная точность, измеряемая значением единицы младшего разряда числа;
- реальная точность, определяемая значением единицы последнего разряда числа, верность которого гарантируется;
- максимальная точность, которую можно получить в конкретных условиях функционирования системы;
- необходимая точность, определяемая функциональным назначением показателя.

Достоверность информации — свойство информации быть правильно воспринятой. Она измеряется доверительной вероятностью необходимой точности, т. е. вероятностью того, что отображаемое информацией значение параметра отличается от истинного значения этого параметра в пределах необходимой точности. Расчет может быть рассчитан следующим процент совпадающих по смыслу сообщений от этого источника с сообщениями других (проверочных) источников в общем количестве его сооб-щений за некоторый промежуток времени (например, год). При более 70–80% совпадающих можно говорить о надежном источни-ке и достоверной информации, при 50–70% как об информирован-ном источнике, и требующей проверки информации, ниже 50% — о случайном источнике и информации низкой достоверности или не достоверной. Таким образом, достоверность источника — апос- териорная оценка, полученная в результате наблюдения за его «ин-формационной активностью».

Устойчивость информации отражает ее способность реагировать на изменения исходных данных без нарушения необходимой точности. Устойчивость информации, как и репрезентативность, обусловлена выбранной методикой ее отбора и формирования.

Такие параметры качества информации, как репрезентативность, содержательность, достаточность, доступность, устойчивость, целиком определяются на методическом уровне разработки информационных систем. Параметры актуальности, своевременности, точности и достоверности обусловливаются в большей степени также на методическом уровне, однако на их величину существенно влияет и характер функционирования системы, в первую очередь ее надежность. При этом параметры актуальности и точности жестко связаны соответственно с параметрами своевременности и достоверности.

Прагматические свойства информации:

− потенциальная эффективность; − тиражируемость и многократность использования;

− неаддитивность, некоммуникативность, неассоциативность;

− коммулятивность; − зависимость фактической реализуемости и эффективности от степени использования информации;

− абстрактность; − ценность информации субъективна;

− потребление информации требует определенных навыков и усилий;

− адресность; − парность философских категорий: материи и информации.

Основой современных баз данных чаще является экономическая информация:

Экономическая информация выступает как предмет; средство; результат.

Экономическая информация — информация об экономических процессах (производстве, распределении, обмене и потреблении материальных благ).

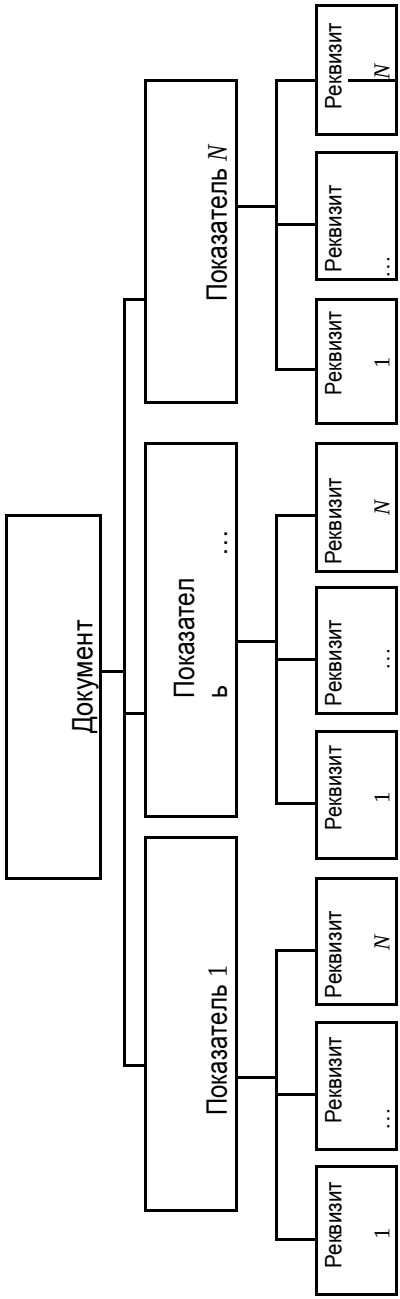
Свойства экономической информации:

− ценность и полезность;

− полнота;

− независимость от форм предоставления.

Для обработки экономической информации характерны сравнительно простые алгоритмы, расчеты (4 арифметических действия плюс проценты), преобладание логических операций (упорядочение, корректировка, выборка) над арифметическими, табличная форма представления исходных и результатных данных.



Для разработки баз данных необходимо иметь представление о классификации информации:

1. По месту в экономическом про-- цессе.

2. По отношению к данной управляющей системе.

В дальнейшем реквизиты становятся атрибутами, из которых строятся отношения в базах данных.

Экономическая информация имеет- следующую структуру (рис. 1.2).

Реквизиты подразделяются на реквизиты-признаки и реквизиты основания. Реквизиты-основания содержат количественную информацию над которой возможны ариф-- метические действия. Реквизиты-при-знаки содержат качественную- ин-формацию.

Примером реквизита может быть:

Прибыль предприятия ОАО «За- ря» за 2 квартал 2014 г. составила 34 млн руб.

В *данном показателе реквизитами являются:*

- *название предприятия;*
- *единица измерения;*
- *объем прибыли;*
- *период.*

Реквизитом-основанием являет- ся объем прибыли.

Рис. 1.2. Структура экономического документа

Вопросы для самоконтроля:

1. Дайте определение информации.
2. В чем заключается сущность статистической теории информации?
3. Перечислите атрибутивные свойства информации
4. Перечислите прагматические свойства информации.
5. Приведите примеры информации:
   а) достоверной и недостоверной;
   б) полной и неполной;
   в) доступной и недоступной для усвоения.
6. В каком случае используются формулы Хартли и Шеннона? Приведите примеры.
7. Опишите структуру экономической информации.

Термины:

Информация
Данные
Знания
Реквизит
Документ
Показатель

Тема 2 ВВЕДЕНИЕ

В БАЗЫ ДАННЫХ

1. Понятие банка данных, Компоненты банка данных.
2. Классификация компонентов банка данных.
3. История развития БД.
4. Пользователи банков данных.

Банк данных — это система специальным образом организованных данных, программных, технических, языковых, организационно-методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.

Структуру банка данных можно предоставить как совокупность следующих компонентов:

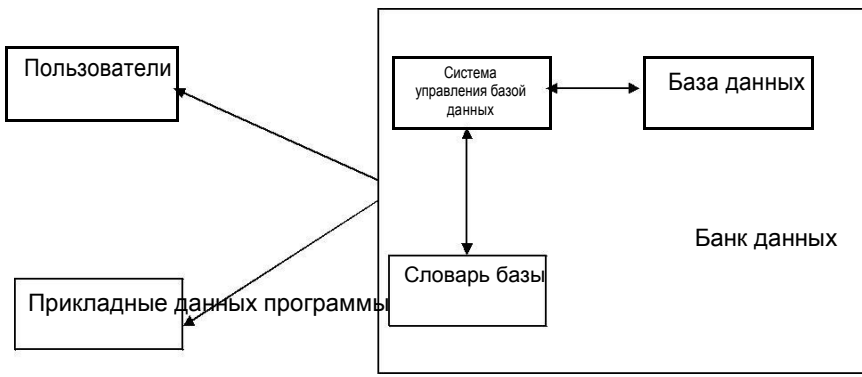


Рис. 2.1. Компоненты банка данных

База данных — это совокупность сведений о конкретных объектах реального мира в какой-либо предметной области или разделе предметной области.

Словарь базы данных — это централизованное хранилище сведений об объектах базы данных, составляющих их элементах данных, взаимосвязях между этими объектами, их источниках, значе-

ниях, использовании и форматах представления. Помимо этого он содержит сведения о группах элементов данных, базы данных и пе-рекрестные ссылки между группами элементов данных и базами данных. Кроме того, в нем фиксируется, сведения об используемых базах данных программами, и имеются сведения о кодах защиты и разграничении доступа. Совокупность базы данных и словаря ба-зы данных составляет информационную базу. В зависимости от категории пользователей метаданные и базы данных представля-ются по-разному. Это зависит от уровня, на котором функциони-рует пользователь. В соответствии с концепцией ANSI SPARC ар-хитектура имеет три ступени: концептуальный уровень, даталоги-ческий и внутренний. На каждом уровне определены свои группы пользователей.

СУБД (система управления базами данных) (DBMS) — совокупность языковых и программных средств, предназначенных для работы с базами данных.

Пользователи баков данных делятся на две группы: конечные пользователи и сотрудники информационных служб.

Каждый из компонентов банка данных имеет свою историю и классификацию. На сегодняшний день-это емкое понятие в узком смысле включает в себя понятие базы данных и системы управления базами данных (см. рис. 2.2).

Они классифицируются следующим образом:

По форме предоставляемой информации: мультимедийные, тек- стовые.

По содержимому: фактографические, документальные. Документальные- предоставляются в виде гипертекста. Гипертекст (англ. *hypertext* — букв. сверхтекст), текст состоящий из ссылок (введен американским социологом и философом Т. Нельсоном в 1965). Для создания гипертекстовых баз данных используют языки разметки.

XML = e**X**tensible **M**arkup **L**anguage (расширяемый язык разметки). Поддерживается W3C (World Wide Web Consortium); первая рекомендация (описание) — 1998 г.

Подмножество SGML (**S**tandard **G**eneralized **M**arkup **L**anguage): упрощенная версия SGML, который был утвержден ISO в качестве стандарта еще в 80-х годах. Этот язык предназначен для создания других языков разметки, он определяет допустимый набор тегов (ссылок), их атрибуты и внутреннюю структуру документа. Кон-

Рис. 2.2. Классификация банков данных

троль за правильностью использования тегов осуществляется при помощи специального набора правил, называемых DTD-описаниями, которые используются программой клиента при разборе документа. Для каждого класса документов определяется свой на-бор правил, описывающих грамматику соответствующего языка разметки. С помощью SGML можно описывать структурированные данные, организовывать информацию, содержащуюся в докумен-тах, представлять эту информацию в некотором стандартизован-ном формате. Но ввиду некоторой своей сложности SGML исполь-зовался в основном для описания синтаксиса других языков (наи-более известным из которых является HTML), и немногие прило-жения работали с SGML-документами напрямую.

Гораздо более простой и удобный, чем SGML, язык HTML позволяет определять оформление элементов документа и имеет некий ограниченный набор инструкций — тегов, при помощи которых осуществляется процесс разметки. Инструкции HTML в первую очередь предназначены для управления процессом вывода содержимого документа на экране программы-клиента и определяют этим самым способ представления документа, но не его структуру. В качестве элемента гипертекстовой базы данных, описы-ваемой HTML, используется текстовый файл, который может легко передаваться по сети с использованием протокола HTTP. Эта особенность, а также то, что HTML является открытым стандартом и огромное количество пользователей имеет возможность применять возможности этого языка для оформления своих документов, безусловно, повлияли на рост популярности HTML и сделали его сегодня главным механизмом представления информации в Интернете.

Однако HTML сегодня уже не удовлетворяет в полной мере требованиям, предъявляемым современными разработчиками к языкам подобного рода. И ему на смену был предложен новый язык гипертекстовой разметки, мощный, гибкий и, одновремен-но с этим, удобный язык XML. В чем же заключаются его досто-инства?

 XML (Extensible Markup Language) — это язык разметки, описывающий целый класс объектов данных, называемых XML-документами. Он используется в качестве средства для описания грамматики других языков и контроля за правильностью составления документов. То есть сам по себе XML не содержит никаких

тегов, предназначенных для разметки, он просто определяет порядок их создания.

□• XML — метаязык.

□• Документ — дерево элементов.

□• Элемент состоит из открывающего и закрывающего тегов.

□• Описание элемента может быть дополнено атрибутами, поме- щаемыми в открывающий тег.

□• Элемент может ссылаться или описывать мультимедийный объект.

□• Отличия от HTML (также базирующегося на SGML):

□• HTML — для описания внешнего представления документа, XML — для описания структуры и семантики документа.

□• Расширяемый: можно задавать свои собственные теги.

□• HTML — язык для публикации в Веб; XML — для более широкого применения.

Долгое время существовали только фактографические базы дан- ных, содержащие фактические сведения (прежде всего статисти-ку), библиографическую информацию (сведения о документах) и полнотекстовые (полные тексты книг и статей из газет, журналов и сборников).

По способу предоставления информации различают коммерческие и бесплатно предоставляемые. Среди наиболее известных производителей и поставщиков коммерческих баз данных в «доинтернетовский» период выделялись LEXIS/NEXIS, Dialog, Silver Platter, EBSCO Information Services, STN Internetional, H.W.Wilson, UMI (ныне ProQuest).

Ленинская библиотека занимается оцифровкой своего библиотечного фонда, а диссертационный зал предоставляет информацию в электронном виде на платной основе.

Рассмотрим наиболее известные из коммерческих баз данных.

LexisNexis (www.lexis-nexis.com)

Одна из крупнейших информационных корпораций мира. Комплекс баз данных LexisNexis включает в общей сложности бо-лее 31 тыс. файлов, многие из которых представляют собой полнотекстовое содержание ведущих периодических изданий.

ProQuest (www.proquest.com)

Линия продуктов ProQuest включает порядка сотни основных баз данных, среди которых отраслевые и тематические полнотекстовые собрания, реферативные и библиографические базы дан-

ных, электронные архивы известнейших газет и журналов. Имеет договоры с более чем 8500 издательств по всему миру, в соответствии с которыми получает компьютерные версии изданий в момент или даже до появления печатных оригиналов.

Questel o Orbit (www.questel.orbit.com)

Компания, существующая более 30 лет, изначально специализируется на предоставлении доступа к данным, связанным с интеллектуальной собственностью и бизнесом. Основу ее информационного потенциала составляют сведения о патентах, товарных знаках, состоянии рынков, компаниях, различного рода финансовая информация, а также полнотекстовые и реферативные базы в области нефтехимии, физики, медицины, механики, электроники, геологии, геофизики, архитектуры и некоторым другим областям знания.

ScienceDirect (www.sciencedirect.com)

Была изначально образована в 1999 г. в качестве базы данных, предоставлявшей доступ к продукции издательской корпорации Elsevier. В дальнейшем ее содержание расширилось и в настоящее время ScienceDirect является одним из крупнейших источников научной и технической информации, а также информации по медицине. Включает порядка 5,5 млн полных текстов научных журналов и книг, базы данных рефератов, фундаментальные энциклопедические и справочные издания.

NetLibrary (www.netLibrary.com)

Проект образован в 1998 г. как служба, ориентированная на работу, прежде всего с реальными библиотеками. Предлагается более 37 тыс. электронных книг (eBooks).

Научная Электронная Библиотека (elibrary.ru)

Проект Российского фонда фундаментальных исследований, целью которого является обеспечение доступа российских научных организаций, включая научные библиотеки, к зарубежной академической периодике преимущественно естественнонаучного профиля.

Публичная библиотека (www.public.ru)

Проект предназначен прежде всего для библиотек, которым предлагается оформить подписку на электронные версии российских центральных и региональных периодических изданий. Публичная библиотека дает возможность бесплатного библиографического поиска — «Открытый доступ» и возможность пользования полными текстами статей — «Профессиональный поиск».

По технологии обработки БД подразделяются на централизованные и распределительные.

Централизованные БД — хранятся в одной вычислительной системе. Если эта вычислительная система является компонен-том сети ЭВМ, возможен распределенный доступ к такой базе.

Распределенная БД — несколько пересекающихся или даже дуб- лирующих друг друга частей, хранимых в различных ЭВМ вычис-лительной сети.

По способу доступа к данным БД распределяются на БД с локальным и распределенным доступом.

Системы централизованных БД с сетевым доступом предполагают различные архитектуры подобных систем:

Файл-сервер (Microsoft Access, Paradox, dBase).

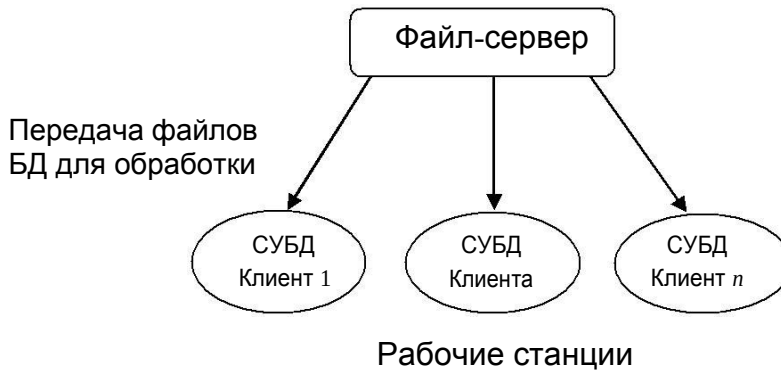


Рис. 2.3. Структура файл-серверной архитектуры

Архитектура систем с сетевым доступом предлагает выделение одной из машин сети в качестве центральной. На такой машине храниться совместно используемая централизованная БД. Файлы БД передаются на рабочие станции, где в основном производится их обработка. Недостатки:

Вся тяжесть ложится на компьютер клиента, который произво-дит вычисления, т. е. если необходимо узнать адрес Магомедова студента ЮФ, все данные о студентах передаются в компьютер по сети, что перегружает сеть, а затем только выбирается необходимая.

БД не защищена от случайных посетителей? и данные могут быть удалены в любой момент, так как доступна каждому клиенту.

Клиент-сервер (InterBase, Oracle, MS SQL Server).

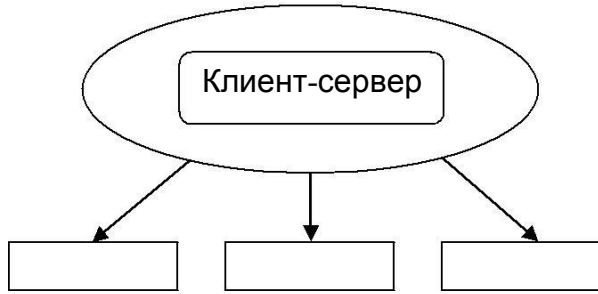


Рис. 2.4. Структура клиент-серверной архитектуры

Преимущества:

- Компьютеры клиента не нуждаются в большой мощности.
- Снижается переполнение сети при пересылке.
- Возрастает безопасность БД.
- Не нарушается целостность, благодаря хранению и обработке данных на одно компьютере.
- В данной системе центральная машина не только хранит данные, но и обрабатывает их. Найденные данные транспортируются клиенту. Эта архитектура может быть двухуровневой и трехуровневой. Двухровневая архитектура (толстый клиент) и трехуров- невая архитектура (тонкий клиент) отличаются нагрузкой на кли- ента засчет добавления промежуточного сервера приложений.

Во взаимодействии клиента и сервера можно выделить следующие компоненты:

- презентационная логика (Presentation Layer — **PL**), предназначенная для работы с данными пользователя;
- бизнес-логика (Business Layer — **BL**), предназначенная для проверки правильности данных, поддержки ссылочной целостности;
- логика доступа к ресурсам (Access Layer — **AL**), предназначенная для хранения данных.

Реализация данных технологий осуществляется засчет СУБД. Компоненты СУБД:

- генератор форм;
- генератор отчетов;

− модуль формирования интерактивных запросов;

− прикладная программа; − клиентское

программное обеспечение БД.

Сегодня СУБД стали настолько доступны, что пользователи стали сами создавать базы данных, что приводит к плачевным последствиям. Не зная методов проектирования допускаются ошибки в структуре базы данных, что приводит к ошибкам в функционировании информационной системы предприятий.

Их классификацию можно представить в следующем виде:

По типу поддерживаемой модели данных: реляционная, объектно-ориентированная, иерархическая, сетевая.

По виду специализации: специализированные СУБД и СУБД общего назначения.

По типу используемых ресурсов: локальная, сетевая.

По типу использования распределенных ресурсов: гомогенная, гетерогенная, мультибазовая.

По степени универсальности различают 2 класса СУБД:

1. Системы общего назначения (универсальные) СУБД- сложные программные комплексы, предназначенные для выполнения всей совокупности функций, связанных с созданием, обновлением и эксплуатацией базы данных информационной системы.

2. Специализированные СУБД создаются в редких случаях при невозможности или нецелесообразности использования СУБД общего назначения.

Рассмотрим историю развития баз данных.

В своем развитии базы данных прошли несколько этапов. Их развитие связано с моделями баз данных, развитием вычислительной техники.

В истории вычислительной техники известны 2 основные области их применения:

1) вычислительная техника применяется для проведения вычислений;
2) использование автоматизированных средств для информационных систем.

Появление второй области вслед за первой произошло из-за несовершенства вычислительной техники. Оперативная (основная) память компьютеров этим свойством обычно не обладает. В пер-вых компьютерах использовались два вида устройств внешней па-мяти — магнитные ленты и барабаны. Емкость магнитных лент

была достаточно велика, но по своей физической природе они обеспечивали последовательный доступ к данным. Магнитные же барабаны (они ближе всего к современным магнитным дискам с фиксированными головками) давали возможность произволь-ного доступа к данным, но имели ограниченный объем хранимой информации.

Эти ограничения не являлись слишком существенными для чисто численных расчетов. Даже если программа должна обработать большой объем информации, при программировании можно продумать расположение этой информации во внешней памяти (например, на последовательной магнитной ленте), обеспечивающее эффективное выполнение этой программы. Однако в информационных системах совокупность взаимосвязанных информационных объектов фактически отражает модель объектов реального мира. А потребность пользователей в информации, адекватно отражающей состояние реальных объектов, требует сравнительно быстрой реакции системы на их запросы. И в этом случае наличие сравнительно медленных устройств хранения данных, к которым относятся магнитные ленты и барабаны, было недостаточным.

Можно предположить, что именно требования нечисловых приложений вызвали появление съемных магнитных дисков с подвижными головками, что явилось революцией в истории вычислительной техники. Эти устройства внешней памяти обладали существенно большей емкостью, чем магнитные барабаны, обеспе-чивали удовлетворительную скорость доступа к данным в режиме произвольной выборки, а возможность смены дискового пакета на устройстве позволяла иметь практически неограниченный ар-хив данных.

С появлением магнитных дисков началась история систем управления данными во внешней памяти. До этого каждая приклад-ная программа, которой требовалось хранить данные во внешней памяти, сама определяла расположение каждой порции данных на магнитной ленте или барабане и выполняла обмены между опера-тивной памятью и устройствами внешней памяти с помощью про-граммно-аппаратных средств низкого уровня (машинных команд или вызовов соответствующих программ операционной системы). Такой режим работы не позволяет или очень затрудняет поддержа-ние на одном внешнем носителе нескольких архивов долговремен-

но хранимой информации. Кроме того, каждой прикладной программе приходилось решать проблемы именования частей данных и структуризации данных во внешней памяти. Важным шагом в развитии информационных систем явился переход к централизованным системам управления данными в системах управления файлами применялся следующий подход. В операции открытия файла среди прочих параметров указывался режим работы (чте-ние или изменение). Если к моменту выполнения этой операции файл был открыт другим пользователем в режиме изменения, то системе сообщалось, либо о невозможности открытия файла, ли-бо он блокировался до момента закрытия. При подобном способе работа нескольких пользователей была ограничена.

Эти недостатки послужили созданию СУБД, а сами хранилища информации назывались базами или банками данных.

Таким образом, в своем развитии в связи с вычислительной техникой они прошли три основных этапа:

Базы данных на больших ЭВМ (1968 г. — введена в эксплуатацию промышленная СУБД IMS фирмы IBM; в 1975 г. — первый стандарт ассоциации по языкам систем обработки данных, который определил понятия в теории систем БД, до сих пор актуальные для сетевой модели; в 1981 г. — премия Тьюринга Э. Ф. Кодду за создание реляционной модели данных. На этом этапе значительная роль отводится администрированию данных).

Эпоха персональных компьютеров (характеризуется появлением настольных СУБД, вытеснивших администрирование MS Access97).

Распределенные БД (с появлением локальных сетей и стандартов языков описания и манипулирования данными; объектно-ориентированные БД).

Традиционно развитие банков в связи с обработкой данных шло по следующим этапам:

1. Файловые системы.
2. Системы управления базами данных.

Первое поколение СУБД поддерживали: иерархические модели, (IMS система), сетевые модели (CODASYL). На этом этапе стан-дартизируются три языка (группой DDTG) DDLсхемы, DDL под-схемы, DML.

Следующее поколение СУБД основывается на реляционных и объектно-ориентированных моделях.

Далее находя широкое применение гипертекстовые и мультимедийные базы данных.

3. Распределенные базы данных.

Первые базы данных связаны с библиотечными каталогами: правительственными, деловыми и медицинскими записями. Существует- очень долгая история хранения, индексирования и поиска информации.

Рассмотрим развитие моделей данных.

1960-е гг.: компьютеры стать экономически эффективным для частных компаний наряду с увеличением хранения данных в компьютерах. Две основные модели данных были разработаны: модель сети (CODASYL) и иерархические (IMS). Доступ к базе данных через низкоуровневые операции с указателями связи записей. Подробнее хранения зависит от типа данных, которые будут сохране-ны. Таким образом добавить дополнительные поля в базе данных необходимо переписать основные доступа / изменения схемы-. Особое внимание было уделено записей, подлежащих обработке, не общей структуры системы. Пользователю необходимо знать физические структуры базы данных для того, чтобы запрос о предоставлении информации. Одним из основных коммерческого успе-ха SABRE системы от IBM и American Airlines.

1970–1972 гг.: Е. Ф. Кодд предлагает основы теории реляционной модели баз данных, этот документ систематизирует и формализует реляционную теорию, и позволил упростить работу пользователей с базами данных. Его труд позволил разделить понятия логическая и физическая модель. Эта система является стандартом до сих пор.

Появление реляционной модели принесло большой прорыв в теории баз данных. Благодаря организации данных в виде таблиц сейчас создавать базы данных могут и прикладные программисты.

Рассмотрим поэтапное развитие реляционных баз данных:

Этап (1) первые дни.

Теоретические основы реляционных баз данных были предложены Dr. Edgar (Тед) Кодд в IBM Research, Алмаден, Калифорния, и им же опубликована работа под названием «Реляционная модель данных для больших совместных банков данных».

Проект Ingres был впервые создан в рамках исследовательского проекта в Университете Калифорнии, Беркли, начиная с ранних 1970-х и заканчивается в начале 1980-х гг. Исходный код, как и от

других проектов, в Беркли был доступен за минимальную плату под версией лицензии BSD. С середины 1980-х гг. Ingres породил целый ряд коммерческих приложений баз данных, включая Sybase, Microsoft SQL Server, SQL NonStop и ряд других. Postgres (Post B GRES), проект, начавшийся в середине 1980-х, позднее преврати-лась в PostgreSQL. На базе этого проекта был развит и Oracle как первый коммерческий продукт на основе этой работы.

В отличие от System R, исходный код Ingres был доступен (на кассете) за умеренную плату. К 1980 г. около 1000 копий был распространен главным образом в университетах. Многие студенты из Калифорнийского университета в Беркли и других университе-тах, которые использовали источник код Ingres работал в различ-ных коммерческих систем баз данных. В отличие от System R, ис-ходный код Ingres был доступен (на кассете) за умеренную плату.
К 1980 г. около 1000 копий был распространен главным образом в университетах. Многие студенты из Калифорнийского универси-тета в Беркли и других университетах, которые код Ingres для ра-боты в различных коммерческих системах баз данных.

В 1976 г. Петр Чен опубликовал очень важный документ по моделированию данных, озаглавленный «Модель сущность-связь». Это позволило сделать еще один шаг к единому представлению данных в ACM (Association for Computing Machinery) Transactions (представляет собой научный журнал, который публикует высокое качество документов о разработке и оценке компьютерного программного обеспечения) по системам баз данных.

Этап (2) Middle Years.

На арену выходят IBM, Informix и Sybase. Наконец Microsoft купила технологию Sybase, чтобы вступить в бой.

Этап (3) будущее.

Век Интернета и информатизации базы данных используются на глобальном уровне, информация переместилась в Интернет. Наиболее вероятный сценарий видит выживания всего трех продавцов (что было подтверждено в приобретении IBM Informix за 1 млрд долл., Unica, которая занимается разработкой программного обеспечения для автоматизации деятельности маркетинга, за 480 тыс. долл.): Oracle, IBM и Microsoft.

Как любой программно-организационно-технический комплекс, банк данных существует во времени и в пространстве. Он имеет определенные стадии своего развития:

□• Проектирование.
□• Реализация.
□• Эксплуатация.
□• Модернизация и развитие.
□• Полная реорганизация.

На каждом этапе своего существования с банком данных связаны разные категории пользователей.

Пользователи, которые работают с банком данных, делятся на несколько категорий:

Пользователи

Конечные пользователи. Это могут быть случайные пользователи, обращающиеся к БД время от времени за получением некото-рой информации, а могут быть регулярные пользователи.

В качестве случайных пользователей могут рассматриваться, например, возможные клиенты фирмы, просматривающие каталог продукции или услуг с обобщенным или подробным описанием того и другого.

Регулярными пользователями могут быть сотрудники, работающие со специально разработанными для них программами, которые обеспечивают автоматизацию их деятельности при выполнении своих должностных обязанностей. От конечных пользователей не должно требоваться каких-либо специальных знаний в области вычислительной техники и языковых средств.

Администраторы банка данных. Это группа пользователей, которая на начальной стадии *разработки* банка данных отвечает за его оптимальную организацию с точки зрения одновременной работы множества конечных пользователей, на стадии *эксплуата-ции* отвечает за корректность работы данного банка информации в многопользовательском режиме. На стадии *развития* и *реорганизации* эта группа пользователей отвечает за возможность коррект-ной реорганизации банка без изменения или прекращения его те-кущей эксплуатации. Администратор БД отвечает за целостность информационных ресурсов компании. На нем лежит ответственность по созданию, обновлению и сохранности связанных между собой резервных копий файлов, исходя из задач предприятия. Этот человек должен в мельчайших подробностях знать существующие механизмы восстановления программного обеспечения БД.

Разработчики и администраторы приложений. Это группа пользователей, которая функционирует во время проектирова-

ния, создания и реорганизации банка данных. Администраторы приложений координируют работу разработчиков при разработке конкретного приложения или группы приложений, объединенных в функциональную подсистему. Разработчики конкретных приложений работают с той частью информации из базы данных, которая требуется для конкретного приложения

Наиболее сложные обязанности возложены на группу администратора БД. Рассмотрим их более подробно.

В составе группы администратора БнД должны быть:

- системные аналитики;
- проектировщики структур данных и внешнего по отношению к банку данных информационного обеспечения;
- проектировщики технологических процессов обработки данных;
- системные и прикладные программисты; - операторы и специалисты по техническому обслуживанию.

Если речь идет о коммерческом банке данных, то важную роль здесь играют специалисты по маркетингу.

Основные функции группы администратора БД:

Анализ предметной области: описание предметной области, выявление ограничений целостности, определение статуса (доступности, секретности) информации, определение потребностей пользователей, определение соответствия «данные-пользователь», определение объемно-временных характеристик обработки данных.

Проектирование структуры БД: определение состава и структуры файлов БД и связей между ними, выбор методов упорядочения данных и методов доступа к информации, описание БД на языке описания данных (ЯОД).

Задание ограничений целостности при описании структуры БД и процедур обработки БД:

- задание декларативных ограничений целостности, присущих предметной области;
- определение динамических ограничений целостности, присущих предметной области в процессе изменения информации, хранящейся в БД;
- определение ограничений целостности, вызванных структурой БД;
- разработка процедур обеспечения целостности БД при вводе и корректировке данных;

- определение ограничений целостности при параллельной работе пользователей в многопользовательском режиме.

Первоначальная загрузка и ведение БД:

- разработка технологии первоначальной загрузки БД, которая будет отличаться от процедуры модификации и дополнения данными при штатном использовании базы данных;
- разработка технологии проверки введенных данных реальному состоянию предметной области. База данных моделирует реальные объекты некоторой предметной области и взаимосвязи между ними, и на момент начала штатной эксплуатации эта модель должна полностью соответствовать состоянию объектов предметной области на данный момент времени;
- в соответствии с разработанной технологией первоначальной загрузки может понадобиться проектирование системы первоначального ввода данных.

Защита данных:

- определение системы паролей, принципов регистрации пользователей, создание групп пользователей, обладающих одинаковыми правами доступа к данным;
- разработка принципов защиты конкретных данных и объектов проектирования; разработка специализированных методов кодирования информации при ее циркуляции в локальной и глобальной информационных сетях;
- разработка средств фиксации доступа к данным и попыток нарушения системы защиты;
- тестирование системы защиты; - исследование случаев нарушения системы защиты и развитие динамических методов защиты информации в БД.

Обеспечение восстановления БД:

- разработка организационных средств архивирования и принципов восстановления БД;
- разработка дополнительных программных средств и технологических процессов восстановления БД после сбоев.

Анализ обращений пользователей БД: сбор статистики по характеру запросов, по времени их выполнения, по требуемым выходным документам.

Анализ эффективности функционирования БнД:

- анализ показателей функционирования БнД;

− планирование реструктуризации (изменение структуры) БД и реорганизации БнД.

Работа с конечными пользователями:

− сбор информации об изменении предметной области;
− сбор информации об оценке работы БнД;
− обучение пользователей, консультирование пользователей;
− разработка необходимой методической и учебной документации по работе конечных пользователей.

Подготовка и поддержание системных средств:

− анализ существующих на рынке программных средств и анализ возможности и необходимости их использования в рамках БнД;
− разработка требуемых организационных и программно-технических мероприятий по развитию БнД;
− проверка работоспособности закупаемых программных средств перед подключением их к БнД;
− курирование подключения новых программных средств к БнД.

Организационно-методическая работа по проектированию БнД:

− выбор или создание методики проектирования БД; − определение целей и направления развития системы в целом;
− планирование этапов развития БнД;
− разработка общих словарей-справочников проекта БнД и концептуальной модели;
− стыковка внешних моделей разрабатываемых приложений;
− курирование подключения нового приложения к действующему БнД; − обеспечение возможности комплексной отладки множества
приложений, взаимодействующих с одной БД.

Классификация АБД (по данным сайта pinnaclepublishing.com, журнал «Oracle Professional», October 2001): существует несколько видов администраторов БД, а их обязанности вполне могут отличаться от компании к компании. Вот характеристики некоторых типов АБД и занимаемых ими положений.

Оперативные (operational) АБД:

1) манипулируют дисковым пространством;
2) наблюдают за текущей производительностью системы;
3) реагируют на возникающие неисправности БД;
4) обновляют системное ПО и ПО базы данных;

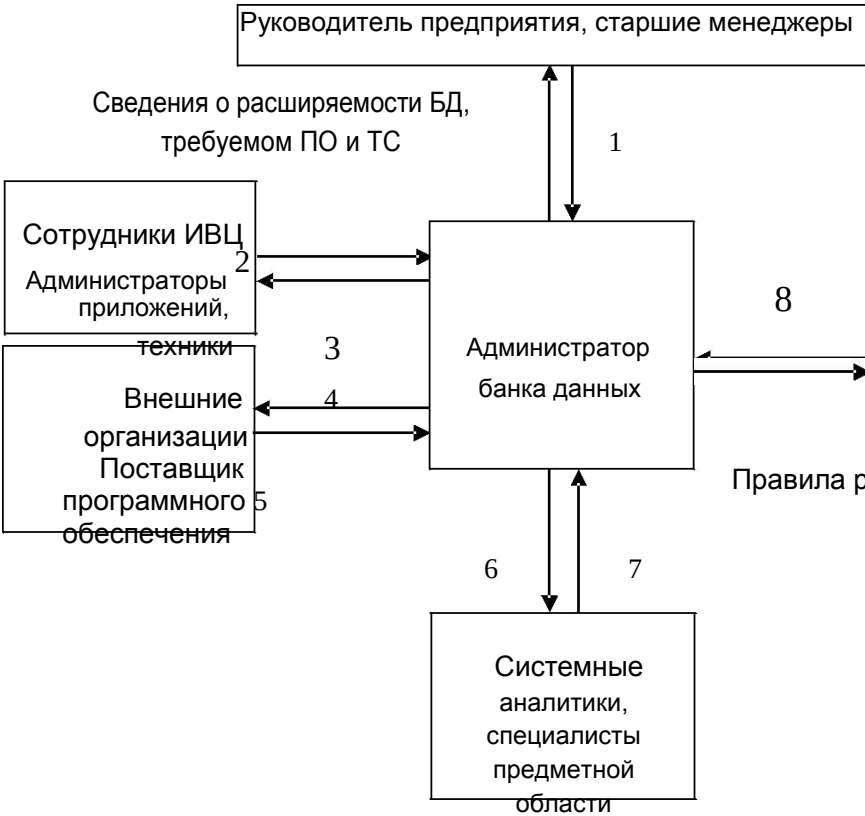


Рис. 2.5. Организация работы Администратора банка д

5) контролируют структурные изменения БД;
6) запускают процедуры резервного копирования данных;
7) выполняют восстановление данных;
8) создают и управляют тестовыми конфигурациями БД.

Тактические (tactical) АБД:
1) реализуют схемы размещения информации;
2) утверждают процедуры резервного копирования и восстановления данных;
3) разрабатывают и внедряют структурные элементы БД: таблицы, столбцы, размеры объектов, индексацию и т. п.; сценарии (scripts) изменения схемы БД; конфигурационные параметры БД;
4) утверждают план действий в случае аварийной ситуации.

Стратегические (strategic) АБД:
1) выбирают поставщика БД;
2) устанавливают корпоративные стандарты данных;
3) внедряют методы обмена данных в рамках предприятия;
4) определяют корпоративную стратегию резервирования и вос- становления данных;
5) устанавливают корпоративный подход к ликвидации послед- ствий аварии и обеспечению доступности данных.

Старшие (senior) АБД:
1) досконально знают свой персонал;
2) пользуются высоким спросом;
3) могут написать скрипт, который освободит их из запертого сундука, брошенного в океан, и чрезвычайно гордятся своими произведениями;
4) тратят уйму времени на подготовку младших АБД;
5) очень ценятся руководством и получают бешеные деньги.

Младшие (junior) АБД:
1) мечтают стать старшим АБД;
2) не слишком сильны в написании скриптов;
3) имеют большую склонность к использованию средств управ-ления БД;
4) тоже неплохо получают.

Прикладные (application) АБД:
1) в курсе информационных нужд компании;
2) помогают в разработке прикладных задач;
3) отвечают за разработку схемы и ее изменения;

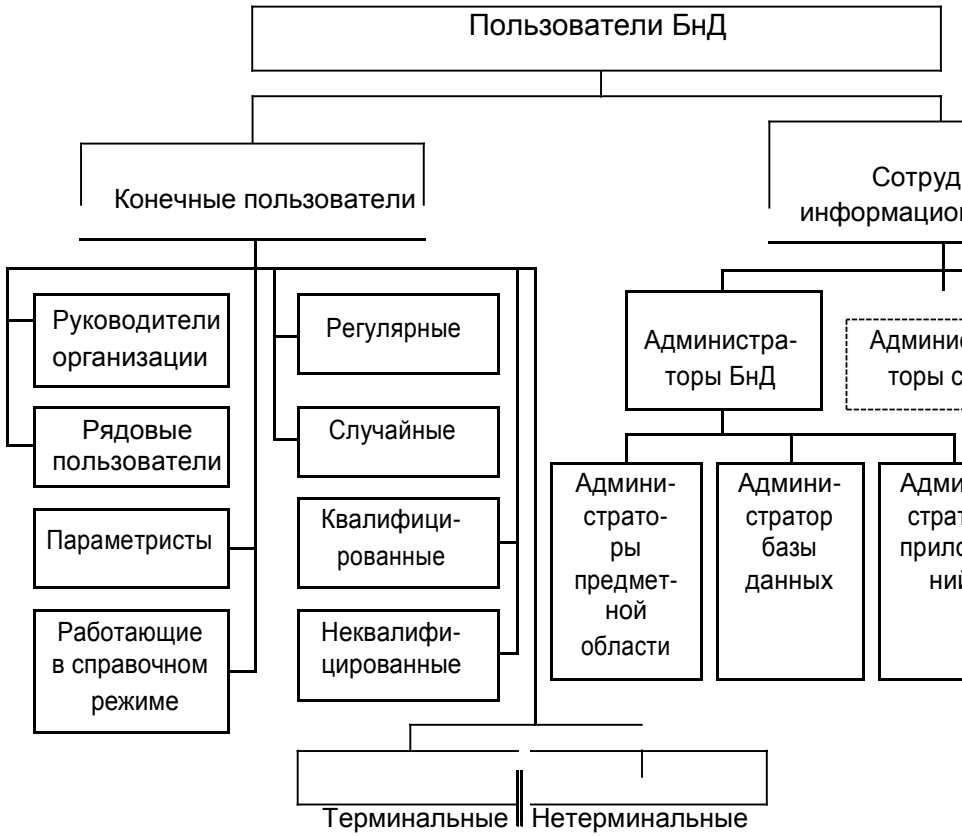


Рис. 2.6. Классификация пользователей банка данных

4) вместе с системным АБД обеспечивают должный уровень резервирования/ восстановления данных;
5) занимаются построением тестовых БД.

Системные (system) АБД:

1) отвечают за все необходимое для резервирования и восстановления данных;
2) контролируют производительность системы в целом;
3) осуществляют поиск и устранение неисправностей;
4) в курсе нынешних и будущих потребностей БД в плане емкости;
5) в курсе текущего состояния и нужд БД.

Наемные (contract) АБД:

1) приглашаются под конкретную задачу или в качестве консультантов;
2) передают персоналу необходимые знания;
3) фиксируют свои действия;
4) должны прекрасно разбираться в соответствующей области;
5) хороши в качестве временного персонала, для оценки проекта или системы.

Администраторы-руководители:

1) проводят еженедельные совещания;
2) определяют перечень первоочередных задач;
3) устанавливают и оглашают официальный курс и стратегию;
4) утверждают и корректируют должностные инструкции и спи- сок обязанностей;
5) следят за наличием соответствующей документации.

На рис. 2.6 представлена классификация пользователей, предложенная в источнике [12].

Вопросы для самоконтроля:

1. Что, на ваш взгляд, явилось основной причиной появления баз данных?
2. Что входит в структуру банка данных?
3. По каким признакам классифицируются базы данных?
4. По каким признакам классифицируются системы управления базами данных.
5. Перечислите основные модели баз данных в порядке их становления.
6. Как классифицируются пользователи банка данных.

Термины:

Банк данных
Базы данных
Словарь базы данных
Администратор базы данных
СУБД Фактографические базы
данных,
документальные базы
данных Тонкий клиент
Толстый клиент

Тема 3

ПРИНЦИПЫ ПОСТРОЕНИЯ БД

1. Жизненный цикл баз данных.
2. Архитектура БД.
3. Трехуровневая модель СУБД.
4. Модели данных, их классификация.

В понятии жизненного цикла используется выражение «снятия с эксплуатации». Вот как оно трактуется в действующем ГОСТе: retirement прекращение активной поддержки действующей систе-мы со стороны эксплуатирующей или сопровождающей организа-ции, частичная или полная замена ее новой системой или ввод в действие модернизированной системы (ГОСТ 15288-2005).

С 1960 г. и по сей день продолжается депрессия программного обеспечения. Если рассмотреть литературу по информатике, то в каждой книге можно увидеть статьи о развитии компьютерных технологий, а программное обеспечение упоминается в небольших статьях. В чем же дело, почему внедрение и использование но-вого программного обеспечения приводит к высоким затратам материальным и временным? В результате анализа было выяснено, что отсутствуют единые требования, методология разработки.

Для решения данных проблем был предложен структурный подход по разработке программного обеспечения, названный жизненным циклом информационных систем.

Этот жизненный цикл неотделим от БД, как и сама ИС.

Этапы	Описание
Планирование разработки БД	Планирование наиболее эффективного способа реализации этапов жизненного цикла. Четко определяются технические требования предъявляемые к БД и соответствие их выделенным материальным ресурсам
Определение требований к системе	Определение диапазона действий и границ приложения БД. Состава его пользователей из всех возможных областей применения и взаимодействие с остальными частями ИС

Продолжение таблицы

Этапы	Описание
Сбор и анализ требований пользователей	Процесс анализа и сбора информации о той части организации, работа которой будет поддерживаться с помощью создаваемого приложения. Для решения данной задачи требуется следующая информация: 1. Описание применяемых или вырабатываемых данных. 2. Подробные сведения о способах применения или выработки данных. 3. Все дополнительные требования к создаваемому приложению БД. Данная информация более четко формулируется с использованием технологии структурного анализа и проектирования SAD, диаграммы потоков данных DFD
Проектирование БД	Полный цикл разработки включает концептуальное, логическое и физическое проектирование
Выбор целевой СУБД	Необязательный этап. Этапы процедуры выбора СУБД: 1. Определение предметной области проводимого исследования (в соответствии с материальными возможностями и совместимостью с уже используемыми программными продуктами). 2. Сокращение списка выбора до 2–3 продуктов. 3. Оценка продуктов (по определенным параметрам оценивается рейтинг по 10-балльной шкале и вес по степени важности не более 1; можно проверить непосредственно при демонстрации продукта). 4. Проведение обоснованного выбора и подготовка отчета
Разработка приложений	Проектирование пользовательского интерфейса и прикладных программ, предназначенных для работы с БД. Включает в себя: Проектирование транзакций. Проектирование пользовательского интерфейса. Транзакция — одно действие или последовательность действий. Выполняемых одним и тем же пользователем или прикладной программой. Которые получают доступ к БД или изменяют ее содержимое (регистрация предлагаемого для сдачи в аренду объекта недвижимости). Она может состоять из нескольких операций. Различают транзакции: извлечения, обновления, смешанные. Цель проектирования транзакций заключается в определении и документировании характеристик всех необходимых

Окончание таблицы

Этапы	Описание
	транзакций, которые должны будут выполняться в разрабатываемой БД. При проектировании пользовательского интерфейса необходимо соблюдать следующие правила: ясные и четкие названия полей. Согласованность цветов, удобное перемещение курсора. Средства исправления отдельных ошибочных символов и целых полей и т. д.
Создание прототипов	Создание рабочей модели приложения БД: 1. Прототип — рабочая модель, обладающая лишь частью функциональных возможностей БД. 2. Создание прототипа для определения требований (после апробации прототип отбрасывается). 3. Создания прототипа путем последовательной доработки (прототип после пробного использования пользователем дорабатывается и постепенно превращается в приложение)
Реализация	Физическая реализация БД и разработанных приложений
Преобразование и загрузка данных	Перенос любых существующих данных в новую БД и модификация всех существующих приложений с целью организации совместной работы с новой БД
Тестирование	Процесс выполнения прикладных программ с целью поиска ошибок
Эксплуатация и сопровождение	Наблюдение за системой и поддержка ее нормального функционирования по окончании развертывания

Архитектура — разновидность (обобщение) структуры, в которой какой-либо элемент может быть заменен на другой элемент, характеристики входов и выходов которого идентичны первому элементу. Рассмотрим работу DBMS с помощью анализа архитектуры.

Основная цель системы управления базами данных заключается в том, чтобы предложить пользователю абстрактное представление данных, скрыв конкретные особенности хранения и управления ими. Следовательно, отправной точкой при проектировании БД должно быть общее описание информационных потребностей пользователей, которые должны найти свое отражение в создаваемой базе данных.

Более того, поскольку база данных является общим ресурсом, то каждому пользователю может потребоваться свое, отличное от других представление о характеристиках информации, сохраняемой в базе данных. Для удовлетворения этих потребностей архитектура большинства современных СУБД в той или иной степени строится на базе так называемой архитектуры ANSI-SPARC. Как известно из истории на первоначальном этапе существовала двухуровневая архитектура, состоящая из схем и подсхем. Появление реляционной модели обусловило существование трехуровневой архитектуры, состоящей из внешней схемы (пользовательской), логической и внутренней схемы (схемы хранения) предложенной в 1975 г. Комитетом планирования стандартов и норм SPARC (Stan- dards Planning and Requirements Committee) Национального Ин- ститута- Стандартизации США (American National Standard Insti- tute — ANSI).

Во внешнюю схему включают различные пользовательские про- цессы. Термин «пользовательский процесс» (user process) вводится для того, чтобы разрешить доступ к системе базы данных с помо-щью любого вида программного обеспечения от имени пользова-теля. Далее мы более подробно исследуем эти концепции для по-нимания приницпов их работы.

Схема пользователей (userschema) — это определение логических свойств данных, которые пользовательский процесс должен отсылать тому пользователю, которому необходимы эти данные. Как и логическая схема, схема пользователей представляет собой определения свойств типов данных и они определяются в форме, которая может быть обработана компьютером. Так как, обычно, база данных необходима нескольким пользователям, то существует много схем пользователей, каждая из которых основана на одной и той же единственной логической схеме. Только через пользовательскую схему пользователь может получить доступ к базе данных. Пользовательский процесс может использовать только одну схему пользователя. Обычно несколько пользовательских процессов могут потребовать одну и ту же схему пользователей — просмотр данных — и поэтому могут совместно использовать одну и ту же схему пользователей.

Логическая схема (logical schema) — это одиночное, центральное описание логических свойств данных в конкретной базе дан-

ных. Логическая схема больше описывает, какими являются данные, чем то, как их можно хранить и как получить к ним доступ. Для любой данной системы баз данных имеется одна логическая схема. Она является описанием свойств типов данных, которые затем определяют свойства экземпляров данных в базе данных. Она описывается формальным языком, который может быть обработан компьютером. Этот язык иногда называют языком описания данных логической схемы (logical schema data definition language или logical schema DDL).

Схема хранения — это определение того, как хранить и полу-чать доступ к данным, определенным в логической схеме. Опре- деление происходит с помощью языка, который иногда называют языком описания данных схемы хранения (storage schema data definition language (или storage schema DDL) Здесь используются понятия «хранение файлов», «структуры файлов» и «методы до-ступа». Данные в каждой таблице, описанные в логической схеме, необходимо как-то физически хранить. Это осуществляется путем размещения данных каждой таблицы в запоминающем файле (sto- red file), который управляется операционной системой компьюте-ра. Каждый хранимый файл хранится в соответствии со структу-рой (организацией) файла (file organization) (упорядоченной, ин-дексной или хэшированной), поддерживающей метод доступа (ac- cess method), с помощью которого хранимые записи (stored re- cords), составляющие данных файл, могут быть сохранены и из-влечены. Таким образом, схема хранения применяет методы из технологии массовой памяти (запоминающего устройства боль-шой емкости), используемой в вычислительных системах.

Таким образом, в архитектуре базы данных ANSI-SPARC используются три уровня абстракции: внешний, концептуальный и внутренний. Внешний уровень состоит из пользовательских представлений базы данных. Концептуальный уровень является обобщенным представлением о структуре базы данных. Он определяет информационное содержимое базы данных, не зависящее от спо-соба хранения информации. Внутренний уровень является компьютерным представлением базы данных. Он определяет, как представлены данные, в какой последовательности располагаются записи, какие созданы индексы и указатели, используется ли схема хеширования и какая именно.

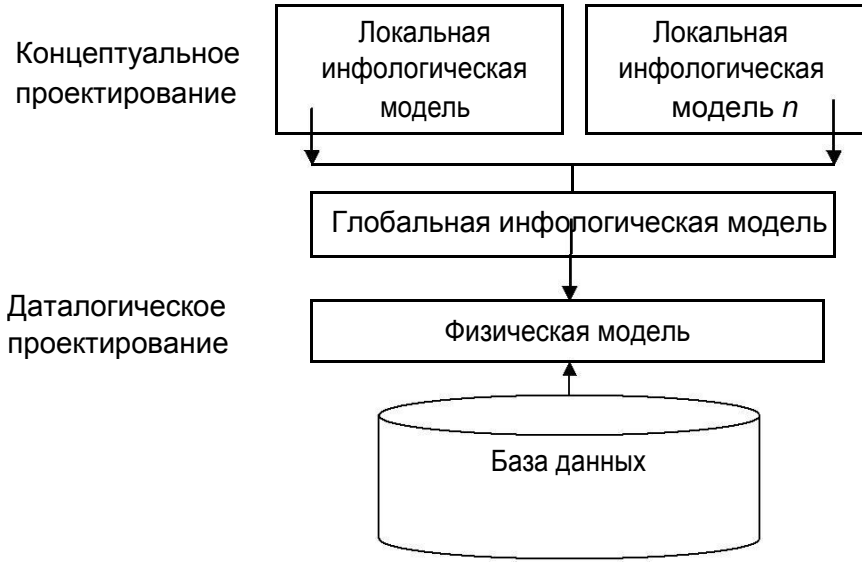


Рис. 3.1. Трехуровневая модель СУБД

Модель данных — интегрированный набор понятий для описа-ния и обработки данных, связей между ними и ограничений накладываемых на данные в некоторой организации.

Архитектура DBMS напрямую связана с классификацией моделей данных. Представления пользователей строятся на основании локальных инфологических моделей. Глобальная инфологическая модель позволяет представить всю базу данных.

Эта архитектура позволяет обеспечить логическую (между 1 и 2) и физическую между (2 и 3) независимость при работе с данными. Логическая несовместимость предполагает возможность переноса хранимой информации с одних носителей на другие с сохранением работоспособности.

Выделение концептуального уровня позволило разработать аппарат централизованного управления базой данных (см. рис. 3.2).

Групповые отношения удобно изображать с помощью диаграм-мы Бахмана (названа по имени одного из разработчиков сетевой модели данных). Диаграмма Бахмана представляет собой ориен-тированный граф, в котором вершины соответствуют группам (типам записей), а дуги — иерархическим групповым отношени-

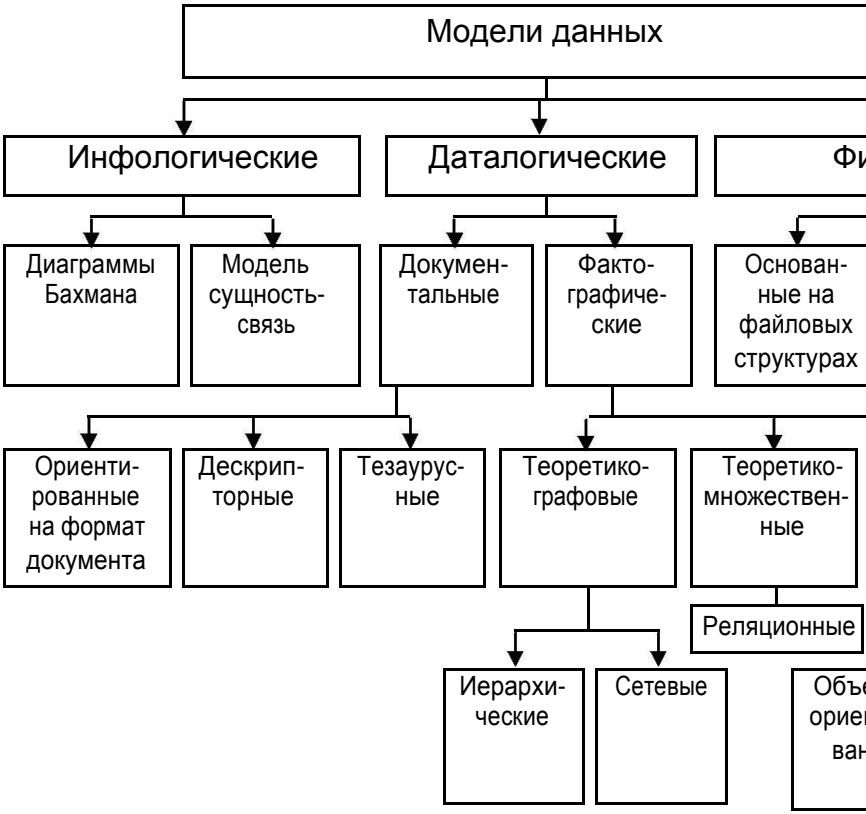


Рис. 3.2. Классификация моделей данных

ям. Модель сущность-связь впервые предложена П. Ченом в 1976 г. В диаграммах Бахмана объекты (сущности) представляются вершинами некоторого математического графа, а связи — дугами графа. Виды и свойства связей-отношений объектов отображаются направленностью, специальным оформлением дуг и расположением вершин графа.

С его именем связана и концепция сетевой модели данных, так как он оказал определяющее влияние на создание проекта DBTG CODASYL (1971). Сетевая модель данных является моделью объектов-связей, где допускаются только бинарные связи типа «многие-к-одному», что позволяет использовать для представления данных простую модель ориентированных графов. В некоторых определениях сетевой модели допускаются связи типа «многие-ко-многим», но требование бинарности связи остается в силе.

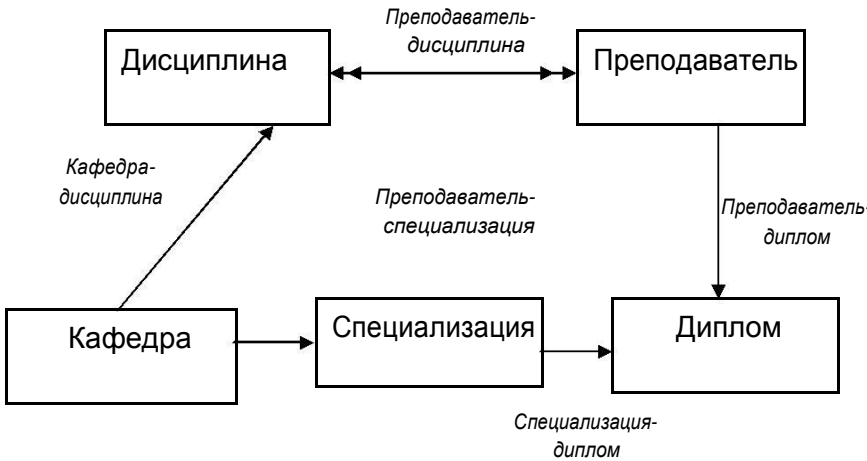


Рис. 3.3. Диаграмма Бахмана

На приведенном рисунке однонаправленность дуг означает структурность связи «владелец-подчиненный», двунаправленность дуг означает одноуровневые связи, двойные стрелки означают множественность отношения «один-ко-многим», двунаправлен-ность двойных стрелок означает одноуровневые отношения «мно- гие-ко-многим».

Описания формализованных схем (моделей) предметных областей информационных систем является их статичность, не позво-

ляющая наглядно и непосредственно отображать процессы, в которые вовлечены сущности и которым подвержены отношения (связи). Отчасти подобные проблемы преодолеваются введением дополнительных сущностей, выражающих собственно процессы и ситуации — событие, действие, момент времени. Аналогичным образом в некоторых случаях вводятся пространственные сущности для адекватного представления сущностей и отношений предметной области — маршрут, место, населенный пункт, здание, элемент здания, зона и т. д.

Если инфологические модели ориентированы на объекты и их свойства, то даталогические модели уже поддерживаются конкретной СУБД. Физические модели описывают структуры и принципы их хранения во внешней памяти, а также доступа к ним в зависимости от используемых аппаратных средств и программного обеспечения низкого уровня.

Документальные модели данных соответствуют представлению о слабоструктурированной информации, ориентированной в ос-новном на свободные форматы документов, текстов на естествен-ном языке. Фактографические — соответствуют представлению о четко структурированной информации, формализованных данных.

Модели, ориентированные на формат документа, основаны на языках разметки документов и связаны прежде всего со стандартным общим языком разметки — SGML.

Тезаурусные модели основаны на принципе организации словарей, содержат определенные языковые конструкции и принципы их взаимодействия в заданной грамматике. Эти модели эффективно используются в системах-переводчиках, особенно многоязыковых переводчиках. Принцип хранения информации в этих системах и подчиняется тезаурусным моделям.

Дескрипторные модели — самые простые из документальных моделей, они широко использовались на ранних стадиях использования документальных баз данных. В этих моделях каждому документу соответствовал дескриптор — описатель. Этот дескрип-тор имел жесткую структуру и описывал документ в соответствии с теми характеристиками, которые требуются для работы с доку-ментами в разрабатываемой документальной БД. Например, для БД, содержащей описание патентов, дескриптор содержал назва-ние области, к которой относился патент, номер патента, дату вы-дачи патента и еще ряд ключевых параметров, которые заполня-

лись для каждого патента. Обработка информации в таких базах данных велась исключительно по дескрипторам, т. е. по тем параметрам, которые характеризовали патент, а не по самому тексту патента.

Теоретико-графовые модели данных. Эти модели отражают совокупность объектов реального мира в виде графа взаимосвязан-ных информационных объектов. Язык графов оказывается удоб-ным для описания многих физических, технических, экономиче-ских, биологических, социальных и других систем.

Иерархическая модель данных. Связи между данными описываютсяспомощьюупорядоченногографа(илидерева).Иерархическая БД представляет собой упорядоченную совокупность экземпля-ров данных типа «дерево», содержащих экземпляры типа «запись».

Сетевая модель данных. Сетевая модель данных позволяет отображать разнообразные взаимосвязи элементов данных в виде произвольного графа, обобщая тем самым иерархическую модель данных. Сетевая БД состоит из набора записей и набора соответствующих связей. На формирование связи особых ограничений не накладывается.

Теоретико-множественные модели данных *Реляционная модель данных:* наиболее распространенная модель. Основана на понятии «отношение» (привычные связанные таблицы).

По характеру организации хранения данных и обращения к ним различают *локальные* (персональные), *общие* (интегрирован-ные) и распределенные базы данных.

Документальные модели лежат в основе документальных баз данных. Документальная база данных: база данных, в которой каждая запись отражает конкретный документ, содержит его библиографическое описание и, возможно, иную информацию о нем (ГОСТ 7.73-96).

В развитии программного обеспечения СУБД в 70–80-е годы превалировало направление, связанное с фактографическими информационными системами, т. е. с системами, ориентированными на работу со структурированными данными. Были разработаны основы и модели организации фактографических данных, отрабо-таны программно-технические решения по накоплению и физиче-скому хранению таких данных, реализованы специальные языки запросов к базам данных и решен целый ряд других задач по эф-

фективному управлению большими объемами структурированной информации. В результате основу информационного обеспечения деятельности предприятий и организаций к началу 90-х годов составили фактографические информационные системы, вобравшие в себя в совокупности колоссальный объем структурированных данных.

Вместе с тем создание и эксплуатация фактографических информационных систем требует либо изначально структурирован-ных данных, таких, например, как отчеты датчиков в АСУ ТП, финансовые массивы бухгалтерских АИС и т. д., либо предваритель-ной структуризации данных, как, например, в информационной системе кадрового подразделения, где все данные по сотрудникам структуризируются по ряду формализованных позиций. При этом зачастую структуризация данных требует больших накладных, в том числе и организационных расходов, что, в конечном счете, приводит к материальным издержкам информатизации.

Кроме того, входные информационные потоки в целом ряде организационно-технологических и управленческих сфер представлены неструктурированными данными в виде служебных документов и иных текстовых источников. Извлечение из текстов дан-ных по формализованным позициям для ввода в фактографиче-ские системы может приводить к ошибкам и потере части инфор-мации, которая в исходных источниках имеется, но в силу отсут-ствия в схеме базы данных адекватных элементов не может быть отражена в банке данных фактографических АИС.

В результате, несмотря на интенсивное развитие и распространение фактографических информационных систем, огромная часть неструктурированных данных, необходимых для информационного обеспечения деятельности различных предприятий и организаций, остается в неавтоматизированном или слабо автоматизированном виде. К таким данным относятся огромные массивы различной периодики, нормативно-правовая база, массивы служебных документов делопроизводства и документооборота.

Потребности в системах, ориентированных на накопление и эффективную обработку неструктурированной или слабоструктурированной информации привели к возникновению еще в 70-х годах отдельной ветви программного обеспечения систем управления базами данных, на основе которых создаются документальные информационные системы.

Однако теоретические исследования вопросов автоматизированного информационного поиска документов, начавшись еще в 50–60-х годах, к сожалению, не получили такой строгой, полной и в то же время технически реализуемой модели представления и обработки данных, как реляционная модель в фактографических системах. Не получили также стандартизации (как язык SQL) и многочисленные попытки создания универсальных так называемых информационно-поисковых языков, предназначенных для формализованного описания смыслового содержания документов и запросов по ним. В итоге, несмотря на то что первые системы автоматизированного информационного поиска документов появились еще в 60-х годах, развитые коммерческие информационно-поисковые системы, ориентированные на накопление и обработку текстовых документов, получили распространение лишь в конце 80-х — начале 90-х годов.

Для более полного отражения реального мира появились семантические модели, к которым относятся модели на объектах объектно-реляционные и объектно-ориентированные. Их принципы лежат во многих разрабатываемых системах.

Вопросы для самоконтроля:

1. Перечислите и охарактеризуйте этапы жизненного цикла БД.
2. Охарактеризуйте трехуровневую архитектур СУБД.
3. По каким признакам классифицируются модели данных?

Термины:

Модель данных
Тезаурус
Дескриптор
Сетевая
Реляционная
Иерархическая модель данных
Документальная база данных
Фактографическая база данных
ANSI
Жизненный цикл

Тема 4

ДАТАЛОГИЧЕСКИЕ МОДЕЛИ ДАННЫХ

1. Анализ иерархических и сетевых моделей данных.
2. Основные характеристики объектно-ориентированных моделей.
3. Характеристики реляционных моделей.
4. Общая характеристика и сравнительный анализ современных реляционных СУБД.

В теме 3 была рассмотрена классификация моделей данных. Рассмотрим теоретико-графовые как самые первые из моделей СУБД.

К основным понятиям иерархической структуры относятся уровень, элемент, связь.

Исторически иерархическая модель появилась раньше сетевой. Она наиболее проста из всех моделей данных. Самой известной иерархической системой позволяющей создавать иерархические базы данных является система IMS (Information Management System) фирмы IBM, используемая в свое время для поддержки лунного проекта «Аполлон». Появление иерархической модели связано с тем, что в реальном мире очень многие связи соответствуют иерархии, когда один объект выступает как родительский, а с ним может быть связано множество подчиненных объектов.

Основными информационными единицами в иерархической модели являются: база данных (БД), сегмент и поле. Поле данных определяется как минимальная, неделимая единица данных, доступная пользователю с помощью СУБД. Выделяют также тип поля, представляющий собой совокупность полей одного типа. Сегмент состоит из конкретных экземпляров полей. Тип сегмен-та — совокупность входящих в него типов полей. Иерархическая модель представляет собой неориентированный граф, в вершинах которого располагаются сегменты (или типы сегмента). Дуги, соединяющие узлы, представляют собой связи или типы связей. Особенностью такой модели является то, что каждый сегмент может иметь не более одного предка, произвольное количество по-

томков и, по крайней мере, одно поле. Сегмент, который не имеет потомков, называют листовым сегментом. Иерархическое дерево начинается с одного сегмента, называемого корневым сегментом. Очень важно, что каждый сегмент должен иметь свое уникальное имя или идентификатор.

Сетевая модель относится к ранним моделям данных. Сейчас информации об этой модели данных почти не встретишь, даже та-кой специалист как К. Дейт при переиздании своей классической книги «Введение в системы баз данных» исключил вопросы, каса-ющиеся сетевой и иерархической модели данных. В 1971 г. группа DTBG (Database Task Group) представила в американский нацио-нальный институт стандартов отчет, который послужил в даль-нейшем основой для разработки сетевых систем управления база-ми данных. Стандарт сетевой модели впервые был определен в 1975 г. организацией CODASYL (Conference of Data System Langu- ages), которая определила базовые понятия модели и формальный язык описания.

Сетевая модель данных опирается на математическую теорию направленных графов. Базовыми элементами сетевой модели являются:

Элемент данных — минимальная информационная единица доступная пользователю.

Агрегат данных — именованная совокупность элементов данных внутри записи или другого агрегата. Агрегат бывает двух видов — агрегат типа вектор и агрегат типа повторяющаяся группа. Например, агрегат <город, улица, дом, квартира>, которому можно присвоить имя Адрес, является агрегатом типа вектор. Примером, агрегата типа повторяющаяся группа может служить агрегат <месяц, сумма> с названием Зарплата. Агрегат повторяющаяся группа характеризуется числом повторений. В данном примере это число повторений равно 12.

Запись — совокупность агрегатов или элементов данных, отражающих некоторую сущность предметной области. Например, записью будет <Фамилия, Зарплата>, где Фамилия — это элемент данных, а Зарплата — агрегат. Данную запись можно назвать Зарплата сотрудника.

Тип записей — эта совокупность подобных записей. Например, в предыдущем случае типом записи будет совокупность всех запи-сей Зарплата сотрудника, выражающая множество сотрудников

некоторого отдела. Тип записей представляет (моделирует) некоторый класс реального мира.

Набор — именованная двухуровневая иерархическая структу-ра, которая содержит запись владельца и запись (или записи) чле-нов. Наборы отражают связи «один ко многим» и «один к одному» между двумя типами записей. На рис. 4.1 представлен пример на-бора. Здесь «Отдел» — запись-владелец, «сотрудник» — запись-член. Тип набора определяет связь между двумя типами записей. Каждый экземпляр типа набора содержит один экземпляр записи владельца и произвольное количество записей-членов (для связи типа «один-ко-многим»). Среди всех наборов в сетевой модели до-пускается существование наборов, не имеющих владельцев. Такие наборы называются сингулярными. Владельцами сингулярных на-боров формально считается система. Сингулярные наборы пред-назначены для доступа к экземплярам отдельных записей.

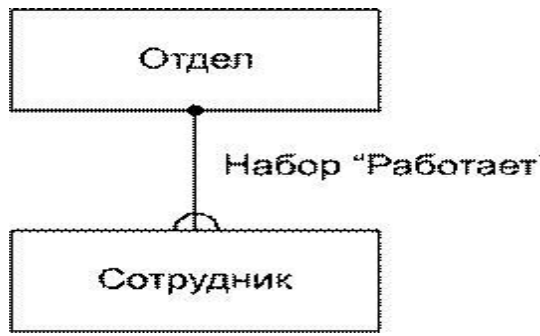


Рис. 4.1. Набор данных сетевой модели

Операторы в языке манипулирования иерархической модели делятся на:

- операторы поиска данных;
- операторы поиска данных с возможностью модификации;
- операторы модификации данных.

Синтаксис операторов поиска:

- GET HOLD UNIQUE <имя сегмента> WHERE <список поиска>;
- GET HOLD NEXT [WHERE дополнительные условия>];
- GET HOLD NEXT WITHIN PARENT [where <дополн.условия>];

 GET UNIQUE <имя сегмента> WHERE <список поиска>.

Список поиска состоит из последовательности условий вида:

 <имя сегмента>.<имя поля>ОС <constant или имя другого поля данного сегмента или имя переменной>;

 ОС — операция сравнения;

 условия могут быть соединены логическими операциями И и ИЛИ {&, }.

Назначение: получить единственное значение.

Пример

Найти студента с номером зачетной книжки К309080827 стоимостью не более 600 долл., которая существует не менее чем в 10 экземплярах.

GET UNIQUE СТУДЕНТЫ WHERE Студенты.Номер_ЗК = ‘К309080827’

Данная команда всегда ищет с начала БД и останавливается, найдя первый экземпляр сегмента, удовлетворяющий условиям поиска.

Найти и удержать единственный экземпляр сегмента. Эта операция подобна первой операции поиска GET UNIQUE, единственным отличием этой операции является то, что после выполнения этой операции над найденным экземпляром сегмента допустимы операции модификации (изменения) данных.

Синтаксис операторов поиска с дальнейшей модификацией:

 GET HOLD UNIQUE <имя сегмента> WHERE <список поиска>;

 GET HOLD NEXT [WHERE дополнительные условия>];

 GET HOLD NEXT WITHIN PARENT [where <дополн.условия>];

 GET HOLD UNIQUE <имя сегмента>;

 WHERE <список поиска>.

Найти и удержать следующий с теми же условиями поиска. Ана- логично операции 4 эта операция дублирует вторую операции по-иска GET NEXT с возможностью выполнения последующей моди-фикации данных.

Синтаксис:

GET HOLD NEXT [WHERE дополнительные условия>]

Получить и удержать следующий для того же родителя. Эта операция является аналогом операции поиска 3, но разрешает выполнение операций модификации данных после себя.

Синтаксис:
GET HOLD NEXT WITHIN PARENT [where
<дополн.условия>] Операторы модификации данных:
Удалить Операторы
модификации: DELETE

UPDATE
INSERT <имя сегмента>
Это первая из трех операций модификации.
Синтаксис:
DELETE
Эта команда не имеет параметров. Потому что операции модификации действуют на экземпляр сегмента, найденный командами поиска с удержанием. А он всегда единственный текущий найденный и удерживаемый для модификации экземпляр конкретного сегмента. Поэтому при выполнении команды удаления будет удален именно этот экземпляр сегмента.
Обновить
Синтаксис:
UPDATE
Как же происходит обновление, если мы и в этой команде не задаем никаких параметров. СУБД берет данные из рабочей области пользователя, где в шаблонах записей соответствующих внутренних переменных находятся значения полей каждого сегмента внешней модели, с которой работает данный пользователь. Именно этими значениями и обновляется текущий экземпляр сегмента. Значит, перед тем как выполнить операции модификации UPDATE, необходимо присвоить соответствующим переменным новые зна-чения.
Ввести новый экземпляр
сегмента. INSERT <имя сегмента>
Эта команда позволяет ввести новый экземпляр сегмента, имя которого определено в параметре команды. Если мы вводим данные в сегмент, который является подчиненным некоторому родительскому экземпляру сегмента, то он будет внесен в БД и физически подключен к тому экземпляру родительского сегмента, который в данный момент является текущим.

Как видим, набор операций поиска и манипулирования данными в иерархической БД невелик, но он вполне достаточен для по-

лучения доступа к любому экземпляру любого сегмента БД. Однако следует отметить, что способ доступа, который применяется в данной модели, связан с последовательным перемещением от одного экземпляра сегмента к другому. Такой способ напоминает движение летательного аппарата или корабля по заданным координатам и называется навигационным.

Однако иерархическая модель имеет свои недостатки:

- сложность реализации. От проектировщиков и программистов требуется достаточно высокий уровень подготовки;
- сложность управления. Любые изменения в структуре БД, например, перемещение сегментов, вызовут необходимость изменений во всех прикладных программах. Хотя иерархическая БД обеспечивает целостность БД, в то же время она дает возможность удалить один сегмент, что приведет к автоматическому удалению всех сегментов-потомков;
- недостаток структурной независимости. Структурная независимость имеет место, если изменения в структуре БД не влияют на возможность доступа СУБД к данным. Для навигации по заданным сегментам используется маршрут физического хранения. Программист должен знать маршрут к соответствующим сегментам;
- сложность программирования и использования приложе-ний. Для доступа к данным необходимо точно знать, каким образом физически данные размещены в БД. Поэтому говорят, что иерархические БД были созданы программистами для программистов.

В конце 1970-х — начале 1980-х гг. иерархические БД стали терять свою популярность. Стали разрабатываться альтернативные модели данных. Одной из таких моделей стала сетевая модель данных.

Основные информационные единицы сетевой модели: агрегат данных, элемент данных, запись, тип записи, набор.

Базовыми языками в сетевых СУБД могут быть COBOL, PL/I.

Выделяют язык описания данных и язык манипулирования данными.

Язык описания данных в сетевой модели имеет несколько разделов:

- описание базы данных — области размещения;
- описания записей — элементов и агрегатов (каждого в отдельности);

 описания наборов (каждого в отдельности).

SCHEMA IS <Имя БД>.

AREA NAME IS <Имя физической области>

RECORD NAME IS <Имя записи (уникальное)>

Для каждой записи определяется способ размещения экземпля-ров записи данного типа:

LOCATION MODE IS {DIRECT (напрямую) |

CALC <Имя программы> USING <[Список пер.>] DUPLICATE ARE [NOT] ALLOWED

VIA <Имя набора> SET (рядом с записями владельца) SYSTEM (решать будет система)}

Каждый тип записи должен быть приписан к некоторой физической области размещения:

WITHIN <Имя области размещения> AREA

После описания записи в целом идет описание внутренней структуры:

<Имя уровня> <Имя данного> <Шаблон> <Тип> Номер уровня определяет уровень вложенности при описании элементов и агрегатов данных. Первый уровень — сама запись. Поэтому элементы или агрегаты данных имеют уровень начиная со второго. Если данное соответствует агрегату, то любая его составляющая добавляет один уровень вложенности.

Если агрегат является вектором, то он описывается как

<Номер уровня> <Имя агрегата>.<Номер уровня>

<Имя 1-й сост.>

А если — повторяющейся группой, то следующим образом:

<Номер уровня> <Имя агрегата>.OCCURS <N> TIMES,

где N — среднее количество элементов в группе.

Описание набора и порядка включения членов в него выглядит следующим образом:

SET NAME IS <Имя набора>

OWNER IS (<Имя владельца> | SYSTEM)

Далее указывается порядок включения новых экземпляров члена данного набора в экземпляр набора:

ORDER PERMANENT INSERTION IS {SORTED | NEXT | PREV | LAST | FIRST}

После этого описывается член набора с указанием способа вклю- чения и способа исключения экземпляра — члена набора из экзем-пляра набора:

MEMBER IS <Имя члена набора>
{AUTOMATIC | MANUAL} {MANDATORY | OPTIONAL}
KEY IS (ACCENDING | DESCENDING) <Имя элемента данных>
При автоматическом включении каждый новый экземпляр чле-на набора автоматически попадает в текущий экземпляр набора в соответствии с заданным ранее порядком включения. При ручном способе экземпляр члена набора сначала попадает в БД, а только потом командой CONNECT может быть включен в конкретный экземпляр набора.

Если задан способ исключения MANDATORY, то экземпляр записи, исключаемый из набора, автоматически исключается и из базы данных. Иначе просто разрываются связи.

Внешняя модель при сетевой организации данных поддерживается путем описания части общего связного графа.

READY Обеспечение доступа данного процесса или пользователя к БД (сходна по смыслу с операцией открытия файла)

FINISH Окончание работы с БД

FIND Группа операций, устанавливающих указатель найденного объекта на текущий объект

GET Передача найденного объекта в рабочую область. Допустима только после FIND

STORE Помещение в БД записи, сформированной в рабочей области

CONNECT Включение текущей записи в текущий экземпляр набора

DISCONNECT Исключение текущей записи из текущего экземпляра набора

MODIFY Обновление текущей записи данными из рабочей области пользователя

ERASE Удаление экземпляра текущей записи

02 Товар VIRTUAL
SOURCE IS Товары.НаименованиеТовара
OF OWNER OF Товар-Цены SET

Наиболее интересна операция поиска (FIND), так как именно она отражает суть навигационных методов, применяемых в сетевой модели. Всего существует семь типов операций поиска:

По ключу (запись должна быть описана через CALC USING ...): FIND <Имя записи>
RECORD BY CALC KEY <Имя параметра>

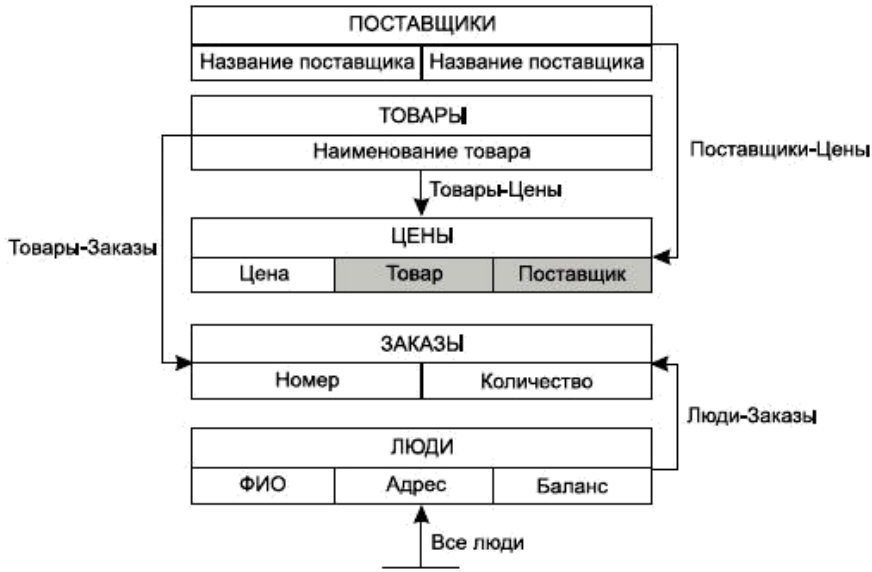


Рис. 4.2. Пример сетевой модели

Последовательный просмотр записей данного типа:
FIND DUPLICATE <Имя записи>
RECORD BY CALC KEY
Найти владельца текущего экземпляра набора:
FIND OWNER OF CURRENT <Имя набора> SET
Последовательный просмотр записей — членов текущего экземпляра набора:
FIND (FIRST | NEXT) <Имя записи>
RECORD IN CURRENT <Имя
набора> SET
Просмотрзаписей-членовэкземпляранабора,специфицированных рядом полей:
FIND [DUPLICATE] <Имя записи>
RECORD IN CURRENT <Имя
набора> SET USING <Список полей>
Сделать текущей записью процесса текущий экземпляр
набора: FIND CURRENT OF <Имя набора> SET
Установить текущую запись процесса: FIND
CURRENT OF <Имя записи> RECORD

Преимущества сетевой модели

В сетевой модели сохранились преимущества иерархической мо- дели данных, но были исправлены многие недостатки:

- концептуальная простота. Абстрактное представление БД достаточно простое;
- поддержка других типов связей. Связь типа m:n в сетевой модели реализуется проще, чем в иерархической за счет того, что в ней допускается наличие нескольких владельцев записи;
- гибкий доступ к данным значительно выше, чем в иерархической и в системах файлов;
- любое приложение может получить доступ к записи-владельцу и всем записям-членам набора без долгого прямого обхода;
- обеспечение целостности данных за счет того, что пользователь должен сначала определить запись-владелец, а потом уже записи-члены (запись-член не может существовать без записи-владельца);
- независимость данных. Частично избавляет от программирования сложных деталей, связанных с методами физического хранения информации. Поэтому изменения в свойствах данных не потребуют переделки тех участков прикладных программ, где выполняется доступ к данным;
- соответствие стандартам. Стандарты DDL и DML значительно улучшили возможности администрирования БД.

Недостатки сетевой модели Сложность системы в целом. Так же, как и иерархическая, пре-

доставляет навигационные средства доступа к данным, поэтому администраторы, программисты для получения доступа должны хорошо знать внутреннюю структуру БД. Сетевую модель, как и иерархическую, нельзя считать дружественной системой;

Недостаточная структурная независимость. Поскольку сетевая модель обеспечивает доступ к данным с помощью средств навигации, трудно производить структурные изменения. Если делаются какие-либо изменения, то необходимо проверить все прикладные программы.

В 1980-х гг. сетевая модель была вытеснена реляционной моделью данных.

Объектно-ориентированные модели

Термин «объект» в программной индустрии впервые был введен в языке Simula (1967) и означал какой-либо аспект моделируемой реальности. Сейчас под объектом понимается «нечто, имеющее четко определенные границы» (определение известного американского специалиста Г. Буча). Объекты, обладающие одинаковыми свойствами, составляют классы (например, курица, пингвин и чайка — объекты класса «птицы»). Обычно класс описывается как новый тип данных, а объекты (экземпляры класса) — определенные на его основе переменных.

Сразу же необходимо заметить, что общепринятого определения «объектно-ориентированной модели данных» не существует. Сейчас можно говорить лишь о неком «объектном» подходе к логическому представлению данных и о различных объектно-ориентированных способах его реализации.

Объектно-ориентированная база данных (ООБД) — база данных, основанная на принципах объектно-ориентированной технологии. К основным описательным моментам, связанным с ООБД, в литературе [26] относят:

- объекты (в ООБД любая сущность — объект и обрабатывает-ся как объект); классы (понятие «тип данных» реляционной модели заменяется понятиями «класс» и «подкласс»);
- метод — функция, определяющая поведение объекта;
- наследование (классы образуют иерархию наследования, заимствуя свойства друг друга);
- атрибуты (характеристики объекта моделируются его атрибутами);
- сообщения и методы (каждый класс имеет определенную совокупность методов, классы взаимодейству-ют друг с другом посредством механизма сообщений);
- инкапсуляция (внутренняя струк- тура объектов скрыта);
- идентификаторы объектов — дескрипторы.

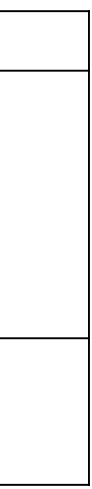


Рис. 4.3. Определение класса «Предприятие»

Используя наследование, всем объектам ПРЕДПРИЯТИЕ можно приписать свойство объекта-родителя (ФИЛИАЛ) — название филиала, номер филиала. Схема представления класса ПРЕДПРИЯТИЕ приводится на рис. 4.3.

Классы организуются в иерархию классов (рис. 4.4).

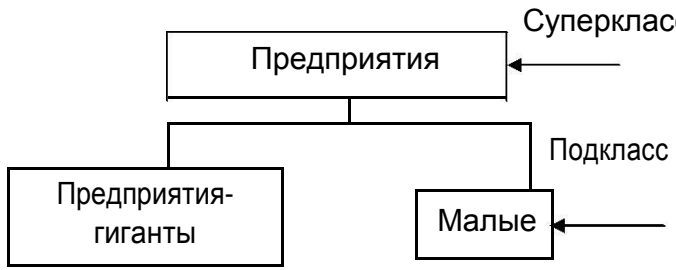


Рис. 4.4. Пример иерархии наследования

Иерархия классов обеспечивает мощную концепцию ОО-подхода, которая называется наследованием. Для расчета налоговой нагрузки малых предприятий используется УСН, а следовательно, необходимы и другие переменные экземпляра. Для этого определяется метод налоговая нагрузка для всего класса Предприятие, для подкласса Малые предприятия происходит переопределение метода.

При этом переопределение метода Налоговая нагрузка() в УСН() в подклассе Малые предприятия не влияет на расчет налоговой нагрузки в классе Предприятия-гиганты.

В отличие от переопределения методов полиморфизм позволяет различным объектам реагировать на одно и тоже сообщение различными способами.

Достоинства модели:

- улучшенные возможности моделирования;
- расширяемость; - более выразительный язык запросов;
- повышенная производительность.

Недостатки модели:

- отсутствие стандартов;
- сложность;
- недостаточность средств обеспечения защиты данных в СУБД этого типа.

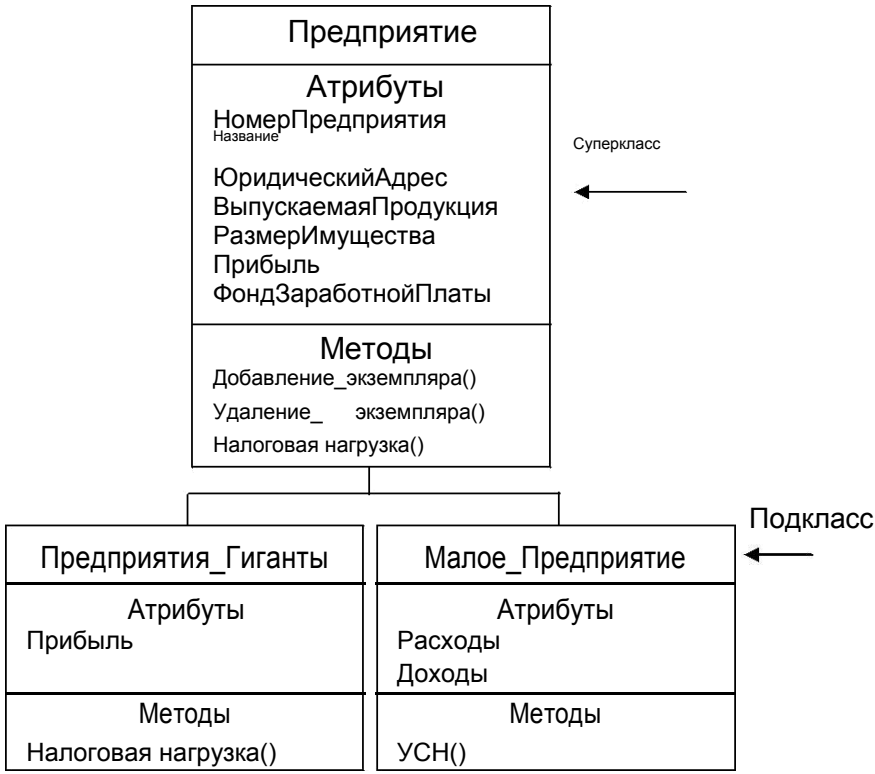


Рис. 4.5. Пример иерархии наследования с переопределением методов

Реляционные модели

Множество-совокупность элементов, обладающих некоторым общим свойством и отвечающих следующим условиям:

1. Должно существовать правило, позволяющее определить, принадлежит ли указанный элемент данной совокупности.

2. Должно существовать правило, позволяющее отличать элементы друг от друга (это, в частности, означает, что множество не может содержать двух одинаковых элементов).

Множества обозначаются заглавными буквами латинского алфавита. Основными операциями над множествами являются объединение, пересечение и разность, декартово произведение.

Отношение — это подмножество декартового произведения множеств. Отношения состоят из однотипных кортежей. Каждое

отношение имеет предикат отношения и каждый *n*-местный предикат задает *n*-арное отношение.

Отношения обладают степенью и мощностью. Степень отношения — это количество элементов в каждом кортеже отношения (аналог количества столбцов в таблице). Мощность отношения — это мощность множества кортежей отношения (аналог количества строк в таблице).

Теорию реляционных БД создал в конце 1960-х гг. Э. Кодд, кото-рый предложил использовать для обработки данных аппарат тео-рии множеств. Он показал, что любое представление данных сво-дится к совокупности двумерных таблиц особого вида, известных в математике как отношения.

На каждой схеме отношения (схеме базы данных) может быть построено несколько отношений (баз данных), которые могут трактоваться как версии одной и той же таблицы (базы данных). На практике же обычно схеме соответствует лишь одно отноше-ние с тем же именем, и, тем самым, РБД не используют напрашива-ющуюся аналогию между схемой и структурным типом данных в языках программирования (в С этот тип — struct). В связи с этим схему отношения иногда называют заголовком отношения, а отно-шение как набор кортежей — телом отношения. Домен-это множе-ство атомарных значений одного и того же типа. Атрибуты одного и того же домена можно сравнивать.

Основной моделью в БД является отношение именно поэтому модель получила название реляционной (relation — отношение). *N*-арным отношением множества R называют подмножество декартова произведения D1 * D2 * … * D*n* (*n* <= 1). Необязательно различных. Исходные множества D1D2D*n* называются доменами. Полное декартово произведение — это набор всевозможных сочетаний из *n* элементов каждое, где каждый элемент берется из своего домена. Например, D1 — набор из трех фамилий, D2 — набор дисциплин, R1 = {Иванов, Магомедов}, R2 = {Базы данных, Информатика, Математика}.

Тогда декартово произведение содержит набор из 6 троек:

«Иванов, Базы данных» «Иванов, Информатика» «Иванов, Математика» «Магомедов, Базы данных» «Магомедов,

Информатика» «Маго-
медов, Математика».

Данное отношение моделирует реальную ситуацию. Оно может быть представлено в виде таблицы, столбцы которой соответству-

ют вхождениям доменов в отношение, а строки — наборам из *n* значений, взятых из исходных доменов, которые расположены в строго определенном порядке в соответствии с заголовком.

Фамилия	**Дисциплина**
Иванов	Базы данных
Иванов	Информатика
Иванов	Базы данных
Магомедов	Информатика
Магомедов	Базы данных
Магомедов	Информатика

Данная таблица обладает рядом специфических свойств:

□• В таблице нет двух одинаковых строк.

□• Таблица имеет столбцы, соответствующие атрибутам отно--шения.

□• Каждый атрибут в отношении имеет уникальное имя.

□• Порядок строк произвольный.

Вхождение домена в отношение принято называть атрибутом, строки отношения называются кортежами. Количество атрибутов называется степенью или рангом отношения. Каждое отношение обладает хотя бы одним ключом. Один из возможных ключей принимается за первичный.

Итак:

Отношение-таблица (иногда файл)

Кортеж-строка, запись Атрибут-

столбец, поле

Таким образом, реляционная база данных — это множество таблиц (отношений), которые обладают следующими свойствами:

В таблице отсутствуют одинаковые строки (отношение — это множество, а любое множество состоит из различных элементов). В некоторых реализациях РСУБД, тем не менее, строки-дубликаты допускаются в производных таблицах, получаемых при выполне-нии запросов (соответствующие им отношения являются уже мультимножествами). Из уникальности строк в таблице вытекает наличие у нее ключа, которым, по крайней мере, может быть пол-ный набор атрибутов (на самом деле, ключом является минималь-ный набор атрибутов, обеспечивающий уникальность строк).

Строки таблицы неупорядочены (что также следует из определения отношения как множества). Это не противоречит возможности сортировки результатов запросов к РБД по значениям некоторых атрибутов (такие результаты — не просто мультимножества, а упорядоченные мультимножества).

Столбцы таблицы имеют уникальные имена и неупорядочены, что теоретически позволяет, например, модифицировать схемы заполненных данными таблиц не только путем добавления новых атрибутов, но и путем удаления существующих. На практике не все РСУБД имеют такую возможность; более того, доступ к значениям столбцов может осуществляться по их номеру, который был им присвоен при задании схемы (этот способ быстрее, чем доступ по имени столбца).

Каждый столбец содержит однородные (однотипные) данные (принадлежащие домену).

Значения столбцов атомарны, т. е. имеют простой тип и не являются множествами. Если таблица обладает этим свойством, то говорят, что она является нормализованной (находится в первой нормальной форме).

Реляционная модель данных и ограничения целостности

Реляционная модель данных состоит из трех частей: структурной, целостной и манипуляционной. В структурной части фиксируется, что единственной структурой данных, используемой в РБД, является нормализованная таблица каждая строка которой обычно соответствует некоторой сущности реального мира.

В первую очередь при выборе СУБД необходимо принимать во внимание следующие факторы:

- максимальное число пользователей одновременно обращающихся к базе;
- характеристики клиентского ПО; -
аппаратные компоненты сервера; -
серверную операционную систему; -
уровень квалификации персонала.

Критерии оценивания СУБД

1. Экономические (цена приобретения, «полная стоимость владения» или TCO — total cost ownership, кэффициент возврата инвестиций или ROI, жизнеспособность поставщика, риски, выгоды, неоцениваемые в денежных единицах, и др.).

2. Технические (функции, включая приложения, уровень информационных технологий, принятых в реализации, открытость, надежность, эффективность, перспективы развития, уровень технической поддержки, и др.).

3, 4. Политические и культурологические (политика рыночных отношений, техническая политика центра, корпоративная культура, групповые и личные предпочтения, и др.).

На сегодняшний момент реляционные СУБД являются самы-ми распространенными из всех моделей. При выборе СУБД используется множество критериев на которые влияет и требова-ния к разрабатываемой системе и возможности разработчика, а также заказчика. Лидерами на рынке реляционных СУБД являются ORACLE и MICROSOFT.

Oracle Database — объектно-реляционная система управления базами данных (СУБД).

Microsoft SQL Server — система управления реляционными базами данных (СУБД), разработанная корпорацией Microsoft.

Ниже приведено сравнение СУБД с использованием трех характеристик: отлично, хорошо, приемлемо, плохо.

Таблица 4.1

Сравнительный анализ реляционных СУБД

Сравнительные характеристики	Microsoft SQL Server (Microsoft)	Oracle (Oracle Corporation)
Административное управление	Хорошо	Отлично
Простота обслуживания	Отлично	Отлично
Одновременный доступ нескольких пользователей	Хорошо	Отлично
Обработка мультимедиа-данных	Плохо	Отлично
Работа под управлением различный ОС	Приемлемо	Хорошо
Язык SQL	Отлично	Отлично
Распределенная обработка транзакций	Отлично	Отлично
Организация хранилищ данных и подготовка отчетов	Отлично	Отлично
Построение баз данных	Хорошо	Отлично

Таблица 4.2

Сравнительный анализ возможностей реляционных СУБД

Возможность	Майкрософт	Oracle
Регулятор ресурсов	+	+
Индексированные представления, выровненные по секциям	+	+
Оболочка PowerShell	+	
Управление на основе политик	+	
Отфильтрованные индексы	+	
Расширенные разреженные столбцы	+	
Многопоточная работа с секциями таблиц	+	
Сжатие префиксов столбцов	+	
Подписание модулей с помощью сертификатов	+	
Службы данных SQL Server	+	

Сравним персональные СУБД.

Visual FoxPro (VFP) **—** визуальная среда разработки систем управления реляционными базами данных, выпускаемая в настоящее время корпорацией Майкрософт.

Основные характеристики:

− высокая скорость; − высокий уровень объектной модели;

− монопольный и раздельный доступ пользователей; − применяется для приложений масштаба предприятия для работы на различных платформах.

MS Access — это функционально полная реляционная СУБД. MS Access одна из самых мощных, гибких и простых в использовании СУБД.

Основные характеристики:

− Access является одной из самых легкодоступных и понятных систем как для профессионалов, так и для начинающих пользователей, позволяющая быстро освоить основные принципы работы с базами данных;

− система имеет полностью русифицированную версию;

− полная интегрированность с пакетами Microsoft Office: Word, Excel, Power Point, Mail;

− идеология Windows позволяет представлять информацию кра- сочно и наглядно;

- возможность использования OLE технологии, что позволяет установить связь с объектами другого приложения или внедрить какие-либо объекты в базу данных Access;
- технология WYSIWIG позволяет пользователю постоянно видеть все результаты своих действий;
- широко и наглядно представлена справочная система; - существует набор «мастеров» по разработке объектов, облегчающий создание таблиц, форм и отчетов.

Таблица 4.3

Сравнительный анализ персональных СУБД

Сравнительные характеристики	Access	Visual FoxPro
Простота освоения и использования	Приемлемо	Хорошо
Сетевые возможности	Приемлемо	Хорошо
Скорость разработки	Приемлемо	Хорошо
Объем хранимых данных	Плохо	Приемлемо
Защита данных	Приемлемо	Приемлемо
Надежность	Приемлемо	Приемлемо
Требования к устройствам хранения данных	Хорошо	Хорошо
Простота администрирования	Приемлемо	Хорошо

В настоящее время характерными представителями профессиональных СУБД являются такие программные продукты, как Oracle, DB2, Sybase, Informix, Ingres, Progress.

Основоположниками СУБД Oracle стала группа американских ученых (Лари Эллисон, Роберт Майнер, Эдвард Оутс), которые более 20 лет назад создали фирму Relation Software Inc. и поставили перед собой задачу создать систему на практике, реализующую идеи Кодда и Дейта. Результатом их деятельности стала реализация переносимой реляционной системы управления БД с базовым языком обработки SQL. В своем развитии СУБД совершенствовалась. Уже пятая версия была реализована по принципу «клиент-сервер».

Первые версии распространялись бесплатно. В начале 80-х гг. появилась коммерческая версия. Среди персональных СУБД Foxbase, Foxpro, Access, Paradox.

Промежуточное положение между персональными и профессиональными СУБД занимают: SQL Windows Microsoft SQL Server.

Access является одной из самых популярных среди персональных СУБД.

Среди бесплатных СУБД выделяется:

Firebird (FirebirdSQL) — компактная, кроссплатформенная, сво-- бодная система управления базами данных (СУБД), работающая на GNU/Linux, Microsoft Windows и разнообразных Unix-плат- формах.

Соответствие требованиям ACID: Firebird сделан специально, чтобы удовлетворять требованиям «атомарности, целостности, изоляции и надежности» транзакций («Atomicity, Consistency, Isolation- and Durability»).

Декларативное описание ссылочной целостности: Обеспечивает непротиворечивость и целостность многоуровневых отношений «master-detail» между таблицами.

Наборы символов: Firebird поддерживает множество международных наборов символов (включая Unicode) с множеством вариантов сортировки.

Вопросы для самоконтроля:

1. Перечислите достоинства и недостатки теоретико-графовых моделей.
2. Перечислите основные элементы объектно-ориентированных моделей
3. Какие математические понятия лежат в основе реляционной модели?
4. Перечислите СУБД, поддерживающие реляционные модели.

Термины:

Элемент
данных Агрегат
данных
Сегмент Домен
Атрибут
Кортеж
Целостность
Отношения

Тема 5

ОПЕРАЦИИ РЕЛЯЦИОННОЙ АЛГЕБРЫ

1. Теоретико-множественные операции.
2. Специальные операции.
3. Реляционное исчисление.

Алгебра называется множеством объектов с заданной на нем совокупностью операций, замкнутых относительно этого множества. Это множество называется основным. Было предложено 8 операций, объединенных в 2 группы. К группе теоретико-множественных операций относят: объединение, пересечение, разность, декартово произведение.

Объединение. Объединение отношений *R* и *S*, которое представляет собой множество кортежей, которые принадлежат *R* или *S* либо им обоим. Оператор объединения применяется только к отношениям одной и той же арности. Поэтому все кортежи в результате имеют одинаковое число компонентов, кортеж *d*, *a*, *f* принадлежит обоим отношениям одновременно.

Разность. Разностью отношения *R* и *S*, обозначаемой как *R*-*S*, называется множество кортежей, принадлежащих *R*, но не принадлежащих *S*.

Пересечение. Результатом пересечения *R* и *S*, обозначаемого как *RS*, называется множество кортежей, принадлежащих *R* и принадлежащих *S*.

Декартово произведение. Пусть *R* и *S* — отношения арности k_1 и k_2 соответственно. Тогда декартовым произведением отношений *R* и *S* называется множество кортежей длины $k_1 + k_2$, первые k_1 ком-понентов которых образуют кортежи, принадлежащие *R*, а послед-ние k_2 — кортежи, принадлежащие *S*.

Рассмотрим операции на примере базы данных «Заказы» (см. рис. 5.1–5.4).

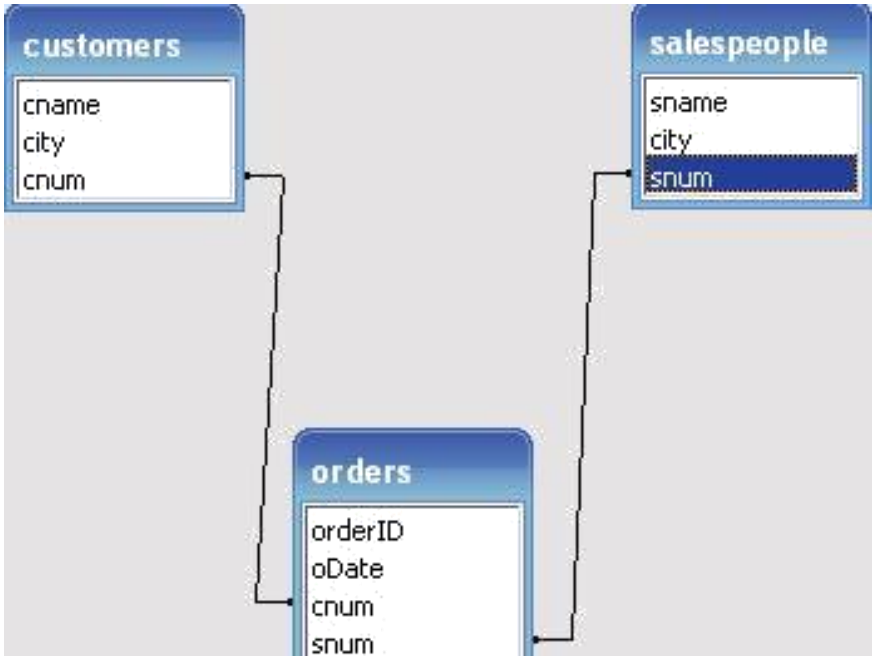


Рис. 5.1. Схема базы данных «Заказы»

customers : таблица

	cname	city	cnum
	Магомедов	Кизляр	1
	Петров	Махачкала	2
	Кураев	Хасавюрт	3
*			

Рис. 5.2. Содержимое таблицы «Заказчики»

salespeople : таблица

	sname	city	snum
▶	Алиев	Кизляр	01
	Магомедов	Кизляр	02
*			

Рис. 5.3. Содержимое таблицы «Продавцы»

	orderID	oDate	cnum	snum
▶	11	12.12.09	1	01
	12	13.12.09	1	02
*				

Рис. 5.4. Содержимое таблицы «Заказы»

Пример реализации операции объединения с помощью операторов SQL:

```
SELECT customers.cname,
customers.city FROM customers
UNION
SELECT salespeople.sname, salespeople.
city FROM salespeople
```

Результат операции представлен на рис. 5.5. В полученном отношении содержатся все города и заказчики или продавцы, которые в них находятся.

	cname	city
▶	Алиев	Кизляр
	Кураев	Хасавюрт
	Магомедов	Кизляр
	Петров	Махачкала

Рис. 5.5. Результат выполнения операции объединения

Пример выполнения операции пересечения на языке на
SQL: SELECT customers.cname, customers.city
FROM customers,salespeople
WHERE customers.cname=salespeople.sname

Результат выполнения представлен на рис. 5.6. В полученном отношении содержатся только те сведения, которые совпадают в обоих отношениях.

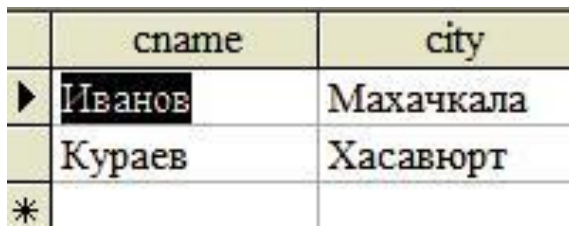
	cname	city
▶	Магомедов	Кизляр

Рис. 5.6. Результат выполнения операции пересечения

Пример реализации операции разности с помощью операторов SQL:

SELECT cname, city
FROM customers
WHERE cname NOT IN (SELECT sname FROM salespeople)

В представленном на рис. 5.7 результате содержатся только те строки из отношения Заказчики, которые не соответствуют строкам в отношении продавцы.



	cname	city
▶	Иванов	Махачкала
	Кураев	Хасавюрт
*		

Рис. 5.7. Результат выполнения операции разности

Операции 1 и 2 являются коммутативными.

Пример реализации операции декартово произведение с помощью операторов SQL:

SELECT customers.cname, customers.city, salespeople.sname, sales-people.city

FROM customers, salespeople

Результат выполнения операции декартово произведения представлен на рис. 5.8.

	cname	customers.city	sname	salespeople.city
▶	Магомедов	Кизляр	Магомедов	Кизляр
	Магомедов	Кизляр	Алиев	Кизляр
	Иванов	Махачкала	Магомедов	Кизляр
	Иванов	Махачкала	Алиев	Кизляр
	Кураев	Хасавюрт	Магомедов	Кизляр
	Кураев	Хасавюрт	Алиев	Кизляр

Рис. 5.8. Результат выполнения операции декартово произведения

Реляционная алгебра набор операторов, использующих отношения в качестве аргументов, и возвращающие отношения в качестве результата.

Информационная алгебра — это множество объектов с заданной на нем совокупностью операций, замкнутых относительно этого множества.

Информационный объект — это описание некоторой сущности (реального объекта, явления, процесса, события) в виде совокупности логически связанных реквизитов (информационных элементов).

Свойства информационного объекта определяются информационными параметрами, называемыми реквизитами.

Продавец:

Номер

Название

Адрес Отношения обозначаются заглавными буквами

латинского ал-

фавита, реквизиты обозначаются малыми латинскими буквами. Например:

salespeople

(S) snum (s1)

city (s2)

sname (s3)

Специальность по диплому (a4)
Заголовок отношения
R1(<a1:D1>,<a2:D2>,<a3:D3>,<a4:D4>)
R1(<a1.Val1>,<a2.Val2>,<a3.Val3>,<a4.Val4>)
Отдел (B)
Номер отдела (в1)
Название отдела (в2)
Заголовок отношения
R2(<b1:D1>,<b2:D2>)
R2(<b1.Val1>,<b2.Val2)

Отношение состоит из двух частей: заголовка отношения и тела отношения. Заголовок отношения — это аналог заголовка таблицы.

Заголовок отношения состоит из атрибутов. Количество атрибутов называется степенью отношения. Тело отношения — это аналог тела таблицы. Тело отношения состоит из кортежей. Кортеж отношения является аналогом строки таблицы. Количество корте-жей отношения называется мощностью отношения.

По способам реализации ограничения целостности делятся на:

- декларативные, выполняемые средствами языка SQL;
- процедурные, выполняемые посредством триггеров и хранимых процедур.

Специальные операции реляционной алгебры делятся на 4 вида.

Проекция — операция результатом, которой является подмножество кортежей, атрибуты которых соответствуют значениям $a_1 \ldots a_n$.

$\Pi_{a \ldots an}(R)$

Пример кода SQL, реализующего эту операцию: Select cname
From customers

Выборка — операция результатом, которой является подмножество кортежей, значения атрибутов которых соответствуют заданному условию.

$\sigma_{\text{предикат}}(R)$

Пример кода SQL, реализующего эту операцию: Select cname
From customers
Where cname = ‘Магомедов’

Соединение — операция результатом, которой является подмножество кортежей, атрибуты которых соответствуют значениям двух или нескольких отношений.

Можно заменить операциями декартово произведения и выборка.

$\sigma_F(R \times S)$

Операция соединения подразделяется на:

1. Тетасоединение:

$_F S$

2. Внешнее соединение:

$R \supset\!\triangleleft S$

3. Соединение по эквивалентности:

$R \triangleright\!\triangleleft S$ $R^{\triangleright\!\triangleleft}$

Пример выполнения операции соединения по эквивалентности с помо-щью операторов SQL:

```
SELECT Customers.cname, Salespeople.sname,
Salespeople.city FROM Salespeople, Customers
WHERE Salespeople.city =
Customers.city Тета-соединение:
SELECT sname, cname FROM
Salespeople, Customers
WHERE sname < cname
```

Результат выполнения представлен на следующем рис. 5.9.

	sname	cname
▶	Алиев	Магомедов
	Алиев	Петров
	Магомедов	Петров
	Алиев	Кураев

Рис. 5.9. Результат выполнения операции соединения по эквивалентности

Деление — операция результатом, которой является значения атрибута первого отношения, встречающееся во всех кортежах второго отношения.

$R \ S$

В основе лежат декартово произведение, разность, проекция.

$\Pi_a((R \times S) - \Pi(S))$

Пример реализации операции с помощью операторов SQL:

SELECT DISTINCT customers.cname, salespeople.sname

FROM customers INNER JOIN salespeople ON customers.city = sa- les-people.city

На рис. 5.10 представлен результат внутреннего соединения отношения Заказчики и Продавцы по равенству городов.

cname	sname
Магомедов	Алиев
Магомедов	Магомедов

Рис. 5.10. Результат выполнения операции внутреннего соединения

Рассмотрим пример выполнения операции деления. Эта операция является наиболее сложной из всех. Рассмотрим ее на примере следующих отношений:

Даны отношения: *R*

— список курсов;

S — список поездок, которые совершали курсы.

Найти город, в котором побывал каждый курс.

R

1 ФиК
1 ПИвЭ
1 Юр
1 Бух и А

S

Махачкала	1 ФиК
Пятигорск	1 ФиК
Дербент	1 ФиК
Махачкала	1 ПИвЭ
Махачкала	1 Юр
Дербент	1 Юр
Махачкала	1 Бух и А

Результата первого действия

Дербент	1 ФиК
Махачкала	1 ФиК
Пятигорск	1 ФиК
Дербент	1 ПИвЭ
Махачкала	1 ПИвЭ
Пятигорск	1 ПИвЭ
Дербент	1 Юр
Махачкала	1 Юр
Пятигорск	1 Юр
Дербент	1 БухА и А
Махачкала	1 БухА и А
Пятигорск	1 БухА и А

Результата второго действия

Дербент	1 ПИвЭ
Пятигорск	1 ПИвЭ
Пятигорск	1 Юр
Пятигорск	1 БухА и А
Дербент	1 БухА и А

Результат третьего действия

Дербент
Пятигорск

Результата четвертого действия

Дербент
Махачкала
Пятигорск

Результат пятого действия

Махачкала

Чаще эта операция реализуется как набор запросов.

Реляционное исчисление

Реляционное исчисление базируется на исчислении предикатов, которые используют интерпретированные понятия и методы

математической логики Реляционное исчисление представляет собой непроцедурный (декларативный) язык эквивалентный реляционной алгебре.

Реляционное исчисление определяется над отношениями реляционных баз данных, и результатом вычисления также является отношение. Преимуществом реляционного исчисления перед реляционной алгеброй является то, что пользователю не требуется самому строить алгоритм выполнения запроса, программа СУБД (при достаточной ее интеллектуальности) сама строит эффективный алгоритм.

В основе исчисления лежит понятие переменной с определенной для нее областью допустимых значений и понятие правильно построенной формулы, опирающейся на переменные, предикаты и кванторы.

Переменные определены на множестве доменов и кортежей, предикаты — это истинностная функция с параметрами, кванторы — префиксы, ограничивающие область истинности предиката. В качестве кванторов используются символы: ∀ (общности), ∃ (существования).

В основе реляционного исчисления лежит понятие предиката:

$$C|F(C),$$

где C — переменная, которая определена на каком-либо множестве;

$F(C)$ — предикат (в математической логике называется правильно по-строенной формулой — Well-Formed Formula, или сокращенно WFF).

Различают два вида реляционного исчисления:

− реляционное исчисление доменов (предложенного Лакруа);

− реляционное исчисление кортежей (предложенного Кодом).

В первом случае для описания отношений используются переменные, допустимыми значениями которых являются кортежи отношения, а во вто-ром случае — элементы домена.

В отношении логических операций существуют следующие фор- мулы эквивалентности:

$$(\exists X)(F(X)) \equiv \sim(\forall X)(\sim(F(X)));$$
$$(\forall X)((X)) \equiv \sim(\exists X)(\sim(F(X)));$$
$$(\exists X)(F1(X)) \equiv (F_2(X) \equiv \sim(\forall X)(\sim(F1(X))\vee\sim(F_2(X)));$$
$$(\forall X)(F1(X)) \equiv (F_2(X) \equiv \sim(\exists X)(\sim(F1(X))\vee\sim(F_2(X))).$$

Переменные кортежа называются свободными переменными, если они не квалифицируются кванторами ∀ или ∃; в противном случае они называ-ются связанными переменными. В выражении, составленном по правилам реляционного исчисления, свободные переменные могут находиться только слева от знака вертикальной черты (|).

Проще говоря, квантор общности используется в выражении, которое для всех экземпляров и чаще всего изображается рядом с символом отношения. Квантор существования означает, что существует хотя бы один отображается рядом с символом атрибута и чаще всего для тех, которые далее отражаются в предикате в качестве условия.

В реляционном исчислении не каждая последовательность формул является допустимой. Допустимыми формулами могут быть только недвусмысленные и небессмысленные последовательности.

Исчисление кортежей

В исчислении кортежей, как и в процедурных языках программирования, сначала нужно описать используемые переменные, а затем записать выражения запроса к данным.

Например, для отношения Orders:

Выбрать значения cnum для заказа от 12.12.2009

г. {O|Orders(O)∧O.o2=‘12/12/09’}

или {O.o3|Orders(O)∧

O.o2=‘12/12/09’}

Более сложные варианты WFF строятся с помощью логических связок NOT, AND, OR и IF ... THEN или используются операторы ∧, ∨, ~. В качестве кванторов используются символы: ∀ (общности), ∃ (существования).

В качестве итоговых могут выступать следующие функции: COUNT (количество), SUMM (сумма), AVG (среднее), MAX (максимальное), MIN (минимальное).

Пример использования кванторов в исчислении кортежей:

СОТРУДНИКИ (С) (СОТР_НОМЕР(с1), СОТР_ИМЯ(с2), СОТР_ЗАРПЛ(с3), ОТД_НОМЕР(с4))

ОТДЕЛЫ (О)(ОТД_НОМЕР(о1), ОТД_КОЛ(о2), ОТД_НАЗ(о3), ОТД_КОЛ(о4))

Создадим список сотрудников, которые работают в бухгалтерии:

{(C.c2|Сотрудники(C)∧(∃C)(Отделы(O)∧(C.c4=O.o4)∧(O.o3=’бухгалтерия’))}

Исчисление доменов

Реляционное исчисление доменов является основой для языка Query-by-Example.

В исчислении доменов областью определения переменных являются не отношения, а домены.

{o1,o2,o3,o4 | Orders(O)∧O.o2=‘12/12/09’}

Квантор — общее название для логических операций, ограничивающих область истинности какого-либо предиката:

Общности (∀) (для всех, любой)

Существования (∃) (существует хотя бы один, найдется)

Рассмотрим пример применения реляционной алгебры и реляционного исчисления на следующих отношениях:

СОТРУДНИКИ (СОТР_НОМЕР, СОТР_ИМЯ, СОТР_ЗАРПЛ, ОТД_НОМЕР)

ОТДЕЛЫ (ОТД_НОМЕР, ОТД_КОЛ, ОТД_НАЧ)

Создать список сотрудников, зарплата которых более 25 000 руб. (∀C)(C.c3 > 25000)

Задача: узнать имена и номера сотрудников, являющихся начальниками отделов с количеством работников более 10.

Выполнение с помощью реляционной алгебры:

(1). Выполнить соединение отношений СОТРУДНИКИ и ОТДЕЛЫ- по условию СОТР_НОМ = ОТДЕЛ_НАЧ.

C1 = (Сотрудники ∏ Отделы)сотрудники.сотр_ном=отделы.отд_нач или

C1 = СОТРУДНИКИ [СОТР_НОМ = ОТД_НАЧ] ОТДЕЛЫ.

(2). Из полученного отношения произвести выборку по условию ОТД_КОЛ > 10.

C2 003D$\sigma_{c1.отд_кол>10}$

или C2 = C1[ОТД_КОЛ > 10].

(3). Спроецировать результаты предыдущей операции на атрибуты СОТР_ИМЯ, СОТР_НОМЕР или

C3 = C2[СОТР_ИМЯ, СОТР_НОМЕР].

Выполнение с помощью реляционного исчисления кортежей:

(С.СОТР_НОМЕР,С.СОТР_ИМЯ|Сотрудники(С)∧((∃О) (ОТДЕЛЫ(О)∧С.СОТР_ИМЯ=О.НАЧ_ОТД)∧(О. ОТД_КОЛ>10)).

Вопросы для самоконтроля:

1. Перечислите операции, относящиеся к группе теоретико-мно- жественных?
2. Какая из специальных операций реляционной алгебры реализуется последовательностью других операций.
3. Дайте определение алгебры.
4. Что лежит в основе реляционного исчисления.
5. Каково назначение кванторов?

Термины

Информационная алгебра
Информационный объект
Объединение
Пересечение Вычитание
Декартово произведение
Соединение Проекция
Выборка Деление

Реляционное
исчисление Предикат
Квантор

Задачи для самостоятельной работы:

Задание 1

Выполните запрос с помощью реляционного исчисления:

Выдать СОТР_ИМЯ и СОТР_НОМ для СОТРУДНИКИ таких, что су-ществует ОТДЕЛ с таким же, что и СОТР_НОМ, значением ОТД_НАЧ и значением ОТД_КОЛ, большим 50.

Задание 2

Даны следующие отношения:

Товар (инвентарный№, наименование, единицы измерения, цена).

Поставка (№ накладной, датаПоставки, инвентарный№, количество).

Продажа (номерЧека, инвентарный№, датаПродажи, количество).

Ознакомьтесь с видами операций и их подвидами и запишите отношения, которые будут результатом выполнения следующих операций (в описание должны входить атрибуты отношений и описание содержимого кортежей):

1. R1 = (Продажа[Продажа.инвентарный№=Товар.инвентарный №] Товар).
2. R2 = (Продажа[Продажа.инвентарный№=Товар.инвентарный-№∧Продажа.датаПродажи=12.03.09]Товар).
3. R3 = (Поставка[Поставка.инвентарный№=Товар.инвентарный № ∧ Поставка.датаПоставки=12.03.09]Товар)[№ накладной, наименование, цена].
4. R4 = (Товар[инвентарный№] – Продажа[инвентарный№]).
5. R5 = Продажа[Продажа.[инвентарный№]=Товар.[инвентарный№]∧Товар.цена*Продажа.количество]Товар).
6. R6 = Продажа(инвентарный №,количество).
7. R7 = Поставка(инвентарный№,количество).
8. R8 = R7/R1.
9. R9 = R7–R8.

Задание 3

Даны отношения R1(ФИО, Дисциплина, Оценка), R2 (ФИО, Группа) и R3(Группа, Дисциплина), где R1 — список студентов сдавших экзамены, R2 — состав группы, R3 — список дисциплин по которым должны сдавать экзамен студенты группы. Используя операции реляционной алгебры определить список студентов, кто должен сдавать экзамен по БД, но еще не сдавал.

a) R = (R2[R2.Группа=R3.Группа∧R3.Дисциплина=БД]R3)[ФИО]
b) R = (R1–(R2[R2.Группа=R3.Группа ∧ R3.Дисциплина=БД]R3)) [ФИО]
c) R = (R2[R2.Группа=R3.Группа∨R3.Дисциплина=БД]R3)[ФИО]
d) R = σдисциплина=«БД»(ПФИО,Дисциплина((R2⊗R3)–R1))
e) R = ((R2[R2.Группа=R3.Группа∧R3.Дисциплина=БД]R3)/R1) [ФИО]
f) R = ((R2[R2.Группа=R3.Группа∨R3.Дисциплина=БД]R3)–R1) [ФИО]

Задача 4

Клиент может иметь несколько счетов, размещенных как в одном, так и в разных банках. В отношении R1(№ паспорта, ФИО_ Клиента, № _счета, назв_банка, сумма) содержится информация обо всех клиентах и их счетах. Используя операции реляционного исчисления составить запросы:

1. Клиенты, которые имеют счета более чем в 1 банке.

2. Клиенты, имеющие сумма средств которых на всех счетах составляет более чем 100 000 руб.

Тема 6

ОСНОВЫ ПРОЕКТИРОВАНИЯ РЕЛЯЦИОННЫХ СУБД

1. Методология проектирования
2. Нисходящий подход к проектированию
3. Восходящий подход к проектированию
4. Переход к реляционной модели.

Проектирование — процесс создания проекта базы данных, предназначенной для поддержки функционирования предприятия способствующей достижению его целей. Один из методов — MoSCoW-анализ. Буквы О используются для связки букв, являющимися первыми буквами в названиях:

Must have (необходимые функции), т. е. то, что должно быть включено в проектную поставку, или какие функции должна выполнять система.

Should have (желательные функции) — то, что хотелось бы иметь в проектной поставке, но этого может и не быть. Эти функции не будут планироваться заранее по различным причинам, иногда из-за них может возникать нерентабельность выполнения этих функций.

Could have (возможные функции). Won't have (отсутствующие функции).

Описанные функции реализуются в зависимости от приоритетности. Все это анализируется вместе с заказчиком.

Методология проектирования — структурированный подход, предусматривающий использование специализированных процедур, технических приемов, инструментов, документации и ориентированный на поддержку и упрощение процесса проектирования.

Первый этап — концептуальное проектирование — конструирование информационной модели предприятия, не зависящей от каких-либо физических условий реализации.

Концептуальному проектированию предшествует этап планирования базы данных, определение системы, сбор и анализ требований. На ранних этапах жизненного цикла применяются методики сбора фактов, к ним относятся: изучение документации, собеседование, наблюдение за работой предприятия, исследование, анкетирование.

Изучение документации, необходимый первоначальный этап оформляется в виде таблицы.

Таблица 6.1

Примеры типов документации, необходимых для изучения

Назначение документации	Примеры полезных источников
Описывает проблему и необходимость в базе данных	Отчеты о работе, структурная схема подразделений, жалобы служащих и заказчиков.
Описывает задачи предприятия, связанные с данной проблемой	Стратегический план предприятия, должностные инструкции сотрудников, чью работу необходимо автоматизировать, примеры заполненных рукописных и компьютеризированных форм и отчетов
Описывает существующую систему (если она имеется)	Блок-схемы и диаграммы, словарь данных, программная документация, руководства по обучению пользователей, проект приложения БД

Собеседование — интервью, чаще всего структурированное. Состоит из вопросов подразумевающих конкретные ответы, например: почему вам не нравится формат отчета по регистрации клиентов?

В процессе *наблюдения за работой* предприятия можно самому принять участие в этой работе.

Исследование проводиться через журналы, газеты, Internet.

Анкеты — документы специального назначения, позволяющие получить сведения от большого количества людей. При составлении анкет используются вопросы свободной формы, например, какие отчеты вы получили на сегодняшний момент, как вы их используете? И вопросы фиксированной формы, которые подразумевают выбор ответов из списка: да, нет, не знаю, не полностью со-гласен и т. д.

Построение концептуальной модели рассмотрим на примере фирмы по аренде недвижимости. В реестре недвижимости квартиры и дома. За каждым сотрудником закрепляется группа объектов. Сотрудники фирмы, выезжают на осмотр объекта недвижимости на служебном автомобиле. Рассмотрим этапы, предшествующие этапу концептуального проектирования (табл. 6.2).

Таблица 6.2

Изучение документации

Назначение документации	**Примеры полезных источников**
Описывает проблему и необходимость в базе данных	Отчеты отдела кадров о принятых и уволенных сотрудниках, о закрепленных за ними объектах недвижимости. Организационная схема организации. Отчеты бухгалтерии о расходах на бензин по выездам. Договора на оказание услуг. Реестр недвижимости на продажу
Описывает задачи предприятия, связанные с данной проблемой	Стратегический план организации, должностные инструкции риелторов, форма договора на аренду и договора купли-продажи, форма отчета для бухгалтерии о выезде, структура реестра объектов
Описывает существующую систему (если она имеется)	Существующей автоматизированной системы нет. Но работа ведется по следующему сценарию: 1. Объект недвижимости вносится в реестр и закрепляется за конкретным сотрудником. 2. Поступает заказ. 3. Сотрудник выбирает те объекты, которые удовлетворяют требованиям клиента по цене и состоянию. 4. Варианты обсуждаются с клиентом. 5. Если определен выбор, то осуществляется выезд на объект. 6. Выбирается объект, заключается договор. 7. Предоставляется отчет в бухгалтерию о сделанных выездах

На следующем этапе проводим собеседование с каждым из будущих пользователей. Обсуждаем их должностные инструкции, а также уровень владения средствами вычислительной техники. Обсуждаем формы документов и выслушиваем предложения по их изменению.

На этапе исследования знакомимся с газетами им формами объявлений данной фирмы, а также читаем отзывы клиентов для выяснения достоинств и недостатков в работе.

На этапе анкетирования составляем вопросник:

1. Должность.
2. С какими видами отчетов вы работаете?
3. В какие сроки составляются отчеты?
4. Какие проблемы испытываете при работе с входящей и исходящей документацией?

Построение концептуальной модели относится к нисходящему подходу проектирования.

Для построения концептуальной модели необходимо решить следующие задачи:

1. Определение типов сущностей.
2. Типов связей.
3. Определение атрибутов.
4. Определение доменов.
5. Определение ключевых атрибутов.
6. Проверка соответствие локальным пользовательским транзакциям.

Первый этап чаще всего завершается документированием ти-пов сущностей и составлением словаря данных. Ознакомьтесь с табл. 6.3 и охарактеризуйте отношение «Автомобиль сотрудника».

Таблица 6.3

Словарь данных

Имя сущности	Описание	Псевдоним	Характеристика
Сотрудники	Общее обозначение всех сотрудников компании по аренде недвижимости	Служащий	Каждый сотрудник компании работает в конкретном отделении
Недвижимость в аренду	Общее обозначение для всех объектов недвижимости	Недвижимость	Каждый объект недвижимости имеет одного владельца и передается в аренду в определенном отделении компании, в котором этим объектом недвижимости

Окончание таблицы 6.3

Имя сущности	Описание	Псевдоним	Характеристика
			управляет один сотрудник. Объект недвижимости сдаваемый в аренду, осматривают многие клиенты, а один клиент берет в аренду на определенный срок
Осмотр недвижимости	Сведения о проведенных осмотрах недвижимости, сдаваемой в аренду	Осмотр	Каждый объект недвижимости осматривается сотрудником для уточнения его состояния, требуемого ремонта, замены посуды, мебели и т. д.
Автомобиль сотрудника	Общее сведения о номерах автомобилей выезжавших на осмотр	Автомобили	Заполнить самостоятельно

Документирование типов связей:

Имя сущности	Кратность	Название связи	Имя сущности	Кратность
Сотрудники	0..1	ездит	Автомобиль сотрудника	0..10
Сотрудники	0..1	осматривает	Недвижимость	1..100

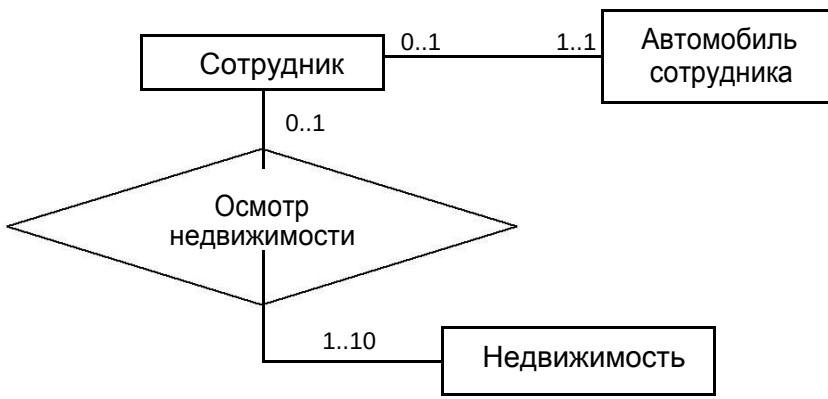


Рис. 6.1. Первая версия ER-диаграммы

В данном случае связь Осматривает будет характеризоваться дополнительными атрибутами: Дата осмотра, Комментарии. Поэтому выделяем ее в ассоциацию.

Первоначально модель будет иметь вид, представленный на рис. 6.1.

Определение атрибутов производится в соответствии со схема-ми объектов, представленных в учебнике Диго «Базы данных».

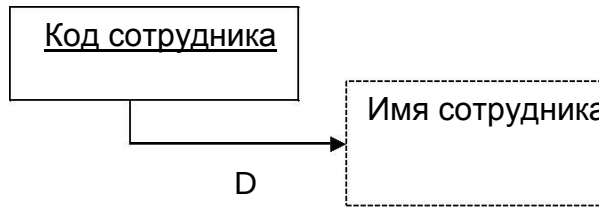


Рис. 6.2. Документирование Сущности Сотрудники

Проект базы данных — набор документации, включающей схемы графический материал, словесное описание, реализацию. различают 3 вида проектирования (проекта) инфологическое, даталогическое, физическое. Их последовательность позволяет реализовать принцип базы данных: *каждое сообщение хранится в БД один раз*.

Инфологическое проект включает в себя конструирование информационной модели не зависящей от каких-либо физических условий реализации. На этом этапе проводят системный анализ предметной области и построение инфологической модели как завершающую стадию Системный анализ предметной области проводится с использованием двух подходов: объектного и функционального. Функциональный характеризует принцип движения от задач и комплексов задач. Объектный построен на описании объектов и связей между ними Завершение системного анализа сопровождается дефиницией выходных документов, алгоритмов, задач, которые будут решаться с использованием базы данных.

Инфологическая модель представляется в виде тройки множеств:

$$\text{Мил} = <S_{\text{ил}}, P_{\text{ил}}, Q_{\text{ил}}>,$$

где $S_{\text{ил}}$ — множество сущностей и связей, задаваемых именами, ха-рактеристиками (свойствами) и их значениями;

$P_{ил}$ — правила интерпретации семантической сети (инфологического графа) данных;

$Q_{ил}$ — закономерности предметной области, существенные для контроля целостности и согласованности инфологической модели.

Рассмотрим составляющие этой модели:

$$S_{ил} = <G(E, R), A, N>,$$

где G — семантическая сеть (инфологический граф) данных (ССД/ ИГД):

- $E = \{e1, e2, \ldots, en\}$ — множество сущностей модели;
- $R = \{r1, r2, \ldots, rm\}$ — множество связей модели;
- $A = <AE, AR>$ — множество характеристик (свойств) модели;
- AE — множество характеристик (свойств) сущностей,
- AR — множество характеристик (свойств) связей;

$$Q_{ил} = <T, D, F>,$$

где T — множество закономерностей моделируемой предметной области;

D — множество значений характеристик (свойств);

F — множество функциональных зависимостей.

Порядок построения разработки ССД

Наиболее распространенным средством разработки ССД является модель «сущность-связь» (ER-модель). Автор — П. Чен (1976). В настоящее время широко применяются три нотации ER-моде- лирования:

- CASE-метод Баркера (СУБД ORACLE, 1990 г.);
- Stradis (CASE Vantage Team Builder);
- методология IDEF1 (автор Т. Ремей) и ее модификация (IDEF1X) (CASE ERWin, Design/IDEF).

Рассмотрим элементы ССД.

Сущность (entity) — это то, что функционирует в некоторой предметной области (абстрактные понятия, объекты реального мира, люди, явления природы и т. п.). Каждая сущность является множеством подобных индивидуальных объектов, называемых экземплярами сущности.

Особенности сущности:

1. Каждая сущность должна именоваться существительным в единственном числе с четким смысловым значением. Примером может быть сущность СТУДЕНТ (но не СТУДЕНТЫ).

2. Каждая сущность обладает одним или несколькими атрибутами. Атрибуты бывают простыми и составными.

Простые атрибуты используются для обозначения семантически неделимых характеристик (свойств) сущностей, выявленных при анализе предметной области (Фамилия СТУДЕНТА).

Составные атрибуты отображают сложные свойства сущностей (Адрес студента=(улица, дом, квартира)). Почему составной? Определяется задачей.

Каждое имя атрибута содержит имя сущности в родительном падеже.

Имена атрибутов непосредственно связываются с описанием множеств своих значений (доменов), на которых они должны обозначаться.

Каждый домен определяется на некотором типе данных: числовых (целых, вещественных), символьных (или строковых), бито-вых строк, специализированных числовых данных («деньги»), а также специальных темпоральных данных (дата, время, времен-ной интервал). Таким образом, домен это допустимое множество значений атрибута, определенный на некотором типе данных.

3. Каждая сущность обладает одним или несколькими атрибутами, которые однозначно идентифицируют экземпляры данной сущности. Пример. Фамилия курсанта или Номер военного билета курсанта. Такие атрибуты называются ключевыми (или просто ключом, или идентификатором). Если ключевых атрибутов несколько, то среди них выбирают один, называют его первичным ключом и помечают знаком PK.

Существующие между экземплярами сущностей предметной области- взаимосвязи (ассоциации, отношения) моделируются, так называемым, множеством связей, которые так же, как и сущности, могут обладать определенными свойствами (атрибутами).

Связи (relationship) может даваться имя, выражаемое грамматическим оборотом глагола и помещаемое возле линии связи. Имя связи выражает некоторое ограничение и облегчает чтение диаграммы, например:

Каждый студент <обучается в> ГРУППЕ
Каждая ГРУППА <состоит из> студентов В
группе <читает> лекции ПРЕПОДАВАТЕЛЬ
Каждую группу <курирует> ПРЕПОДАВАТЕЛЬ
ПРЕПОДАВАТЕЛЬ <является РУКОВОДИТЕЛЕМ
СТУДЕНТ <обучается> у ПРЕПОДАВАТЕЛЯ
СТУДЕНТАМ <назначается> СТИПЕНДИЯ

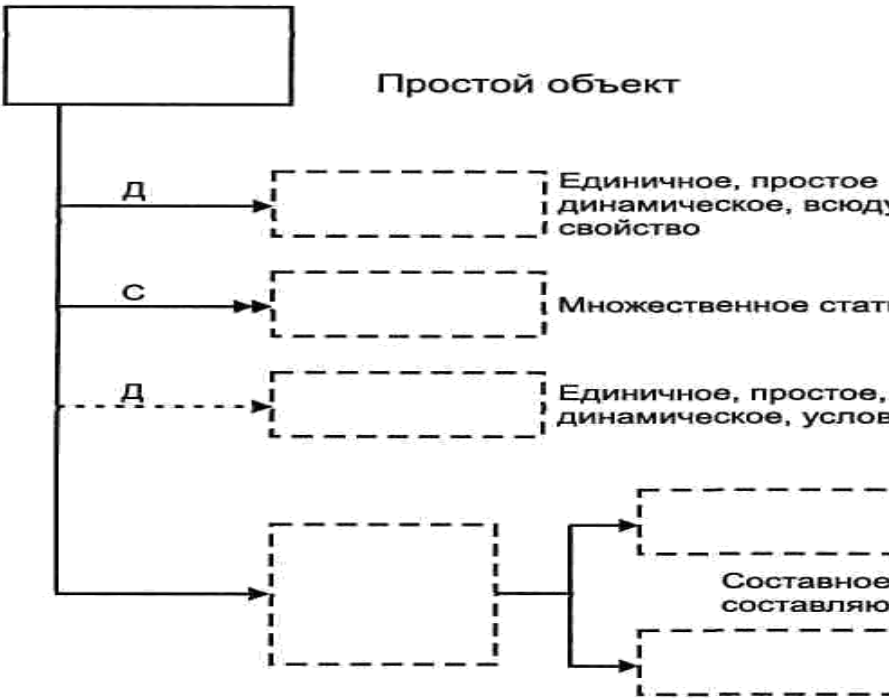


Рис. 6.3. Свойства сущности

Даталогическое проектирование представляет собой процесс создания информационной модели работы организации на основе конкретной модели данных, строится на основе инфологической модели.

Физическое проектирование процесс подготовки описания того, каким образом в базе данных будет представлена во внешней памяти с использованием возможностей выбранной СУБД. Включает в себя описание файловой организации, перечень индексов, обеспечивающих эффективный доступ, а также все соответствующие ограничения целостности и меры защиты.

Рассмотрим поэтапное построение инфологической модели.

Определяем свойства объекта (см. рис. 6.3).

Для определения типов связей строим диаграмму ER-экземпляров, которая отражает чтение преподавателями лекций в группах (рис. 6.4).

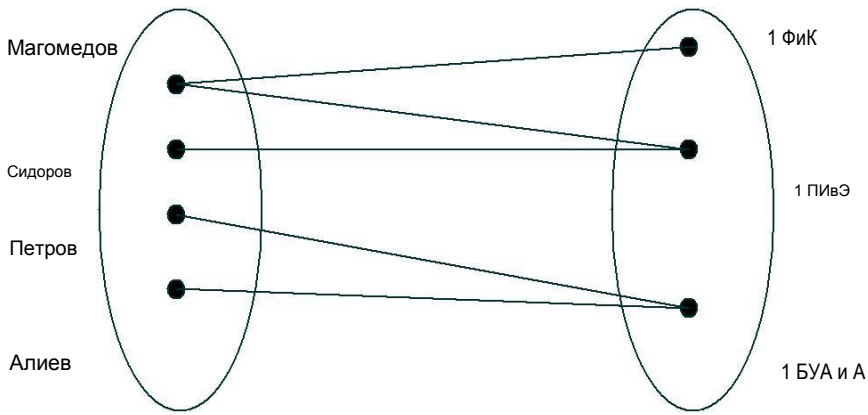


Рис. 6.4. Диаграмма ER-экземпляров

Далее рассматриваем диаграмму ER-типов (рис. 6.5).

Рис. 6.5. Диаграмма ER типов

Выбираем нотацию для построения модели (рис. 6.6, 6.7).

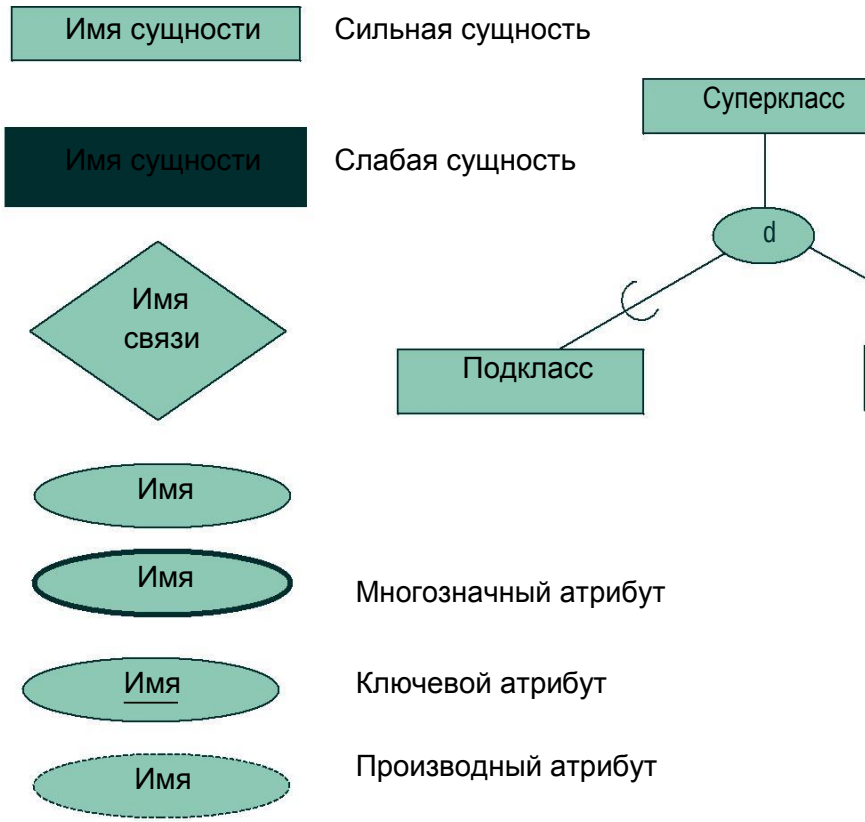


Рис. 6.6. Элементы ER-модели в нотации Питера Ч

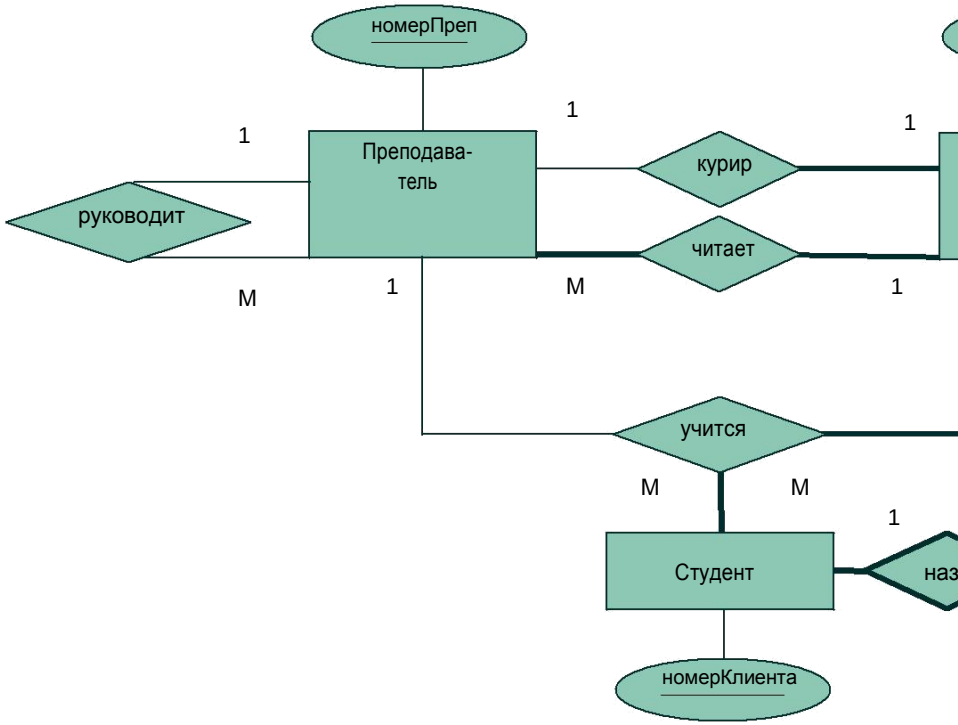


Рис. 6.7. ER-модель в нотации Питера Чена

Нотация «воронья лапка» представлена на рис. 6.8:

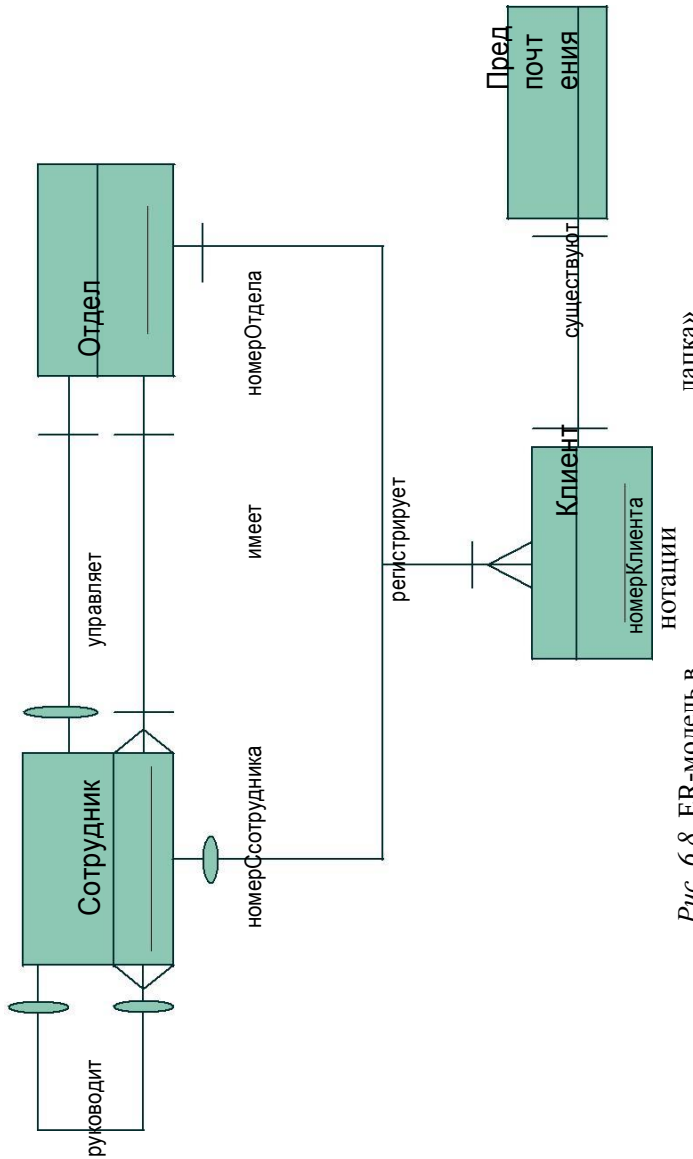


Рис. 6.8. ER-модель в нотации «воронья лапка»

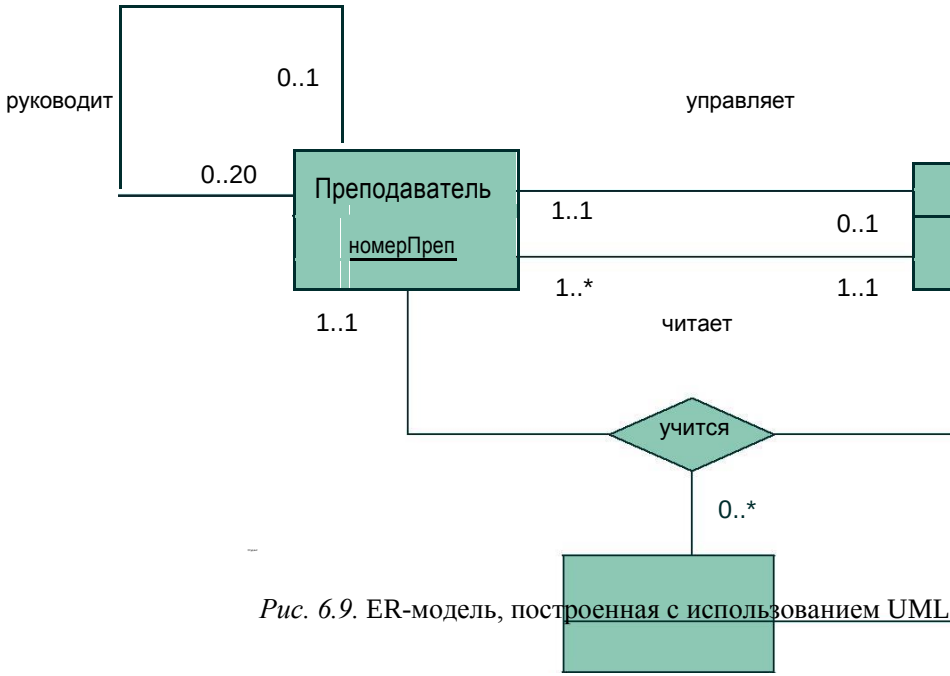


Рис. 6.9. ER-модель, построенная с использованием UML

Кратность-количество возможных экземпляров сущности некоторого типа (рис. 6.10):

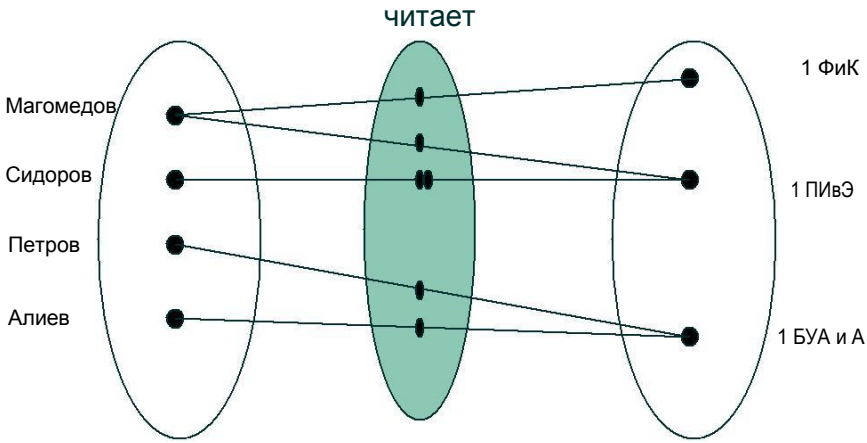


Рис. 6.10. Семантическая сеть

Семантическая сеть для определения кратности (рис. 6.11):

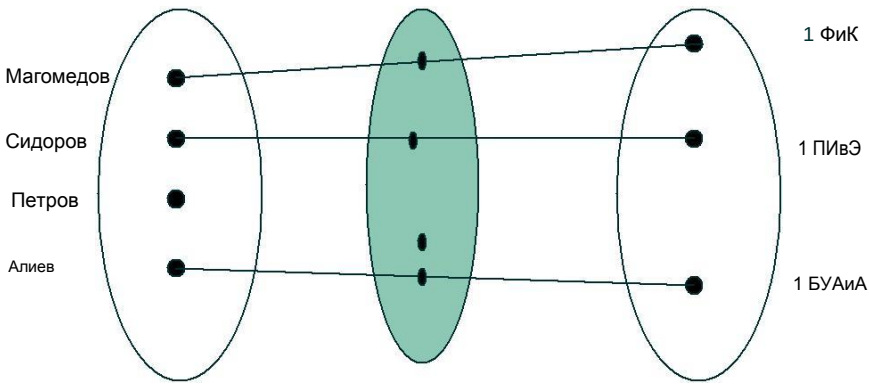


Рис. 6.11. Семантическая сеть

Ограничения ER-модели

Кардинальность определяет максимально возможное количество экземпляров связи для каждой сущности, участвующей в связи конкретного типа.

Степень участия определяет, участвуют в связи все или лишь некоторые экземпляры сущности.

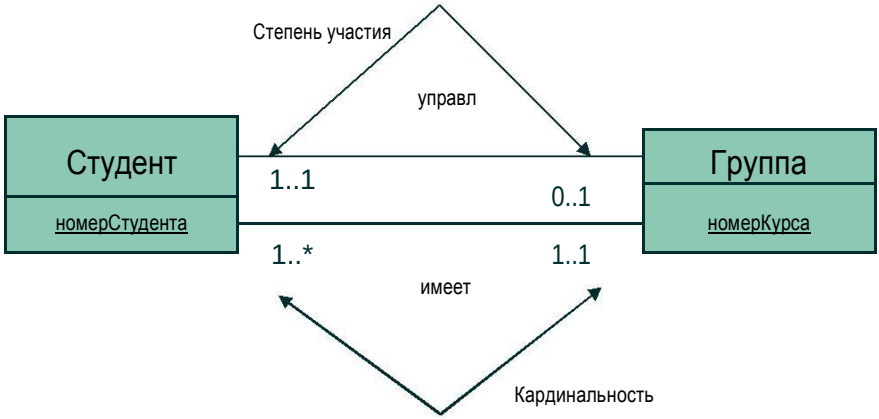


Рис. 6.12. Пример инфологической модели с ограничениями

Сущность слабого типа — тип сущности, существование которого зависит от какого-либо другого типа сущности. Сущность сильного типа — тип сущности, существование которого не зависит от какого-либо иного типа сущности.

Дефекты инфологической модели

Дефект типа разветвление (рис. 6.13):

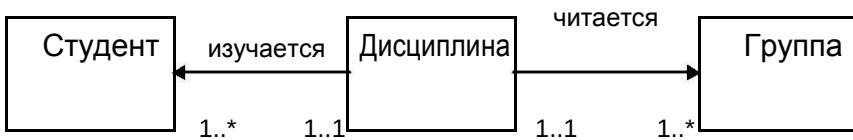


Рис. 6.13. ER-модель типа «Разветвление»

Семантическая сеть позволяет выявить дефекты инфологического моделирования.

На рис. 6.14 представлен дефект типа Разветвление. Здесь невозможно определить в какой группе учатся студенты Сидоров и Магомедов, Петров и Алиев. Из данного рисунка невозможно узнать, на каком курсе учатся студенты.

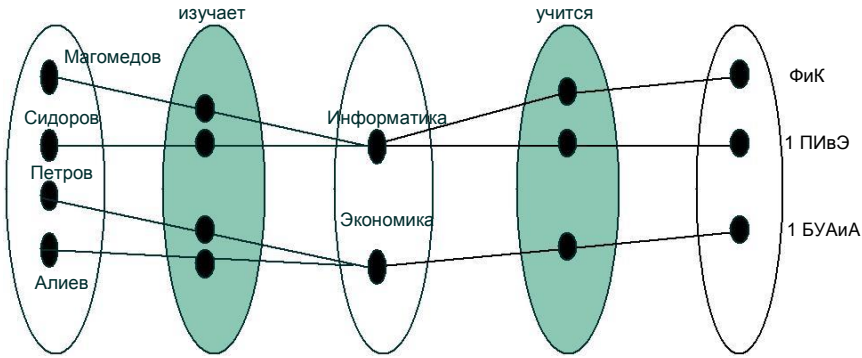


Рис. 6.14. Схема семантической сети для модели, представленной на рис. 6.11

Дефект типа разрыв (рис. 6.15):

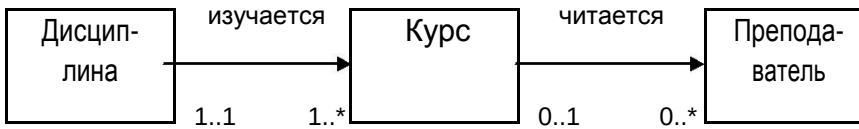


Рис. 6.15. ER-модель типа «Разрыв»

Нормализация

Одним из методов проектирования БД является метод нормальных форм, основанный на понятии зависимости между атрибутами отношений. Он относится к восходящему подходу в проектировании.

Функциональная зависимость (Functional Dependency — FD). Виды функциональных зависимостей:

- Обязательное условие идентификации атрибутов A -> B.
- Взаимооднозначная зависимость A ↔ B.
- Многозначная зависимость- A ->> B.
- Транзитивная A -> B и B -> C, A -> C.
- Полная зависимость A, C ->B и A ->B, C -> B.
- Тривиальная зависимость — такая многозначная зависимость X -» Y, для которой Y ∈X или X U Y = R, где R — рассматриваемое отношение.

□• Зависимостью проекции-соединения называется свойство де- композиции, которое означает, что при комбинировании отношений с помощью операции естественного соединения не возникают фиктивные строки.

Отношение в первой нормальной форме (сокращенно 1НФ) — это обычное отношение с двухуровневой структурой.

Отношение находится в первой нормальной форме тогда и только тогда когда значения его атрибутов атомарны.

ВЕДОМ

ПОДР	ДАТА
067	20.12.08

ФИО	ТАБ_НОМ	СУММА
Петров С.П.	67005	13 450
Сергеев И.А.	67014	15 413

В 1 НФ

ПОДР	ДАТА	ФИО	ТАБ_НОМ	СУММА
067	20.12.10	Петров С.П.	67005	13 450
067	20.12.10	Сергеев И.А.	67014	15 413

Отношение находится во второй нормальной форме тогда и только тогда когда он находится в первой нормальной форме и не содержит неполных функциональных зависимостей.

ПОДР(A)	ДАТА(B)	ФИО(C)	ТАБ_НОМ(D)	СУММА(S)
067	20.12.08	Петров С.П.	67005	13 450
067	20.12.08	Сергеев И.А.	67014	15 413

Находим минимальное покрытие

ADB’+ADC’+ADS’

При этом имеем неполную функциональную зависимость, по- этому:

DC’+ DA’+ DBS’ имеем три отношения:

ПОДР(A)	ДАТА(B)	ТАБ_НОМ(D)	СУММА(S)
067	20.12.08	67005	13 450
067	20.12.08	67014	15 413

ФИО(C)	ТАБ_НОМ(D)
Петров С.П.	67005
Сергеев И.А.	67014

ТАБ_НОМ(D)	ПОДР(A)
67005	067
670014	067

Отношение находится в третьей нормальной форме тогда и только тогда когда оно находится во второй нормальной форме и не содержит транзитивных функциональных зависимостей.

Отношения представленные на предыдущем рисунке находятся в третьей нормальной форме.

Рассмотрим следующий пример:

Дано отношение R(ФИО,Группа,Факультет).

Функциональные зависимости:

(1) ФИО → Группа;

(2) Группа → Факультет;

(3) ФИО → Факультет.

Ключ отношения Т3 — ФИО. Зависимости (1) и (3) вместе образуют транзитивную ФЗ, поэтому отношение находится в 2НФ, но не в ЗНФ.

Отношение находится в нормальной форме Бойса — Кодда тог-да и только тогда когда оно находится в третьей нормальной форме и каждый детерминант отношения является первичным ключом.

Отношение находится в четвертой нормальной форме тогда и только тогда когда оно находится в нормальной форме Бойса — Кодда и не содержит нетривиальных функциональных зависимостей.

Отношение находится в пятой нормальной форме тогда и толь-ко тогда когда оно находится в четвертой нормальной форме и не содержит зависимостей соединения.

Задание

Нормализуйте представленное отношение:

Наименование	**Тип**	**Дата выпуска**	**Цена**	**Заказчик**	**Дата приобретения**	**Количество**	**№ заказа**
АО КЭМЗ	Акции	12.03.10	1000	Иванов	11.11.10	12	1
		12.04.10		Сидоров	11.11.10	100	2
АО КЭАЗ		04.04.08		Иванов	10.12.09	11	3
		29.03.09		Сидоров	04.03.10	12	4

Дано отношение, отражающее осмотр недвижимости для сдачи в аренду сотрудниками фирмы, сдающей эту недвижимость (табл. 6.4).

Таблица 6.4

Отношение «Осмотр недвижимости»

Код недвижимости	Адрес недвижимости	Дата осмотра	Время осмотра	Комментарии	Код сотрудника	Имя сотрудника	Номер машины, на которой производился осмотр
1	2	3	4	5	6	7	8
ЧД345	Кизляр, ул. Набережная 36	01.12.07	10.00	В хорошем состоянии	С24	Магомедов Магомед	05-6789
		02.10.09	12.00	Требуется замена сантехники в ванной	С24	Алиев Али	05-6789
ДК345	Кизляр, ул. Урицкого36	01.12.07	10.00	Требуется замена полов в прихожей	С24	Алиев Али	05-4590
		07.01.08	12.00	В хорошем состоянии	С23	Магомедов Магомед	05-6456

Рассмотрим отношение и по определению первой НФ приведем его к этому виду.

Отношение находится в первой нормальной форме тогда и только тогда, когда ни один из кортежей не содержит в своем поле более одного значения и ни одно из ключевых значений не пусто.

Изобразим полученное отношение (табл. 6.5).

Таблица 6.5

Отношение «Осмотр недвижимости в 1 НФ»

1	2	3	4	5	6	7	8
ЧД345	Кизляр, ул. Набережная, 36	01.12.07	10.00	В хорошем состоянии	С24	Магомедов Магомед	05-6789

Окончание таблицы 6.5

1	2	3	4	5	6	7	8
ЧД345	Кизляр, ул. Набережная, 36	02.10.09	12.00	Требуется замена сантехники в ванной	С24	Алиев Али	05-6789
ДК345	Кизляр, ул. Урицкого, 36	01.12.07	10.00	Требуется замена полов в прихожей	С24	Алиев Али	05-4590
ДК345	Кизляр, ул. Урицкого, 36	07.01.08	12.00	В хорошем состоянии	С23	Магомедов Магомед	05-6456

Получили отношение в 1 НФ. Переходим к следующему этапу нормализации.

Отношение находится во второй нормальной форме тогда и только тогда, когда она удовлетворяет условию 1НФ и все ее поля не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом.

Полная функциональная зависимость — это когда поле В находится в полной функциональной зависимости от составного поля А,

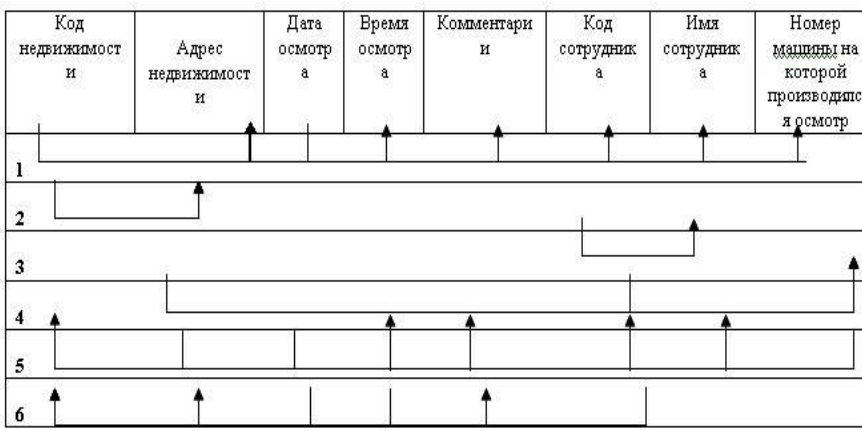


Рис. 6.16. Определение функциональных зависимостей

если оно зависит функционально от поля А и не зависит функцио-нально от любого подмножества А.

Рассмотрим процесс преобразования нашего отношения к 2НФ.

Определим функциональные зависимости для рассмотренного выше отношения (см. рис. 6.16), на котором обозначены:

1. Определен первичный ключ.
2. Частичная зависимость.

3–4. Транзитивная зависимость.

5. Потенциальный ключ.
6. Потенциальный ключ.

Для приведения отношения к 2НФ необходимо устранить частичную зависимость, в результате устранения выделяем два отношения:

Недвижимость (Код недвижимости, адрес)

Осмотр недвижимости (Код недвижимости, Дата осмотра, время осмотра, комментарии, номер сотрудника, имя сотрудника, номер машины)

Продолжаем процесс нормализации. Отношение находится в третьей нормальной форме тогда и только тогда, когда оно находится во второй нормальной форме и не содержит транзитивных зависимостей.

Для приведения отношений к 3НФ устраним транзитивную зависимость:

Код сотрудника → Имя сотрудника.

В результате получаем 3 отношения:

Недвижимость (Код недвижимости, адрес)

Сотрудники (Код сотрудника, имя сотрудника)

Осмотр недвижимости (Код недвижимости, Дата осмотра, время осмотра, комментарии, номер сотрудника, номер машины)

Проанализируем отношения для приведения их к нормальной форме Бойса — Кодда.

Отношения Сотрудники и Недвижимость находятся в НФБК, Осмотр недвижимости содержит детерминант Дата осмотра, номер сотрудника, который не является потенциальным ключом. Вследствие этого отношение осмотр недвижимости может быть подвержено аномалиям обновления.

Например, для изменения сведений о машине для сотрудника с номером С24 на 01.12.07 придется изменять содержимое сразу

двух строк. Если изменения будут произведены только в одной строке, то это приведет базу данных в противоречивое состояние. Таким образом, устраним зависимость, которая нарушает требования НФБК, получим следующие отношения:

Автомобиль сотрудника (номер сотрудника, дата осмотра, номер автомобиля)

Осмотр (номер недвижимости, дата осмотра, время осмотра, комментарии, номер сотрудника)

Сотрудники (номер сотрудника, имя сотрудника)

Недвижимость (код недвижимости, адрес)

Самостоятельно: используя рассмотренную выше нормализацию, выполните следующее задание.

Данное отношение моделирует ситуацию аренды недвижимости, предоставляемой агентством по недвижимости (у каждого объекта недвижимости есть свой владелец).

Код клиента	Имя клиента	Код недвижимости	Адрес недвижимости	Начало аренды	Окончание договора аренды	Арендная плата	Код владельца	Имя владельца
С376	Джон Кэй	ЧД345	Кизляр, ул. Набережная, 36	01.02.08	31.08.10	1000	С040	Айшат Магомедова
		КВ16	Кизляр, ул. Восточная, 36, 45	01.09.07	01.09.02	1300	С093	Айгуль Алиева
С356	Петр Петров	ЧД345	Кизляр, ул. Набережная, 36	01.07.99	30.06.07	900	С040	Айшат Магомедова
		ДК345	Кизляр, ул. Урицкого, 36	10.10.09	01.12.11	1000	С093	Айгуль Алиева
		КВ16	Кизляр, ул. Восточная, 36, 45	03.09.02	01.09.03	1500	С093	Айгуль Алиева

Привести отношение к нормальной форме Бойса — Кодда.

4 НФ

Преподаватель	Дисциплина	Форма контроля
Магомедова М. Н.	Информатика и программирование	экзамен
Магомедова М. Н.	Компьютерная графика	зачет
Абдулаева З. Л.	Информационный бизнес	зачет
Абдулаева З. Л.	Информационный менеджмент	курсовая
Абдулаева З. Л.	Информационный менеджмент	зачет
Абдулаева З. Л.	Информационный менеджмент	экзамен

Переход к реляционной модели

Инфологическая модель используется на ранних стадиях разработки проекта. Она имеет однозначную интерпретацию в отличие от некоторых предложений естественного языка. Общим языком описания был ER-модель. Для модели существует алгоритм однозначного преобразования ее в реляционную модель, что позволило создать множество инструментальных систем, поддерживающих процесс разработки информационных систем, базирующих на технологии БД. Рассмотрим правила преобразования ER-модели в реляционную:

1. Каждой сущности ставиться в соответствие отношение реляционной модели данных. Имена сущности и отношения могут быть различными, так как здесь большую роль играет ограничения самой СУБД. Например, отношение называется Товарно-транспортная- накладная, а сущность naklad (без пробелов и латинскими буквами).

2. Каждый атрибут сущности становится атрибутом соответствующего отношения. Переименование атрибутов должно происходить в соответствии с тем, какие данные содержать в кортежах данного атрибута. Каждому атрибуту устанавливается соответствующий тип данных (допустимый в данной СУБД) и обязательность или необязательность данного атрибута.

3. Первичный ключ сущности становится ключом первичного отношения.

4. В каждое отношение подчиненной сущности добавляются на- бор атрибутов главной сущности, соответствующий набору ключе-- вых атрибутов.

5. Для моделирования необязательного типа связи на физическом уровне у атрибутов соответствующих первичному ключу мож-

но установить допустимое значение NULL при обязательном типе нулевых значений не допускать.

6. Для отражения категоризации сущностей при переходе к реляционной модели возможно создать только одно отношение подтипа для данного супертипа:

а) когда отношения наследуют только идентификатор сущности;

б) когда отношения наследуют все атрибуты сущности.

7. Изменить связи многие ко многим на один ко многим. Это де- лается путем добавления связующего отношения.

Вопросы для самоконтроля:

1. Перечислите виды функциональных зависимостей
2. Для чего необходимо инфологическое проектирование
3. Перечислите основные этапы построения инфологической модели.
4. Какие признаки имеют нормальные формы.
5. Перечислите этапы перехода от инфологической модели к ре- ляционной.

Самостоятельно:

Задание

Используя этапы, предшествующие этапу концептуального проектирования, опишите следующие ситуации:

1. Директор театра решил создать собственную базу данных для учета театральных постановок и правильного подбора артистического состава.

2. Федерация по шахматам решила создать свою базу данных, в которой будут содержаться все сведения обо всех партиях, сыгранных гроссмейстерами мира на официальных турнирах за определенный период.

3. Представьте, что вы наблюдаете за работой театра. Опишите как вы себе это представляете.

Тема 7

ФИЗИЧЕСКИЕ МОДЕЛИ

1. Основные характеристики физических моделей.
2. Методы доступа.
3. Классификация файлов.
4. Структура файлов в SQL Server.
5. Хеширование.

Показатели для оценки эффективности БД:

- производительность выполнения транзакций;
- время ответа; - дисковая память.

Физические модели данных описывают, как данные хранятся в компьютере, представляя информацию о структуре записей, их упорядоченности и существующих методах доступа.

Рис. 7.1. Классификация физических моделей

Страница = физическая запись.

Физическая запись = логическая запись.

Порядок хранения записей зависит от организации файла.

Организация файла — физическое распределение данных файла по записям и страницам на вторичном устройстве хранения.

Метод доступа — действия, выполняемые при сохранении или извлечении записей из файла Метод доступа может быть:

последовательным для неупорядоченных файлов, прямым — для упорядоченных файлов и хешированных файлов.

Применительно к иерархической модели данных основные методы получили свои названия и аббревиатуры:

1) иерархический последовательный метод доступа (Hierarchical Sequential Access Method — HSAM);
2) иерархический прямой метод (Hierarchical Direct Access Method — HDAM);
3) иерархический индексно-последовательный метод (Hierarchical- Indexed Sequential Access Method — HISAM);
4) иерархический индексно-произвольный метод (Hierarchical Indexed Direct Access Method — HIDAM).

Хеширование — метод доступа, обеспечивающий прямую адре-сацию данных путем преобразования значений ключа в относи-тельный или абсолютный физический адрес.

Для вычисления адреса страницы, на которой должна находиться запись, используют хэш-функцию (hash function).

Методы создания хэш-функций:

- свертка; - остаток

от деления;

- значение хэш-функции соответствует адресу страницы (сегменту bucket) с несколькими ячейками (слотами);
- если один и тот же адрес генерируется для нескольких записей, то возникают конфликты (collisions), а подобные записи становятся синонимами.

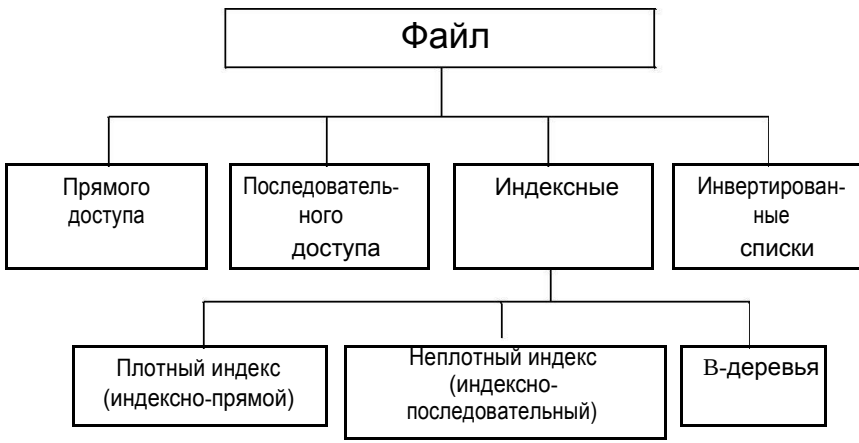


Рис. 7.2. Классификация файлов

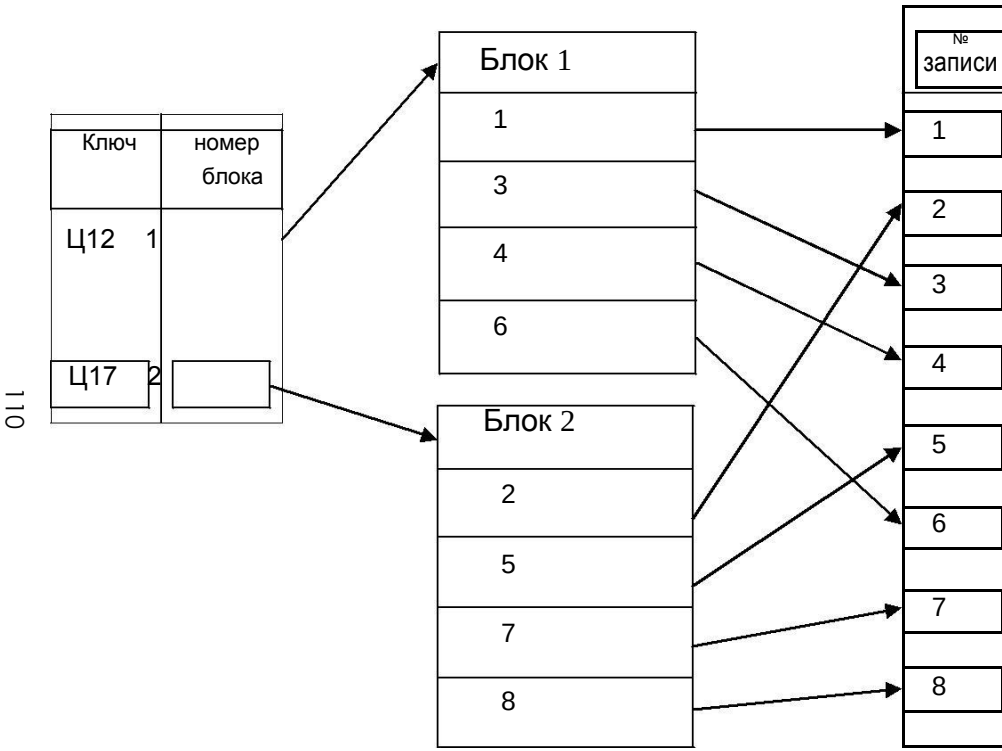


Рис. 7.3. Инвертированный список

Для устранения синонимом используют следующие методы:

− открытая адресация; − несвязанная

область переполнения;

− связанная область переполнения;

− многократное хэширование.

Файлы прямого и последовательного доступа упорядочены по ключу и зависят от типа носителя, если это магнитная лента, то файл последовательного доступа.

Индексные файлы имеют более сложную структуру: двухуровневую и т. д.

Инвертированные списки используются для доступа по вторичным ключам. Они имеют трехуровневую структуру:

1. Часть файла с упорядочением по вторичным ключам.

2. Цепочка блоков с номерами записей соответствующим вторичным ключам.

3. Основной файл (см. рис. 7.3).

B-деревья:

− используются при увеличении индексного файла для ускорения доступа;

− на каждой странице могут храниться две записи.

Пример индекса со структурой B-дерева (рис. 7.4).

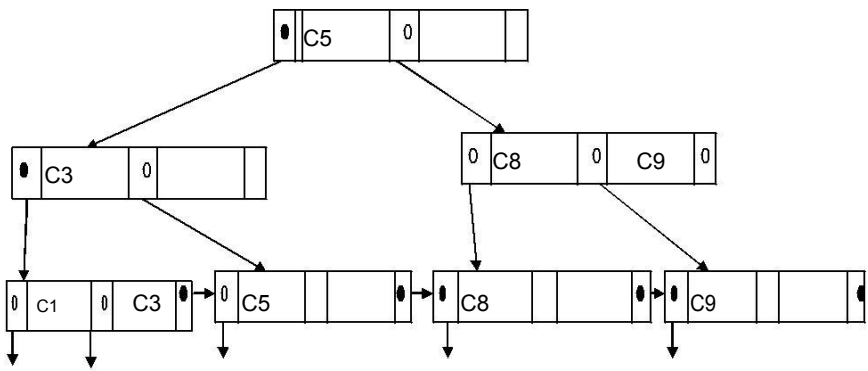


Рис. 7.4. Структура B-дерева

В иерархической модели имеются понятия глубина дерева, арность модели (степень узла), верхний узел имеет адрес 0.

Структура файла в SQL SERVER имеет странично-сегментную организацию (рис. 7.5).

96б

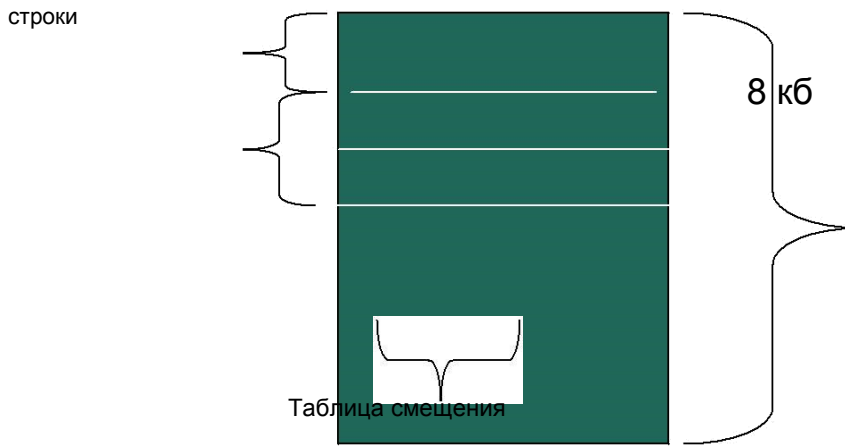


Рис. 7.5. Физическая структура БД в SQL Server

Экстент — единица выделения памяти для таблиц и индексов (8 страниц). Строки не содержат данных типа text, ntext, image.

Группы файлов в SQL Server:

- основной mdf;
- дополнительный ndf;
- файл журнала ldf.

Файл — это линейная последовательность записей, поэтому в файле можно определить последнюю, первую запись, последующую запись. Для определения адреса записи в файле прямого доступа с постоянной длиной записей необходимо знать длину записи, номер записи, базовый адрес.

Для определения номера записи в иерархической модели необходимо знать ее арность, номер узла, уровень.

Самостоятельно:

Объясните, что означают данные обозначения в функции, вычисляющей адрес размещения записи для файлов с постоянной длиной

$$BA + (K - 1) * LZ + 1.$$

Прочитайте задачи, объясните решение:

Задача 1

Определить количество индексных записей в одном блоке файла с неплотным индексом, если длина блока 1024 байта, а индексной записи 14 байт:

1024 / 14 = 73.

Задача 2

Для хранения всех записей файла требуется 12 500 блоков. Зная количество индексных записей в блоке, определите количество индексных блоков, необходимых для хранения 73 индексных записей:

125 000 / 73 = 172 блока.

Задача 3

В иерархической 5-арной иерархической структуре определить номер 27-го узла на 4-м уровне:

$1 + 5 + 5^2 + 5^3$ 27 = 183.

Решите самостоятельно:

В 5-арной иерархической структуре найдите номер записи БД узла-родителя 27-го узла на 4-и уровне.

Определить количество индексных записей в одном блоке файла с неплотным индексом, если длина блока 1000 байта, а индексной записи 20 байт.

В иерархической 4-арной иерархической структуре определить номер 4-го узла на 3-м уровне.

Вопросы для самоконтроля:

1. Какие организации файлов различают?
2. Как классифицируются файлы?
3. Перечислите основные элементы в структуре файла SQL Server.

Термины:

Экстент
Запись
Файл Организация
файла Метод
доступа

Тема 8

ОРГАНИЗАЦИЯ ПРОЦЕССОВ ОБРАБОТКИ ДАННЫХ

Транзакция (*англ.* transaction) — в информатике, группа последовательных операций (совокупность запросов), которая представляет собой логическую единицу работы с данными (это те операции, что в примере между BEGIN TRAN и COMMIT TRAN / ROLLBACK- TRAN). Транзакция может быть выполнена либо целиком и успешно (COMMIT TRAN), соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще (ROLLBACK TRAN) и тогда она не должна произвести никакого эффекта. Транзакции обрабатываются транзакционными системами, в процессе работы которых создается история транзакций.

Рассмотрим обработку транзакций на примере систему вуза. Студенты одного филиала проходят тестирование онлайн с использованием тестовой базы вуза. Если студент прошел тестирование и нажимает завершить, то результат тестирования сохраняется, т. е. вызывается оператор commit. А можно ROLLBACKом от-катить изменения (это может произойти если отключился компьютер) и результат не сохранится.

Ярким примером транзакции является банковская. Перечисление денег со счета на счет состоит из нескольких запросов: снятие с одного счета, зачисление на другой. Если второй запрос не выполняется, то происходит откат.

Свойства транзакции — элементарность, последовательность, изолированность, устойчивость.

Эти 4 принципа называют ACID (atomicity — элементарность, consistency — последовательность, isolation — изолированность, durability — устойчивость).

Элементарность: этот принцип требует, чтобы транзакция бы-ла либо выполнена, либо отменена (аварийно прекращена). В лю-бом из этих двух случаев данные должны быть правильно сохране-ны в конечном состоянии, либо в начальном в случае аварии.

Последовательность: этот принцип основан на логичности. То есть транзакция должна быть максимально простой и давать предсказуемый результат. Например, при поступлении денег их записывают в кредит всегда. Реализация этого принципа ложиться на разработчика системы.

Изолированность: при рассмотрении транзакции возникает проблема промежуточного состояния. При переводе товара скажем с одного склада на другой он некоторое время как бы висит в воздухе, пока не будет завершена транзакция. Это время может быть маленьким, а может и большим. Это промежуток и называют переходным состоянием. Если система однопользовательская то проблем нет. А что, если несколько транзакций пытаются изменить одни и те же данные? Так вот, они должны быть изолированы друг от друга. К сожалению, тут возникает много вопросов, например о версиях данных и так далее. Проверка версий данных ложиться на разработчика.

Устойчивость: данный принцип заключается в том, что в случае успешного выполнения транзакции в базе данных сохраняются внесенные данные, которые отменить невозможно.

Вопросы для самоконтроля:

1. В чем разница между транзакцией и запросом?
2. В каком случае происходит откат транзакции?
3. В чем заключается сущность ACID?

Термины:

Транзакция
Запрос Свойства транзакции

Тема 9

ПРИНЦИПЫ ПОДДЕРЖКИ ЦЕЛОСТНОСТИ

1. Понятие целостности. Классификация ограничений целостности.
2. Языки манипулирования данными.
3. Причины, вызывающие нарушение ограничений целостности.
4. Способы задания ограничений целостности в современных СУБД. Ограничения целостности в стандартах SQL.

Целостность-неприкосновенность — понимается как правильность данных в любой момент времени. Целостность это состояние базы данных в непротиворечивых состояниях. Целостность каждой модели определяется ограничениями. Для реляционной модели:

- по сущностям;
- по ссылкам;
- доменная;
- языковая.

Ограничения по сущностям трактуются как то, что каждая сущность содержит в себе атрибуты, минимальный набор которых содержит:

- атрибуты, отражающие идентификаторы объекта.
- атрибуты, отражающие признак времени.
- атрибуты, отображающие некоторое количественное свойство объекта.

Второе требование называется требованием ссылочной целостности; оно возникло потому, что при соблюдении нормализованности таблиц сложные сущности реального мира неизбежно представляются в РБД в виде нескольких строк нескольких таблиц. Так реализуются бинарные связи типа «один к одному» и «один ко многим» («многие к одному»); причем в первом случае (симметричная связь) выбор из двух таблиц таблицы с внешним ключом определяется только семантикой соответствующих им сущностей. В случае же связи типа «один ко многим» внешний ключ всегда находится в той таблице, которая может иметь много строк, связан-

ных с одной строкой другой таблицы. Наконец, связь типа «многие ко многим» реализуется в реляционной модели за счет введения специальной таблицы, превращающей связь в отдельную сущность. Каждая строка этой таблицы соответствует одной связи «один к одному», а атрибуты таблицы делятся на два внешних ключа, которые связывают ее с двумя исходными таблицами связью «один ко многим».

Таким образом, второе требование целостной части реляцион-ной модели (ограничение ссылочной целостности) состоит в том, что либо значение внешнего ключа (foreign key) должно быть неопределенным (null), либо для каждого значения внешнего ключа в соответствующей таблице должна найтись строка с таким же значением первичного ключа (primary key). «Соответствующая» таблица может совпадать с таблицей внешнего ключа, если моделируется связь между однотипными сущностями.

Ограничения всех типов автоматически поддерживаются СУБД которая при вставке в таблицу новой строки (или при ее измене-нии) проверяет пользовательские условия, отсутствие в данной таблице строки с тем же значением первичного ключа, а также на-личие в других таблицах (на которые делаются ссылки) строк с первичным ключом, равным внешним ключам вставляемой стро-ки (если значения внешних ключей определены). Кроме того, при удалении строк из таблицы целостность по ссылкам поддержива-ется одним из трех существующих способов (в развитых СУБД выбор способа делается пользователем при задании внешнего ключа). Первый способ запрещает производить удаление строки, на которую существуют ссылки (т. е. сначала нужно либо удалить ссылающиеся строки, либо изменить значения их внешнего клю-ча). При втором (наименее распространенном) способе при удале-нии строки, на которую имеются ссылки, во всех ссылающихся строках значение внешнего ключа автоматически становится не- определенным. Наконец, третий способ (каскадное удаление) со-стоит в том, что при удалении такой строки автоматически удаля-
ются все ссылающиеся строки.

Манипулирование реляционными данными

В манипуляционной части реляционной модели утверждаются два фундаментальных механизма работы с реляционными БД — реляционная алгебра и реляционное исчисление (причем доказывается, что они эквивалентны друг другу). Первый механизм бази-

руется в основном на классической теории множеств, а второй — на классическом логическом аппарате исчисления предикатов первого порядка. Их назначением является обеспечение меры реляционности любого конкретного языка РБД: язык называется реляционным, если он обладает не меньшей мощностью (имеет не меньшее число соответствующих операций), чем реляционная алгебра или реляционное исчисление.

Для поддержания целостности объектно-ориентированный подход предлагает использовать следующие средства:

- автоматическое поддержание отношений наследования;
- возможность объявить некоторые поля данных и методы объекта как «скрытые», не видимые для других объектов; такие поля и методы используются только методами самого объекта;
- создание процедур контроля целостности внутри объекта.

Для поддержания целостности в сетевой модели из функционального характера реализуемых в МД связей следует второе внутреннее ограничение целостности — экземпляр записи может быть членом только одного экземпляра среди всех экземпляров набора одного типа (он может входить в состав двух и более экземпляров наборов, но различных типов).

В модели КОДАСИЛ основным внутренним ограничением целостности является функциональность связей, т. е. с помощью наборов можно реализовать непосредственно связи типа 1:1, 1:M, M:1.

В иерархической модели данных действуют более жесткие внутренние ограничения на представление связей между сущностями, чем в сетевой модели. Основные внутренние ограничения иерархической модели данных:

1) все типы связей функциональные (1:1, 1:M, M:1);

2) структура связей древовидная.

Из приведенного ниже примера выбрать ограничения целостности и распределить их в тетради в 4 столбца (в соответствии с следующими типами):

1) ограничения по сущностям;

2) по ссылкам;

3) доменная целостность;

4) пользовательские ограничения.

База данных книжного магазина, реализующего редкие книги, представлена следующими таблицами: Сотрудники, Книги, Зака-

зы, Должности сотрудников, Покупатели, Формы оплаты, Статус заказа, Заказанные книги.

В таблице Сотрудники первичным ключом является Номер сотрудника.

Таблица сотрудники соединена с таблицей Должности по полю Код должности.

Возраст сотрудника не может быть старше 50 и меньше 20 лет.

В таблице Сотрудники поле Имя не может содержать нулевого значения.

В таблице Книги поле Год издания не может быть меньше 1600 и больше 2099.

Каждая книга учитывается по индивидуальному номеру (даже если она одного и того же года издания, названия, автора).

Идентификатор книги может состоять из символов и цифр.

Самостоятельно: определите ограничения целостности для примеров, рассмотренных в предыдущих лекциях.

Термины:

Целостность Классификация ограничений целостности

Тема 10

РАСПРЕДЕЛЕННЫЕ БД

1. Понятие распределенных БД.
2. Удаленный запрос, распределенная транзакция.
3. Функции приложения и их распределение в различных моделях. Двух- и трехуровневые системы клиент-сервер. Модели транзакций. Журнал транзакций. Проблемы параллельного выполнения транзакций. Блокировки, виды блокировок. Технология тиражирования.

Распределенная база данных — это база данных, которая распределена по различным серверам имеющим общий сервер, базы данных на различных серверах могут состоять из пересекающихся частей.

Распределенные базы данных должны проектироваться с учетом использования распределенных транзакций и удаленных запросов.

• Удаленный запрос — это запрос, который выполняется с использованием модемной связи.

• Возможность реализации удаленной транзакции — это обработка одной транзакции, состоящей из множества SQL-запросов, на одном удаленном узле.

• Поддержка распределенной транзакции допускает обработку транзакции, состоящей из нескольких SQL-запросов, которые выполняются на нескольких узлах сети (удаленных или локальных), но каждый запрос в этом случае обрабатывается только на одном узле, т. е. запросы не являются распределенными. При обработке одной распределенной транзакции разные локальные запросы мо- гут обрабатываться в разных узлах сети.

Пример удаленной транзакции:

```
UPDATE
sales.parts@east.compworld SET...
WHERE ...
UPDATE
sales.parts@east.compworld SET ...
```

WHERE ...
UPDATE sales.parts@east.compworld
SET ...
WHERE ...
COMMIT
Пример распределенной транзакции:
UPDATE sales.parts
SET ...
WHERE ...
UPDATE sales.parts@east.compworld
SET ...
WHERE ...
UPDATE sales.parts@east.compworld
SET ...
WHERE ...
COMMIT

Основным назначением журнала транзакций (transaction log) является протоколирование всех транзакций и сделанных ими изменений. Таким образом, поддерживается целостность данных.

Блокировки

Блокировка — это объект, с помощью которого программы показывают зависимость пользователя от ресурса. Блокировкой называется временное ограничение на выполнение некоторых операций обработки данных.

Запрещают пользователям читать данные, изменяемые другими пользователями, не дают изменять одни и те же данные одновременно несколькими пользователями. Обращение нескольких пользователей к БД называется параллельным выполнением. При этом возникают проблемы следующего характера:

Потерянные или скрытые обновления

Потерянные обновления появляются при выборе одной строки двумя или более транзакциями, которые, которые обновляют эту строку на основе ее первоначального значения. Например, два редактора одновременно скопировали документ, внесли свои изменения, причем редактор, который вносил свои изменения позже, внес их поверх тех, которые внес предыдущий редактор.

Зависимость от незафиксированных данных (грязное чтение)

В документ были внесены изменения, они были зафиксированы и розданы сотрудникам. В это время тот, кто их внес, решает ото-

звать изменения, но сотрудники уже извещены о неправильных изменениях. Распространенный документ содержит несуществующие поправки.

Несогласованный анализ

Чтение фантомов

Документ прочитан, но в этот момент из него удаляются или добавляются строки, которые не проверяются, так как документ проверен.

Различают 2 вида параллельного выполнения: оптимистическое и пессимистическое. При оптимистическом допускается выполнения явных транзакций без блокирования ресурсов.

Проблемы, связанные с параллельным выполнением, разреша-ют, используя уровни изоляции. Среди них выделяют следующие:

Уровень изоляции чтения	**Чтение грязное**	**Неповторяемость фантомов**	**Чтение**
Неподтверждаемое	Да	Да	Да
Подтверждаемое	нет	Да	Да
Повторяемое	нет	нет	Да
Упорядочение	нет	нет	Нет

SQL реализует блокирование автоматически, в приложениях это настраивается следующими способами:

□• Задавая обработку взаимоблокировок и установку приоритетов при взаимоблокировках.

□• Устанавливая уровень изоляции транзакций.

□• Задавая обработку тайм-аутов и определяя их продолжитель-- ность.

□• Назначая блокировки на уровне таблицы с операторами SELECT, INSERT, UPDATE, DELETE.

Однако такая блокировка создает новые проблемы — задержку выполнений транзакций из-за блокировок.

Типы конфликтов между двумя параллельными транзакциями:

□• W-W — транзакция 2 пытается изменять объект, измененный незакончившейся транзакцией 1;

□• R-W — транзакция 2 пытается изменять объект, прочитанный незакончившейся транзакцией 1;

□• W-R — транзакция 2 пытается читать объект, измененный незакончившейся транзакцией 1.

Вопросы для самоконтроля:

1. Виды параллельного выполнения транзакций.
2. Для чего нужны блокировки?
3. Виды блокировок.
4. Типы конфликтов между параллельными транзакциями.

Термины:

Удаленная транзакция
Распределенная база
данных Блокировка

Тема 11

ЯЗЫК SQL

Основными объектами являются базы данных, таблицы, представления, диаграммы, процедуры, ограничения, правила. Обработка данных в SQL SERVER ведется с помощью операторов языка TRANSACT SQL.

Операторы TRANSACT SQL делятся на три группы:

- операторы манипулирования;
- операторы создания;
- операторы управления.

Основным оператором является SELECT. Синтаксис оператора SELECT:

SELECT список выбора
INTO имя новой таблицы
FROM список таблиц
WHERE условия поиска
GROUP BY группировка по списку HAVING условия поиска
ORDER BY поле для сортировки ASC|DESC

Информация, хранящая в базе данных SQL SERVER, может иметь следующие типы данных:

Категория типа данных	**Описание**	**Базовый тип данных**	**Описание**
Двоичные	Двоичные данные, хранящие строки бит	Binary	8 кб
		varbinary	8 кб
		image	8 кб
символьные	Символьные данные	Varchar	Различная длина до 8 кб
		Char	Фиксированная длина 8 кб
		text	8 кб
Дата и время	Время и дата	datetime	01.01.1753-31.12.9999 (8 б)
		smalldatetime	01.01.1900-06.06.2079 (4 б)
десятичные	Данные, которые сохраняют-	Decimal	До 38 знаков

Окончание таблицы

Категория типа данных	Описание	Базовый тип данных	Описание
	ся до наименьшего значащего разряда	Numeric	
С плавающей запятой	Приближенные, с плавающей запятой	Float	
		real	
целочисленные	целочисленные	bigint	8 байт
		Int	4 байта
		Smallint	2 байта
		tinyint	До 255, 2 байта
денежные	Финансовые данные	Money	8 байт
		smallmoney	4 байта
специальные		bit	

Создания баз данных, таблиц и других объектов осуществляется с помощью оператора CREATE.

Операторы управления связаны с назначением и отзывом разрешений на пользование объектами БД.

Группы операторов в SQL SERVER объединяются в хранимые процедуры и триггеры.

Хранимая процедура — это группа операторов, которая компилируется один раз и затем может использоваться многократно.

Триггер — это специальный класс хранимой процедуры. Триггер запускается автоматически системой SQL Server при модифицировании какой-либо таблицы одним из трех операторов: INSERT, UPDATE, DELETE. Они могут быть простыми и сложными.

Триггер можно создать только по определенным пользователем таблицам или представлениям.

Триггеры нужны для поддержки семантической целостности данных, сравнения данных до и после модификации, создания сообщения об ошибке. При этом триггер не должен возвращать результат.

Вопросы для самоконтроля:

1. Назовите основные объекты базы данных SQL.
2. Перечислите виды операторов Transact SQL.

3. Каково назначение триггеров.
4. Перечислите основные типы данных SQL Server.

Термины:

Триггер Хранимая процедура

Тема 12

БЕЗОПАСНОСТЬ ДАННЫХ

1. Понятие безопасности данных.
2. Способы обеспечения безопасности в современных СУБД.

Потенциальные опасности:
- Похищение и фальсификация данных.
- Утрата конфиденциальности.
- Нарушение неприкосновенности личных данных.
- Утрата целостности.
- Потеря доступности.

Безопасность баз данных решается на двух уровнях:
- СУБД.
- Операционной системы.

Система безопасности в SQL Server 2000:
1. Физическая система безопасности.
2. Безопасность сетевого протокола.
3. Доменная безопасность.
4. Безопасность локального компьютера.
5. Аутентификация. Авторизация, шифрование.
6. Безопасность приложений.

Физическая безопасность

Подразумевает защиту сервера и сети путем расположения их на территориях с ограниченным доступом. Она включает в себя совокупность систем охранного мониторинга, системы принятия управляющих решений, средств обеспечения безопасности, направленных на защиту объектов, связанных с базой данных. Совокупность этих объектов представлена на рис. 12.1.

Данные, передаваемые с клиента, могут быть зашифрованы с помощью протокола SSL. Утилита CLIENT NETWORK UTILITY позволяет выполнить шифрование с помощью включения шифрования пакетов для всех активных протоколов.

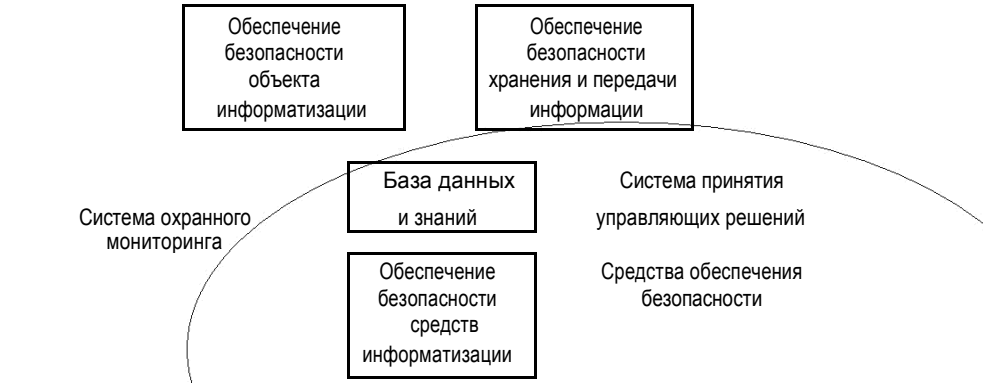


Рис. 12.1. Управляющий контур организации физической безопасности

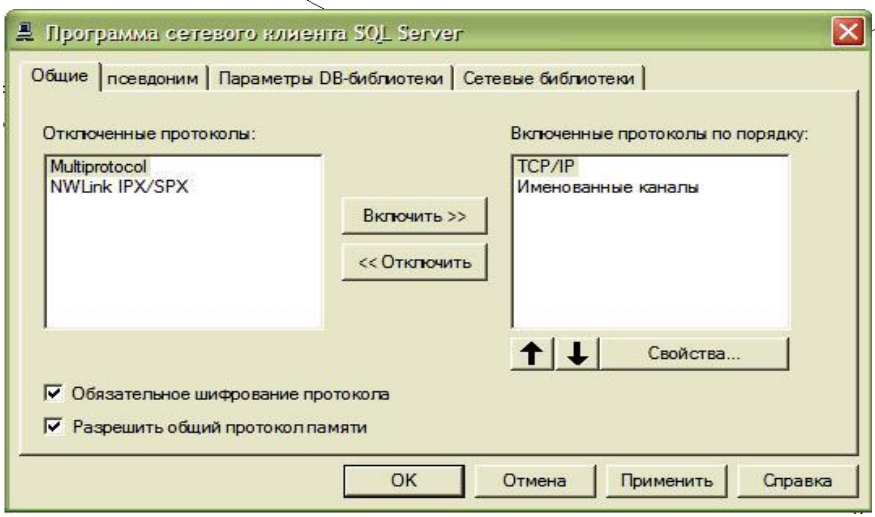


Рис. 12.2. Диалоговое окно сетевого клиента в SQL Server

Доменная безопасность

Если компьютер, на котором работает SQL Server, является членом домена WINDOWS, то можно проверить пользователя с помощью его идентификатора безопасности.

Безопасность локального компьютера

Файловая система NTFS поддерживает дополнительные функции безопасности. Поэтому для работы необходимы ОС Windows NT, Windows 2000.

Аутентификация, авторизация, аудит, шифрование:

□• Подключение к SQL SERVER.

□• Открывается доступ к БД и ее объектам.

□• Действия отслеживаются.

□• Некоторые объекты шифруются для защиты содержимого.

□• Аутентификация.

□• Для управления существуют процедуры sp_grantlogin, sp_revokelogin управляют аутентификацией учетных записей WINDOWS.

Следующие операторы добавляют учетные записи WINDOWS и предоставляют им доступ:

□• EXEC sp_grantlogin @LOGINAME='DOMAIN01\user01'.

□• EXEC sp_grantlogin @LOGINAME='SQLSERVER01\user01'.

□• EXEC sp_grantlogin @LOGINAME='BUILTIN\POWERuserS'.

Для временного запрета каких-либо действий членам группы POWERuserS:

□• EXEC sp_denylogin @LOGINAME='BUILTIN\POWERuserS'.

Для удаления учетной записи WINDOWS из SQLSERVER используется следующий оператор:

□• EXEC sp_revokelogin @LOGINAME='SQLSERVER01\user01'. Аутентификация:

□• Sp_addlogin.

□• @logname='user01'.

□• @password='pasword01'.

□• создаем идентификатор и предоставляем ему доступ с паролем.

Некоторые учетные имена (sa) создаются вовремя установки. Этому идентификатору соответствует особое учетное имя dbo См. рис. 12.3).

Авторизация

□• SQL SERVER 2000 поддерживает группы WINDOWS и роли SQL SERVER.

□• По умолчанию созданные пользователи, группы, роли являются членами роли PUBLIC.

Авторизация:

□• USE PUBS.

Рис. 12.3. Задание разрешений на работу с объектами созданному учетному имени

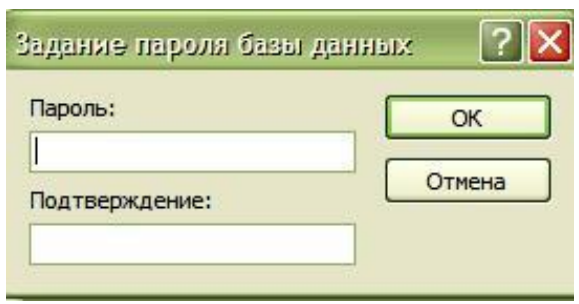


Рис. 12.4. Окно для ввода пароля

□• GRANT UPDATE, SELECT.
□• ON DBO.SALES (ORD_DATE, ORD_NUM).
□• TO USER01.

Защита в MS Access

□• Установка пароля.
□• Защита на уровне пользователя.

Установка пароля (рис. 12.4):

□ открываем БД в режиме монопольного доступа;
□ выбираем команду: Сервис-Защита — Задать пароль базе данных.

Защита на уровне пользователя осуществляется в окне Пользователи и группы (рис. 12.5).

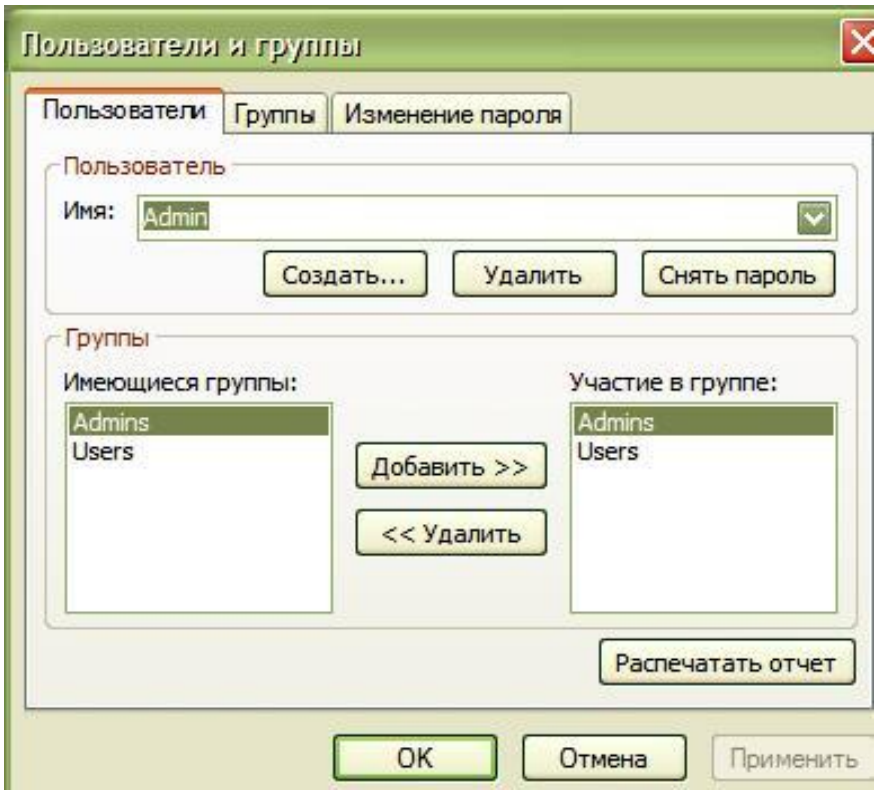


Рис. 12.5. Задание пользователя и определение для него группы в MS Access

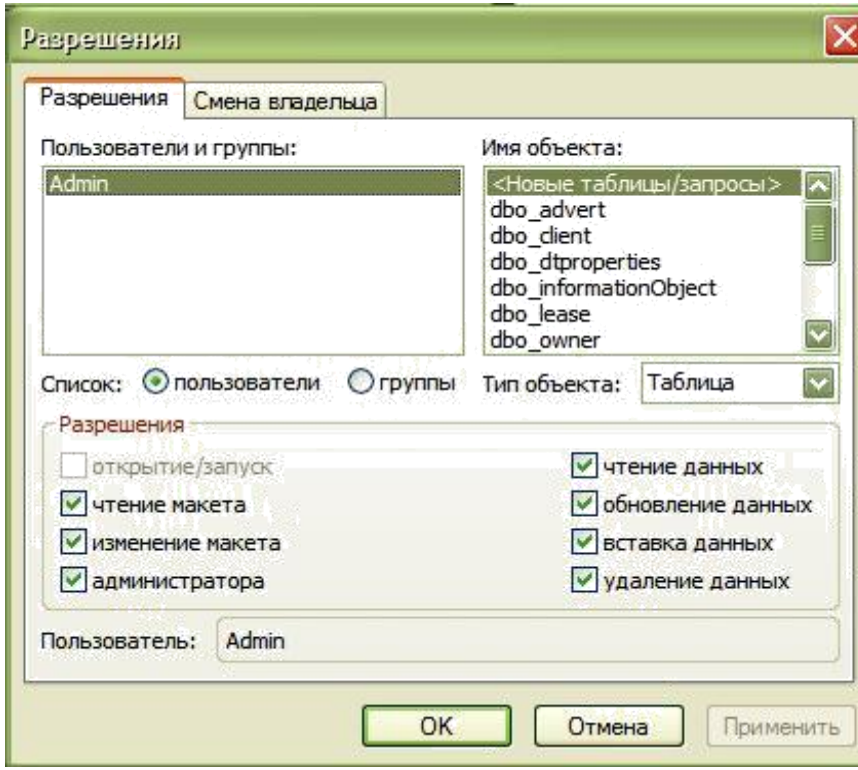


Рис. 12.6. Диалоговое окно для установления разрешений

Тема 13

ФОРМАЛИЗАЦИЯ ЗНАНИЙ В ИС

Знания — это совокупность сведений о сущностях (объектах, предметах) реального мира, их свойствах и отношениях между ними в определенной предметной области. Иными словами, зна-ния — это выявленные закономерности предметной области (ПрО) (принципы, связи, законы), позволяющие решать задачи в этой области. С точки зрения ИИ знания можно определить как формализованную информацию, на которую ссылаются в процессе логического вывода.

В этом случае, под ПрО понимается область человеческих знаний, в терминах которой формулируются задачи и в рамках которой они решаются. То есть ПрО представляется описанием части реального мира, которое в силу своей приближенности рассматривается как ее информационная модель.

Проблемная область — это содержательное описание в терминах ПрО проблемы совместно с комплексом условий, факторов и обстоятельств, вызвавших ее возникновение.

В исследованиях по ИИ можно выделить два основных направления: программно-прагматическое («не имеет значения, как устроено «мыслящее» устройство, главное, чтобы на заданные входные воздействия оно реагировало, как человеческий мозг») и бионическое («единственный объект, способный мыслить — это человеческий мозг, поэтому любое «мыслящее» устройство должно каким-то образом воспроизводить его структуру»).

Ярким представителем программно-прагматического направления можно считать экспертные системы — это сложные программные комплексы, аккумулирующие знания специалистов-экспертов для обеспечения высокоэффективного решения неформализованных задач в узкой предметной области.

Приведенные выше понятия могут рассматриваться как пример системного изложения аксиоматических основ дисциплины в авторском изложении.

Общепризнанного определения знания, как и определения искусственного интеллекта, не существует. Известные трактовки

этого понятия отражают его различные аспекты, поэтому приведем несколько определений.

Наиболее общее определение трактует знание как всю совокупность данных (информации), необходимую для решения задачи. В этом определении подчеркивается, что данные в привычном понимании также являются знаниями. Однако знания в информационном плане не ограничиваются рамками данных. В полном объеме информация, содержащаяся в знаниях, должна включать сведения о: системе понятий предметной области, в которой решаются задачи; системе понятий формальных моделей, на основе которых решаются задачи; соответствии систем понятий, упомянутых выше; методах решения задачи; текущем состоянии предметной области.

Из перечисленных компонентов только последний в явном виде соответствует понятию «данные».

В целом обо всей приведенной выше информации иногда гово-рят, что она составляет проблемную область решаемой задачи.

Несмотря на сложности формулировки определения знания считается общепризнанным, что знания имеют ряд свойств, позволяющих отличать их от данных: внутреннюю интерпретируемость; внутреннюю (рекурсивную) структурированность; внешнюю взаимосвязь единиц; шкалирование; погружение в пространство с семантической метрикой; активность.

Если данные обладают этими свойствами, можно говорить о перерастании данных в знания.

Внутренняя интерпретируемость означает наличие в памяти ЭВМ сведений не только о значении, но и о наименовании информационной единицы. Следует отметить, что это свойство присуще некоторым моделям представления данных, например реляционной.

Внутренняя (рекурсивная) структурированность отражает вложенность одних информационных единиц в другие или в самих себя. Она предусматривает установку отношений принадлежности элементов к классу, родовидовые отношения типа «часть-целое» и т. п. В целом внутренняя структурированность характеризует структуру знания.

Внешняя взаимосвязь единиц определяет, с какой информационной единицей имеет связь данная информационная единица и какова эта связь. С помощью этого свойства устанавливается связь различных отношений, отражающих семантику и прагматику свя-

зей понятий, а также отношений, отражающих смысл системы в целом.

Отдельные информационные единицы не могут описывать динамические ситуации, когда некоторые факты, содержащиеся в структуре одной единицы, вступают в ситуативную связь с факта-ми или явлениями, описанными в структуре другой единицы. Для описания таких связей используются специальные информацион-ные элементы, в которых указываются имена взаимосвязанных информационных единиц и имена существующих отношений.

Шкалирование означает использование шкал, предназначенных для фиксации соотношения различных величин. Прежде всего шкалирование необходимо для фиксации соотношений качественной информации.

Погружение в пространство с семантической метрикой используется для задания меры близости информационных единиц. Это свойство можно проиллюстрировать на следующем примере. Пусть разработано несколько вариантов построения системы свя-зи. Требуется определить, насколько структура существующей си-стемы связи близка к одному из имеющихся вариантов. Для этого можно использовать метод матриц сходства — различия, в соот-ветствии с которым матрицы заполняются оценками попарного сходства и различия элементов структур реальной системы связи и рассматриваемого варианта. В качестве оценок обычно выступа-ют числа в диапазоне от –1 до +1 при условии, что –1 характеризу-ет полное различие, а +1 — полное сходство.

На основании матриц сходства — различия определяется сте-пень сходства текущей ситуации с заранее заданной (планируемой).

Активность знаний выражается в возможности вызова той или иной процедуры в зависимости от структуры, сложившейся между информационными единицами.

Активность знаний обусловлена тем, что в отличие от обычных программ, в которых процедуры играют роль активаторов дан-ных, в интеллектуальных системах определенная структура дан-ных активизирует выполнение той или иной процедуры. Прак- тически это осуществляется включением в состав информацион-ной единицы элемента, содержащего имя процедуры, или пред-ставлением знаний в виде правил, причем правила записываются в следующем виде: «Если произошли события A1 и A2 и ... и A*N*, то необходимо выполнить процедуру B». Использование правил зна-

чительно упрощает объяснение того, как и почему получено то или иное заключение (вывод).

Перечисленные особенности информационных единиц определяют ту грань, за которой данные превращаются в знания, а базы данных перерастают в базы знаний (БЗ). Совокупность средств, обеспечивающих работу со знаниями, образует систему управления базой знаний (СУБЗ). В настоящее время не существует баз знаний, в которых в полной мере были бы реализованы все пять особенностей знаний.

Базы данных фиксируют экстенсиональную семантику заданной проблемной области, состояние конкретных объектов, конкретные значения параметров для определенных моментов времени и временных интервалов. База знаний определяет интенсиональную семантику моделей и содержит описание абстрактных сущностей: объектов, отношений, процессов.

Если рассматривать знания с точки зрения решения задач в некоторой предметной области, то их удобно разделить на две боль-шие категории — факты и эвристику. Первая категория указывает обычно на хорошо известные в данной предметной области обсто-ятельства, поэтому знания этой категории иногда называют тексто-выми, подчеркивая их достаточную освещенность в специальной литературе или учебниках. Вторая категория знаний основывается на собственном опыте специалиста (эксперта) в данной предмет-ной области, накопленном в результате многолетней практики.

Знания можно разделить на процедурные и декларативные. Декларативные знания — это совокупность сведений о качественных и количественных характеристиках конкретных объектов, явлений и их элементов, представленных в виде фактов и эвристик. Традиционно такие знания накапливались в виде разнообразных таблиц и справочников, а с появлением ЭВМ приобрели форму информационных массивов (файлов) и баз данных. Процедурные знания хранятся в памяти ИИС в виде описаний процедур, с помощью которых их можно получить. В виде процедурных знаний обычно описывается информация о предметной области, характеризующая способы решения задач в этой области, а также различ-ные инструкции, методики и тому подобная информация. Другими словами, процедурные знания — это методы, алгоритмы, програм-мы решения различных задач, последовательности действий (в выбранной проблемной области) — они составляют ядро баз знаний.

Таким образом, при использовании знаний происходит переход к формуле: знания + система.

Работа со знаниями, иначе называемая обработкой знаний, лежит в основе всего современного периода развития ИИ. В свою очередь обработка знаний включает в себя:

- извлечение знаний из источников (под источниками понимаются материальные средства хранения знаний, а также события и явления, но при этом считается, что человек источником не является);
- приобретение знаний от профессионалов (экспертов);
- представление знаний, т. е. их формализация, позволяющая в дальнейшем использовать знания для проведения логического вывода на ЭВМ;
- манипулирование знаниями, включающее пополнение, классификацию, обобщение знаний и вывод на знаниях;
- объяснение на знаниях, позволяющее дать ответ, как и почему проведен тот или иной вывод.

В памяти ЭВМ знания представляются в виде некоторой знако-вой системы. С понятием «знак» связываются понятия «экстенси-онал» и «интенсионал». Экстенсионал знака — это его конкретное значение или класс допустимых значений. Интенсионал знака — это его смысл, характеристика содержания. Интенсионал знака определяет содержание связанного с ним понятия.

Соответственно различают два типа знаний: экстенсиональные и интенсиональные.

Экстенсиональные знания — это набор количественных и качественных характеристик различных конкретных объектов. Они представляются перечислениями объектов предметной области, экземпляров объектов, свойств объектов. Иными словами, экстенсиональные знания — это данные, хранящиеся в базах данных.

Иногда экстенсиональные знания называются предметными, или фактографическими знаниями.

Интенсиональные знания — это совокупность основных терминов, применяемых в проблемной области, и правил над ними, позволяющих получать новые знания. Интенсиональные знания описывают абстрактные объекты, события, отношения.

Интенсиональные знания подразделяются на декларативные, процедуральные и метазнания.

Декларативные знания отражают понятия проблемной области и связи между ними. Они не содержат в явном виде описания ка-

ких-либо процедур. Иначе декларативные знания называются понятийными, или концептуальными.

Процедуральные знания описывают процедуры, т. е. указывают операции над понятиями, позволяющие получать новые понятия. В отличие от декларативных знаний они содержат в явном виде описания процедур. Примером процедуральных знаний является программа, хранящаяся в памяти ЭВМ. Иногда процедуральные знания называются алгоритмическими.

Метазнания — это знания об организации всех остальных типов знаний. Иначе они называются специальными. Метазнания содержат признаки декларативных и процедуральных знаний.

Поверхностные — знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области.

Глубинные — абстракции, аналогии, схемы, отображающие структуру и природу процессов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для прогнозирования поведения объектов [28].

Вопросы для самоконтроля:

1. На каком этапе данные становятся знаниями?
2. Что включает в себя обработка знаний?
3. Что включает в себя проблемная область?
4. Как классифицируются знания?

Термины:

Знания Проблемная область
Шкалирование
Экстенсиональные знания
Интенсиональные знания

Тема 14

МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ

1. Понятие модели предоставления знаний.
2. Продукционная модель представления знаний.
3. Модель исчисления предикатов первого порядка.
4. Фреймовая модель представления знаний.
5. Семантические сети.

Приобретение знаний осуществляется на основе четко представленной информации в виде конкретных значений данных и описывающих их понятий, представленных в виде таблиц, графиков, гистограмм, а также на основе слабо структурированной информа-ции, представленной экспертом заданной предметной области.

В качестве инженера по знаниям могут выступать: человек, программа распознавания естественного языка, экспертная система.

На этапе приобретения знаний осуществляются следующие функции:- описание (данных, ситуаций и т. д.), концептуализа-ция — описание объектов, свойств, рассуждений, формализа-ция — представление информации в заданных структурах, реали-зация осуществляется с проверкой на корректность и на непротиворечивость.

Существуют следующие модели представления знаний:

- продукционная;
- исчисление предикатов первого порядка;
- фрейм;
- семантическая сеть.

Продукционные модели в последнее время широко используются в системах представления знаний. Первоначально предложенные Постом в 1943 г., они были впервые применены в системах ИИ в 1972 г.

Продукционные модели могут быть реализованы как процедур-но, так и декларативно. Их простота и строгая форма послужили основой ряда интересных свойств, что сделало их удобным сред-ством представления знаний. Рядом исследователей отмечалось,

что использование продукционных моделей имеет уже само по се-бе особую психологическую важность, хотя они могут быть с успе-хом использованы и вне рамок психологического моделирования.

Продукционные модели — это набор, правил вида «условия — действие», где условиями являются утверждения о содержимом некой базы данных, а действия представляют собой процедуры, которые могут изменять содержимое БД.

В продукционных системах можно выделить три основные компоненты: неструктурированная или структурированная БД и некоторое число продукционных правил или просто продукций. Каждая продукция состоит из двух частей:

- условий (антецендент); в этой части определяются некоторые условия, которые должны выполняться в БД для того, чтобы были выполнены соответствующие действия;
- действий (консеквент); эта часть содержит описание действий, которые должны быть совершены над БД в случае выполнения соответствующих условий.

В простейших продукционных системах они только определяют, какие элементы следует добавить (или иногда удалить) в БД.

Интерпретатор, который последовательно определяет, какие продукции могут быть активированы в зависимости от условий, в них содержащихся; выбирает одно из применимых в данной ситуации правил продукций; выполняет действие из вы-бранной процедуры.

Продукционные модели в основном находят применение в качестве решателей или механизмов выводов. В БД системы хранятся известные факты о некоторой предметной области. Продукции содержат специфические для данной области знания о том, какие дополнительные факты могут быть допущены, если специфические данные найдены в БД.

Действия продукций могут состоять из активных процедур, которые автоматически производят необходимые операции над содержимым БД (либо подобно «демонам» проверять самих себя на предмет того, выполняются ли их условия активации). В этом случае форма представления знаний является процедурной, хотя и в весь-ма ограниченном виде. В последующих итерациях факты, добавлен-ные в БД, могут подключать (активировать) другие продукции и т. д.

В классических продукционных системах БД представляют собой переменную часть системы, в то время как правила и интерпре-

татор чаще всего не меняются. Будучи реализованы процедурно, классические продукционные модели обладают весьма привлекательным свойством модульности. Поэтому правила могут быть добавлены или удалены без возникновения неожиданных побочных эффектов. Причина этого заключается также в том, что в классических системах вызов процедур осуществляется только в зависимо-сти от состояния данных; процедуры, как правило, не активируют-ся другими процедурами. Поэтому продукционные системы могут быть с большим успехом использованы для областей знаний, о кото-рых располагаем только некоторым набором независимых правил (эвристик), а не четкой теорией, вполне завершенной и последовательной, и где поэтому нет алгоритмов, прямо приводящих к цели.

Продукционные системы все более широко используются для реализации продукционных ЭС.

Как правило, классические продукционные системы не содер-жат сведений о применении, т. е. знаний о том, например, какие продукции использовать для достижения цели. Это ведет к значительному снижению эффективности их работы: хотя в каждой итерации только одна продукция может быть активирована, долж-ны быть проверены условия всех продукций. Для большого числа правил это может потребовать значительного расхода ресурсов. Более того, последовательность выполнения продукций зависит на каждой итерации от состояния всех переменных системы.

В этом случае появляется проблема комбинаторного «взрыва». Для решения этих проблем предлагались подходы, связанные с методами структурного совершенствования БД и условий в продукциях, что позволило бы повысить эффективность функционирования. Предпринимаются также попытки повлиять на ход управления.

В каждом цикле все правила, условия которых удовлетворены содержимым БД, считаются определенными. Если существует несколько таких правил, то вопрос о том, какое из правил выбрать, решается с помощью какой-либо приемлемой стратегии «разрешения конфликтов» (например, выбирается правило с наивысшим приоритетом из заранее определенного перечня приоритетов). Все действия, связанные с выбранным правилом, выполняются и вызывают соответствующие изменения в БД.

Другая возможность заключается в осуществлении точного контроля последовательности выполнении продукций. В простей-

шем случае продукции могли бы формировать специальные сигналы в БД, которые подключали бы соответствующие продукции в других циклах. В некоторых системах сами продукции могут активировать или дезактивировать другие продукции и даже влиять на работу интерпретатора.

В настоящее время разработано множество моделей представления знаний, используемых для реализации систем, основанных на знаниях, в которых знания представлены с помощью правил вида ЕСЛИ-ТОГДА (явление — реакция, условие — действие). Систему продукций можно считать наиболее распространенной моделью представления знаний. Примерами реальных систем, основанных на знаниях, в которых в качестве основной модели представления знаний использовалась система продукций, являются EMYCIN, OPS-5, AGE.

Если систему продукций рассматривать как модель представления знаний, то правилам, рассматриваемым с точки зрения человека как средства прямого описания способа логического вывода для решения задач в предметной области, можно придать ясный смысл. При этом отличительной чертой представления знаний с высокой модульностью является простота дополнения, модификации и аннулирования.

Кроме того, со стороны компьютера имеется возможность определения простого и точного механизма использования знаний с высокой однородностью, описанных по одному синтаксису. Эти две отличительные черты, по видимому являются причинами столь широкого распространения метода представлений знаний правилами [30].

Классическим механизмом представления знаний в системах является исчисление предикатов (использовалось уже в 50-х годах в исследованиях по ИИ). В системах, основанных на исчислении предикатов, знания представляются с помощью перевода утверждений об объектах некоторой предметной области в формулы логики предикатов и добавления их как аксиом в систему. Рассмотрим основные положения логики предикатов.

Пусть имеется некоторое множество объектов, *называемых предметной областью М*. Знаки, обозначающие элементы этого множества, называют *предметными константами*, а знак, обозначающий произвольный элемент этого множества, — *предметной переменной*. Терм — это всякая предметная область или пред-

метная константа. Если f — функциональная n-местная буква, и $t1$, $t2$, ..., tn — термы, то $f(t1,\ldots,tn)$ есть терм.

Выражение $P(x1, x2, \ldots, xn)$, где xi, $i = 1, n$ — предметные переменные, а P принимает значения 0 и 1, называется *логической функ- цией* или предикатом. Переменные принимают значения из произвольного конечного и бесконечного множества M.

Предикатом или логической функцией называется функция от любого числа аргументов, принимающая истинные значения 1 и 0. Если в данном выражении заменить xi на yi, где yi — пред-метные константы, то получим *элементарную формулу*, т. е. пре-дикатные буквы применимы также и к предметным константам. Элементарные формулы иногда называют атомными. Из элементарных формул с помощью логических связок (или), (и), (отрицание), (импликация) строят предметные формулы (иногда их называют *правильно построенными формулами —*
ППФ). ППФ — один из важных классов выражений в исчисле-нии предикатов. Кроме логических связок в рассмотрение вво-дят кванторы общности $\forall$ или существования $\exists$. Если P — пред-метная формула, а x — предметная переменная, то выражения $\forall x\ P(x)$ и $\exists x\ P(x)$ также считаются предметными формулами. В логи-ке предикатов для компактной записи высказываний типа: для любого x истинно $P(x)$ и существует такое x, для которого ис-тинно $P(x)$ вводится две новые дополнительные операции: кантор общности $\forall$ и квантор существования $\exists$. Посредством этих операций приведенные выше высказывания записываются в виде $\forall xP(x)$ и $\exists xP(x)$. Выражение $\forall xP(x)$ обозначает высказывание истинное, когда $P(x)$ истинно при всех x M и ложно в противном случае Если $P(x)$ в действительности не зависит от x, то выражения $\forall xP(x)$ и $\exists xP(x)$ обозначают то же, что и $P(x)$. Конкретное вхождение переменной x в формулу P называется связанным, если оно либо непосредственно следует за каким-либо квантором, либо содержится в области действия некоторого квантора $\forall$ или $\exists$. Вхождение переменной является свободным, если оно не является связанным. В выражении $\forall xP(x, y)$ x — связанная, y — связанная. *Связанной переменной* называется переменная, если в P имеется вхождение этой переменной. Использование обоих кванторов не является обязательным.

Выводом системы представления знаний на предикатах являются формулы, выводимые из аксиом с помощью правил выво-

да. Для организации логического вывода могут использоваться правила.

Для решения конкретной задачи начальное состояние и доступные операторы действий переводятся в формулы исчисле-ния предикатов и добавляются к множеству аксиом. Целевое состояние также выражается формулой и рассматривается как теорема, которая должна быть выведена из аксиом с помощью активного *механизма вывода*. Выделяют следующие основные формы логического вывода:

• *Индукция* (лат. наведение) — это форма мышления, посредством которой мысль наводится на какое-либо общее правило, общее положение, присущее всем единичным предметам какого-либо класса. Индуктивный логический вывод является перспективным направлениеминженериизнаний,здесьнерассматривается.

• *Дедукция* (лат. выведение) — такая форма мышления, когда новая мысль выводится чисто логическим путем (т. е. по законам логики) из предшествующих мыслей. Такая последовательность мыс- лей называется *выводом*, а каждый компонент этого вывода явля- ется либо ранее *доказанной мыслью*, либо *аксиомой*, либо *гипотезой*. Последняя мысль данного вывода называется *заключением*.

Последовательность дедукции определяет план того, как достигнуть цели из начального состояния.

Дедукция обычно выполняется с помощью попытки вывести противоречие из получаемого в результате преобразований множества аксиом. Либо: для того, чтобы показать, что некоторое множество ППФ неудовлетворительно, надо доказать, что нет такой интерпретации, при которой каждая из ППФ в этом множестве имеет значение 1 (истинно). Хотя эта задача и кажется трудоемкой, существуют довольно эффективные процедуры ее решения. Для выполнения этих процедур требуется представить ППФ данного множества в специальном удобном виде — в виде *предложений*.

Для представления и описания стереотипных объектов, событий или ситуаций были введены понятия «фреймы», которые являются сложными структурами данных.

Теория фреймов — это парадигма для представления знаний с целью использования этих знаний компьютером. Впервые была представлена в 1975 г. как попытка построить фреймовую сеть или парадигму с целью достижения большего эффекта понимания.

Фреймовая модель представления знаний представляют собой такую схему, в которой информация содержится в специальных ячейках, называемых фреймами, объединенными в сеть, называемую системой фреймов. Новый фрейм активизируется с наступлением новой ситуации. Отличительной его чертой является то, что он одновременно содержит большой объем знаний и в то же время является достаточно гибким для того, чтобы быть использованным как отдельный элемент базы данных. Термин «фрейм» был наиболее популярен в середине 70-х годов, когда существовало много его тол-кований.

Фреймы — это фрагменты знания, предназначенные для представления стандартных ситуаций. Фреймы имеют вид структурированных компонентов ситуаций, называемых слотами. Слот может указывать на другой фрейм, устанавливая, таким образом, связь между двумя фреймами. Могут устанавливаться общие связи типа связи по общению. С каждым фреймом ассоциируется разнообразная информация (в том числе и процедуры), например ожидаемые процедуры ситуации, способы получения информации о слотах, значения принимаемые по умолчанию, правила вывода.

Каждый фрейм как структура хранит знания о предметной обла-сти (фрейм — прототип), а при заполнении слотов знаниями пре-вращается в конкретный фрейм события или явления.

Фреймы можно разделить на две группы: фреймы-описания; ро-левые фреймы. Рассмотрим пример.

Фрейм — описание: [<программное обеспечение>, <программа 1С бухгалтерия, версия 8.2>, <программа 1С торговля, версия 8.2>, <правовая программа «Консультант +» проф.>].

Ролевой фрейм: [<заявка на продажу>, <что, установка и покуп-ка программы 1С торговля, версия 8.2>, <откуда, фирма Х>, <куда, фирма «У»>, <кто, курьер Петров>, <когда, 27 января 2014 г.>].

Во фрейме-описании в качестве имен слотов задан вид программного обеспечения, а значение слота характеризует массу и производителя конкретного вида продукции. В ролевом фрейме в качестве имен слотов выступают вопросительные слова, ответы на которые являются значениями слотов. Для данного примера представлены уже описания конкретных фреймов, которые могут называться либо фреймами — примерами, либо фреймами-экземпляра-ми. Если в приведенном примере убрать значения слотов, оставив только имена, то получим так называемый фрейм–прототип.

Достоинства фрейма-представления во многом основываются на включении в него предположений и ожиданий. Это достигается за счет присвоения по умолчанию слотам фрейма стандартных ситуаций. В процессе поиска решений эти значения могут быть заменены более достоверными. Некоторые переменные выделены таким образом, что об их значениях система должна спросить пользователя. Часть переменных определяется посредством встроенных процедур, называемых внутренними. По мере присвоения переменным определенных значений осуществляется вызов других процедур. Этот тип представления комбинирует декларативные и процедур-ные знания. Фреймовые модели обеспечивают требования структурированности и связанности. Это достигается за счет свойств наследования и вложенности, которыми обладают фреймы, т. е. в качестве слотов может выступать система имен слотов более низкого уровня, а также слоты могут быть использованы как вызовы каких-либо процедур для выполнения. Для многих предметных областей фреймовые модели являются основным способом формализации знаний.

Согласно определения Минского, фреймы — это минимальные структуры информации, необходимые для представления класса объектов, явлений или процессов. В общем виде фрейм может быть представлен в виде, показанном на рис. 14.1 и описан строкой:

<ИФ, (ИС, ЗС, ПП), ..., (ИС, ЗС, ПП)>,

где ИФ — имя фрейма;
ИС — имя слота; ЗС —
значение слота;
ПП — имя присоединенной процедуры.

Слоты — это некоторые незаполненные подструктуры фрейма, заполнение которых приводит к тому, что данный фрейм ставится в соответствие некоторой ситуации, явлению или объекту.

С каждым фреймом связана информация: как использовать фрейм; что делать, если происходит что-либо непредвиденное; недостающие значения для слотов. Фрейм с заполненными слотами называется экземпляром фрейма. Для организации связи между объектами предметной области строится сеть фреймов. Связь может быть организована путем указания в качестве значений некоторых слотов одного фрейма имен других фреймов.

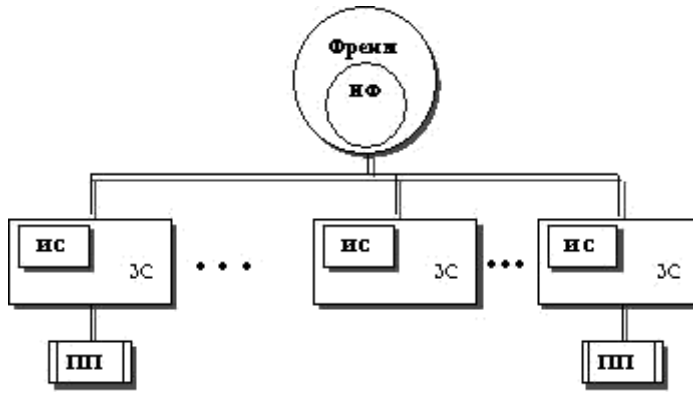


Рис. 14.1. Схема фрейма

В качестве данных фрейм может содержать обращения к процедурам (так называемые присоединенные процедуры). Выделяют два вида процедур: процедуры-демоны и процедуры-слуги. Проце- дуры-демоны активизируются при каждой попытке добавления или удаления данных из слота (по умолчанию). Процедуры-слуги активизируются только при выполнении условий, определенных пользователем при создании фрейма.

Для уменьшения информационной избыточности во фреймовых системах реализуют принцип наследования информации, позволяющий общую (глобальную) для системы информацию хранить в отдельном фрейме, а во всех остальных фреймах указывать лишь ссылку на место хранения этой информации.

Рассмотрим подробнее основные свойства фреймов.

1. *Базовый тип.* При эффективном использовании фреймовой системы, можно добиться быстрого понимания сущности данного предмета и его состояния, однако для запоминания различных позиций в виде фреймов необходимы большие объемы памяти. Поэтому только наиболее важные объекты данного предмета запо-минаются в виде базовых фреймов, на основании которых строятся фреймы для новых состояний. При этом каждый фрейм содержит слот, оснащенный указателем подструктуры, который позволяет различным фреймам совместно использовать одинаковые части.

2. *Процесс сопоставления.* Процесс, в ходе которого проверяется правильность выбора фрейма, называется процессом сопо-

ставления. Обычно этот процесс осуществляется в соответствии с текущей целью и информацией (значениями), содержащийся в данном фрейме. То есть фрейм содержит условия, ограничивающие значения слота, а цель используется для определения, какое из этих условий, имея отношение к данной ситуации, является суще-ственным.

3. *Иерархическая структура.* Фрейм обычно соответствует представлению общего понятия с классификационной иерархической структурой. Особенность иерархической структуры заключается в том, что информация об атрибутах, которую содержит фрейм верхнего уровня, совместно используются всеми фреймами нижних уровней, связанных с ним.

4. *Сети фреймов.* Если процесс сопоставления не привел к успеху, возникает необходимость поиска фрейма, подобного предыдущему. Такой поиск, осуществляемый с использованием указателей различия, возможен благодаря соединению фреймов, описывающих объекты с небольшими различиями, с данными указателями и образованию сети подобных фреймов.

5. *Отношения «абстрактное — конкретное» и «целое — часть».* Рассмотренная иерархическая структура основывается на отношениях «абстрактное — конкретное», однако помимо такого типа структур существуют и другие, основанные на отношениях «целое — часть».

Отношения «абстрактное — конкретное» характерны тем, что на верхних уровнях расположены абстрактные объекты, а на нижних — конкретные объекты, при чем объекты нижних уровней наследуют атрибуты объектов верхних уровней.

Если одно отношение «целое — часть» касается структурированных объектов и показывает, что объект нижнего уровня является частью объекта верхнего уровня.

Наибольшее практическое применение во фреймовых системах получили лишь отношения «абстрактное — конкретное». Но в некоторых областях иногда требуется описывать и управлять структурированным объектом. Поэтому в таких случаях не обойтись без обработки отношений типа «целое — часть».

Важной схемой представления знаний являются семантические сети. Впервые это понятие было введено в 60-х годах для представления семантических связей между концепциями слов. Семантические сети не являются однородным классом схем представле-

ния. Имеется лишь несколько общих черт, объединяющих ряд механизмов представления, называемых семантическими сетями.

Семантическая сеть — это ориентированный граф, вершины которого — понятия, а дуги — отношения между ними. Часто общей основой являются лишь сходство формального обозначения и основной принцип, заключающийся в том, что элементы знаний должны храниться смежно, если они семантически связаны.

Существуют в то же время достаточно широко различающиеся мнения о том, что должно быть представлено в семантических сетях, какие семантические элементы должны быть использованы для представления и, наконец, в чем же заключается семантика этих сетей.

Изначально семантическая сеть была задумана как модель представления структуры долговременной памяти в психологии, но впоследствии стала одним из основных способов представления знаний в инженерии знаний. Моррис дал точное определение семантическим и прагматическим отношениям в семиотике и определил их как проблемы различных функциональных уровней. Другими словами, семантика означает определенные (общие) отношения между символами и объектами, представленными этими символами, а прагматика — в ыразительные (охватывающие) отношения между символами и создателями (или пользователями) этих символов. Первоначально в психологии изучались объекты, именуемые семантическими с точки зрения известных ассоциативных свойств, накапливаемых в системе обучения и поведения человека. Однако с развитием психологии познания стали изучать-ся семантические структуры, включающие некоторые объекты. Затем были изучены принципы действия человеческой памяти (способы хранения информации и знаний), в частности предполо-жительные (гипотетические) структурные модели памяти, и созда-ны моделирующие программы, понимающие смысл слов.

Одной из структурных моделей долговременной памяти явля-ется предложенная Куиллианом модель понимания смысла слов, получившая название TLC-модели (Teachable Language Comprehender: доступный механизм понимания языка). В данной модели для описания структуры долговременной памяти была использована сетевая структура как способ представления семантических отношений между концептами (словами). Данная модель имитирует естественное понимание и использования языка человеком. По-

этому основной ее идеей было описание значений класса, к которо-му принадлежит объект, его прототипа и установление связи со словами, отображающими свойства объекта.

В модели Куиллиана концептуальные объекты представлены ассоциативными сетями, состоящими из вершин, показывающих концепты, и дуг, показывающих отношения между концептами. Подобная ассоциативная структура называется плоскостью, описываемые концепты объекта называются вершинами типа, а связанные с ними соответствующие отдельные слова — вершинами лексем. В любой плоскости существуют одна вершина типа и необходимое для определения концептов, описывающих его, число вершин лексем. Вершины лексем определяют всевозможные сущно-сти, имеющие место в реальном мире, например классы, свойства, примеры, время, место, средство, объекты и т. п. Преимущество лексем по сравнению с типами заключается в экономии пространства памяти ЭВМ. Это означает и факт предотвращения дублирования определения концептов. Итак, исходя из приведенных выше соображений, можно сделать вывод, что в TLC-модели использует-ся представление знаний в форме элемент и свойства. Другими словами, можно попытаться структурировать знания, заменив вер- шину типа на элемент, а вершину лексемы на свойство. Благодаря этому данные, основанные на фактах, в долговременной памяти можно представить с помощью структур трех типов: элементы, свойства и указатели. Элемент представлен заключением, называе-мым фактом, например объектом, событием, понятием и т. п.; обычно за элемент принимаются отдельное слово, имя существи-тельное, предложение или контекст. Свойство — это структура, описывающая элемент, оно соответствует таким частям речи, как имя прилагательное, наречие, глагол и т. д. указатели связывают элементы и свойства.

Важность модели семантической сети Куиллиана с точки зрения многочисленных приложений определяется следующими моментами. В отличие от традиционных методов семантической обработки с анализом структуры предложения были предложены новые парадигмы в качестве модели представления структуры долговременной памяти, в которой придается значение объему языковой активности.

Были предложены способ описания структуры отношений меж-ду фактами и понятиями с помощью средства, называемого семан-

тической сетью, отличающейся несложным представлением понятий, а также способ семантической обработки в мире понятий на основе смысловой связи (смыслового обмена) между прототипа-ми. Была создана реальная система TLC, осуществлено моделиро-вание человеческой памяти и разработана технологическая сторо-на концепции понимания смысла.

Таким образом, под семантической сетью понимают направленный граф с помеченными вершинами и дугами, в котором вершины соответствуют конкретным объектам, а дуги, их соединяющие, отражают имеющиеся между ними отношения. Отношения, используемые в семантических сетях, можно разделить на следующие:

- лингвистические, в частности надежные, включающие в себя отношения типа «объект», «агент», «условие», «место», «инструмент», «цель», «время» и др.;
- атрибутивные, к которым относят форму, размер, цвет и т. д.;
- характеризации глаголов, т. е. род, время, наклонение, залог, число;
- логические, обеспечивающие выполнение операций для исчисления высказываний (дизъюнкция, конъюнкция, импликация, отрицание);
- квантифицированные, т. е. использующие кванторы общности и существования;
- теоретико-множественные, включающие понятия «элемент множества», «подмножество», «супермножество» и др.

Если имеется конечное множество атрибутов $A = \{Ai,\ i = i,\ n\}$ и конечное множество отношений $R = \{Rj,\ j = i,\ m\}$, то под интенсионалом отношения Rj понимают набор пар вида: $INT\ (\ Rj) = \{..., [Ai, DOM(Ai)], ...\}$, в которых $DOM\ (Ai)$ означает домен Ai, т. е. множество значений атрибута Ai соответствующего отношения Rj. Под экстенсионалом отношения Rj понимают множество $EXT\ (Rj) = \{F1, ..., Fp\}$, где FK — факт отношения Rj, задаваемый в виде совокупности пар вида «атрибут — значение».

Интенсиональная семантическая сеть описывает предметную область на обобщенном, концептуальном уровне, в то время как в экстенсиональной сети производятся конкретизация и наполнение фактическими данными.

Выразительная сила семантических сетей несколько слабее, чем в логике предикатов. В частности, представляет определенную

сложность отображение квантификаторов. Некоторые недостат-ки могут быть устранены с помощью реализации механизма- наследования: субконцепции наследуют свойства суперконцепций-если только это явно не запрещено.

Статические базы знаний, представленные с помощью семантических сетей, могут быть объектом действий, производимых активными процессами. Стандартные операции включают в се-бя процессы поиска и сопоставления, с помощью которых определяется, представлена ли в семантической модели (и где именно) специфическая информация.

По сравнению с логикой предикатов семантические сети имеют то важное преимущество, что вся точно известная информация о той или иной концепции расположена в базе знаний круг соответствующей вершины.

Семантические сети нашли применение в основном в системах обработки естественного языка, частично в вопросно-ответных-системах, а также в системах искусственного видения. В последних семантические сети используются для хранения знаний-о структуре, форме и свойствах физических объектов. В области обработки естественного языка с помощью семантических сетей представляют семантические знания, знания о мире, эпизодические знания (т. е. знания о пространственно-временных событи-ях и состояниях).

В качестве примера, рассмотрим представление знаний, содержащихся в высказывании «Поставщик *N* отгрузил товар из склада *M* автотранспортом». На рис. 14 . 2 представлена интенсиональная, а на рис. 14.3 — экстенсиональная семантическая сеть. Факты обо-значим овалом, а понятия и объекты прямоугольником [30].

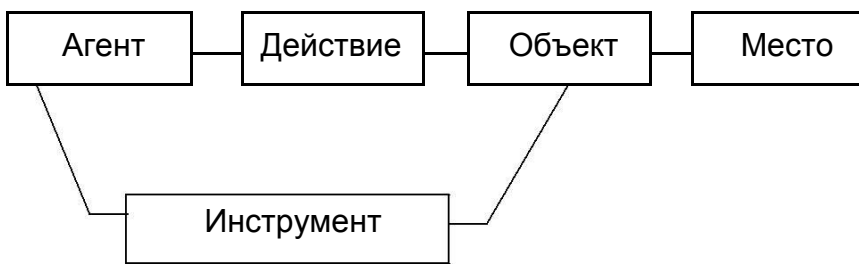


Рис. 14.2. Интенсиональная семантическая модель

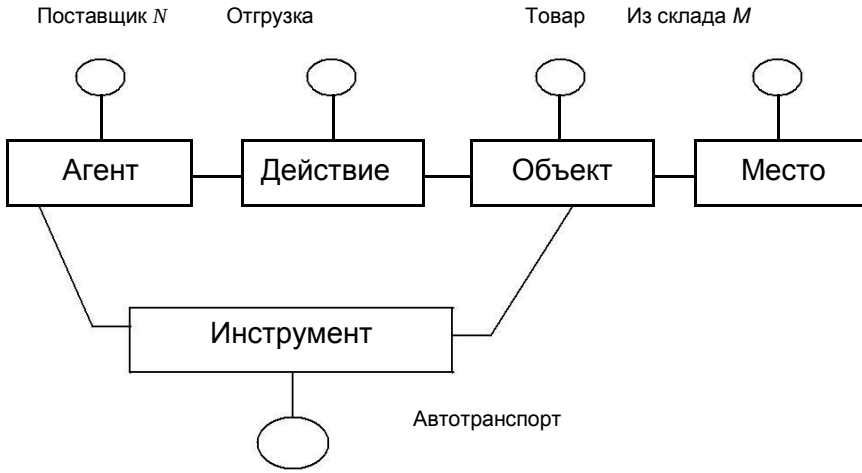


Рис. 14.3. Экстенсиональная семантическая сеть

Понятиями обычно выступают абстрактные или конкретные объекты, а отношения — это связи типа: «это» («is»), «имеет частью» («has part»), «принадлежит», «любит». Характерной особенностью семантических сетей является обязательное наличие трех типов отношений:

- класс — элемент класса;
- свойство — значение;
- пример элемента класса.

Существует несколько классификаций семантических сетей. Например, по количеству типов отношений семантические сети разделяются на однородные (с единственным типом отношений), неоднородные (с различными типами отношений). По типам отношений: бинарные (в которых отношения связывают два объек-та); парные (в которых есть специальные отношения, связываю-щие более двух понятий). Наиболее часто в семантических сетях используются следующие отношения: связи типа «часть-целое» («класс-подкласс», «элемент-множество» и т. п.), функциональные связи (определяемые обычно глаголами «производит», «влия-ет», ...); количественные (больше, меньше, равно, ...); простран-ственные (далеко от, близко от, за, под, над, ...); временные (рань-ше, позже, в течение, ...); атрибутивные связи (иметь свойство, иметь значение, ...); логические связи (и, или, не) и др.

Проблема поиска решения в базе знаний типа семантической сети сводится к задаче поиска фрагмента сети, соответствующего некоторой подсети, соответствующей поставленному вопросу.

На рис. 14.4 изображена семантическая сеть. В качестве вершин — понятия: Человек, Иванов, Волга, Автомобиль, Вид транспорта, Двигатель.

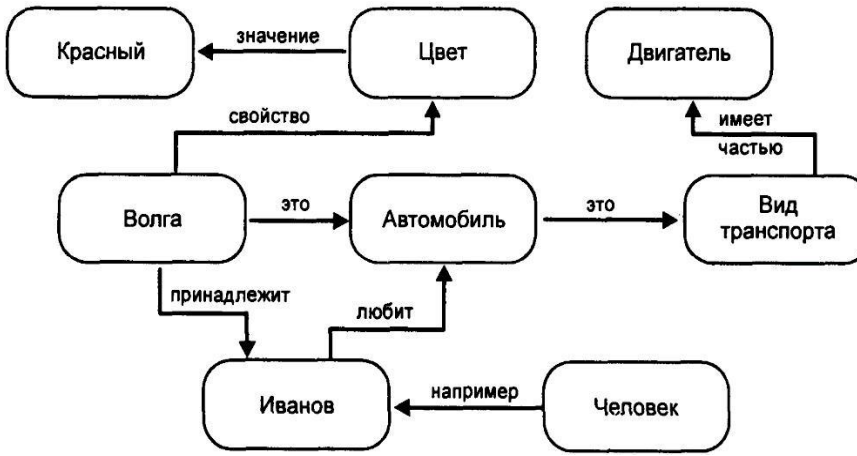


Рис. 14.4. Семантическая сеть

Основное преимущество этой модели — в соответствии современным представлениям об организации долговременной памяти человека. Недостаток модели — сложность поиска вывода на семантической сети.

Для реализации семантических сетей существуют специальные сетевые языки, например NET [12] и др. Широко известны экспертные системы, использующие семантические сети в качестве языка представления знаний — PROSPECTOR, CASNET, TORUS и др. [8; 10].

Вопросы для самоконтроля:

1. Дайте определение модели знаний.
2. В чем состоит сущность продукционной модели?
3. Из каких частей состоит продукция?
4. В чем сущность фреймовой модели.

5. Назовите виды фреймов.
6. Перечислите свойства фреймов.
7. Дайте определение семантической сети.

Тема 15

ЭКСПЕРТНЫЕ СИСТЕМЫ

1. Экспертные системы и их основные характеристики.
2. Состав и взаимодействие участников построения и эксплуатации экспертных систем.
3. Архитектура экспертных систем.
4. Этапы разработки экспертных систем.

В начале 80-х годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название «экспертные системы» (ЭС). Основным назначением ЭС является разработка программных средств, которые при решении задач, трудных для человека, получают результаты, не уступающие по качеству и эффективности решения, решениям получаемым человеком-экспертом. ЭС используются для решения так называемых неформализованных задач, общим для которых является то, что:

- задачи не могут быть заданы в числовой форме;
- цели нельзя выразить в терминах точно определенной целевой функции;
- не существует алгоритмического решения задачи;
- если алгоритмическое решение есть, то его нельзя использовать из-за ограниченности ресурсов (время, память).

Кроме того неформализованные задачи обладают ошибочностью, неполнотой, неоднозначностью и противоречивостью как исходных данных, так и знаний о решаемой задаче.

Экспертная система — это программное средство, использующее экспертные знания для обеспечения высокоэффективного решения неформализованных задач в узкой предметной области. Основу ЭС составляет база знаний (БЗ) о предметной области, которая накапливается в процессе построения и эксплуатации ЭС. Накопление и организация знаний — важнейшее свойство всех ЭС (рис. 15.1).

Рис. 15.1. Основные свойства ЭС

Знания являются явными и доступными, что отличает ЭС от традиционных программ, и определяет их основные свойства, такие, как:

1. Применение для решения проблем высококачественного опыта, который представляет уровень мышления наиболее квалифицированных экспертов в данной области, что ведет к решениям творческим, точным и эффективным.

2. Наличие прогностических возможностей, при которых ЭС выдает ответы не только для конкретной ситуации, но и показывает, как изменяются эти ответы в новых ситуациях, с возможностью подробного объяснения каким образом новая ситуация привела к изменениям.

3. Обеспечение такого нового качества, как институциональная память, за счет входящей в состав ЭС базы знаний, которая разработана в ходе взаимодействий со специалистами организации,
и представляет собой текущую политику этой группы людей. Этот набор знаний становится сводом квалифицированных мнений и постоянно обновляемым справочником наилучших стратегий и методов, используемых персоналом. Ведущие специалисты уходят, но их опыт остается.

4. Возможность использования ЭС для обучения и тренировки руководящих работников, обеспечивая новых служащих обширным багажом опыта и стратегий, по которым можно изучать рекомендуемую политику и методы.

ДЭС ориентированы на работу с постоянно меняющейся информацией в информационной базе и в базе знаний системы. Компонентом базы знаний является база данных.

Основные назначения экспертной системы:

1. Интерпретация — это анализ данных с целью определения их смысла. Интерпретатор должен быть в состоянии обрабатывать

информацию представленную частично, выдвигать гипотезы о доверии данным. При ненадежных данных интерпретация также будет ненадежной, поэтому для достижения доверия необходимо определить, какая информация была неточной или неопределенной. Так как цепочки рассуждений в ИС могут быть достаточно длинными, интерпретатору необходимо располагать средствами объяснения того, как интерпретация обусловлена имеющимися данными.

2. Планирование — составление планов. Планировщик должен уметь делать пробные шаги и исследовать возможные планы, уметь сосредотачивать внимание на наиболее важных гипотезах, работать в условиях неопределенности. Планирование должно быть условным, зависящим от поступления новых сведений.

3. Прогнозирование — прогнозирование действий на определенный (заданный) промежуток времени. Прогнозирование — это определение хода событий в будущем на основании модели прошлого и настоящего. Ключевыми проблемами задачи является требование соединения в единое целое неполной информации. Прогнозирование должно рассматривать различные варианты будущего и указывать их чувствительность к изменению входных данных. Решаемая задача прогнозирования должна носить условный характер, поскольку вероятность определенных событий в будущем будет зависеть от более близких, но не предсказуемых со-бытий.

4. Мониторинг — непрерывное оповещение о состоянии системы или процесса. Это непрерывная интерпретация сигналов и выдача оповещений при возникновении ситуаций, требующих вмешательства.

5. Проектирование — использование экспертной системы для исключения профессионала из задачи проектирования или выполнение рутинных действий по обработке информации в конкретной прикладной системе. Ключевыми проблемами проектирования являются: отсутствие исчерпывающей информации, позволяющей увязать ограничения проектирования с принимаемыми решениями, взаимодействие подзадач, взаимное влия-ние подзадач, умение видеть картину в целом, чтобы уходить из точек в пространстве проекта, которые являются лишь локаль-но-оптимальными, оценивание последствий принимаемых ре-шений.

6. Обучение — рассматривается в двух аспектах: обучение поль-зователя, а также самообучение системы, как на этапе приобрете-ния знаний, так и в процессе работы.

7. Диагностика — это процесс поиска неисправностей в системе или определение стадии заболевания в медицине, основанный на интерпретации данных (возможно зашумленных). Основные проблемы, возникающие при решении задачи: недоступность или малодоступность некоторых данных, сочетание не вполне совместимых частных моделей объектов или процессов и пр.

Познакомившись с тем, что такое экспертные системы и каковы их основные характеристики, попробуем теперь ответить на вопрос: «Кто участвует в построении и эксплуатации ЭС?»

К числу основных участников следует отнести саму экспертную систему, экспертов, инженеров знаний, средства построения ЭС и пользователей. Их основные роли и взаимоотношение приведены на рис. 15.2.

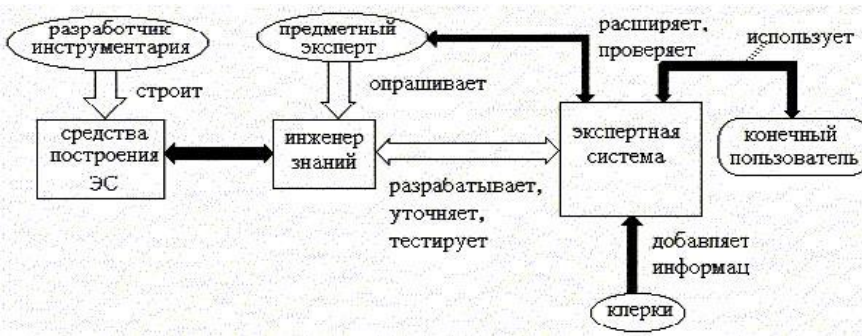


Рис. 15.2. Основные роли и взаимоотношение основных участников построении и эксплуатации ЭС

Экспертная система — это программное средство, использующее знания экспертов, для высокоэффективного решения задач в интересующей пользователя предметной области. Она называется системой, а не просто программой, так как содержит базу знаний, решатель проблемы и компоненту поддержки. Последняя из них помогает пользователю взаимодействовать с основной программой.

Эксперт — это человек, способный ясно выражать свои мысли и пользующийся репутацией специалиста, умеющего находить правильные решения проблем в конкретной предметной области. Экс-

перт использует свои приемы и ухищрения, чтобы сделать поиск решения более эффективным, и ЭС моделирует все его стратегии.

Инженер знаний — человек, как правило, имеющий познания в информатике и искусственном интеллекте и знающий, как надо строить ЭС. Инженер знаний опрашивает экспертов, организует знания, решает, каким образом они должны быть представлены в ЭС, и может помочь программисту в написании программ.

Средство построения ЭС — это программное средство, используемое инженером знаний или программистом для построения ЭС. Этот инструмент отличается от обычных языков программирования тем, что обеспечивает удобные способы представления сложных высокоуровневых понятий.

Пользователь — это человек, который использует уже построенную ЭС. Так, пользователем может быть юрист, использующий ее для квалификации конкретного случая; студент, которому ЭС помогает изучать информатику и т. д. Термин пользователь несколько неоднозначен. Обычно он обозначает конечного пользователя. Однако из рис. 15.2 следует, что пользователем может быть:

- создатель инструмента, отлаживающий средство построения ЭС;
- инженер знаний, уточняющий существующие в ЭС знания;
- эксперт, добавляющий в систему новые знания;
- клерк, заносящий в систему текущую информацию.

Важно различать инструмент, который используется для построения ЭС, и саму ЭС. Инструмент построения ЭС включает как язык, используемый для доступа к знаниям, содержащимся в системе, и их представления, так и поддерживающие средства — программы, которые помогают пользователям взаимодействовать с компонентой экспертной системы, решающей проблему [28].

Экспертные системы (ЭС) относятся к задачам, которые претендуют на NP-полноту. NP-полнота систем характеризуется заданной сложностью. Сложность задачи характеризуется количеством шагов поиска решения.

1. Если задача сходится за экспоненциальное количество шагов, т. е. алгоритм решения задачи можно представить в виде экспоненты, то сложность задачи определяется классом E.

2. Если задача сходится за полиномиальное количество шагов, т. е. формулу нахождения решения можно свести к полиному, то данная задача характеризуется сложностью P.

3. Если для задачи отсутствует формула нахождения решения, отсутствует алгоритм получения результата или получить алгоритм достаточно сложно и трудоемко, но возможно, то сложность такой задачи характеризуется, как неопределенное полиномиальное (NP).

Экспертные системы, или системы, основанные на знаниях, предназначены для решения не формализуемых или слабо формализуемых задач. Там, где используется враждебная человеку среда, отсутствует алгоритм решения задачи, или для решения задачи требуется достаточно много времени (машинного) и имеется недостаток в числе экспертов для решения поставленной задачи. Данные задачи решают проблемы в условиях неполной, нечеткой, размытой и, возможно, ошибочной информации.

Эксперт — это человек, который является профессионалом высокой квалификации в данной проблемной области, для которой предназначена разработка системы.

Экспертная система разрабатывается в том случае, если ее разработка, во-первых, необходима, во-вторых, оправдана и неоценима.

База знаний это правила логического вывода, не меняющиеся в промежутке времени, представлены в установленном для системы формализмах (фреймах, семантических сетях, исчисление предикатов первого порядка). Без данных знаний не бывает!

База данных — содержит постоянные, не меняющиеся в заданном промежутке времени данные, при активизации знаний из базы знаний. Блок приобретения знаний предназначен для работы с экспертом и когнитологом.

Интерфейс пользователя предназначен для взаимодействия с пользователем, отображает состояние рабочей памяти на основе рекомендаций и комментариев о состоянии рабочей памяти. Блок проверки на корректность и противоречивость есть блок проверки знаний.

Динамическая система включает:

- рабочую память;
- блок логического вывода;
- базу знаний;
- базу данных;
- блок записи знаний;
- блок рекомендаций и комментариев;
- интерфейс пользователя;

- блок преобразования знаний;
- прикладную систему;
- систему датчиков взаимодействия с естественной средой;
- информационную базу.

Экспертные системы разрабатываются по следующим этапам

1. Приобретение знаний. Приобретение знаний для системы пред- ставления знаний осуществляется от эксперта в заданной приклад-ной области, когнитологом (инженером по знаниям). Существу-ет- три способа приобретения знаний:наблюдение за работой экспер-та; опрос эксперта; интервьюирование.

2. Представление знаний. Описание проблемной информации. Оно осуществляется с помощью:

- таксономической классификационной схемой (объекты, их свойства, отношения);
- выделение фактов, правил, отношений для заданной проблемной области;
- факты есть описание значений данных;
- правила есть интеграция фактов в каузальной форме (если ..., то ...) для получения решения;
- отношения есть совокупность семантических отношений для интерпретации правил в стратегиях поиска решений;
- идентификация знаний.

Факты и правила составляют знания. Факты, правила и отношения составляют стратегию поиска решения. Знания и стратегия поиска решения составляют декларативные и процедурные знания (для констатирования фактов и для выполнения действий соответственно) [29].

Структура экспертной системы должна быть прозрачна для конечного пользователя. Конечным пользователем экспертной системы является и необученный пользователь, и эксперт в заданной предметной области, и прикладной программист.

Ядром экспертной системы является естественный язык и объектно-ориентированное представление информации. Средства реализации экспертной системы (представление): графика, анимация, обработка естественного языка, обработка изображений. Различают два вида экспертных систем: статистические (СЭС) и динамические (ДЭС).

В СЭС оценивается информация на основе постоянных и неизменных данных в базе данных и правил в базе данных. Данные и знания могут добавляться, но не в процессе выполнения.

Задания для проверки знаний

Тесты

1. В таблице, находящейся в первой нормальной форме в каждой ячейке находится:

a) много значений
b) массив данных
c) два значения
d) одно значение

2. Функционально определять — это: a) присваивать
b) однозначно определять
c) идентифицировать
d) задавать

3. Если отношение содержит кортежей, принадлежащих либо первому, либо второму исходным отношениям, либо обоим отношениям одновременно, то такое отношение называется:

a) естественным соединением
b) пересечением
c) разностью
d) объединением

4. Помещение в один блок записей таблиц, которые с большей вероятностью будут часто подвергаться соединению — это:

a) кластеризация
b) копирование
c) хеширование
d) индексация

От какого слова произошло название «реляционные»:

a) таблица
b) отношение
c) подмножество
d) сущность

6. Операция реляционной алгебры, создающая новую табли-цу путем выбора строки одной.... таблицы, связанной с каждой строкой другой таблицы — это:

a) разность
b) проекция
c) деление
d) произведение

7. Трехуровневая модель СУБД обеспечивает:

a) логическую и физическую независимость при работе с данными
b) логическую независимость данных, хранящихся в БД
c) логическую независимость приложений
d) централизованный доступ к данным

8. Модель, фиксирующая значение сущностей и отношений реального мира, называется:

a) мультимодельной
b) фактографической
c) реляционной
d) семантической

9. По технологии обработки базы данных подразделяют-ся на:

a) централизованные и распределенные
b) общего назначения и специализированные
c) объектно-ориентированные и теоретико-множественные

10. Отношение, связывающее объектное множество самим собой, называется:

a) сложным
b) реляционным
c) рекурсивным
d) связанным

11. Первичным ключом таблицы Покупка (ИндексПокупки, ДатаПокупки, ИндексТовара, ОбъемПокупки, Сумма) является:

a) ИндексТовара, Сумма
b) ИндексТовара

c) Сумма
d) ИндексПокупки

12. Домен — это:
a) множество значений, которые могут принимать столбцы в таблице
b) множество значений, которые могут принимать данные в таблице
c) множество значений, которые могут принимать ключевые столбцы в таблице

13. Модели, основанные на принципе организации словарей и содержащие языковые конструкции:
a) объектно-ориентированные
b) тезаурусные
c) дескрипторные
d) даталогические

14. Физический блок данных — это:
a) бакет
b) дорожка
c) физическая запись
d) цилиндр

15. СУБД — это
a) средство манипулирования данными
b) совокупность средств, предназначенных для центрального многоцелевого использования данных
c) программное средство поддержки приложений
d) совокупность языковых и программных средств, предназначенных для создания, ведения и использования БД

16. Организация, определившая основные типы логических структур данных и правила их построения — это:
a) DBTG
b) ANSY
c) ISO
d) CODASYL

17. Отношение находится в этой нормальной форме, когда все его поля, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом:

a) первой
b) второй
c) третьей
d) Бойса — Кодда

18. В качестве данной модели используются различные методы размещения данных, основанных на дисковых структурах:

a) инфологическая
b) даталогическая
c) физическая

19. В отношении Студент (НомерСтудента, Фамилия, Имя, Отчество, Курс, Группа) все остальные атрибуты функциональ-но зависят от атрибута:

a) Фамилия
b) Курс
c) Группа
d) НомерСтудента

20. Функции управления информационными ресурсами в модели файлового сервера расположены на:

a) клиенте и сервере
b) на узле сети
c) сервере
d) клиенте

21. Совокупность сведений о конкретных объектах реально-го мира в какой-либо предметной области или разделе предмет-ной области — это:

a) база данных
b) банк данных
c) информационная система

22. По технологии обработки базы данных подразделяются- на:

a) централизованные и распределенные
b) общего назначения и специализированные
c) объектно-ориентированные и теоретико-множественные

23. Администратор банка данных:

a) отвечает за оптимальную организацию одновременной работы множества пользователей
b) функционирует во время проектирования
c) координирует работу разработчиков

24. Набор конкретных значений, характеризующих объект:

a) данные
b) модель данных
c) СУБД

25. К какому из уровней относят сетевые, иерархические, ре-ляционные модели:

a) инфологическому
b) датологическому (концептуальному)
c) физическому

26. От какого слова произошло название «реляционные»:

a) таблица
b) отношение
c) подмножество
d) сущность

27. Первым этапом проектирования базы данных является:

a) построение инфологической модели
b) выбор СУБД
c) анализ предметной области
d) построение модели сущность-связь

28. Фигурой обозначают:

a) стержневую сущность
b) ассоциативную
c) обозначающую
d) характеристическую

29. Внешний ключ — это:

a) ключ, который находится за отношением
b) первичный ключ противоположного отношения
c) набор атрибутов, однозначно определяющий кортеж отношения

30. Из отношений Служ (таб_ном, ФИО,Проф,№ отд,зарпл) и Отд(№ отд, тема, объем_фин) получить отношение R, содержащее признаки ФИО и тема для всех лиц с профессией программист:

a) Служ,Отд| (∃ (Служ.Проф=’Программист’)
b) (Служ.ФИО,Отд.Тема)|((Служ.Проф=’Программист’)
c) (Служ.ФИО,Отд.Тема)|((Служ.Проф=’Программист’ Служ.№ отд=Отд.№ отд))
d) (Служ.ФИО,Отд.Тема)|(Служ.Проф=’Программист’ Служ.№ отд=Отд.№ отд)

31. Какое количество байт используется для типа данных smallint в SQL Server 2000:

a) 8
b) 4
c) 3
d) 2

32. Целые числа от 0 до 255 обозначаются типом данных:

a) Int
b) Smallint
c) Tinyint
d) Bigint

33. Тип файла при котором поиск нужной записи осуществляется при помощи специальной функции называется:

a) неупорядоченным
b) упорядоченным
c) хешированным
d) кластеризованным

34. Как называются методы упорядочивания по вторичным индексам:

a) инвертированный список
b) плотный индекс
c) неплотный индекс
d) B-дерево

35. Что такое ODBC?

a) международный стандарт
b) стандарт открытых систем

c) американский комитет по стандартизации
d) название компоненты SQL Server 2000

36. В трехуровневой архитектуре клиент-сервер на клиенте распо-ложены:
a) алгоритмы расчетов
b) алгоритмы обработки данных
c) пользовательский интерфейс
d) проверка данных

37. Расставьте по порядку этапы проектирования БД:
a) концептуальный
b) анализ предметной области
c) логический
d) физический

Задачи по теме «Физическая организация данных»

Задача № 1

На каком уровне находится узел с номером 20 в 5-арной иерархической структуре.

Задача № 2

Для хранения всех записей файла требуется 12 500 блоков, зная количество индексных записей в блоке, определите количество индексных блоков необходимых для хранения 73 индексных записей.

Задача № 3

В иерархической 5-арной иерархической структуре определить номер 27-го узла на 4-м уровне.

Задача № 4

В 5-арной иерархической структуре найдите номер записи БД узла-родителя 27-го узла на 4-м уровне.

Задачи № 5

Определить количество индексных записей в одном блоке фай-ла с неплотным индексом, если длина блока 1024 байта, а индексной записи 14 байт.

Задача № 6

Для хранения всех записей файла требуется 10 000 блоков, зная количество индексных записей в блоке определите количество индексных блоков необходимых для хранения 70 индексных записей.

Задача № 7

В 4-арной иерархической структуре найдите номер записи БД узла-родителя 27-го узла на 4-м уровне.

Задачи № 8

Определить количество индексных записей в одном блоке файла с не-плотным индексом, если длина блока 1000 байта, а индексной записи 20 байт.

Задача № 9

Для хранения всех записей файла требуется 14 500 блоков, зная количество индексных записей в блоке, определите количество индексных блоков необходимых для хранения 65 индексных записей.

Задача № 10

В иерархической 4-арной иерархической структуре определить номер 20-го узла на 4-м уровне.

Задача № 11

В 4-арной иерархической структуре найдите номер записи БД узла-родителя 20-го узла на 4-м уровне.

Задачи № 12

Определить количество индексных записей в одном блоке файла с не-плотным индексом, если длина блока 1024 байта, а индексной записи 14 байт.

Задача № 13

Для хранения всех записей файла требуется 10 000 блоков, зная количество индексных записей в блоке, определите количество индексных блоков необходимых для хранения 70 индексных записей.

Задача № 14

В иерархической 4-арной иерархической структуре определить номер 4-го узла на 3-м уровне.

Задача № 15

В 4-арной иерархической структуре найдите номер записи БД узла-родителя 5-го узла на 3-м уровне.

Задача № 16

В иерархической 4-арной иерархической структуре определить номер 27-го узла на 4-м уровне.

Задача № 17

В 5-арной иерархической структуре найдите номер записи БД узла-родителя 26-го узла на 5-м уровне.

Задача № 18

В 5-арной иерархической структуре найдите номер записи БД 26-го узла на 5-м уровне.

Задача № 19

В 6-арной иерархической структуре найдите номер записи БД 26-го узла на 4-м уровне.

Задача № 20

В 6-арной иерархической структуре найдите номер записи БД узла-родителя 26-го узла на 4-м уровне.

Задача № 21

В 4-арной иерархической структуре, на каком уровне находится узел с номером 26.

Задача № 22

На каком уровне находится узел с номером 26 в 5-арной иерархической структуре.

Задача № 23

Определить количество индексных записей в одном блоке файла с не-плотным индексом, если длина блока 1024 байта, а индексной записи 14 байт.

Задача № 24

Найдите номер узла родителя для узла 19, который находится 4-м уровне 5-арной иерархической модели.

Задача № 25

Найдите номер узла родителя для узла 10, который находится на 5-м уровне 4-арной иерархической модели.

Задача № 26

Даны отношения:

R1(№л/дПреподавателя, ФИОПреподавателя, Степень, ДатаРож-- дения,- ТелефонДом, ТелефонМоб, Адрес);

R2(НомерЗаписи, НазваниеДисциплины, №л/дПреподавателя, КоличествоЧасов, ФормаКонтроля).

Получить отношение, содержащее ФИО и мобтелефоны препо-давателей, читающих лекции по дисциплинам БД и ТЭИС.

Задача № 27

Даны отношения:

R1(№л/дПреподавателя, ФИОПреподавателя, Степень, ДатаРож-- дения,- ТелефонДом, ТелефонМоб, Адрес);

R2(НомерЗаписи, НазваниеДисциплины, №л/дПреподавателя, КоличествоЧасов, ФормаКонтроля).

Получить отношение, содержащее ФИО преподавателей млад-ше 30 лет и имеющих степень кандидата наук.

Задача № 28

Товар (Инвентарный№, Наименование, ЕдиницаИзмерения, Цена);

Поставка (№накладной, ДатаПоставки, Инвентарный№, Коли-чество); Продажа (НомерЧека, Инвентарный№, ДатаПродажи,

Количе--

ство).

Получить отношение, содержащее сведения наименование то-вара, дата продажи, сумму на которую продали.

Задача № 29

Товар (Инвентарный№, Наименование, ЕдиницаИзмерения, Цена);

Поставка(№накладной, ДатаПоставки, Инвентарный№, НомерПо-- ставщика, Количество); Поставщики (НомерПоставщика,

Наименование, Город, Улица,

Теле-фон).

Получить отношение, содержащее сведения о товарах, продан-ных по-ставщиками, зарегистрированными в г. Кизляре.

Вопросы для дополнительных занятий по разделу «Базы данных»

1. Именно эти проекты послужили растущему к ней интересу:
1) проект, который в конце 70-х годов разрабатывался в Калифорнии под руководством Астрахана;
2) проект INGRES, над которым работа велась одновременно с первым;
3) велся в Великобритании, был более теоретическим по сравнению с предыдущим.

2. Что означает представленные аббревиатуры:
CODASYL
SQL
DBMS

3. Реальное распространение архитектуры «клиент-сервер» стало воз-можным благодаря развитию и широкому внедрению в практику концепции открытых систем. Основным смыслом подхода открытых систем является упрощение комплексирования вычислительных систем за счет международной и национальной стандартизации аппаратных и программных интерфейсов. Это интерфейс позволивший создавать и распространять приложения архитектура клиент-сервер способные работать с широким спектром различных целевых СУБД. О каком интерфейсе идет речь?

4. Если отношение содержит множество кортежей, принадлежащих от-ношению R1 и не принадлежащих R2, то это отношение называется …

5. Какая БД создается с помощью данного кода? Что означает данный программный код:

```
DBD NAME=DB3, ACCESS=HISAM
DATASET DO1=DEPTDO1, DEVICE=3380,
OWFLW=DEPTOVF SEGM NAME=PART, BYTES=32
LCHILD NAME=(COMP_ASSEMB, DB3), PAIR=ASSEMB_
COMP
FIELD NAME=(PS, SEQ), BYTES=5, START=1
FIELD NAME=PD, BYTES=25, START=6
FIELD NAME=CL, BYTES=2, START=31
```

SEGM NAME=ASSEMB_COMP, BYTES=10
POINTER=(LPART, TWIN, LTWIN)
PARENT=(PART), (PART, PHYSICAL, DB3)
FIELD NAME=(PS, SEQ), BYTES=5, START=1
FIELD NAME=OTY, BYTES=5, START=6
SEGM NAME=COMP_ASSEMB, BYTES=10
POINTER=PAIRED
PARENT=PART, SOURCE=(ASSEMB_COMP, DB3)
FIELD NAME=(PS, SEQ), BYTES=5, START=1
FIELD NAME=OTY, BYTES=5, START=6

6. Эта модель основывается на некоторой важной семантической ин-формации о реальном мире. Вводится специальный диаграммный метод как средство проектирования баз данных. Приводится пример проектирования и описания базы данных с использованием этой модели и диаграммного метода. Обсуждаются некоторые аспекты понятий целостности данных, поиска информации и манипуляций с данными. Что за модель?

7. Определите свойства и положения объектно-ориентированной модели данных.

1) способность изменять реализацию любого класса объектов без опасения, что это вызовет нежелательные побочные последствия в программной среде; полиморфизм;
2) возможность создавать из имеющихся классов новые по принципу от общего к частному; наследование;
3) способность объектов выбирать метод обработки на основе типов данных, принимаемых в сообщении.

8. Дана инфологическая модель, найдите ошибку (дефект) и объясните свой ответ.

Рис. А. Инфологическая модель

9. Что выполнялось в процессе этих 3 проектов:

1) проект: была создана система System R — прототип реляционной СУБД, что практически подтвердило применимость

реляционной модели. Создание SQL, коммерческих реляционных СУБД;
2) проект: появилась академическая версия INGRES, отпачковались коммерческие СУБД;
3) проект: имел значение в области применения запросов.

10. Как известно, объекты, выделяемые в качестве сущностей имеют различную классификацию, объясните, что означают понятия:
1) составной объект;
2) агрегированный объект;
3) обобщенный объект.

11. Эта модель данных (МД) была исторически первой структурой БД, видимо, из-за того, что древовидные иерархические структуры широко используются в повседневной человеческой деятельности.

12. Написать программный код для создания базы данных на языке SQL по представленной модели.

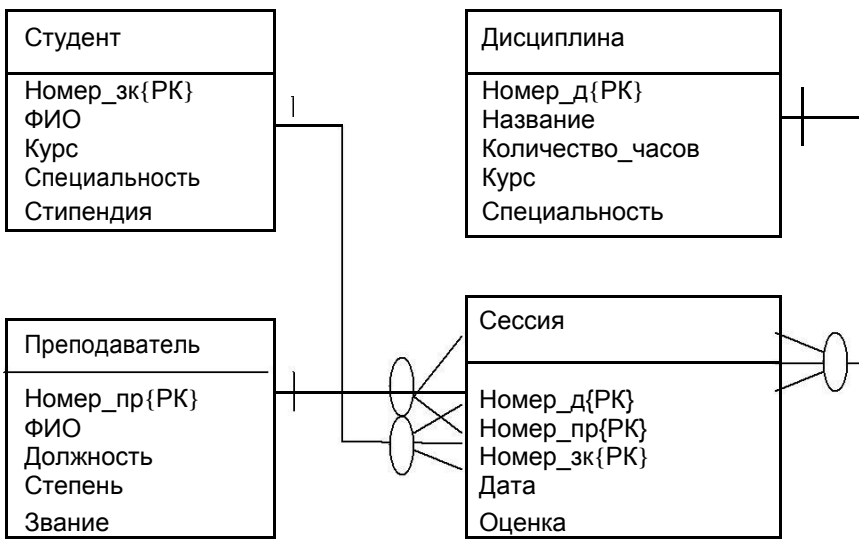


Рис. Б. Модель «Сущность-связь»

Задания для самостоятельной работы по разделу «Базы знаний»

Практическое задание № 1 Создание экспертных систем средствами ПРОЛОГа

Экспертные системы (ЭС) — это сложные программные комплексы, аккумулирующие знания специалистов в конкретных предметных областях и тиражирующие этот эмпирический опыт для консультаций менее квалифицированных пользователей.

Построим небольшую экспертную систему, которая будет опре-делять одну из нескольких рыб по признакам, указанным пользо-вателем. Система будет задавать вопросы и строить логические выводы на основе полученных ответов.

Типичный диалог экспертной системы с пользователем может выглядеть следующим образом (рис. В):

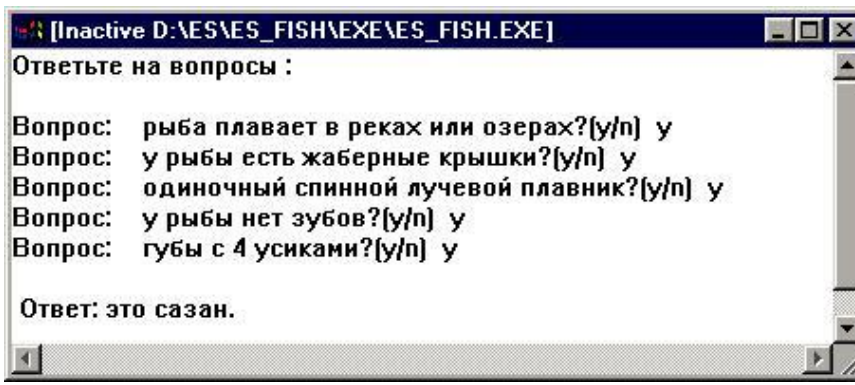


Рис. В. Диалог экспертной системы с пользователем

Первым шагом построения такой системы является обеспече-ние ее знаниями, необходимыми для выполнения рассуждений. Программа должна во время консультаций выводить заключения из информации, имеющейся в базе знаний, а также использовать новую информацию, полученную от пользователя. Поэтому мини-мальная ЭС должна включать:

– базу знаний;
– механизм вывода;
– пользовательский интерфейс.

Разработку любой ЭС следует начать с исследования предметной области. Пусть на основе бесед с экспертом были получены следующие эмпирические правила:

1) ЕСЛИ это отряд
   карпообразные
   И
   у рыбы желто-золотистый окрас
   И
   губы с 4
   усиками ТО это
   сазан
2) ЕСЛИ это отряд
   карпообразные
   И
   у рыбы плавники с розовыми перьями
   ТО
   это плотва
3) ЕСЛИ спинной
   плавник узкий
   И
   у рыбы желто-золотистый окрас
   И
   это отряд карпообразные
   ТО
   это лещ
4) ЕСЛИ
   у рыбы нет зубов
   И
   одиночный спинной лучевой плавник
   И
   это костная рыба
   И
   это пресноводная рыба
   ТО
   это отряд карпообразные
5) ЕСЛИ
   у рыбы есть костный скелет
   ИЛИ у рыбы есть жаберные
   крышки

ТО
это костная рыба

6) ЕСЛИ рыба плавает в
озерах ИЛИ

рыба плавает в реках
ТО это пресноводная
рыба.

Для создания базы знаний используем
предикаты: fish(symbol)
otrajd(symbol)
vid(symbol)
priznak(symbol)
Базу знаний будут составлять следующие
правила: fish(“это сазан”):-
otrajd(“отряд карпообразные”)
priznak(“губы с 4 усиками”)
fish(“это плотва”):-otrajd(“отряд
карпообразные”)
priznak(“плавники с розовыми перьями”)
fish(“это лещ”):-
otrajd(“отряд карпообразные”)
priznak(“у рыбы желто-золотистый окрас”)
priznak(“у рыбы спинной плавник узкий”)

Необходимо предусмотреть, что искомой рыбы в базе знаний нет:

fish(“данной рыбы в базе знаний не
обнаружено”) otrajd(“отряд карпообразные”):-
vid(“пресноводная рыба”)
vid(“костная рыба”)
priznak(“одиночный спинной лучевой плавник”)
priznak(“у рыбы нет зубов”)
vid(“костная рыба”):-
priznak(“у рыбы есть жаберные крышки”)
priznak(“у рыбы есть костный скелет”)
vid(“пресноводная рыба”):-
priznak(“рыба плавает в реках или озерах”)

Для хранения информации, полученной от пользователя, используются предикаты **yes** и **no**, составляющие внутреннюю базу

фактов. Предикат **yes** служит для хранения фактов, соответствующих положительному ответу, а предикат **no** — для хранения отрицательных ответов. То есть предикат **yes** утверждает наличие како-го-либо признака у рыбы, а **no** — отсутствие указанного признака. Эти предикаты объявляются в разделе внутренней базы фактов:

```
global facts
   yes(symbol)
   no (symbol)
```

Добавить новые факты во внутреннюю базу можно с помощью правила add_to_database, состоящего из двух частей. Первая часть добавляет факты, соответствующие положительному ответу (с кла- виатуры вводится «y»). Вторая часть правила добавляет факты, указывающие на отсутствие данного признака у рыбы.

```
add_to_database(Y,'y') :- assertz(yes(Y))
add_to_database(Y,'n') :- assertz(no(Y)), fail
```

Необходимо предусмотреть очистку внутренней базы фактов. Для этого создадим правило:

```
clear_from_database :- retract(yes(_)), fail
clear_from_database :- retract(no(_)), fail
```

Для проверки наличия у рыбы определенного признака создадим правило priznak(Y):

```
priznak(Y) :- yes(Y),!
priznak(Y) :- not(no(Y))
   question(Y)
```

Формулировка вопроса, ввод ответа и сохранение соответствующего правила осуществляется с помощью правил:

```
answer :- fish(X),!, nl
   save(«BF1.dbf»)
   write(«Ответ:»,X,».»), nl
question(Y) :-
   write(«Вопрос:»,Y,»?(y/n)»)
   otvet(X)
   write(X), nl
   add_to_database(Y,X)
otvet(C) :- readchar(C)
```

И, наконец, правило begin, запускающее сеанс консультации:

```
begin :- write("Ответьте на вопросы:"), nl, nl
   answer
   clear_from_database
```

```
   nl, nl, nl,
   nl exit
```
Полный листинг программы выглядит следующим образом:
```
GLOBAL FACTS
   yes(symbol)
     no(symbol)
 PREDICATES
    fish(symbol)
   otrajd(symbol)
   vid(symbol)
   begin
   answer
   question(symbol)
   add_to_database(symbol,char)
   otvet(char)
   clear_from_database
   priznak(symbol)
GOAL
   begin
CLAUSES
    begin:-
        write(«Ответьте на вопросы:»), nl,
        nl answer
        clear_from_database
        nl, nl, nl, nl
        exit
    answer:-
        fish(X),!, nl
        save(«BF1.dbf»)
        write(«Ответ:»,X,».»), nl
    question(Y):-
        write(«Вопрос:»,Y,»?
        «) otvet(X)
        write(X), nl
        add_to_database(Y,X)
    otvet(C):-
        readchar(C)
    priznak(Y):-
        yes(Y),!
```

```
priznak(Y):-
    not(no(Y))
    question(Y)
add_to_database(Y,'y'):-
    assertz(yes(Y))
add_to_database(Y,'n'):-
    assertz(no(Y)), fail
clear_from_database :- retract(yes(_)), fail
clear_from_database :- retract(no(_)), fail
fish("это сазан"):-
    otrajd("отряд карпообразные")
    priznak("губы с 4 усиками")
fish("это плотва"):-
    otrajd("отряд карпообразные")
    priznak("плавники с розовыми перьями")
fish("это лещ"):-
    otrajd("отряд карпообразные")
    priznak("у рыбы желто-золотистый окрас")
    priznak("у рыбы спинной плавник узкий")
fish("данной рыбы в базе знаний не обнаружено")
otrajd("отряд карпообразные"):-
    vid("пресноводная рыба")
    vid("костная рыба")
    priznak("одиночный спинной лучевой плавник")
    priznak("у рыбы нет зубов")
vid("костная рыба"):-
    priznak("у рыбы есть жаберные крышки")
    priznak("у рыбы есть костный скелет")
vid("пресноводная рыба"):-
    priznak("рыба плавает в реках или озерах")
```

Задание для самостоятельной работы:

1. Реализуйте данную программу в среде Visual Prolog и протестируйте ее.

2. Расширьте базу знаний экспертной системы, добавив следующие правила:

1) ЕСЛИ
   у рыбы есть электрические органы
   И

это отряд скаты
ТО
это электрический скат
2) ЕСЛИ
у рыбы на хвосте ядовитый шип
И
это отряд скаты
ТО
это скат-хвостокол
3) ЕСЛИ
у рыбы серо-коричневый окрас
И
у рыбы коническая морда
И
это отряд
акулы ТО
это гиганская акула
4) ЕСЛИ это
отряд акулы
И
рыба нападает на людей
И
у рыбы молотообразная морда
ТО
это рыба молот
5) ЕСЛИ
у рыбы нет хвостового плавника
И
у рыбы тонкий длинный хвост
И
это хрящевая рыба
И
это морская рыба
ТО
это отряд скаты
6) ЕСЛИ это
морская рыба
И
это хрящевая рыба

И
плавники не гибкие
И
хвост ассиметричный
ТО
это отряд акулы

7) ЕСЛИ
у рыбы нет плавательного пузыря
ИЛИ
у рыбы есть хрящевый
скелет ТО это хрящевая
рыба

8) ЕСЛИ
рыба плавает в морях
ТО это морская рыба

3. Протестируйте полученную экспертную систему.

Практическое задание № 2 Разработка интеллектуальной системы анализа данных

Отработку материала практического занятия необходимо проводить с использованием MS SQL 2000 и MS Access.

OLAP и многомерные кубы

В качестве примера реляционной базы данных используется Northwind, входящая в комплекты поставки Microsoft SQL Server или Microsoft Access и представляющей собой типичную базу дан-ных, хранящую сведения о торговых операциях компании, занима-ющейся оптовыми поставками продовольствия. К таким данным относятся сведения о поставщиках, клиентах, компаниях, осуществ- ляющих доставку, список поставляемых товаров и их категорий, данные о заказах и заказанных товарах, список сотрудников компа-нии. Подробное описание базы данных Northwind можно найти в справочных системах Microsoft SQL Server или Microsoft Access.

Запрос, в результате которого получаются подробные сведения о всех заказанных товарах и выписанных счетах из представления Invoices и таблиц Products и Categories:

```
SELECT dbo.Invoices.Country
dbo.Invoices.City
```

dbo.Invoices.CustomerName
dbo.Invoices.Salesperson
dbo.Invoices.OrderDate
dbo.Categories.CategoryName
dbo.Invoices.ProductName
dbo.Invoices.ShipperName
dbo.Invoices.ExtendedPrice FROM
dbo.Products INNER JOIN
dbo.Categories ON dbo.Products.CategoryID = dbo.Categories.CategoryID- INNER JOIN
dbo.Invoices ON dbo.Products.ProductID = dbo.Invoices.ProductID B

Access 2000 аналогичный запрос имеет вид:

```
SELECT Invoices.Country, Invoices.City
Invoices.Customers.CompanyName AS
CustomerName, Invoices.Salesperson
Invoices.OrderDate, Categories.CategoryName
Invoices.ProductName
Invoices.Shippers.CompanyName AS
ShipperName, Invoices.ExtendedPrice
FROM Categories INNER JOIN (Invoices
INNER JOIN Products ON Invoices.ProductID =
Products.ProductID) ON Categories.CategoryID =
Products.CategoryID
```

Этот запрос обращается к представлению Invoices, содержащему сведения обо всех выписанных счетах, а также к таблицам Categories и Products, содержащим сведения о категориях продуктов, которые заказывались, и о самих продуктах соответственно. В результате этого запроса получен набор данных о заказах, включающий категорию и наименование заказанного товара, дату размещения заказа, имя сотрудника, выписавшего счет, город, страну и название компании-заказчика, а также наименование компании, отвечающей за доставку. Сохраненный запрос Invoices1 представлен на рис. Г.

На основе Invoices1 могут быть получены ответы на следующие вопросы путем выдачи соответствующих запросов:

SELECT SUM (ExtendedPrice) FROM invoices1 WHERE Coun- try='France' — какова суммарная стоимость заказов, сделанных клиентами из Франции?

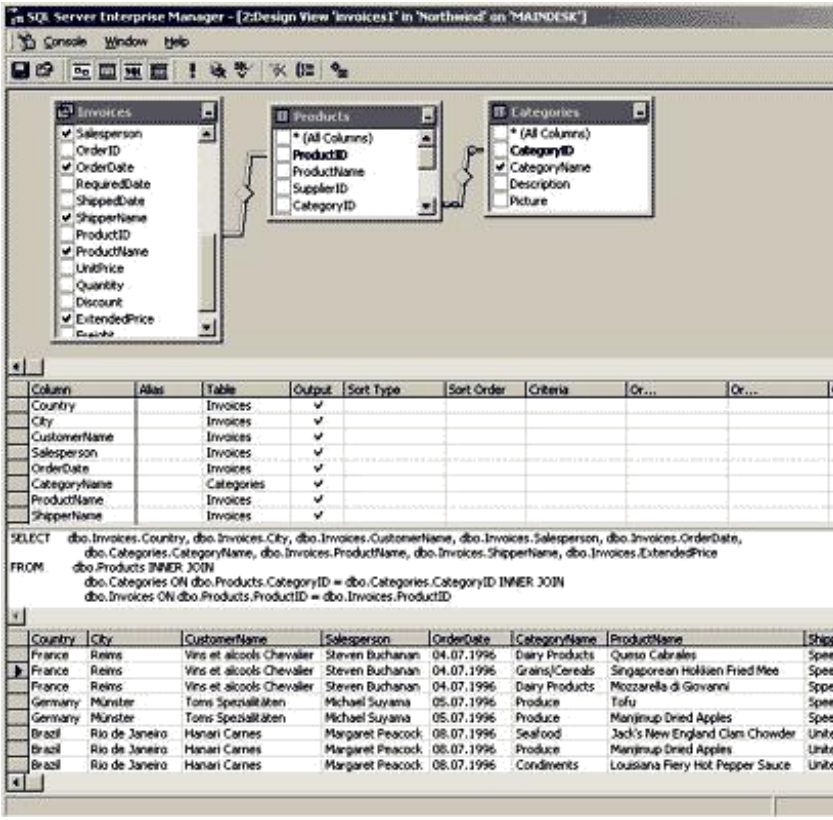


Рис. Г. Результат обращения к представлению Invoices

SELECT SUM (ExtendedPrice) FROM invoices1 WHERE Coun- try='France' AND ShipperName='Speedy Express' — какова суммар- ная стоимость заказов, сделанных клиентами из Франции и до-ставленных компанией Speedy Express?

SELECT SUM (ExtendedPrice) FROM Ord_pmt WHERE CompanyName='Speedy Express' AND OrderDate BETWEEN 'December 31, 1995' AND 'April 1, 1996' AND ShipperName='Speedy Express' — какова суммарная стоимость заказов, сделанных клиентами из Франции в 1997 г. и доставленных компанией Speedy Express?

Результатом любого из перечисленных выше запросов является число. Если в первом из запросов заменить параметр 'France' на 'Austria' или на название иной страны, можно снова выполнить этот запрос и получить другое число. Выполнив эту процедуру со всеми странами, получается следующий набор данных (фрагмент):

Country	SUM (ExtendedPrice)
Argentina	7327.3
Austria	110788.4
Belgium	28491.65
Brazil	97407.74
Canada	46190.1
Denmark	28392.32
Finland	15296.35
France	69185.48
Germany	209373.6
…	…

Полученный набор агрегатных значений (в данном случае — сумм) может быть интерпретирован как одномерный набор данных. Этот же набор данных можно получить и в результате запроса с предложением GROUP BY следующего вида:

SELECT Country, SUM (ExtendedPrice) FROM invoices1 GROUP BY Country

Второй из приведенных выше запросов содержит два условия в предложении WHERE. Если выполнять этот запрос, подставляя в него все возможные значения параметров Country и ShipperName, получается двухмерный набор данных следующего вида (фрагмент):

Country	Federal Shipping	Speedy Express	United Package
Argentina	1 210.30	1 816.20	5 092.60
Austria	40 870.77	41 004.13	46 128.93
Belgium	11 393.30	4 717.56	17 713.99
Brazil	16 514.56	35 398.14	55 013.08
Canada	19 598.78	5 440.42	25 157.08
Denmark	18 295.30	6 573.97	7 791.74
Finland	4 889.84	5 966.21	7 954.00
France	28 737.23	21 140.18	31 480.90
Germany	53 474.88	94 847.12	81 962.58
…	…	…	…

Такой набор данных называется сводной таблицей (pivot table) или кросс-таблицей (cross table, crosstab). Создавать подобные таблицы позволяют многие электронные таблицы и настольные СУБД — от Paradox для DOS до Microsoft Excel 2000. Вот так, например, выглядит подобный запрос в Microsoft Access 2000:

TRANSFORM Sum(Invoices1.ExtendedPrice) AS SumOfExten- dedPrice
SELECT Invoices1.Country
FROM Invoices1
GROUP BY Invoices1.Country
PIVOT Invoices1.ShipperName;

Агрегатные данные для подобной сводной таблицы можно получить и с помощью обычного запроса GROUP BY:

SELECT Country,ShipperName, SUM (ExtendedPrice) FROM in-voices1 GROUP BY COUNTRY,ShipperName

Однако что результатом этого запроса является не сама сводная таблица, а лишь набор агрегатных данных для ее построения (фрагмент):

Country	ShipperName	SUM (ExtendedPrice)
Argentina	Federal Shipping	845.5
Austria	Federal Shipping	35696.78
Belgium	Federal Shipping	8747.3
Brazil	Federal Shipping	13998.26
…	…	…

Третий из рассмотренных выше запросов имеет уже три параметра в условии WHERE. Варьируя их, можно получить трехмерный набор данных (рис. Д).

1998
1997
1996

	Federal Shipping	Speedy Express	United Package
Argentina	11806.28	9190.48	1263.9
Austria			4039.5
Belgium	1745.42	1207.28	14924.12
Brazil			5208.28
Canada	2952.4		
Denmark	1739.76	1376	
Finland	5470.98	3538.92	2328.46
France	11927.48	9823.43	11052.28
Germany	2208.62	1739.6	4681.16
Ireland		330.9	608
Italy	2139.1		1357.6
Mexico			786
Norway	459		
Poland	1268.3	716.72	285.12
Portugal	236.5	220.3	2235.8
Spain	3021.23	2380	1488.8
Sweden	2490.5		1628.32
Switzerland	5094.88	1520.8	901.2
UK	11192.65	6347.52	14091.93
USA	3925.58	3171.92	
Venezuela	11806.28	9190.48	1263.9

Рис. Д. Трехмерный набор агрегатных данных

Ячейки куба, показанного на рис. 6.3, содержат агрегатные данные, соответствующие находящимся на осях куба значениям параметров запроса в предложении WHERE.

Можно получить набор двухмерных таблиц с помощью сечения куба плоскостями, параллельными его граням (для их обозначения используют термины cross-sections и slices).

Очевидно, что данные, содержащиеся в ячейках куба, можно получить и с помощью соответствующего запроса с предложением GROUP BY. Кроме того, некоторые электронные таблицы (в частности, Microsoft Excel 2000) также позволяют построить трехмерный набор данных и просматривать различные сечения куба, параллельные его грани, изображенной на листе рабочей книги (workbook).

Если в предложении WHERE содержится четыре или более параметров, результирующий набор значений (также называемый OLAP-кубом) может быть 4-мерным, 5-мерным и т. д.

Наряду с суммами в ячейках OLAP-куба могут содержаться результаты выполнения иных агрегатных функций языка SQL, таких как MIN, MAX, AVG, COUNT, а в некоторых случаях — и других (дисперсии, среднеквадратичного отклонения и т. д.). Для описа-ния значений данных в ячейках используется термин summary (в общем случае в одном кубе их может быть несколько), для обо-значения исходных данных, на основе которых они вычисляют-ся, — термин measure, а для обозначения параметров запросов — термин dimension (переводимый на русский язык обычно как «из-мерение», когда речь идет об OLAP-кубах, и как «размерность», когда речь идет о хранилищах данных). Значения, откладываемые на осях, называются членами измерений (members).

Говоря об измерениях, следует упомянуть о том, что значения, наносимые на оси, могут иметь различные уровни детализации. Например, может интересовать суммарная стоимость заказов, сделанных клиентами в разных странах, либо суммарная стоимость заказов, сделанных иногородними клиентами или даже отдельными клиентами. Естественно, результирующий набор агрегатных данных во втором и третьем случаях будет более детальным, чем в первом. Возможность получения агрегатных данных с различной степенью детализации соответствует одному из требований, предъявляемых к хранилищам данных, — требованию доступности различных срезов данных для сравнения и анализа.

Задание для самостоятельной работы:

Составить OLAP-куб, по выбранной студентом предметной области и идентифицировать результаты.

Практическая работа № 3 Разработка вопросно-ответной системы с использованием логической модели представления знаний

В качестве инструментальной среды для начального изучения принципов работы Пролога используется ArityProlog 6.0. Его запуск осуществляется командой API.EXE из папки API. В папке DOC находится подробное описание языка и самого интерпретатора.

Начальным этапом разработки любой информационной системы (ИС) является изучение и анализ предметной области и постановка задачи. Экспертные ИС не являются исключением. Более того, максимально полное и конкретное знание области применения создаваемой ИС является залогом ее успешности и полезности. Поэтому ведущую роль при создании экспертных ИС играет сам эксперт, и не менее важную — инженер по знаниям.

Например, требуется создать экспертную систему, осуществляющую выбор ноутбука по следующим характеристикам, задаваемым пользователем:

- тактовая частота процессора;
- объем оперативной памяти;
- объем жесткого диска;
- наличие/отсутствие модуля Wi-Fi;
- фирма-производитель;
- ценовая категория.

В качестве ответа экспертной системе необходимо получить название модели ноутбука или ноутбуков, в случае если заданным параметрам удовлетворяет несколько таких.

В связи с целями решения поставленной задачи интерфейс диалога с пользователем будет выглядеть достаточно примитивно и, возможно, не иметь достаточной гибкости. Несмотря на это, он является вполне функциональным и может быть признан работоспособным.

Для запуска экспертной системы в окне интерпретатора необходимо набрать «vibor». В начале работы она приветствует пользователя и дает ему необходимые указания по работе с ней.

Это реализуется с помощью набора предикатов «write», выводящих необходимые сообщения на экран пользователя, разделенных предикатами «nl», обеспечивающих построчный вывод подсказок:

vibor :- write(‘Привет! Вы оказались в экспертной системе.’), nl, write(‘Будем выбирать ноутбук.’), nl, write(‘Подсказка 1: для выбора подходящего варианта введите?’), nl, write(‘соответствующий ему номер, поставьте точку и нажмите Enter.’), nl, write(‘Подсказка 2: при отсутствии подходящего следует набрать 0 и точку.’), nl.

«Vibor» — это цель экспертной системы.

Диалог представляет собой последовательность вопросов пользователю для определения характеристик, используемых при выборе ноутбука.

Вопрос представляет собой непосредственно вопросительное предложение, а также пронумерованные варианты ответов. Выбор подходящего осуществляются указанием его номера. После ввода номера варианта ответа необходимо набрать символ «.» (точка) и нажать клавишу «Enter».

Для обеспечения минимальной гибкости выбора параметров, имеется вариант с номером «0», который подразумевает неважность параметра ноутбука при выборе.

Средствами языка «Пролог» вывод вопроса, получение ответа реализуется следующими предикатами:

```
question1(Param1) :- write(‘Наличие Wi-Fi:’),
nl write(‘1 - есть Wi-Fi’), nl
write(‘2 - нет Wi-Fi’), nl
read(Vv1),
vwf(Vv1,Param1) vwf(0,X)
vwf(1,’1’)
vwf(2,’0’)
```

Question1 — клоз, возвращающий значение первой выбираемой характеристики.

Первый предикат write выводит на экран сообщение пользователю с описанием выбираемой характеристики.

Две последующих «write» предлагают варианты ответа.

Предикат «read» считывает номер выбранного варианта. Далее следует вызов правила, устанавливающего соответствие между номером выбранного варианта и конкретным значением параметра.

Это возможно благодаря описанию возможных соответствий в последующем далее блоке из трех правил. С их помощью устанавливается соответствие между номером выбранного варианта и конкретным значением параметра.

Таким образом, на выходе «question1» в переменной Param1 имеется одно из трех значений характеристики, связанной с наличием/отсутствием Wi-Fi-модуля:

1 — есть модуль Wi-Fi;

2 — нет модуля Wi-Fi;

0 — параметр не имеет значение.

Аналогичным образом строятся выборы значений оставшихся пяти характеристик.

Следующая последовательность предикатов обеспечивает последовательный вывод всех вопросов пользователю и получение ответов:

question1(Param1), question2(Param2), question3(Param3), question4(Param4),- question5(Param5), question6(Param6).

Следующим шагом, после того как получены значения характеристик в переменных Param1, Param2, Param3, Param4, Param5, Param6, необходимо выполнить поиск соответствующего(-их) этим критериям ноутбука(-ов). Для этого предназначен вызов правила predmet с передачей в него полученных параметров:

predmet(Param1,Param2,Param3,Param4,Param5,Param6, Result). Пример описания правила «predmet»:

predmet('1','2','250','2048','Acer','60000','ACER AS9920G-302G25').

Данное правило записывает в переменную Result ответ экспертной системы. Если найден соответствующий критериям предмет, то в переменную записывается его название, стоящее последним в описании правила. В случае если ответов несколько, в Result последовательно будут записываться выбранные названия предметов. Это обеспечивает следующей последовательностью предикатов:

write('Результат выбора ноутбука по параметрам:'), nl,!, predmet(Param1, Param2, Param3, Param4, Param5, Param6, Result), write(Result), nl, fail.

Конструкция из предикатов «!» и «fail» позволяет вывести все подходящие варианты, поскольку заключенный между ними вызов правила predmet и вывод полученного результата Result будет выполняться то тех пор, пока не останется удовлетворяющих параметрам результатов. После вывода всех возможных вариантов система выдаст «no».

В случае если соответствующий выбранным критерия ноутбук не найден, система выдаст «no».

На этом работа экспертной системы завершается. Полный текст Пролог-программы без базы правил представлен ниже.

```
question1(Param1) :- write('Наличие Wi-Fi:'),
nl write('1 - есть Wi-Fi'), nl
write('2 - нет Wi-Fi'), nl
read(Vv1),
vwf(Vv1,Param1) vwf(0,X)
vwf(1,'1')
vwf(2,'0')

question2(Param2) :- write('Частота процессора:'),
nl write('1 - менее 1.5 GHZ'), nl
write('2 - от 1.5 до 2 GHz'),
nl write('3 - от 2 до 2.5
GHz'), nl write('4 - более 2.5
GHz'), nl read(Vv2),
vghz(Vv2,Param2) vghz(0,X)
vghz(1,'1')
vghz(2,'1.5')
vghz(3,'2')
vghz(4,'2.5')

question3(Param3) :- write('Объем HDD:'),
nl write('1 - 80'), nl
write('2 - 120'), nl
write('3 - 160'), nl
write('4 - 200'), nl
write('5 - 250'), nl
write('6 - 300'), nl
write('7 - 400'), nl
write('8 - 500'), nl
read(Vv3),
vhdd(Vv3,Param3) vhdd(0,X)
vhdd(1,'80')
vhdd(2,'120')
vhdd(3,'160')
vhdd(4,'200')
vhdd(5,'250')
```

```
vhdd(6,’300’)
vhdd(7,’400’)
vhdd(8,’500’).

question4(Param4) :- write(‘Объем ОЗУ:’),
nl write(‘1 - 512’), nl
write(‘2 - 1024’), nl
write(‘3 - 2048’), nl
write(‘4 - 3072’), nl
write(‘5 - 4096’), nl
read(Vv4), vram(Vv4,Param4)
vram(0,X)
vram(1,’512’)
vram(2,’1024’)
vram(3,’2048’)
vram(4,’3072’)
vram(5,’4096’)

question5(Param5) :- write(‘Фирма-производитель:’),
nl write(‘1 - Acer’), nl
write(‘2 - Asus’), nl
write(‘3 - Dell’), nl
write(‘4 - Fujitsu-Siemens’),
nl write(‘5 - HP’), nl
write(‘6 - Lenovo’), nl
write(‘7 - LG’), nl
write(‘8 - MSI’), nl
write(‘9 - Samsung’), nl
write(‘10 - SONY’), nl
write(‘11 - Toshiba’), nl
read(Vv5), vcompany(Vv5,Param5)
vcompany(0,X)
vcompany(1,’Acer’)
vcompany(2,’Asus’)
vcompany(3,’Dell’)
vcompany(4,’Fujitsu-Siemens’)
vcompany(5,’HP’)
vcompany(6,’Lenovo’)
vcompany(7,’LG’)
```

```
vcompany(8,'MSI')
vcompany(9,'Samsung')
vcompany(10,'SONY')
vcompany(11,'Toshibag')

question6(Param6) :- write('Ценовая категория:'),
nl write('1 - до 20 000 руб.'), nl
write('2 - от 20 000 до 30 000 руб.'),
nl write('3 - от 30 000 до 40 000
руб.'), nl write('4 - от 40 000 до 50
000 руб.'), nl, write('5 - от 50 000 до
60 000 руб.'), nl write('6 - от 60 000
до 70 000 руб.'), nl write('7 - более
70 000 руб.'), nl read(Vv6),
vprice(Vv6,Param6) vprice(0,X)
vprice(1,'0')
vprice(2,'20000')
vprice(3,'30000')
vprice(4,'40000')
vprice(5,'50000')
vprice(6,'60000')
vprice(7,'70000')

vibor :- write('Привет! Вы оказались в экспертной системе.'),
nl write('Будем выбирать ноутбук.'), nl
write('Подсказка 1: для выбора подходящего варианта введи-
те?'), nl
write('соответствующий ему номер, поставьте точку и
нажмите Enter.'), nl
write('Подсказка 2: при отсутствии подходящего следует на-
брать 0 и точку.'), nl
question1(Param1),question2(Param2),question3(Param3),
question4(Param4), question5(Param5), question6(Param6),
write('Результат выбора ноутбука по параметрам:'), nl,!
predmet(Param1,Param2,Param3,Param4,Param5,Param6,Result),
write(Result), nl, fail

predmet('1','2','250','2048','Acer','60000','ACER AS9920G-302G25')
```

Задание для самостоятельной работы:

1. Разработка вопросно-ответной системы осуществляется индивидуально каждым студентом.

2. Студент должен выбрать произвольную предметную, в которой он является экспертом и согласовать ее с преподавателем.

3. Количество правил не менее 100.

4. Вопросно-ответная система должна быть разработана на языке Пролог и функционировать в среде ArityProlog 6.0.

5. Вопросы, задаваемые вопросно-ответной системой, должны быть осмысленны на русском языке.

6. Для объяснения полученных результатов целесообразно использовать встроенный отладчик.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. http://www.mstu.edu.ru/education/materials/kulikova/book3.html [Электронный ресурс].

2. http://www.tspu.tula.ru/ivt/old_site/umr/info_net/lek/lek02.htm [Электронный ресурс].

3. Microsoft Access 2000: учебный курс / Робинсон С. СПб: Пи- тер, 2000.

4. Microsoft SQL Server в клиент-серверных технологиях / Мер- дина- О. Д., Шленов В. В. СПб.: Изд-во СПбГИЭУ, 2003.

5. *Конолли Т., Бегг К.* Базы данных. Проектирование, реализация, сопровождение. Теория и практика. 3-е изд. / пер. с англ. М.: Изд. дом «Вильямс», 2003.

6. Microsoft SQL Server / Маммаев Е. В. СПб.: БХВ-Петербург, 2005.

7. *Robinson H.* Database Architectures // Relational database systems, Introduction to database technology. Milton Keynes: Open University. P. 20–31.

8. http://rema.44.ru/resurs/study/dblectio/dblect2.html [Электронный ресурс].

9. Базы данных: модели, разработка, реализация / Карпова Т. С. СПб.: Питер, 2001.

10. *Бойко В. В., Савинков В. М.* Проектирование баз данных информационных систем. 2-е изд. М.: Финансы и статистика, 1989.

11. *Голицына О. Л.* Базы данных: учеб. пособие. М.: Форум, 2006.

12. *Диго С. М.* Базы данных: проектирование и использование. М.: Финансы и статистика, 2005.

13. *Евдокимов В. В.* [и др.]. Экономическая информатика: учебник для вузов / под ред. д. э. н., проф. В. В. Евдокимова. СПб.: Пи- тер, 1997.

14. *Карпова Т. С.* Базы данных: модели, разработка, реализация. СПб.: Питер, 2001.

15. *Когаловский М. Р.* Энциклопедия технологий баз данных. М.: Финансы и статистика, 2002.

16. *Корнеев В. В., Гареев А. Ф., Васютин С. В., Райх В. В.* Базы данных. Интеллектуальная обработка информации. М.: Издатель Молгачева С. В.; Нолидж, 2001.

17. *Кренке Д.* Теория и практика построения БД. СПб.: Питер, 2003.

18. *Кузнецов С. Д.* Основы баз данных.: учеб. пособие. М.: ИНСТИТУ, 2005.

19. *Марков А. С., Лисовский К. Ю.* Базы данных. Введение в теорию и методологию: учебники. М.: Финансы и статистика, 2004.

20. Программирование в среде Access 2000. Энциклопедия поль- зователя / Форт Стивен, Том Хоуи, Релстон Джеймс / пер. с англ. К.: ДиаСофт, 2000.

21. *Пушников А. Ю.* Введение в системы управления базами данных. Часть 1. Реляционная модель данных: учеб. пособие. Уфа: Изд-во Башкирского ун-та, 1999.

22. *Пушников А. Ю.* Введение в системы управления базами данных. Часть 2. Нормальные формы отношений и транзакции: учеб. пособие. Уфа: Изд-во Башкирского ун-та, 1999.

23. *Ревунков Г. И., Самохвалов Э. Н., Чистов В. В.* Базы и банки данных и знаний: учебник для вузов / под ред. В. Н. Четверикова. М.: Высшая школа, 1992.

24. *Тиори Т., Фрай Дж.* Проектирование структур баз данных: в 2 кн. / пер. с англ. М.: Мир, 1985.

25. *Уткин В. Б., Балдин К. В.* Информационные системы и технологии в экономике: учебник для вузов. М.: ЮНИТИ-ДАНА, 2003.

26. Эффективная работа с Microsoft Access 2000 / Дж. Вейскас. СПб.: Питер, 2001.

27. *Мигас С. С.* Интеллектуальные информационные системы: конспект лекций: метод. издание. СПб.: Изд-во СПбГИЭУ, 2009.

28. *Прохорова О. В.* Интеллектуальные информационные системы: конспект лекций. Самара: Изд-во СГАСУ, 2010.

29. http://www.stgau.ru [Электронный ресурс].

30. http://infosphere.narod.ru/files/monografy/dubrovsky/chap5.htm [Электронный ресурс].

СОДЕРЖАНИЕ

Введение 3
Тема 1. Понятие информации 4
Тема 2. Введение в базы данных 14
Тема 3. Принципы построения БД 36
Тема 4. Даталогические модели данных 48
Тема 5. Операции реляционной алгебры 68
Тема 6. Основы проектирования реляционных СУБД 83
Тема 7. Физические модели 108
Тема 8. Организация процессов обработки данных 114
Тема 9. Принципы поддержки целостности 116
Тема 10. Распределенные БД 120
Тема 11. Язык SQL 124
Тема 12. Безопасность данных 127
Тема 13. Формализация знаний в ИС 133
Тема 14. Модели представления знаний 139
Тема 15. Экспертные системы 156
Задания для проверки знаний 163
Библиографический список 197

САНКТ-ПЕТЕРБУРГСКИЙ УНИВЕРСИТЕТ
УПРАВЛЕНИЯ И ЭКОНОМИКИ

Татьяна Анатольевна Черняк
Светлана Вячеславовна Удахина
Мария Александровна Косухина

ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ СУБЪЕКТОВ ЭКОНОМИЧЕСКОЙ ДЕЯТЕЛЬНОСТИ:
Базы данных и знаний

Учебное пособие

Заведующий редакцией научной и учебно-методической литературы
Издательства СПбУУиЭ
А. В. Блажко
Подписано в печать 28.05.2015 г.
Формат 60×84 1/16. Уч.-изд. л. 10,32. Усл. печ. л. 12,75.
Гарнитура Minion Pro. Тираж 600 экз. Заказ № 039

Издательство Санкт-Петербургского университета управления и экономики
198103, Санкт-Петербург, Лермонтовский пр., д. 44
E-mail: izdat-ime@spbume.ru, izdat-ime@yandex.ru
Отпечатано в типографии ООО «РАЙТ ПРИНТ ГРУПП»
198095, Санкт-Петербург, ул. Розенштейна, д. 21