



В. Ю. Пирогов

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ

организация и проектирование



В. Ю. Пирогов

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ организация и проектирование

Рекомендовано УМО в области инновационных
междисциплинарных образовательных программ в качестве учебного пособия
по специальности 010503 «Математическое обеспечение
и администрирование информационных систем»

Санкт-Петербург
«БХВ-Петербург»
2009

УДК 681.3.06
ББК 32.973.26-018.2
ПЗЗ

Пирогов В. Ю.

ПЗЗ Информационные системы и базы данных: организация и проектирование: учеб. пособие. — СПб.: БХВ-Петербург, 2009. — 528 с.: ил. — (Учебная литература для вузов)

ISBN 978-5-9775-0399-0

Излагаются основные вопросы по организации и проектированию информационных систем: классификация, структура, безопасность и принципы проектирования; а также архитектура информационной системы: интерфейсы и протоколы, клиентские приложения. Большое внимание уделяется базам данных и их программному управлению, языкам SQL и QBE. Приводятся примеры новых технологий в области баз данных.

*Для студентов вузов, специалистов и руководителей,
применяющих информационные системы*

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Татьяна Лапина</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Анна Кузьмина</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Елены Беляевой</i>
Фото	<i>Кирилла Сергеева</i>
Зав. производством	<i>Николай Тверских</i>

Рецензенты:

Б. А. Новиков, д. ф.-м. н., профессор, заведующий лабораторией исследования операций математико-механического факультета Санкт-Петербургского государственного университета;

В. А. Костин, к. ф.-м. н., доцент кафедры информатики Санкт-Петербургского государственного университета.

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.07.09.

Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 42,57.

Тираж 1500 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"

199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0399-0

© Пирогов В. Ю., 2009

© Оформление, издательство "БХВ-Петербург", 2009

Оглавление

Введение	10
Содержание книги.....	11
Благодарности	13
Глава 1. Информационные системы	14
1.1. Компьютерная техника и информационные системы.....	14
1.2. Понятие "информационная система"	16
1.3. Системный подход и информационная система	19
1.4. О некоторых терминах	21
1.5. Классификации информационных систем	21
Типы ИС	21
Типы данных ИС.....	24
1.6. Функции ИС	25
1.7. Структура ИС	27
Общая структура ИС	27
Файл-серверная архитектура	28
Клиент-серверная архитектура	29
Многоуровневые ИС	31
1.8. Безопасность информационных систем	32
Основные понятия	32
Классификации угроз	34
Угрозы доступности информации.....	35
Угрозы целостности информации	38
Угрозы конфиденциальности информации	39
Идентификация и аутентификация.....	40
Разграничение доступа	42
Шифрование.....	43
Электронная подпись	44
Туннелирование	44
Обзор стандартов безопасности	45
Законодательное регулирование.....	48
1.9. Вопросы для самопроверки.....	52
Глава 2. Базы данных как часть информационной системы.....	53
2.1. Базы данных	53
Общие понятия.....	53
СУБД.....	55

Модели данных	56
Файловая модель	56
Сетевая модель.....	58
Иерархическая модель	60
Реляционная модель	62
Объектная и объектно-реляционная модели	62
2.2. Основы теории реляционных баз данных	63
Исторические заметки.....	63
Основные положения реляционной теории баз данных.....	66
Основные понятия	66
Некоторые выводы	72
Ключи	73
Типы таблиц.....	77
О "значении" <i>NULL</i>	78
Правила Кодда	81
Реляционная алгебра	86
Унарные операции.....	86
Бинарные операции	88
Еще о реляционной алгебре.....	98
Реляционное исчисление.....	99
Реляционное исчисление кортежей.....	99
Реляционное исчисление доменов	101
Нормальные формы	102
Избыточность данных и аномалии модификации.....	105
Декомпозиция	106
Функциональная зависимость в отношении.....	109
Первая нормальная форма	111
Вторая нормальная форма	115
Третья нормальная форма.....	117
Нормальная форма Бойса — Кодда	119
Четвертая нормальная форма и множественные зависимости	121
Пятая нормальная форма	122
Резюме	123
Связи между таблицами	124
Связь "один-к-одному"	124
Связь "один-ко-многим"	127
Другие типы связи	129
Реляционная целостность.....	135
Денормализация.....	138
2.3. Вопросы для самопроверки.....	141
Глава 3. Принципы проектирования ИС.....	143
3.1. О проектировании информационных систем	143
Некоторые определения и термины	143
Общие замечания	144

Общие требования к разрабатываемым информационным системам	146
Достоверность информации	146
Оперативность результатов	149
Соответствие уровню руководства	149
Системный подход.....	150
Обеспечение безопасности информации	150
Общие принципы разработки информационных систем.....	150
Централизованность разработки	151
Системность	151
Конкретность	151
Участие заказчика.....	151
Возможность модернизации разрабатываемой системы	152
Сопровождение системы.....	152
Учет требований безопасности.....	152
Совместимость.....	153
Стандартизация и унификация	153
Технологии проектирования.....	153
3.2. Жизненный цикл ИС.....	155
Процессы жизненного цикла	155
Основные процессы	156
Вспомогательные процессы.....	156
Организационные процессы	156
Модели жизненного цикла.....	157
Каскадная модель	157
Обзор этапов жизненного цикла в каскадной модели	161
V-образная каскадная модель.....	170
Спиральная модель.....	172
Прототипирования.....	174
RAD-технология	176
Краткий обзор других технологий разработки ИС	178
Технология RUP	178
Технология MSF	179
Технология CDM	180
Технология XP.....	182
3.3. Проектирование баз данных	184
Об этапах проектирования БД.....	185
Понятие сущности. Типы сущностей.....	186
Основные понятия	188
Система диаграмм	190
Правила порождения	193
Другие элементы ER-модели	195
CASE-средства	198
Общие понятия.....	198
Структура	198
Классификация CASE-средств	199

3.4. Моделирование предметной области	200
Функциональные диаграммы	200
Диаграммы потоков данных	204
О проектировании на основе языка UML	207
Общие сведения	207
Диаграммы прецедентов	208
Диаграммы классов	209
CASE-средства, поддерживающие язык UML	214
3.5. Проектирование пользовательского интерфейса	215
О пользовательском интерфейсе	215
Общие соображения	215
Стили пользовательского интерфейса	217
Критерии эффективности пользовательского интерфейса	218
Принципы и стандарты	219
Источники пользовательского интерфейса	219
Руководящие принципы и проектирование пользовательского интерфейса	220
Некоторые правила проектирования пользовательского интерфейса	222
Этапы	225
3.6. Вопросы для самопроверки.....	226

Глава 4. Программное управление реляционными базами данных.

Язык SQL.....227

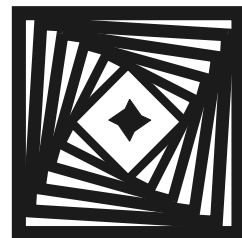
4.1. О языках управления базами данных	227
Общие замечания	227
Пример процедурного языка управления базами данных	229
4.2. О языке SQL	233
Историческое введение	233
Стандарты SQL	234
Схема выполнения команды SQL.....	236
Формы языка SQL.....	237
Интерактивный SQL.....	238
Статический и встраиваемый SQL	240
Динамический SQL.....	248
Расширения SQL. Диалекты	252
4.3. Описание языка SQL.....	254
Общие положения.....	254
Типы данных	254
Элементы языка SQL.....	257
Подмножество DML (SQL).....	259
Вставка строк	259
Обновление строк	261
Удаление строк	263
Команда выборки.....	263

Подмножество DDL (SQL).....	296
Создание базы данных	296
Создание таблиц (<i>CREATE TABLE</i>).....	298
Создание таблицы на основе запроса <i>SELECT</i>	301
Изменение структуры таблиц (<i>ALTER TABLE</i>)	302
Удаление таблиц (<i>DROP TABLE</i>)	303
Представления	303
Программные объекты базы данных	305
Объекты безопасности	306
Общие положения.....	306
Назначение привилегий	308
Отмена привилегий.....	310
Группы и схемы	311
4.4. Словарь базы данных.....	313
4.5. Язык QBE.....	315
4.6. О стандарте SQL99 (SQL3)	321
Уровни соответствия	321
Документы стандарта	324
Новые типы данных.....	325
Расширения операторов SQL.....	326
Процедурные расширения	326
Элементы объектного подхода	327
4.7. Вопросы для самопроверки.....	328
Глава 5. СУБД.....	330
5.1. О СУБД.....	330
Достоинства и недостатки СУБД	330
Преимущества.....	330
Недостатки	333
Функции СУБД	334
Типовая организация современной СУБД.....	338
Уровни СУБД.....	339
5.2. Низкоуровневая организация СУБД	340
Структура баз данных низкого уровня.....	341
Oracle	341
MS SQL Server	344
PostgreSQL.....	353
Технология доступа к данным	357
Хэширование.....	357
Индексы	359
Секционирование.....	367
Кластеризация в Oracle	372
5.3. Основы теории транзакций	373
Транзакция	373
Понятие транзакции	373

Свойства транзакций	374
Программное управление транзакциями	375
Журнал транзакций	377
Журнал транзакций и буферизация	378
Транзакции в многопользовательском режиме	381
Параллельное выполнение и конфликты транзакций	381
Графики выполнения транзакций	383
Распределенные транзакции	385
Блокировки	387
Типы блокировок	387
Конфликты блокировок (выход из тупиковых ситуаций)	390
Уровни изоляции	392
Другие средства устранения конфликтов транзакций	393
5.4. Резервное копирование	394
Стратегия резервного копирования и восстановления	394
Общие соображения	394
Устройства копирования	397
Что копировать	398
Время копирования (расписание)	400
Восстановление данных	403
5.5. Основы программирования на стороне СУБД	404
Принципы программирования на стороне СУБД	405
Хранимые процедуры и функции	406
Типы хранимых процедур	406
Структура хранимых процедур и функций	407
О расширениях языка SQL	414
Триггеры	414
5.6. Вопросы для самопроверки	419
Глава 6. Архитектура ИС	421
6.1. Интерфейсы и протоколы	421
Модель OSI	421
Протокол ODBC	425
Общие положения	425
Архитектура и настройка	426
Функции API ODBC	430
Примеры программ	431
Другие интерфейсы и протоколы	440
Интерфейсы доступа к базам данных	440
Интерфейсы Java	448
Протоколы семейства TCP/IP	452
6.2. Клиентские приложения (средства построения и архитектура)	462
О разработке клиентского приложения	463
Выбор системы программирования	463
Средства отображения табличных данных	465

Построение ИС на основе Web-сервера	467
О протоколе HTTP.....	468
О формате данных XML	469
Технология CGI	475
Другие технологии на стороне Web-сервера.....	477
"Тонкие" клиенты	479
JavaScript	480
Ajax	481
Требования к средствам разработки информационных систем на основе Web-технологий	483
6.3. Вопросы для самопроверки.....	484
Глава 7. Новые технологии в области баз данных	485
7.1. Хранилища данных (OLAP)	485
Общие положения.....	485
OLAP и OLTP.....	485
Основные концепции.....	487
Потоки информации в OLAP	489
Об архитектуре и типах хранилищах данных	491
Проектирование хранилища данных (пример).....	492
Многомерные базы данных	495
Организация OLAP на базе MS SQL Server 2005	496
7.2. Объектные и объектно-реляционные СУБД.....	497
О стандарте объектных баз данных.....	498
Манифест.....	498
Стандарт	500
Примеры объектных СУБД	506
Объектно-реляционные СУБД	510
7.3. Распределенные информационные системы	513
Положения Дейта.....	513
Управление системным каталогом.....	515
Первичная копия	516
7.4. Вопросы для самопроверки.....	517
Литература	518
Предметный указатель	522

ГЛАВА 2



Базы данных как часть информационной системы

Глава посвящена рассмотрению такого понятия, как *база данных*. Мы подробно рассматриваем одну из наиболее употребляемых в настоящее время моделей данных — реляционную модель: основные положения, реляционную алгебру и реляционное исчисление, нормальные формы, связи между таблицами, а также место, которое занимает базы данных в структуре информационных систем.

2.1. Базы данных

Общие понятия

Базы данных — что это? Ведь ранее мы говорили об информационных хранилищах. Информационное хранилище является частью информационной системы. Но мы никак не уточняли, какова структура таких хранилищ. В действительности обычному пользователю нет нужды знать эту структуру. При работе с ИС он оперирует профессиональными терминами и понятиями. Например, бухгалтер при работе с бухгалтерской информационной системой имеет дело с такими объектами, как счет, проводка, журнал, платежный документ и т. д. Он не знает, как в действительности устроено информационное хранилище, да и не обязан знать, т. к. каждый должен заниматься своим делом, а поверхностные знания часто приводят к отрицательным результатам.

Таким образом, информационное хранилище является общим понятием, отражающим факт наличия в системе хранимых данных. Для пользователя те объекты, которыми он оперирует при работе с ИС, и есть те самые хранимые данные. С другой стороны, грамотный компьютерщик знает, что все данные хранятся в файлах. Файлы — это те структуры хранения данных, которыми манипулирует операционная система. Но файлы — это всего лишь именован-

ная последовательность хранимых во внешней памяти байтов. Однако при создании информационной системы не слишком удобно оперировать последовательностями байтов. Непосредственно с файлами работают системные программисты. Прикладные программисты работают с другой абстракцией. Эта абстракция находится на среднем уровне между уровнем файловой системы (*физический или внутренний уровень*) и уровнем пользователя (*внешний уровень*). Назовем этот промежуточный уровень *логическим* или *прикладным уровнем*. Сказанное проиллюстрировано на рис. 2.1.

Итак, для конечного пользователя информационное хранилище состоит из значений тех показателей, которыми он оперирует в силу своей профессиональной деятельности. Для системного программиста информационное хранилище представляет собой набор файлов. Для прикладного же программиста и администратора ИС информационное хранилище представляет собой некоторую абстрактную структуру, отражающую предметную область. Этот уровень и принято описывать с помощью понятия "*база данных*". Информационное хранилище может состоять из нескольких баз данных, идентифицируемых своим именем. Структура базы данных строится на основе *модели данных*, о которых мы будем говорить в следующем разделе. Важно отметить, что модель данных, по сути, представляет собой некоторый язык, с помощью которого может быть описана предметная область. Такое описание не должно зависеть от аппаратной и программной части системы. В описание должен входить и набор типовых операций над данными.



Рис. 2.1. Три уровня восприятия данных

ОПРЕДЕЛЕНИЕ

Базой данных будем называть именованную часть информационного хранилища, структура которой описывается на языке некоторой модели данных. Описание структуры конкретной базы данных называется *схемой*, *системным каталогом* (или просто *каталогом*) *базы данных* или *словарем базы данных*¹.

¹ Чрезвычайно интересное обсуждение понятия "база данных" можно найти в книге [34].

СУБД

СУБД — это система управления базами данных. В *главе 5* мы подробно остановимся на этом понятии. Но для дальнейшего рассмотрения требуется хотя бы кратко пояснить читателю, что это такое.

Для того чтобы понять, что такое СУБД, обратимся снова к рис. 2.1. Для того чтобы прикладные программисты воспринимали базу данных не в виде набора файлов, а как некоторую структуру, которая описывает предметную область, между ними и файловой системой должна быть некоторая прослойка. Это прослойка представляет собой набор процедур (программный интерфейс), с помощью которого прикладной программист может управлять базой данных. Эту функцию выполняет СУБД. Другими словами, СУБД отделяет прикладного программиста от физической структуры базы данных. В простейшем случае СУБД состоит только из программного интерфейса. Так обстоит дело в некоторых системах программирования. В более сложных системах мы имеем также и интегрированную среду, позволяющую в интерактивном режиме управлять базами данных. Кроме этого, СУБД также предоставляет разработчикам средства безопасности, такие, например, как утилиты для резервного копирования, поддержка параллельной работы приложений, система поддержки целостности баз данных, транзакционные механизмы, парольный вход и разделение доступа, и многое другое (*см. главу 5*). Большинство современных информационных систем строится именно на основе СУБД.

На рис. 2.2 представлена схема взаимодействия СУБД и прикладного программного обеспечения в схеме построения информационной системы. Тут важно понять, что база данных в определенной модели существует для прикладного программного обеспечения, тогда как программное обеспечение СУБД взаимодействует с данными на уровне файловой системы и системных вызовов операционной системы.

Все СУБД могут быть поделены на настольные и промышленные. Настольные СУБД, такие как Access, FoxPro, предназначены для создания либо автономных информационных систем, либо ИС файл-серверного типа. Промышленные СУБД¹, такие как Oracle, MS SQL Server, PostgreSQL и др., предназначены для построения клиент-серверных информационных систем. СУБД, как правило, предоставляет разработчику язык программирования, который включает в себя специализированный язык управления базами данных. Для наиболее распространенных баз данных *реляционного* типа таким языком является язык SQL.

¹ Другое название промышленных СУБД — серверы баз данных.

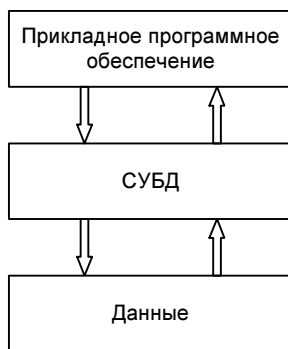


Рис. 2.2. Информационная система на основе СУБД

Использование СУБД при построении информационных систем призвано реализовать *физическую* и *логическую* независимость прикладного программирования от данных. Физическая независимость от данных заключается в том, что работа программного обеспечения ИС не будет зависеть от изменений, которые могут происходить на внутреннем, физическом уровне. Эти изменения могут заключаться, например, в том, что будет изменена файловая система или же в том, что изменится структура тех файлов, которые составляют базу данных.

Логическая независимость прикладного программирования от данных при использовании СУБД в трехуровневой структуре доступа к данным (см. рис. 2.1) заключается, прежде всего, в том, что добавление новых элементов (например, добавление нового столбца в таблицу) в структуру данных никак не влияет на функционирование программного обеспечения. Добиться логической независимости от операций удаления элементов из базы данных в общем случае невозможно, поскольку на удаляемые элементы, как правило, имеются явные ссылки в программном обеспечении.

Модели данных

Рассмотрим основные модели данных, которые используются (или использовались) при создании информационных систем. Перечень существующих моделей данных является в то же время и хронологическими метками истории развития баз данных и информационных систем.

Файловая модель

Иногда файловую модель, на мой взгляд, очень неудачно называют файловой системой. При этом упускается из вида тот факт, что данный термин служит для обозначения организации данных во внешней памяти, которую поддерживают и используют операционные системы. Файловая модель была первой

моделью, используемой при разработке информационных систем¹. Точнее, модель, как таковая, отсутствовала. Можно сказать, что файловая модель — это модель без СУБД. Прикладные программисты разрабатывали базы данных непосредственно на внутреннем уровне (см. рис. 2.1), т. е. имели дело напрямую с файлами (логический и внутренний уровень совпадали). Другими словами базы данных представляли собой наборы файлов, трактовка внутренней структуры которых принадлежала непосредственно разработчикам данной информационной системы, т. е. была уникальна. Файловая модель обладала рядом недостатков, но, несмотря на это, она дожила до наших дней и иногда используется для разработки небольших однопользовательских информационных систем. Перечислим основные недостатки данной модели.

- ❑ Структуру базы данных приходилось разрабатывать каждый раз при разработке информационной системы, что требовало определенных усилий и времени.
- ❑ Поскольку алгоритм управления такой базой данных был полностью заложен в программном обеспечении информационной системы, то при необходимости изменить структуру данных каждый раз приходилось вносить изменения и в программное обеспечение. Кроме этого, каждый раз приходилось писать утилиты для преобразования базы данных от одной структуры к другой. Такие утилиты часто были одноразового использования. Все усложняющиеся структуры данных и алгоритмы их обработки приводили в конечном итоге к увеличению программных ошибок.
- ❑ Часто даже в одной фирме создавалось несколько информационных систем (для каждого отдела, а зачастую для каждого рода деятельности), каждая из которых оперировала своими структурами данных. Бесконечной головной болью был перенос данных из одной ИС в другую, ведь структуры данных еще и периодически менялись. Так что часть программистов непрерывно писала программы преобразования данных из одного формата в другой.
- ❑ Децентрализованное хранение приводило к тому, что в различных отделах приходилось дублировать одни и те же данные, что, во-первых, требовало дополнительных ресурсов (времени и денег), а во-вторых, хранение дополнительных данных требовало и дополнительной внешней памяти. Но головная боль начиналась тогда, когда нарушалась согласованность данных об одном и том объекте, но из разных информационных систем. Для обнаружения и исправления таких ошибок требовались большие усилия.

¹ Можно сказать, что первые файловые модели были простой заменой ручных картотек. Каждый ящик картотеки соответствовал одному файлу. Например, в одном файле хранился список работников предприятия, второй файл хранил перечень выпускаемой продукции и т. д.

- ❑ Работа того или иного отдела требовала все новых и новых форм отчетности. Но для каждого нового отчета приходилось писать программный код, что увеличивало дополнительную нагрузку программистов. При этом, опять же после изменения структуры данных, все процедуры формирования отчетов приходилось снова переписывать.
- ❑ Разные фирмы часто пользовались различными языками программирования, отличающимися друг от друга структурой создаваемых ими файлов. Возникали, таким образом, дополнительные сложности совместимости форматов файлов.

ЗАМЕЧАНИЕ

Строго говоря, файловую модель данных нельзя назвать моделью в полном смысле этого слова, т. к. здесь мы четко видим зависимость описания данных от таких факторов, как файловая система и программное обеспечение.

Сетевая модель

Сетевая модель относится к ранним моделям данных. Сейчас информации об этой модели данных почти не встретишь, даже такой специалист как К. Дейт при переиздании своего классического труда "Введение в системы баз данных" исключил вопросы, касающиеся сетевой и иерархической модели данных (см. [8, 15]). Впрочем, в последнее время снова заговорили о сетевой модели в связи с распределенными информационными системами, а также с развитием объектных моделей данных.

В 1971 году группа DTBG (Database Task Group) представила в Американский национальный институт стандартов отчет, который послужил в дальнейшем основой для разработки сетевых систем управления базами данных. Стандарт сетевой модели впервые был определен в 1975 году организацией CODASYL (Conference of Data System Languages), 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 записей **Зарплата сотрудника**, выражающая множество сотрудников некоторого отдела. Тип записей представляет (моделирует) некоторый класс реального мира;
- *набор* — именованная двухуровневая иерархическая структура, которая содержит запись владельца и запись (или записи) членов. Наборы отражают связи "один-ко-многим" и "один-к-одному" между двумя типами записей. На рис. 2.3 представлен пример набора. 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"). Обратите внимание, как на рис. 2.3 обозначается связь "один-ко-многим". Разветвление на конце указывает на множество экземпляров некоторого объекта. Такой способ обозначения связей мы будем использовать и далее. Среди всех наборов в сетевой модели допускается существование наборов, не имеющих владельцев. Такие наборы называются *сингулярными*. Владельцами сингулярных наборов формально считается система. Сингулярные наборы предназначены для доступа к экземплярам отдельных записей.

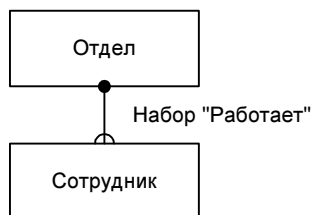


Рис. 2.3. Набор в сетевой модели данных

Резюмируя вышесказанное, будем говорить, что структура базы данных в сетевой модели задается типами записей и типами наборов.

Отметим некоторые особенности построения сетевой модели:

- база данных может состоять из произвольного количества записей и наборов различных типов;
- связь между двумя записями может выражаться произвольным количеством наборов;

- ☐ в любом наборе может быть только один владелец;
- ☐ тип записи может быть владельцем в одних типах наборов и членом в других типах наборов;
- ☐ тип записи может не входить ни в один из определенных типов наборов.

Для управления сетевой базой данных используется специальный язык, который можно разбить на следующие разделы:

- ☐ язык описания данных в сетевой модели:
 - описание базы данных (размещение);
 - описание элементов, агрегатов и записей;
 - описание наборов;
- ☐ язык манипулирования данными:
 - навигационные операции. С помощью операций навигации (группы операций FIND), двигаясь по связям, можно переходить от одной текущей записи к другой. Соответственно операции модификации осуществляются над текущей записью;
 - операции модификации, которые осуществляют:
 - добавление новых экземпляров отдельных типов записей;
 - добавление экземпляров новых наборов;
 - удаление экземпляров записей и наборов;
 - модификацию отдельных составляющих внутри конкретных экземпляров записей.

Иерархическая модель

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*¹ и *поле*. Поле данных определяется как минимальная, неделимая единица данных, доступная пользователю с помощью

¹ Вместо термина "сегмент" используется также термин "запись".

СУБД. Выделяют также *тип поля*, представляющий собой совокупность полей одного типа. Сегмент состоит из конкретных экземпляров полей. *Тип сегмента* — 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 потомков 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.

На рис. 2.4 схематически представлена иерархическая структура. Узлы (сегменты) соединены друг с другом связующими дугами. Сегмент А является корневым сегментом. Сегменты В, Е, Н, J, I являются листовыми сегментами. Каждый сегмент при этом может содержать произвольное количество полей.

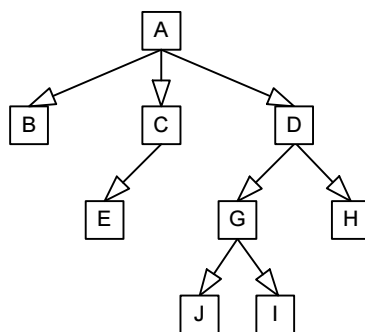


Рис. 2.4. Иерархическая структура

Для иерархической модели данных выделяют два языковых средства: язык описания данных и язык модификации данных. Описание базы данных предполагает описание всех ее сегментов и установление связей между ними.

Иерархическая модель довольно удобна для представления предметных областей, т. к. иерархические отношения довольно часто встречаются между сущностями реального мира. Но иерархическая модель не поддерживает отношения "многие-ко-многим", когда множество объектов одного типа связаны с множеством объектов другого типа. Предположим, что требуется построить модель отношения между множеством собственников жилья и множеством квартир. Если основной вопрос будет заключаться в определении того, каким жильем владеет тот или иной собственник, то естественно взять в качестве родительских узлов данные о собственнике. При этом каждый сегмент — собственник будет связан с n узлами — квартирами. Таким образом,

по собственнику мы легко найдем все квартиры, которые находятся в его собственности. Однако проблема заключается в том, что у одной и той же квартиры может быть несколько собственников. То есть одна и та же квартира может встречаться в разных деревьях. В результате решения таких задач, как получение списка всех квартир или получение всех собственников конкретной квартиры, будут уже не столь очевидными. Кроме того, сложной выглядит даже операция удаления из базы конкретной квартиры, поскольку для этого придется просматривать все деревья. Можно, конечно, построить параллельно деревья, в которых родительскими сегментами будут данные о квартирах, а порождаемыми сегментами — данные о владельцах, но в результате мы получим еще избыточность данных, что породит дополнительную проблему их согласованности.

Основной единицей обработки в иерархической модели является сегмент. К сегментам могут применяться такие операции, как запомнить, модифицировать, удалить, извлечь, найти. Операция поиска сводится к одной из возможных процедур обхода дерева. Иерархические СУБД поддерживают, обычно, правило: никакой сегмент не может существовать без своего родителя (исключая корневой сегмент). Подобные правила, поддерживаемые СУБД, называют *ограничениями целостности*.

В качестве примера реально работающей иерархической модели данных можно привести устройство реестра, поддерживаемое операционными системами семейства Windows.

Реляционная модель

Основателем реляционной модели данных является сотрудник фирмы IBM Э. Ф. Кодд. В статье "A Relation Model of Data for Large Shared Data Banks", которая вышла в 1970 году, он показал, что любое представление данных может быть сведено к совокупности двумерных таблиц, которые в математике называются отношениями (relations, отсюда термин "*реляционный*"). Мы подробно рассмотрим реляционную модель данных в *разд. 2.2*.

Объектная и объектно-реляционная модели

Объектные СУБД призваны интегрировать свойства баз данных и объектных языков программирования. В 1993 году был принят стандарт объектных баз данных ODMG-93 (Object Database Management Group — группа управления объектными базами данных, образована в 1991 году). В последнее время ведущие производители реляционных СУБД ради повышения конкурентоспособности своих продуктов пытаются внести в них элементы объектного проектирования. Мы подробнее поговорим об объектных и объектно-реляционных моделях в *главе 5*.