С. А. Нестеров

БАЗЫ ДАННЫХ

УЧЕБНИК И ПРАКТИКУМ
ДЛЯ АКАДЕМИЧЕСКОГО БАКАЛАВРИАТА

Рекомендовано Учебно-методическим отделом высшего образования в качестве учебника для студентов высших учебных заведений, обучающихся по инженерно-техническим направлениям и специальностям

Книга доступна в электронной библиотечной системе biblio-online.ru

Москва ▪ Юрайт ▪ 2016

УДК 004.65(075.8)
ББК 32.973-018.2я73
Н56

***Автор*:**

Нестеров Сергей Александрович — кандидат технических наук, доцент кафедры системного анализа и управления института компьютерных наук и технологий Санкт-Петербургского политехнического университета Петра Великого.

***Рецензенты*:**

Грачев В. А. — генеральный директор ООО «Регул+», резидент инновационного центра «Сколково»;

Шведенко В. Н. — доктор технических наук, профессор, заместитель генерального директора ООО «Регул+» по науке;

Фирсов А. Н. — доктор технических наук, профессор кафедры системного анализа и управления Санкт-Петербургского политехнического университета Петра Великого.

Нестеров, С. А.

Н56 Базы данных : учебник и практикум для академического бакалавриата / С. А. Нестеров. — М. : Издательство Юрайт, 2016. — 230 с. — Серия : Бакалавр. Академический курс.

ISBN 978-5-9916-6427-1

В учебнике системно излагаются основы теории баз данных, рассматриваются вопросы, связанные с их проектированием и разработкой в среде современных систем управления базами данных. Основное внимание уделяется вопросам создания и использования реляционных баз данных, являющихся на сегодняшний день наиболее распространенными. Приводится описание языка SQL и сравнение стандарта SQL и его диалекта, используемого в популярной системе Microsoft SQL Server.

Теоретические сведения сопровождаются примерами и лабораторными работами, выполняемыми в Microsoft Access и Microsoft SQL Server, а также в среде CA ERwin Data Modeler.

Содержание учебника соответствует актуальным требованиям Федерального государственного образовательного стандарта высшего образования.

Учебник предназначен для студентов высших учебных заведений, обучающихся по направлениям подготовки бакалавров «Системный анализ и управление», «Информационные системы и технологии». Может быть полезен для студентов, обучающихся по другим специальностям, при изучении дисциплин «Базы данных» и «Управление данными». Также может использоваться в системах повышения квалификации, в учреждениях дополнительного профессионального образования.

УДК 004.65(075.8)
ББК 32.973-018.2я73

ISBN 978-5-9916-6427-1

Оглавление

Список принятых сокращений

БД — база данных
ДКНФ — доменно-ключевая нормальная форма
ИС — информационная система
НФ — нормальная форма
НФБК — нормальная форма Бойса — Кодда
ОС — операционная система
ПО — программное обеспечение
СУБД — система управления базами данных
ФЗ — функциональная зависимость
ЭВМ — электронно-вычислительная машина
CASE (от *англ.* computer-aided software engineering) — набор инструментов и методов программной инженерии для проектирования программного обеспечения
ER-диаграмма — диаграмма «сущность — связь» (от *англ.* entity-relation)
RI (от *англ.* referential integrity) — ссылочная целостность
SQL (от *англ.* structured query language) — структурированный язык запросов

Предисловие

Широкое использование технологий баз данных в различных областях деятельности привело к тому, что современный специалист по информационным технологиям просто обязан иметь познания в этой области, даже если его непосредственные обязанности напрямую не связаны с базами данных.

В представляемом учебнике изложен материал учебного курса по дисциплине «Базы данных», читаемого автором в Санкт-Петербургском политехническом университете Петра Великого. В ходе его изучения студенты получают знания о моделях данных, проектировании реляционных баз данных, языке SQL. Материал разбит на 11 глав.

В главе 1 рассмотрены основные понятия теории баз данных, сделан краткий обзор этапов развития систем управления базами данных. В главе 2 рассмотрены архитектура ANSI/SPARC и основные архитектуры многопользовательских систем баз данных. Глава 3 посвящена моделям данных: перечислены основные модели данных, подробно рассмотрены иерархическая и сетевая модели. В главе 4 представлена реляционная модель данных. Описаны основные структуры и ограничения, разобраны операции реляционной алгебры. Глава 5 посвящена вопросам нормализации реляционных баз данных. Описаны задачи нормализации, определения нормальных форм, приведены примеры нормализации. В главе 6 разбираются вопросы инфологического проектирования баз данных; построение ER-диаграмм в различных нотациях, в том числе IDEF1X. Приведены примеры использования современного CASE-средства ERwin Data Modeler. Тема главы 7 — основные конструкции языка SQL, являющегося стандартным языком запросов при работе с реляционными базами данных. В главе 8 рассмотрены понятие транзакции, основные свойства транзакций, проблемы, возникающие при параллельном выполнении транзакций, вопросы обеспечения отказоустойчивости баз данных. Глава 9 посвящена вопросам организации физического хранения данных в реляционных СУБД и использованию индексов для увеличения скорости поиска данных. В главе 10 рассмотрено создание программируемых объектов баз данных. На примере СУБД Microsoft SQL Server показаны различные программируемые объекты (хранимые процедуры, функции, триггеры), описаны правила их создания, приведены примеры использования. Глава 11 посвящена использованию данных в формате XML совместно с реляционными данными. Изучены вопросы преобразования результатов запросов к реляционным данным в формат XML, а также хранения и обработки данных в формате XML в таблицах реляционных СУБД.

В приложении приведены описания лабораторных работ по темам, рассматриваемым в учебнике. Для некоторых из представленных работ потребуются дополнительные файлы, которые можно найти на странице книги в электронно-библиотечной системе издательства «Юрайт» www.biblio-online.ru.

В результате изучения материалов курса «Базы данных» студент должен:

знать

- базовые понятия теории баз данных;
- основные модели данных;
- нормальные формы реляционных отношений;
- язык структурированных запросов SQL;
- особенности создания и использования программируемых объектов баз данных;
- способы совместного использования реляционных данных и данных в формате XML;

уметь

- проектировать БД реляционного типа;
- проводить нормализацию БД;
- писать запросы на языке SQL;
- средствами современных СУБД провести преобразования реляционных данных в формат XML и обратно, писать запросы к XML-данным;

владеть

- навыками проектирования БД с использованием современных CASE-средств;
- навыками разработки и администрирования БД в среде современной СУБД.

Учебник рекомендуется для студентов высших учебных заведений, обучающихся по направлениям подготовки бакалавров «Системный анализ и управление», «Информационные системы и технологии» при изучении дисциплин «Базы данных» и «Управление данными». Также может использоваться в учреждениях дополнительного профессионального образования.

Глава 1
ВВЕДЕНИЕ В ТЕОРИЮ БАЗ ДАННЫХ

1.1. Основные понятия

Исторически сложились два основных направления использования вычислительной техники, первое из которых связано с проведением сложных преобразований над относительно небольшими объемами данных с простой структурой. Здесь компьютеры позволили быстрее проводить расчеты по вычислительно сложным алгоритмам. Подобные задачи дали толчок к созданию первых ЭВМ, их актуальность не снижается и сейчас.

Другое направление связано с созданием информационных систем. В них необходимо не только обрабатывать, но и хранить большие объемы данных со сложной внутренней структурой, обеспечивать быстрый поиск нужной информации. Создание подобных систем стало возможным после появления надежных, емких и быстродействующих устройств энергонезависимой памяти: в первую очередь речь идет о накопителях на жестких магнитных дисках. Классическим примером систем подобного типа являются системы резервирования железнодорожных и авиационных билетов. Последовательность операций, выполняемых при каждом заказе, относительно проста, но для корректного функционирования всей системы необходимо хранить и постоянно актуализировать большие объемы данных, выполнять в них поиск и т.п.

Автоматизированная информационная система — это функционирующий на основе ЭВМ комплекс, обеспечивающий сбор, хранение, актуализацию и обработку информации в целях поддержки какого-либо вида деятельности, т.е. автоматизированная ИС разрабатывается для определенной предметной области.

Предметная область — часть реального мира, подлежащая изучению с целью организации управления и, в конечном счете, автоматизации. Создавая ИС, мы, в некотором смысле, формируем информационную модель, позволяющую описать значимые характеристики реальных объектов и их взаимосвязи.

Автоматизированная ИС может функционировать самостоятельно или служить компонентом более сложной системы, например автоматизированной системы управления предприятием. По типу хранимой и обрабатываемой информации выделяют два больших класса автоматизированных информационных систем — документальные и фактографические.

Документальные системы служат для работы с текстами на естественном языке — статьями, научными отчетами, текстами законодательных

актов и т.д. Наиболее распространенным видом документальных систем являются информационно-поисковые системы, предназначенные для накопления и поиска документов на естественном языке. Их иногда еще называют полнотекстовыми базами данных.

Документы, хранящиеся в подобных системах, составляют поисковый массив документов системы. Для каждого документа формируется поисковый образ — некое формальное описание документа в терминах языка системы, которое отражает его содержание. Например, поисковый образ может быть сформирован указанием набора ключевых слов. Запрос пользователя представляется в виде поискового образа запроса, который сопоставляется с поисковыми образами хранимых документов. Отобранные в результате документы называются *релевантными запросу*.

Фактографические системы составляют другой большой класс автоматизированных информационных систем. Они оперируют фактическими данными, представленными в виде специальным образом организованных совокупностей записей. Именно им и посвящена основная часть данного курса, так как именно в фактографических системах в полной мере используются методы и инструменты теории БД. Фактографические системы, создаваемые средствами технологии БД, часто называют банками данных (см. определение ниже).

Иногда в дополнение к выделенным двум классам вводят понятие *лексикографических* баз данных и информационных систем, относя к ним различного рода словари и классификаторы.

В отечественных нормативных документах в сфере разработки БД даются следующие определения.

Банк данных [1, 2] — это система специальным образом организованных данных (баз данных), программных, технических, языковых, организационно-методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.

В англоязычной литературе понятие, по сути близкое к банку данных, обозначается термином *система баз данных* (*англ.* database system).

База данных — именованная совокупность данных, отражающая состояние объектов и их отношений в заданной предметной области.

Базу данных можно рассматривать как электронную картотеку, хранилище для некоторого набора занесенных в компьютер данных. Выполняют следующие операции над БД:

- добавить новые данные в БД;
- изменить существующие данные;
- удалить данные из БД;
- найти данные в БД и т.д.

Базы данных организуются на основе различных моделей данных. Пример фрагмента БД реляционного типа представлен в табл. 1.1. Данные в этом случае организуются в виде реляционных таблиц, строки таблиц называют записями, а столбцы — полями или атрибутами. Принципиально важной особенностью БД является то, что они содержат дополнитель-

ную служебную информацию о своей структуре, иначе говоря, являются самодокументируемыми.

Таблица 1.1

Фрагмент реляционной БД

StudID	FIO	Group
123	Иванов И.И.	382
124	Петров П.П.	382

1.2. Компоненты системы баз данных

Рассмотрим упрощенную схему системы БД (рис. 1.1). Она включает следующие основные компоненты [3]: данные, аппаратное обеспечение, программное обеспечение, пользователи.

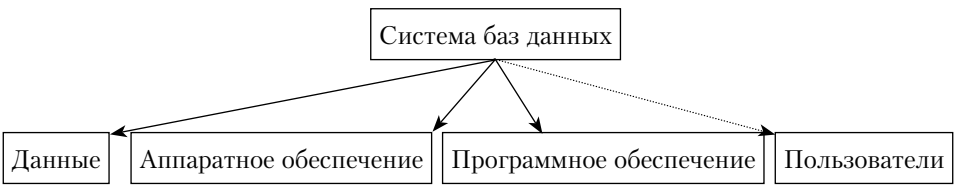

Рис. 1.1. **Обобщенная схема системы БД**

Данные. Базы данных состоят из некоторого набора *постоянных данных*. Выделяют также *транзитные данные*, такие как промежуточные результаты, входные и выходные данные. *Входные данные* — информация, передаваемая системе (например, вводимая с клавиатуры). Такая информация может стать причиной изменений постоянных данных (она может стать частью постоянных данных), но не является частью БД как таковой. *Выходные данные* — сообщения и результаты, выдаваемые системой. Они, как правило, берутся из постоянных данных, но их нельзя рассматривать как часть БД.

Например, пусть в систему каждые 10 мин поступают данные о температуре воздуха, в базе сохраняется среднее значение за час, а запрос выводит среднесуточную температуру. В этом случае хранимые значения могут отличаться и от входных, и от выходных.

Кроме данных, описывающих предметную область, в БД обычно содержатся данные, описывающие элементы и структуры самой базы. Подобные описания относятся к разряду *метаинформации*, т.е. «информации об информации». Централизованное хранилище метаинформации называется *словарем данных*, или *репозиторием*. Именно наличие репозитория позволяет говорить о свойстве самодокументированности БД. В современных СУБД реляционного типа такое хранилище реализуется в виде системного каталога — набора служебных таблиц, куда заносится информация о структуре объектов (БД, таблиц, представлений и т.д.), пользователях, разрешениях и т.п.

По виду отношения «пользователь — данные» можно выделить два типа систем баз данных:

1) *однопользовательская система* (*англ.* single-user system) — система, в которой в одно и то же время к БД может получить доступ только один пользователь;

2) *многопользовательская система* (*англ.* multi-user system) — система, в которой к БД могут получить доступ одновременно несколько пользователей. При этом для конечного пользователя необходимо обеспечить такие условия, чтобы результат его работы не зависел от того, работает он с данными в однопользовательском режиме или совместно с другими.

Данные в БД должны быть интегрированными и общими.

Когда говорят про *интегрированные данные*, подразумевают, что к данным, собранным из разных источников, предоставляется единый способ доступа. Например, система позволяет получить данные с кафедр университета об успеваемости студентов, из библиотеки — об использовании студентами литературы, и совместно использовать их для решения какой-то задачи.

Общие данные подразумевают возможность использования отдельных наборов данных из общей БД разными группами пользователей для решения своих специфических задач. Например, менеджер интернет-магазина может работать с данными о конкретном заказе, а руководитель — с итоговыми данными, характеризующими деятельность магазина за определенный период.

Эти два свойства представляют собой наиболее важное преимущество использования систем БД корпоративного уровня, а «интеграция» является преимуществом при использовании настольных (персональных) систем БД.

Аппаратное обеспечение. В наиболее общем виде можно выделить две группы устройств, принципиально важных для систем БД: 1) устройства хранения данных; 2) устройства обработки данных. Для небольших систем и обработка, и хранение могут производиться на одном и том же компьютере. Крупная система БД может использовать различные типы систем хранения и множество серверов для обработки данных. Здесь возникает целый класс новых задач, связанных с разработкой и эксплуатацией распределенных систем.

Программное обеспечение. Между физической БД и пользователями системы располагается уровень ПО, основной компонент которого — *система управления базами данных* (*англ.* database management system).

Система управления базами данных [2, 4] — совокупность языковых и программных средств, предназначенная для создания, ведения и совместного использования БД многими пользователями. Основная функция СУБД — предоставление пользователю БД возможности работать с ней, не вникая в детали на уровне аппаратного обеспечения.

Кроме СУБД система БД, как правило, включает еще ряд программных компонент — утилиты, генераторы отчетов, пользовательское прикладное ПО и т.д.

Пользователи. Пользователей системы БД можно разделить на три класса [3].

Прикладные программисты отвечают за написание прикладных программ, использующих БД. Разрабатываемые ими программы обращаются с запросами к СУБД и получают результаты запросов. Выделяют программы пакетной обработки и оперативные приложения, функция которых — поддержка работы конечного пользователя, имеющего интерактивный доступ к системе.

Конечные пользователи работают с системой БД непосредственно с рабочей станции или терминала. Они могут воспользоваться разработанным для них прикладным ПО или встроенными средствами СУБД (графическими или с интерфейсом командной строки). Нужно понимать, что система БД создается для поддержания деятельности конечных пользователей.

Администраторы данных и *администраторы баз данных*. Администратор данных — человек, который несет ответственность за данные предприятия. Он принимает решения, какие данные необходимо вносить в БД, кому и к каким данным можно иметь доступ, и т.д. Иногда таких специалистов называют аналитиками. Администратор базы данных — технический специалист, который отвечает за реализацию решений администратора данных. На этапе разработки системы он занимается созданием БД, на этапе эксплуатации — настройкой, обслуживаем, резервным копированием и другими подобными задачами.

1.3. Этапы развития систем управления базами данных и ведущие производители

До появления СУБД вопросы хранения данных разработчики каждой программы решали самостоятельно, используя при этом функции ОС или даже напрямую обращаясь к устройствам ввода-вывода. Однако ОС предоставляет функции по работе с файлами, а вопросы организации хранения записей внутри файла, поиска данных, проверки ограничений для записи средствами ОС не решить. Кроме того, при одновременном доступе нескольких пользователей к одним и тем же данным необходимы дополнительные механизмы, позволяющие централизованно управлять этим процессом. Эти и ряд других причин привели к созданию отдельного класса программного обеспечения — СУБД.

Первый этап развития СУБД связан с «большими» ЭВМ (мейнфреймами). Первая коммерческая СУБД называлась IMS (от *англ.* Information Management System, система управления информацией) и была выпущена корпорацией IBM в 1968 г. для платформы IBM System/360. Этот этап характеризуется централизованным хранением данных. СУБД должны были обеспечивать коллективный доступ к БД, а сами они работали на «больших» машинах под управлением сложных и достаточно развитых ОС.

На первом этапе исследователями были получены существенные результаты в области теории БД. В частности, это создание иерархической, сетевой и реляционной моделей данных. Реляционную модель предложил работавший в IBM математик Э. Ф. Кодд (Edgar Frank Codd, 1923—2003;

в 1981 г. получил премию Тьюринга). В 1970 г. он опубликовал статью «A Relational Model of Data for Large Shared Data Banks», в которой описал основные идеи реляционного подхода. В дальнейшей работе над моделью принял участие и К. Дейт (Christopher J. Date), автор классического учебника «Введение в системы баз данных» [3]. Реляционные СУБД на сегодняшний день являются наиболее распространенными.

Следующий этап развития СУБД связан с появлением персональных компьютеров. Их широкое распространение, ограниченные вычислительные возможности и в среднем более низкий (по сравнению с большими ЭВМ) уровень подготовки пользователей привели к возникновению целого класса настольных СУБД. Изначально это были, в основном, однопользовательские системы, с достаточно ограниченными возможностями, но простым пользовательским интерфейсом и невысокими требованиями к аппаратуре. Многие из них не выдержали конкуренции и сейчас не поддерживаются. Оставшиеся в процессе развития стали приобретать черты многопользовательских СУБД, такие как механизмы совместного использования и защиты данных. В качестве примера популярных сейчас настольных СУБД можно назвать Microsoft Access и OpenOffice Base.

Параллельно существенные изменения происходили и с СУБД корпоративного уровня. Они были связаны с распространением компьютерных сетей, в результате чего доминирующей стала клиент-серверная технология, в том числе с поддержкой распределенной обработки данных.

Большое влияние на СУБД оказало и развитие сети Интернет. При динамическом формировании веб-страниц в большинстве случаев задействуются СУБД и обслуживаемые ими БД. Это привело к появлению ряда СУБД, чья популярность, в первую очередь, связана с их использованием при создании веб-приложений. Наиболее яркий пример — реляционная СУБД MySQL.

В то же время выяснилось, что реляционные СУБД и используемый для работы с ними язык запросов SQL подходят далеко не для всех задач. Появилась и активно развивается идеология NoSQL (*англ.* Not only SQL, не только SQL), объединяющая ряд подходов и проектов, связанных с созданием нереляционных БД.

Наиболее именитый производитель серверных СУБД — это корпорация Oracle, выпустившая в 1979 г. первую коммерческую реляционную СУБД Oracle v2 и с тех пор являющаяся ключевым производителем в области серверов БД.

Существенное место на рынке занимает корпорация IBM, выпускающая реляционную СУБД DB2 и иерархическую СУБД IMS. Приобретя в 2001 г. подразделение корпорации Informix, IBM добавила в свою линейку продуктов одноименную СУБД.

Заметное место занимает корпорация Microsoft с ее серверным продуктом MS SQL Server и настольной СУБД Access, входящей в пакет Microsoft Office. Несмотря на то, что MS SQL Server выпускается только для ОС семейства Windows, популярность данной платформы, поддержка в средствах разработки Microsoft и широкие возможности самой СУБД привели к ее широкому распространению.

Основанная в 1984 г. компания Sybase может быть также названа одним из пионеров в области разработки реляционных СУБД. В конце 1980-х — начале 1990-х гг. Sybase вела разработку SQL Server в альянсе с Microsoft, но в дальнейшем продукты стали независимыми. На сегодняшний день в линейке продуктов Sybase есть реляционный сервер БД Adaptive Server Enterprise, встраиваемая реляционная СУБД SQL Anywhere и нереляционная СУБД с «поколоночным» хранением данных Sybase IQ, предназначенная для задач аналитической обработки данных и построения хранилищ данных. В 2010 г. Sybase была приобретена компанией SAP AG, ведущим поставщиком программных решений для управления бизнесом.

Среди приверженцев свободно распространяемого ПО широкую популярность приобрела СУБД MySQL, изначально разрабатывавшаяся созданной в Швеции компанией MySQL AB. В настоящее время у MySQL лидирующие позиции в качестве СУБД, используемой в области веб-разработки. В 2008 г. компания MySQL AB была приобретена Sun Microsystems, а в 2010 г. уже саму Sun приобрела Oracle. Сейчас выпускаются как коммерческие, так и бесплатно распространяемая версия MySQL (MySQL Community Edition). Кроме того, существуют разрабатываемые сообществом свободно распространяемые ответвления MySQL, например MariaDB.

Также необходимо отметить, что у многих коммерческих разработчиков есть бесплатно распространяемые версии СУБД, такие как Oracle Database Express Edition, IBM DB2 Express-C, Microsoft SQL Server Express Edition.

Если говорить о СУБД, основанных на объектной модели данных, то наиболее известным на сегодняшний день проектом в этой области является система Caché, разрабатываемая компанией InterSystems. Особенность данной СУБД заключается в том, что она реализует объектное представление данных, сохраняя в то же время возможность доступа к данным средствами языка SQL как к реляционной БД.

1.4. Преимущества и недостатки систем баз данных

Преимущества централизованного подхода в управлении данными, по сравнению с ситуацией, когда каждая программа независимо реализует хранение своих данных, заключаются в следующем.

1. *Сокращение избыточности*. При отсутствии централизованной БД каждое приложение будет отдельно хранить свои данные. Нередко одни и те же данные используются несколькими приложениями. Например, и список сотрудников в отделе кадров, и список сотрудников, записанных в библиотеку предприятия, содержат имя, адрес, паспортные данные. В централизованной БД такие данные можно объединить с полным или частичным устранением избыточности.

2. *Возможность устранения противоречивости*. Зачастую противоречия являются следствием избыточности. Если одинаковые данные об одном человеке представлены в двух различных записях и это «раздвоение» не учтено, то рано или поздно две записи могут перестать согласовываться:

например в одну запись внесут изменения, а в другую — нет. В этом случае БД станет противоречивой. Противоречий можно избежать, устраняя избыточность или контролируя ее. В последнем случае может использоваться множественное обновление, когда при внесении изменений в одну из записей автоматически будут изменены и записи, связанные с ней.

3. *Возможность общего доступа к данным*. При наличии централизованной БД сотрудники разных подразделений в соответствии с их полномочиями могут совместно использовать эти данные.

4. *Возможность соблюдения стандартов*. Внедрение единых стандартов по обработке данных намного проще осуществить в централизованной системе.

5. *Возможность введения ограничений для обеспечения безопасности*. При централизованном хранении и обработке данных проще разработать и внедрить правила разграничения доступа к ним.

6. *Возможность обеспечения целостности данных*. Задача обеспечения целостности заключается в обеспечении правильности и точности данных в БД. Возникновение противоречий — это пример нарушения целостности. Другой пример: при централизованном хранении проще организовать процедуры резервного копирования и восстановления БД.

В то же время, неудачное проектирование и внедрение системы БД, может повлечь ряд проблем. Основная из них связана с тем, что вся работа информационной системы предприятия будет зависеть от системы БД, и сбои в ней могут привести к катастрофическим последствиям.

Контрольные вопросы и задания

1. Опишите основные классы автоматизированных информационных систем.
2. Дайте определения терминов «банк данных» и «база данных».
3. В чем отличия БД от обычных файлов с данными?
4. Что такое репозиторий?
5. Охарактеризуйте основные классы пользователей систем БД.
6. Поясните понятия «общие данные» и «интегрированные данные».
7. Какие программные компоненты может включать система баз данных?
8. Опишите этапы развития СУБД. Перечислите ведущие мировые компании — разработчики СУБД и их основные продукты.
9. В чем заключаются преимущества централизованного подхода к управлению данными?

Глава 2
ВВЕДЕНИЕ В АРХИТЕКТУРУ СИСТЕМ БАЗ ДАННЫХ

Под *архитектурой системы БД* понимается совокупность ее функциональных компонентов, а также средств обеспечения их взаимодействия друг с другом, с пользователями и с системным персоналом.

Представленный далее материал обобщенно описывает архитектуру системы БД. Это вовсе не означает, что данное описание подходит для каждой системы БД. Например, «малые» системы, возможно, не будут поддерживать все аспекты архитектуры.

Ниже в упрощенном виде рассматривается архитектура, предложенная подкомитетом SPARC (*англ.* Standards Planning and Requirements Committee, комитет по планированию стандартов) американского национального института стандартов ANSI, так называемая архитектура ANSI/SPARC, впервые представленная в 1975 г.

2.1. Трехуровневая архитектура систем баз данных ANSI/SPARC

Архитектура систем БД ANSI/SPARC описывает логическую организацию системы с точки зрения представления данных пользователям и включает три уровня: внутренний, концептуальный и внешний. Каждый из них состоит из одного или нескольких представлений (рис. 2.1). Уровни определяются следующим образом:

- *внутренний уровень* — уровень, наиболее близкий к физическому хранению;
- *внешний уровень* наиболее близок к пользователям, он связан со способами представления данных для отдельных пользователей;
- *концептуальный уровень* — промежуточный уровень между двумя первыми, на котором представлены все имеющиеся данные, но в более абстрактном по сравнению с внутренним уровнем виде.

Прежде чем более подробно рассматривать архитектуру БД, введем несколько определений [3]. *Хранимое поле* — наименьшая единица хранимых данных. БД содержит экземпляры каждого из нескольких типов хранимых полей. *Хранимая запись* — набор связанных хранимых полей. *Хранимый файл* — набор всех экземпляров хранимых записей одного типа.

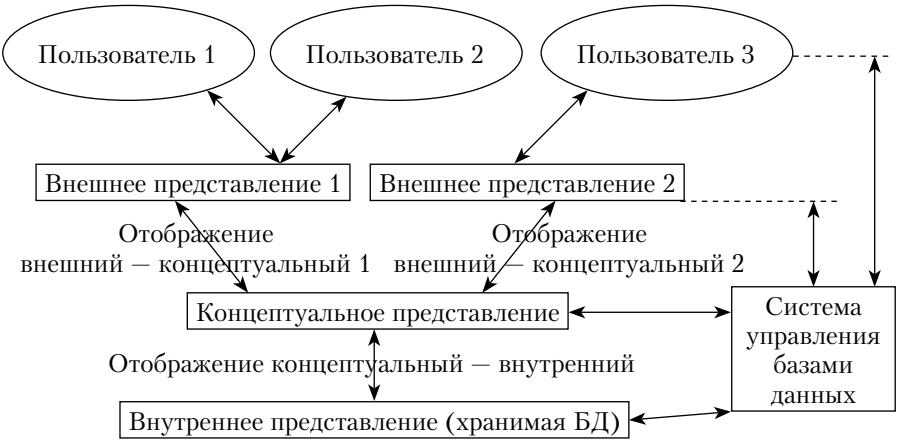

Рис. 2.1. **Схематичное представление архитектуры ANSI/SPARC**

Внешний уровень. Индивидуальный уровень пользователя называется *внешним уровнем*. Пользователи могут относиться к различным группам: прикладные программисты, конечные пользователи, администраторы (они работают также с внутренним и концептуальным уровнем). У каждого пользователя есть свой язык общения с СУБД. У конечного пользователя это может быть язык запросов или специальный язык, основанный на формах и меню. Для прикладных программистов им может быть один из языков высокого уровня. Все эти языки включают подъязык данных, т.е. подмножество операторов базового языка, связанное только с объектами и операциями БД. Наиболее распространенный подобный язык — SQL, он поддерживается большинством систем реляционного типа.

Любой язык данных является комбинацией, по крайней мере, двух подчиненных языков — *языка определения данных* (*англ.* data definition language, DDL), который поддерживает определение и объявление объектов БД, и *языка обработки данных* (*англ.* data manipulation language, DML), который поддерживает операции с объектами БД, их обработку.

Вернемся к рассмотрению трехуровневой архитектуры систем БД. Отдельного пользователя интересует только некоторая часть всей БД. Кроме того, пользовательское представление этих данных может существенно отличаться от того, как они хранятся. В соответствии с терминологией ANSI/SPARC представление отдельного пользователя называется внешним представлением. *Внешнее представление* — это содержимое БД, каким его видят определенный конечный пользователь или группа пользователей. Можно сказать, что для пользователя его внешнее представление и есть БД.

Внешних представлений обычно бывает несколько. Например, пользователь из отдела кадров может рассматривать БД как набор записей об отделах и служащих. Он может ничего не знать про записи о деталях и поставщиках, с которыми работают пользователи в отделе обеспечения.

В общем случае, внешнее представление состоит из множества экземпляров *внешних записей*, которые могут не совпадать с хранимыми запи-

сями. В системах, отличных от БД, логическая (внешняя) запись обычно совпадает с хранимой.

Концептуальный уровень состоит из одного представления. *Концептуальное представление* — это представление всей информации БД в несколько более абстрактной форме (как и в случае внешнего представления) по сравнению с физическим способом хранения данных.

Концептуальное представление состоит из множества экземпляров каждого типа концептуальной записи. Концептуальная запись не обязательно должна совпадать с внешней записью, с одной стороны, и с хранимой записью — с другой.

Концептуальное представление — это представление всего содержимого БД, а концептуальная схема — это определение такого представления. Определения в концептуальной схеме могут включать определения многих дополнительных средств, таких как средства безопасности или правила для обеспечения целостности. Администратор данных, определяя, какие характеристики предметной области требуется сохранять в системе, фактически определяет основу концептуальной схемы.

Внутренний уровень. Так же как и концептуальный, внутренний уровень состоит только из одного представления. *Внутреннее представление* описывает все подробности, связанные с хранением данных в базе. Оно состоит из экземпляров каждого типа внутренней записи. Термин «внутренняя запись» принадлежит терминологии ANSI/SPARC и фактически соответствует хранимой записи. Внутреннее представление, так же как внешнее и концептуальное, не связано с аппаратным уровнем и не включает подробностей, связанных с размещением данных на дисках, таких как номера секторов и т.п.

Внутреннее представление описывается с помощью внутренней схемы, которая определяет типы хранимых записей, индексы (служебные структуры, упрощающие поиск данных), способы представления хранимых полей, физическую последовательность хранимых записей и т.д.

Отображения. В представленной архитектуре присутствует отображение двух уровней. Отображение концептуального уровня на внутренний определяет соответствие между концептуальным представлением и хранимой БД. При изменении структур хранения изменяется и отображение «концептуальный — внутренний» таким образом, чтобы концептуальная схема осталась неизменной. Это обеспечивает так называемую *физическую независимость данных*.

Отображение внешнего уровня на концептуальный определяет соответствие между внешними представлениями и концептуальным. Например, несколько концептуальных полей «индекс», «города», «улица», «дом» для пользователя могут быть объединены в одно внешнее поле «адрес». Появляется возможность менять отдельные внешние представления, дополнительно изменяя только отображения и не затрагивая остальные уровни системы. Отделение внешнего уровня от концептуального обеспечивает *логическую независимость данных* [5].

Определения представлений каждого из уровней и отображений СУБД должна хранить вместе с прочей метаинформацией и использовать их при обработке запросов.

Архитектура ANSI/SPARC имеет большое теоретическое значение, определяя пути обеспечения логической и физической независимости данных. Однако два уровня отображения приводят к дополнительным накладным расходам при обработке запросов пользователя, поэтому разработчики СУБД, стараясь увеличить быстродействие систем, обычно отходят от строгой реализации этой архитектуры.

2.2. Архитектура многопользовательских систем баз данных

Рассмотрим аспекты архитектуры систем БД, связанные с обеспечением совместной работы пользователей. В несколько абстрактной форме можно представить многопользовательскую систему БД совокупностью двух компонент — создающего запросы клиента, и сервера, который выполняет приходящие запросы. В частном случае, и клиентское, и серверное приложения могут выполняться на одном и том же компьютере. Но они могут находиться и на разных компьютерах, связанных телекоммуникационной сетью. В зависимости от разделения функций между клиентом и сервером выделяются несколько типов архитектур систем БД.

Файл-серверная архитектура. Некоторые СУБД могут работать только в режиме «файл-сервер». Это означает, что СУБД находится на клиентской рабочей станции и даже может быть скомпонована с прикладным ПО. Для осуществления совместного доступа к данным файлы БД в этом случае размещаются на файловом сервере (рис. 2.2).

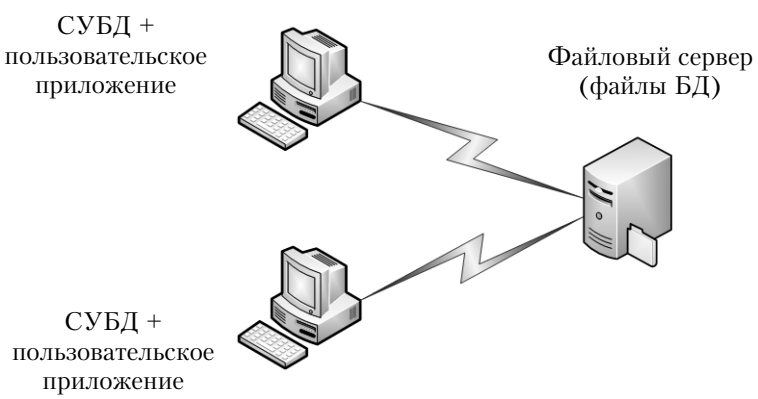

Рис. 2.2. **Файл-серверная архитектура**

Если работа с данными ведется на нескольких компьютерах, то на каждом из них будет запущен экземпляр СУБД. Для обработки задания пользователя с сервера запрашиваются файлы с данными, а их обработка производится локально. В результате в таких системах приходится передавать по сети много данных для того, чтобы на стороне клиента среди них найти те, которые удовлетворяют запросу.

Основным достоинством подобной архитектуры является простота: для того чтобы из локального приложения сделать многопользовательское сетевое, надо просто переместить файлы БД на файловый сервер. При этом

СУБД должна обеспечить минимально необходимый набор функциональности в области совместного использования файлов БД: блокировку изменяемых записей и т.п.

Недостатков у такой архитектуры существенно больше. Во-первых, это неэффективная загрузка сети передачи данных: по сети передается во много раз больше данных, чем это необходимо приложению. Например, если в таблице 100 000 записей, из которых нам нужно 10, в неудачном случае может потребоваться передача всей таблицы на клиентский компьютер для последующей обработки. Помимо загрузки сети это существенно увеличит время выполнения операции по сравнению с обработкой базы, хранящейся локально.

Во-вторых, могут возникать существенные проблемы при синхронизации совместной работы пользователей. Для поддержания информации об изменяемых в данный момент записях файл-серверные СУБД могут создавать на сервере так называемые файлы блокировок или другие подобные структуры. Если один из клиентов заблокировал часть записей (СУБД внесла информацию в файл блокировок, что эти записи будут изменяться и их нельзя использовать другим клиентам), после чего потерял связь с сервером, экземпляры СУБД на других клиентах не смогут использовать соответствующие фрагменты БД, пока эту проблему не решит администратор, откорректировав файл блокировок.

В-третьих, файл-серверная архитектура предъявляет дополнительные требования к производительности клиентских рабочих мест: вся обработка данных производится на клиенте.

Примером многопользовательской системы с файл-серверной архитектурой является совместное использование БД Microsoft Access в локальной сети. В таком режиме с одной базой могут нормально работать несколько пользователей: в зависимости от размеров базы, характеристик сети и интенсивности работы с данными, это может быть до 5—10 одновременных сеансов.

Двухзвенная архитектура «клиент-сервер». Данная архитектура предполагает, что СУБД находится на сервере и только она имеет доступ к файлам БД (рис. 2.3).

На клиентских компьютерах работают пользовательские приложения и клиентские компоненты СУБД, осуществляющие взаимодействие с сервером. От клиента на сервер приходят запросы, которые обрабатываются СУБД, и результат отправляется на клиентский компьютер. По сравнению с файл-серверной архитектурой, в этом случае минимизируется сетевой трафик: по сети передаются только затребованные данные.

Централизованная работа с файлами БД позволяет более эффективно решать вопросы, связанные с совместным доступом к данным и с обеспечением безопасности. В связи с тем, что СУБД работает на сервере, снижаются и требования к оборудованию на стороне клиента. Прикладная программа может быть написана на любом языке, который поддерживает взаимодействие с используемой СУБД через имеющиеся клиентские компоненты: могут использоваться такие технологии, как ODBC, ADO или ADO.Net, JDBC и др.