内容提要

本书系统全面地阐述了数据库系统的基础理论、基本技术和基本方法，具体内容包括数据库的基本概念、数据模型、关系数据库理论、关系数据库、数据完整性、数据库设计、结构化查询语言(SQL)、索引、触发器、函数、存储过程、事务管理、并发控制、数据库安全、MySQL 数据库在 Web 开发中的应用等。本书注重理论与实践结合，内容丰富全面，非常具有实用性。

本书适合作为高等教育数据库相关课程的教材，也可供相关人员参考。

图书在版编目(CIP)数据

数据库原理与 MySQL 应用 / 魏再超主编. — 上海：上海交通大学出版社，2024.3

ISBN 978-7-313-30297-7

Ⅰ. ①数… Ⅱ. ①魏… Ⅲ. ①SQL 语言—数据库管理系统 Ⅳ. ①TP311.132.3

中国国家版本馆 CIP 数据核字(2024)第 051043 号

数据库原理与 MySQL 应用
SHUJUKU YUANLI YU MySQL YINGYONG

主　　编：魏再超
出版发行：上海交通大学出版社　　地　　址：上海市番禺路 951 号
邮政编码：200030　　电　　话：021-64071208
印　　制：三河市骏杰印刷有限公司　　经　　销：全国新华书店
开　　本：787 mm×1 092 mm　1/16　　印　　张：13.5
字　　数：299 千字
版　　次：2024 年 3 月第 1 版　　印　　次：2024 年 3 月第 1 次印刷
书　　号：ISBN 978-7-313-30297-7
定　　价：42.00 元

前言

随着互联网技术的不断发展，海量信息不断涌现，数据库技术应运而生。如今的数据库技术已经渗透各个领域，各行各业利用计算机处理数据已十分普遍，掌握数据收集、存储、统计与分析等技能成为数据时代人们应该具备的一种基本能力。

本书系统、全面地阐述了数据库系统的基础理论、基本技术和基本方法，内容主要包括数据库的基本概念、数据模型、关系数据库理论、关系数据库、数据完整性、数据库设计、结构化查询语言(SQL)、索引、触发器、函数、存储过程、事务管理、并发控制、数据库安全、MySQL 数据库在 Web 开发中的应用等，注重理论与实践结合，内容丰富全面，非常具有实用性。

本书具有以下特点。

1. 数据库理论与 MySQL 具体实践相结合

本书介绍了数据库的通用理论，为学生在今后的学习生活中学习使用各种数据库打下基础；同时以广泛使用的数据库系统软件 MySQL 为例，介绍数据库理论在 MySQL 下的实践操作，使学生能掌握通用的 SQL 查询语言及 MySQL 这种特定数据库系统的应用，为学习计算机专业各后续课程做好必要的知识和技能准备。

2. 强化 SQL 及 MySQL 编程

本书全面介绍了结构化查询语言(SQL)、函数、存储过程、事务管理、并发控制、数据库安全，学生很容易将所学知识迁移到其他数据库系统上。

3. 强调实际应用

本书以学生熟悉的、能接触到的数据作为案例，理论与实践相结合，引导学生进行数据库的设计、实现、管理及应用开发。实现知识积累、技能实践、能力增强，符合学生的认知规律和接受能力。

本书由保山学院魏再超任主编，由虞翔、杨薇、杨朝凤任副主编。本书在编写过程中参考了国内外有关专家和学者在数据库方面的一些新的理念和成果，在此一并表示感谢。

由于编者水平有限，书中不足之处在所难免，恳请广大读者批评指正。

编　者

目录

第 1 章　数据库系统概述

数据库是指以一定的方式存储在一起、能为多个用户共享、具有尽可能小的冗余度，并与应用程序彼此独立的数据集合。目前使用最为广泛的是关系型数据库，它是建立在关系模型基础上的数据库，借助于集合代数等数学概念和方式来处理数据库中的数据。现在常用的数据库有 Oracle、Microsoft SQL Server、Access 和 MySQL 等，它们大多是关系型数据库。

目前，数据库是数据管理的最新技术，是计算机科学的重要分支。对于一个国家来说，数据库的建设规模、数据库信息量的大小和使用频度已成为衡量这个国家信息化程度的重要标志。因而数据库知识变得越来越重要，而且它无处不在。例如，今日头条等网络新闻的存储、QQ 好友信息同步、求职信息发布、选课信息呈现、电子商务平台的个性化信息推荐等。

数据库已经成为现代信息系统不可或缺的重要组成部分。具有数百万甚至数十亿字节信息存储量的教据库已经普遍存在于金融、教育、工业、农业、服务业和政府等诸多行业部门的信息系统中，数据库技术是计算机领域中发展较快的技术之一。

1.1　数据库系统的概念

数据库系统是指在计算机系统中引入数据库后的系统，一般由数据库、数据库管理系统（database management system，DBMS）（及其开发工具）、数据库应用系统（database application system，DBAS）、用户[其中包括数据库管理员（database administrator，DBA）等]构成。

数据库的作用是帮助用户更好地管理数据，它是相互联系的表和其他结构的集合。DBMS 是用于创建、处理和管理数据库的计算机系统软件。

DBAS 包括为特定应用环境建立的数据库、开发的各类应用程序及编写的文档资料，它们是一个有机整体。DBAS 涉及各个方面，如高考志愿填报系统、学生选课系统等。通过运行 DBAS 可以实现对数据库中数据的添加、删除、修改、查询等操作。

用户是指使用数据库应用程序记录信息，并使用程序界面读取、录入和查询数据的人。

1.1.1 数据库

1. 信息与数据

为了了解世界，与世界交流，人们需要描述各种事物。用自然语言描述虽然很直接，但过于复杂，不利于用计算机表达。为此，人们常常只抽取那些感兴趣的事物特征或属性来描述事物，将描述事物的符号记录为数据。数据是数据库中存储的基本对象，而信息是从数据中获得的有意义的内容。数据在大多数人头脑中的直观反映就是数字。实际上数字只是数据的一种简单的形式，是对数据的一种传统和狭义的理解。从广义去理解，数据的种类很多，文本(text)、图形(graph)、图像(image)、音频(audio)、视频(video)、学生的档案记录、商品的销售情况等都是数据。通常将描述事物的符号记录称为数据。数据有多种形式，它们都可以经过数字化后保存在计算机中。在日常生活中，人们可以直接用自然语言(如汉语)来描述事物。例如，可以这样描述某高校计算机系一名同学的基本信息：李明，女，1998 年 6 月出生，云南昆明人，2018 年入学。在计算机中描述如下：

(李明，女，1998-06，云南昆明人，计算机系，2018)

即把学生的姓名、性别、出生年月、出生地、所在院系、入学时间组织在一起，组成一条记录，描述了李明同学的信息。因此，将从数据中获得的有意义的内容称为信息。这里的学生记录就是描述学生的数据，这样的数据是有结构的。记录是计算机中表示和存储数据的一种格式或方法。

数据和信息之间是相互联系的。数据是反映客观事物属性的记录，是信息的具体表现形式。数据经过加工处理后，就成为信息；而信息需要经过数字化转变成数据才能存储和传输。

2. 数据库概念

数据库可以形象地理解为存放数据的仓库，只不过这个仓库是放在计算机存储设备上且数据是按一定格式存放的。在科学技术飞速发展的今天，数据量急剧增加，海量数据涌现，过去人们把数据存放在文件柜中，现在人们借助计算机和数据库技术科学地保存和管理大量复杂的数据，以便能方便而充分地利用这些宝贵的信息资源。

严格地讲，数据库是长期存储在计算机内、有组织的、可共享的大量数据的集合。数据库中的数据按一定的数据模型组织、描述和存储，具有较小的冗余度、较高的数据独立性和易扩展性，并可为各种用户共享。

3. 数据处理与数据管理

数据处理也称信息处理，就是将数据转换为信息的过程。数据处理的内容主要包括数据的收集、整理、存储、加工、分类、维护、排序、检索和传输等一系列活动。数据处理的目的是从大量的数据中，根据数据自身的规律及其相互关系，通过分析、归纳、推理等科学的方

法，利用计算机技术、数据库技术等手段，提取有效的信息资源，为进一步分析、管理和决策提供依据。

所谓数据管理，是指对各种数据进行分类、组织、编码、存储、检索和维护。发展到现在，数据管理技术经历了3个阶段，分别为人工管理阶段、文件系统阶段和数据库系统阶段。

1)人工管理阶段

20世纪50年代中期以前，计算机还没有磁盘和专门管理数据的软件，计算机只应用于科学技术方面，数据则由计算和处理它的程序自行携带，该时期被称为人工管理阶段。

人工管理阶段的特点如下。

(1)数据不能长期保存。

(2)由程序本身管理数据。

(3)数据不能共享。

(4)数据不具有独立性。

2)文件系统阶段

随着科学技术的发展，在20世纪50年代后期到20世纪60年代中期，计算机不仅应用于科学技术，而且开始应用于管理。该时期的计算机硬件出现了磁盘，计算机软件出现了高级语言和操作系统，因此，程序和数据有了一定的独立性，出现了程序文件和数据文件，这就是所谓的文件系统阶段。

文件系统阶段的特点如下。

(1)数据可以长期保存。

(2)数据由文件系统来管理。

(3)数据冗余度大，共享性差。

(4)数据独立性差。

3)数据库系统阶段

随着网络技术的发展，以及计算机软、硬件功能的进步，在20世纪60年代后期，计算机可以管理规模巨大的数据，这时如果计算机仍使用文件系统来管理数据，则远远不能满足当时各种应用需求，于是出现了数据库技术，特别是关系型数据库技术，该阶段就是所谓的数据库系统阶段。

数据库系统阶段的特点如下。

(1)数据实现结构化。

(2)数据实现了共享性。

(3)数据独立性强。

(4)数据粒度变小。

1.1.2 DBMS

1. DBMS 的定义

DBMS 安装在操作系统之上，是一个管理、控制数据库中各种数据库对象的系统软件，如图 1-1 所示。

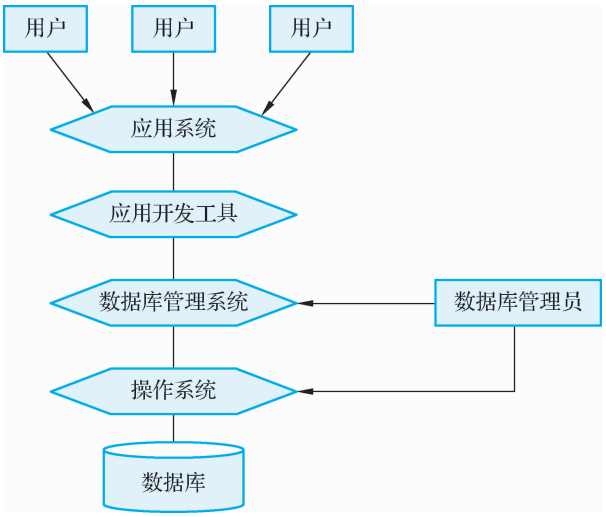

图 1-1 DBMS 图

数据库用户无法直接通过操作系统获取数据库文件中的具体内容；DBMS 通过调用操作系统的服务，如进程管理、内存管理、设备管理及文件管理等，为数据库用户提供管理、控制数据库中各种数据库对象、数据库文件的接口，实现对数据的管理和维护。

DBMS 通常会选择某种数学模型存储、组织、管理数据库中的数据，常用的数学模型包括“层次模型”“网状模型”“关系模型”及“面向对象模型”等。基于“关系模型”的 DBMS 称为关系型数据库管理系统(relational database management system，RDBMS)。随着关系型数据库管理系统的日臻完善，目前关系型数据库管理系统已占据主导地位。

通过关系型数据库管理系统，数据库开发人员可以轻而易举地创建关系型数据库容器，并在该数据库容器中创建各种数据库对象(表、索引、视图、存储过程、触发器、函数等)，以及维护各种数据库对象。

DBMS 的目标是让用户能够更方便、更有效、更可靠地建立数据库和使用数据库中的信息资源。DBMS 不是应用软件，它不能直接用于诸如工资管理、人事管理、资料管理等事务管理工作，但 DBMS 能够为事务管理提供技术和方法、应用系统的设计平台和设计工具，使相关的事务管理软件较方便、容易地实现。

2. DBMS 的主要功能

DBMS 是位于用户与操作系统之间的一层数据管理软件，和操作系统一样，它是计算机

的基础软件，也是一个大型的软件系统。

其主要功能包括如下几个方面。

1)数据库的建立和维护

数据库初始数据的输入、转换功能，数据库的转储、恢复功能，数据库的重组织功能和性能监听、分析功能等，通常是由一些实用程序或管理工具完成的。

2)数据定义功能

DBMS 提供数据定义语言(data definition language，DDL)，用户通过它可以方便地对数据库中的数据对象进行定义。

3)数据组织、存储和管理

DBMS 要分类组织、存储和管理各种信息，包括数据字典、用户数据、数据的存取路径等，要确定以何种文件结构和存取方式在存储器上组织这些数据，如何实现数据之间的联系。数据组织和存储的基本目标是提高存储空间利用率和方便存取，提供多种存取方法(如索引查找、Hash 查找、顺序查找等)来提高效率。

4)数据操作功能

DBMS 还提供了数据操作语言(data manipulation language，DML)，用户可以使用 DML 操作数据，实现对数据库的基本操作，如查询、插入、删除和修改等。

5)数据库事务管理和运行管理

数据库的建立、运用和维护是由 DBMS 统一管理、统一控制的，以保证数据的安全性、完整性，多用户对数据的并发使用及发生故障后的系统恢复。

6)其他功能

其他功能包括 DBMS 与网络中其他软件系统的通信功能，一个 DBMS 和另一个 DBMS 或文件系统的数据的转换动能，异构数据之间的互访和互操作功能等。

3. 常见的关系型数据库管理系统

目前，商品化的 DBMS 以关系型数据库为主导产品，技术比较成熟，主要有 Oracle 公司的 Oracle 和 MySQL，IBM 公司的 DB2，Sybase 公司的 Sybase，Microsoft 公司的 SQL Server、Access 和 Visual FoxPro 等。

1)Oracle

Oracle 是美国 Oracle 公司开发的一种适用于大型、中型和微型计算机的关系型数据库管理系统，提供以分布式数据库为核心的一组软件产品，是目前最流行的客户/服务器(client/server，C/S)或浏览器/服务器(browser/server，B/S)体系结构的数据库之一。Oracle 数据库由 3 种类型的文件组成，即数据库文件、日志文件和控制文件。Oracle 自动建立并更新一组数据字典，用来记录用户名、数据库元素及用户权限等信息。数据库管理员(DBA)可以通过数据字典来监视 Oracle 的状态，并帮助用户完成其应用。Oracle 本身也是根据数据字典来管理和控制整个数据库的。

2）DB2

DB2 是 IBM 公司推出的一种关系型数据库管理系统，可以在不同的操作系统平台上服务。DB2 主要应用于大型应用系统，具有较好的可伸缩性，可以支持从大型机到单用户环境，应用于 OS/2、Windows 等平台下。DB2 提供了高层次的数据利用性、完整性、安全性和可恢复性，以及小规模到大规模应用程序的执行能力，具有与平台无关的基本功能和 SQL 命令。DB2 还采用了数据分级技术，能够使大型数据很方便地下载到 LAN 数据库服务器，使 C/S 用户和基于 LAN 的应用程序可以访问大型数据，并使数据库本地化及远程连接透明化。

3）Sybase

Sybase 是美国 Sybase 公司推出的 C/S 模式的关系型数据库管理系统，也是世界上第一个真正的基于 C/S 架构的关系型数据库管理系统。Sybase 数据库将用户分为 4 种不同的类型，即系统管理员、DBA、数据库对象管理员和其他一般用户。系统管理员可以访问所有数据库和数据库对象。

4）MySQL

MySQL 是目前最流行的关系型数据库管理系统之一，由瑞典 MySQL AB 公司开发，目前属于 Oracle 公司。

在 Web 应用方面，MySQL 是最好的关系型数据库管理系统应用软件之一。MySQL 所使用的 SQL 是用于访问数据库的最常用的标准化语言。MySQL 软件分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。由于其社区版的性能卓越，搭配 PHP 和 Apache 可组成良好的开发环境。

5）SQL Server

SQL Server 是美国微软公司开发的一个关系型数据库管理系统，采用 C/S 体系结构，以 T-SQL 作为其数据库查询和编程语言。SQL Server 采用二级安全验证、登录验证以及数据库用户许可验证等安全模式。SQL Server 支持两种身份验证模式：Windows NT 身份验证和 SQL Server 身份验证，权限分配非常灵活。SQL Server 可以在不同的 Windows 操作平台上运行，并支持多种不同类型的网络协议，如 TCP/IP、IPX/SPX 等。近年来，SQL Server 不断更新版本，最新版本为 SQL Server 2022。

6）Access

1992 年，Microsoft 公司首次发布 Access。Access 是 Microsoft 公司推出的基于 Windows 的桌面关系型数据库管理系统，是 Office 系列应用软件之一。它提供了表、查询、窗体、报表、页、宏、模块 7 种用来建立数据库系统的对象；提供了多种向导、生成器、模板，把数据存储、数据查询、界面设计、报表生成等操作规范化，为建立功能完善的 DBMS 提供了方便，也使得普通用户不必编写代码就可以完成大部分数据管理功能。由于 Access 只是一

种桌面数据库，它适合数据量少（记录数不多和数据库文件不大）的应用。

7）Visual FoxPro

Visual FoxPro 简称 VFP，是 Microsoft 公司推出的数据库开发软件，用它来开发数据库，既简单又方便。Visual FoxPro 源于美国 Fox Software 公司推出的数据库产品 FoxBase，在 DOS 上运行，与 xBase 系列兼容。FoxPro 原来是 FoxBase 的加强版，最高版本为 2.6。之后，Fox Software 被 Microsoft 公司收购，加以发展，使其可以在 Windows 系列操作系统上运行，并更名为 Visual FoxPro，目前最新版本为 Visual FoxPro 9.0。

Visual FoxPro、Access 和 SQL Server 都是 Microsoft 公司的产品，只能在 Microsoft 公司的 Windows 系列操作系统上运行。而 Oracle、DB2、MySQL 等数据库是可以跨平台的，它们不仅可以在 Windows 系列操作系统上运行，还可以在其他操作系统（如 UNIX、Linux 和 MacOS）上运行。

1.1.3 DBA

DBA 是负责管理和维护数据库服务器的人，其负责全面管理和控制数据库系统。

在数据库系统环境下有两类共享资源：一类是数据库，另一类是数据库管理系统软件，因此，需要有专门的管理机构来监督和管理数据库系统。DBA 则是这个机构的人员，负责全面管理和控制数据库系统，具体职责如下。

1. 决定数据库中的信息内容和结构

数据库中要存放哪些信息，DBA 要参与决策。因此，DBA 必须参加数据库设计的全过程，并与用户、应用程序员、系统分析员密切合作、共同协商，搞好数据库设计。

2. 决定数据库的存储结构和存取策略

DBA 要综合各用户的应用要求，和数据库设计人员共同决定数据的存储结构和存取策略，获得较高的存取效率和存储空间利用率。

3. 定义数据的安全性要求和完整性约束条件

DBA 的重要职责是保证数据库的安全性和完整性。因此，DBA 负责确定各个用户对数据库的存取权限、数据的保密级别和完整性约束条件。

4. 监控数据库的使用和运行

DBA 还有一个重要职责就是监控数据库系统的运行情况，及时处理运行过程中出现的问题。例如，系统发生各种故障时，数据库会因此遭到不同程度的破坏，DBA 必须在最短时间内将数据库恢复到正确状态，并尽可能不影响或少影响计算机系统其他部分的正常运行，为此，DBA 要定义和实施适当的后备和恢复策略，如周期性的转储数据、维护日志文件等。

5. 数据库的改进和重组、重构

DBA 还负责在系统运行期间监视系统的空间利用率、处理效率等性能指标，对运行情

况进行记录、统计分析，依靠工作实践，并根据实际应用环境不断改进数据库设计。另外，在数据库运行过程中，大量数据被不断插入、删除、修改，时间一长，会影响系统的性能，因此，DBA 要定期对数据库进行重组，以提高系统的性能。当用户的需求增加和改变时，DBA 还要对数据库进行较大的改造，包括修改部分设计，即数据库的重构。

1.1.4 DBAS

DBAS 是在 DBMS 支持下建立的计算机应用系统。例如，以数据库为基础的财务管理系统、人事管理系统、图书管理系统等。无论是面向内部业务和管理的管理信息系统，还是面向外部提供信息服务的开放式信息系统，从实现技术角度而言，都是以数据库为基础和核心的计算机应用系统。

DBMS 与 DBAS 的区别如下。

(1)前者是提供数据库管理功能的计算机系统软件，后者是实现某种具体信息管理功能的计算机应用软件。

(2)前者为后者提供了数据库的定义、存储和查询方法，后者通过前者管理数据库。

(3)前者及其数据库安装在服务器端，它们之间通过数据访问技术进行数据通信。后者安装在客户端，由专门的开发系统或语言来设计。

1.1.5 用户

用户是指最终用户(end user)。最终用户通过应用系统的用户接口使用数据库。常用的接口方式有浏览器、菜单驱动、表格操作、图形显示、报表等，给用户提供简明、直观的数据表示。

最终用户可以分为如下 3 类。

(1)偶然用户。偶然用户不经常访问数据库，但每次访问数据库时往往需要不同的数据库信息，这类用户一般是企业或组织机构的高、中级管理人员。

(2)简单用户。数据库的大多数最终用户都是简单用户，其主要工作是查询和修改数据库，一般都是通过程序员精心设计并具有友好界面的应用程序来存取数据。例如，高考在线报名的考生、旅馆前台的服务员等都属于这类用户。

(3)复杂用户。复杂用户包括工程师、科学家、经济学家、科学技术工作者等具有较高学历背景的人员。这类用户一般都比较熟悉 DBMS 的各种功能，能够直接使用数据库语言访问数据库，甚至能够基于 DBMS 的应用程序接口(application programming interface，API)编制自己的应用程序。

1.2 数据库系统的特点

通过前面的学习可知，数据处理的中心问题是数据管理。随着计算机硬件和软件的发展，数据管理经历了人工管理、文件系统和数据库系统3个发展阶段。数据库技术正是应数据管理任务的需要而产生和发展的。与人工管理和文件系统相比，数据库系统的特点主要有以下几个方面。

1. 数据结构化

数据库在描述数据时不仅要描述数据本身，还要描述数据之间的联系。在文件系统中，尽管记录内部已经有了某些结构，但记录之间没有联系。数据库系统实现了整体数据的结构化，这是数据库的主要特征之一，也是数据库系统与文件系统的本质区别。在数据库系统中，数据不再针对某一个应用，而是面向全组织，具有整体的结构化。

2. 数据共享性高，冗余度低，易扩充

数据库系统从整体角度描述数据，数据不再面向某一个应用而是面向整个系统，因此，数据可以被多个用户、多个应用共享使用。数据共享可以大大减少数据冗余，节约存储空间。数据共享还能够避免数据之间的不相容性与不一致性。数据中的相容性指的是表示同一事实的两个数据应相同，否则就不相容；或者满足某一约束关系的一组数据不应该发生互斥，否则就不相容。例如，同一个人不能有两个性别。所谓数据的不一致性，是指数据的矛盾性、不相容性。产生数据不一致的原因主要有三种：一是数据冗余；二是并发控制不当；三是各种故障、错误。

第一种情况的出现往往是由于重复存放的数据未能进行一致性的更新。例如，教师工资的调整，如果人事处的工资数据已经改动了，而财务处的工资数据未改变，就会产生矛盾。

第二种情况是由于多用户共享数据库，而更新操作未能保持同步进行。例如，在飞机票订购系统中，如果不同的两个购票点同时查询某张机票的订购情况，而且分别为顾客订购了这张机票，就会造成一张机票分别卖给两名顾客的情况。这是由于系统没有进行并发控制，造成了数据的不一致性。

在第三种情况下，当由于某种原因（如硬件故障或软件故障）数据丢失或数据损坏时，可以利用各种数据库维护手段（如转存等）和数据恢复措施将数据库恢复到某个正确的、完整的、一致性的状态。

3. 数据独立性高

数据独立性包括数据的物理独立性和逻辑独立性。

(1)物理独立性是指用户的应用程序与存储在磁盘上的数据库中的数据是相互独立的。也就是说,数据在磁盘上的数据库中的存取是由 DBMS 管理的,用户的应用程序不需要了解,应用程序要处理的只是数据的逻辑结构,这样当数据的物理存储改变时,应用程序不用改变。

(2)逻辑独立性是指用户的应用程序与数据的逻辑结构是相互独立的,也就是说,数据的逻辑结构改变时,用户的应用程序可以不变。这样就将数据的定义从程序中分离出去,加上数据的存取又由 DBMS 负责,从而简化了应用程序的编制,大大减少了应用程序的维护和修改工作。

4. 数据由 DBMS 统一管理和控制

数据由 DBMS 统一管理和控制,用户和应用程序通过 DBMS 访问和使用数据库。数据库的共享是并发的共享,即多个用户可以同时存取数据库中的数据,甚至可以同时存取数据库中的同一数据。为此,DBMS 还必须提供以下几方面的数据控制功能。

1)数据的安全性保护

数据的安全性(security)是指保护数据以防止不合法的使用造成数据的泄密和破坏,使每个用户只能按规定对某些数据以某些方式进行使用和处理。

2)数据的完整性检查

数据的完整性(integrity)指数据的正确性、有效性和相容性。数据的完整性检查将数据控制在有效的范围内或保证数据之间满足一定的关系。

3)并发控制

当多个用户的并发(concurrency)进程同时存取、修改数据库时,可能会发生相互干扰而得到错误的结果或使数据库的完整性遭到破坏,因此,必须对多用户的并发操作进行控制和协调。

4)数据库恢复

计算机系统的硬件故障、软件故障、操作员的失误及人为破坏会影响数据库中数据的正确性,甚至造成数据库部分或全部数据丢失。DBMS 必须具有将数据库从错误状态恢复到某一个已知的正确状态(也称为完整状态或一致状态)的功能,这就是数据库的恢复(recovery)功能。

1.3 数据库系统结构

数据库系统结构可以分为用户级、概念级和物理级三个层次，反映观察数据库的三种角度。三个层次分别由用户、DBA 和系统程序员使用。每个层次的数据库都有自身对数据进行逻辑描述的模式，分别称为外模式（external schema）、逻辑模式和内模式（internal schema）。模式之间通过映射关系进行联系和转换。在数据库的三级模式结构中，数据库模式（概念模式）即全局逻辑结构是数据库的中心与关键，它独立于数据库的其他层次。

1.3.1 三级模式的内部结构

1. 三级模式结构

数据库系统的三级模式结构也称数据模式，是指数据库系统是由外模式、逻辑模式和内模式三级构成的，如图 1-2 和图 1-3 所示。

1）外模式

外模式也称子模式或用户模式，它是对数据用户（包括程序员和最终用户）能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图，是与某一个应用有关的数据的逻辑表示。

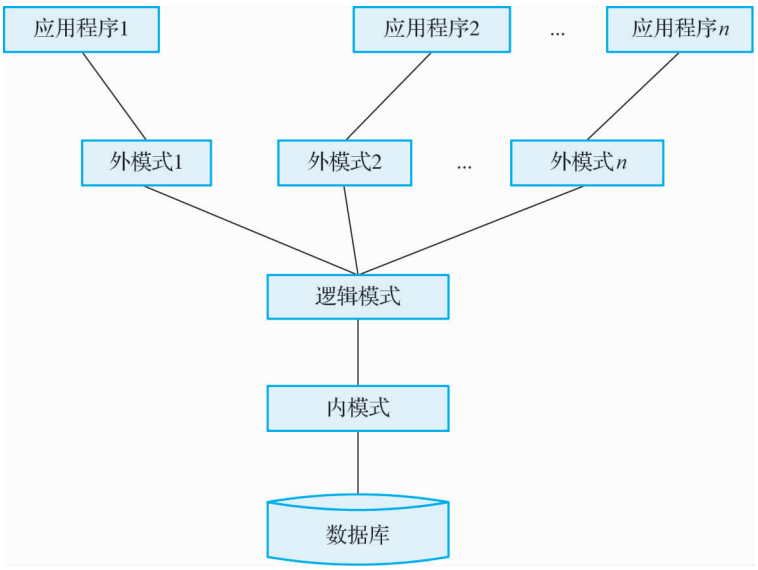

图 1-2 数据库系统的三级模式结构 1

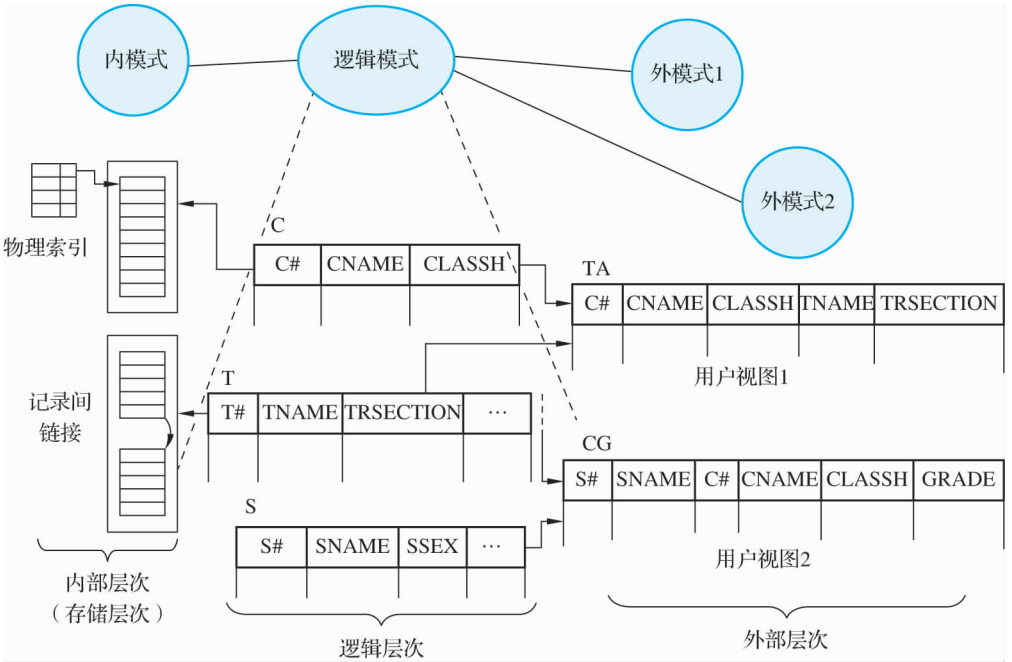

图 1-3　数据库系统的三级模式结构 2

2)逻辑模式

逻辑模式是由数据库设计者综合所有用户的数据，按照统一的观点构造的全局逻辑结构，是对数据库中全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图。它是数据库系统模式结构的中间层，既不涉及数据的物理存储细节和硬件环境，也与具体的应用程序无关。模式实际上是数据库在逻辑上的视图。一个数据库只有一个模式。它是用模式描述语言来描述的，是系统分析员及数据库管理员所看到的全局数据库视图。

3)内模式

内模式也称存储模式，一个数据库只有一个内模式。它是对数据物理结构和存储方式的描述，是对全体数据库数据的机器内部表示或存储结构的描述。它描述了数据在介质上的存储方式和物理结构，是数据库管理员创建和维护数据库的视图。例如，记录的存储方式是顺序存储、B 树结构存储还是 Hash 方式存储；索引按照什么方式组织；数据是否压缩存储，是否加密；数据的存储记录结构有何规定等。

内模式的设计目标是将系统的全局逻辑模式组织成最优的物理模式，以提高数据的存取效率，改善系统的性能指标。

2. 二级映像与数据独立性

数据库系统的三级模式是对数据的三个抽象级别，它把数据的具体组织留给 DBMS 管理，使用户能逻辑地、抽象地处理数据，而不必关心数据在计算机中的具体表示方式和存储方式。为了能够在内部实现这三个抽象层次的联系和转换，DBMS 在这三级模式之间提供

了两层映射:外模式/逻辑模式映射和内模式/逻辑模式映射。正是这两层映射保证了数据库系统中的数据具有较高的逻辑独立性和物理独立性。

1)外模式/逻辑模式映射

逻辑模式描述的是数据的全局逻辑结构,外模式描述的是数据的局部逻辑结构。对应于同一逻辑模式可以有任意多个外模式。对于每一个外模式,数据库系统都有一个外模式/逻辑模式映射,它定义了该外模式与逻辑模式之间的对应关系。这些映射定义通常包含在各自外模式的描述中。当逻辑模式改变时(如增加新的关系、属性,改变属性的数据类型等),由 DBA 对各个外模式/逻辑模式的映射做相应改变,可以使外模式保持不变。应用程序是依据数据库的外模式编写的,因此应用程序不必修改,保证了数据与程序的逻辑独立性,简称数据的逻辑独立性。

2)内模式/逻辑模式映射

数据库中只有一个逻辑模式,也只有一个内模式,所以,内模式/逻辑模式映射是唯一的,它定义了数据库全局逻辑结构与存储结构之间的对应关系。例如,说明逻辑记录和字段在内部是如何表示的,该映射定义通常包含在模式描述中。当数据库的存储结构改变(如选用另一种存储结构)时,由数据库管理员对内模式/逻辑模式映射做相应改变,可以使逻辑模式保持不变,从而应用程序也不必改变,保证了数据与程序的物理独立性,简称数据的物理独立性。

数据与程序之间的独立性使得数据的定义和描述可以从应用程序中分离出去。另外,由于数据的存取由 DBMS 管理,用户不必考虑存取路径等细节,从而简化了应用程序的编制,大大减少了应用程序维护和修改方面的工作。

1.3.2 B/S 与 C/S 应用结构

目前,数据库系统常见的运行与应用结构有 C/S 结构和 B/S 结构。

1. C/S 结构

C/S 结构是软件系统体系结构,通过它可以充分利用两端硬件环境的优势,将任务合理分配到客户端和服务器端来实现,降低了系统通信开销。目前,大多数应用软件系统都是 C/S 形式的两层结构。

C/S 结构的基本原则是将计算机应用任务分解成多个子任务,由多台计算机分工完成,即采用“功能分布”原则。客户端完成数据处理、数据表示及用户接口功能,服务器端完成 DBMS 的核心功能。这种客户端请求服务、服务器端提供服务的处理方式是一种新型的计算机应用模式。

2. B/S 结构

B/S 结构是 Web 兴起后的一种网络结构模式,Web 浏览器是客户端最主要的应用软件。这种模式统一了客户端,将系统功能实现的核心部分集中到服务器端,简化了系统的

开发、维护和使用。只要在客户端安装一个浏览器，在服务器端安装 Oracle、Sybase、SQL Server 等数据库，浏览器就可以通过 Web 服务器同数据库进行数据交互。

B/S 结构最大的优点就是可以在任何地方进行操作而不用安装任何专业软件，只要有一台能联网的计算机就能使用，客户端零安装、零维护，系统的扩展非常容易。

习题

1. 什么是 DBMS？它的主要功能有哪些？
2. 什么是数据库系统？它有什么特点？
3. 简述逻辑模式、外模式和内模式三者是如何保证数据独立性的。
4. 简述 C/S 结构与 B/S 结构的区别。
5. DBA 的职责有哪些？
6. 常用的 DBMS 有哪些？

第 2 章　信息与数据模型

数据库中的数据是结构化的，即建立数据库时要考虑如何组织数据，如何表示数据之间的关系，并合理地存放在计算机中，才能便于对数据进行有效的处理。数据模型就是描述数据之间联系的结构形式，它的主要任务就是组织数据库中的数据。

要为一个数据库建立数据模型，需要经过以下过程。

(1)要深入现实世界进行系统需求分析，认识客观世界。

(2)用概念模型真实、全面地描述客观世界中的管理对象及联系。

(3)利用一定的方法将概念模型转换为数据模型。

本章介绍了两个模型：数据模型和概念模型。其中，介绍了数据模型的概念、数据抽象的三个层次，还介绍了常见的概念模型(E-R 模型)及其向逻辑模型的转换原则与转换实例。

2.1　信息的三种世界及描述

信息的三种世界是指现实世界、信息世界和计算机世界(也称数据世界)。数据库是模拟世界中某些事物活动的信息集合，数据库中所存储的数据来源于现实世界的信息流。

信息流用来描述现实世界中一些事物某些方面的特征及事物间的相互联系。在处理信息流前需要先对其进行分析，并用一定的方法加以描述，然后将描述转换成计算机所能接受的数据形式。

2.1.1　现实世界

现实世界泛指存在于人脑之外的客观世界。信息的现实世界是指人们要管理的客观存在的各种事物、事物之间的相互联系及事物的发展和变化过程。通过对现实世界的了解和认识，人们对要管理的对象、管理的过程和方法有一个概念模型。认识信息的现实世界并用概念模型加以描述的过程称为系统分析。

1. 实体

现实世界中存在的可以相互区分的事物或概念称为实体(entity)。实体可以分为事物实体和概念实体。例如,一名学生、一台计算机、一位教师、一部汽车等是事物实体,选课、比赛等为概念实体。

2. 实体的特征

每个实体都有自己的特征(characteristic),利用实体的特征可以区分不同的实体。例如,学生通过姓名、性别、年龄、身高、体重等特征来描述自己。尽管实体具有许多特征,但是在研究时只选择其中对管理及处理有用的或有意义的特征。例如,对于人事管理,职工的特征可选择姓名、性别、年龄、工资、职务等;而在描述一名职工的健康状况时,可以用职工的身高、体重、血压等特征表示。

3. 实体集及实体集间的联系

具有相同特征或能用相同的特征描述的实体的集合称为实体集(entity set)。例如,学生、工人、课程、汽车等都是实体集。

实体集不是孤立存在的,它们之间有着各种各样的联系。例如,学生和课程之间有“选课”的联系。

2.1.2 信息世界

现实世界中的事物反映到人们的头脑中,经过认识、选择、命名、分类等综合分析而形成了印象和概念,从而得到了信息。当事物用信息来描述时,即进入了信息世界。

在信息世界中,实体的特征在头脑中形成的知识称为属性,实体通过其属性表示称为实例,同类实例的集合称为对象,对象即实体集中的实体用属性表示得出的信息集合。

实体与实例是不同的。例如,张三是一个实体,而“张三,男,25 岁,计算机系学生”是实例,现实世界中的张三除了姓名、性别、年龄和所在系外还有其他特征,而实例仅对需要的特征通过属性进行了描述。在信息世界中,实体集间的联系用对象间的联系表示。

信息世界通过概念模型(也称信息模型)、过程模型和状态模型反映现实世界,它要求对现实世界中的事物、事物间的联系和事物的变化情况进行准确、如实、全面的表示。

概念模型通过 E-R 图中的对象、属性和联系对现实世界的事物及关系给出静态描述。过程模型通过信息流程图和数据字典描述事物的处理方法和信息加工过程。状态模型通过事物状态转换图对事物进行动态描述。

三种模型的作用:数据库主要是根据概念模型设计的,而数据处理方法主要是根据过程模型设计的,状态模型对数据库的系统功能设计有重要的参考价值。

2.1.3 计算机世界

信息世界中的信息经过数字化处理形成计算机能够处理的数据，就进入了计算机世界（机器世界、数据世界）。在信息转换为数据的过程中，对计算机硬件和软件（主要指数据库管理系统）都有限定，所以，信息的表示方法和信息处理能力要受到计算机硬件和软件的限制。也就是说，数据模型应符合具体计算机系统和 DBMS 的要求。

在计算机世界中会用到下列术语。

1. 数据项

数据项（item）是对象属性的数据表示。数据项有型和值之分，都要符合数据的编码要求。

（1）型。型是对数据特性的表示，它通过数据项的名称、数据类型、数据宽度和值域等来描述。例如，学号可以存储为字符型，15 个字符宽度。

（2）值。值是数据项的具体取值，如 1418855232。

2. 记录

记录（record）是实例的数据表示。记录有型和值之分。

（1）型。型是结构，由数据项的型构成。

（2）值。值表示对象中的一个实例，它的分量是数据项值。

例如，“姓名，性别，年龄，所在系”是学生数据的记录型，而“张三，男，23，计算机系”是一个学生的记录值，它表示学生对象的一个实例，“张三”“男”“23”“计算机系”都是数据项值。

3. 文件

文件（file）是对象的数据表示，是同类记录的集合，即同一个文件中的记录类型都应是一致的。例如，将所有学生的登记表组成一个学生数据文件，文件中的每条记录都要按“姓名，性别，年龄，所在系”的结构组织数据项值。

4. 数据模型

现实世界中的事物反映到计算机世界中就形成了文件的记录结构和记录，事物之间的相互联系就形成了不同文件间的记录的联系。记录结构及其记录联系的数据化的结果就是数据模型（data model）。

2.1.4 三种世界之间的对应关系

现实世界、信息世界和计算机世界这三个领域是由客观到认识、由认识到使用管理的三个不同层次，后一个领域是前一个领域的抽象描述。三个领域之间的术语对应关系如表 2-1 所示。

表 2-1　信息的三种世界术语的对应关系

现 实 世 界	信 息 世 界	计 算 机 世 界
实体	实例	记录
特征	属性	数据项
实体集	对象	数据或文件
实体间的联系	对象间的联系	数据间的联系
	概念(信息)模型	数据模型

现实世界、信息世界和计算机世界的转换关系如图 2-1 所示。

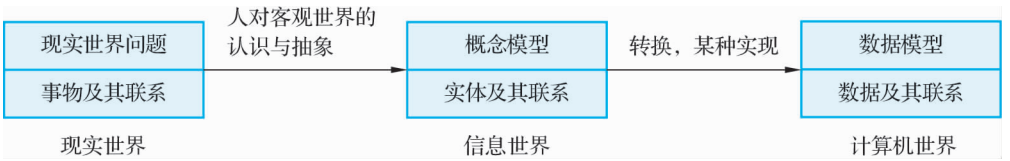

图 2-1　信息的三种世界的联系和转换过程

从图 2-1 中可以看出，现实世界的事物及联系通过系统分析成为信息世界的概念模型，而概念模型经过数据化处理转换为数据模型，进入数据世界。

2.2 数 据 模 型

现实世界中的事物必须先转换成计算机能够处理的数据，这需要采用数据模型来表示和抽象现实世界中的数据与信息。不同的数据模型实际上是提供给人们模型化数据和信息的不同工具。

2.2.1　数据模型的概念

数据模型主要用来抽象、表示和处理现实世界中的数据和信息，以便于采用数据库技术对数据进行集中管理和应用，是对客观事物及其联系的数学描述。

数据模型应满足三个方面的要求：一是能比较真实地模拟现实世界；二是容易被人们理解；三是便于在计算机上实现。

2.2.2　数据抽象的三个层次

数据模型要很好地满足上述提到的三个方面的要求目前尚很困难，在数据库系统中针对不同的使用对象和应用目的，通常采用逐步抽象的方法，在不同层次采用不同的数据模型，一般可分为概念层、逻辑层和物理层。

1. 概念层

概念层是数据抽象的最高层，其目的是按用户的观点对现实世界建模。概念层的数据模型称为概念数据模型，简称概念模型。概念模型独立于任何 DBMS，但容易向 DBMS 所支持的逻辑模型转换。

常用的概念模型有实体-联系模型（entity-relationship model，E-R 模型）。

2. 逻辑层

逻辑层是数据抽象的中间层，描述数据库数据整体的逻辑结构。这一层的数据抽象称为逻辑数据模型，简称数据模型。它是用户通过 DBMS 看到的现实世界，基于计算机系统的观点来对数据进行建模和表示。因此，它既要考虑用户容易理解，又要考虑便于 DBMS 实现。不同的 DBMS 提供不同的逻辑数据模型。常见的数据模型有层次模型（hierarchical model）、网状模型（network model）、关系模型（relation model）和面向对象模型（object oriented model）。

3. 物理层

物理层是数据抽象的底层，用来描述数据的物理存储结构和存储方法。这一层的数据抽象称为物理数据模型，它不但由 DBMS 的设计决定，而且与操作系统、计算机硬件密切相关。物理数据结构一般都向用户屏蔽，用户不必了解其细节。

2.3 概 念 模 型

概念模型用于信息世界的建模，是现实世界到信息世界的第一层抽象，是数据库设计人员进行数据库设计的有力工具，也是数据库设计人员和用户之间进行交流的语言，因此，概念模型一方面应该具有较强的语义表达能力，能够方便、直接地表达应用中的各种语义知识；另一方面应该简单、清晰，易于用户理解。

2.3.1 信息世界的七大基本概念

从现实抽象来的信息世界具有以下七大基本概念。

1. 实体

实体（entity）是客观存在的事物，也可以是抽象的概念或关系，如教师、学院、教师和学院之间的工作关系等。

2. 属性

属性（attribute）是实体所具有的某一特征，一个实体可以由若干个属性来描述。例如，实体学生可以用学号、姓名、性别、出生年月、所在院系、入学时间等属性来描述。

3. 实体型

实体型(entity type)即用实体类型名和所有属性来共同表示同一类实体,如学生(学号、年龄)。

4. 实体集

实体集(entity set)即同一类型实体的集合,如全体学生。

注意区分实体、实体型、实体集三个概念:实体是某个具体的个体,如学生中的王明,而实体集是一个个实体的某个集合,如王明所在的 2015 级计算机 2 班的所有学生,而实体型则是实体的某种类型(该种类型的所有实体具有相同的属性而已)。例如,学生这个概念,王明是学生,王明所在班级的所有同学都是学生,显然学生是一个更大且更抽象的概念,王明和王明所在班级的所有同学都比学生要更加具体。

5. 码

码(key)可以唯一标识一个实体的属性或属性集,如学号和每个学生实体一一对应,则学号可以作为码。

6. 域

简单地说,域(domain)就是指实体中属性的取值范围(属于某个域)。例如,学生的年龄的域为整数,因此,精确地讲,域是某种数据类型的值的集合。例如,学生的年龄是整数,但是又取不到所有整数,一般取值范围为 7～40 岁,而这个范围就属于整数集合。

7. 联系

联系(relationship)是指实体内部的联系(各属性之间的联系)和实体间的联系(数学抽象概念中强调实体型之间的联系,而现实生活中更加关注某几个具体的实体集之间的联系)。

2.3.2 E-R 模型

概念层数据模型是面向用户、面向现实世界的数据模型,它是对现实世界真实、全面的反映,它与具体的 DBMS 无关。常用的概念层数据模型有实体-联系(E-R)模型、语义对象模型。这里只介绍实体-联系模型。E-R 图由实体、属性和联系三个要素构成。

1. E-R 图的基本概念

1)实体

客观存在且可相互区别的事物称为实体。

E-R 图中的实体用于表示现实世界具有相同属性描述的事物的集合,它不是某一个具体事物,而是某一种类别所有事物的统称。实体可以是具体的人、事、物,也可以是抽象的概念或联系。例如,职工、学生、部门、课程等都是实体。

在 E-R 图中用矩形框表示具体的实体,把实体名写在框内。实体中的每一个具体的记录值(一行数据),如学生实体中的每个具体的学生称为一个实体的一个实例。

数据库开发人员在设计 E-R 图时，一个 E-R 图中通常包含多个实体，每个实体由实体名唯一标记。开发数据库时，每个实体对应于数据库中的一张数据库表，每个实体的具体取值对应于数据库表中的一条记录。

2)属性

E-R 图中的属性通常用于表示实体的某种特征，也可以使用属性表示实体间关系的特征。一个实体通常包含多个属性，每个属性由属性名唯一标记，画在椭圆内。E-R 图中实体的属性对应于数据库表中的字段。

在 E-R 图中，属性是一个不可再分的最小单元，如果属性能够再分，则可以考虑将该属性进行细分，或者可以考虑将该属性“升格”为另一个实体。例如，学生实体及属性实例如图 2-2 所示。

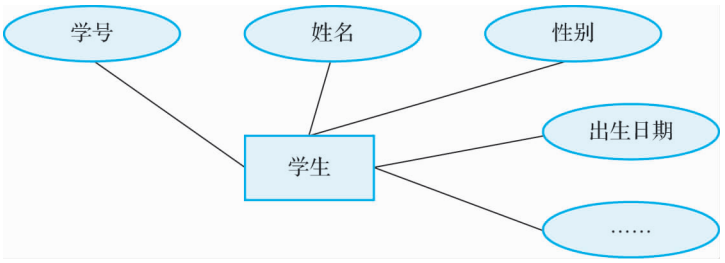

图 2-2　学生实体及属性实例

3)联系

联系是数据之间的关联集合，是客观存在的应用语义链。在现实世界中，事物内部及事物之间是有联系的，这些联系在信息世界中反映为实体内部的联系和实体之间的联系：实体内部的联系通常是指组成实体的各属性之间的联系；实体之间的联系通常是指不同实体集之间的联系。

在 E-R 图中，联系用菱形表示，框内写上联系名，并用连线将联系框与它所关联的实体连接起来，如图 2-3 所示。

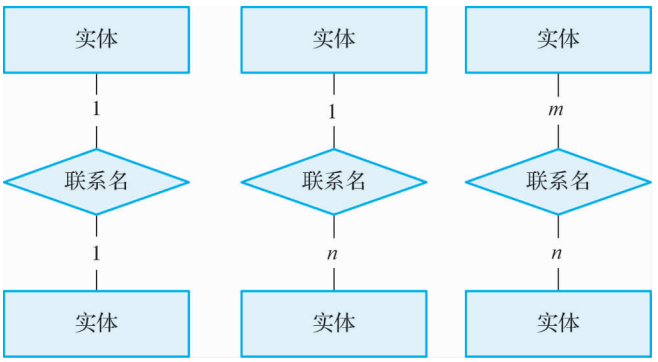

图 2-3　两个实体之间的三类联系

在 E-R 图中，基数表示一个实体到另一个实体之间关联的数目。基数是针对关系之间

的某个方向提出的概念，可以是一个取值范围，也可以是某个具体数值。从基数的角度可以将关系分为一对一(1∶1)、一对多(1∶n)、多对多(m∶n)关系。两个实体之间的联系可分为如下三类。

(1)一对一联系(1∶1)。如果对于实体集 A 中的每一个实体，在实体集 B 中至多有一个(也可以没有)实体与之联系，反之亦然，则称实体集 A 与实体集 B 有一对一联系，记为 1∶1。例如，学校里的一个系和正系主任(假设一个系只有一个正系主任，一个人只能担任一个系的正系主任)，则系和正系主任是一对一联系。

(2)一对多联系(1∶n)。如果对于实体集 A 中的每一个实体，在实体集 B 中有 n 个实体(n≥0)与之联系，反之，对于实体集 B 中的每一个实体，在实体集 A 中至多有一个实体与之联系，则称实体集 A 与实体集 B 有一对多联系，记为 1∶n。例如，一个系有多名教师，而每名教师只能在一个系工作，则系和教师之间是一对多联系。

(3)多对多联系(m∶n)。如果对于实体集 A 中的每一个实体，在实体集 B 中有 n 个实体(n≥0)与之联系，反之，对于实体集 B 中的每一个实体，在实体集 A 中有 m 个实体(m≥0)与之联系，则称实体集 A 与实体集 B 有多对多联系，记为 m∶n。例如，一门课程同时有若干个学生选修，而一个学生可以同时选修多门课程，则课程与学生之间具有多对多联系。

2.E-R 模型的设计原则与设计步骤

1)E-R 模型的设计原则

(1)属性应该存在于且只存在于所属的相关实体或关联中。该原则确保了数据库中的某个数据只存储在相关数据库表中，避免了数据冗余。

(2)实体是一个单独的个体，不能存在于另一个实体中成为其属性。该原则确保了一个数据库表中不能包含另一个数据库表，即不能出现“表中套表”的现象。

(3)同一个实体在同一个 E-R 图内仅出现一次。

2)E-R 模型的设计步骤

(1)划分和确定实体。

(2)划分和确定联系。

(3)确定属性。

(4)画出 E-R 模型。

2.4 逻 辑 模 型

逻辑模型(logical model)是具体的 DBMS 所支持的数据模型，任何 DBMS 都基于某种逻辑模型。目前数据库领域中最常用的逻辑模型有层次模型、网状模型、关系模型和面向对象模型。

2.4.1 层次模型

层次模型是数据库系统中最早出现的数据模型，它用树形结构表示各类实体及实体间的联系。现实世界中许多实体之间的联系本来就呈现出一种很自然的层次关系，如行政机构、家族关系等。层次模型数据库系统的典型代表是 IBM 公司的 IMS（information management systems），这是一个曾经被广泛使用的 DBMS，如图 2-4 所示。

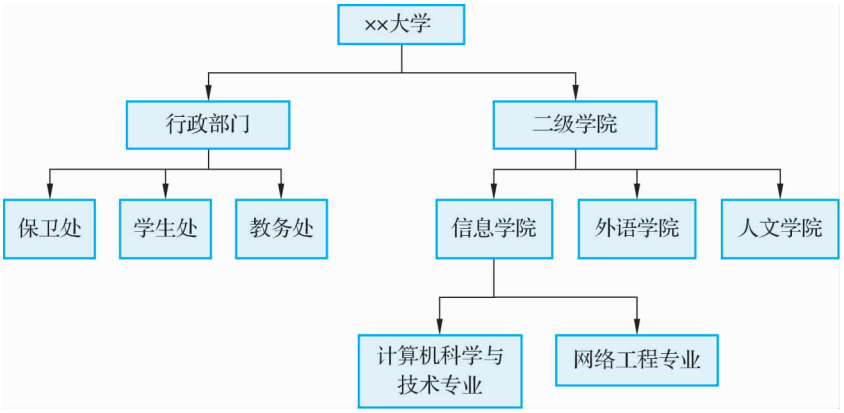

图 2-4 层次模型实例

2.4.2 网状模型

在现实世界中实体型间的联系更多的是非层次关系，用层次模型表示非树形结构是很不直接的，采用网状模型作为数据的组织方式可以克服这一弊病。网状模型去掉了层次模型的两个限制，允许节点有多个双亲节点，允许多个节点没有双亲节点。图 2-5 所示是网状模型的一个简单实例。

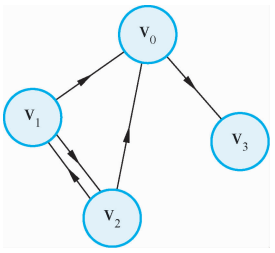

图 2-5 网状模型实例

2.4.3 关系模型

关系模型是目前最重要的也是应用最广泛的数据模型。简单地说，关系就是一张二维表，它由行和列组成。关系模型将数据模型组织成表格的形式，这种表格在数学上被称为关系，表格中存放数据。在关系模型中，实体及实体之间的联系都是用关系（二维表）来表示

的，见表 2-2。

表 2-2　用关系表表示的学生实体

Sno	Sname	Ssex	Sbirth	Sno	Sclass
1418855233	王一	男	1997-01-01	1102	商务 1301
1418855234	李三	男	1996-07-08	1102	商务 1301
1418855235	张平	女	1996-03-15	1102	商务 1301
…	…	…	…	…	…

2.4.4　面向对象模型

面向对象模型把实体表示为类，一个类描述了对象属性和实体行为。例如，CUSTOMER 类不仅含有客户的属性（如 CUST. ID、CUST. NAME 和 CUST. ADDRESS 等），还包含模仿客户行为（如修改订单）的过程。类-对象的实例对应于客户个体。在对象内部，类的属性用特殊值来区分每个客户（对象），但所有对象都属于类，共享类的行为模式。面向对象数据库通过逻辑包含（logical containment）来维护联系。

面向对象数据库把数据和与对象相关的代码封装成单一组件，外面不能看到其中的内容。因此，面向对象模型强调对象（由数据和代码组成）而不是单独的数据。这主要是从面向对象程序设计语言继承而来的。在面向对象程序设计语言中，程序员可以定义包含其自身的内部结构、特征和行为的新类型或对象类。这样，不能认为数据是独立存在的，而是与代码（成员函数的方法）相关，代码定义了对象能做什么（它们的行为或有用的服务）。面向对象模型的结构是非常容易变化的。与传统的数据库（如层次、网状或关系）不同，面向对象模型没有单一固定的数据库结构。

2.5　概念模型向逻辑模型的转换

概念模型中最常用的是 E-R 模型，逻辑模型中最常用的是关系模型，下面讲解 E-R 模型如何有效地转换为关系模型。

2.5.1　转换原则

E-R 模型由实体、实体的属性及实体之间的联系三部分组成，因此，将 E-R 模型转换为关系模型实际上就是将实体、实体的属性及实体之间的联系转换为关系模式，转换的一般规则如下：一个实体转换为一个关系模式。实体的属性就是关系的属性。对于实体之间的联系，则有以下不同的情况。

（1）一个 1∶1 联系可以转换为一个独立的关系模式，也可以与任意一端所对应的关系

模式合并，如果可以转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性，每个实体的码均是该关系的候选码。如果某一端实体对应的关系模式合并，则需要在该关系模式的属性中加入一个关系模式的码和联系本身的属性。

(2)一个 1∶n 联系可以转换为一个独立的关系模式，也可以与 n 端所对应的关系模式合并。若转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性，而关系的码为 n 端实体的码。

(3)一个 m∶n 联系可以转换为一个独立的关系模式，与该联系相连的各实体的码，以及联系本身的属性均转换为关系的属性，而实体的码组成关系的码或关系码的一部分。

2.5.2 转换实例

概念模型向逻辑模型转换的过程中要遵循以下规则。

(1)实体类型的转换：将每个实体类型转换成一个关系模式，实体的属性即为关系的属性，实体的码即为关系模式的码。

(2)联系类型的转换：根据不同的联系类型做不同的处理。

①若实体之间的联系是 1∶1，可以在两个实体类型转换成的两个关系模式中的任意一个关系模式中加入另一个关系模式的码和联系类型的属性。

②若实体之间的联系是 1∶n，则在 n 端实体类型转换成的关系模式中加入 1 端实体类型的码和联系类型的属性。

③若实体之间的联系是 m∶n，则将联系类型也转换成关系模式，其属性为两端实体类型的码加上联系类型的属性，而码为两端实体码的组合。

④3 个或 3 个以上实体之间的一个多元联系，不管联系类型是何种方法，总是将多元联系类型转换成一个关系模式，其属性为与该联系相连的各实体的码及联系本身的属性，其码为各实体码的组合。

习题

1. 信息的三种世界是什么？它们之间有什么联系？
2. 什么是概念模型？什么是逻辑模型？
3. 什么是实体、实体型、实体集、属性、码、E-R 图？它们之间有什么关系？
4. 如何将概念模型转化为逻辑模型？

第 3 章　关系模型和关系运算理论

关系数据模型是以集合论中的关系概念为基础发展起来的。关系模型中无论是实体还是实体间的联系均由单一的结构类型——关系(relation)来表示。在实际的关系数据库中的关系也称表。一个关系数据库就是由若干个表组成的。

3.1　关系数据结构

关系是关系模型的基本数据结构，是关系模型中最基本的概念。实体集及实体集之间的联系都可以用关系来表示，关系模型数据操作的对象和结果都是关系。关系可以有基于集合论的、直观的和基于代数的三种定义方法。基于集合论的方法是把关系定义为它的域的笛卡儿积的子集，直观的方法是把关系看作由行和列组成的二维表格，基于代数的方法则是把关系定义为关系模式从属性名到具体属性值的映射的集合。

3.1.1　关系的笛卡儿积

1. 域

域(domain)是一组具有相同数据类型的值的集合。例如，自然数集合、偶数集合、实数集合、长度不大于 50 的字符串集合、{男，女}表示的性别集合、0 到 100 之间的自然数集合等都是域。

2. 笛卡儿积的定义

给定一组域 $D_1, D_2, \cdots, D_n$，则

$$D_1 \times D_2 \times \cdots \times D_n = \{(d_1, d_2, \cdots, d_n) \mid d_i \in D_i, i = 1, 2, \cdots, n\}$$

称为域 $D_1, D_2, \cdots, D_n$ 的笛卡儿积。

由于域是值的集合，因此笛卡儿积是域上面的一种集合运算。笛卡儿积也是一个集合，它的每一个元素$(d_1, d_2, \cdots, d_n)$称为一个元组，它的每一个分量 d_i 都属于相应的域 D_i。n 为笛卡儿积的域的个数，称为笛卡儿积的元(也称度或目)，它表示元组中分量的个数，当 $n=$

4 时称元组为四元组，当 $n=5$ 时称元组为五元组。笛卡儿积的域可以部分相同甚至全部相同。例如，D_1 和 D_2 可以是相同的域。

3. 笛卡儿积的基数

若 $D_i(i=1,2,\cdots,n)$ 为有限集，其基数记为 $m_i(i=1,2,\cdots,n)$，则笛卡儿积 $D_1\times D_2\times\cdots\times D_n$ 的基数 M 为 $\prod_{i=1}^{n}m_i$。

【例 3-1】 设 $D_1=\{090101,090102,090201\}$，$D_2=\{$天津，河北$\}$，$D_3=\{$软件工程，信息管理$\}$，域 D_1、D_2、D_3 分别表示学生的学号集合、籍贯集合、专业集合，则 D_1、D_2、D_3 的笛卡儿积为：$D_1\times D_2\times D_3=\{$(090101，天津，软件工程)，(090101，天津，信息管理)，(090101，河北，软件工程)，(090101，河北，信息管理)，(090102，天津，软件工程)，(090102，天津，信息管理)，(090102，河北，软件工程)，(090102，河北，信息管理)，(090201，天津，软件工程)，(090201，天津，信息管理)，(090201，河北，软件工程)，(090201，河北，信息管理)$\}$，该笛卡儿积的基数为 $M=3\times2\times2=12$，一共有 12 个三元组。

4. 关系

笛卡儿积 $D_1\times D_2\times\cdots\times D_n$ 的子集称为域 $D_1,D_2,\cdots,D_n$ 上的关系，通常记为 $R(D_1,D_2,\cdots,D_n)$。这里 R 表示关系的名字，n 是关系的元(也称度或目)。

关系是一个集合，关系中的每个元素是关系中的元组，本章通常用 t 表示，元组的第 i 个分量用 $t[i]$ 表示。如果一个关系的元组数目是无限的，则称其为无限关系，否则，称其为有限关系。

笛卡儿积的定义在实际应用中存在的问题：在关系模型中，关系中的元组是对现实世界中的实体及实体间联系的抽象描述，具有特定的语义。而笛卡儿积在域上做乘积运算时并没有考虑所得元组是否具有实际意义，在【例 3-1】中，该笛卡儿积中的大部分元组是没有意义的。对一个实体或实体间的联系而言，其属性的次序是可以交换的，因而从理论上讲，关系的元组分量应该是无序的，而笛卡儿积的元组分量并不满足交换律。另外，笛卡儿积的定义并没有对其元组数目进行限制，笛卡儿积可以是一个无限集合，但是计算机的存储空间及运算能力却是有限的，因此在关系数据库中研究无限关系是没有意义的。

基于上述原因及关系操作的需要，当关系作为关系模型的数据结构时，除了要满足集合的特性外，还需要在笛卡儿积的基础上做出如下限定和扩充。

(1)关系必须是一个有限集合，关系模型中只限于研究有限关系。

(2)关系的元组分量是无序的，元组分量可以交换次序。

(3)关系的元组分量必须取原子值，即每一个分量都是不可分解的数据项。

(4)关系的元组具有特定的语义。

因此，关系是一种经过规范化的笛卡儿积的有限子集。

3.1.2 关系的二维表格

关系采用笛卡儿积进行定义尽管十分严格，但是对一般用户而言显得有些抽象与晦涩。从直观形式上看，如果将关系中的一个元组看作二维表格中的一行，那么关系实质上是一张规范化的二维表格，这就是关系的非形式化定义。

1. 关系与一般表格的术语对比

关系与一般表格的术语对比见表 3-1。

表 3-1 关系与一般表格的术语对比

关 系 术 语	一般表格术语
关系名	表名
关系模式	表头(表格的描述)
关系	规范化的二维表格
元组	行或记录
属性	列
属性名	列名
属性值	列值
分量	一条记录中的一个列值
非规范关系	表中有表(表中嵌有小表)

2. 关系表的规范化限定

关系从直观上看是一张二维表格，但严格地说，其并不是通常意义下的“表格”，而是一种规范化的二维表格，称为关系表，在不引起混淆的情况下，简称表。在关系模型中，根据关系的笛卡儿积定义以及关系作为一个集合所具有的特性，通常对关系表做出如下规范化限定。

(1)列值同质性：关系表中同一列的列值是同一类型的数据，来自同一个域。

(2)异列同域性：关系表中不同的列可以来自同一个域。

(3)列名唯一性：关系表的每一列称为关系的一个属性，对应一个域，由于不同列可以来自同一个域，为了加以区分，必须对每列起一个不相同的名字。

(4)列的无序性：关系表中列的次序可以任意交换，从理论上讲关系中的属性是无序的。

(5)行的无序性：关系表中行的次序可以任意交换，因为关系的元组作为一个集合元素是无序的。

(6)行值相异性：关系表中的任意两行不能完全相同，因为关系作为一个集合，其中不允许存在两个完全相同的元素(元组)。

(7)列值原子性：关系表中不允许“表中有表”，因为关系中每一个分量是不可分解的，不

允许出现组合数据项。

(8)行值有义性:关系表中的每一行是对某一实体或实体间联系的抽象描述,具有特定的语义。

关系表是对关系的一种非形式化定义,关系表和关系这两个术语通常可以互相通用。对一般用户而言,通过关系表来描述关系比通过笛卡儿积定义来描述关系更为直观、更易理解。需要说明的是,在许多实际的关系数据库产品中,关系表并不完全符合以上规范化限定,有的数据库产品仍然区分了属性顺序和元组顺序;有的数据库产品允许关系表中存在两个完全相同的元组,除非用户特别定义了相应的约束条件。

3. 关系表的三种类型

在关系数据模型中,关系表可以有以下三种类型。

(1)基本表:通常又称为基表或基本关系,它是实际存在的表,是实际存储数据的逻辑表示。

(2)查询表:是对一个或多个基本表进行查询所得结果对应的表。

(3)视图表:是由基本表或其他视图导出的表,是虚表,不对应实际存储的数据。

3.1.3 关系模式

关系模式是对关系的逻辑结构和属性的描述,是关系的型;关系是关系模式的实例,是其关系模式的属性名到具体属性值的映射的集合。

1. 关系模式的定义

定义 3-1 关系的描述称为关系模式,它可以形式化地表示为一个五元组 $R(U,D,DOM,F)$。其中,R 为关系名,U 为属性集(组成该关系的属性名的集合),D 为 U 中属性的域的集合,DOM 为属性向域的映像集合(各属性分别来自哪个域),F 为 U 中属性与属性之间的数据依赖集合。

关系模式通常可以简记为 $R(U)$ 或 $R(A_1,A_2,\cdots,A_n)$,其中,R 为关系名,$A_1,A_2,\cdots,A_n$ 为属性名。而域名及属性向域的映像常常直接说明为属性的类型、长度。

从定义 3-1 可知,关系模式是对关系的结构性描述,具体来说,主要从以下三个方面描述了关系的一般特征。

(1)关系的元组组成描述:关系模式通过属性集 U 描述了元组由哪些属性构成。

(2)关系的元组语义描述:元组语义实质上是一个 n 目谓词(n 是属性集中属性的个数),凡是使该 n 目谓词为真的笛卡儿积中的元素(或者说凡符合元组语义的那部分元素)的全体就构成了该关系模式的关系。关系模式描述了表现元组语义的谓词。

(3)关系的属性约束描述:关系模式通过域集 D、属性向域的映像集合 DOM 和属性间数据的依赖关系集合 F 描述了关系的属性的数据类型和长度、属性与属性之间的数据依赖等约束条件。

2. 关系与关系模式的联系和区别

关系是关系模式的实例，是关系模式的具体取值。在数据模型中有型和值的概念，型是指对某一类数据的结构和属性的说明，值是型的一个具体赋值。关系模式是型，关系是值。从直观上看，关系模式是一张已经编制好的待填的空表，而关系是一张已经填好值的具体表；一个空表可以填入不同的内容构成不同的具体表，一个关系模式可以实例化成不同的关系。

【例 3-2】 设【例 3-1】中笛卡儿积的子集{(090101，天津，软件工程)、(090102，河北，软件工程)、(090201，河北，信息管理)}是一个关系，其关系名设为“学生”，三个属性名分别设为“学号”“籍贯”和“专业”，则此关系的数据结构分别可用关系的笛卡儿积定义、关系表和关系模式表示。

3.1.4 键

键(key)也称为码，是关系数据结构中的一个重要概念。键由一个或多个属性组成，常用来标识关系的元组，在关系数据库中常用它来导航数据。键又可分为超键(super key)、候选键(candidate key)、主键(primary key)和外键(foreign key)四种类型。

1. 超键

在关系中能够唯一标识元组的属性或属性组称为关系模式的超键。例如，【例 3-2】中的属性子集{学号，籍贯}是学生关系模式的超键。

2. 候选键

不含多余属性的超键称为关系模式的候选键。也就是说，在候选键中如果再删除任意一个属性，就不是键了。包含在候选键中的属性称为主属性，不包含在任何候选键中的属性称为非主属性或非键属性。例如，【例 3-2】中的属性子集{学号}是学生关系模式的候选键，{学号，籍贯}是超键但不是候选键，因为其中含有非主属性“籍贯”。

3. 主键

用户选作元组标识的候选键称为关系模式的主键。主键的值可以用来识别和区分元组，每个元组的主键的值不能相同。在关系模式中，主键用下划线来标识。例如，【例 3-2】中如果用户使用{学号}作为查询元组的标识，则{学号}是学生关系模式的主键。在实际应用中，如果不加说明，键通常是指主键。

4. 外键

如果关系模式 R_1 中的某一属性(或属性组)F 与关系模式 R_2 的主键相对应，但不是 R_1 的超键，则称 F 是关系模式 R_1 的外键。需要说明的是，R_1、R_2 不一定是不同的关系模式，也可以是同一关系模式；外键不一定要与相对应的主键同名，定义在相同的值域上即可。不过

在实际应用中，为了便于辨识，当外键与相对应的主键位于不同关系模式中时，通常会给它们取相同的名字。在关系模式中，外键用下划波浪线来标识。例如，如果【例 3-2】中的属性子集{专业}是某关系模式的主键，则{专业}是学生关系模式的外键。

3.2 关系的完整性约束

3.2.1 实体完整性

实体完整性规则(entity integrity rule)是指关系中的元组在组成主键的属性上不能取空值。

实体完整性规则规定的“元组在组成主键的属性上不能取空值”指的是元组在组成主键的所有属性上都不能取空值，对多个属性组成的主键的取值即使部分为空也不允许。

实体完整性规则的实质是“要保证实体集中的实体能相互区分”。如果某元组主键的值为空，就说明该元组不可标识，即意味着实体集中存在不可区分的实体。

3.2.2 参照完整性

参照完整性也称引用完整性。与实体完整性是一种关系内部的约束不同的是，参照完整性是不同关系之间或同一关系不同属性之间的一种数据引用约束。

1. 关系间的属性引用类型

在关系模型中实体集及实体集间的联系都是用关系来描述的，这样就自然存在着关系与关系间的属性引用。关系间的引用类型有以下三种情况。

1)两个关系间的属性引用

【例 3-3】 设学生实体和专业实体可用如下两个关系模式表示，其中，主键用下划线标识，外键用下划波浪线标识：

学生(学号，姓名，出生年月，籍贯，专业号)

专业(专业号，专业名)

显然，如果一个学生实体分配有专业号，则该专业号的取值是专业关系中确实存在的专业的专业号。也就是说，学生关系中的“专业号”属性在取值时需要参照专业关系的“专业号”属性值。

2)两个以上关系间的属性引用

【例 3-4】 设学生实体、课程实体及它们之间的联系选修可以用如下三个关系模式表示：

学生(学号，姓名，年龄，籍贯，专业号)

课程(课程号,课程名)

选修(学号,课程号,成绩)

在选修关系模式中,“学号”属性不仅是外键,而且是主键的一部分,“课程号”属性也如此。与【例 3-3】相同,选修关系中的“学号”和“课程号”属性在取值时需要分别参照学生关系的“学号”属性值和课程关系的“课程号”属性值。

3)同一关系内部属性间的引用

【例 3-5】 设课程之间有先修、后修联系,用关系模式可表示为:课程(课程号,课程名,先修课程号)。在该关系模式内部存在属性间的引用,外键{先修课程号}引用了位于同一模式中的主键{课程号},外键{先修课程号}取值时需要参照主键{课程号}的值。

2. 参照完整性规则

参照完整性规则(referential integrity rule)是指若关系模式 R_1 的外键 F 与关系模式 R_2 的主键相对应,那么在 R_1 的关系中,元组在 F 上的取值要么为空值,要么等于 R_2 关系中的某个主键值。

在参照完整性规则中,关系模式 R_1 的关系称为参照关系,关系模式 R_2 的关系称为被参照关系。

参照完整性规则的实质是“不允许引用不存在的实体”。在参照关系中,元组在外键的属性上可以取空值,表示该元组在此属性上的值尚未确定,但前提条件是该外键的属性不是其所在关系的主键的属性。

3.2.3 用户自定义的完整性

用户自定义的完整性约束是用户针对某一关系数据库所设置的约束条件,它反映了某一具体应用领域中的数据所应满足的特定语义要求。

关系数据库中的数据通常具有特定的应用环境,往往需要遵循一些特殊的约束条件。在关系模式中,尽管对属性定义了相应的数据类型和长度,但有时仍然满足不了用户的需求,为了解决这一问题,需要用户对关系数据库中的数据做进一步约束。

3.3 关 系 操 作

3.3.1 关系操作的类型

关系操作主要包括关系查询和关系更新两种类型。

(1)关系查询是最常用的一类关系操作,是关系操作的核心。通过关系查询操作,用户