

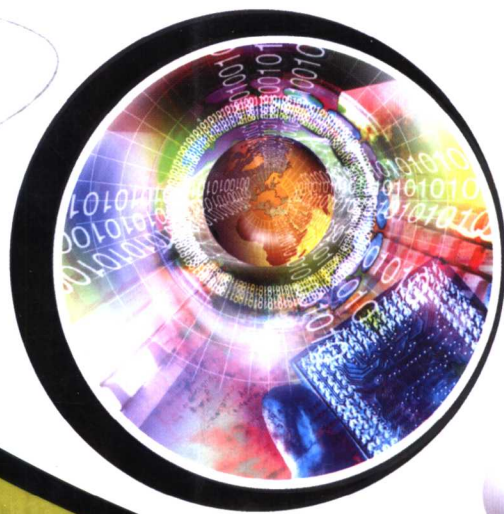
现代软件工程专业系列教材

# 数据库系统设计

SHUJUKU XITONG SHEJI  
YU KAIFA 与开发

朱扬勇

编著



清华大学出版社

<http://www.tup.tsinghua.edu.cn>



北京交通大学出版社

<http://press.bjtu.edu.cn>



TP311.13/304

2007

现代软件工程专业系列教材

# 数据库系统设计与开发

朱扬勇 编著

清华大学出版社  
北京交通大学出版社

· 北京 ·

## 内 容 简 介

本书全面地介绍了当前数据库系统开发的各个方面,内容包括数据库的基本知识、数据库设计的基本原理、数据库概念设计的ER方法和语义对象方法、关系数据库规范化原理和方法、数据持久化技术、数据库保护技术、数据库性能调优、遗留数据库重建技术,等等。本书还介绍了数据资源开发利用技术、数据库系统的开发方法和项目管理等内容。

本书在介绍各种原理、方法和技术时,强调它们的实际应用,以满足数据库系统开发高级工程师的需要。阅读本书的读者需要具有计算机及相关专业的基础知识,尤其需要具备数据库知识、软件工程知识及一定的程序设计经验。

本书主要作为计算机软件专业硕士研究生教材,亦可作为计算机软件专业大学本科高年级教材或教学参考书,以及软件产业高级技术管理人员的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13501256678 13801310933

## 图书在版编目(CIP)数据

数据库系统设计与开发 / 朱扬勇编著. —北京:清华大学出版社;北京交通大学出版社, 2007.8

(现代软件工程专业系列教材)

ISBN 978-7-81123-131-1

I. 数… II. 朱… III. 数据库系统 IV. TP311.13

中国版本图书馆CIP数据核字(2007)第114019号

策划编辑:刘 洵 招富刚

责任编辑:刘 洵

出版发行:清华大学出版社 邮编:100084 电话:010-62776969

北京交通大学出版社 邮编:100044 电话:010-51686414

印刷者:北京鑫海金澳胶印有限公司

经 销:全国新华书店

开 本:203×280 印张:16 字数:452千字

版 次:2007年8月第1版 2007年8月第1次印刷

书 号:ISBN 978-7-81123-131-1/TP·371

印 数:1~4000册 定价:29.00元

本书如有质量问题,请向北京交通大学出版社质监组反映。对您的意见和批评,我们表示欢迎和感谢。

投诉电话:010-51686043, 51686008; 传真:010-62225406; E-mail: press@bjtu.edu.cn。

## 前言

从20世纪80年代开始,计算机得到普及,被应用到了各行各业各领域。其中有两样东西功不可没,一是微型计算机,二是数据库。微型计算机让每个人都有能力、有机会使用计算机,数据库让每个人都有需要、有必要使用计算机。进入21世纪后,大众可能更加熟悉“Intel Inside; Windows Outside”,但人们真正需要的是其中的数据库。所谓的国民经济和社会的信息化其实就是将现实世界中的事物以数据的形式存储到计算机系统中,即信息化是一个生产数据的过程。因此,如何在计算机系统中存储、管理和使用数据应该是一个长期的问题。

数据库技术已经经历了近四十年的发展和应用,先后出现过层次数据库、网状数据库、关系数据库、演绎数据库(逻辑数据库,知识库)、面向对象数据库、工程数据库、时态数据库、地理数据库、多媒体数据库,等等。其中,关系数据库从20世纪80年代以来获得了广泛的应用,到目前为止仍然是主流数据库。这是因为关系数据库是建立在良好数学基础之上的,包括以关系代数为基础的数据库管理系统和以规范化理论为基础的关系数据库设计技术。

长期以来,数据库系统是由“数据库和访问数据库的软件系统”两部分组成。因此,需要先设计好一个数据库,然后在这个数据库上开发应用程序,即所谓的数据驱动。由于数据库设计技术和应用程序设计技术分别来自于数据库研究领域和软件工程研究领域,使得在实际系统开发时,常常有一个数据库小组和一个应用程序小组各行其是,在技术和文化上都存在严重的“阻抗不匹配”问题,难以融合,从而导致数据库系统开发危机重重。虽然两个领域的专家都在努力解决这个问题,也提出了面向对象技术和持久程序设计技术等,但到目前为止并没有取得里程碑式的进展。“一个位于对象空间的应用系统”这样的表述听起来很美,但由于没有好用的面向对象数据库管理系统,所以对象要在关系数据库管理系统上做持久化,持久化为关系表。因此,系统开发在对象空间上进行,并最终在对象空间上建立数据库系统,数据库只是持久化对象的一个存放地,即所谓的对象驱动。但是,开发出来的系统本质上仍然是“一个数据库和访问数据库的软件系统”。数据库系统开发仍然需要良好的数据库设计(如规范化),数据库性能调优也常常在数据库逻辑模式上进行。这对数据库系统开发工程师而言是一个巨大的挑战。

信息化是一个生产数据的过程,随着信息化进程的发展,数据被大量生产并存储在计算机系统中形成了数据资源。这些数据就是社会发展和人们工作、生活的记录。因此,发现这些数据所含的规律也就是发现社会发展和人们工作、生活中的规律,这是非常有价值的。数据资源是战略资源,需要良好地加以开发利用。近年来,数据资源的开发利用引起了广泛重视,发展了存储历史数据的数据仓库技术,并在统计学方法和人工智能技术的基础上发展了联机分析处理技术和数据挖掘技术。进入21世纪,数据仓库技术和数据挖掘技术更是被认为是互联网之后的技术热点,具有极大的发展潜力和极好的应用前景。这些技术形成了数据利用的技术内涵。另外,如果一个数据库系统不再使用了,就产生了遗留数据库。显然,遗留数据库仍然是有价值的数据库资源,可以用于新系统开发或旧系统重建,也可以用于历史数据分析。但由于遗留数据库往往残缺不全、文档稀少,所以需要对遗留数据库进行重建,并加以利用。

本书全面地介绍当前数据库系统开发的各个方面,内容包括数据库的基本知识、数据库设计的基本原理、数据库概念设计的ER方法和语义对象方法、关系数据库规范化原理和方法、数据持久化技术、数据库保护技术、数据库性能调优技术、遗留数据库重建技术,等等。本书还介绍了数据资源开发利用技术、数据库系统的开发方法和项目管理等内容。

本书的内容安排大致如下。

第1章介绍了数据库的基本知识,主要包括数据库概念、数据库发展、数据库分类、数据库系统类型、数据建模与数据库系统开发,还介绍了运行数据库的计算机系统。其中,关于数据建模方面的讨论建议读者做一些思考,这对于从宏观上把握数据库系统开发方法及其应用有益。关于基本知识更详细的介绍需要参考“数据库概论”方面的书。

第2章介绍了数据库设计原理,包括数据库设计框架、数据库等价性等内容,是全书指导性和纲领性的一章。尤其是数据库等价性方面的内容,是研究数据库设计方法的指导。

第3章系统地介绍了数据库概念设计的ER方法,主要有基本ER构造、高级ER构造、建立ER模型的方法(包括划分实体和属性、找出泛化和汇集层次、视图集成等技术)、ER模型的集成等内容。

第4章系统地介绍了数据库概念设计的语义对象方法,主要有语义对象概念、语义对象种类、语义对象建模等。

第5章系统地介绍了关系数据库的规范化原理和方法,包括范式、规范化框架和规范化算法等内容。这一章数学理论性较强,一般读者只要理解基本概念和规范化算法就可以了,本章中的一些必要的证明是为有兴趣理解规范化原理的读者准备的。

第6章介绍了概念模型向关系数据库逻辑模式的转换,包括将ER模型转换成逻辑模式、将语义对象模型转换成逻辑模型、将UML类图映射到关系数据库逻辑模式和XML的持久化等。

第7章介绍了数据库保护,包括数据库完整性、安全性、事务、数据库备份与恢复、长事务管理等。其中的长事务管理是当前大型应用开发所常常面临的问题,本章做了原理性介绍。

第8章系统地介绍了数据库系统性能调优的各个方面,包括硬件调优、数据库逻辑设计调优、数据库物理设计调优、数据库访问调优等,还介绍DBMS的性能基准方面的内容。

第9章原理性地介绍了遗留数据库重建技术。遗留数据库重建是一个还没有引起足够重视的领域,所以提出的具体方法还不多。但是,遗留数据库越来越多这个事实,将推动该领域发展。

第10章介绍了数据利用技术,包括数据仓库、联机分析处理和数据挖掘技术。

第11章简要介绍了数据库系统开发方法,主要针对数据库系统的应用程序开发,介绍软件工程项目管理技术。

本书的各个章节基本上是独立的,读者可以根据需要进行选择。阅读本书时,读者需要有“数据库概论”和“软件工程概论”方面的知识,最好有数据库系统的开发经验。全书教学可以在54学时内完成。每章后面的思考题不是作为作业设计的,而是为了对数据库系统开发所面临的技术问题有更深入的研究而设计的。希望对这些问题的思考、讨论、分析和研究,有助于读者全面掌握相关技术和方法,其中一些问题可作为硕士研究生论文选题。需要本书的Power Point课件的读者可联系 [yyzhu@fudan.edu.cn](mailto:yyzhu@fudan.edu.cn)。

由于数据库的应用对象可能是企业、机构、学校、部门等,所以为了叙述上的方便,我们将它们统称为“应用体”,权且作为enterprise一词的翻译。

感谢北京交通大学出版社的编辑们为本书的尽早出版所做的工作。

书中的不妥之处全属作者水平有限所致,如蒙读者指正,不胜感谢。

朱扬勇

2007年8月于复旦大学

# 目 录

|                          |      |
|--------------------------|------|
| 第1章 数据库系统基础知识 .....      | (1)  |
| 1.1 数据库概念 .....          | (1)  |
| 1.1.1 从文件系统到数据库系统 .....  | (1)  |
| 1.1.2 数据库系统 .....        | (2)  |
| 1.1.3 数据模型 .....         | (3)  |
| 1.1.4 数据库系统的体系结构 .....   | (4)  |
| 1.1.5 与数据库相关的人员 .....    | (4)  |
| 1.2 数据库发展 .....          | (5)  |
| 1.2.1 层次数据库和网状数据库 .....  | (5)  |
| 1.2.2 关系数据库 .....        | (6)  |
| 1.2.3 阻抗不匹配与后关系数据库 ..... | (7)  |
| 1.2.4 数据库现状 .....        | (7)  |
| 1.3 数据库分类 .....          | (10) |
| 1.3.1 根据数据模型分类 .....     | (10) |
| 1.3.2 根据体系结构分类 .....     | (10) |
| 1.3.3 根据数据类型分类 .....     | (11) |
| 1.4 数据库系统类型 .....        | (12) |
| 1.5 数据建模与数据库系统开发 .....   | (13) |
| 1.5.1 人与计算机的交流 .....     | (13) |
| 1.5.2 建模的困难与方法的局限 .....  | (14) |
| 1.5.3 数据库系统开发内容 .....    | (15) |
| 1.5.4 数据建模与软件工程 .....    | (15) |
| 1.6 运行数据库的计算机系统 .....    | (16) |
| 1.6.1 集中式系统 .....        | (16) |
| 1.6.2 文件服务器系统 .....      | (17) |
| 1.6.3 分布式系统 .....        | (18) |
| 1.6.4 客户-服务器数据库系统 .....  | (18) |
| 1.6.5 浏览器系统 .....        | (20) |
| 1.6.6 说明 .....           | (21) |
| 1.7 小结 .....             | (21) |
| ◇思考题 .....               | (21) |
| 第2章 设计原理 .....           | (23) |
| 2.1 为什么要进行数据库设计 .....    | (23) |
| 2.1.1 现实数据表格的转换 .....    | (23) |
| 2.1.2 数据操作的异常现象 .....    | (24) |

|            |               |             |
|------------|---------------|-------------|
| 2.1.3      | 关系表的意义        | (25)        |
| 2.1.4      | 错误的数据库        | (26)        |
| 2.1.5      | 对象持久化         | (26)        |
| 2.2        | 数据库概念设计       | (27)        |
| 2.2.1      | 为什么要进行概念设计    | (27)        |
| 2.2.2      | ER 方法         | (28)        |
| 2.2.3      | 语义对象方法        | (29)        |
| 2.2.4      | UML 方法        | (30)        |
| 2.2.5      | XML 方法        | (31)        |
| 2.2.6      | 概念设计讨论        | (32)        |
| 2.3        | 数据库设计的框架      | (33)        |
| 2.3.1      | 数据库设计的目标      | (33)        |
| 2.3.2      | 数据库生命周期       | (33)        |
| 2.3.3      | 数据库设计的内容      | (34)        |
| 2.3.4      | 数据库设计运用的方法    | (35)        |
| 2.4        | 数据库的等价性       | (35)        |
| 2.4.1      | 等价性的意义        | (35)        |
| 2.4.2      | 等价性的定义        | (36)        |
| 2.4.3      | 等价性的实现        | (36)        |
| 2.4.4      | 对象持久化的等价性     | (36)        |
| 2.5        | 其他问题          | (37)        |
| 2.5.1      | 数据库保护         | (37)        |
| 2.5.2      | 数据库性能         | (38)        |
| 2.5.3      | 遗留数据库         | (38)        |
| 2.5.4      | 数据字典          | (38)        |
| 2.6        | 小结            | (40)        |
| ◇          | 思考题           | (41)        |
| <b>第3章</b> | <b>ER 方法</b>  | <b>(42)</b> |
| 3.1        | 基本 ER 构造      | (42)        |
| 3.1.1      | 实体            | (42)        |
| 3.1.2      | 联系            | (43)        |
| 3.1.3      | 属性            | (44)        |
| 3.2        | 高级 ER 构造      | (45)        |
| 3.2.1      | 泛化            | (46)        |
| 3.2.2      | 汇集            | (47)        |
| 3.2.3      | 弱实体、存在依赖和标识依赖 | (48)        |
| 3.3        | 建立 ER 模型      | (48)        |
| 3.3.1      | 建立 ER 模型的基本问题 | (49)        |
| 3.3.2      | 区分实体和属性       | (50)        |
| 3.3.3      | 找出汇集层次        | (51)        |
| 3.3.4      | 找出泛化层次        | (51)        |

|                               |      |
|-------------------------------|------|
| 3.3.5 找出弱实体 .....             | (52) |
| 3.3.6 定义联系 .....              | (52) |
| 3.3.7 建立 ER 模型的几点原则 .....     | (54) |
| 3.4 ER 模型的集成 .....            | (54) |
| 3.4.1 集成的基本问题 .....           | (54) |
| 3.4.2 简单的 ER 模型集成 .....       | (55) |
| 3.4.3 ER 模型集成的基本步骤 .....      | (56) |
| 3.4.4 举例 .....                | (58) |
| 3.5 实际应用中的注意事项 .....          | (59) |
| 3.5.1 ER 方法和 ER 工具软件的差异 ..... | (59) |
| 3.5.2 ER 图的复杂性 .....          | (60) |
| 3.6 小结 .....                  | (61) |
| ◇思考题 .....                    | (61) |
| <b>第4章 语义对象模型及其建模</b> .....   | (62) |
| 4.1 语义对象的概念 .....             | (62) |
| 4.1.1 语义对象 .....              | (62) |
| 4.1.2 属性 .....                | (63) |
| 4.1.3 语义对象标识 .....            | (64) |
| 4.1.4 语义对象实例 .....            | (65) |
| 4.1.5 语义对象视图 .....            | (65) |
| 4.2 语义对象的种类 .....             | (66) |
| 4.2.1 简单语义对象 .....            | (66) |
| 4.2.2 组合语义对象 .....            | (67) |
| 4.2.3 复合语义对象 .....            | (68) |
| 4.2.4 混合语义对象 .....            | (70) |
| 4.2.5 关联语义对象 .....            | (70) |
| 4.2.6 父/子类型语义对象 .....         | (71) |
| 4.2.7 原型/版本语义对象 .....         | (73) |
| 4.3 语义对象的建模 .....             | (73) |
| 4.3.1 某大学数据库 .....            | (73) |
| 4.3.2 语义对象模型 .....            | (77) |
| 4.3.3 语义对象建模的步骤 .....         | (77) |
| 4.4 与相关概念的比较 .....            | (78) |
| 4.4.1 面向对象程序设计与语义对象 .....     | (78) |
| 4.4.2 语义对象模型和 ER 模型的比较 .....  | (78) |
| 4.5 小结 .....                  | (79) |
| ◇思考题 .....                    | (79) |
| <b>第5章 关系数据库模式规范化</b> .....   | (81) |
| 5.1 规范化的意义和基本假设 .....         | (81) |
| 5.1.1 关系数据库 .....             | (81) |

|            |                    |              |
|------------|--------------------|--------------|
| 5.1.2      | 规范化的意义             | (82)         |
| 5.1.3      | 泛关系假设              | (82)         |
| 5.2        | 基本概念               | (83)         |
| 5.2.1      | 投影与联结              | (83)         |
| 5.2.2      | 函数依赖               | (84)         |
| 5.2.3      | Armstrong 公理系统     | (85)         |
| 5.2.4      | 闭包和投影              | (86)         |
| 5.2.5      | 覆盖                 | (87)         |
| 5.2.6      | 范式                 | (88)         |
| 5.3        | 模式规范化              | (89)         |
| 5.3.1      | 规范化的框架             | (89)         |
| 5.3.2      | 无损联结分解             | (90)         |
| 5.3.3      | 保持函数依赖的分解          | (91)         |
| 5.4        | 规范化算法              | (92)         |
| 5.4.1      | 计算属性闭包             | (92)         |
| 5.4.2      | 计算投影               | (92)         |
| 5.4.3      | 计算最小覆盖             | (94)         |
| 5.4.4      | 通用分解方法             | (95)         |
| 5.4.5      | 3NF 分解             | (97)         |
| 5.4.6      | BCNF 分解            | (98)         |
| 5.5        | 小结                 | (100)        |
| ◇          | 思考题                | (100)        |
| <b>第6章</b> | <b>概念模型转换到关系模式</b> | <b>(101)</b> |
| 6.1        | 基本问题               | (101)        |
| 6.1.1      | 正确的概念模型转换          | (101)        |
| 6.1.2      | 数据库实现的难易程度         | (102)        |
| 6.1.3      | 转换方法问题             | (103)        |
| 6.2        | 将 ER 模型转换成关系模式     | (103)        |
| 6.2.1      | 转换实体               | (103)        |
| 6.2.2      | 转换弱实体              | (103)        |
| 6.2.3      | 转换汇集实体             | (104)        |
| 6.2.4      | 转换泛化层次             | (105)        |
| 6.2.5      | 转换联系               | (105)        |
| 6.2.6      | 从 ER 模型中产生函数依赖     | (109)        |
| 6.3        | 将语义对象模型转换成关系模式     | (109)        |
| 6.3.1      | 转换简单语义对象           | (110)        |
| 6.3.2      | 转换组合语义对象           | (110)        |
| 6.3.3      | 转换复合语义对象           | (110)        |
| 6.3.4      | 转换混合语义对象           | (112)        |
| 6.3.5      | 转换关联语义对象           | (113)        |
| 6.3.6      | 转换父/子类语义对象         | (113)        |

|                            |       |
|----------------------------|-------|
| 6.3.7 转换原型/版本语义对象 .....    | (114) |
| 6.4 将 UML 对象类映射为关系模式 ..... | (114) |
| 6.4.1 映射单个类 .....          | (114) |
| 6.4.2 映射泛化(继承)结构或类层次 ..... | (115) |
| 6.4.3 映射类的联系 .....         | (116) |
| 6.5 XML 文档的持久存储 .....      | (118) |
| 6.5.1 持久存储方式 .....         | (118) |
| 6.5.2 持久存储的框架 .....        | (118) |
| 6.5.3 XML 文档与关系模式的映射 ..... | (119) |
| 6.6 小结 .....               | (120) |
| ◇思考题 .....                 | (120) |
| <b>第7章 数据库保护</b> .....     | (121) |
| 7.1 为什么要保护数据库 .....        | (121) |
| 7.1.1 操作错误 .....           | (121) |
| 7.1.2 恶意访问 .....           | (122) |
| 7.1.3 自然灾害 .....           | (122) |
| 7.1.4 数据库故障的严重程度 .....     | (123) |
| 7.2 数据库备份和恢复 .....         | (123) |
| 7.2.1 备份 .....             | (123) |
| 7.2.2 恢复 .....             | (124) |
| 7.2.3 事务 .....             | (124) |
| 7.2.4 日志 .....             | (126) |
| 7.2.5 事务的故障与恢复 .....       | (127) |
| 7.3 完整性 .....              | (129) |
| 7.3.1 数据约束 .....           | (129) |
| 7.3.2 确保完整性的方法 .....       | (130) |
| 7.3.3 SQL 中有关完整性的语句 .....  | (131) |
| 7.3.4 触发器 .....            | (134) |
| 7.3.5 维护数据依赖 .....         | (134) |
| 7.4 并发控制 .....             | (135) |
| 7.4.1 并发的问題 .....          | (135) |
| 7.4.2 锁机制 .....            | (136) |
| 7.4.3 事务串行化与两段封锁 .....     | (138) |
| 7.5 长事务 .....              | (139) |
| 7.5.1 什么是长事务 .....         | (139) |
| 7.5.2 嵌套事务、多级事务、补偿事务 ..... | (140) |
| 7.5.3 并发与串行化 .....         | (141) |
| 7.5.4 多数据库中的事务管理 .....     | (142) |
| 7.6 安全性 .....              | (144) |
| 7.6.1 定义合法用户 .....         | (144) |
| 7.6.2 权力与特权 .....          | (144) |

|            |                        |              |
|------------|------------------------|--------------|
| 7.6.3      | 用户视图 .....             | (145)        |
| 7.6.4      | 环境安全 .....             | (145)        |
| 7.6.5      | 建立私有数据库 .....          | (146)        |
| 7.7        | 小结 .....               | (146)        |
| ◇          | 思考题 .....              | (146)        |
| <b>第8章</b> | <b>数据库系统性能调优 .....</b> | <b>(147)</b> |
| 8.1        | 系统性能调优问题 .....         | (147)        |
| 8.1.1      | 性能瓶颈 .....             | (147)        |
| 8.1.2      | 数据库保护与性能 .....         | (148)        |
| 8.1.3      | 可调优性能的几个方面 .....       | (148)        |
| 8.1.4      | 系统硬件选择 .....           | (149)        |
| 8.1.5      | 体系结构设计 .....           | (150)        |
| 8.2        | 数据库逻辑设计调优 .....        | (151)        |
| 8.2.1      | 根据操作需求调优 .....         | (151)        |
| 8.2.2      | 非规范化数据库设计 .....        | (152)        |
| 8.2.3      | 不改变规范化的表调优 .....       | (153)        |
| 8.2.4      | 数据库约束调优 .....          | (153)        |
| 8.3        | 数据库物理设计调优 .....        | (154)        |
| 8.3.1      | 键与外键 .....             | (154)        |
| 8.3.2      | 索引 .....               | (154)        |
| 8.3.3      | 存储空间与数据存储 .....        | (155)        |
| 8.3.4      | 数据库配置调优 .....          | (156)        |
| 8.4        | 数据库访问调优 .....          | (157)        |
| 8.4.1      | 选择正确的访问策略 .....        | (157)        |
| 8.4.2      | 存储过程 .....             | (158)        |
| 8.4.3      | 物化视图 .....             | (159)        |
| 8.4.4      | 事务调优 .....             | (160)        |
| 8.4.5      | SQL 代码调优 .....         | (161)        |
| 8.4.6      | 应用程序调优 .....           | (161)        |
| 8.5        | 性能基准测试 .....           | (162)        |
| 8.5.1      | 系统事务吞吐量 .....          | (162)        |
| 8.5.2      | 数据库应用类型 .....          | (163)        |
| 8.5.3      | 基准程序 .....             | (163)        |
| 8.5.4      | 性能模拟 .....             | (164)        |
| 8.6        | 小结 .....               | (165)        |
| ◇          | 思考题 .....              | (165)        |
| <b>第9章</b> | <b>遗留系统重建 .....</b>    | <b>(166)</b> |
| 9.1        | 遗留系统重建概述 .....         | (166)        |
| 9.1.1      | 为什么需要遗留系统重建 .....      | (166)        |
| 9.1.2      | 遗留系统重建的工作 .....        | (167)        |

|                                |              |
|--------------------------------|--------------|
| 9.1.3 遗留系统重建的发展 .....          | (168)        |
| 9.2 遗留系统重建技术 .....             | (169)        |
| 9.2.1 遗留系统重建的基本技术 .....        | (169)        |
| 9.2.2 遗留系统重建的分类 .....          | (170)        |
| 9.2.3 遗留系统的重建模型 .....          | (171)        |
| 9.3 数据库重建 .....                | (172)        |
| 9.3.1 数据库重建的必然性 .....          | (173)        |
| 9.3.2 数据库重建需求 .....            | (173)        |
| 9.3.3 数据库重建模型 .....            | (173)        |
| 9.3.4 数据库重建过程 .....            | (175)        |
| 9.4 数据库逆向工程 .....              | (176)        |
| 9.4.1 软件逆向工程 .....             | (176)        |
| 9.4.2 数据库逆向的概念 .....           | (177)        |
| 9.4.3 数据库逆向的步骤 .....           | (178)        |
| 9.4.4 数据库逆向过程中的重要因素 .....      | (179)        |
| 9.5 数据迁移与集成 .....              | (180)        |
| 9.5.1 元数据管理 .....              | (180)        |
| 9.5.2 数据迁移 .....               | (181)        |
| 9.5.3 数据集成 .....               | (181)        |
| 9.6 小结 .....                   | (182)        |
| ◇思考题 .....                     | (182)        |
| <b>第10章 数据利用技术 .....</b>       | <b>(183)</b> |
| 10.1 从信息化到数据资源利用 .....         | (183)        |
| 10.1.1 数据处理技术 .....            | (183)        |
| 10.1.2 DBMS 系统不能支持数据分析需求 ..... | (184)        |
| 10.1.3 数据利用 .....              | (184)        |
| 10.2 数据仓库技术 .....              | (184)        |
| 10.2.1 数据仓库的概念 .....           | (185)        |
| 10.2.2 数据仓库应用系统的体系结构 .....     | (187)        |
| 10.2.3 数据加载 .....              | (188)        |
| 10.2.4 数据仓库主题的设计 .....         | (190)        |
| 10.3 联机分析处理技术 .....            | (191)        |
| 10.3.1 多维数据模型 .....            | (191)        |
| 10.3.2 基本概念 .....              | (193)        |
| 10.3.3 多维数据操作 .....            | (194)        |
| 10.3.4 OLAP 三层体系结构 .....       | (196)        |
| 10.3.5 OLTP 与 OLAP 的关系 .....   | (197)        |
| 10.4 数据挖掘技术 .....              | (199)        |
| 10.4.1 数据挖掘的定义 .....           | (199)        |
| 10.4.2 数据挖掘的发展 .....           | (200)        |
| 10.4.3 数据挖掘过程 .....            | (202)        |

|                            |       |
|----------------------------|-------|
| 10.4.4 数据挖掘任务和内容 .....     | (204) |
| 10.4.5 相关技术的差异 .....       | (206) |
| 10.5 数据准备 .....            | (208) |
| 10.5.1 数据清洗 .....          | (208) |
| 10.5.2 数据抽取、转换和加载 .....    | (209) |
| 10.5.3 数据综合与汇总 .....       | (210) |
| 10.5.4 数据规约 .....          | (210) |
| 10.6 小结 .....              | (210) |
| ◇思考题 .....                 | (211) |
| <b>第11章 数据库系统开发</b> .....  | (212) |
| 11.1 软件工程 .....            | (212) |
| 11.1.1 不成功的案例 .....        | (212) |
| 11.1.2 软件系统生命周期 .....      | (213) |
| 11.1.3 软件开发方法 .....        | (215) |
| 11.1.4 敏捷软件开发 .....        | (218) |
| 11.1.5 数据驱动与对象驱动 .....     | (219) |
| 11.2 以用户为中心的系统开发方法 .....   | (219) |
| 11.2.1 以用户为中心的开发思想 .....   | (220) |
| 11.2.2 用户需求与系统开发需求 .....   | (221) |
| 11.2.3 以用户为中心的需求分析 .....   | (223) |
| 11.2.4 需求收集技术 .....        | (223) |
| 11.2.5 灵活性需求与灵活性设计 .....   | (228) |
| 11.3 客户-服务器式开发方法 .....     | (229) |
| 11.3.1 基本分析模型及需求分析 .....   | (229) |
| 11.3.2 数据分析的多级CS集成技术 ..... | (231) |
| 11.3.3 泛模式下的功能设计 .....     | (232) |
| 11.3.4 数据库设计 .....         | (233) |
| 11.3.5 客户-服务器式开发的优点 .....  | (233) |
| 11.4 项目开发问题 .....          | (233) |
| 11.4.1 项目团队 .....          | (233) |
| 11.4.2 项目管理 .....          | (237) |
| 11.4.3 索赔 .....            | (241) |
| 11.4.4 合作 .....            | (243) |
| 11.5 小结 .....              | (243) |
| ◇思考题 .....                 | (243) |
| <b>参考文献</b> .....          | (244) |

## 第 1 章

# 数据库系统基础知识

数据库系统是对数据进行存储、管理、处理和维护的计算机软件系统，主要包括数据库管理系统（DBMS）、数据库和应用程序等内容。因此，数据库系统开发主要是数据库设计和应用程序的开发，涉及数据建模和系统建模，以及系统实现和运行。本章介绍了数据库系统的基本知识，主要有数据库系统概念、数据库发展、数据库分类、数据建模与数据库系统开发，还介绍了运行数据库的计算机系统。其中，关于数据建模方面的讨论建议读者做一些思考，这对于从宏观上把握数据库系统开发方法及其应用有益。关于基本知识更详细的介绍需要参考“数据库概论”方面的书。

## 1.1 数据库概念

数据库里存放的是数据，“数据”的含义很广，不仅指 321、897 这样一些数字，还指“abc”、“李明”、“96/10/11”等符号、字符、日期形式的数据。确切地说，这里的数据是指能够输入到计算机中的任何东西，如：“数字”、“字符”、“声音”、“图像”、“照片”等，甚至处理数据的计算机程序本身也是计算机的“数据”。当计算机所处理的数据量越来越大时，如何快速准确地找到所要的数据？如何保证数据正确、安全？另外，自从使用了磁存储介质以后，数据作为一种资源和产品有了一个重要的特性，即给别人一个数据，自己并不损失这个数据。因此，数据共享是人们所希望的，如何实现数据共享呢？这些都是数据库系统要解决的问题。

### 1.1.1 从文件系统到数据库系统

计算机处理数据已经历了手工处理、文件系统和数据库系统三个阶段。在手工处理阶段，数据的组织管理都由手工完成，如用一叠卡片存放数据，插入一条记录就要在一张新卡片上打孔，然后将这张卡片插入到这叠卡片中的某处，卡片排列的次序决定了要处理数据的次序。当磁介质（如磁带）用于计算机外存储器后，就不可能以手工方式在磁介质上组织数据（因为看不见摸不着），于是就专门编制了软件（称为文件系统）来组织管理数据。在文件系统阶段，程序所处理的数据一般存放数据文件中（.TXT 或 .DAT 文件中），程序打开数据文件后，才能使用和处理数据，结束后要关闭数据文件。

要注意的是，各种计算机语言或软件，只要它们具有图灵机的能力，那么它们就没有能力上的差别，而只有程度上的差别。这种程度是指人们使用它们的方便程度。如机器语言和高级语言在能力上是一样的，但高级语言使用比较方便。数据库系统和文件系统的根本区别在于数据库系统提供了对数据库中任意部分数据的快速准确的访问。众所周知，当处理的数据量越来越大时，文件系统就显得力不从心。这主要表现在数据不能快速访问，数据冗余量大，数据的一致性维护

困难，等等。那么，如何实现对数据库中任意部分数据的快速准确的访问呢？

首先，需要精心设计存储技术、索引技术和查询优化技术等。这些技术是一般程序开发人员难以掌握的，因此需要有专门技能的人员来实现，形成专门的软件。这是产生数据库系统的必要性。

另一方面，对数据进行的操作通常是简单的，操作的数量是有限的，主要有查询、插入、删除和修改等。这样，开发专门的软件才是可行的。

鉴于上述的必要性和可行性，为了实现对数据库中任意部分数据的快速准确的访问，人们开发了数据库管理系统（DBMS）这一系统软件。DBMS 提供了数据定义语言 DDL 用于定义数据库，提供数据操纵语言 DML 来实现对数据库的任何操作（查询、插入、删除和修改等），用户或应用程序必须通过 DBMS 才能对数据库进行数据处理操作（见图 1-1）。

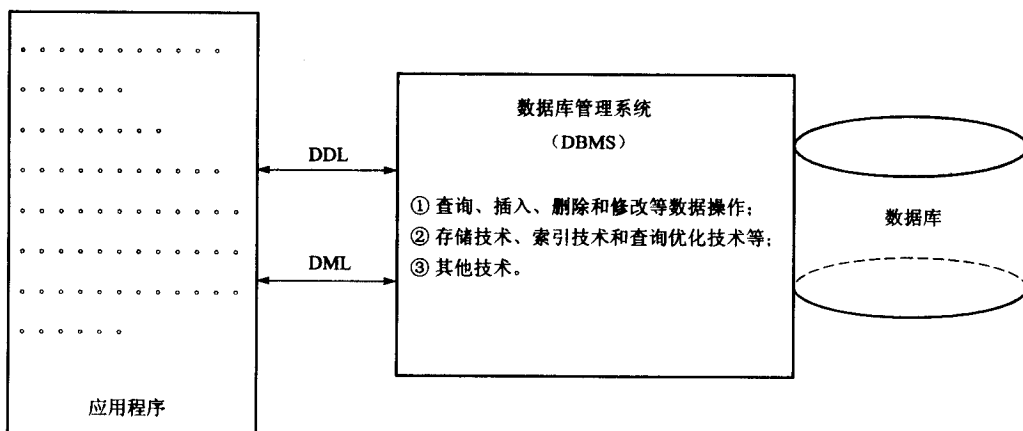


图 1-1 数据库系统阶段的数据处理方式

这样，数据从应用程序中分离出来，统一存入数据库之中，因而数据库又可以被多个应用程序所共享，同时数据的冗余度也就减小了，数据的一致性就比较容易维护。数据共享要求 DBMS 提供数据安全性机制和并发控制机制。另外，还必须保证数据库中的数据是正确的和有效的，这就是数据库的完整性约束。总之，相对于文件系统，数据库系统具有可快速准确地访问数据库中任意部分数据，数据可共享且冗余度较小，数据和程序的独立性较强，统一数据控制（安全性控制、完整性控制和并发控制）等特征。其中，数据库系统提供了对数据库中任意部分数据进行快速准确的访问是数据库系统和文件系统的根本区别。

### 1.1.2 数据库系统

严格地说，数据库系统是对数据进行存储、管理、处理和维护的计算机软件系统。它由计算机软件、数据库和有关人员组成。其软件主要包括数据库管理系统、宿主语言、开发工具和应用程序。其中，数据库管理系统是用于建立、使用、维护数据库的软件系统；宿主语言是可以嵌入数据库 DML 语言的程序设计语言；数据库是长期存储在计算机内的有组织的、大量的、可共享的相关数据的集合，它独立于具体的应用程序，为多个应用程序所共享；有关人员包括负责数据库的设计、修改和日常维护的数据库管理员（DBA）、应用系统开发人员和用户。

由于数据库管理系统只管理数据库，所以应用程序还是要用高级程序设计语言来编写，但不能再由单个高级程序设计语言完成，在要访问数据库时要用 DML 语句实现，这就要求该高级语言能够处理数据库 DML 语言。这种能够嵌入 DML 语言的高级语言称为宿主语言。这种宿主语言是一种通用的高级程序设计语言（如：C，COBOL 等），宿主语言与 DBMS 通常由 DBMS 厂商一

起提供。

DML 语言的命令通过两种途径包含在宿主语言中：

- 宿主语言对 DML 语句的引用通过 DBMS 提供的过程调用实现；
- 将 DML 作为宿主语言的一部分，提供一个预处理器来处理应用程序中的 DML 语句，并将它们转换为 DBMS 提供的过程调用。

显然，第二种形式本质上是用预处理器转换成第一种形式。但在大多数情况下第二种形式使用起来比较方便，这是因为把 DML 作为宿主语言的一部分使得编写程序比较自然。

从应用方面讲，数据库管理系统是为应用提供定义、建造和操纵数据库的功能。定义数据库是指对数据库中数据的类型、结构和约束进行说明；建造数据库是指在存储设备上存放被数据库管理系统控制的数据；操纵数据库是指对数据库的查询、更新和统计等。

数据库系统的工作流程见图 1-2。

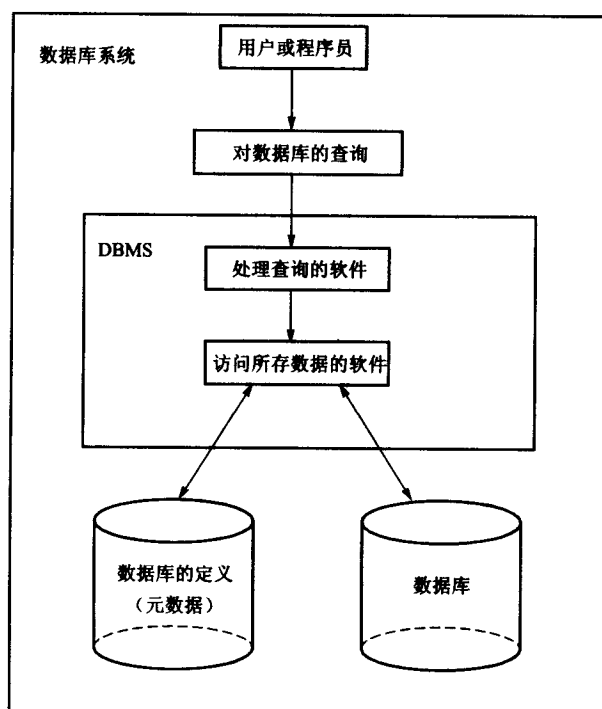


图 1-2 数据库系统的工作流程

### 1.1.3 数据模型

一个数据库一般是某个应用体（Enterprise，指企业、机构或部门等，以下本书都统称为应用体）所涉及数据的综合。由于现实世界中的数据是相互联系和制约的，因此，需要用一种方法来抽象、表示和处理现实世界中的数据及其数据之间的联系。在数据库中是用数据模型来完成这一工作的。

数据模型是数据库系统中用于表示现实世界数据及数据之间联系的形式构架，由数据结构、数据操作和数据约束三要素组成。常见的数据模型有层次数据模型、网状数据模型、关系数据模型、逻辑数据模型和对象数据模型等。每个 DBMS 至少支持一种数据模型，并且到目前为止大多数 DBMS 也只支持一种数据模型，因此又常常根据 DBMS 所支持的数据模型来称呼数据库，如：层次数据库、网状数据库、关系数据库、逻辑数据库（演绎数据库）和对象数据库等。其中，层次数据库和网状数据库被认为是第一代数据库，关系数据库被认为是第二代数据库，而第三代

数据库目前仍然处于研究与开发之中，其重要特征是演绎和面向对象。当前最流行的数据库是关系数据库。

### 1.1.4 数据库系统的体系结构

数据库系统的体系结构是一个三级抽象的结构（见图 1-3）。

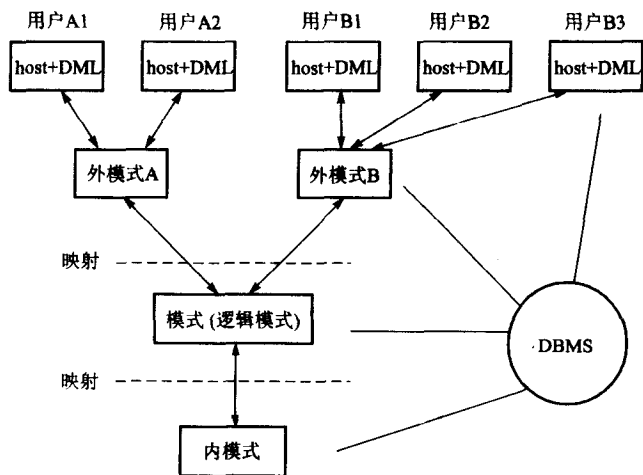


图 1-3 数据库的三级抽象

① 内模式（又称存储模式）：是文件、索引和其他一些存储结构的汇集。它定义了所有内部记录类型、索引、数据在存储介质上的安排等。从这一级看到的数据库称为物理级的数据库。

② 模式（常称逻辑模式）：是对现实世界全局逻辑的抽象。它描述了全部数据的逻辑结构，包括数据之间的联系、数据的约束、安全性要求等。从这一级看到的数据库称为逻辑级的数据库。

③ 外模式（又称子模式）：是由模式导出的局部数据逻辑，是单个用户所要处理的数据集合。应用程序只能接触到外模式。模式与外模式之间的映射由 VIEW 语句实现。

通过三级抽象，数据库就有了物理独立性和逻辑独立性。物理独立性是指：如果内模式改变，则可以通过修改映射，而使逻辑模式以上的内容不变。逻辑独立性是指：如果模式改变，则可以通过修改映射，而使外模式不变，也不用修改程序。

### 1.1.5 与数据库相关的人员

与数据库相关的人员有数据库管理员、数据库设计师、系统分析师、应用程序员和最终用户等。

- 数据库管理员的职责是管理数据库、DBMS 和其他相关的软件等资源，具体有授权对数据库的访问、监控数据库的使用、配置所需的软件和硬件资源等。

- 数据库设计师的职责是设计数据库，具体为分析未来系统的数据需求，建立应用体的概念数据模型（如 ER 模型），找出数据库完整性约束，设计系统安全性计划（包括数据库备份策略），设计逻辑数据库（包括所有的表、索引、视图、触发器和存储过程）和管理数据字典等。在系统交付使用时，数据库设计师应负责为用户培养一个数据库管理员。

- 系统分析师负责未来系统的功能分析和数据分析，其主要工作为：制订需求收集计划和方案、分析用户需求、写出需求分析报告（即系统说明书）。

- 应用程序员负责应用系统的开发（主要是编程）工作。具体有编写触发器、存储过程、数据录入界面和其他程序。