

Visual FoxPro 程序设计教程

主 编 杨七九 丁爱芬 石宜金

副主编 王勇刚 马瑞英 张晓波



高等教育出版社
HIGHER EDUCATION PRESS

高等学校教材

Visual FoxPro 程序设计教程

Visual FoxPro Chengxu Sheji Jiaocheng

主 编 杨七九 丁爱芬 石宜金
副主编 王勇刚 马瑞英 张晓波



高等教育出版社·北京
HIGHER EDUCATION PRESS BEIJING

内容提要

本书是根据教育部高等学校计算机基础课程教学指导委员会制定的《高等学校计算机基础教学发展战略研究报告暨计算机基础课程教学基本要求》编写的。全书既有必要的理论知识,又强调“以应用为出发点”的原则。

本书共分8章,主要内容包括数据库基础、Visual FoxPro 程序设计基础、数据库与表、结构化查询语言、查询与视图、表单设计、菜单设计、报表设计。本书中的操作均有图示及步骤说明。本书配有《Visual FoxPro 程序设计教程实验指导》,帮助学生自主训练,巩固所学知识。

本书可作为应用型本科院校和独立学院 Visual FoxPro 课程的教材,也可作为全国计算机等级考试 Visual FoxPro 的参考书。

图书在版编目(CIP)数据

Visual FoxPro 程序设计教程 / 杨七九, 丁爱芬, 石宜金主编. --北京: 高等教育出版社, 2013.2

ISBN 978-7-04-036864-2

I. ①V… II. ①杨… ②丁… ③石… III. ①关系数据库系统-程序设计-高等学校-教材 IV. ①TP311.138

中国版本图书馆 CIP 数据核字(2013)第 014033 号

策划编辑 耿芳
插图绘制 尹文军

责任编辑 时阳
责任校对 窦丽娜

封面设计 王洋
责任印制 韩刚

版式设计 王艳红

出版发行 高等教育出版社
社 址 北京市西城区德外大街4号
邮政编码 100120
印 刷 高教社(天津)印务有限公司
开 本 787mm × 1092mm 1/16
印 张 14.25
字 数 340千字
购书热线 010-58581118

咨询电话 400-810-0598
网 址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>
网上订购 <http://www.landraco.com>
<http://www.landraco.com.cn>
版 次 2013年2月第1版
印 次 2013年2月第1次印刷
定 价 23.00元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换
版权所有 侵权必究
物 料 号 36864-00

前 言

飞速发展的信息社会离不开数据，大量数据的处理需要使用数据库技术，因此数据库课程已经成为目前大学生必须了解和掌握的一门计算机类基础课程。针对不同院校、不同层次、不同专业的大学生而言，使用与之相适应的教学方法和教材成为教师“教好”、学生“学好”该课程的重要因素。

云南大学旅游文化学院是一所位于云南西北部的独立学院。针对学生特点，在 10 年来的教学与教改研究基础上，本书编者感到，针对独立学院非计算机专业学生而言，Visual FoxPro 程序设计课程采用任务驱动式教学模式对培养应用型人才的目标要求有较好的效果。基于以上认识，云南大学旅游文化学院一批从事该课程教学与教改多年的教师根据自己多年积累的教学经验，编写了针对独立学院非计算机专业学生使用的教材。本书初稿经学校几轮试用，在总结同行专家提出的宝贵意见，以及学生及任课教师从不同角度给予的反馈基础上，编者反复进行了修改补正，使其内容尽可能完善。

本书共 8 章，每章编写遵从以“课程大纲”为主线，准确简洁描述概念，习题案例驱动，尽快进入可视化界面及应用，力求易学实用，循序渐进，实例丰富。基础知识部分每章后均附习题供学生练习之用。书中涉及的案例素材可从中国高校计算机课程网（<http://computer.cncourse.com>）下载。

本书由杨七九、丁爱芬、石宜金任主编，王勇刚、马瑞英、张晓波任副主编。

本书是一本适合应用型本科院校使用的教材。由于编著者水平所限，本书一定会有不足之处。教材的不断完善离不开读者的批评指正，编者真诚期待专家、同行、学生们提出宝贵意见。

编 者

2012 年 10 月 18 日于丽江

目 录

第1章 数据库基础	1	1.8 Visual FoxPro 的辅助设计工具	26
1.1 数据库系统概述	1	1.8.1 Visual FoxPro 6.0 向导	26
1.1.1 数据、数据库、数据库管理系统、 数据库系统	1	1.8.2 Visual FoxPro 6.0 设计器	28
1.1.2 数据管理技术的产生和发展	2	1.8.3 Visual FoxPro 6.0 生成器	28
1.2 数据模型	4	1.9 项目管理器	29
1.2.1 概念模型	4	1.9.1 创建项目	29
1.2.2 逻辑模型和物理模型	7	1.9.2 项目管理器的组成	30
1.3 关系数据库	7	1.9.3 项目管理器的使用	30
1.3.1 关系术语	8	习题	33
1.3.2 关系的基本性质	8	第2章 Visual FoxPro 程序设计基础	38
1.3.3 关系模式	8	2.1 常量与变量	38
1.3.4 关系运算	9	2.1.1 常量	38
1.3.5 关系完整性约束	11	2.1.2 变量	40
1.4 数据库设计	12	2.1.3 数组内存变量的应用	43
1.4.1 需求分析阶段	12	2.2 表达式	45
1.4.2 概念设计阶段	13	2.2.1 数值表达式	45
1.4.3 逻辑设计阶段	13	2.2.2 字符表达式	46
1.4.4 物理设计阶段	13	2.2.3 日期时间表达式	46
1.5 Visual FoxPro 6.0 初步认识	13	2.2.4 关系表达式	46
1.5.1 Visual FoxPro 6.0 的运行环境	14	2.2.5 逻辑表达式	48
1.5.2 Visual FoxPro 6.0 中文版的安装	14	2.2.6 运算符优先级	48
1.5.3 Visual FoxPro 6.0 的启动与退出	18	2.3 结构化程序设计	48
1.5.4 Visual FoxPro 6.0 系统的窗口组成	20	2.3.1 程序文件的建立与运行	49
1.5.5 Visual FoxPro 6.0 的菜单及操作	21	2.3.2 简单的输入输出命令	50
1.6 Visual FoxPro 6.0 的文件类型	23	2.3.3 程序的基本结构	51
1.6.1 常用的文件类型	23	2.3.4 程序的模块化	60
1.6.2 文件调用格式	24	2.3.5 程序调试	63
1.7 Visual FoxPro 6.0 的工作方式和 命令结构	24	2.4 常用函数	64
1.7.1 Visual FoxPro 6.0 的工作方式	24	2.4.1 数值函数	64
1.7.2 Visual FoxPro 6.0 的命令结构	25	2.4.2 字符函数	66
		2.4.3 日期和时间函数	68
		2.4.4 类型转换函数	69

2.4.5 其他常用函数	70	4.3.2 DELETE	113
习题	73	4.3.3 UPDATE	113
第3章 数据库与表	79	4.4 数据查询语言	113
3.1 数据库的建立	79	4.4.1 单表查询	114
3.1.1 数据库设计	79	4.4.2 多表查询	117
3.1.2 建立数据库	81	4.4.3 嵌套查询	120
3.1.3 打开数据库	83	4.4.4 集合的并运算	121
3.1.4 在项目中添加、移去和删除数据库	84	习题	122
3.1.5 修改数据库	85	第5章 查询与视图	131
3.1.6 关闭数据库	86	5.1 查询	131
3.2 数据表的建立	86	5.1.1 基本概念	131
3.2.1 建立表	86	5.1.2 建立查询	132
3.2.2 字段属性	87	5.1.3 运行查询	141
3.3 数据表的操作	89	5.2 视图	142
3.3.1 打开/关闭表	89	5.2.1 基本概念	142
3.3.2 输入表记录	90	5.2.2 建立视图	142
3.3.3 追加记录	91	5.2.3 视图的基本操作	151
3.3.4 复制表	94	习题	152
3.3.5 删除记录/恢复记录	94	第6章 表单设计	154
3.3.6 删除数据表	95	6.1 基本概念	154
3.3.7 记录定位	96	6.1.1 对象	154
3.3.8 修改表名	97	6.1.2 类	155
3.3.9 修改表结构	97	6.2 创建表单	157
3.3.10 数据查询	98	6.2.1 使用表单向导创建表单	157
3.3.11 替换数据	98	6.2.2 使用表单设计器创建表单	164
3.4 数据完整性及表间关系	99	6.2.3 创建快速表单	165
3.4.1 创建索引	99	6.3 表单对象的操作	167
3.4.2 表间关系	100	6.3.1 【属性】窗口	167
3.4.3 表间关系管理	101	6.3.2 控件的基本操作	169
习题	102	6.3.3 表单的运行与关闭	170
第4章 结构化查询语言	106	6.4 为表单添加对象	170
4.1 基本概念	106	6.4.1 设计启动与退出界面	170
4.2 数据定义语言	107	6.4.2 登录界面的设计	174
4.2.1 CREATE TABLE	107	6.4.3 设计添加学生信息界面	177
4.2.2 ALTER TABLE	108	6.4.4 查询界面设计	180
4.2.3 DROP TABLE	111	6.4.5 页框控件显示表信息	182
4.3 数据操纵语言	112	6.4.6 单选/复选控件应用	184
4.3.1 INSERT	112	6.4.7 创建单文档和多文档界面	185

习题	187	第 8 章 报表设计	203
第 7 章 菜单设计	191	8.1 快速创建报表	203
7.1 基本概念	191	8.1.1 使用报表向导创建报表	203
7.1.1 菜单的基本组成	191	8.1.2 创建快速报表	207
7.1.2 菜单系统的规划原则	192	8.2 报表设计器	208
7.1.3 菜单设计步骤	192	8.2.1 报表设计步骤	208
7.2 创建菜单	193	8.2.2 启动报表设计器	209
7.2.1 使用菜单设计器创建菜单	193	8.2.3 设置报表的数据环境	209
7.2.2 创建快捷菜单	199	8.2.4 报表布局设计	210
7.3 菜单系统的应用	200	8.2.5 报表的控件设计	211
7.3.1 创建 SDI 菜单	200	8.3 报表的打印输出	215
7.3.2 将菜单添加到系统菜单	202	习题	216
习题	202	参考文献	217

第1章 数据库基础

数据库是数据管理的技术，是计算机科学的重要分支。随着信息社会的迅速发展，信息资源已成为各个部门的重要财富和资源。因此，建立一个满足各级部门信息处理要求、行之有效的信息系统已成为一个企业或组织生存和发展的重要条件。作为信息系统核心和基础的数据库技术也得到越来越多的应用。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一个国家信息化程度的重要标志。因此，数据库基础知识、数据库应用技术是大学生需要了解或掌握的内容。

本章主要介绍数据库系统的基本概念、数据模型、关系数据库设计等内容。读者从中可以了解到为什么要使用数据库技术、数据库技术的重要性以及关系数据库的基本概念等。

1.1 数据库系统概述

数据库系统体系结构复杂，本节先从数据库常用的术语和基本概念来加以介绍。

1.1.1 数据、数据库、数据库管理系统、数据库系统

数据、数据库、数据库管理系统、数据库系统是与数据库技术密切相关的4个基本概念。

1. 数据

数据(Data)是数据库中存储的基本对象，是用于描述事物的符号记录。数据的种类有很多，数字、文字、图像、声音、学生的档案记录、货物的运输情况等都是数据。数据是一个抽象的概念，简单地说，凡是能被计算机识别和处理的符号记录都称为数据。

2. 数据库

数据库(DataBase, DB)，即存放数据的仓库，是指长期存储在计算机存储设备上，有组织、可共享、结构化的相关数据集合。数据库不仅存放数据本身，同时还存放了数据与数据之间的联系、数据处理必须满足的要求等相关信息。

3. 数据库管理系统

数据库管理系统(DataBase Management System, DBMS)是位于用户和操作系统之间的一层数据库管理软件，是数据库系统的核心。其主要功能如下。

1) 数据定义

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。

2) 数据组织、存储和管理

DBMS 要分类组织、存储和管理各种数据，即要确定以何种文件结构和存取方式来组织这些数据，如何实现数据之间的联系，如何提高存储空间利用率，如何提高数据处理的效率等。

3) 数据操纵

DBMS 提供数据操纵语言 (Data Manipulation Language, DML)，用户可以使用它来操纵数据，实现对数据库的查询、插入、删除和修改等基本操作。

4) 数据库的事务管理和运行管理

数据库在建立、运用和维护时由 DBMS 统一管理、统一控制，以保证数据的安全性、完整性，以及对数据的并发控制和处理数据发生故障后的系统恢复等。

5) 数据库的建立和维护功能

数据库的建立和维护功能包括初始数据的输入、转换，数据转储、恢复，数据库重组，系统性能监测、分析等。

4. 数据库系统

数据库系统 (DataBase System, DBS) 由计算机硬件、数据库、数据库管理系统、操作系统及数据库管理人员 (DataBase Administrator, DBA) 等组成，如图 1.1 所示。需要指出的是，数据库的建立、使用和维护等工作除了由 DBMS 管理之外，还必须有专门的人员 (DBA) 参与。

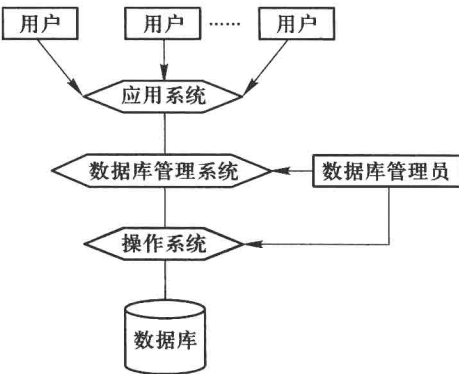


图 1.1 数据库系统

1.1.2 数据管理技术的产生和发展

随着计算机技术的广泛应用，各企事业单位对信息管理的要求也越来越高，因此对数据管理技术的研究也就应运而生了。

数据管理是指对数据进行分类、组织、编码、存储、检索、维护等操作。随着计算机技术的发展，数据管理技术也经历了人工管理、文件系统、数据库系统三个阶段。

1. 人工管理阶段

人工管理阶段中数据与程序不分离，数据集由专门的应用程序来处理，如图 1.2 所示。其特点如下。

1) 数据不保存

由于当时计算机主要用于科学计算，一般不需要将数据长期保存，应用程序只是临时地使用某些数据来计算或验证结果。

2) 应用程序管理数据

数据的表示、存储、处理等操作都由程序员在编写应用程序时完成，没有专门的软件系统来管理，因此程序员负担很重。

3) 数据不共享

应用程序与数据集是一一对应的关系，一个应用程序专门用来处理某一特定的数据集，某

一数据集也只能由某一特定的应用程序来处理。各个应用程序在处理同一数据集时，必须重复定义各自的数据集，因此造成了大量的冗余。

4) 数据不具有独立性

数据的逻辑结构或物理结构发生变化后，处理该数据的应用程序也必须随之修改，加重了程序员的负担。

2: 文件系统阶段

数据以文件的形式独立存储，操作系统有专门的文件系统来统一管理文件的修改、插入和删除等操作，如图 1.3 所示。程序员可以利用文件系统提供的数据存储方法来处理数据，而不必考虑数据处理的细节操作，减轻了程序员的负担。



图 1.2 人工管理阶段程序与数据之间的关系 图 1.3 文件系统阶段程序与数据之间的关系

但文件系统还存在以下缺点。

1) 数据共享性差，冗余度大

应用程序还是与某一文件组一一对应，不同的应用程序处理同一数据文件时，各个应用程序还是要重复存储、各自管理，容易造成数据的不一致性，难于修改和维护。

2) 数据独立性差

尽管数据以文件的形式独立存储，但大部分数据处理任务仍由应用程序完成，当数据的逻辑结构发生变化时，应用程序也必须进行相应的修改，程序员的负担仍然较重。

3. 数据库系统阶段

数据独立存储于数据库中，数据由应用程序通过 DBMS 统一管理和控制，大大减轻了程序员的负担，如图 1.4 所示。其特点如下。

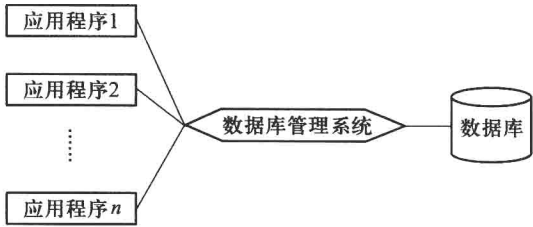


图 1.4 数据库系统阶段程序与数据之间的关系

1) 数据由 DBMS 统一管理和控制

数据定义，数据组织、存储和管理，数据操纵，数据库的事务管理和运行管理，数据库的建立和维护功能等都由 DBMS 统一管理和控制。程序员无须考虑这些细节内容，大大减轻了程

序员的负担，提高了软件开发的效率。

2) 采用特定的数据模型

数据的结构用统一的数据模型描述，无须程序定义和解释。数据模型不仅描述了实体内部的组织结构，而且还描述了不同实体间的联系，同时数据模型还提供了数据完整性约束机制，极大地保证了数据操作过程中的正确性、一致性。

3) 数据的共享性高、冗余度低、易扩充

多用户通过 DBMS 可以同时访问同一数据库中的数据。所有用户操作的数据由 DBMS 来统一管理，极大地实现了数据的共享，降低了冗余度，且易于扩充系统功能。

4) 数据独立性高

应用程序通过 DBMS 访问数据库中的数据，数据独立存放于数据库中，应用程序无须考虑数据的组织形式和具体操作，因此数据具有很高的独立性，不同应用程序对数据的操作都不会影响数据在数据库中的存储组织方式，从而大大提高了应用程序的可扩充性。

1.2 数据模型

数据模型是现实世界数据特征的抽象，是为了将计算机不可能直接处理的现实世界中的具体事物转换成计算机能够处理的数据而采用的抽象、表示和处理的方法，如图 1.5 所示。

数据模型应满足三方面要求：一是能比较真实地模拟现实世界；二是容易为人所理解；三是便于在计算机上实现。

数据模型的组成要素由数据结构、数据操作和完整性约束三部分组成。数据结构描述数据在计算机中的存储方式，数据操作描述在相应的数据结构上可以进行的操作，完整性约束描述在对数据操作的过程中必须满足的要求，以保证数据的正确性、完整性。

根据应用的不同目的，数据模型一般划分为面向用户的概念模型和面向计算机的逻辑模型和物理模型两类。

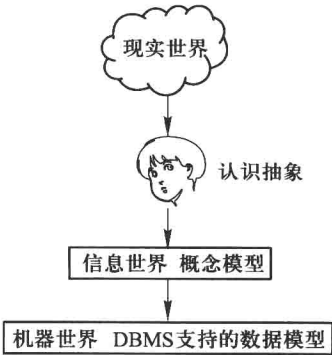


图 1.5 现实世界的抽象过程

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，是数据库设计人员和用户之间进行交流的语言，因此概念模型应能完整有效地抽象描述现实世界的事物，同时又简单、清晰，易于被用户理解。

1. 信息世界中的基本概念

1) 实体

客观存在并且可以相互区别的事物称为实体 (Entity)。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，一个学生、一门课程、一条选课信息等。实体是一个个体的概念。

2) 属性

用来描述实体所具有的某一特征称为属性 (Attribute)。例如，学生由学号、姓名、性别、

民族、出生日期等属性构成。每个属性一般是一个不能再细分的数据项。若给每个属性指定一个具体的值就表示一个具体的实体。例如，(001、张三、男、汉、1999-9-9)就表示一个学生实体的信息。

3) 码

唯一标识实体的属性集称为码 (Key)。例如，学号是学生实体的码。

4) 域

属性的取值范围称为该属性的域 (Domain)。例如，学号的域为 11 位的字符序列，性别的域为男或女。

5) 实体型

具有相同属性的实体具有共同的特征和性质。用实体名及其属性名集合来抽象和描述同类实体，称为实体型 (Entity Type)。例如，学生 (学号，姓名，性别，民族，出生日期) 就是一个实体型。

6) 实体集

同型实体的集合称为实体集 (Entity Set)。例如，全体学生就是一个实体集，即由同类的多个实体所构成的集合。

7) 联系

在现实世界中，联系 (Relationship) 是无处不在的，事物内部以及事物之间都存在相应的联系。在信息世界中，实体内部各属性之间的联系表示了事物内部的联系，不同实体集之间的联系表示了事物与事物之间的联系。

2. 联系的种类

1) 一对一联系 (1:1)

如果对于实体集 A 中的每一个实体，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 具有一对一联系，记为 1:1。

例如，一个班只有一个班长，一个班长只负责一个班，则班长和班级之间就是一对一的联系。

2) 一对多联系 (1:n)

如果对于实体集 A 中的每一个实体，实体集 B 中有 n 个实体 ($n \geq 0$) 与之联系，反之，对于实体集 B 中的每一个实体，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 具有一对多联系，记为 1:n。

例如，一个班级包含若干学生，而一个学生只能在一个班级中学习，则班级和学生之间就是一对多的联系。

3) 多对多联系 (m:n)

如果对于实体集 A 中的每一个实体，实体集 B 中有 n 个实体 ($n \geq 0$) 与之联系，反之，对于实体集 B 中的每一个实体，实体集 A 中也有 m 个实体 ($m \geq 0$) 与之联系，则称实体集 A 与实体集 B 具有多对多联系，记为 m:n。

例如，一个学生可同时选修多门课程，而一门课程可由多个学生选修，则学生和课程之间就是多对多的联系。

无论是单个实体中各属性之间的联系，还是两个或两个以上的实体都存在一对一、一对多

和多对多的联系。

3. 概念模型的表示方法——实体-联系方法（E-R 模型）

表示概念模型的方法很多，其中最常用的是 P. P. S. Chen 于 1976 年提出的实体-联系方法（Entity-Relationship Approach, E-R），该方法用 E-R 图来描述现实世界的模型概念。

E-R 图提供了表示实体型、属性和联系的方法。

1) 实体型

用矩形表示，矩形框内写明实体名。

2) 属性

用椭圆形表示，并用无向边将其与相应的实体连接起来。

3) 联系

用菱形表示，菱形框内写明联系名，并用无向边分别与有关实体连接起来，同时无向边旁标上联系的类型（1:1、1:n 或 m:n）。

假设有如下多个实体型：

班长（班级号，班长号）：班长也是某个班的学生，因此班长号就可以用学号表示

班级（班级号，班级名，辅导员）

学生（学号，姓名，性别，出生日期）

课程（课程号，课程名）

教师（教工号，姓名，性别）

参考书（书号，书名，作者，出版社）

如图 1.6 所示，班级内部辅导员与班级成员之间是一种一对多的内部联系，教师、课程、参考书是 3 个实体间的联系，班长与班级是一对一的联系，班级与学生是一对多的联系，学生与课程是多对多的联系。E-R 模型通过直接的联系间接地反映出了所有实体间存在的关联。

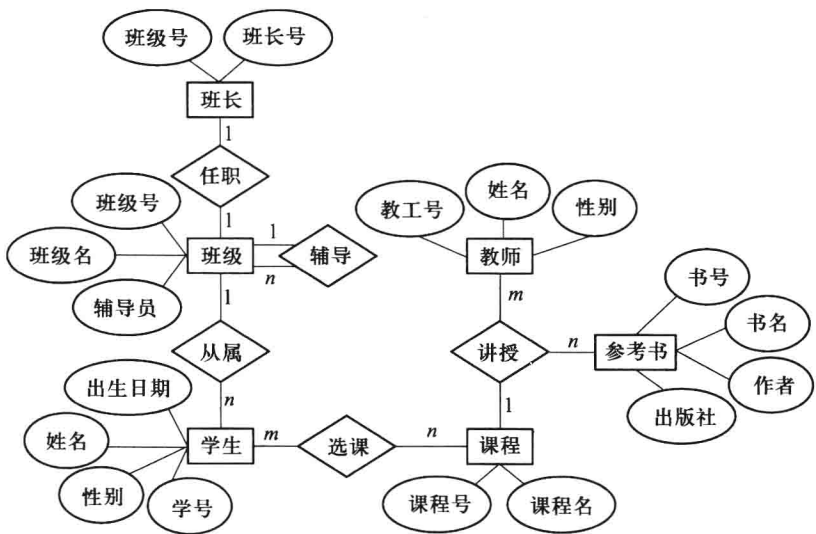


图 1.6 E-R 模型示例

1.2.2 逻辑模型和物理模型

逻辑模型主要包括层次模型、网状模型、关系模型等。它是按计算机系统的观点对数据建模，主要用于 DBMS 的实现。物理模型是对数据最底层的抽象，它描述数据在系统内部的表示方式和存取方法，一般用户不必考虑物理级的细节。

1. 层次模型

层次模型（Hierarchical Model）用树状结构来表示各类实体以及实体间的联系。其特点是：有且只有一个结点没有双亲结点，这个结点称为根结点；根以外的其他结点有且只有一个双亲结点，如图 1.7 所示。

2. 网状模型

网状模型（Network Model）是用网状结构表示数据之间联系的模型。网状模型是层次模型的一般形式。其特点是：允许结点有多于一个的父结点，可以有一个或以上的结点没有父结点，如图 1.8 所示。

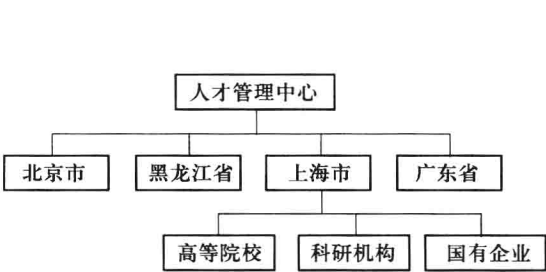


图 1.7 层次模型示例

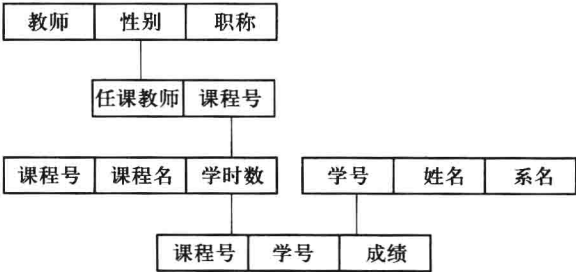


图 1.8 网状模型示例

3. 关系模型

关系模型（Relational Model）是目前最重要的一种数据模型。关系模型建立在严格的数学概念的基础上，用二维表结构来表示实体及实体间的联系。其特点是：表中的分量是不可以再分割的最小数据项，即表中不允许有子表；表中的任意两行不能完全相同；每一列中的分量是类型相同的数据，如图 1.9 所示。

学号	姓名	性别	民族	出生日期	年级
20071200001	张佳吟	女	纳西	02/05/91	2007
20071200002	李是晨	男	汉	02/16/91	2007
20071200003	蒙振	男	汉	02/27/91	2007
20071200004	杨晓龙	男	藏	03/10/91	2007
20071200005	杨烨	女	汉	03/21/91	2007
20071200006	雷文波	男	汉	04/01/91	2007
20071200007	王鹏程	男	汉	04/12/91	2007
20071200008	刘翰芳	女	汉	04/23/91	2007

图 1.9 关系模型示例

1.3 关系数据库

关系数据库系统采用关系模型作为数据的组织方式。从用户观点看，关系模型由一组关系

组成，每个关系的数据结构是一张规范化的二维表，由行和列组成。

1.3.1 关系术语

1. 关系

一个关系 (Relation) 即一张二维表，如图 1.9 所示的学生表就是一个关系。

2. 元组

表中的一行即为一个元组 (Tuple)，也称为一条记录。

3. 属性

表中的一列即为一个属性 (Attribute)，也称为一个字段。给每个属性起一个名称即属性名，如学生表中包含 6 个属性 (学号、姓名、性别、民族、出生日期、年级)。

4. 域

域 (Domain) 是一组具有相同数据类型的值的集合，它表示了属性的取值范围。例如，学生表的性别字段只能是“男”或者“女”，出生日期的月份只能是 1~12 之间的整数，而日期只能是 1~31 之间的整数。

5. 关键字

关系中唯一能区分、确定不同元组的属性或属性组合称为该关系的关键字 (Key)，即作为关键字的属性或属性组对应的元组值不允许出现相同的情况，如学生表中的学号字段。

一个关系中具有关键字特征的字段或字段组合可以有多个，假如学生表中除了学号字段之外，还记录了学生的身份证号，则学号和身份证号都可以作为关键字，但关系数据库中规定，只能选其中一组作为主关键字 (Primary Key)，其他的作为候选关键字 (Candidate Key)。

6. 外部关键字

关系中某个属性或属性组合并非关键字，但却是另一个关系的主关键字，则称此属性或属性组合为本关系的外部关键字 (Foreign Key)。

1.3.2 关系的基本性质

- (1) 列是同质的，即每个属性字段是同一类型的数据，来自同一个域。
- (2) 不同属性字段的域可以相同，但要给予不同的属性名。
- (3) 属性字段在表中的先后次序可以任意交换。
- (4) 任意两个元组的主关键字、候选关键字不能相同。
- (5) 表中各元组的先后顺序可以任意交换。
- (6) 表中各属性字段是不需再分的数据项。

1.3.3 关系模式

关系的描述称为关系模式。它可以形式化地表示为：

$$R(U, D, DOM, F)$$

其中， R 为关系名， U 为组成该关系的属性名集合， D 为属性组 U 中属性所来自的域， DOM 为属性向域的映像集合， F 为属性间的依赖关系集合。

从用户的角度一般将关系模式简记为：

$$R(A_1, A_2, \dots, A_n)$$

其中, R 为关系名, $A_1, A_2, \dots, A_n$ 为属性名, 而域即属性向域的映像常常直接说明为属性的类型、长度等。

关系数据库中, 关系模式是型的概念, 关系是值的概念, 即关系是关系模式在某一时刻的状态或内容。关系模式是静态的、稳定的, 而关系是动态的、随时间不断变化的, 因为关系操作在不断地更新着数据库中的数据。例如, 学生表的各属性确定好之后, 在以后的数据库操作中不再改变, 但属性对应的值会由于增、删、改的操作发生变化。

1.3.4 关系运算

1. 传统的集合运算

传统的集合运算是二目运算, 包括并、交、差、笛卡儿积 4 种运算。设关系 R 和关系 S 具有相同的模式, t 是元组变量, $t \in R$ 表示 t 是 R 的一个元组。则传统的集合运算规则如下所述。

1) 并

关系 R 与关系 S 的并 (Union) 记做:

$$R \cup S = \{t | t \in R \vee t \in S\}$$

其结果得到的关系模式与 R 、 S 相同, 但元组是两个关系中元组集合的并集, 且 R 和 S 中相同的元组在结果中只显示一次。如图 1.10 所示, $T=R \cup S$ 。

2) 交

关系 R 与关系 S 的交 (Intersection) 记做:

$$R \cap S = \{t | t \in R \wedge t \in S\}$$

其结果得到的关系模式与 R 、 S 相同, 但元组是两个关系中元组集合的交集, 且 R 和 S 中相同的元组在结果中只显示一次。如图 1.11 所示, $T=R \cap S$ 。

R			S			T		
A	B	C	A	B	C	A	B	C
1	1	2	3	1	3	1	1	2
2	2	3				2	2	3
						3	1	3

图 1.10 关系并运算

R			S			T		
A	B	C	A	B	C	A	B	C
3	1	3	3	1	3	3	1	3
2	2	3	1	1	4			

图 1.11 关系交运算

3) 差

关系 R 与关系 S 的差 (Except) 记做:

$$R - S = \{t | t \in R \wedge t \notin S\}$$

其结果得到的关系模式与 R 、 S 相同, 但元组由属于关系 R 而不属于关系 S 的元组组成。如图 1.12 所示, $T=R-S$ 。

4) 笛卡儿积

关系 R 与关系 S 的笛卡儿积 (Cartesian Product) 记做:

$$R \times S = \{\widehat{t_r t_s} \mid t_r \in R \wedge t_s \in S\}$$

其结果得到的关系模式中的属性组由 R 和 S 两个关系的属性组成。其元组由关系 R 中的第一条记录分别和关系 S 中的每条记录依次进行连接，然后关系 R 中的第二条记录再次分别和关系 S 中的每条记录依次进行连接，直到关系 R 中的每条记录依次和关系 S 中的每条记录合并完为止。如图 1.13 所示， $T=R \times S$ 。

其中并、交、差运算要求关系 R 和关系 S 的关系模式相同，而笛卡儿积中关系 R 和关系 S 的关系模式可以不同。

R			S			T		
A	B	C	A	B	C	A	B	C
3	1	3	3	1	3	2	2	3
2	2	3	1	1	4			

图 1.12 关系差运算

R		S		T		
A		B	C	A	B	C
m		1	3	m	1	3
n				n	1	3

图 1.13 关系笛卡儿积

2. 专门的关系运算

专门的关系运算包括选择、投影、联接、除运算等。

1) 选择

选择 (Selection) 是从关系中选择满足给定条件的诸元组，是从行的角度进行的运算。其中条件是用比较运算符 $>$ 、 $>=$ 、 $<$ 、 $<=$ 、 $=$ 或 $<>$ 连接常量、变量、函数等构成的逻辑表达式来描述的。关系中的各元组满足该逻辑表达式时才会被选择出来显示在结果关系中。例如，从关系 R 中选择出字段 $B=3$ 的记录得到关系 S ，如图 1.14 所示。

2) 投影

投影 (Projection) 是从关系中选择出若干属性列组成新的关系，是从列的角度进行的运算。投影的属性或属性组中对应的元组值若出现了重复，必须取消重复值，所以投影操作可能会减少元组数。例如，从关系 R 中投影出字段 A 、 B 得到关系 S ，如图 1.15 所示。

R		S	
A	B	A	B
a	3	a	3
b	3	b	3
c	2		

图 1.14 选择运算

R			S	
A	B	C	A	B
a	3	2	a	3
b	0	1	b	0
c	2	1	c	2

图 1.15 投影运算

3) 联接

联接 (Join) 是从两个关系的笛卡儿积中选取属性间满足一定条件的元组。若联接的条件为 $=$ ，则称其为等值联接；若在等值联接的结果中去除重复字段，则称其为自然联接。例如，关系 R 和关系 S 自然联接得到关系 T ，如图 1.16 所示。

4) 除运算

给定关系 $R(X, Y)$ 和 $S(Y, Z)$ ，其中 X 、 Y 、 Z 为属性组。 R 中的 Y 与 S 中的 Y 可以有不同的属