第一篇　理论篇

第1章 嵌入式数据库系统概述

核心要点

- 数据管理技术的发展过程
- 嵌入式数据库基本概念
- 数据模型
- 三级模式和二级映像

学习目标

通过本章学习，应该能够了解数据管理技术发展的历程，掌握数据、数据库的基本概念，掌握数据模型的建立方法，理解数据库系统的体系结构以及SQLite数据库的基本特点和应用。

一、单元概述

本章首先介绍了数据管理技术的发展过程，详细描述了数据库的基本概念，介绍了数据库的概念模型、逻辑模型和物理模型，重点讲解了数据库的概念模型，最后从数据库系统结构的角度介绍了数据的独立性。

数据管理技术是随着企业应用的越来越广泛而逐渐兴起和发展的，至今经历了人工管理、文件系统管理、数据库系统管理三个阶段。作为信息技术的重要组成部分，数据库具有独特的优势与特征：集成性、共享性、独立性及其强大的管理和控制能力。当然，这也与数据库系统的外模式、模式、内模式的三级体系结构是密不可分的。如今，越来越多的企业应用倾向于通过数据库进行数据管理，数据模型也就成了企业关注的重点。尤其是概念模型，不仅是信息世界里的模型，更重要的是计算机世界与现实世界的桥梁。

本章通过理论教学、讨论教学、练习教学，循序渐进地向学生介绍数据库的基本概念和基本原理，并通过练习和讨论的方式加深学生对抽象概念的理解和掌握。

二、教学重点与难点

重点：

(1)理解数据库系统特点。

(2)理解数据库的概念模型。

难点：

理解数据库系统的三级模式和二级映像。

解决方案：

可以通过课堂讨论、综合练习，帮助学生理解。

【知识正文】

随着计算机科学技术的迅速发展，数据库技术已经被广泛应用。数据库是衡量一个国家信息化程度的重要标志，也是企业应用的基础。数据库应用已经渗透到社会领域的方方面面。从 20 世纪 60 年代末期开始到现在，数据库技术已经发展了 50 多年。在这 50 多年的历程中，人们在数据库技术的理论研究和系统开发上都取得了辉煌的成就，而且已经开始对新一代数据库系统的深入研究。数据库系统已经成为现代计算机系统的重要组成部分。

随着微电子技术和存储技术的不断发展，嵌入式系统的内存和各种永久介质的内容均不断增加，这意味着嵌入式系统处理的数据会不断增加，大量的数据如何及时处理，成为嵌入式系统必须面对的现实问题。为了解决这个问题，将原本在企业级运用的复杂数据库处理技术引入到嵌入式系统，应用于嵌入式系统的数据库技术应运而生。

嵌入式数据库系统可以从体系结构方面来定义：嵌入式数据库系统是指支持移动计算或某种特定计算模式的数据库管理系统，它通常与操作系统和具体应用集成在一起，运行在智能型嵌入式设备或移动设备上。在不引起混淆的情况下，通常把数据库系统简称为数据库。

数据库技术是现代信息科学与技术的重要组成部分，是计算机数据处理与信息管理系

统的核心。数据库技术研究和解决了计算机信息处理过程中大量数据有效地组织和存储的问题，在数据库系统中减少数据存储冗余、实现数据共享、保障数据安全以及高效地检索数据和处理数据。

嵌入式数据库技术涉及数据库、分布式计算以及移动通信等多个学科领域，是 20 世纪 90 年代中期开始产生的一个较新的研究领域。最近几年，随着移动设备和通信网络的技术进展以及硬件价格的逐步降低，对适合于移动环境下的应用的数据管理技术提出了迫切的需求，并且这种技术已经成为研究的热点。

本章主要介绍数据库技术的应用与发展、数据库的基本概念、数据模型以及数据库的系统结构等内容，是学习和掌握现代数据库技术的基础。

1.1　数据管理技术的产生与发展

从 20 世纪 50 年代中期开始，计算机应用从科学研究部门扩展到企业管理及政府行政部门，人们对数据处理的要求也越来越高。数据库技术作为数据管理的主要技术目前已广泛应用于各个领域，数据库系统已成为计算机系统的重要组成部分。

计算机对数据的管理是指对数据的组织、分类、编码、存储、检索和维护提供操作手段。

随着计算机硬件、软件技术和计算机应用范围的发展而不断发展，计算机数据管理技术的发展可以大体归为三个阶段：人工管理阶段（50 年代中期以前）；文件系统阶段（50 年代后期到 60 年代中期）；数据库系统阶段（60 年代后期之后）。

1.1.1　人工管理阶段

20 世纪 50 年代以前，计算机主要用于数值计算。从当时的硬件看，外存只有纸带、卡片、磁带，没有直接存取设备；从软件看，没有操作系统以及管理数据的软件；从数据看，数据量小，数据无结构，由用户直接管理，而且数据之间缺乏逻辑组织，数据依赖于特定的应用程序，缺乏独立性。数据处理方式基本是批处理。这个阶段有如下几个特点：

(1)数据不保存在计算机内

由于计算机的软件和硬件的发展水平有限，一般不需要数据长期保存，通常数据随程序一起输入计算机，处理结束后将结果输出，数据空间随着程序空间一起被释放。

(2)只有程序的概念，没有文件的概念

数据的组织方式必须由程序员自行设计与安排。数据需要由应用程序自己管理，没有相应的软件来处理数据。所有的数据库设计包括逻辑结构、物理结构、存取方法及输入方式等，都由应用程序完成。

(3)数据面向程序

数据面向程序，即一组数据对应一个程序。因此，数据不能共享，程序之间互不干扰，数

据的冗余大。

(4)数据不具有独立性

应用程序发生改变,数据的逻辑结构和物理结构就相应发生变化。

人工管理阶段应用程序与数据之间的关系如图 1-1 所示。

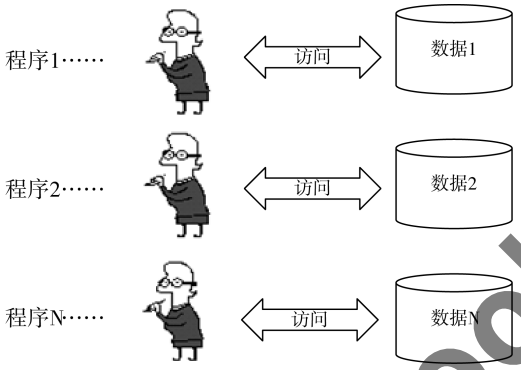


图 1-1 人工管理阶段应用程序与数据之间的对应关系

1.1.2 文件系统阶段

在这一阶段(20 世纪 50 年代后期至 60 年代中期),计算机不仅用于科学计算,还利用在信息管理方面。随着数据量的增加,数据的存储、检索和维护问题成为紧迫的需要,数据结构和数据管理技术迅速发展起来。此时,外部存储器已有磁盘、磁鼓等直接存取的存储设备。软件领域出现了操作系统和高级软件。操作系统中的文件系统是专门管理外存的数据管理软件,文件是操作系统管理的重要资源之一。数据处理方式有批处理,也有联机实时处理。

在数据文件中常涉及下列术语:

数据项:描述事物性质的最小单位。

记录:若干数据项的集合,一个记录表达一个具体事物。

文件:若干记录的集合。

这个阶段有如下几个特点:

(1)数据可长期保存。

(2)简单的数据管理功能。由文件系统进行数据管理,程序和数据之间有了一定的独立性,减少了程序员的工作量。

(3)数据共享性差。在文件系统中,文件仍然面向应用的,当不同文件具有相同数据时,须建立各自的文件,而不能共享这些数据,因此数据的冗余度大,浪费储存空间。

(4)数据的独立性差。文件系统中的文件是面向应用服务的,数据的结构发生改变,必须修改应用程序,修改文件的结构定义;而应用程序的改变也将改变数据的结构,因此文件系统仍然是一个无结构的数据集合。

该阶段应用程序与数据之间的关系如图 1-2 所示。

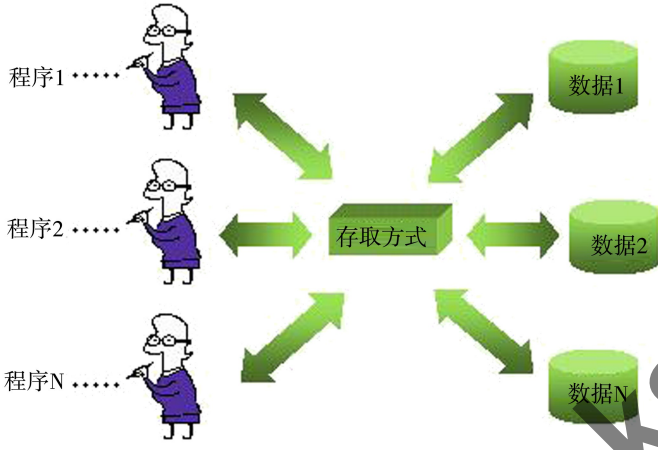


图 1-2　文件系统阶段应用程序与数据之间的对应关系

随着数据管理规模的扩大，数据量急剧增加，文件系统显露出一些缺陷：

(1)数据冗余。

由于文件之间缺乏联系，造成每个应用程序都有对应的文件，有可能同样的数据在多个文件中重复存储。

例如，在人事管理子系统中，需要存储人员基本信息和有关人员管理的相关数据信息；在人员培训子系统中，需要存储人员基本信息和有关人员培训的相关数据信息。尽管两个系统都使用到了人员基本信息，但是在文件系统阶段却需要分别存储，因此存在着大量的数据冗余，如图 1-3 所示。

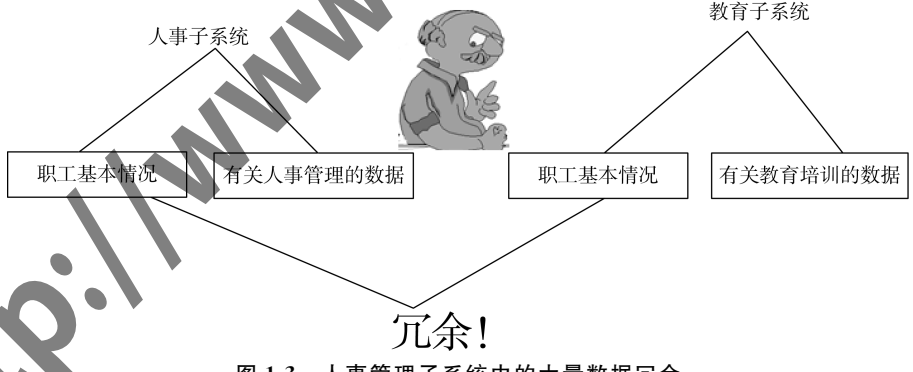


图 1-3　人事管理子系统中的大量数据冗余

(2)不一致性。

这往往是由数据冗余造成的，在进行更新操作时，稍不谨慎，就可能使同样的数据在不同的文件中不一样。正如上面的例子，可能会出现同样的一个员工，在人事管理子系统中出生年月是 1979 年 2 月 12 日，而在人员培训子系统中是 1979 年 8 月 12 日的情况，造成了数据的不一致。

(3)数据联系弱。

这是由于文件之间相互独立，缺乏联系造成的。

文件系统阶段是数据管理技术发展中的一个重要阶段。在这一阶段中，得到充分发展

的数据结构和算法丰富了计算机科学，为数据管理技术的进一步发展打下了基础，现在仍是计算机软件科学的重要基础。

1.1.3 数据库系统阶段

由于文件系统的缺陷，60 年代末期，人们对文件系统进行了扩充，研制了一种结构化的数据组织和处理方式，才出现了真正的数据库系统。数据库为统一管理与共享数据提供了有力支撑，这个时期数据库系统蓬勃发展形成了有名的“数据库时代”。数据库系统建立了数据与数据之间的有机联系，实现了统一、集中、独立地管理数据，使数据的存取独立于使用数据的程序，实现了数据的共享。

(1)数据的集成性

数据库系统中采用统一的数据结构方式，数据的结构化是数据库系统与文件系统的根本区别；数据库系统中的全局的数据结构是多个应用程序共用的，而每个应用程序调用的数据是全局结构的一部分，称为局部结构(即视图)，这种全局与局部的结构模式构成数据库系统数据集成性的主要特征。

(2)数据的高度共享性与低冗余性

数据库系统从整体角度看待和描述数据，数据不再面向某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。尤其是数据库技术与网络技术的结合扩大了数据库系统的应用范围。数据的共享程度可以极大地减少数据的冗余度，节约存储空间，又能避免数据之间的不相容性和不一致性。所谓数据的不一致性是指同一数据在不同的系统拷贝的值不一样。

(3)数据独立性高

数据的独立性是指用户的应用程序与数据库中数据是相互独立的，即当数据的物理结构和逻辑结构发生变化时，不影响应用程序对数据的使用。数据的独立性是由 DBMS 的二级映象的功能来保证的。数据的独立性一般分为两种：一种是物理独立性，另一种是逻辑独立性。

物理独立性是指数据的物理结构(包括存储结构、存取方式等)的改变，如存储设备的更换、物理存储的更换、存取方式改变等都不影响数据库的逻辑结构，从而避免引起应用程序的改变。

逻辑独立性是指数据的总体逻辑结构改变时，如：修改数据模式、改变数据间的联系等，不需要修改相应的应用程序。

(4)数据的管理和控制能力

数据由数据库管理系统进行统一管理和控制，保证了数据的安全性和完整性，如图 1-4 所示。数据库系统对访问数据库的用户进行身份及其操作的合法性检查，保证了数据库中数据的安全性；数据库系统自动检查数据的一致性、相容性，保证数据应符合完整性约束条件；数据库系统提供并发控制手段，能有效控制多个用户程序同时对数据库数据的操作，保证共享及并发操作；数据库系统具有恢复功能，即当数据库遭到破坏时能自动从错误状态恢复到正确状态的功能。

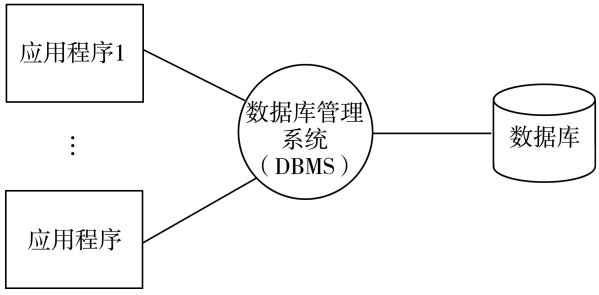


图 1-4　数据库系统阶段应用程序与数据之间的对应关系

综上所述，数据库技术从产生至今仅仅有 40 年的历史，但其发展速度之快，应用范围之广是其他技术可望而不可及的。20 世纪 60 年代末出现第一代数据库——网状数据库、层次数据库，20 世纪 70 年代出现第二代数据库——关系数据库，并成为目前数据库技术的主流，20 世纪 80 年代出现面向对象数据库。数据库技术与网络通信技术、人工智能技术、并行计算技术等互相渗透，成为当前数据库技术发展的主要特征。

1.2　数据库基本概念

首先，我们先介绍一下数据库最常用的术语和基本概念。

1.2.1　数据

我们经常说“今天的天气是－5℃至 2℃，沸水的温度是 100℃，小刚今年 20 岁，购买水果的重量是 2 斤，木头的长度是 1.5 米，大楼的高度 24 层”等。通过这些词，我们的大脑里就形成了对客观世界的印象。

1. 数据的表现形式

说起数据，人们首先想到的是数字。其实，数字是最简单的一组数据。数据的种类很多，在日常生活中数据无处不在。除了数字，文本、图形、图像、音频、视频等，这些都是数据。

数据就是对客观事物特征和特性进行描述的一种抽象性的、符号化的记录。数据可以对事物进行定量或定性的描述，其中对事物进行定量描述常用数字来进行，对事物进行定性描述则常用文字、符号或者图形。当需要对事物的形态进行记录时，还可以使用图象和影象。

例如，在学生证里面填写的学号、姓名、性别、年龄、照片等等，这些记录在上面的文字、数字和图像都是数据。

2. 数据的语义

数据的表现形式并不能完全表达其内容，又是需要经过解释。我们可以把数据的解释称之为数据的语义，数据与其语义是密不可分的。某一具体信息与表示它的数据的这种对应关系因环境而异。同一信息可能有不同的符号表示，同一数据也可能有不同的解释。

例如，85 是一个数据，可以表示一个人的体重是 85kg，也可以表示一个同学数据库课程的成绩是 85 分，还可以表示某高校 85 级学生的基本信息。

数据具有一个使用范围。不同领域的人在描述同一事物会出现不同的数据。例如，中国人会称每个星期的最后一天为“星期天”。美国人会把这一天叫作“Sunday”。基督教徒会称这一天为“礼拜天”。由数据的范围性导致由此建立的信息世界，知识世界在不同的国家、不同的宗教，不同的阶级中会产生差异。

3. 数据和信息

信息就是对客观事物的反映，从本质上看信息是对社会、自然界的事物特征、现象、本质及规律的描述。数据仅涉及事物的表现形式，而信息则涉及这些数据所表示的内容。

信息蕴含于数据之中，信息是数据的内涵；数据是信息的载体，是信息的符号表示。信息必然来源于数据并高于数据，二者不可分离。

例如，上课用的黑板，它的颜色是黑的，形状是矩形，尺寸是长 3.2 米，高 1.4 米，材料是木材，这些都是关于黑板的信息，都是关于黑板的存在状态的反映，从不同角度“反映”或“刻画”了黑板这个事物。再如，看到今天的气温是 15℃～20℃后，就知道今天的天气不错，不冷不热；上午第四节课的时候看时间到了 11 点 48 分，就知道快要下课了，赶紧收拾好书包为奔向食堂做准备。

信息与数据是不同的，尽管人们有时把这两个词互换使用。一般可以认为数据是原料，信息是产品，如图 1-5 所示。数据是信息的符号表示，或称载体，数据不经加工只是一种原始材料，其价值只是在于记录了客观数据的事实。信息是数据的内涵，是数据的语义解释。信息来源于数据，是对数据进行加工处理的产物。其价值在于人类认识世界和改造世界活动的现实意义。也就是说，信息系统把不适合用户使用的数据加工成适合用户使用而形成的信息。同原料与成品概念相似，一个系统的成品可能是另一个系统的原料，那么一个系统的信息也可能成为另一个系统的数据。数据是原材料，而信息对决策或行动是有价值的。

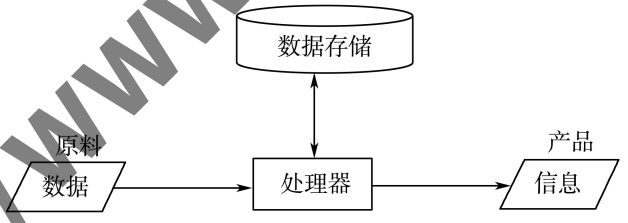


图 1-5　数据到信息的转换

信息有意义，而数据没有。我们知道像 7 度、50 米、300 吨、大楼、桥梁这些数据是没有联系的，孤立的。只有当这些数据用来描述一个客观事物和客观事物的关系，形成有逻辑的数据流，他们才能被称为信息。例如，“94 克”是一个数据，但是什么意思？什么重 94 克？当我们明确“冷库中某一份种子的重量是 94 克”，这就成了信息。

除此之外，信息事实上还包括有一个非常重要的特性：时效性。例如天气预报说大连气温 12℃，这个信息对我们是无意义的，它必须加上今天或明天大连气温 12℃。再例如，通知说在会议室三楼开会，这个信息也是无意义的，必须告诉我们是哪天的几点钟在会议室三楼开会。

1.2.2　数据库

数据库，顾名思义，是存放数据的仓库。只不过这个仓库是在计算机存储设备上，而且

数据是按一定的格式存放的。

我们举个例子说明一下，每个人都有很多亲戚和朋友，为了保持与他们的联系，我们常常用一个笔记本将他们的姓名、地址、电话等信息都记录下来，这样要查谁的电话或地址就很方便了。这个“通讯录”就是一个最简单的“数据库”，每个人的姓名、地址、电话等信息就是这个数据库中的“数据”。我们可以在笔记本这个“数据库”中添加新朋友的个人信息，也可以由于某个朋友的电话变动而修改他的电话号码这个“数据”。不过说到底，我们使用笔记本这个“数据库”还是为了能随时查到某位亲戚或朋友的地址、邮编或电话号码这些“数据”。

其实我们在无意识中，一直在使用数据库。数据库已经和我们的工作、学习、生活紧密相连，密不可分。每当从自己的电子邮件地址簿里查找名字时，就在使用数据库。如果在某个因特网搜索站点上进行搜索，也是在使用数据库。如果在工作中登录网络，也需要依靠数据库验证自己的名字和密码。如果是新生入学报到，学校的教务处、后勤部门等均需要在相应的数据库中添加该学生的信息。如果需要买火车票，则需要访问全国的铁路数据库系统。即使是在 ATM(自动柜员机，Automatic Teller Machine)机上自动取款，也要利用数据库进行 PIN(个人身份号码，Personal Identification Number)码验证和余额检查。

人们收集并抽取出一个应用所需要的大量数据之后，应将其保存起来以供进一步加工处理，进一步抽取有用信息。在科学技术飞速发展的今天，人们的视野越来越广，数据量急剧增加。过去人们把数据存放在文件柜里，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的复杂的数据，以便能方便而充分地利用这些宝贵的信息资源。

严格来讲，数据库是长期储存在计算机内的、有组织的、可共享的数据集合。数据库中的数据按一定的数据模型组织、描述和储存，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。

例如，企业或事业单位的人事部门常常要把本单位职工的基本情况(职工号、姓名、年龄、性别、籍贯、工资、简历等)存放在表中，这张表就可以看成是一个数据库。有了这个“数据仓库”我们就可以根据需要随时查询某职工的基本情况，也可以查询工资在某个范围内的职工人数等等。这些工作如果都能在计算机上自动进行，那我们的人事管理就可以达到极高的水平。此外，在财务管理、仓库管理、生产管理中也需要建立众多的这种“数据库”，使其可以利用计算机实现财务、仓库、生产的自动化管理。

阅读

数据仓库简介

数据仓库(datawarehouse，DW)概念由 William. HInmon 在 1993 年首次提出。他把数据仓库定义为“一个面向主题的、集成的、随时间变化的非易失性数据的集合，用于支持决策层的决策过程”。数据仓库从一诞生就决定了它面向管理和支持决策的特性，它的基本目的就是辅助了解过去，把握未来。大量数据是数据仓库的基础，没有大量的历史和当前数据，数据仓库就是空谈。但数据仓库不是一个 datastorage，不是简单地把各个业务系统的数据装进去。而是需要很好地重新组织，形成一个良好的 datamodel。对于数据仓库的概念我们可以从两个层次予以理解，首先，数据仓库用于支持决策，面向分析型数据处理，它不同于企业现有的操作型数据库；其次，数据仓库是对多个异构的数据源有效集成，集成后按照主题进行了重组，并包含历史数据，而且存放在数据仓库中的数据一般不再修改。

1.3 嵌入式数据库

目前嵌入式软件系统开发的挑战之一，体现在对各种数据的管理能否建立一套可靠、高效、稳定的管理模式，嵌入式数据库可谓应运而生。

嵌入式数据库是嵌入式系统的重要组成部分，也成为对越来越多的个性化应用开发和管理而采用的一种必不可少的有效手段。

嵌入式数据库用途广泛，如用于消费电子产品、移动计算设备、企业实时管理应用、网络存储与管理以及各种专用设备，这一市场目前正处于高速增长之中。举个简单例子，手机原来只用来打电话、发短信，现在手机增加了很多新的功能，比如彩信、音乐、摄影、视频等等，应用的功能多了，系统就变得复杂。

1.3.1 几种主流的嵌入式数据库介绍

1. Berkeley 数据库

Berkeley DB 是美国 Sleepycat Software 公司开发的一个开放源代码的内嵌式数据库管理系统，能够为应用程序提供高性能的数据管理服务。针对 Berkeley DB 的使用，程序员只需要调用一些简单的 API 就可以完成对数据的访问和管理（不需要使用 SQL 语言）。

Berkeley DB 为许多编程语言提供了实用的 API 接口，包括 C、C++、Java、Perl、Tcl、Python 和 PHP 等。所有同数据库相关的操作都由 Berkeley DB 函数库负责统一完成。

Berkeley DB 轻便灵活(Portable)，可以运行于几乎所有的 UNIX 和 Linux 系统及其变种系统、Windows 操作系统以及多种嵌入式实时操作系统之下。Berkeley DB 被链接到应用程序中，终端用户一般根本感觉不到有一个数据库系统存在。

Berkeley DB 是可伸缩(Scalable)的，这一点表现在很多方面。Database library 本身是很精简的(少于 300KB 的文本空间)，但它能够管理规模高达 256TB 的数据库。它支持高并发度，成千上万个用户可同时操纵同一个数据库。Berkeley DB 能以足够小的空间占用量运行于有严格约束的嵌入式系统。

Berkeley DB 在嵌入式应用中比关系数据库和面向对象数据库要好，有以下两点原因：

(1)因为数据库程序库同应用程序在相同的地址空间中运行，所以数据库操作不需要进程间的通讯。在一台机器的不同进程间或在网络中不同机器间进行进程通讯所花费的开销，要远远大于函数调用的开销；

(2)因为 Berkeley DB 对所有操作都使用一组 API 接口，因此不需要对某种查询语言进行解析，也不用生成执行计划，大大提高了运行效率。

2. SQLite 数据库

SQLite 是一个开放源码的嵌入式关系型数据库。最初在 2000 年发布，它被设计用于向应用程序提供一个没有额外负荷的管理数据的方式，被广泛用于专用的关系型数据库管理系统。SQLite 在便携性、易用性、精简性、高效性和可靠性等方面享有很高的声誉。

SQLite 通过利用虚拟机和虚拟数据库引擎(VDBE)，使调试、修改和扩展 SQLite 的内

核变得更加方便。所有 SQL 语句都被编译成易读的、可以在 SQLite 虚拟机中执行的程序集。

SQLite 支持大小高达 2TB 的数据库，每个数据库完全存储在单个磁盘文件中，这些磁盘文件可以在不同字节顺序的计算机之间移动。这些数据以 B＋树(B＋Tree)数据结构的形式存储在磁盘上，SQLite 根据文件系统获得其数据库权限。

在数据类型方面，SQLite 不支持静态数据类型，而是使用列关系。这意味着它的数据类型不具有表列属性，而具有数据本身的属性。当某个值插入数据库时，SQLite 将检查它的类型。如果该类型与关联的列不匹配，则 SQLite 会尝试将该值转换成列类型。如果不能转换，则该值将作为其本身。SQLite 支持 NULL、INTEGER、REAL、TEXT 和 BLOB 数据类型。

3. Empress 数据库

Empress 软件公司在嵌入式数据库领域拥有将近 30 年历史，总部位于加拿大多伦多市。Empress 公司所出品的 Empress 嵌入式实时数据库系列产品属于商业数据库系统，其实时性和稳定性在业界享有很好的声誉。

Empress 不仅可以处理文本数据、货币数据、时间数据等常规数据格式，而且可以处理多媒体数据甚至是应用程序。优秀的表现使 Empress 应用领域不仅包括天气预报、空间探索、飞行模拟及地理信息系统等常规数据库的典型应用领域，而且涵盖了嵌入式实时应用领域，如：电信设备、工业控制、医疗仪器及网络管理等多种领域。美国的军事装备和火星探测等都选用该产品作为嵌入式实时数据库。

Empress 主要特点：

(1)可嵌入程序。该特性使应用程序和数据库工作于统一地址空间，增强了系统的稳定性，提高了系统的效率。

(2)确定的响应时间。Empress 可以使数据的响应时间相对一致，使用者可以设定一个超时限制，如果在规定时间内没有完成插入、修改等操作，系统会报错。

(3)快速的操作。Empress 提供了内核级的 CAPI，称为 MR，用 MR 编写的应用程序在执行时不需要解析。另外在 MR 中加速的机制还包括优秀的加锁控制，内存管理和基于记录数量的选择功能。

(4)灵活的开发方式。Empress 提供多种开发接口，加快开发进程而无需开发者重新学习开发语言和熟悉开发环境。

(5)友好的存储方式。Empress 数据库可以放在操作系统支持的任何存储设备中，Empress 的表单甚至可以分割放在不同的存储设备中，比如在内存、硬盘和 CD-ROM 中。

(6)微型内核结构。Empress 高度单元化，可根据需要选择需要的单元，从而缩小产品中 Empress 数据库所占用的资源。

(7)宽广的平台支持。Empress 支持多种硬件平台和软件平台，也可移植到客户要求的硬件平台或操作系统。

4. eXtreme 数据库

eXtreme 数据库是一种内存嵌入式实时数据库，以其高性能、低开销、稳定可靠的快速实时数据管理能力在嵌入式数据管理领域及服务器实时数据管理领域应用广泛。目前

eXtremeDB 已经广泛应用于众多行业及领域，如：网络设备、消费电子、国防、航空航天、工业控制、轨道交通、能源电力、医疗设备、地理信息、汽车电子、金融实时交易、通信技术、互联网等，得到了国内外许多知名客户的一致好评与青睐。

eXtremeDB 的主要特点：

(1)内存数据库

eXtreme DB 将数据以程序直接使用的格式保存在主内存之中，不仅避免了文件 I/O 的开销，也消除了文件系统数据库所需的缓冲和 Cache 机制。其结果是使每个交易的完成只需要一微秒甚至更少的极限速度，相比于类磁盘数据库而言，速度成百上千倍地提高。作为内存数据库，eXtreme DB 不仅性能高，而且数据存储的效率也非常高。为了提高性能并方便程序使用，数据在 eXtreme DB 中不做任何压缩，100M 的空间可以保存高达 70M 以上的有效数据，这是其他数据库不能比拟的。

(2)混合数据库

eXtreme DB 不仅可以建立完全运行在主内存的内存数据库，更可以建立磁盘/内存混合介质的数据库。在 eXtreme DB，我们把这种建立在磁盘、内存或磁盘＋内存的运行模式称为 eXtreme DB Fusion 融合数据库。eXtreme DB Fusion 兼顾数据管理的实时性与安全性要求，是实时数据管理的巨大进步。

(3)嵌入式数据库

eXtreme DB 内核以链接库的形式包含在应用程序之中，其开销只有 50KB～130KB。无论在嵌入式系统还是在实时系统之中，eXtreme DB 都天然地嵌入在应用程序之中，在最终用户毫不知情的情况下工作。eXtreme DB 的这种天然嵌入性对实时数据管理至关重要：各个进程都直接访问 eXtreme DB 数据库，避免了进程间通信，从而消除了进程间通信的开销和不确定性。同时，eXtreme DB 独特的数据格式方便程序直接使用，消除了数据复制及数据翻译的开销，缩短了应用程序的代码执行路径。

(4)应用定制的 API

应用程序对 eXtreme DB 的操作接口是根据应用数据库设计而自动产生，不仅提升了性能，也消除了通用接口所必不可少的动态内存分配，从而提高了应用系统的可靠性。定制过程简单方便，由高级语言定制 eXtreme DB 数据库中的表格、字段、数据类型、事件触发、访问方法等应用特征，通过 eXtreme DB 预编译器自动产生访问该数据库的 C/C＋＋ API 接口。

(5)可预测的数据管理

eXtreme DB 独特的体系结构，保证了数据管理的可预测性。eXtreme DB 不仅更快、更小，而且更确定。在双核 CPU 的服务器上，eXtreme DB 在 1TB 内存里保存 15B 条记录；无论记录数多少，eXtreme DB 可以在八十分之一微秒的时间内提取一条记录。

5. Solid 数据库

Solid 数据库是一款轻量级的数据库，小巧轻便(安装介质为 30M 左右，只需要 10M 左右的系统资源就可运行)，安装部署维护都非常简单，极大地降低了客户的维护管理成本。不过，Solid DB 同样是标准的关系型数据库，并不会因为轻量级而损失任何功能，支持 SQL / ACID / 事务隔离级别等标准，也支持数据库内部编程，例如存储过程、触发器、事件等。任何有其他关系型数据库使用经验的技术人员都能轻松上手。

Solid 数据库覆盖几乎所有的操作系统平台，可顺畅地运行在嵌入式操作系统、Windows、Linux 、AIX、HP-UX 和 Solaris 等多种环境之中，在全局网络内为用户提供端到端的数据共享平台。

Solid 数据库是全球第一款将内存库和硬盘库功能合二为一的关系型数据库管理平台。Solid 数据库自从 1994 年推出第一个商用正式版本以来，全球范围内已在通信、金融、制造业、公共事业、教育、政府以及医疗行业有着大量的用户，Solid 拥有像 Nokia、Siemens、HP、NEC、Buscom、7-11、ABB、Panasonic、Sony、Agilent、Motorola、Nortel 等 100 多家知名客户，在全球范围内有超过 300 万个 Solid 数据库在运行。

1.3.2　几种主流的嵌入式数据库性能比较

嵌入式技术正在提供越来越多有趣而新颖的服务，在许多的嵌入式系统的开发中，比如电信交换机、消费类电子、办公自动化设备等高科技产品中都会用到各种各样的嵌入式数据库。

编程者在开发时选择何种嵌入式数据库需要综合考虑嵌入式数据库的性价比以及可移植性、可扩展性、支持的接口、支持的 OS 等多种因素。

表 1-1 中对 Berkeley DB、SQLite、EMPRESS、eXtreme DB、Solid 等数据库的综合性能进行了逐个比较，可供用户在选择嵌入式数据库时进行参考。

表 1-1　　几种嵌入式数据库的性能比较

数据库	Berkeley DB	SQLite	EMPRESS	eXtremeDB	Solid
运行时占用 RAM	300KB 左右	小于 250K	200K～800K	内核尺寸 50KB～120KB 并可裁剪，数据库尺寸由用户确定，效率比 1:1.3 左右	450K～2M
使用风险	开源数据库，技术支持不到位，而且商业性应用也不是免费的	开源数据库，Source 完全的 Open，你可以用于任何用途，包括出售它	长时间的开发经验，得到广泛的应用，免维护性高	30 年实时数据库行业经验，eXtremeDB 是该公司当前产品，具有强实时性、高稳定性、超强壮，高可靠性等优点	主要应用于内存数据库当中，不具备嵌入式软件特点，维护性能差
数据库分类	非关系、非对象型数据库	嵌入式关系型数据库	嵌入式关系型数据库	具有关系型、对象型双重特征	内存式关系型数据库
开发工具	任何一种便携式开发工具（支持 Berkeley DB 接口语言）	Gcc、Qt、Eclipse、.net 等	EmpHTML、EmpPerl、EmpTcl/Tk、JumpStart	各种 ANSI 编译器如 gCC、aCC、Tornado2，Workbench、VC＋＋、VS、eVC 等	N/A

（续表）

数据库	Berkeley DB	SQLite	EMPRESS	eXtremeDB	Solid
应用模式	以组件形式内嵌在程序中	1. SQLite 引擎连接到程序中成为它的一个主要部分；2. 整个数据库（定义、表、索引和数据本身）都在宿主主机上存储在一个单一的文件中	1. 以组件形式内嵌在程序中；2. 数据库服务器模式	1、嵌入、融入应用程序；2. 数据库服务器模式	1. 以组件形式内嵌在程序中；2. 数据库独立使用；3. 与大型磁盘数据库配合（Oracle、DB2）使用
支持接口	支持 C、C++、JAVA、TCL 接口，不支持 SQL	比如 C、C++、Tcl、C#、PHP、Java 等，还有 ODBC 接口	支持 Shell、C、C++、JAVA、ODBC、JDBC、SQL、HTML、XML、Perl、Tcl/Tk 接口	C、C++、嵌入式 SQL、Java Native Interface、XML 以及 Rsql/ODBC 等等	支持 ODBC、JDBC、SQL 和 C 语言接口
存储过程、触发器、函数等	不支持	支持	支持	支持	支持
支持的 OS	Linux、QNX、Windows、VxWorks	支持 Windows/Linux/Unix 等等主流的操作系统和嵌入式 linux、Android、iOS 等嵌入式操作系统	Unix、Windows、Linux、Vxworks、Windows CE、OSE、QNX	eXtremeDB 支持各种平台，各种 Windows、Linux、Solaris、HPUX、AIX、VxWorks、eCos、QNX、国产 ReWorks 系统等各种 OS	Unix、Windows、Linux、Vxworks、SymbianOS、Windows CE、OSE、QNX

1.3.3 SQLite 数据库的基本特点和应用

由于资源占用少、性能良好和零管理成本，嵌入式数据库有了用武之地，它为那些基于嵌入式设备的移动应用程序提供了高效的数据存储和管理性能。鉴于 SQLite 的简单、易用以及开源免费等优点，本教材将选择 SQLite 数据库为例来介绍基于嵌入式数据库的各种应用和开发。

1. SQLite 的特点

SQLite 是一个轻量级的关系数据库，具有三级模式的结构体系，即用户模式、逻辑模式和存储模式。相对于传统数据库来说 SQLite 具有更好的实时性、系统开销小、底层控制能力强。SQLite 能够高效地利用嵌入式系统的有限资源，提高数据的存取速度，增强系统的安全性，并具有如下特点：

（1）零配置。SQLite 在使用前不需要安装设置，不需要进程来启动、停止或配置，不需要管理员去创建新数据库或分配用户权限，在系统崩溃或失电之后自动恢复。

（2）对标准 SQL 的支持。SQLite 虽然简单，但其内嵌的 SQL 在很大程度上实现了

ANSI SQL92 标准。特别是 SQLite 支持视图、触发器、支持嵌套 SQL；SQLite 还具有事务处理功能，自动维护事务的完整性、原子性等特性，支持实体完整性和参照完整性，充分满足了嵌入式应用开发的需求。

(3) 无服务器。大多数 SQL 数据库引擎是作为一个单独的服务器进程被执行。访问数据库的程序使用某种内部进程通信(典型的是 TCP/IP) 与服务器通信，完成发送请求到服务器和接收查询结果的工作。SQLite 不采用这种工作方式，使用 SQLite 时，访问数据库的程序直接从磁盘上的数据库文件读写，没有中间的服务器进程。

(4) 精简性。当尺寸优化后，在不减少功能的情况下，整个 SQLite 库小于 225KB。如果在编译时去掉一些不需要的特性，库的大小能被减小到 170KB。

(5) 简单的访问。一个 SQLite 数据库是一个单独的普通磁盘文件，能够被定位在路径层次的任何地方。如果 SQLite 能读写磁盘文件，则它也能访问数据库。大多数 SQL 数据库引擎趋向于把数据存为一个大的文件集合，通常这些文件在一个标准的定位中，只有数据库引擎本身能访问它。

(6) 可变长度的记录。一般的 SQL 数据库引擎在表中为每一个记录分配一个固定的磁盘空间数，SQLite 只使用一个记录中实际存储信息的磁盘空间数。显然，这会使数据库非常小，同时，由于在磁盘上移动的信息很少，也使数据库访问速度很快。

(7) 数据类型。SQLite 最大的特点在于其数据类型为无数据类型(typelessness)。这意味着可以保存任何类型的数据到所想要保存的任何表的任何列中，无论这列声明的数据类型是什么。虽然在生成表结构的时候，要声明每个域的数据类型，但 SQLite 并不做任何检查。开发人员要靠自己的程序来控制输入与读出数据的类型。这里有一个例外，就是当主键为整型值时，如果要插入一个非整型值时会产生异常。

(8) 使用虚拟机。在 SQLite 中使用虚拟机对于库的发展有很大的好处。虚拟机在前端与后端之间准备了一个好的连接。虚拟机对于它所编译的每一个声明使开发者看起来清楚且简单易读，这在调试时有帮助。依赖于它的编译，SQLite 同时具有跟踪虚拟机执行打印指令以及执行结果的能力。

(9) 可靠性较好。SQLite 有良好注释的源代码，并且有着 98%以上的测试覆盖率，有一支专业技术队伍对系统进行测试与维护工作。并且配备独立的命令行程序方便管理数据库。

2. SQLite 的应用

SQLite 不同于其他大部分的 SQL 数据库引擎，因为它的首要设计目标就是简单化，具体体现在以下几个方面：易于管理、易于使用、易于嵌入其他大型程序、易于维护和配置。

许多人喜欢 SQLite 因为它的小巧和快速，但是这些特性只是它的部分优点，使用者还会发现 SQLite 是非常稳定的。出色的稳定性源于它的简单，越简单就越不容易出错。除了上述的简单、小巧和稳定性外，最重要的在于 SQLite 力争做到简单化。

简单化在一个数据库引擎中可以说是一个优点，但也可能是个缺点，主要决定于用户想要做什么。为了达到简单化，SQLite 省略了一些人们认为比较有用的特性，例如高并发性、严格的存取控制、丰富的内置功能、存储过程、复杂的 SQL 语言特性、XML 以及 Java 的扩展、超大的万亿级别的数据测量等等。

如果用户更看重简单的管理、使用和维护数据库，而不是那些企业级数据库提供的不计其数的复杂功能的时候，使用 SQLite 是一个比较明智的选择。目前，从具体的应用来说，SQLite 适用于以下场合：

(1)网站

作为数据库引擎，SQLite 适用于中小规模流量的网站(也就是说，99.9%的网站)。SQLite 可以处理多少网站流量在于网站的数据库有多大的压力。通常来说，如果一个网站的点击率少于 100000 次/天的话，SQLite 是可以正常运行的。100000 次/天是一个保守的估计，不是一个准确的上限。事实证明，即使是 10 倍的上述流量的情况下 SQLite 依然可以正常运行。

(2)嵌入式设备和应用软件

因为 SQLite 数据库几乎不需要管理，因此对于那些无人值守运行或无人工技术支持的设备或服务，SQLite 是一个很好的选择。SQLite 能很好地适用于手机、PDA、机顶盒以及其他智能仪器。作为一个嵌入式数据库它也能够很好地应用于客户端程序。

(3)应用程序文件格式

SQLite 作为桌面应用程序的本地磁盘文件格式取得了巨大成功，例如金融分析工具、CAD 包、档案管理程序等等。一般的数据库打开操作需要调用 SQLite3_open()函数，并且标记一个显式本地事务的起始点(BEGIN TRANSACTION)来保证以独占的方式得到文件的内容。文件保存将执行一个提交(COMMIT)同时标记另一个显式本地事务起始点，这种事务处理的作用就是保证对于应用程序数据文件的更新是原子的、持久的、独立的和一致的。

数据库里可以加入一些临时的触发器，用来把所有的改变记录在一张临时的取消/重做日志表中。当用户按下取消/重做按钮的时候这些改变将可以被回滚。应用这项技术实现一个无限级的取消/重做功能只需要编写很少的代码。

(4)替代某些特别的文件格式

许多程序使用 fopen()、fread()、或 fwrite()函数创建和管理一些自定义的文件用来保存数据，使用 SQLite 替代这些自定义的文件格式将是一种很好的选择。

(5)内部的或临时的数据库

对于那些有大量的数据需要用不同的方式筛选分类的程序，相对于编写同样功能的代码，如果你把数据读入一个内存中的 SQLite 数据库，然后使用连接查询和 ORDER BY 子句按一定的顺序和排列提取需要的数据，通常会更简单和快速。按照上述的方法使用内嵌的 SQLite 数据库将会使程序更富有灵活性，因为添加新的列或索引不用重写任何查询语句。

(6)命令行数据集分析工具

有经验的 SQL 用户可以使用 SQLite 命令行程序去分析各种混杂的数据集。原始数据可以从 CSV(逗号分隔值文件)文件中导入，然后被切分产生无数的综合数据报告。可能的用法包括网站日志分析、运动统计分析、编辑规划标准和分析试验结果。

当然用户也可以用企业级的客户端/服务器数据库来做同样的事情。在这种情况下使

用 SQLite 的好处是：SQLite 的部署更为简单并且结果数据库是一个单独的文件，可以将其存储在软盘或者 U 盘或者直接通过 email 发给同事。

(7)在 Demo 或测试版的时候作为企业级数据库的替代品

如果用户正在编写一个使用企业级数据库引擎的客户端程序，那么使用一个允许连接不同 SQL 数据库引擎的通用型数据库后台将是很有意义的，其更大的意义在于将 SQLite 数据库引擎静态地连接到客户端程序当中，从而内嵌 SQLite 作为混合的数据库支持，这样客户端程序就可以使用 SQLite 数据库文件做独立的测试或者验证。

(8)数据库教学

因为 SQLite 的安装和使用非常简单(安装过程几乎忽略不计，只需要拷贝 SQLite 源代码或 SQLite.exe 可执行文件到目标主机，然后直接运行就可以)，所以它非常适合用来讲解 SQL 语句。初学者可以非常简单地创建他们喜欢的数据库，然后通过电子邮件发给老师批注或打分，对于那些感兴趣怎样实现一个关系型数据库管理系统(RDBMS)的高层次的学生，按照模块化设计且拥有很好的注释和文档的 SQLite 源代码，将为他们打下良好的基础。这并不是说 SQLite 就是如何实现其他数据库引擎的精确模型，但是很适合学生们了解 SQLite 是如何快速工作的，从而掌握其他数据库系统的设计实现原则。

(9)试验 SQL 语言的扩展

SQLite 简单且模块化的设计使得它可以成为一个用来测试数据库语言特性或新想法的优秀的原型平台。

以上是 SQLite 适用的场合，那么哪些场合不适合 SQLite 而适合使用其他的关系型数据库管理系统(RDBMS)呢？简单描述如下：

(1)客户端/服务器程序

如果有许多的客户端程序要通过网络访问一个共享的数据库，应当考虑用一个客户端/服务器数据库来替代 SQLite。SQLite 可以通过网络文件系统工作，但是因为大多数网络文件系统都存在延时，因此执行效率不会很高。此外大多数网络文件系统在实现文件逻辑锁方面都存在着 bug(包括 Unix 和 windows)。如果文件锁没有正常工作，就可能出现在同一时间两个或更多的客户端程序更改同一个数据库的同一部分，从而导致数据库出错。因为这些问题是文件系统执行的时候本质上存在的 bug，因此 SQLite 没有办法避免它们。

好的经验告诉我们，应该避免在许多计算机需要通过一个网络文件系统同时访问同一个数据库的情况下使用 SQLite。

(2)高流量网站

SQLite 通常情况下用作一个网站的后台数据库可以很好地工作，但是如果你的网站的访问量大到你开始考虑采取分布式的数据库部署，那么你应当毫不犹豫地考虑用一个企业级的客户端/服务器数据库来替代 SQLite。

(3)超大的数据集

当你在 SQLite 中开始一个事务处理的时候(事务处理会在任何写操作发生之前产生,而不是必须要显示地调用 BEGIN...COMMIT),数据库引擎将不得不分配一小块脏页(文件缓冲页面)来帮助它自己管理回滚操作。每 1MB 的数据库文件 SQLite 需要 256 字节。对于小型的数据库这些空间不算什么,但是当数据库增长到数十亿字节的时候,缓冲页面的尺寸就会相当大了。因此如果用户需要存储或修改几十 GB 的数据,应该考虑用其他的数据库引擎。

(4)高并发访问

SQLite 对于整个数据库文件进行读取/写入锁定,这意味着如果任何进程读取了数据库中的某一部分,其他所有进程都不能再对该数据库的任何部分进行写入操作。同样,如果任何一个进程在对数据库进行写入操作,其他所有进程都不能再读取该数据库的任何部分。对于大多数情况这不算是什么问题,在这些情况下每个程序使用数据库的时间都很短暂,并且不会独占,这样锁定至多会存在十几毫秒。但是如果有些程序需要高并发,那么这些程序就需要寻找其他的解决方案了。

1.4 数据模型

1.4.1 数据模型的组成

1. 数据模型的作用

模型是对现实世界的抽象。在数据库技术中,表示实体类型及实体类型间联系的模型称为“数据模型”。正如航模、楼盘的沙盘、地图等一样,数据模型也是一种模型,是对现实世界某个对象特征的模拟和抽象。

为什么要建立数据模型(Data Model)呢? 首先正如盖大楼的设计图一样,DM 可使所有的项目参与者都有一个共同的数据标准。其次,数据模型可以避免出现问题再解决(边干边改的方式),第三,数据模型的使用可以及早发现问题,最后,可以加快应用开发速度。

数据模型是连接客观信息世界和数据库系统数据逻辑组织的桥梁,也是数据库设计人员与用户之间进行交流的共同基础。

例如,用大家熟悉的文件系统为例,它所包含的概念有文件、记录和字段。其中,结构和约束条件对每个字段定义数据类型和长度;系统的数据操作包括打开、关闭、读写等文件操作。

上述的是一个简单的数据模型,没有描述数据间的联系。

2. 数据模型的分类

数据模型是用来描述一组数据的概念和定义,包括三个方面:

(1)概念数据模型(Conceptual Data Model)

这是面向数据库用户的实现世界的数据模型,主要用来描述世界的概念化结构,它使数

据库的设计人员在设计的初始阶段，摆脱计算机系统及 DBMS 的具体技术问题，集中精力分析数据以及数据之间的联系等，与具体的 DBMS 无关。概念数据模型必须换成逻辑数据模型，才能在 DBMS 中实现。

(2)逻辑数据模型(Logical Data Model)

这是用户从数据库所看到的数据模型，是具体的 DBMS 所支持的数据模型，如网状数据模型、层次数据模型等等。此模型既要面向用户，又要面向系统。

(3)物理数据模型(Physical Data Model)

这是描述数据在储存介质上的组织结构的数据模型，它不但与具体的 DBMS 有关，而且还与操作系统和硬件有关。每一种逻辑数据模型在实现时都有相对应的物理数据模型。DBMS 为了保证其独立性与可移植性，大部分物理数据模型的实现工作由系统自动完成，而设计者只设计索引、聚集等特殊结构。

3. 数据模型的三要素

一般而言，数据模型是严格定义的一组概念的集合，这些概念精确地描述了系统的静态特征(数据结构)、动态特征(数据操作)和完整性约束条件，这就是数据模型的三要素。

(1)数据结构

数据结构是所研究的对象类型的集合。这些对象是数据库的组成成分，数据结构指对象和对象间联系的表达和实现，是对系统静态特征的描述，包括两个方面：

数据本身：类型、内容、性质。例如关系模型中的域、属性、关系等。

数据之间的联系：数据之间是如何相互关联的，例如关系模型中的主码、外码、联系等。

例如在图书馆管理中，要管理的数据对象有图书、读者、借阅等基本情况。图书对象集中，每本图书包括编号、书名、作者、出版社、出版日期、定价等信息，这些基本信息描述了每本图书的特性，构成在数据库中存储的框架；读者信息包括读者编号、读者姓名、读者性别、读者专业等信息；借阅信息包括读者编号、图书编号、借出日期、应还日期等。借阅信息通过图书编号和读者编号描述了与图书和读者之间的关联。

(2)数据操作

对数据库中对象的实例允许执行的操作集合，主要指检索和更新(插入、删除、修改)两类操作。数据模型必须定义这些操作的确切含义、操作符号、操作规则(如优先级)以及实现操作的语言。数据操作是对系统动态特性的描述。

同样在上面的图书馆管理系统中，对图书的添加、修改、删除和检索均属于数据操作。例如，将新购买的图书添加到数据库中，检索 2009 年以来“数据库原理与应用”的书籍，检索在 2009 年 5 月 1 日之前应该归还的图书等等。

(3)数据完整性约束

数据完整性约束是一组完整性规则的集合，规定数据库状态及状态变化所应满足的条件，以保证数据的正确性、有效性和相容性。

例如，某单位人事数据库中，要求在职的“男”职工的年龄必须大于 18 岁小于 60 岁，工

程师的基本工资不能低于 2000 元，每个职工可担任一个工种，这些要求可以通过建立数据的约束条件来实现。

1.4.2 概念模型

人们把客观存在的事物以数据的形式存储到计算机中，经历了对现实生活中事物特性的认识、概念化到计算机数据库里的具体表示的逐级抽象过程，即现实世界－概念世界－机器世界三个领域。有时也将概念世界称为信息世界；将机器世界称为存储或数据世界。

(1)现实世界

人们管理的对象存于现实世界中。现实世界的事物及事物之间存在着联系，这种联系是客观存在的，是由事物本身的性质决定的。例如学校的教学系统中有教师、学生、课程，教师为学生授课，学生选修课程并取得成绩。

(2)概念世界

概念世界是现实世界在人们头脑中的反映，是对客观事物及其联系的一种抽象描述，从而产生概念模型。概念模型是现实世界到机器世界必然经过的中间层次。

(3)机器世界

存入计算机系统里的数据是将概念世界中的事物数据化的结果。为了准确地反映事物本身及事物之间的各种联系，数据库中的数据必须有一定的结构，这种结构用数据模型来表示。数据模型将概念世界中的实体及实体间的联系进一步抽象成便于计算机处理的方式。如图 1-6 所示。

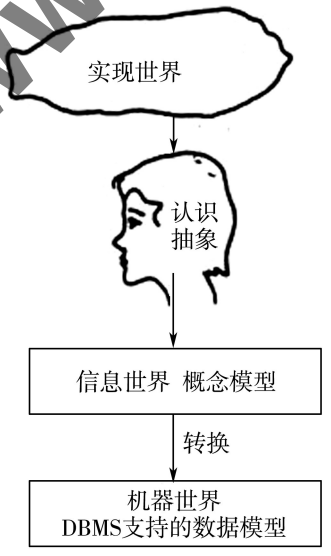


图 1-6 现实世界中客观对象的抽象过程

在三个世界中，对于现实世界中存在的事物的范畴描述是不完全一样的，主要的区别如表 1-2 所示：

表 1-2　　三个世界中的事物描述

现实世界范畴	信息世界范畴	机器世界范畴
所有客观对象	条理化的信息	数据库
实体集	实体记录集	文件
实体	实体记录	记录
特征	属性	字段或数据项
标识特征	标识属性	关键字

概念数据模型也称信息模型，是按用户的观点对数据和信息建模，是现实世界到信息世界的第一次抽象，强调其语义表达功能，易于用户理解，是用户和数据库设计人员交流的语言，主要用于数据库设计。这类模型中最著名的是实体联系模型，简称 E-R 模型。下面我们先来看一下在信息世界中的几个基本概念。

1. 信息世界的基本概念

(1)实体(Entity)

现实世界中存在的可以相互区分的事物或概念称为实体。例如，一个学生、一个部门、一门课程、一种商品、一次采购等都是实体。实体是现实世界中各种事物的抽象，一般来说，每个实体都相当于数据库中的一个表。

(2)实体集(Entity Set)

同一类实体的集合称为实体集。如：全体职工、全体学生、所有商品等。注意区分“型”与“值”的概念。如每个职工是职工实体“型”的一个具体“值”。

一般情况下，我们把实体集简称为实体。

(3)属性(Attribute)

属性是实体所具有的某些特征，通过属性对实体进行刻画。实体由属性组成的。例如，职工由职工号、姓名、性别、出生日期、职称等属性组成。顾客实体可以由商品编号、商品名称、商品价格、商品类别、供应商等属性组成。学生实体可以由学号、姓名、性别、年龄、专业等属性组成。一次采购的实体可以由顾客编号、商品编号、购买日期、购买数量等属性组成。实体的属性值是数据库中存储的主要数据，一个属性实际上相当于表中的一个列。

(4)码(Key)

一个实体本身具有许多属性，能够唯一标识实体的属性或者属性组称为该实体的码(键)。例如，在学生实体中，通过学号可以唯一地标识某一个学生，因此，学号是学生实体的码。然而，在顾客采购商品的采购实体中，需要通过顾客编号、商品编号和购买日期才能够唯一地确定某一次购买，因此，顾客编号、商品编号和购买日期的组合为采购实体的码。

(5)联系(Relationship)

在现实世界中，事物内部以及事物之间是有关系的。实体集之间的对应关系称为联系，它反映现实世界事物之间的相互关联。联系分为两种，一种是实体内部各属性之间的联系。另一种是实体之间的联系。一般情况下，我们在实体联系模型中考虑的是后者。

2. 实体联系模型

实体联系模型(E-R 模型)简称 E-R 图。它是描述概念世界，建立概念模型的实用工具。

E-R 图主要包括三个要素：

(1)实体——用矩形框表示，框内标注实体名称。

(2)属性——用椭圆形表示，并用连线与实体连接起来。

(3)实体之间的联系——用菱形框表示，框内标注联系名称，并用连线将菱形框与有关实体相连，并在连线上注明联系类型。

E-R 图的图例如图 1-7 所示：

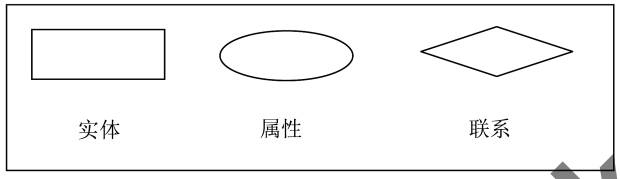


图 1-7　E-R 图例

一般来说，两个实体之间的联系类型归结为以下三种：

(1)一对一联系(1∶1)

设 A、B 为两个实体集。若 A 中的每个实体至多和 B 中的一个实体有联系，反过来，B 中的每个实体至多和 A 中的一个实体有联系，称 A 对 B 或 B 对 A 是 1∶1 联系。注意，1∶1 联系不一定都是一一对应的关系，可能存在着无对应。如一个公司只有一个总经理，一个总经理不能同时在其他公司再兼任总经理，某公司的总经理也可能暂缺。

(2)一对多联系(1∶n)

如果 A 实体集中的每个实体可以和 B 中的几个实体有联系，而 B 中的每个实体至多和 A 中的一个实体有联系，那么 A 对 B 属于 1∶n 联系。如一个部门有多名职工，而一名职工只在一个部门就职，部门与职工属于一对多的联系。

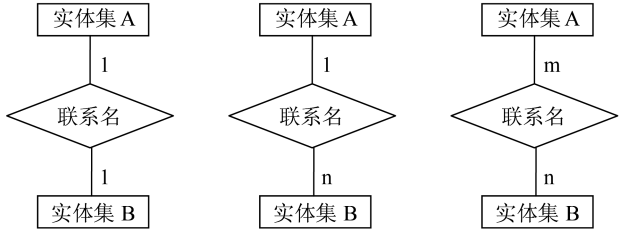
(3)多对多联系(m∶n)

若实体集 A 中的每个实体可与 B 中的多个实体有联系，反过来，B 中的每个实体也可以与 A 中的多个实体有联系，称 A 对 B 或 B 对 A 是 m∶n 联系。如一个学生可以选修多门课程，一门课程由多个学生选修，学生和课程间存在多对多的联系。再如，一个顾客可以购买多种商品，一种商品可以被多个顾客购买，因此，顾客和商品之间也是多对多的关系。

必须强调指出，有时联系也有属性，这类属性不属于任一实体只能属于联系。

我们可以用图形来表示两个实体之间的三类联系，如图 1-8 所示。

实体之间的联系类型并不取决于实体本身，而是取决于现实世界的管理方法，或者说取决于语义，即同样两个实体，如果有不同的语义则可以得到不同的联系类型。



(a) 1:1 联系　　(b) 1:n 联系　　(c) m:n 联系

图 1-8　两个实体集的三类联系

下面以仓库和器件两个实体之间的关联为例来说明这个问题。

如果规定一个仓库只能存放一种器件，并且一种器件只能存放在一个仓库，这时仓库和器件之间的联系是一对一的；

如果规定一个仓库可以存放多种器件，但是一种器件只能存放在一个仓库，这时仓库和器件之间的联系是一对多的；

如果规定一个仓库可以存放多种器件，同时一种器件可以存放在多个仓库，这时仓库和器件之间的联系是多对多的。

(4)两个实体以上的联系

E-R 图不仅能描述两个实体之间的联系，而且还能描述两个以上实体之间的联系。比如有顾客、商品、售货员三个实体，并且有语义：每个顾客可以从多个售货员那里购买商品，并且可以购买多种商品；每个售货员可以向多名顾客销售商品，并且可以销售多种商品；每种商品可由多个售货员销售，并且可以销售给多名顾客。描述顾客、商品和售货员之间的关联关系的 E-R 图如图 1-9 所示，这里联系被命名为“销售”。

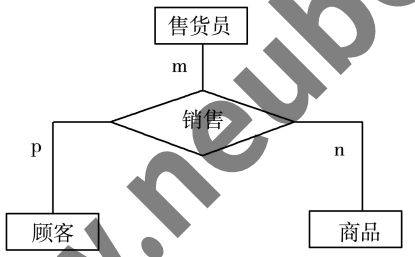


图 1-9　三个实体之间的联系实例

当然，也存在单个实体之间的特殊联系，这里我们就不再赘述。

3. E-R 图的建立步骤

(1)对需求进行分析，从而确定系统中所包含的实体；

(2)分析得出每个实体所具有的属性；

(3)找出每个实体的码；

(4)确定实体之间的联系。

4. 实例

[例 1-1]　某百货公司管辖若干连锁商店，每家商店经营若干商品，每家商店有若干职工，但每个职工只能服务于一家商店。

实体类型“商店”的属性有：店号、店名、店址。

实体类型“商品”的属性有：商品号、品名、单价、产地。

实体类型“职工”的属性有：工号、姓名、性别、工资。

在联系中应反映出职工参加某商店工作的开始时间、商店销售商品的月销售量。

下面给出该百货公司的 E-R 图，如图 1-10 所示。

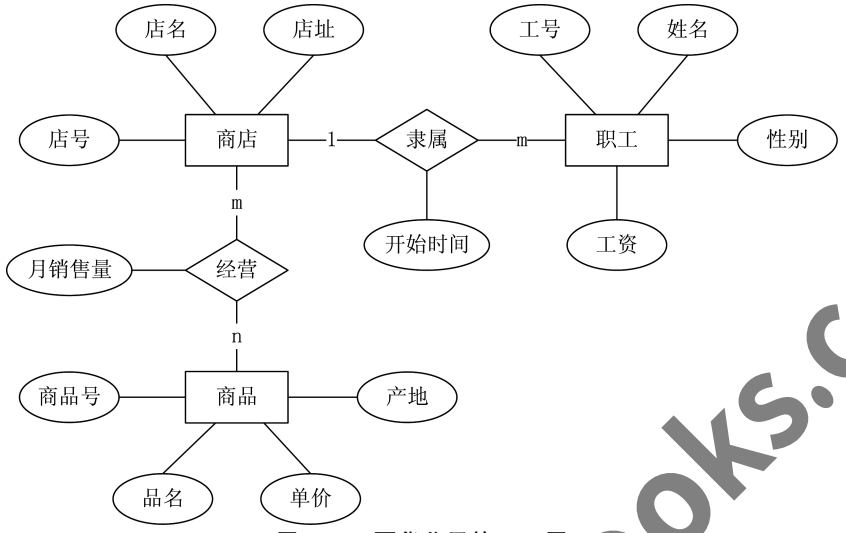


图 1-10　百货公司的 E-R 图

[例 1-2]　某企业拟建立库存——订购数据模型，主要的语义如下：

(1)在一个仓库可以存放多种器件，一种器件也可以存放在多个仓库中；

(2)一个仓库有多个职工，而一个职工只能在一个仓库工作；

(3)一个职工可以保管一个仓库中的多种器件，由于一种器件可以存放在多个仓库中，当然可以由多名职工保管；

(4)一名职工可以经手多张订购单，但一张订购单只能由一名职工经手；

(5)一个供应商可以接受多张订购单，但一张订购单只能发给一个供应商；

(6)一个供应商可以供应多种器件，每种器件也可以由多个供应商供应；

(7)一张订购单可以订购多种器件，对每种器件的订购也可以出现在多张订购单上。

通过分析，我们发现该系统主要有五个实体：

仓库：属性有仓库号、仓库面积、所在城市。

职工：属性有职工号、职工姓名、工资。

器件：属性有器件号、器件名称、单价、描述。

订购单：属性有订购单号、订购日期。

供应商：供应商号、供应商名、地址、联系人、联系电话。

各实体的码均为第一个属性。

实体之间的联系如下：

仓库和器件之间的联系(库存)：m∶n

仓库和职工之间的联系(工作)：1∶n

职工和器件之间的联系(保管)：m∶n

职工和订单之间的联系(发出订单)：1∶n

供应商和订单之间的联系(接收订单)：1∶n

供应商和器件之间的联系(库存)：m∶n

订购单和器件之间的联系(库存)：m∶n

属性图如图 1-11 所示。