数据库高级开发与专业应用系列

Oracle 8i 数据库开发与专业应用

敬 铮 主编

丛治琪 编著

国防工业出版社

·北京·

第 1 章　数据库的基本知识

1.1　数据库技术

1.1.1　了解数据库

数据库技术是研究数据库的结构、存储、设计和使用的一门软件科学，是进行数据管理和处理的技术。

随着计算机应用的普及，数据库应用领域不断向纵横两个方向扩展。现在，企事业、交通运输、情报检索、金融等各行业，纷纷建立以数据库为核心的信息系统。人们在实际应用中向数据库技术提出更新、更高的要求，推动着数据库技术不断向前发展。

数据库在当今信息管理和处理中的重要作用越来越明显。从某种意义上来讲，数据库建设的规模，数据库信息的质量和数量，数据库的使用程度，是衡量一个国家信息化程度的标志。

在过去的许多年里，有许多关于“数据库”这个名词的定义。数据库是一个服务于一个核心目标的数据的有组织的集合。数据库中的数据是有组织的，从某种意义上来说，数据库中存储的数据采用一种不变的方式被存储、格式化、存取以及显示。因为数据库不含有无关的或者冗余的数据，它可以适用于一个核心目标。

一本电话簿就是一个很好的数据库例子，它包含有关的数据（名字），让人们能够查找电话号码；它不包含无关的数据，如某人的电话机的颜色；它只储存那些与它的目标相关的信息。最常见的，一个数据库的目标是商务应用，但是也可能储存科学、军事或者其他数据，这些数据通常不能当作商务数据看待。

因此，有商业数据库、科学数据库、军事数据库以及其他的数据库等等。另外，数据不仅能根据它的应用分类，还能根据它的格式分类，现代数据库包括多种类型的数据。例如，现在数据库储存图像、图表、声音、视频或者包括两种或多种类型的复合文档，已经是很普通的事了。

当讨论数据库特别是数据库设计时，通常称数据库所服务的核心目标为它的业务（business），而不管它属于什么特殊的领域，如太空、生物医学或其他。此外，在实际生活中，你会发现数据通常有它明确的业务应用。

在早些年，编写程序实现自动数据处理（Automatic Data Processing,ADP）要求的程序员发现，在每次运行中，他们需要频繁地存取数据，这就是通常所说的对永久内存的需要，即从程序的一次运行到下一次运行时，需要将数据存留或储存。这个基本的需要成为数据库发展的开端。

其次的需要——简单的数据存储，也促进了数据库的产生。在线归档和历史数据就是

一对特别的例子。尽管文件、目录和文件系统能满足多数普通的数据存储需要（包括索引变化），但数据库可以做文件系统所能做的工作，并且还可做更多的工作。

现代数据库通常为上级组织或者企业的部门，或者它们的小型组织单位实现存储处理需求。因此，用术语“企业范围（enterprisewide）”来代表整个组织的商务范围，用“部门范围（departmentwide）”来代表一个部门级的范围，用“工作组（workgroup）”代表一个部门内的一些单位。通常，在部门范围和工作组级上查找数据库。

偶尔，你会查找服务于企业范围的数据库，例如工资单和人事数据库，但是他们在数量上远远少于那些较小的数据库。实际上，当几个部门的数据库放在一起或合并成为一个大数据库时，这就是建立一个数据库(Data Warehouse，DW)的本质。

那些充当大型数据库的数据源的小型数据库，称为可操作数据库。然而，这不是新东西，一个操作数据库就是产生数据的数据库，多年来一直被叫做成品数据库；只有在建立数据仓库的前提下，你才会发现成品数据库是指可操作数据库或者有时是指可操作的数据存储。随着因特网技术的出现，数据库和数据仓库现在常常作为前端 Web 浏览器的后端使用。

当工作组数据库合并起来以满足一个大型部门需要时，结果通常相当于一个数据中心（Data Mart ,DW），一个 DW 就是一个部门规模的数据仓库。和术语“数据库”那样，术语“数据仓库”也会有多种定义。然而，当你把几个小型数据库集成为一个大型数据库，为一个较广泛的组织服务时，如果该数据库存储历史数据、提供决策支持、提供数据汇总、提供只读数据，并且实质上充当所有向它提供数据的相关成品数据库的数据接收器，那么它通常被看作是一个数据仓库。

另外，如果一个数据库增大的原因是因为该数据库是一个长时期储存数据的历史数据库（例如一个人口普查数据库），或者因为它必须储存数据的类型（例如一个卫星遥测数据库），那么它通常被称为一个非常大的数据库（Very Large Database,VLDB）。

随着时间的发展，由于磁盘容量的增大和价钱的下降、均衡多处理技术机器的出现、冗余廉价磁盘阵列（Redundant Array of Inexpensive Disks,RAID）技术的发展、数据库软件的增多或规模化，判定一个数据库是否为 VLDB 的条件不断变化。当前，一个常用的准则是任何 100GB 或 100GB 以上的数据库就可以被看作是一个 VLDB；而就在几年前，10GB 就被看作是分割点了。

1.1.2 信息、数据、信息处理

归纳起来，信息、数据、信息与数据的联系，可以这样来定义：

1. 信息（Ioformation）

信息是指现实世界事物的存在方式或者运动状态的反应。

信息具有可感知、可存储、可加工、可传递和可再生等自然属性，信息也是社会上各行各业不可缺少的资源，这是它的社会属性。

2. 数据（Data）

数据是指用物理符号记录下来的可以鉴别的信息。

数据在形式上可以表现为数字、文字、图像或其他特殊符号。

3. 信息与数据的关系

数据是信息的符号表示，或称载体；信息是数据的内涵。信息与数据是密切相关的。

因此，在某些不需要严格分辨的场合，可以把两者不加区分地使用，例如信息处理也可以说成数据处理。

例如：　700~1000　　某校每年学生入学人数

　　数据　　信息

4. 信息处理的基本环节

人们将原始信息表示成数据，然后对这些数据进行汇集、存储、综合、推导，从这些原始、杂乱、难以理解的数据中抽去或推导出新的数据，这些结果对某些特定的人们来说是有价值的、有意义的，它表示了新的信息，可以作为决策或推导的依据。这一过程通常称为数据处理或信息处理，其处理活动的基本环节如图 1-1 所示。

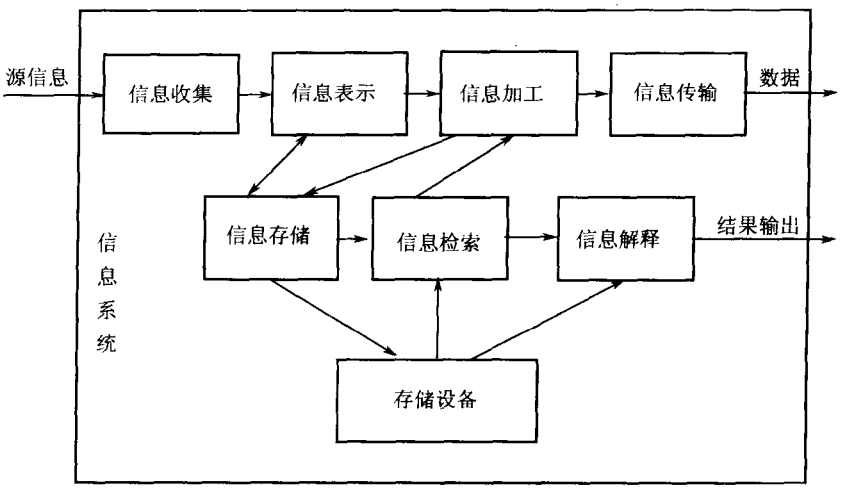

图 1-1　信息处理基本环节

众所周知，信息是有价值的，信息的价值与它的准确性、及时性、完整性和可靠性有关。因为信息的价值必须通过使用信息的决策者的行为结果来实现，所以，为了提高信息的价值，就要用科学的方法来管理信息，这种科学的方法就是数据库技术。

1.1.3　数据库管理技术的发展

数据库管理技术是指对数据进行分类、组织、编码、存储、检索和维护的技术。数据管理技术的发展是和计算机技术及其应用的发展联系在一起的，经历了由低级到高级的发展过程。这一过程大致可分为四个阶段：人工管理阶段；文件系统阶段；数据库系统阶段；高级数据库技术阶段。

1. 人工管理阶段

人工管理阶段是指 20 世纪 50 年代中期以前的阶段。当时计算机处于发展的初期，计算机主要用于科学计算，所用的数据并不多，而且数据的结构一般都比较简单，计算机系统本身的功能还很弱，没有大容量的外存和操作系统，程序的运行由简单的管理程序来控制。这一阶段的特点如图 1-2 所示，可概括为：

(1) 数据不能长期保存在计算机中；

(2) 数据作为程序的组成部分不能独立存在，即数据和程序完全结合成一个不可分割的整体；

(3) 数据由程序员在程序中进行管理，无专门的软件对数据进行管理；

(4) 数据面向应用，不同应用的数据之间相互独立、彼此无关，即便两个不同应用涉及到相同的数据，也必须各自定义，无法互相参照和使用；

(5) 数据大量冗余（Redundancy）。

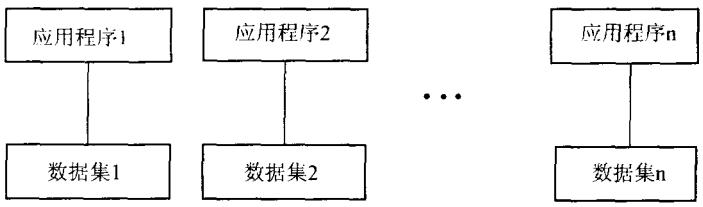

图 1-2 人工管理阶段

2. 文件系统阶段

文件系统阶段是从 20 世纪 50 年代后期到 60 年代中期这一阶段。在这一阶段，由于计算机技术的发展，出现了磁带、磁鼓和磁盘等较大容量的存储设备，软件方面有了操作系统，计算机的应用范围也由科学计算领域扩展到数据处理领域。如图 1-3 所示，这一阶段的特点是：

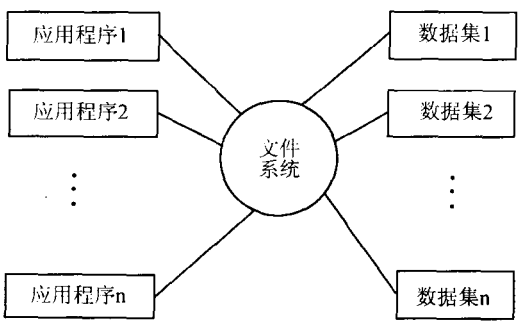

图 1-3 文件系统管理阶段

(1) 数据可以以操作系统的文件形式长期保存在计算机中，文件的组织方式由顺序文件逐步发展到随机文件；

(2) 操作系统的文件管理系统提供了对数据的输入和输出操作接口，进而提供数据存取方法；

(3) 一个应用程序可以使用多个文件，一个文件可以为多个应用程序使用，数据可以共享；

(4) 数据仍然是面向应用的，文件之间彼此孤立，不能反映数据之间的联系，因而仍存在数据的大量冗余和不一致性（Inconsistency）。

3. 数据库系统阶段

数据库系统阶段从 20 世纪 60 年代后期开始。随着计算机硬件和软件技术的发展，开展了对数据组织方法的研究，并开发了对数据进行统一管理和控制的数据库管理系统，在计算机科学领域中逐步形成了数据库技术这一独立分支。数据管理中数据的定义、操作及控制统一由数据库管理系统来完成。图 1-4 说明了这一阶段的特点。

(1) 采用一定的数据模型来组织数据，数据不再面向应用，而是面向系统。

(2) 程序独立于数据，实现了数据的独立性。

(3) 数据的冗余度明显减少，从而减少了数据的不一致性。

(4) 为用户的数据操作提供了方便的用户接口，实现了数据共享。

(5) 提供了下列数据控制功能：

数据的完整性（数据的正确性和一致性）；

数据的安全性（数据的安全和保密，以防被盗窃和失密）；

数据的并发控制（实现数据共享，防止相互干扰和恶意破坏）；

数据库的恢复（发生数据损坏时尽可能恢复到一致状态）。

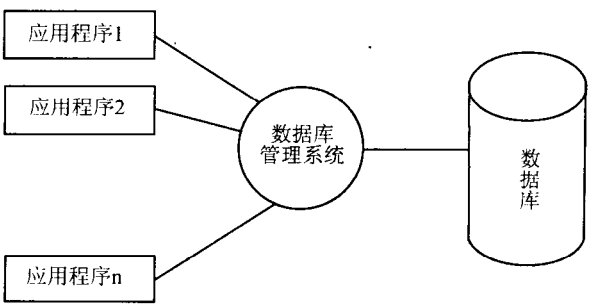

图 1-4　数据库阶段

4. 高级数据库技术阶段

高级数据库技术阶段大约从 20 世纪 70 年代后期开始。在这一阶段中，计算机技术获得更快的发展，并更加广泛地与其他科学技术相互结合和相互渗透，在数据库领域中诞生了很多高新技术，并产生了许多新型数据库，其中有些已经成熟并进入了实用阶段。

(1) 分布式数据库

分布式数据库是数据库技术和计算机网络相互渗透和有机结合的产物。分布式数据库是由一组数据组成，这些数据物理上分布在计算机网络的不同结点上（结点也称为场地），既能完成本地的局部应用，又参与设计多个场地的全局应用。即这些分布的数据逻辑上属于同一个整体。

分布式数据库的这个定义强调了数据与处理的分布性、各场地的自治性和数据的逻辑整体性。分布性是指数据不是存储在一台计算机的存储设备中，从而和集中式数据库相区别。

自治性是指各个场地相互独立，完成本地应用，并无主次之分。

逻辑性是指在逻辑上与集中式数据相同，数据是一个整体，而不是分散在计算机网络不同结点上的各自逻辑独立的数据库（或文件系统），如图 1-5 所示。

分布式数据库的重要特性是数据分布的透明性，分布式数据库是一个统一整体。用户不必关心数据的逻辑分布，更不必关心数据的物理分布的细节。

对用户来说，访问分布式数据库如同访问集中式数据库一样，只要指出访问哪些数据，而不需要指出到哪里或如何访问这些数据。

(2) 面向对象数据库

20 世纪 60 年代末期，在程序设计语言中引入了面向对象的概念。通过面向对象的程

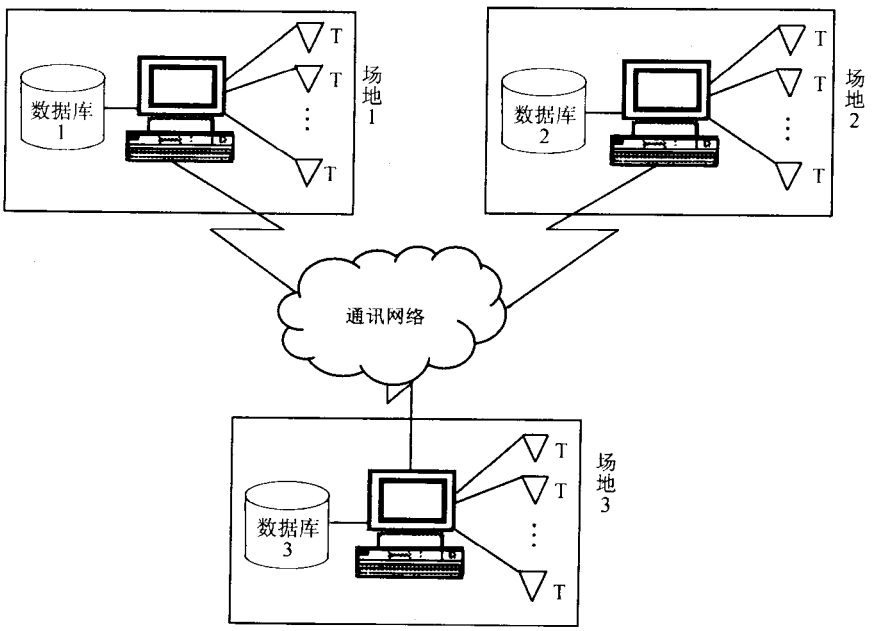

图 1-5 分布式数据库系统示意图

序设计来解决程序的重用问题。将面向对象的概念引入数据库领域，产生了面向对象数据库系统。

面向对象技术最重要的进展是，数据和数据操作的方法作为对象由面向对象的数据库管理系统来统一管理，任何被开发的应用都成为对象目标库的一部分，由开发者和用户所共享。

共享缩小了数据库和应用程序的差距，降低了应用程序的开发费用，同时也减少了系统出现问题的可能性。

同时，面向对象技术中所用的方法能精确处理现实世界中复杂的目标对象。例如，图像、声音、文本文件等，都可以被定义为抽象数据类型，而且在系统运行时对它们的内容进行检查。

在面向对象技术中，属性的继承性使得可以在对象中共享数据和操作，对象之间的通信成为数据和程序间交换信息的标准。面向对象的数据库技术已经可以处理复杂的企业范围内变化的事物对象。

1.2 数 据 模 型

模型是抽象地模仿现实世界的事物，在数据库技术中，使用数据模型（Data Model）的概念描述数据库的结构和语义。根据应用的不同，数据模型可分为两类或两个层次：

(1) 概念数据模型：只描述信息特性和强调语义，而不涉及信息在计算机中的表示，是现实世界到信息世界的第一层抽象。最常用的是实体联系模型（Etinity Relation Model）。

(2) 数据结构模型：直接描述数据库中数据的逻辑结构，这类模型涉及到计算机系统，

又称为基本数据模型。它是用于机器世界的第二层抽象，通常包括一组严格定义的形式化语言，用来定义和操纵数据库中的数据。最常用的有：

层次模型(Hierarchical Model)；

网状模型(Network Model)；

关系模型(Relational Model)；

面向对象模型(Object-Oriented Model)。

1. 实体联系模型（Etinity Relation Model）

实体联系模型（简记为 E-R 模型）是 P.P.Chen 于 1971 年提出的。E-R 模型中的基本语义单位是实体与联系，它可以形象地用图形表示，称为 E-R 图。

E-R 图是直观表示概念模型的有力工具。在 E-R 图中，以矩形框表示实体类型（考虑问题的对象），用菱形框表示联系类型（实体间的联系），用椭圆形框表示实体类型和联系类型的属性，相应的名字均记入框中。联系类型与其涉及的实体类型之间以直线连接，并在直线端部标注联系的种类，如：

1∶1 表示 1 对 1 的联系；

n∶n 表示 1 对多的联系；

m∶m 表示多对多的联系。

下面通过例子来说明 E-R 模型。

例 1-1：为仓库管理设计一个 E-R 模型。仓库主要管理零件的进库、出库、采购等事项。仓库根据需要向外面厂家购买零件，而许多工程项目需要仓库供应零件。这个 E-R 模型如图 1-6 所示。

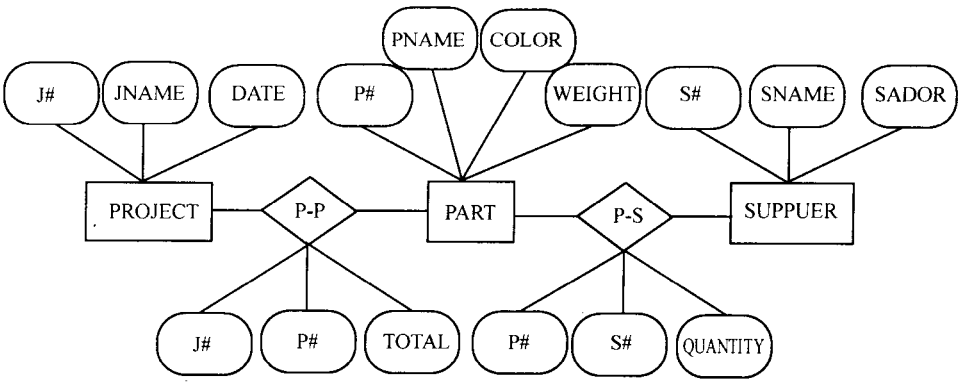

图 1-6　E-R 图示例

联系类型也可以发生在多于两个实体类型之间。如上例中，如果规定某个工程项目是指定需要某个厂家的零件，那么 E-R 图如图 1-7 所示。

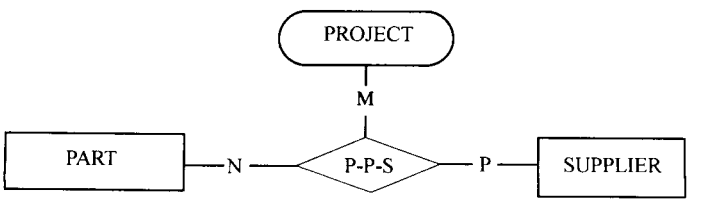

图 1-7　三个实体类型之间的联系

需要注意的是，在 E-R 图中，联系类型的属性中可以不包括与它相关的实体的键，当将 E-R 图转化为逻辑模型时再将它给出。

E-R 模型有两个明显的优点：

(1) 接近人的思想，容易理解；

(2) 与计算机无关，用户容易接受，因此 E-R 模型已成为数据库概念设计的一个重要的设计方法。

2. 网状模型(Network Model)

用网状结构表示实体类型及实体之间联系的数据模型称为网状模型。网状模型是美国 DASYL 委员会数据库任务组（DBTG）于 1969 年提出的一种模型，用于设计网状数据库。

在网状模型中，一个子结点可以有多个父结点，在两个结点之间可以有一种或多种联系，如图 1-8 所示。

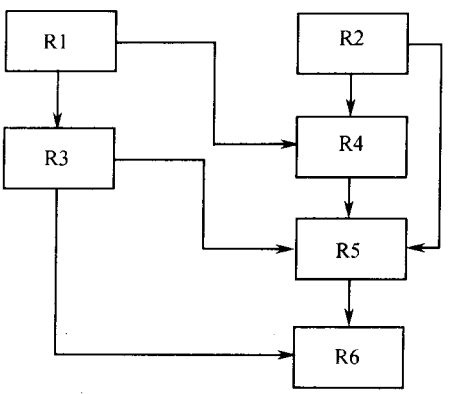

图 1-8 网状模型示例

网状模型有许多成功的数据库管理系统，它们在 20 世纪 70 年代和 80 年代得到了广泛的应用。

网状模型实现实体间 m:n 联系比较容易。记录之间联系是通过指针实现的，因此数据的联系十分密切。网状模型的数据结构在物理上也易于实现，效率较高，但是应用程序的编写较复杂，程序员必须熟悉数据库的逻辑结构。

在网状模型中，任意两个记录类型之间都可以组成一个系类型结构。因此以记录类型为结点的结构图是网络结构。

从 E-R 图到网络模型的转换规则主要有两条：

(1) 把 E-R 图的实体类型和联系类型全部转换成相应的记录类型；

(2) 把 E-R 图中实体类型与有联系的每个联系类型，分别组成一个系类别。

3. 层次模型(Hierarchical Model)

用树型（层次）结构表示实体类型以及实体间的联系是层次模型的主要特征，如图 1-9 所示。

层次结构是一棵有向树，树的结点是记录类型，根结点只有一个，根结点以外的结点有且只有一个父结点。上一层记录类型和下一层记录类型间的联系是 1:n 联系（包括 1:1 联系）。

1969 年，IBM 公司推出 IMS 系统，它是最典型的层次模型系统，曾在 20 世纪 70 年代商业中广泛应用。

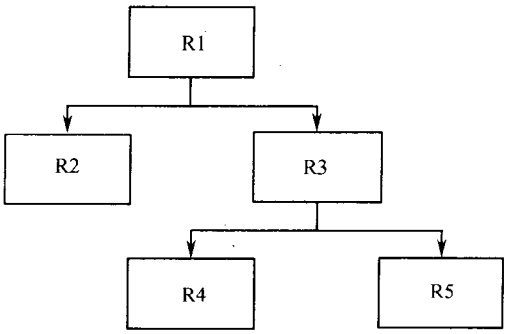

图 1-9　层次模型示例

4. 关系模型(Relational Model)

用二维表格形式结构表示实体类型以及实体间联系的模型称为关系模型。关系模型比较简单，容易被初学者接受。关系在用户看来是一个二维表格，记录是表中的行，属性是表中的列。

例 1-1 的 E-R 图可以转换成如图 1-10 所示的关系模型。

PART (J#,PNAME,COLOR,WEIGHT)
PBOJECT(P#,JNAME,DATE)
SUPPLIE-R(S#,SNAME,SADOR)
P-P(J#,P#,TOTAL)
P-S(P#,S#,QUANTITY)

SUPPLIE-R关系

S#	SNAME	SADOR
S1	PICC	SHANGHAI
S2	PADC	BEIJING

PROJECT关系

J#	JNAME	DATE
J1	JA	95.1
J2	JB	96.5
J3	JC	96.7

PART 关系

P#	PNAME	COLOR	WEIGHT
P1	SOREW	BLUE	14
P2	BOLT	GREEN	17
P3	NUT	RED	12
P4	SCREN	RED	19

P-P 关系

J#	P#	TOTAL
J1	P3	18
J2	P3	25
J3	P3	5
J4	P2	20

P-S 关系

P#	S#	QUANTITY
P3	S1	100
P3	S2	200
P2	S3	500

图 1-10　关系模型示例

关系模型和网状、层次模型的最大区别是：

(1) 关系模型用表格数据而不是通过指针来表示和实体间联系。

(2) 关系模型的数据结构简单、易懂，只需要简单的查询语句就可对数据库进行操作。

关系模型是数学化的模型，可把表格看成一个集合，因此集合论等知识可引入到关系模型中来。关系模型已是一个成熟的有前途的模型，已得到广泛的应用。

5. 面向对象模型（Object-Oriented Model）

由于关系模型比网状、层次模型更为简单灵活，因此在数据处理领域中，关系数据库的使用已相当普遍。但是，现实世界存在着许多含有更复杂数据结构的实际应用领域，例如 CAD 数据、图形数据等，需要有一种数据模型来表达这类信息。随后，在人工智能研究中也出现了类似的需要，这种数据模型就是面向对象数据模型。

面向对象数据模型的新概念有以下几点：

(1) 对象和对象标识（OID）：对象是现实世界中实体的模型化，与记录、属性的概念相似，但远比它们复杂。每一个对象都有一个唯一的标识，称为对象标识（OID）。对象标识 OID 不等于关系模式中的记录标识 RID 或属性标识 TID。

OID 是独立的，全系统唯一。

(2) 封装（Encapsulate）：每一个对象是状态(State)和行为(Behavior)的封装。

对象的状态是该对象属性的集合。对象的行为是在该对象状态上操作的方法（程序代码）的集合。被封装的状态和行为在对象外部是看不见的，只能通过现实定义的消息传递来访问。如图 1-11 所示。

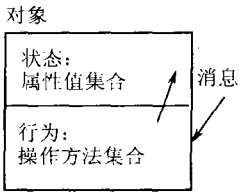

图 1-11 对象示意图

(3) 对象的属性（Object Attribute）：对象的某个属性可以是单值或值的集合。对象的一个属性值本身在该属性看来也是一个对象。

(4) 类和类层次

类：所有具有相同属性和方法集的对象构成了一个对象类。任何一个对象都是某个对象类的一个实例（Instance）。对象类中属性的定义域可以是任何类，包括：

基本类：整型，实型，字串等；

一般类：包含自身属性和方法类本身。

例如，“所有文件”构成文件类，文件类中的每个文件是对象，如图 1-12 所示。

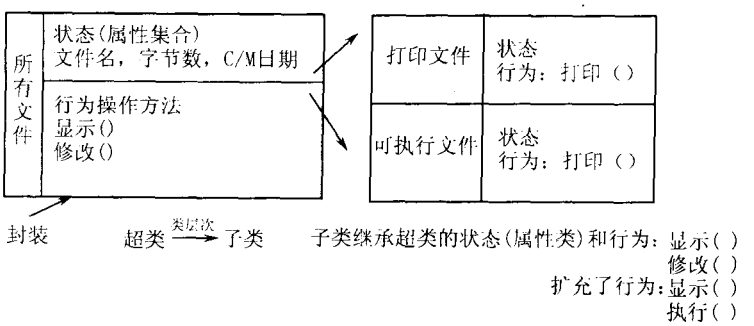

图 1-12 类层次示意图

类层次：所有的类组成了一个有根有向无环图，称为类层次（结构）。

一个类可以从直接或间接祖先（超类）中继承（Inherit）所有的属性和方法，该类称为子类。

(5) 继承（Inherit）：子类可以从超类中继承所有属性和方法。类继承可分为单继承和多重继承两种。

单继承——一个类只能有一个超类；

多重继承——一个类可以有多个超类。

这样，在已有类的基础上定义新的类时，可以定义特殊的属性和方法，而不必重复定义父类已有的东西，这有利于实现可扩充性（Extensibility）。

例 1-1 中的 E-R 图，可以设计成如图 1-13 所示的面向对象模型。

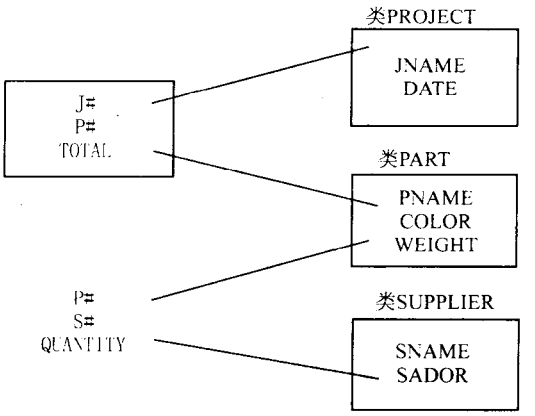

图 1-13　面向对象模型的示例

这个面向对象模型中有五个类，分别是：P-P，P-S，PROJECT，PART，SUPPLIER，其中类 P-P 的属性 J#取值为类 PROJECT 中的对象属性，P#取值为类 PART 中的对象属性，类 P-S 中的属性 P#取值为类 PART 中的对象，属性 S#值为类 SUPPLIER 的对象，这就充分表达了图 1-6 中的 E-R 图的全部语义。

面向对象数据模型比网络、层次、关系数据模型更具有丰富的表达能力。但正因为面向对象模型的丰富表达能力，模型相对复杂，实现起来较困难，所以尽管面向对象系统很多，但大多是试验型的或专用的，尚未通用化。

1.3　数据库管理系统

数据库管理系统（Database Management System，DBMS）就是管理一个数据库的软件，它充当所有数据的知识库，并对它的存储、安全、一致性、并发操作、恢复和访问负责。DBMS 有一个数据词典（有时被称为系统目录）其中储存着它拥有的每个事物的数据，例如名字、结构、位置和类型，这种关于数据的数据也被称为元数据（metadata）。

在一条数据的生存周期里（从它的创建到删除），这条数据的逻辑和物理信息都被记录在数据词典中。数据库系统管理员（Database Administrator, DBA）应该熟悉 DBMS 的数据词典；在数据库的整个生命周期内，数据词典为他服务。

数据库管理系统是基于某种数据结构模型、以统一的方式管理和维护数据库，并提供访问数据库接口的通用软件。DBMS 一般具有以下功能：

(1) 数据库定义功能，提供 Data Defined Language(DDL)及其翻译程序，定义数据库结构（模式及模式间映像）、数据完整性和保密性约束等。

(2) 数据库操纵功能，提供 Data Manipulate Language(DML) 及其翻译程序，实现对数据库数据的查询、插入、更新和删除操作。

(3) 数据库运行和控制功能，包括数据安全性控制、数据完整性控制、多用户环境的并发控制等。

(4) 数据库维护功能，包括数据库数据的载入、转储和恢复，数据库的维护和数据库的功能及性能分析和监测等。

(5) 数据字典（Data Dictionary，DD），存放数据库各级模式结构的描述，是访问数据库的接口。在大型系统中，DD 也可单独成为一个系统。

(6) 数据通信功能，包括与 OS 的联机处理、分时处理和远程作业传输的响应接口等，这一功能对分布式数据库系统尤为重要。

1.3.1 数据的安全

数据库安全管理的基本功能是保护数据库和其内容免受非法暴露、更新或破坏。除此之外，安全还有许多其他功能。安全有法律、社会和伦理方面的考虑，它关系到某人访问信息的权限以及让其他人知道这些权限。

在成品数据库中，安全性一直是一个重点，在开发或者测试数据库中也常如此。这通常不是有没有安全性的问题，而是有多大安全性的问题。除了操作系统和网络安全设施外，一个 DBMS 通常提供几层安全设施。最常见的情形是，DBMS 要求用户登录用户账号的口令，口令被验证为真，才能访问数据库。

DBMS 也提供其他的机制，例如组、角色、权限，这些都是提供更加精良的安全措施。提供这些安全级不是为了加强安全性，而是为了建立商业安全策略。例如，只有一个经过验证属于航空组的用户才能存取航空数据；又例如，只有经过验证拥有操作者角色的用户才能备份数据库。

1. 标识和鉴别

无论是银行、办公室、家、计算机系统，还是数据库系统，安全系统中的初级防卫是在入口点。入口点的控制至少应与前面所列的这些系统差不多。例如，你从银行取钱之前，你必须先提供身份识别表明你在那里有账户。然后，你要从指定账户取钱时，银行通常还要进一步鉴别你的身份。

数据库系统或计算机系统没有区别。在允许某人进入系统前，用户必须首先让系统知道他是谁。然后，系统可根据授权用户的列表，并指示是否允许其进入。这是第一级的防卫：用户名的识别。

第二级的防卫判断用户是否就是他或她声明的人。此鉴别可采用多种形式。在最简单的形式中，系统可能还需要一个只让系统和用户知道的口令。在较复杂的形式中，鉴别可能包括眼睛的视网膜扫描、指纹、密码卡输入、语音识别或一些其他的复杂识别形式。

我们可将鉴别的策略和机制扩充到较低数据库，作为这两个手段的次要用法。我们可将访问入口与文件、对象、关系、操作或数据库系统的其他成分结合起来。要折衷考虑此类系统的构造开销、维护开销和全部开销造成的性能损失。

2. 授权

一旦允许某个用户进入系统，我们要不要限制他访问系统？在你的主机中，如果有人可进入并浏览你的 e-mail 或新产品的方案，你肯定不愿意。你可能希望进一步对用户加以

限制，从而使他们只能看到和使用那些你认为合适他们访问的数据。

授权是限制某个用户在系统内的视图和操作的方法。授权策略和机制用于决定和设置访问级别、访问粒度和被允许访问的关系。授权机制包括各种访问的能力，类似指定座位和访问质量的许可证，或者说该机制是表的访问控制表，他们表示数据库成分或数据库用户，以及它们对数据项的访问和执行的权限。

访问限制级别的范围可从不能访问到访问和操作不受限制。常见的权限介于两者之间。系统可能给一些用户只读权限，而限制其他一些用户读访问。一些用户可能只是生成数据，却没有理由查看内部数据，因而只有写访问权限。一些用户可能获得某个数据项或区域的读和写权限，而其他人可能获得特殊的操作权限。授权机制必须提供用户得到何种数据库视图及其内容。

我们可能希望让一些用户看到一个关系或对象的部分视图而非全部，或只限制访问数据库的一个子集。实现此功能的典型方法是通过一个视图机制，它只让某个用户了解数据库的特定子类型或子成分。最后，即使有了这些限制，数据库安全子系统还必须与真正的破坏分子打交道，这些人企图通过非法手段获得有关数据库的信息。控制这种类型的访问必须考虑访问的关联和访问的持续时间和范围，还应包括一组推断规则，通过这些规则的检查，可以设法监测某个人企图破坏数据库的隐秘手段。要抵制这类人，数据库可能需要对信息加以限制。例如，数据库可能给破坏分子发送错误或不完全数据，以便隐藏保密数据。

出于安全考虑，要严格管理一个数据项的多个变量，这很复杂。安全系统可能需要用多种方法查看合作信息，以便限制和监测某人何时企图用非密信息可能推导出受限或保密信息。

最后，我们讨论一下浏览的额外开销。对真正的用户而言，浏览是一个重要特性，而对安全侵犯者而言是一个强大工具。浏览器能查看不超过她或他许可的安全级别的各种数据，可能收集足够的无关信息演绎出一些重要信息。例如，如果你想要知道何时在五角大楼中有大事发生，只需要查看比萨饼的分送信息。如果在短时期内和一日内各钟点的比萨饼数量大有增长，说明有一项大行动正在密谋或计划中。

安全是数据库领域内的重要问题，虽然其使用和实现要折衷考虑限制访问的需求和限制访问有关的开销。开销可能是一次性的，涉及数据库的初始构造和生成，或者以后可能还有开销。以后的开销来自权限的在线检查和测试访问中可能隐藏的工作。

1.3.2　维护和实施完整性

数据的完整性是指它的一致性和正确性。对于一致的数据，在它所有的出现的场合中，必须以相同的方式建模和实现，对于正确的数据，它必须是正确、精确和有意义的。

DBMS 维护完整性的一个方法是在一条数据项发生改变的过程中进行锁定。数据库通常是在数据库页级或行级锁定数据。锁定偶尔也允许并发操作，在下面你将涉及到这种情形。

DBMS 实施完整性的另一个方法是：如果一条数据储存在多个地方，那么把变化复制到这条数据上。DBMS 实施完整性的最后一个方法是通过监视输入的或改变的数据值，使他们全部符合要求的规格（例如：一个范围检查）。

如果接着的是合适的建模和实现过程，DBMS 会自动地帮助实施这种完整性，例如，

通过一个触发器或者约束条件。没有完整性，数据是没有价值的。有了完整性，数据就是信息。完整性不仅能增强数据，它还给予数据以价值。

当 DBMS 提供多用户存取时，它必须管理并发操作。那就是说，当几个人同时访问相同的数据库（特别是同一条数据）时，DBMS 必须保证这种并发访问能够以某种形式实现。并发可以广义上定义为同时发生，即两个或者多个用户在同一时期访问同一数据。

DBMS 实现并发操作的方法倒不太复杂，但背后的实际程序却很复杂。本质上说，当两个或者多个人只想简单地浏览一下同一个数据而不改变它时，一切还好。但当至少一个人想改变数据而其他人想浏览或者也想改变数据时，DBMS 必须储存多个备份。当每个人都完成改动后，DBMS 必须将所有改变的备份保存为一条正确的数据。

前面已经提到并发管理的一种方式是加锁。通常来讲，锁越精密（越小），并发性就越好（那就是说，更多的用户无需等待即可同时访问）。行通常比较小的数据库页或块都小，因此，行级锁可较好地管理短小、杂乱数据的处理，块级锁可较好地管理长的、连续数据的处理。

这就是并发性和完整性是如何结合的。当一个人想查看或者更改一条数据时，这个人就是执行数据库的一个事务。

1.3.3 理解事务

作为 DBMS 标准的一部分，DBMS 有一个事务管理器(transaction manager)，它的目的是管理并发操作和确保事务的完整性。事务管理器的工作是艰巨的，因为它必须允许许多人同时访问相同的数据，并且在访问之后要把数据放回到数据库中，就好像是某个时间上只有一个人存取数据，他完成工作后另一个人才开始工作，这样确保数据的正确性。DBMS 解决数据的多个备份的基本方案就在这中间。如果（并且是只有）数据是串行的，那么在保持数据的准确性的同时也进行了事务处理。简单地说，DBMS 必须重新整理所有改变，以使得它们的最终结果仿佛发生在一个文件中。

事务是并发或工作的单位。不能产生比一个事务更小或更少的东西，也就是说，没有人能够只完成数据改变的一部分。全部事务必须都是原子的，所以每个单独的事物要么完成，要么不完成。直到 20 世纪现代物理发展起来以前，原子一直被当作物质的最小单位。同样，事务也是并发的最小单位，它要么全有，要么全无。一个被完成的事物可以说是被提交了，没有完成的事物则是被回滚了。

DBMS 用事务作为恢复的单位来控制恢复，正常完成、手工要求中止以及意料之外的退出都要求 DBMS 重新访问数据的多个备份来提交或回滚数据。为了回滚或者前滚，DBMS 保持了一个事务日志。回滚是一个撤销操作，前滚是一个重做操作，例如，由于一个硬件或者软件错误，一个已提交的事物无法将它的操作从内存中存储到磁盘就会发生前滚，DBMS 只是简单地重做这个操作。因此，在 DBMS 中事务恢复的关键是一个事务必须是原子的，而且在必要时可以做、不做或重做。

1.3.4 数据独立性

1. 数据库的体系结构

数据库系统的体系结构是数据库系统的一个重要总框架。尽管数据库软件产品种类繁多，使用的数据库语言各异，基础操作系统不同，采用的数据结构模型相差甚大，但是绝

大多数数据库系统在总体结构上都具有三级模式的结构特征。数据库的三级模式结构有外模式、模式和内模式组成，如图 1-14 所示。

(1) 外模式：又称子模式或用户模式，是模式的子集，是数据的局部逻辑结构，也是数据库用户看到的数据视图。

(2) 模式：又称逻辑模式或概念模式，是数据库中全体数据的全局逻辑结构和特性的描述，也是所有用户的公共数据视图。

(3) 内模式：又称存储模式，是数据在数据库系统中的内部表示，即数据的物理结构和存储方式的描述。

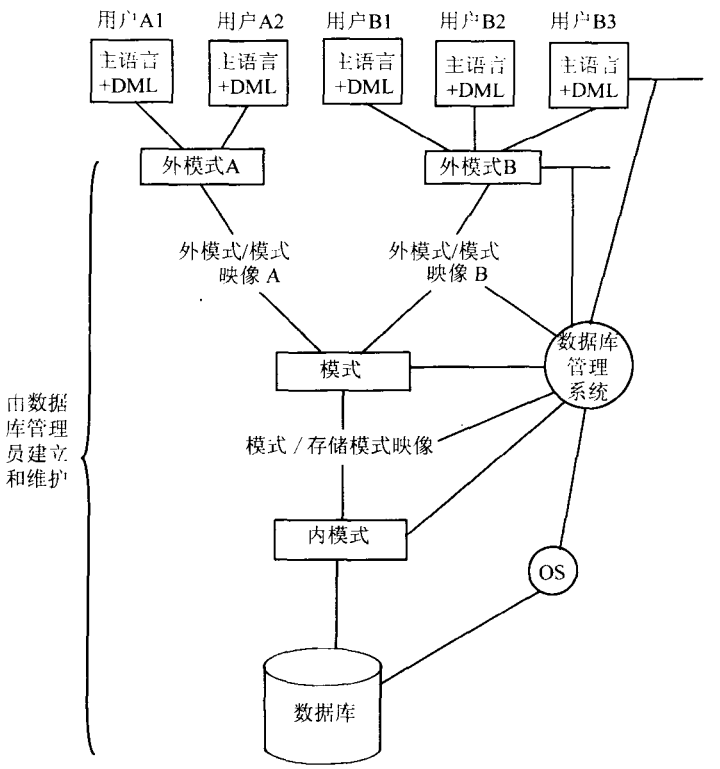

图 1-14　数据库系统的体系结构

2. 数据独立性

数据库系统的三级模式是对数据的三级抽象，数据的具体组织由数据库管理系统负责，使用户能逻辑地处理数据，而不必考虑数据在计算机中的表示和存储方法。为了实现三个抽象层次的转换，数据库系统在三级模式中提供了两次映像：外模式/模式映像和模式/内模式映像。所谓映像就是存在某种对应关系。

外模式到模式的映像，定义了外模式与模式之间的对应关系。模式到内模式的映像，定义了数据的逻辑结构和物理结构之间的对应关系。

由于上述的两次映像，使数据库管理中的数据具有两个层次的独立性。

一个是数据物理独立性。模式和内模式之间的映像是数据的全局逻辑结构和数据的存储结构之间的映像，当数据库的存储结构发生了改变，例如存储数据库的硬件设备变化或存储方法变化，引起内模式发生变化。由于模式和内模式之间的映像，是数据的结构可以

保持不变，因此应用程序可以不必修改。

另一个是数据的逻辑独立性，外模式和模式之间的映像是数据的全局逻辑结构和数据的局部逻辑结构之间的映像。例如，数据管理的范围扩大或某些管理的要求发生改变后，数据的全局逻辑结构发生变化，对不受该全局变化映像的那些局部而言，至多改变外模式与模式之间的映像，基于这些局部逻辑结构所开发的应用程序就不必修改。

数据的独立性是数据库系统的最重要的特征之一，采用数据库技术使得应用程序的维护工作量大大减轻。

1.3.5 与数据库通信

如果你不能和一个 DBMS 对话，那么这个 DBMS 就不是很好的。你可能会问怎样和 DBMS 对话，可以通过一种存取或查询语言访问数据库。结构化查询语言(Structure Query Language，SQL)是当今主要的查询语言，它主要用于管理主流类型的 DBMS——关系型 DBMS(RDBMS)。所有与数据库相关的通信往来都是通过 DBMS 完成的，为了做这件事，你可以使用 SQL 或者其他类似的东西。数据库系统管理员(DBA)使用查询语言来建立并维护数据库，用户使用查询语言来访问数据库并查看或者更改数据库。

1.4 关系数据库管理系统

1970 年，E.F.Codd 创立了关系模式概念。在 RDBMS（例如 DB2）产生之前，层次（IMS）和网状(IDMS)模式是常见的。在这些模式之前，使用平面文件（操作系统文件不一定平面）来建立数据库，并且使用第三代语言（3GL）访问例程。实际上，一些专用系统仍然是按这种方式建立的，只是进行了修改或根本没有变化。在大型机和微机中依然存在着许多这样的遗留数据库。CODASYL（数据系统语言协会）是数据库任务组（Database Task Group, DBTG）创建的一种数据库标准，这是一种基于 COBOL 的网络数据库标准，并且 IDMS 是一个厂商的实现。但是，从 70 年代起，RDBMS 已经逐渐地控制了市场，如 Oracle、Sybase、Informix 和 Ingres 。

最近，面向对象（Object-Oriented, OO）的 DBMS 已经成为最为突出的数据库管理系统，并找到了许多适当的应用环境，如 CAD/CAM、工程、多媒体等等。面向对象的 DBMS 适于在这些领域中应用，因为在一个几乎非事务性的环境中，它们具有控制复型数据类型实力。由于竞争，RDBMS 厂商为了提供包括文本、音频、图像和视频数据类型的面向对象/多媒体性能，已经制造了商业可用的通用服务器。Oracle 的 Universal Server 就是一个例子。另外，用户定义的数据类型或可扩展类型，已经被扩大或增加到核心数据库服务器中，Oracle 8 就提供了这样的性能。类似这样的 RDBMS 产品被认为是混合的，然而它们明显比以前的 RDBMS 更具有主流性。

此外，多维数据库（Multi-Dimensional Database, MDD）也分享了部分市场份额，这些数据库为带有许多必须被多维存取或列表的变量（例如行为科学数据）的应用提供了高度索引化的数据。在传统的 RDBMS 中，这几乎是不可能实现的，数据只允许单独使用。再者，为和 MDD 竞争，RDBMS 供应商提供了一些它们自己的层次产品，这些产品提供超级索引化的数据，并使用了特殊的技术，例如位映射索引。Oracle 的 Express 就是一个多维数