第 1 章 绪　　论

数据库技术作为一种先进的数据管理技术，极大地推进了数据管理技术乃至计算机应用技术的发展，使企业和组织能够高效地存储、管理和使用日益增长的数据。然而，为了深入学习和掌握数据库技术，必须首先清楚数据库领域中的一些基本概念，了解数据库系统体系结构、数据库管理系统的分类、数据库技术的研究领域以及发展历史等内容，从而为后面的学习奠定基础。

1.1 数据库系统概述

数据库系统是对应用了数据库技术的计算机系统的一种概称，因此数据库技术基本都包含在数据库系统的范畴中。数据库系统涉及一些最基本的概念，这些概念在现实应用中很容易混淆，也是学习数据库技术必须首先了解和区分的对象。

1.1.1 数据

数据是数据库中存储和管理的基本对象。数据这个名词在现实生活中并不陌生，如“财务数据”、“采购数据”等都是经常听到或接触的。但是，什么是数据？下面给出数据的定义：

【定义 1-1】 数据：数据是人们用来反映客观世界而记录下来的可以鉴别的符号。

这个定义的核心意思是“数据是符号”。之所以强调这一点是因为数据库系统除了存储和管理数据之外，还同时管理另外一些内容（如后面要介绍的模式等）。

由于现实世界中存在着不同类型的符号，因此数据也可分为两种基本类型：数值数据和非数值数据。数值数据记录了由数字所构成的数值，例如，职工张三的年龄是一个数值数据，学生李四的英语成绩也是数值数据等。非数值数据则包括了像字符、文字、图像、图形、声音等特殊格式的数据。在实际应用中，非数值数据也很常见，如人的姓名（字符）、照片（图

像)等。现有的数据库技术都可以同时支持数值数据和非数值数据的存储与管理。

在实际应用中,如果仅存储数据,一般来说是没有意义的。这是因为数据本身只是符号而已,而同样的符号在不同的应用环境中可能会出现完全不同的解释。例如,65 这一数据在教学管理系统中可能代表了某个学生某门课程的成绩,在职工管理系统中可能表示某个职工的体重,而在学生管理系统中还可能表示计算机系 2011 级的学生人数。因此,数据与其代表的语义是分不开的,在存储数据的同时必须知道数据所代表的语义。

除了 93、"张三"这类表示单一值的简单数据外,现实生活中还存在着复合数据。复合数据是由若干简单数据组合而成的。例如,学生记录"(李明,197205,中国科学技术大学,1990)"就是由简单数据"李明"、"197205"、"中国科学技术大学"和"1990"构成的一个复合数据。复合数据同样也是与其语义不可分的。像上面的学生记录,其语义在不同应用环境下可能完全不同。例如,在高校毕业生管理系统中可能表示"学生姓名,出生年月,所在学校,毕业年份"这样的语义,而在另一个系统中则可能表示"学生姓名,出生年月,录取大学,入学时间"的另一种语义。

1.1.2 数据库与模式

简单地讲,数据库是一个存储数据的仓库。但是,这种定义还不够确切,因为数据库中的数据并不是随意存放的,而是有一定的组织和应用特点。严格的数据库定义如下:

【定义 1-2】 数据库:数据库(database,DB)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。

这个定义指出了数据库的下述几个特点:

(1)数据库是数据的集合,因此数据库只是一个符号的集合,本身是没有语义的。

(2)数据库中的数据不是杂乱无章的,而是有组织的。确切地说,是按一定的数据模型组织、描述和存储的。

(3)数据库中存储的数据量通常是海量的。如果是少量的数据,通常不需要使用数据库技术来管理,借助文件系统就可以实现。实际上,存储的数据量越大,越能体现数据库技术相对于文件系统的优势。

(4)数据库通常是持久存储的,即存储在磁盘等持久存储介质上。

(5)数据库一般是被多用户共享的。换句话说,如果一个数据集合只是为单用户服务的(如手机中的通讯录等),那么依靠传统的文件系统等数据管理技术基本也可以满足要求。只有在多用户共享的环境中,才能充分发挥数据库技术的优点。目前,除了少数专用的数据库产品外,绝大多数商用数据库产品都是面向多用户应用的。

(6)数据库一般服务于某个特定的应用,因此数据间联系密切,具有最小的冗余度和较高的独立性。

现实世界中有银行数据库、航班数据库、图书数据库等面向特定应用的数据库,但是不存在通用的数据库。即便都是图书数据库,不同的应用环境对数据组织、数据存储等也会有不同的要求。例如,某学校中的图书数据库中需要存储每一种图书的供应商,而另一个学校则可能不需要保存。这些都会影响数据库中数据的表示和组织方式。因此,数据库一般都是专门针对某个特定应用的。

由于数据库本身是没有语义的，因此用另一个概念，即数据库模式(database schema)来表达数据库的语义。数据库模式的定义如下：

【定义 1-3】　数据库模式：数据库模式是数据库语义的表达，它是数据库中全体数据的逻辑结构和特征的描述。

图 1-1 显示了数据库与数据库模式之间的关系。两者之间的关系与数据和数据的语义之间的关系是类似的。实际上，由于数据库本身就是数据的集合，因此数据库中所有数据的语义就构成了数据库的语义，即数据库模式。

图 1-1　数据库与数据库模式之间的关系

图 1-2 所示是数据库与数据库模式的一个例子。在这个例子中，假设数据库中只存储了学生数据。图 1-2 的左边显示了使用关系数据模型表示的数据库结构与内容(如前所述，数据库中的数据一般都是按某种数据模型进行组织的)，右边则分别显示了对应的数据库和数据库模式。关系数据模型是目前最流行的数据模型(现有的商用数据库产品基本上都是基于关系数据模型的)，它的基本数据结构就是图 1-2 左边显示的二维表格。但是，二维表格本身包含了表头和下面的数据行，从概念上讲，二维表格的表头表示了下方数据行的语义，所对应的结构就是此二维表格的模式(由于假设数据库中只有这一个表，因此此模式就是数据库的模式)，而下方的数据行集合则构成了数据库，即数据的集合。

学号	姓名	年龄
001	张三	20
002	李四	21
003	王五	22

学生(学号:char,姓名:char,年龄:int)　数据库模式

001	张三	20
002	李四	21
003	王五	22

数据库

图 1-2　数据库与数据库模式示例

1.1.3　数据库管理系统

数据库管理系统的定义如下：

【定义 1-4】　数据库管理系统：数据库管理系统(database management system，DBMS)是一个计算机软件，用于创建和管理数据库。

数据库管理系统从软件的分类角度来说属于计算机系统软件。系统软件一般是管理计

算机资源的软件。常见的系统软件有操作系统、数据库管理系统等。同样是系统软件，操作系统管理计算机中的全部资源，包括处理器、存储器、外部设备等，而数据库管理系统只管理计算机中的数据资源。操作系统本身也有数据管理的能力，即文件管理功能，但正是因为操作系统的文件管理功能在管理大规模共享数据时容易出现存取性能差、数据不一致等问题，所以才有了数据库管理系统。可以这么理解，数据库管理系统是一种专门用于高效管理数据资源的系统软件。

通常情况下，数据库管理系统运行在操作系统之上，也就是说，当涉及底层的磁盘操作时，数据库管理系统通常利用操作系统提供的磁盘存取服务来实现底层数据存取。目前，大多数的商用 DBMS 都采取了这种方式。但是，从理论上来说，数据库管理系统也可以完全绕过操作系统提供的数据输入/输出(input/output，I/O)服务，直接跟底层的磁盘打交道。现在一些大型的 DBMS，如 IBM DB2、Sybase ASE 等已经可以支持这种数据访问方式。此外，数据库管理系统通常不直接面向应用。作为系统软件，其职责在于管理和维护数据资源。同时，用户可以在数据库管理系统之上创建直接服务于应用的数据库应用系统(或称数据库应用软件)，从而构建基于数据库技术的应用软件，满足实际应用的需求。图 1-3 显示了用户应用、DBMS 和操作系统之间的层次架构。

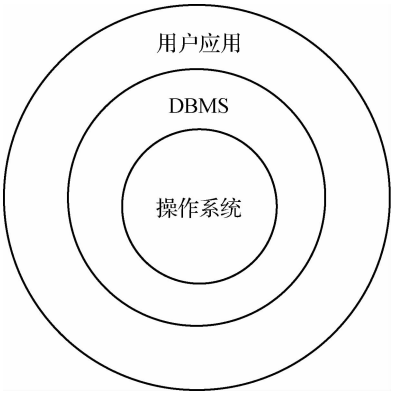

图 1-3　用户应用、DBMS 和操作系统之间的层次架构

由于数据库中的数据需要按某种逻辑结构进行组织，因此任何一个数据库管理系统在实现时必须基于某种数据模型，以保证所管理的数据库都能够按照统一的逻辑结构进行存储和存取。例如，目前常见的数据库管理系统，如 Oracle、Microsoft SQL Server 等都是基于关系数据模型的，因此也被称为关系型 DBMS，而另一些 DBMS，如 Versant、O2 等都是基于面向对象数据模型的，通常称为面向对象 DBMS。正是因为数据模型决定了数据库管理系统中的数据组织和操作方法，所以基于什么样的数据模型成为区分数据库管理系统特征的最主要因素。

在实际应用中常常见到的一些数据库产品，如 Oracle、Microsoft SQL Server 等，严格来讲都是指 DBMS。但随着计算机软件技术和应用的不断发展，目前的 Oracle、Microsoft SQL Server 等已经不单纯是 DBMS，而是一套以 DBMS 为核心的套件，也就是说，它们不仅提供了 DBMS 的核心功能，通常还包含一些其他软件功能，如数据导入/导出、备份管理等。这一点与 C++ 和 Visual C++ 的区别类似，C++ 好比是 DBMS，而 Visual C++ 则好比是

Oracle 等各种大型商用数据库软件。虽然 Visual C++提供了多种多样的开发功能，但C++是其核心，其本质仍然是 C++开发工具。在实际生活中只要知道 Oracle、Microsoft SQL Server 等这些产品本质是 DBMS 就可以了。

1.1.4　数据库系统

数据库系统是一个泛指的概念，其定义如下：

【定义 1-5】　数据库系统：数据库系统(database system，DBS)是指在计算机系统中引入了数据库后的系统，即采用了数据库技术的计算机系统。

数据库系统与其他计算机系统的区别在于它是以数据库为基础的。目前所见的许多系统，如银行信息系统、电子政务系统等，都可以称为数据库系统，因为它们背后都有 DBMS 和数据库的支持。随着应用数据类型和数据量的不断增长，在计算机系统中采用数据库技术来管理数据已经成为一种普遍性的解决方案，因此数据库系统在实际应用中已经非常普及。

作为一个计算机系统，数据库系统同样包含了软件、硬件、用户等要素。一个数据库系统通常包括了计算机硬件平台、操作系统、DBMS(及数据库)、应用程序以及用户。图 1-4 给出了数据库系统的组成。

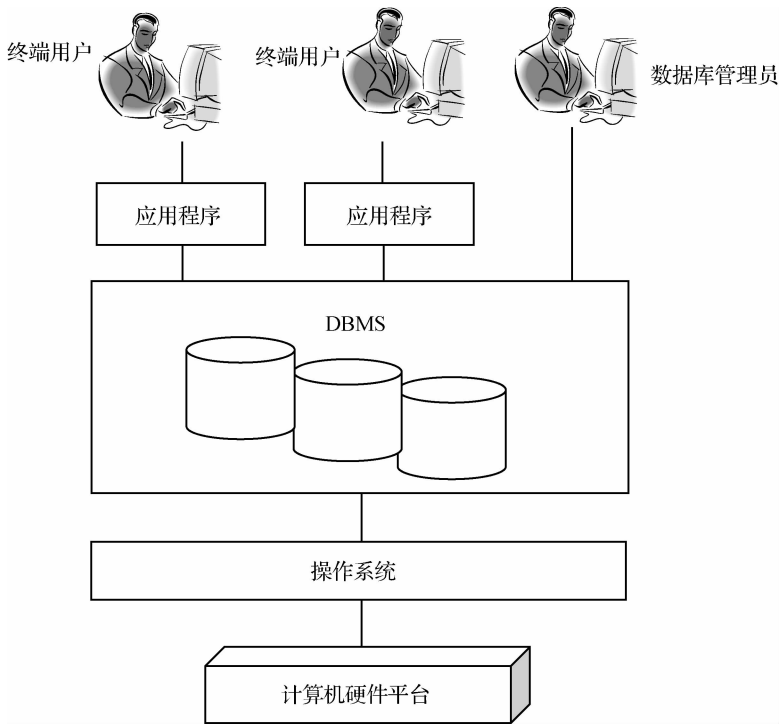

图 1-4　数据库系统的组成

图 1-4 把数据库系统中的用户分为两种类型：终端用户与数据库管理员(database administrator，DBA)。从图 1-4 中可以看到，终端用户直接与应用程序交互，而数据库管理员

则直接与 DBMS 打交道。终端用户相当于银行中的前台柜员，而数据库管理员则好比是银行数据库系统的管理员和维护人员，他们负责管理和维护数据库系统的正常运行。通常，数据库管理员会通过对 DBMS 的一些参数进行配置和调优，或者利用 DBMS 提供的一些管理功能，如备份、监控、存取控制等，进行系统维护。

数据库、数据库管理系统、数据库系统这三个概念在实际应用中容易混用。不过，在上下文清楚的情况下，可以用“数据库”来指代数据库管理系统或者数据库系统。例如，“你们上机用什么数据库啊？”——“Microsoft SQL Server 2008”，这一场景下的“数据库”指的就是 DBMS。但是，在论文或者其他规范化文档的写作中要求严格区分这些概念，以保证论文的严谨性。例如，不能将“高校信息化数据库的设计”混淆为“高校信息化数据库管理系统的设计”，前者是指数据库的设计，而后者是指 DBMS 的设计，两者的内涵和难度不是一个量级的。

1.2 数据库系统体系结构

数据库系统既涉及了 DBMS、数据库应用程序等软件，也包含了存储有海量数据的数据库。数据库系统体系结构从架构的观点对数据库系统的软件以及数据库进行了分析，梳理其内部的层次和相互之间的联系，从而为人们深入理解和应用数据库技术提供帮助。数据库系统体系结构可以从两种不同的角度来理解：一是从 DBMS 的角度来看数据库内部的模式结构如何组织；二是从终端用户的角度来看数据库系统的软件架构如何组织。

1.2.1 研究数据库系统体系结构的原因

数据库系统与平常的计算机系统的主要区别在于它是以 DBMS 为核心的。DBMS 的引入一方面带来了数据库应用程序软件架构上的变化；另一方面也对如何合理组织和使用数据库提出了新的要求。

从软件架构上来看，引入 DBMS 之后系统中开始出现了数据库服务器。由于商用 DBMS 不仅提供了数据存储的功能，还兼有一定的数据库编程能力，也就是说数据库服务器也有能力承担一部分业务处理功能，因此在系统中如何合理地安排数据库服务器以及如何在应用程序和数据库服务器之间合理地分配业务处理能力是搭建数据库系统必须解决的问题。

此外，由于 DBMS 只提供了数据库创建和维护的功能，而不同的应用对数据库的设计和使用要求存在一定的差别，例如，有的应用中数据库模式通常是稳定不变的，而有的应用中数据库模式有可能会变化。由于在软件工程中软件维护的代价越来越高，因此人们在建立数据库系统时不希望由于 DBMS 和数据库的引入而导致软件的维护代价大幅增大。一旦出现这种情形，将十分不利于数据库技术的进一步发展和应用，因为企业可能会放弃 DBMS 的高效数据管理能力而去寻求其他维护代价较小的解决方法。因此，如何合理组织数据库模式结构以保证数据库与应用程序之间的独立性，并最终尽可能地降低软件维护代

价，是数据库技术在实际应用中不得不考虑的一个问题。

以上两点是研究数据库系统体系结构的原因，即 DBMS 的引入使我们必须从终端用户角度考虑合适的软件架构；数据库的实际应用必须考虑数据库模式结构的合理组织以尽可能地降低系统的软件维护代价。

1.2.2 不同视角的体系结构

数据库系统体系结构可以从两种视角加以理解：一是从终端用户的视角；二是从 DBMS 的视角。

从终端用户的视角看，数据库系统是一个软、硬件系统，DBMS 只是其中的一个部件。DBMS 与前端的数据库应用程序一起为用户服务，完成用户需要的各种业务处理需求。按照这一视角，数据库系统的体系结构是由客户端（数据库应用程序）和服务器（数据库服务器、Web 服务器、应用服务器等）组成的一种架构。同时，由于在数据存储方式、业务处理功能实现方式等方面存在着不同选择，数据库系统体系结构也存在多种方案。

从 DBMS 的视角看，数据库系统是以数据库为中心的系统，如何合理地设计数据库模式以及如何将数据库呈现给前端应用程序存在着多种选择。例如，可以直接将整个数据库完整地呈现给所有的用户，也可以为不同的用户封装并呈现需要使用的部分数据。这些不同的设计也会直接影响数据库应用程序的可维护性，即一旦数据库模式发生了变化，如果不采取合理的数据库模式结构设计，很可能会导致前端的数据库应用程序不得不大批量地修改源代码，从而带来极大的软件维护工作量。合理的数据库模式结构可以保证在数据库模式发生改变时只需要很少的工作量就可以保证前端的应用程序源代码不需要修改，从而实现数据库与应用程序之间较好的独立性。

1.2.3 ANSI/SPARC 体系结构

本节首先讨论基于 DBMS 视角的数据库系统体系结构。如前所述，这一体系结构关注于数据库的模式结构，因此也称其为“数据库体系结构”。在这种体系结构中，目前广泛采用“ANSI/SPARC 体系结构”的架构。

ANSI/SPARC 体系结构（ANSI/SPARC architecture）是 1975 年由美国国家标准协会的计算机与信息处理委员会中的标准计划与需求委员会提出的数据库模式结构。ANSI/SPARC 体系结构定义了标准的数据库模式结构。它不仅可以用来解释已有的商用 DBMS 的数据库模式结构，也可以作为研发新型 DBMS 时的数据库模式组织标准。目前，Oracle、Microsoft SQL Server 等商用 DBMS 都遵循和支持 ANSI/SPARC 体系结构。

在详细讨论 ANSI/SPARC 体系结构之前，有必要先说明一下数据库模式和数据库实例的概念。数据库模式是数据库中全体数据的逻辑结构和特征的描述，它仅仅涉及类型的描述，不涉及具体的值。与数据库模式相对的另一个概念是数据库实例（database instance）。数据库实例是数据库模式的一个具体值。在数据库系统中，数据库模式反映了数据的结构与联系，而数据库实例反映的是某一时刻数据库的状态。一个数据库模式可以对应多个数据库实例，例如，不同时刻的数据库状态就对应了一系列的数据库实例。另一方面，数据库

模式一旦设计完成后就相对稳定，修改较少，而数据库实例通常是频繁变化的。

图 1-5 给出了数据库模式和数据库实例的一个示例。图中左边是一个教学信息管理系统中的数据库模式，而右边则显示了不同时刻的两个数据库实例。需要注意的是，在实际的 DBMS 中，数据库模式的描述要比图 1-5 复杂得多，数据库实例所包含的数据也要多得多。这里只是一个例子，使大家可以明白数据库模式与数据库实例之间的差别。

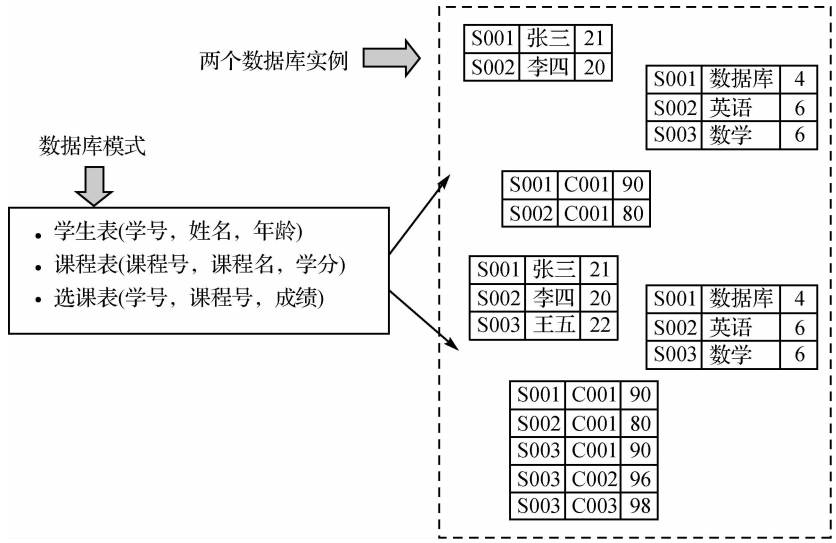

图 1-5　数据库模式与数据库实例示例

ANSI/SPARC 体系结构可以用一句话概括，即三级模式结构＋两级映像。三级模式结构是指数据库模式由外模式(external schema)、内模式(internal schema)和概念模式(conceptual schema)来描述；两级映像是指外模式/模式映像和模式/内模式映像，它们定义了三级模式结构之间的相互关联。

图 1-6 显示了 ANSI/SPARC 体系结构。其中，概念模式定义了逻辑层(logical level)的模式结构，它表示整个数据库的逻辑结构，例如，数据记录由哪些数据项构成，数据项的名字、类型、取值范围，数据之间的联系、数据的完整性等。概念模式不涉及数据物理存储的细节和硬件环境。一个数据库只有一个概念模式。概念模式的实例称为概念视图(conceptual view)，它实际上就是特定时刻的整个数据库，是数据和值的集合。在 ANSI/SPARC 体系结构中规定概念模式通过模式 DDL(data definition language)进行定义。DDL 是数据库语言的一种，主要功能是操纵数据库模式。在后面介绍 SQL 语言时将会对其进行详细介绍。另外，如果不特别说明，模式这一名词往往代表的是概念模式的含义。因此在许多时候也可以直接用模式来指代概念模式。

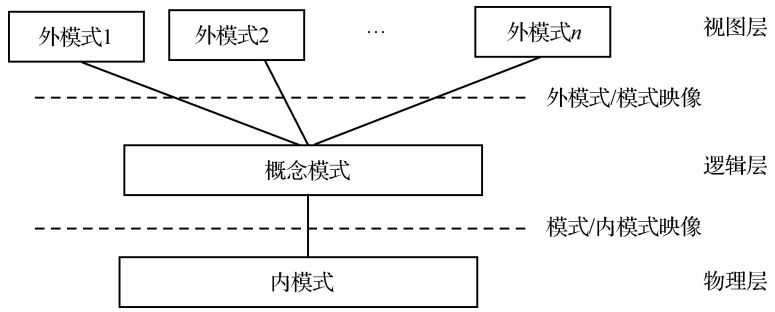

图 1-6　ANSI/SPARC 体系结构

外模式，也称为用户模式(user schema)或子模式(sub schema)，它定义了视图层(view level)的模式结构，它建立在概念模式之上，代表了单个用户所见到的局部数据库数据的逻辑结构。由于数据库系统中一般存在着多个用户，而不同用户对数据库的存取需求是不同的，因此每一个用户都应当有自己对应的外模式，即一个概念模式之上可以定义多个外模式。注意，这里的用户实际对应着数据库应用程序，因为在实际系统中是由数据库应用程序来负责存取数据的。在数据库系统中，外模式定义了用户与数据库系统之间的数据接口，对于用户来说，他们所看到的数据库是由外模式定义的。外模式的引入使得同一个数据库可以给不同用户呈现不同的视图。例如，在一个图书馆数据库中，借书者眼里的数据库内容与图书馆工作人员眼里的数据库内容可能完全不同——借书者眼里的图书只有图书名、作者、出版社、ISBN 等内容；而图书馆工作人员眼中的图书则还可能包含库存数、购买价格、购买单位等信息。外模式的实例称为外部视图(external view)。在 ANSI/SPARC 体系结构中规定外模式通过外模式 DDL 进行定义。

内模式定义了物理层(physical level)的模式结构，它描述了数据库的物理存储结构和存储方式。例如，数据库记录的存储方式是顺序存储、按树组织还是散列存储，索引按什么方式组织，数据是否加密、压缩存储等。内模式所定义的数据库存储结构是一种基于磁盘块的存储结构，即它是以磁盘块为基本单位来描述数据库存储结构的。如前面提到的 B 树、散列等都是指磁盘块的组织方法。但是，内模式不涉及磁盘块的大小，也不考虑具体的磁盘物理参数(如柱面数等)。因此，内模式与底层真正的存储硬件之间实际上是独立的。由于 DBMS 通常建立在操作系统之上，因此这些与具体硬件打交道的工作实际上是由底层的操作系统负责的。与概念模式类似，一个数据库也只有一个内模式。内模式的实例称为内部视图(internal view)。在 ANSI/SPARC 体系结构中规定内模式通过内模式 DDL 进行定义。

图 1-7 给出了一个教学信息管理系统中三级模式结构的例子。假设在该系统中存在三类用户：学生、选课管理人员和课程评价管理人员。学生只能看到课程信息和自己的选课数据；选课管理人员可以看到所有的课程信息和所有学生的选课数据；而课程评价管理人员则可以看到全部的课程信息和课程评价数据。在图中为这三类用户分别定义了三个外模式。

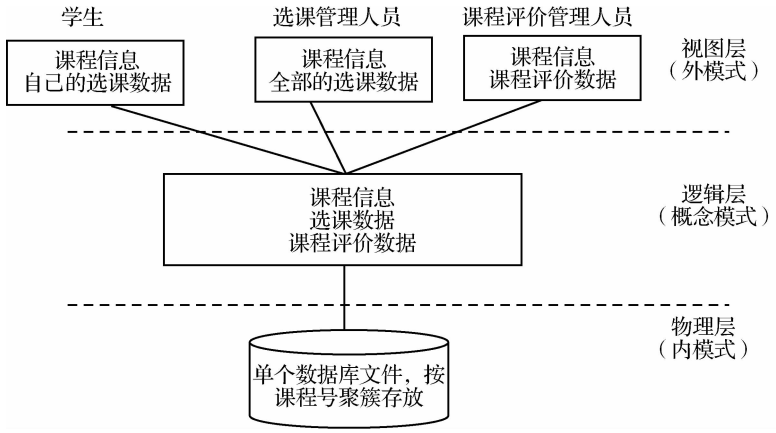

图 1-7　三级模式结构示例

ANSI/SPARC 体系结构中的“两级映像”是指三级模式结构之间的“外模式—模式”映像与“模式—内模式”映像。这两级映像实现了三级模式结构间的联系和转换，使用户可以逻辑地处理数据，而不必关心数据的底层表示方式和存储方式。

“外模式—模式”映像定义了外模式与概念模式之间的联系。外模式是通过“外模式—模式”映像建立在概念模式之上的，“外模式—模式”映像的引入使得用户可以使用不同于概念模式中的属性名称来定义外模式，也可以使用一些属性运算从概念模式中得到外模式的属性。但“外模式—模式”映像最重要的价值在于它可以实现数据库的逻辑数据独立性，即概念模式发生改变时，可保持外模式不变，只要修改“外模式—模式”映像即可，从而保持用户应用程序不变，保证了数据库数据与应用程序的逻辑独立性。

“模式—内模式”映像定义了概念模式与内模式之间的联系。这一级映像的主要作用是保证数据库的物理数据独立性，即当数据库的内部存储结构发生改变时，可保持概念模式不变，只要修改“模式—内模式”映像即可，从而保持外模式以及用户程序的不变，保证了数据库与应用程序之间的物理独立性。

假设有概念模式 Employee(E＃，D＃，Name，Salary)，下面的语句定义了外模式 EMP(Emp，Dept，Name)，同时也定义了“外模式—模式”映像。此处使用的是 SQL 语句，具体细节将在本书后面讨论：

```
Create View EMP(Emp,Dept,Name)
As
Select E# as Emp,D# as Dept,Name
From Employee
```

如果概念模式发生了变化，例如，属性 E＃修改成了 Emp。新的概念模式为 Employee(Emp，D＃，Name，Salary)。此时可以执行下面的操作修改“外模式—模式”映像：

```
Drop View EMP
Create View EMP(Emp, Dept, Name)
As
Select Emp, D# as Dept, Name
From Employee
```

上述操作的含义是先用 Drop View 语句删除原先定义的外模式，即 SQL 中的 View，然后再重新创建一个新的外模式。

执行完这一操作后，应用程序中的外模式仍然是 EMP(Emp，Dept，Name)，没有任何变化，因此原先的源程序也无须做任何修改。这就是所谓的数据库的逻辑数据独立性。物理数据独立性的思想与此类似。

1.2.4 数据库应用系统体系结构

从终端用户的角度看，数据库系统体系结构严格意义讲是数据库应用系统的体系结构，因为它是从使用者(应用程序)角度来理解数据库系统部件的集成方式的。这类体系结构可以分为单用户结构、主从式结构、分布式结构、客户机/服务器结构、浏览器/服务器结构等。这些结构是随着计算机技术的发展而逐步提出的。目前最常见的是客户机/服务器结构和浏览器/服务器结构。

1)单用户结构

单用户结构是早期的数据库应用系统体系结构。在这种结构中，整个数据库系统(应用程序、DBMS、数据库等)装在一台计算机上，为一个用户独占，不同机器之间不能共享数据。

在计算机网络技术出现之前，各个部门之间很难直接共享数据，所以单机应用是最常见的方式。但由于一个企业或组织内部的数据一般都需要被多个部门使用，因此单用户结构很容易造成数据冗余、不一致等问题。例如，一个企业的各个部门都使用本部门的机器来管理本部门的数据，各个部门的机器是独立的，由于不同部门之间不能共享数据，因此企业内部存在大量的冗余数据，例如，人事部门、会计部门、技术部门必须重复存放每一名职工的一些基本信息(如职工号、姓名等)。

目前，单用户结构在企业应用中已经很少使用，在一些新型应用中，如智能手机等，还在继续使用这种体系结构。理论上讲，如果一个单一的数据库不需要多用户共享，可以考虑采用单用户结构。

2)主从式结构

主从式结构也是早期的一种数据库应用系统体系结构。在 20 世纪 90 年代之前，由于计算机的处理能力还比较弱，因此往往采用小型机、中型机甚至大型机来作为数据存储和处理的主机，如果多个用户需要同时使用主机，则通过采取连接多个终端的方式来实现。终端往往只具有简单的输入/输出功能和非常弱的处理能力。主从式结构就是在这一环境下发展起来的数据库应用系统体系结构。在这一结构中，一个主机带多个终端，可以支持多用户的数据存取。数据库系统，包括应用程序、DBMS、数据库等，都集中存放在主机上，所有处理任务都由主机来完成，然后各个用户通过主机的终端并发地存取数据库，共享数据资源。

主从式结构的优点是易于管理、控制与维护。它的缺点主要有如下两点：

(1)当终端用户数目增加到一定程度后，主机的任务会过分繁重，成为瓶颈，从而使系统性能下降。

(2)系统的可靠性依赖主机，当主机出现故障时，整个系统都不能使用。这一问题常被形象地称为“单点故障”。

在主从式结构中，客户机（终端）基本不具备数据处理能力。随着计算机的逐步发展，客户机逐渐开始具备数据存储、处理等能力，可以将系统的一部分负荷迁移到客户机上，从而使得传统的主从式结构逐步地淡出了应用领域。

3）分布式结构

分布式结构是随着计算机网络技术的发展而产生的一种数据库应用系统体系结构。在现实生活中，有一些应用系统，如银行、连锁企业等，由于拥有分布在各地的多个分支机构，它们的数据也同样是分布在各地的，但是整个企业逻辑上又是一个整体，有的业务需要建立在整个企业的全局数据上，而有的业务只需要使用某个分支机构的局部数据。以银行为例，当开设银行账户时，往往只需要访问开户地本地的银行数据库，而当执行异地取款、转账等操作时则需要访问整个银行的全局数据。分布式结构的产生极大地满足了这类物理上分布的企业的需求。

在分布式结构中，数据库中的数据在逻辑上是一个整体，但物理上分布在计算机网络连接的不同结点上。“物理上分布，逻辑上一体”是分布式结构最主要的特征。网络中的每个结点都可以独立地处理本地数据库中的数据，执行局部应用，同时也可以同时存取和处理多个异地数据库中的数据，执行全局应用。需要注意的是，简单地将多个单机的数据库系统通过网络相连接并不能称为分布式结构，因为它们在逻辑上不是一个整体（在用户眼里不是一个单一的数据库），用户不能透明地存取所有的数据。图 1-8 给出了分布式结构的一个示例。在这个例子中，整个图书数据库物理分布在三个校区，但逻辑上是一个整体，在用户眼里仿佛只有一个数据库。

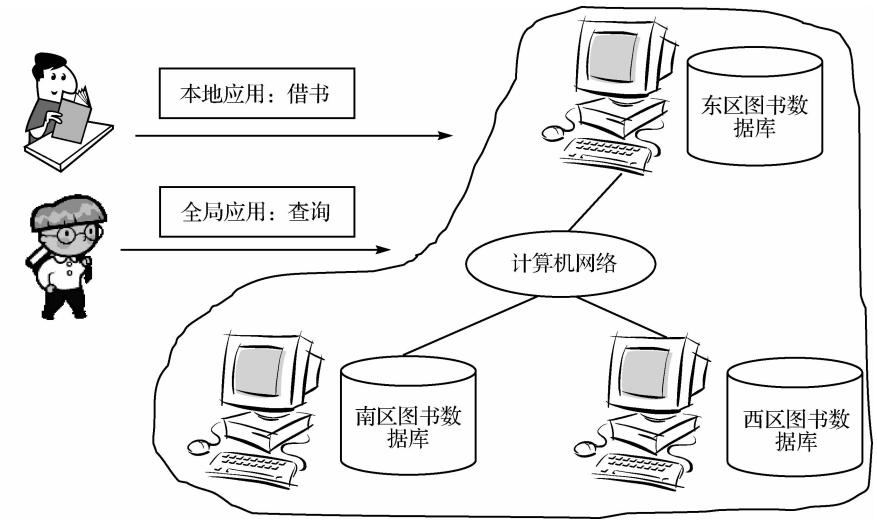

图 1-8　分布式结构示例

分布式结构的优点是适应了地理上分散的企业或组织对于数据库应用的需求。其缺点主要有以下两点：

（1）数据的分布式存放给数据的处理、管理与维护带来困难。

（2）当用户需要经常访问远程数据时，系统效率会明显受到网络传输的制约。

分布式结构需要 DBMS 具有专门处理数据划分、分配的技术，另外，由于数据有可能复

制在各个结点中，因此如何维护数据的一致性是一个关键的问题。分布式结构由于对各个分布结点的处理能力要求较低，因此在 20 世纪 90 年代得到了广泛应用，但由于分布式结构在维护方面存在的问题，近年来大型企业应用又开始逐步从分布式回归到了集中式，即通过建设大规模数据中心将企业的数据集中存储在一个位置。

4)客户机/服务器结构

客户机/服务器结构(client/server architecture，C/S 结构)是 20 世纪 90 年代随着计算机的快速发展而产生的一种数据库应用系统体系结构。随着计算机处理能力的提高，可以将应用系统的业务处理放到计算机上(这与主从式结构不同)，而服务器只提供数据存储和管理等服务。

在客户机/服务器结构的数据库系统中，存在两类结点(见图 1-9)：一类称为数据库服务器，主要提供 DBMS 的功能；另一类称为客户机，主要运行应用程序以及一些前端的数据库开发工具。这两类结点通过计算机网络(一般是局域网)相互连接，用户操作客户机上的应用程序执行业务处理，如果需要访问数据库，则通过网络访问数据库服务器即可。

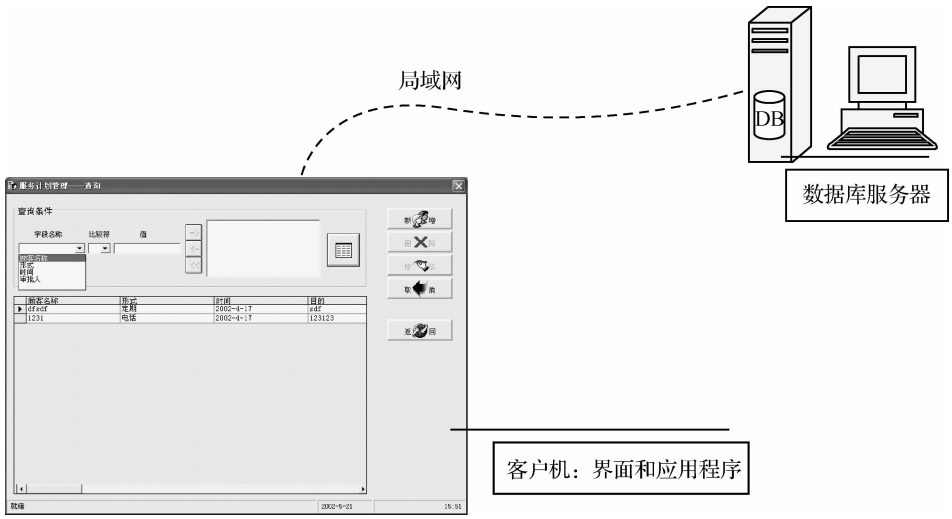

图 1-9　客户机/服务器结构示例

在最早提出的客户机/服务器结构中，一般包含一台数据库服务器和多台客户机，这种最原始的结构现在被称为集中式 C/S 结构。集中式 C/S 结构又可以细分为两种：以前端为主的集中式 C/S 结构和以后端为主的集中式 C/S 结构。以前端为主的 C/S 结构强调把所有的业务处理功能全放在客户机端，服务器只负责存储和维护数据库。由于服务器通常具有更高的配置和处理能力，因此研究者希望能够将部分业务处理任务放到服务器上执行，由此提出了以后端为主的集中式 C/S 结构。在这种结构中，服务器不仅负责数据的存储与维护，还承担一部分业务处理功能。当然，这需要 DBMS 提供一定的编程能力。目前，大部分商用 DBMS，如 Oracle、Microsoft SQL Server、Sybase 等，都支持服务器端的编程。业务处理程序在服务器端需要使用 DBMS 认可的专门的过程化 SQL 语言来编写，编译好后存放在数据库中，然后前端的客户机应用程序可以在需要时调用它们。目前，在较大规模的实际应用中，以后端为主的集中式 C/S 结构更为普遍。

客户机/服务器结构还可以和分布式结构结合形成分布式 C/S 结构。在分布式 C/S 结构中,存在着多台分布的数据库服务器和客户机。客户机透明地访问多台分布的数据库服务器,就好像只有一个服务器,也就是说,从应用程序这一端是感知不到后台数据库服务器结构的变化的。这一点也正是分布式结构的主要特点,即逻辑上一体。分布式 C/S 结构需要 DBMS 具有相应的分布式数据管理的功能,一般要求是一个分布式 DBMS。目前商用的 DBMS 通常通过提供需要另外购买的分布式扩展模块来支持分布式结构。

客户机/服务器结构的主要优点可归纳为以下几点:

(1)客户端的用户请求被传送到数据库服务器,数据库服务器进行处理后,只将结果返回给用户,从而显著减少了数据传输量。

(2)客户机与服务器一般都能在多种不同的硬件和软件平台上运行。

(3)可以使用不同厂商的数据库应用开发工具。

在 Web 技术出现之前,客户机/服务器结构是最流行的架构。主流的软件开发平台也都以 C/S 开发为主,如 VC++、VB、Delphi、Power Builder 等。进入 21 世纪后,随着 Web 开发技术的发展,浏览器/服务器结构开始流行,越来越多的 Web 开发平台开始出现,基本形成了与传统 C/S 开发半分江山的局面。尽管如此,客户机/服务器结构仍具有独特的优点,仍是一种主要的数据库应用系统体系结构,在银行、证券、企业内部管理等有区域性应用需求的系统中广泛使用。

客户机/服务器结构的主要缺点有如下两点:

(1)系统安装和维护复杂,工作量大。每台客户机都需要安装专门开发的应用程序。此外,一旦系统需要升级,则需要耗费大量的人力去重复安装升级。

(2)相同的应用程序要重复安装在每一台客户机上,从系统总体来看,大大浪费了系统资源。

5)浏览器/服务器结构

客户机/服务器结构的缺点在浏览器/服务器结构出现之后得到了显著改善。浏览器/服务器结构(browser/server architecture,B/S 结构)可以看成是 Web 时代的客户机/服务器结构。两者的不同之处在于:在浏览器/服务器结构中,客户机应用程序的界面是统一的,即浏览器。此外,浏览器/服务器结构的应用系统通常运行在 Internet 之上(当然理论上也可以只在局域网内使用,不过要求支持 TCP/IP,即 Intranet)。

由于要支持浏览器,因此在浏览器/服务器结构中,服务器端除了数据库服务器外,还增加了 Web 服务器。Web 服务器相当于一个网页的容器,每次客户机端的浏览器都是首先向 Web 服务器发送请求,然后 Web 服务器根据需要访问数据库服务器,将得到的结果形成网页发送给客户机,并显示在客户机的浏览器上。图 1-10 给出了这种结构的示例。

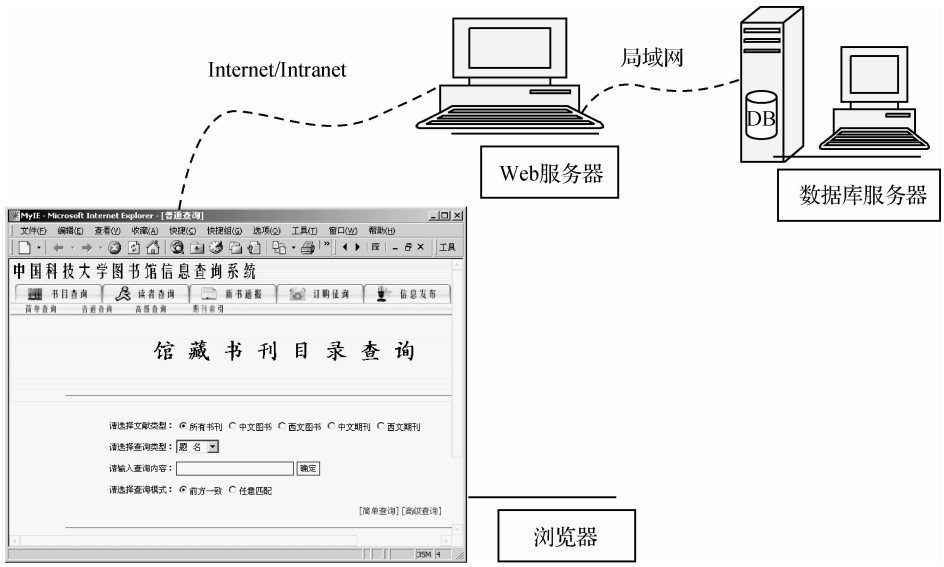

图 1-10　浏览器/服务器结构示例

浏览器/服务器结构可以看成是一种适合互联网应用的客户机/服务器结构。与客户机/服务器结构相比，其主要优点有以下两点：

(1)统一的客户机界面，减少了应用安装和维护的工作量。在浏览器/服务器结构中，客户机只要有浏览器即可，不需要另外安装应用程序。应用程序升级时，也只需要在 Web 服务器升级即可，因此大大降低了系统维护的工作量。

(2)基于 Web 技术，支持互联网应用。互联网应用的优点是可以跨地域运行。传统的 C/S 结构应用一般只能局限在局域网中。因此，对于像电子商务系统、网上银行等应用，浏览器/服务器结构具有明显的优势。

但浏览器/服务器结构也存在以下两个缺点：

(1)安全性问题，用户访问无地域限制。相比之下，由于 C/S 应用只运行在由局域网连接的系统内部，通常是一个部门或者一栋大楼，其用户类型、访问来源、访问数量等都很容易控制，因此安全性要高很多。

(2)开发工具的能力相对较弱。

1.3　DBMS 的功能与分类

DBMS 作为系统软件承担了计算机系统中数据资源管理的主要任务。DBMS 最基本的功能是创建和维护数据库，但还需要提供其他一些功能。此外，数据库技术从 20 世纪 60 年代发展以来，出现了许多新的模型和技术，这些模型和技术使 DBMS 呈现出一些独有的特点，形成了不同类型的 DBMS。

1.3.1 DBMS的功能

DBMS的功能大致可归纳为以下几点。

1)数据库定义

DBMS首先需要提供创建数据库的功能。数据库是由若干对象构成的一个集合,因此DBMS需要提供对不同数据库对象的创建和维护能力,包括表、视图、索引、约束、用户等。

2)数据库操纵

数据库是为前端应用程序服务的,因此DBMS必须提供数据库的存取能力,主要是指对基本表的操纵,包括增、删、改、查等。

3)数据库保护

为了保证数据库的安全,DBMS必须提供一定的数据库保护功能。数据库保护功能通常包括两种方式:一是提供数据库故障后的恢复功能;二是提供防止数据库被破坏的技术。具体的数据库保护功能包括数据库恢复、并发控制、完整性控制、安全性控制等。

4)数据库的建立和维护

DBMS需要提供初始数据的转换和装入、数据备份、数据库的重组织、性能监控和分析等功能,这些功能对于保证DBMS的实用性是必不可少的。在实际的数据库应用系统中,系统正式运行之前通常要求数据库中有一些必需的初始数据。这些数据或者来自遗留的旧系统,或者是应用系统运行必需的一些先验数据。数据库的备份、性能监控等也是维护数据库所不可或缺的功能。这些功能通常由DBMS提供的一些实用程序来完成。

1.3.2 DBMS的分类

由于采用不同的数据模型和实现技术,DBMS也存在着不同的类型。数据库技术自产生以来形成了许多子分支,如面向对象数据库、时空数据库等,这些新型的数据库技术通常都是因为采用了新的数据模型或者新的技术而形成的。DBMS可以从几个不同的角度来分类。

1)按数据模型分类

每个DBMS都有一个数据模型,因此按数据模型来划分DBMS是最常见的一种方式。

按数据模型划分,DBMS可以分为以下几种类型:

(1)网状DBMS。基于网状数据模型的DBMS。

(2)层次DBMS。基于层次数据模型的DBMS。

网状数据模型和层次数据模型都是20世纪60年代提出的数据模型,对于数据库技术的发展有过重要的意义,不过目前已经很少使用。一般把网状DBMS和层次DBMS称为第一代DBMS(或第一代数据库技术)。

(3)关系DBMS。基于关系数据模型的DBMS。关系数据模型采用了完全不同于网状数据模型和层次数据模型的数据建模方法,自20世纪70年代提出后得到了快速发展和广

泛应用，到目前为止仍是主流的数据库技术。已有的商用 DBMS，如 Oracle、Microsoft SQL Server、Sybase 等都是关系 DBMS。一般称关系 DBMS 为第二代 DBMS(或第二代数据库技术)。

(4)对象 DBMS。基于面向对象数据模型的 DBMS。面向对象的 DBMS 是在面向对象程序设计技术的基础上发展起来的，自 20 世纪 80 年代提出后得到了学术界和企业界的大力支持。但是由于对象 DBMS 存在一些固有的问题，目前还难以取代关系数据库技术。20 世纪 90 年代研究者们又将面向对象数据模型的一些特征加入到关系 DBMS 中，推出了对象关系 DBMS。一般把对象 DBMS，包括对象关系 DBMS，统称为第三代 DBMS(或第三代数据库技术)。

(5)其他 DBMS。基于其他数据模型的 DBMS，例如，针对半结构化数据的 XML 数据库技术等。

总体来说，第一代 DBMS 是开创者，它们直接造成了数据库领域的诞生；第二代 DBMS 是推动力，它极大地推动了数据库技术的发展和应用，产生了像 Oracle 这样著名的数据库企业；第三代 DBMS 是探索者，对数据库技术在图像、图形、文档等复杂数据管理方面的应用起到了重要的作用。

2)按支持的用户数分类

按所支持的用户数划分，DBMS 可分为单用户 DBMS 和多用户 DBMS。

(1)单用户 DBMS。同一时刻只允许一个用户使用数据库。这种 DBMS 一般使用在早期的单机上，只能满足单个部门的数据管理要求；或者用在智能手机等单用户系统中。单用户 DBMS 目前应用比较少。

(2)多用户 DBMS。可以同时支持多个用户并发访问数据库。目前常见的基本都是多用户 DBMS，包括 Oracle、Microsoft SQL Server 等。多用户 DBMS 适合企业的实际应用情况，因此得到了广泛的应用和发展。

3)按允许数据库分布的站点数分类

按允许数据库分布的站点数划分，DBMS 可分为集中式 DBMS 和分布式 DBMS。

(1)集中式 DBMS。数据库存放在单个物理站点。这是目前常见的 DBMS 类型。

(2)分布式 DBMS。允许数据库分布存储在多个物理站点上。分布式 DBMS 需要考虑数据如何划分、如何分配到多个物理站点上、如何在多个站点间复制数据等问题，同时在查询处理、恢复、并发控制等方面也都需要研究新的技术。大部分商用集中式 DBMS 都可以提供专门的分布式模块来实现分布式 DBMS 的功能。

4)按用途分类

按用途划分，DBMS 可分为通用型 DBMS 和专用型 DBMS。

(1)通用型 DBMS。这类 DBMS 可应用于不同类型的应用环境，能够适应不同数据的管理需求。如 Oracle、Microsoft SQL Server 等常见的 DBMS 都是通用型 DBMS，它们可以用于小型企业的信息管理系统，也能够支持银行、跨国企业、学校等不同应用系统的数据管理需求。

(2)专用型 DBMS。这类 DBMS 是专门针对某些特殊的数据管理需求而研制的，如针对时间/空间数据管理的时空数据库、针对多媒体数据管理的多媒体数据库等。它们对于这些

特殊的数据管理可能是高效的，但不一定适合其他类型的数据管理要求，因此适合应用在某些特殊的应用系统中，如航空航天、遥感监测等领域。

1.4 数据库系统的几个关键问题

数据库系统是以 DBMS 和数据库为核心的计算机系统。建立和使用数据库系统涉及了若干关键性的问题，包括如何构建 DBMS、如何设计数据库模式、如何存取数据库等。其中如何构建 DBMS 是从系统内核的角度来研究的，而如何设计数据库模式以及如何存取数据库则是从应用的角度来定义的。这些问题各自又包含了许多子问题，几十年来这些问题的分析与解决直接带动了整个数据库领域技术的发展。

数据库设计和数据库存取问题是本书讨论的重点。DBMS 的实现问题由于涉及系统内核，因此在本书后续内容中不对此作进一步的讨论。

1.4.1 DBMS 的实现问题

DBMS 的实现问题是指如何设计和实现一个 DBMS 来高效地组织和管理数据库。DBMS 是数据库系统中最为重要的部件，可以说，DBMS 的优劣直接决定了数据库系统的可用性和性能。

DBMS 的实现涉及许多理论问题，如数据模型、查询处理与查询优化、事务处理、恢复、并发控制、缓冲区管理、存储管理等。图 1-11 给出了一个 DBMS 的系统结构。

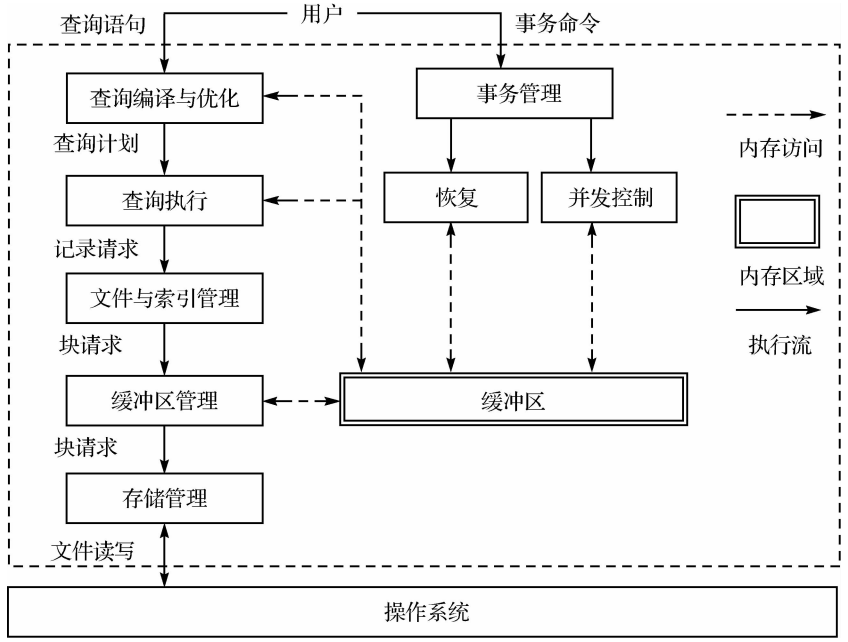

图 1-11 DBMS 系统结构

从理论上来说，最为重要的问题是数据模型问题，即数据库中的数据应该采取什么样的数据结构来表达、数据库应通过什么方法来操作以及数据约束如何表达等问题。由于不同的数据模型在建模思路上有着很大的差别，因此导致 DBMS 的实现技术也出现许多不同。例如，基于关系数据模型的关系 DBMS 采用二维表格的方式来组织数据，而基于面向对象数据模型的对象 DBMS 则采用对象的方式来组织数据。数据库领域到目前为止总共出现了三位图灵奖（计算机领域的最高奖）的获得者，其中两位，即通用电器的 C. W. Bachman 和 IBM 的 E. F. Codd，都是因为在数据模型上的突出贡献而获奖的，可见数据模型问题在数据库领域中的重要地位。

第二个问题是查询处理与查询优化问题。在数据库技术产生之前，一般通过文件系统来管理数据。当数据规模增大并且数据共享越来越迫切时，文件系统方式就会面临着性能低下、数据不一致等问题。DBMS 与文件系统相比，一个最主要的优势就是在大规模数据上仍能够保证较高的存取性能。为了实现这一点，DBMS 中的查询处理与查询优化技术是决定性要素。所谓查询处理，就是 DBMS 接受用户的查询请求并且经过一系列处理之后从数据库中返回查询结果给用户的过程。在过去的文件系统中，一般只能通过遍历文件的方式来查找用户想要的数据，因此当文件增大时性能会大幅下降。而 DBMS 采用了特定的逻辑结构来组织数据，因此在查询处理过程方面可以设计高效的查询处理算法来提高查询响应的速度。另外，同一个查询在回答时存在着多种执行方案，查询优化技术就是通过特定的优化算法选择极优的查询执行计划并交给 DBMS 执行，从而保证较高的查询性能的。查询处理与查询优化技术自数据库技术产生之后就一直是数据库领域中的热点研究问题。

第三个问题是事务处理问题，包括事务理论、并发控制、恢复等，这些问题都是以事务为基础的。事务是 DBMS 中不可分的操作序列。事务理论的提出使得数据库保护技术得到了实质性的发展，对数据库技术的实用化做出了重要的贡献。在事务的基础上，如何设计高效的并发控制机制和恢复机制是保证数据库一致性和正确性的关键问题。虽然在关系数据库上并发控制和恢复技术已经相对成熟，但一旦涉及新型数据管理技术，如 XML 数据、移动对象数据等，仍需要研究新的方法。微软公司的 Jim Gray 正是因为在事务处理方面的贡献而获得了这一计算机领域的诺贝尔奖。

另外一些问题，包括缓冲区管理、存储管理等涉及了外存和内存数据的组织和管理，它们对于 DBMS 的访问性能通常也非常重要。不过，这些问题在操作系统中也同样存在，因此在 DBMS 的实现上可以借鉴操作系统领域的部分成果。例如，在缓冲区管理方面，可以采用最近最少使用（least recently used，LRU）算法作为缓冲区替换算法，在存储管理方面，也可以采取基于磁盘块/页的文件管理方法。

1.4.2 数据库设计问题

数据库本身并不能构成应用程序直接满足用户的需求，而是需要和前端开发的程序相结合构成一个完整的数据库应用系统。对于一个特定的应用环境，应用程序所需要的数据库是需要数据库设计者预先根据需求设计的。所谓的数据库设计问题是指如何针对特定应用的需求设计一个合理的数据库模式结构。

数据库模式结构既包括概念模式，也包括外模式和内模式的设计。在实际应用中，如何

保证数据库模式满足用户的需求是主要问题。这其中既涉及一些理论问题，也涉及工程性问题。

从理论上来说，由于同样的数据需求在设计中存在着多种模式设计方案，那么首先需要对如何判断模式设计的好坏给出回答。其次，如果一个模式存在问题，通常会采用模式分解的方法来进行优化，但是对于如何优化、分解过程何时结束等问题还需要给出理论上的回答。

从工程上来看，数据库设计与软件设计一样，当问题规模变大时，如果没有一定的工程化方法指导，将会导致最终的设计结果出现各种各样的问题，从而极大地降低软件的可维护性和可用性。在这一方面，目前数据库领域一般的做法是借鉴软件工程领域规范化的设计过程来进行数据库设计。

1.4.3 数据库存取问题

数据库作为数据集合是为应用程序服务的。数据库只有被用户使用才能体现出它的价值。但是用户如何才能高效地使用数据库？所谓数据库存取问题是指应用程序如何高效地存储数据库中的数据。

数据库存取问题要求 DBMS 本身必须提供相应的数据存取方式，因此有没有高效的数据存取 API 是衡量 DBMS 性能的一个主要指标。从理论上来说，数据存取可以有多种方式，如利用 API、组件和数据存取语言等。在数据库领域，目前标准化的方式是通过数据存取语言，即数据库语言(database language)来存取。而且，数据库语言是用户存取数据库的唯一方式。

数据库语言一般包括如下几种子语言：

(1)数据定义语言(data definition language，DDL)。专门用于定义和操纵数据库模式结构的语言。

(2)数据操纵语言(data manipulation language，DML)。专门用于操纵数据库数据的语言，包括数据的增、删、改、查等。DML 是用户最常用的语言。

(3)数据控制语言(data control language，DCL)。专门用于数据库存取控制的语言，主要用于用户的授权与验证。

(4)嵌入式数据库语言。专门用于嵌入前端的程序设计语言(称为宿主语言)中完成数据库访问的语言。由于宿主语言与数据库语言的语法、标识符等存在差异，因此需要有专门的嵌入式数据库语言来解决宿主语言如何存取数据库的问题。嵌入式数据库语言在早期数据库应用开发中经常使用，但随着面向对象程序设计语言和开发工具的发展，目前除了在一些嵌入式平台上已经很少使用了。

数据库语言只是从 DBMS 角度为应用程序提供数据库访问的方式。然而，在实际应用中，如何将用户的查询需求表达为数据库语言仍是一个有难度的问题。这要求数据库应用系统的程序员具有较强的数据库语言表达能力，这也是学习和使用数据库技术必须要掌握的内容。

1.5　数据库技术的发展历史

数据库技术是数据管理技术的一个发展台阶。在数据库技术出现之前，人们普遍采用文件系统来管理数据。但是，随着数据规模的不断增长以及数据共享需求的提出，文件系统方式越来越难以适应数据管理的要求。另一方面，数据库技术自 20 世纪 60 年代开始发展以来，经历了网状数据库、层次数据库、关系数据库、对象关系数据库等发展阶段，而且即使到了今天的 XML 数据库、NoSQL 数据库等，还在不断发展。了解数据库技术的发展历史，可以看到整个数据库领域在最近几十年里的历程，也可以对当今一些新的数据库技术的背景有更深入的理解。

1.5.1　数据管理技术的发展历程

数据库技术从概念上讲只是数据管理技术的一种。只不过数据库技术在数据管理方面具有高效、一致等优点，所以才得到了用户的认可。要了解数据库技术的发展历程，首先应当对整个数据管理技术的发展历程有所认识，了解数据库技术的发展背景。

数据管理技术的发展历程大致可归纳为三个阶段，即人工管理阶段、文件系统阶段和数据库管理阶段。

1）人工管理阶段

人工管理阶段主要集中在 20 世纪 50 年代中期以前。当时计算机刚刚面世不久，处理能力还非常弱，数据管理能力也非常有限。

人工管理阶段的数据管理具有以下特点（见图 1-12）：

（1）数据不保存在计算机中。此时还没有出现二级存储概念，如磁盘等，数据都是纯二进制数据，并且以打孔纸带的形式表示。

（2）应用程序自己管理数据。应用程序根据自己的要求准备打孔纸带形式的数据，这些数据只能被自己使用。不同的应用程序需要各自准备数据。

（3）数据无共享。

（4）数据与应用程序之间不具有独立性。如果应用程序发生修改，原先的数据一般不能继续使用，需要再重新准备打孔纸带。同理，如果数据修改了，应用程序也同样无法处理。

（5）只有程序的概念，没有文件的概念。此时还没有文件存储的概念。

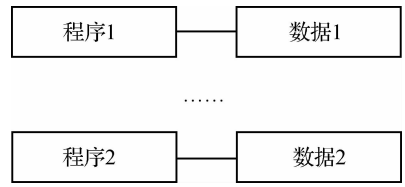

图 1-12　人工管理阶段的数据管理特点

2)文件系统阶段

在20世纪50年代中期到60年代中期,出现了文件系统形式的数据管理技术。它主要是随着操作系统技术的发展而提出的。

这一阶段的主要特点是数据以文件形式保存和管理。图1-13给出了文件系统阶段数据管理的示意图。这一阶段数据管理的主要特点可归纳为以下几点:

(1)数据以文件形式存在,由文件系统管理。

(2)数据可以长期保存在磁盘上。

(3)数据共享性差,冗余大。必须建立不同的文件以满足不同的应用。例如,在一个教学信息管理系统中,教师数据同时被选课、科研管理、人事管理等应用使用,在文件系统阶段只能将教师数据文件复制到这些不同的应用中。这样,一方面带来了数据的冗余存储,另一方面如果某些教师数据发生了修改,则很容易出现数据的不一致。

(4)数据与应用程序之间具有一定的独立性,但非常有限。应用程序通过文件名即可访问数据,但文件结构改变时必须修改程序。

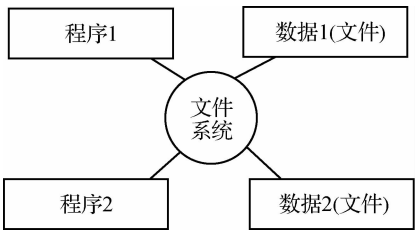

图1-13 文件系统阶段的数据管理特点

3)数据库管理阶段

20世纪60年代末开始,数据管理进入数据库管理阶段。这一阶段引入DBMS实现了数据管理(见图1-14),其主要特点如下:

(1)数据结构化。DBMS采用了数据模型来组织数据,不仅可以表示数据,还可以表示数据间的联系。

(2)高共享,低冗余。应用程序之间可以高度共享数据,并且可以保证数据之间的最小冗余。

(3)数据独立性高。数据的修改不会影响到应用程序的运行,具有高度的数据独立性。

(4)数据由DBMS统一控制,应用系统中所有的数据都由DBMS负责存取。

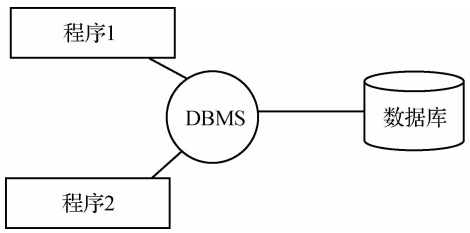

图1-14 数据库管理阶段的数据管理特点

1.5.2 数据库技术的发展历程

数据库技术从20世纪60年代末开始发展，至今仍是计算机领域中一个非常活跃的研究方向。下面简单总结了数据库技术发展历程中的一些里程碑：

(1)1961年：通用电器(GE)的C. W. Bachman设计了历史上第一个DBMS——网状数据库系统集成数据存储(integrated data store，IDS)。Bachman是一名工业界的研究人员，为了解决项目中的复杂数据管理问题而设计了IDS，开创了数据库这一新的研究领域。Bachman本人也因为在网状数据库方面的贡献于1973年获得了计算机领域的最高奖项——图灵奖。这是第一个获得图灵奖的数据库研究人员。

(2)1968年：IBM设计了层次数据库系统IMS(information management system)。

(3)1969年：CODASYL的DBTG(dataBase task group)发表了网状数据模型报告，奠定了网状数据库技术。层次数据库技术和网状数据库技术一般被合称为第一代数据库技术。

(4)1970年：IBM的E. F. Codd在*Communications of ACM*上发表了论文*A Relational Model of Data for Large Shared Data Banks*，提出了关系数据模型，奠定了关系数据库的理论基础。关系数据模型采用了一种简单、高效的二维表形式组织数据，从而开创了数据库技术的新纪元。E. F. Codd本人也因为在关系数据模型方面的贡献于1981年获得了图灵奖。关系数据库技术也被称为第二代数据库技术。

(5)1973—1976年：E. F. Codd牵头设计了System R。System R是数据库历史上第一个关系数据库原型系统，其字母R是Relation的首字母。之所以称为原型系统而不是产品，是因为System R开发完成后并没有及时商业化，从而导致Oracle后来居上。在这期间，加州伯克利大学的M. Stonebraker设计了Ingres。Ingres是目前开源DBMS PostgreSQL的前身。在20世纪70年代，Ingres是少数几个能和IBM的IMS系统竞争的产品。

(6)1974年：IBM的Boyce和Chamberlin设计了SQL。SQL最早是作为System R的数据库语言而设计的。经过Boyce和Chamberlin的不断修改和完善，最终形成了现在流行的SQL。目前，SQL已经成为ISO国际标准。前面提到的Ingres就是因为没有在其系统中支持SQL而导致了最终的没落。

(7)1976年：IBM的Jim Gray提出了一致性、锁粒度等设计，奠定了事务处理基础。Jim Gray本人也因为在事务处理方面的贡献获得了1998年的图灵奖。这是第三位获得该奖项的数据库研究人员，也是到目前为止最后一位。

(8)1977年：Larry Ellison创建了Oracle公司，1979年发布Oracle 2.0，1986年Oracle上市。Larry Ellison早期在为执行美国国防部(DoD)的一个项目时遇到了数据管理方面的问题，后来他看到了E. F. Codd发表的关于System R的论文，于是基于System R的思想很快实现了一个数据管理系统，并且将其商业化。在其后的发展中，Oracle很果断地采取了兼容SQL的做法，使其逐步占领了数据库领域的龙头地位。Oracle的发展对于数据库技术的商业化起到了十分重要的地位。后来Larry Ellison曾说IBM历史上最大的两个失误就是培育出了Microsoft和Oracle。

(9)1983年：IBM发布DB2。由于Oracle在商业领域的成功，使IBM意识到了数据库

技术的发展前景。由于其技术实力雄厚，因此马上推出了商业化 DBMS DB2。这一产品至今仍在市场上占据重要的地位。

(10)1985 年：提出面向对象数据库技术。面向对象数据库技术是随着面向对象程序设计(object oriented programming，OOP)技术提出的，实质上是持久化的 OOP。

(11)1987 年：Sybase 1.0 发布。

(12)1990 年：M. Stonebraker 发表“第三代数据库系统宣言”，提出了对象关系数据模型。面向对象数据库以及对象关系数据库技术标志着第三代数据库技术的诞生。但从商业应用上看，第三代数据库技术还远远赶不上关系数据库技术。

(13)1987—1994 年：Sybase 和 Microsoft 合作，发布 Sybase SQL Server 4.2。后合作破裂，Sybase 继续发布 Sybase ASE 11.0。

(14)1996 年：Microsoft 发布 Microsoft SQL Server 6.5。Microsoft SQL Server 是一个很特殊的产品，其版本号直接从 6.5 开始。因为 Microsoft 和 Sybase 合作破裂后双方都拥有了 Sybase SQL Server 的源码，但 SQL Server 这一名称从此开始属于微软，而 Sybase 则启用了 ASE 产品名称。

(15)1996 年：开源的 MySQL 正式发布。

(16)1998 年：提出了半结构化数据模型(XML 1.0)。由于网络数据管理需求的不断增长，XML 数据管理技术在近些年受到了重视，至今仍是数据库领域的一个研究热点。曾经有人将 XML 数据库技术命名为“第四代数据库技术”，但没有得到认可。

(17)2005 年：M. Stonebraker 等开发完成 C-Store。C-Store 是列存储的 DBMS(column-based DBMS)，它完全抛弃了传统基于行记录的数据库存储方式，从而开创了一个全新的研究方向。

(18)2007 年：NoSQL(非关系型数据库)在 Web 领域大行其道。传统的 SQL 数据库技术经过了几十年的发展和应用，在新的应用领域，如 Web、云计算等，面临着一些数据表示、查询处理方面的新问题。因此，NoSQL 数据库技术开始提出并且马上得到了多个互联网企业的支持，包括 Amazon(SimpleDB/Dynamo)、Google(BigTable)、Facebook(Cassandra)、Yahoo(PNUTS)等。在今后的几年里，NoSQL 数据库能否开创数据库技术的新时代还有待进一步验证。

本章小结

本章主要介绍了数据库系统中的基本概念，包括数据、数据库、数据库模式、数据库管理系统、数据库系统等，另外还着重介绍了数据库系统的体系结构，从 DBMS 视角和应用程序视角对数据库系统体系结构进行了讲述。此外，本章还系统介绍了 DBMS 的功能和分类、数据库系统的关键问题以及数据库技术的发展历程。

通过本章的学习，读者应能够清晰地区分数据库系统中的一些基本概念，并对数据库系统体系结构，尤其是三级模式结构和两类数据独立性有深入的理解，同时对数据库领域的一些研究问题以及发展历史有所了解。

习　　题

(1)数据库和数据库模式之间是什么关系?

(2)数据库管理系统和数据库系统之间是什么关系?

(3)什么是 ANSI/SPARC 体系结构?

(4)什么是逻辑数据独立性和物理数据独立性?

(5)第一代 DBMS、第二代 DBMS 和第三代 DBMS 分别指什么?

(6)DBMS 的基本功能有哪些?

(7)描述 DBMS 的基本组成。