# 专题 1 数据库系统的基本概念

随着信息技术（包括计算机技术、通信技术和网络技术等）的飞速发展，信息系统渗透到社会的各个领域，作为其核心和基础的数据库技术也得到了越来越广泛的应用。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。

“数据库原理”作为研究数据库技术的基本理论已经成为学习信息技术的重要专业课程。在高等教育的计算机科学与技术、电子信息科学与技术、信息管理与信息系统等有关专业，以及自学考试的计算机科学与应用、计算机信息管理、计算机网络等专业都开设了相关的课程。

“数据库原理”所涉及的内容，不仅是这些相关专业必备的知识基础，也是从事信息产业工作人员的必备知识与技能，同时也是进一步深入研究数据管理与应用技术的出发点。

## 1.1 本课程的目标与内容

本课程从软件工程的角度研究数据管理技术，属于软件基础课程，而不是一般程序设计语言的课程。新的软件工具在不断地更新和改进，这种更新和改进是建立在一定的理论基础之上的，并且随着这些更新与改进，理论也在不断地提高与完善。本课程主要包含以下几个方面的问题。

### 1.1.1 数据库系统的概念与特点

在计算机存储设备中，数据是以字节为单位存储的。数据存储的基本单位是数据项（字段），进一步的组织是记录和（数据）文件。但数据库不是文件的简单堆积，而是加入了对数据的组织和管理。

1. 数据管理的意义

数据作为信息的载体，在计算机应用技术深入发展的今天，对数据处理的问题也有了更高的要求。在数据处理中，常常不仅需要应用复杂的数据模型和算法，而且还需要处理大量的数据。因此，数据组织和管理的问题就更加突出了。数据管理手段从人工管理到文件系统再到数据库系统，反映了对信息资源的重要性的认识和管理措施的提高。

2. 数据库系统的特点

（1）数据结构化

在文件系统中，相互独立的文件记录内部结构的最简单形式是等长同格式记录的集合，数据库系统把文件系统中记录内部有结构的思想扩大到了多个记录型之间。从整体观点来看，不仅要考虑一个应用（程序）的数据结构，而且要考虑整个组织的数据结构问题。整个组织的数据结构化要求在描述数据时不仅描述数据本身，还要描述数据之间的联系。文件系统中尽管记录内部已有了某些结构，但记录之间是没有联系的，是孤立的。因此，数据的结构化是数据库主要特征之一，是数据库与文件系统的根本区别。

（2）数据共享性高、冗余度小、易扩充

数据库从整体观点来看待和描述数据，数据不再是面向某一应用，而是面向整个系统，这可以大大减小数据的冗余度，既节约存储空间，减少存取时间，又可避免数据之间的不相容性和不一致性。

对数据库数据的应用可以有很灵活的方式，可以取整体数据的各种合理子集用于不同的应用系统。而且当应用需求改变或增加时，只要重新选取不同子集或者加上一小部分数据，便可以有更多的用途，满足新的要求。这就是弹性大、易扩充的特点。

（3）数据独立性高

数据独立性是数据库领域的一个常用术语，它包括数据的物理独立性和数据的逻辑独立性。数据库系统提供了两方面的映像功能，一个是数据的存储结构与逻辑结构之间的映像或转换功能，另一个是数据的总体逻辑结构与某类应用所涉及的局部逻辑结构之间的映像或转换功能。

第一种映像功能使得当数据的存储结构（或物理结构）改变时，数据的逻辑结构可以不变，从而应用程序也不必改变。这就是数据和程序的物理独立性。简称数据的物理独立性。

数据库系统中某一类应用使用的数据通常是总体数据的子集，而且各类应用对同一数据的使用要求也不一定相同。数据库系统通常提供局部数据结构的说明功能。局部的数据结构可以按具体应用要求作一定的改变。系统提供对这些改变的映像和转换功能（即上面的第二种映像功能），使得当总体逻辑结构改变时，通过对映像的相应改变而保持局部逻辑结构不变。程序员是根据局部逻辑结构编写应用程序的，因而应用程序也就可以不必改变。这就是数据和程序的逻辑独立性，简称数据的逻辑独立性。

数据和程序的独立性，把数据的定义和描述从应用程序中分离出去。此外，数据的存取又由 DBMS 管理，用户不必考虑存取路径等细节，从而简化了应用程序的编制，大大减少了应用程序的维护和修改。

（4）统一的数据管理和控制

数据库是系统中对用户的共享资源。计算机的共享一般是并发的，即多个用户同时存取数据库中的数据甚至可以同时存取数据库中的同一个数据。因此，数据库管理系统必须提供以下几个方面的数据控制功能。

- 数据的安全性（ Security）保

数据的安全性，指保护数据以防止不合法的使用所造成数据的泄密和破坏，使每个用户只能按规定对某些数据以某些方式进行使用和处理。例如，系统用检查口令或其他手段来检查用户的身份，只有身份合格的用户才能进入数据库系统进行操作。提供用户密级和数据存取权限的定义机制，当用户对数据库执行操作时，系统自动检查用户能否执行这些操作。检查通过后才执行允许的操作。

■ 数据的完整性（Integrity）控制

数据的完整性，指数据的正确性，有效性和相容性。完整性检查提供必要的功能，保证数据库中的数据在输入、修改过程中始终符合原来的定义和规范，在有效的范围内，保证数据之间满足一定的关系。例如：月份是 1～12 之间的正整数；研究生性别是男或女；研究生年龄是大于 15 小于 45 的整数；研究生学号是惟一的；研究生所在的系、院必须是存在的有效的系、院等。

■ 数据库恢复（Recovery）

计算机系统的硬件、软件故障，操作员的失误以及人为的攻击和破坏，都会影响数据库中数据的正确性，甚至会造成数据库部分或全部数据的丢失。因此数据库管理系统必须能够进行应急处理，将数据库从错误状态恢复到某一已知的正确状态。

■ 并发控制（Concurrency）

当多个用户的并发进程同时存取、修改数据时，可能会发生相互干扰而得到错误的结果并使数据库完整性遭到破坏，因此必须对多用户的并发操作加以控制、协调。

（5）数据的最小存取单位是数据项

数据库既可以存取数据库中某一个数据项或一组数据项，也可以存取一个记录或一组记录。

综上所述，数据库是长期存储在计算机内有大量的、组织的、共享的数据集合。它可以供各种用户共享且具有最小的冗余度和较高的数据与程序的独立性。由于多种程序并发地使用数据库，为了能及时有效地处理数据，并提高安全性和完整性，必须有一个软件系统——数据库管理系统 DBMS（DataBase Management System），在数据库建立、运用和维护时对数据库进行统一控制，以保证数据的完整性、安全性，同时在多用户使用数据库时进行并发控制，在发生故障后对系统进行恢复。

3．数据库的定义

有关数据库的定义，许多书籍有不同的叙述方式，本书从应用的角度定义如下。

数据库是长期存储在计算机存储设备内、有组织的、共享的数据集合。这些数据按一定的数据模型组织、描述、存储，具有较小冗余度、较高的数据独立性和易扩充性，并可为各种用户共享。

4．数据库系统的组成

使用数据库技术的计算机系统称为数据库系统，由如下几部分组成。

（1）数据库：包括实际存储的数据和对数据库的定义。

（2）硬件支持系统：包括计算机、内外存储器、输入输出设备和通信设备等。

（3）软件支持系统：包括操作系统、数据库管理系统以及应用开发系统。

（4）人员：与数据库系统的设计、创建、使用、维护等工作相关的人员。包括

①数据库管理员。

②数据库设计人员。

③系统分析员。

⑤应用程序设计人员。

⑥ 各种系统用户和其他相关人员。

### 1.1.2 数据库系统的发展

数据库系统起源于 20 世纪 60 年代中期，其发展可以划分为三代。

第一代数据库系统，即层次数据库系统和网状数据库系统，主要支持层次和网状数据模型，其特点是支持三级抽象模式的体系结构；用指针来表示数据之间的联系；数据定义语言和数据操纵语言相对独立；数据库语言采用过程性语言。该数据库系统的发展过程如下：

1．1964 年，美国通用电气公司的 Bachman 等人开发成功世界上第一个数据库管理系统（DBMS）——IDS（Integrated Data Store）系统，奠定了网状数据库系统的基础。

2．1969 年，美国 IBM 公司开发成功世界上第一个商品化 DBMS 产品—— IMS（Information Management System）系统，这是一个层次数据库系统。

3．1969—1970 年，美国 CODASYL（Conference On Data System Language）协会下属的 DBTG（Database Task Group）对数据库方法进行了系统的研究，提出了 DBTG 报告，建立了以网状数据模型为基础的数据库系统概念、方法和技术。

第二代数据库系统，即关系数据库系统（RDBMS），主要支持关系数据模型，有严格的理论基础，概念简单、清晰，易于理解和使用。关系模型一经提出，便迅速发展，成为实用性最强的产品。该数据库系统的主要特点是：概念单一化，数据及其数据间的联系都用关系来表示；以关系代数为理论基础；数据独立性强；数据库语言采用说明性语言，简化了编程难度。该数据库系统的发展如下：

1．1970 年，美国 IBM 公司 San Jose 研究实验室的研究员 E.F.Codd 发表了题为“大型共享数据库数据的关系模型”论文，提出了数据库的关系模型，开创了数据库关系方法和关系数据理论的研究，奠定了关系数据模型的理论基础。由于 E.F.Codd 的杰出工作，他于 1981 年获得了 ACM 图灵奖。

2．1974 年，IBM 公司 San Jose 研究实验室在 IBM System/370 系列机上研制成功关系数据库实验系统 System R（1974—1979 年）。这是世界上最早的、功能强大的关系数据库系统。1981 年 IBM 公司又宣布了具有 System R 全部特征的新的数据库软件产品 SQL/DS 的问世。

3．1980 年以后，RDBMS 的产品迅速推出，如 Oracle、Sybase、Informix、dBASE、Foxbase，FoxPro 等。

4．1990 年以后，RDBMS 产品的版本不断更新，功能更加强大，支持分布式数据库和

客户／服务器数据库以及客户／浏览器／服务器数据库等，同时实现了开放式网络环境下异质数据库的互联操作，以及在整个企业／行业范围内的 OLTP（在线事务处理)应用支持。

第三代数据库系统，即面向对象的数据库系统，它基于扩展的关系数据模型或面向对象的数据模型，是尚未成熟的一代数据库系统，其主要特点是支持包括数据、对象和知识的管理；在保持和继承第二代数据库系统的技术基础上引入面向对象技术；对其他系统开放，具有良好的可移植性、可连接性、可扩展性和可互操作性。

第三代数据库系统的典型代表包括 Servio 公司的 Gem Stone、OWTOS 公司的 ONTOS、Object Design 公司的 Object Stone、Objectivity 公司的 Objectivity/DB、Versant Object Technology 公司的 Versant 等。它们都支持严格面向对象的数据模型。与此同时，大多数商品化的关系数据库系统也对支持的数据模型进行了扩充，发展成了对象关系数据库系统（ORDBMS）.

### 1.1.3 数据库学科的研究领域

数据库学科的研究范围十分广泛，可以概括为 3 个主要领域：

1. 数据库管理系统软件的研制

DBMS 是数据库系统的基础。研制 DBMS 的基本目标是扩大功能、提高性能和可用性，从而提高用户的生产率。研制以 DBMS 为核心的一组相互联系的软件系统已成为当前数据库软件产品的开发方向，这些在 DBMS 基础上运行的软件系统有数据通信（DC）软件、表格软件（Forms）、数据字典、报表书写、图形系统等。

由于数据库应用领域的不断扩大，数据库不仅广泛应用于管理，而且已开始应用到工程设计、图形图像和声音等多介质处理、自动控制和计算机辅助设计等新的应用领域。这些新的应用领域所处理的数据和管理领域中数据的格式有极大的区别，如声音、图像等，称之为非格式化的数据，处理的要求也不大相同。因而，研究这些新的应用领域中的数据库方法、技术是一个新的课题。它不仅涉及应用系统的设计方法，而且涉及数据库系统的模型、实现技术等各种新的问题。面向对象的数据库系统、扩展的数据库系统、多介质数据库等研究方向的兴起就是基于这些新的需求和应用背景而产生的。

2. 数据库设计

在数据库管理系统的支持下，按照应用要求为某一部门或组织设计一个结构良好、使用方便、效率较高的数据库及其应用系统，这是数据库设计的主要含义。在这一领域内，主要的研究课题是数据库设计方法学和设计工具的探索。它包括数据库设计方法、设计工具和理论的研究，数据模型和数据建模的研究，计算机辅助数据库设计方法及其软件系统的研究，数据库设计规范和标准的研究等。

3. 数据库理论

数据库理论研究主要集中于关系的规范化理论及关系数据理论。近年来，随着人工智能与数据库的结合，数据库和逻辑、逻辑演绎和知识推理等理论研究，以及演绎数据库、知识库系统的研制都已成为新的研究方向。

### 1.1.4 学习本课程重点需要领会的问题

本书从如下几个方面探讨学习数据库应深入领会的问题。

1. 关系数据语言

包括：关系代数、元组关系演算和域关系演算。

2. 关系数据标准语言 SQL

包括：数据库与表的定义、数据库的连接查询、数据库的嵌套查询、库函数的应用、视图的定义与应用以及嵌入式 SQL 等。

3. 关系数据理论

包括：规范化理论基础、数据模型分析以及证明题方法。

4. 数据库设计

包括：概念模型的构造、逻辑模型的构造、数据库设计、运行与维护。

5. 数据库的控制与管理

包括：日志文件及其应用、并发控制与封锁技术。

6. 数据库技术的发展

包括：面向对象数据库系统、布式数据库系统、网络环境下的数据库体系、数据仓库与数据挖掘。

## 1.2 数 据 模 型

数据模型是现实世界数据特征的抽象，构造成一些相关的数据组织的集合。任何一种数据模型都是严格定义的概念的集合。这些概念必须能够精确地描述系统的静态特性、动态特性和完整性约束条件。因此数据模型通常都是由数据结构、数据操作和完整性约束 3 个要素组成。

数据库是某个企业、组织或部门所涉及的数据的一个综合，它不仅要反映数据本身的内容，而且要反映数据之间的联系。由于计算机不可能直接处理现实世界中的具体事物，所以读者必须事先把具体事物转换成计算机能够处理的数据。在数据库中用数据模型这个工具来抽象、表示和处理现实世界中的数据和信息。通俗地讲数据模型就是现实世界的模拟。

### 1.2.1 数据模型的基本概念

现有的数据库系统都是基于某种数据模型的。数据模型是数据库系统的数学形式框架，

是用来描述数据的一组概念和定义。它包括以下方面的内容：

- 数据的静态特征，它包括对数据结构和数据间联系的描述。
- 数据的动态特征，它是一组定义在数据上的操作，包括操作的含义、操作符、运算规则及其语言等。
- 数据的完整性约束，这是一组规则，数据库中的数据必须满足这组规则。

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。一种数据模型要很好地满足这三方面的要求，在目前尚很困难。在数据库系统中针对不同的使用对象和应用目的，采用不同的数据模型。

不同的数据模型实际上是提供模型化数据和信息的不同工具。根据模型应用的目的不同，可以将这些模型划分为两类，它们分别属于两个不同的层次。第一类模型是概念模型，也称信息模型，它是按用户的观点对数据和信息建模。另一类模型是结构模型，主要包括网状模型、层次模型、关系模型和面向对象模型等，它是按计算机系统的观点对数据建模。

数据模型是数据库系统的核心和基础。

1．数据模型三要素

（1）数据结构

数据结构用于描述系统的静态特性。

数据结构是所研究的对象类型（object type）的集合。这些对象是数据库的组成成分，它们包括两类，一类是与数据类型、内容、性质有关的对象，如网状模型中的数据项、记录，关系模型中的域、属性、关系等；一类是与数据之间联系有关的对象，如网状模型中的系型（set type）。

数据结构是一个数据模型性质最重要的方面。因此，在数据库系统中，通常按照其数据结构的类型来命名数据模型。例如，层次结构、网状结构和关系结构的数据模型分别命名为层次模型、网状模型和关系模型。

（2）数据操作

数据操作用于描述系统的动态特性。

数据操作是指对数据库中各种对象（型）的实例（值）允许执行的操作的集合，包括操作及有关的操作规则。数据库主要有检索和更新（包括插入、删除、修改）两大类操作。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。

（3）数据的约束条件

数据的约束条件是一组完整性规则的集合。完整性规则是给定的数据模型中数据及其联系所具有的制约和存储规则，用以限定符合数据模型的数据库状态以及状态的变化，以保证数据的正确性、有效性和相容性。

数据模型应该反映和规定本数据模型必须遵守的通用的基本的完整性约束条件。例如，在关系模型中，任何关系必须满足实体完整性和参照完整性两个条件。

此外，数据模型还应该提供定义完整性约束条件的机制，以反映具体应用所涉及的数

据必须遵守的特定的语义约束条件。

2. 主要数据模型

数据模型包括概念数据模型和结构数据模型，关于概念数据模型在后面数据库设计中将详细叙述，现在先介绍结构数据模型。

不同的数据模型具有不同的数据结构形式。目前最常用的数据模型有层次模型（Hierarchical model）、网状模型（Network model）、关系模型(Relational model）和面向对象模型(Object oriented model），其中层次模型和网状模型统称为非关系模型。非关系模型的数据库系统在 20 世纪 70 年代至 80 年代初非常流行，在数据库系统产品中占据了主导地位，现在已逐渐被关系模型的数据库系统所取代。

20世纪 80年代以来，面向对象的方法和技术在计算机各个领域，包括程序设计语言、软件工程、信息系统设计、计算机硬件设计等各方面都产生了深远的影响，也促进了数据库中面向对象数据模型的研究和发展。

（1）层次数据模型

层次模型是数据库系统中最早出现的数据模型，它用树形结构表示各类实体以及实体间的联系。现实世界中许多实体之间的联系本来就呈现出一种很自然的层次关系，如行政机构、家族关系等。层次模型数据库系统的典型代表是 IBM 公司的 IMS（Information Management Systems）数据库管理系统，这是一个曾经广泛使用的数据库管理系统。

（2）网状数据模型

在现实世界中实体型间的联系更多的是非层次关系，用层次模型表示非树形结构是很不直接的，而网状模型采用网状模型作为数据的组织方式，则可以克服这一弊病。网状数据模型的典型代表是 DBTG 系统，也称 CODASYL 系统。

层次数据模型和网状数据模型都是早期的数据库数据模型。数据库系统与文件系统的主要区别就在于不仅定义数据的存储而且还定义存储数据之间的联系，所谓“层次”和“网状”就是指这种联系的方式。

图1-1、图 1-2表示的是这两种数据模型的实例。

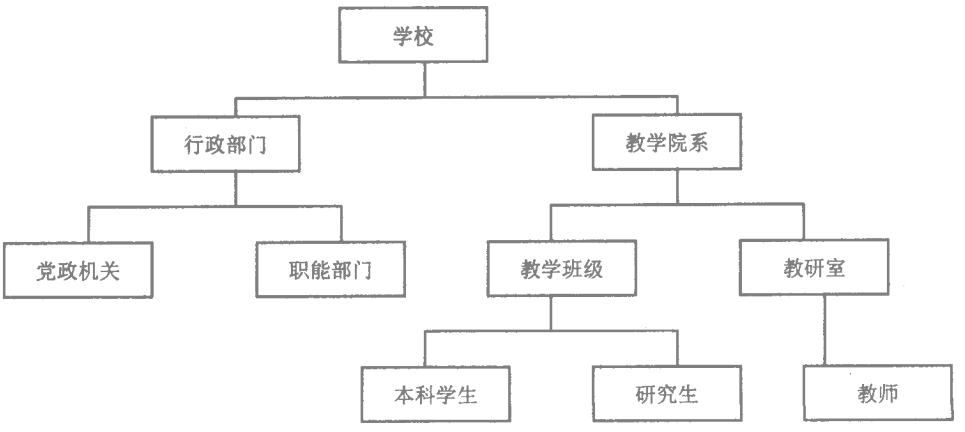


图 1-1 层次数据模型实例

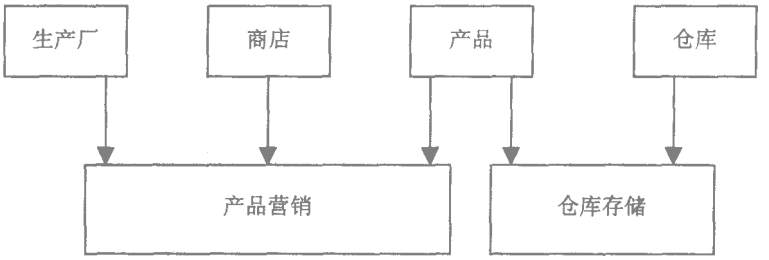


图 1-2　网状数据模型实例

（3）关系数据模型

与以往的模型不同，关系模型是建立在严格的数学理论基础上的。在用户看来，一个关系模型的逻辑结构是一张二维表，它由行和列组成。

关系模型是目前最重要的一种模型。关系数据库系统采用关系模型作为数据的组织方式。美国 IBM 公司的研究员 E.F.Codd 于 1970 年发表题为“大型共享系统的关系数据库的关系模型”的论文，首次提出了数据库系统的关系模型。20 世纪 80 年代以来，计算机厂商新推出的数据库管理系统（DBMS）几乎都支持关系模型，非关系系统的产品也大都加上了关系接口。数据库领域当前的研究工作都是以关系方法为基础。

在关系数据模型中不仅其数据结构——二维表，而且在数据操作中也具有“非过程化”的特点，因此易学易用，所以成为当前数据库应用系统的主流。

从专题 2 开始，本书将详细介绍有关关系数据模型的概念、方法、理论和应用。

（4）面向对象的数据模型

面向对象的数据模型是近年来出现的一种新的数据模型，它比前 3 种数据模型具有对现实世界更广泛、更深入、更丰富的表达能力。所谓对象是对现实世界事物的高度抽象，每个对象是状态和行为的封装。对象的状态是属性的集合，行为是在该对象上操作方法的集合。因此面向对象的模型不仅可以处理各种复杂多样的数据结构，而且具有数据与行为相结合的特点。目前面向对象的方法已经逐渐成为系统开发、设计的全新思路。但这种方法相对复杂，实现起来有一定难度。

在当前信息处理技术中，关系数据模型仍然是数据库数据模型的主流，即使使用面向对象的模型也往往采用关系数据模型的方法和工具。因此，从 1.2.2 节起本书将对关系数据模型的有关概念、方法、理论和应用的问题做详细的介绍。

## 1.2.2　关系数据模型的特点

关系数据库应用数学方法来处理数据库数据，是目前各类数据库中最重要、最流行使用、最广泛的数据库系统。20 世纪 70 年代以后开发的数据库管理系统产品几乎都是基于关系模型的。在数据库发展的历史上，最重要的成就是关系模型。

关系数据库系统与非关系数据库系统的区别是，关系系统只有“表”这种数据结构；

而非关系数据库系统还有其他数据结构，对这些数据结构有其他的操作。

关系数据库系统是支持关系模型的数据库系统。关系模型由关系数据结构、关系操作集合和关系完整性约束 3 部分组成。

1．单一的数据结构——关系

关系模型的数据结构是单一的。在关系模型中，现实世界的实体以及实体间的各种联系均用关系来表示，也就是说，关系（二维表）不仅表示数据的存储，也表示数据之间的联系。在用户看来，关系模型中数据的逻辑结构是一张二维表。

在组成关系的二维表中，表的行称为元组，列称为属性，在这些属性中，如果某个或某些属性的值可以惟一确定元组，就称这个（或这些）属性（组）为关系的码，组成码的属性称为主属性。

2．关系操作

关系操作采用集合操作方式，即操作的对象和结构都是集合。关系模型给出了关系操作能力。关系模型中常用的关系操作包括：选择（ Select）、投影( Project）、连 接（Join）. 除（Divide）、并（Union）、交( Intersction）、差( Difference） 等查询( Query)操作和插 入( Insert）、删 除( Delete）、修改( Update 操作等更新操作两部分。其中查询是最基本的操作。

关系操作的特点是集合操作方式，即操作的对象和结果都是集合。这种操作方式也称为一次一集合（set-at-time）的方式。相应地，非关系数据模式的数据操作方式则为一次一记 录( record-at-time）的方式。

关系模型中的数据操作是集合操作，操作对象和操作结果都是关系，即若干元组的集合，而不像非关系模型中那样是单记录的操作方式。另一方面，关系模型把存取路径向用户隐蔽起来，用户只要指出“ 干什么 ”或“ 找什么 ”，不必详细说明“ 怎么干 ”或“ 怎么找 ”，从而大大地提高了数据的独立性，提高了用户的生产率。

3．关系的 3 类完整性约束

关系模型提供了丰富的完整性控制机制，允许定义 3 类完整性约束：实体完整性、参照完整性和用户定义的完整性。其中实体完整性和参照完整性是关系模型必须满足的完整性约束条件，应该由关系系统自动支持。用户定义完整性是应用领域需要遵循的约束条件，体现了具体领域中的语义约束。

# 专题 2　关系数据语言

数据库语言的作用是为用户提供对数据库数据进行定义、查询、更新等处理方法的工具，不同的数据模型为用户提供的数据库语言也是不同的。关系数据语言的产生是与关系数据库系统的数据结构和数据处理方法密切相关的。

早期的关系操作通常用代数方式或逻辑方式来表示，分别称为关系代数和关系演算。关系代数是用对关系的运算来表达查询要求的方式。关系演算是用谓词来表达查询要求的方式。关系演算又可按谓词变元的处理对象是元组变量还是域变量分为元组关系演算和域关系演算。可以证明，关系代数、元组关系演算和域关系演算 3 种语言在表达能力上是完全等价的。

关系代数、元组关系演算和域关系演算都是抽象的查询语言，这些抽象的语言与具体的 DBMS 中实现的实际语言并不完全一样。但它们能用作评估实际系统中查询语言能力的标准或基础。实际的查询语言除了提供关系代数或关系演算的功能外，还提供了许多附加功能，例如集函数、视图定义、算术运算、嵌入式等。

关系数据语言是一种高度非过程化的语言，用户不必请求 DBA 为其建立特殊的存取路径，存取路径的选择由 DBMS 的优化机制来完成；用户查询时仅仅需要对查询对象所应满足的条件进行描述，而无须写出如何达到查询目标的过程就可以完成数据操作。

另外,还有一种介于关系代数和关系演算之间的语言 SQL（Structurel Query Language）。SQL 不仅具有丰富的查询功能，而且还具有数据定义和数据控制功能，是集数据定义、数据查询、数据更新、数据控制、视图定义、向主语言嵌入等功能于一体的关系数据语言。它充分体现了关系数据语言的特点和优点，是关系数据库的标准语言。

因此，关系数据语言可以分为 3 类：

- 关系数据语言
  - 关系代数语言　例如　ISBL
  - 关系演算语言
    - 元组关系演算语言　例如　APLHA，QUEL
    - 域关系演算语言　例如　QBE
  - 具有关系代数和关系演算双重特点的语言　例如　SQL

这些关系数据语言的共同特点是，具有完备的表达能力，是非过程化的集合操作语言，功能强，能够嵌入高级语言中使用。

# 2.1 关系数据模型的定义和性质

在关系模型中，无论是实体还是实体之间的联系都是由单一的结构类型即关系（表）来表示。前面已经介绍了关系数据模型及有关的特点。关系模型建立在集合代数的基础上，本书首先从集合论角度给出关系数据结构的形式化定义。

1. 关系的形式化定义

首先介绍几个概念。

①域（Domain）

定义 2.1 域是一组有相同数据类型的值的集合。

例如，自然数、整数、实数、长度小于 25 字节的字符串集合、{0，1}、大于等于 0 且小于 10 的正整数等，都可以是域。

②笛卡尔积（Cartesian Product）

定义 2.2 给定一组域 $D_1$，$D_2$，…，$D_n$，其中可以有相同的域。$D_1$，$D_2$，…，$D_n$ 的笛卡尔积为：

$D_1\times D_2\times\cdots\times D_n=\{(d_1, d_2, \cdots, d_n)\mid d_i\in D_i, i=1,2,\cdots,n\}$，其中每一个元素($d_1, d_2, \cdots, d_n$)叫作一个 n 元组（n-tuple）或简称元组（Tuple）。

若 $D_i$（i=1,2,…，n）为有限集，其基数（Cardinal number）为 $m_i$（i=1,2,…，n），则 $D_1\times D_2\times\cdots\times D_n$ 的基数 M 为：

$$M=\prod_{i=1}^{n} m_i$$

笛卡尔积可表示为一个二维表。表中的每行对应一个元组，每列对应一个域。例如给出 3 个域：

$D_1$=导师集合={李清,刘涛}

$D_2$=专业集合={计算机专业,管理工程专业}

$D_3$=研究生集合={李华，刘颖，杨敏}

则 $D_1$，$D_2$，$D_3$ 的笛卡尔积为：

$D_1\times D_2\times D_3=\{$（李清，计算机专业，李华），（李清，计算机专业，刘颖），
（李清，计算机专业，杨敏），（李清，信息专业，李华），
（李清，信息专业，刘颖），（李清，信息专业，杨敏），
（刘涛，计算机专业，李华），（刘涛，计算机专业，刘颖），
（刘涛，计算机专业，杨敏），（刘涛，信息专业，李华），
（刘涛，信息专业，刘颖），（刘涛，信息专业，杨敏）}

其中，（李清，计算机专业，李华）、（李清，计算机专业，刘颖）等都是元组。李清、计算机专业、李华、刘颖等都是分量。

该笛卡尔积的基数为 $2\times2\times3=12$，也就是说，$D_1\times D_2\times D_3$共有 $2\times2\times3=12$ 个元组。这 12 个元组可列成一张二维表，如表 2-1 所示。

表 2-1 $D_1$，$D_2$，$D_3$的笛卡尔积

| 导　师 | 专　业 | 研 究 生 |
|---|---|---|
| 李清 | 计算机专业 | 李华 |
| 李清 | 计算机专业 | 刘颖 |
| 李清 | 计算机专业 | 杨敏 |
| 李清 | 管理工程专业 | 李华 |
| 李清 | 管理工程专业 | 刘颖 |
| 李清 | 管理工程专业 | 杨敏 |
| 刘涛 | 计算机专业 | 李华 |
| 刘涛 | 计算机专业 | 刘颖 |
| 刘涛 | 计算机专业 | 杨敏 |
| 刘涛 | 管理工程专业 | 李华 |
| 刘涛 | 管理工程专业 | 刘颖 |
| 刘涛 | 管理工程专业 | 杨敏 |

③ 关 系（Relation）

**定义 2.3** 给定一组域 $D_1$，$D_2$，…，$D_n$，在这些域上笛卡尔积 $D_1\times D_2\times\cdots\times D_n$ 上的一个子集，称为一个关系，表示为

$$R(D_1, D_2, \cdots, D_n)$$

这里 R 表示关系的名字，n 是关系的目或度（Degree）。

关系中的每个元素是关系中的元组，通常用 t 表示。

当 n=1 时，称该关系为单元（目）关系（Unary relation）。

当 n=2 时，称该关系为二元（目）关系（Binary relation）。

关系是笛卡尔积的有限子集，所以关系也是一个二维表，表的每行对应一个元组，每列对应一个域。由于域可以相同，为了加以区分，必须对每列起一个名字，称为属性（Attribute）。n 目关系必有 n 个属性。

若关系中的某一属性（或属性组）的值能惟一地标识一个元组，则称该属性组为关系的候选码（Candidate key）。

若一个关系有多个候选码，则选定其中一个作为主码（Primary key）。包含在任何一个候选码中的属性称为主属性（Prime attribute），不包含在任何候选码中的属性称为非码属性（Non-key attribute）。在最极端的情况下，关系模式的全部属性组都是这个关系模式的候选码，称为全码（All-key）。

例如，可以在表 2-1 的笛卡尔积中取出一个子集来构造一个关系。由于一个研究生只师从于某一个导师，学习某一个专业，所以笛卡尔积的许多元组是无实际意义的。从中取出有实际意义的元组来构造关系，该关系的名字为 SAP，属性名就取域名，即导师，专业

和研究生。则这个关系可以表示为：

SAP （导师，专业，研究生）

假设导师与专业是一对一的，即一个导师只有一个专业，导师与研究生是一对多，即一个导师可以带多名研究生，而一名研究生却只有一个导师，这样 SAP 关系可以包含 3 个元组，如表 2-2 所示。

表 2-2 SAP 关系

| 导师 | 专业 | 研究生 |
|---|---|---|
| 李清 | 计算机 | 李华 |
| 李清 | 计算机 | 刘颖 |
| 刘涛 | 计算机 | 杨敏 |

假设研究生不重名，则研究生属性的每一个值都惟一地标识了一个元组，因此可以作为 SAP 关系的主码。

关系可以有 3 种类型：基本关系（通常又称为基本表或基表）、查询表和视图表。基本表是实际存在的表，它是实际存储数据的逻辑表示，查询表是查询结果对应的表，视图表则是基本表或其他视图表导出的表，是虚表，不对应实际存储的数据。

2．关系的性质

基本关系具有以下 6 个性质：

①列是同质的（Homogeneous），即每一列中的分量为同一类型的数据，来自同一个域。

②不同的列可出自同一个域，称其中的每列为一个属性，不同的属性要给予不同的属性名。

例如在上面的例子里，也可以只给出两个域：

姓名：{李清,刘涛,李华,刘颖,杨敏}

专业：{计算机专业,管理科学与工程专业}

SAP 关系的导师属性和研究生属性都从域中取值。为了避免混淆，必须给这两个属性取不同的属性名，而不能直接使用域名。例如定义导师属性名为导师，研究生属性名为研究生。

③列的顺序无所谓，即列的次序可以任意交换。

由于列的顺序是无关紧要的，列顺序不同的关系在逻辑上是同一个集合。

④任意两个元组不能完全相同。

⑤行的顺序无所谓，即行的次序可以任意交换。

由于行的顺序是无关紧要的，行顺序不同的关系在逻辑上是同一个集合。

⑥ 分量必须取原子值，即每一个分量都必须是不可分的基本数据项。

由于关系是域上笛卡尔积的子集，而域是值的集合，因此作为行、列交叉点的分量必然具有确定的、具体的值（不能是组合值，也不能是表）。

关系模型要求关系必须是规范化的，即要求关系模式必须满足一定的规范条件。这些规范条件中最基本的一条就是，关系的每一个分量必须是一个不可分的基本数据项。规范化的关系简称为范式（Norma form）。例如，表 2-3 虽然很好地表达了导师与研究生之间的一对多的关系，但由于研究生分量取了两个值，不符合规范化的要求，因此这样的关系在数据库中是不允许的。

表 2-3 非规范化关系

| 导 师 | 专 业 | 研 究 生 | |
|---|---|---|---|
| | | 研究生 1 | 研究生 2 |
| 李清 | 管理工程专业 | 李华 | 刘颖 |
| 刘涛 | 管理工程专业 | 杨敏 | |

3. 关系模式

关系数据库中，关系模式是型，关系是值。关系模式是对关系的描述，它包括如下方面：

首先，关系实质上是一张二维表，表的每一行为一个元组，每一列为一个属性。一个元组就是该关系所涉及的属性集的笛卡尔积的一个元素。关系是元组的集合，因此关系模式必须指出这个元组集合的结构，即它由哪些构成，这些属性来自哪些域，以及属性与域之间的映像关系。

其次，一个关系通常是由赋予它的元组语义来确定的。元组语义实质上是一个 m 目谓词，或者说凡符合元组语义的那部分元素的全体就构成了该关系模式的关系。

现实世界随着时间不断地变化，因而在不同的时刻，关系模式的关系也会有变化。但是，现实世界的许多已有事实限定了关系模式所有可能的关系必须满足一定的完整性约束条件。这些约束或者通过对属性值间的相互关联反映出来，关系模式应当刻划出这些完整性约束条件。

因此一个关系模式应当有一个五元组。

定义 2.4 关系的描述称为关系模式（Relation Schema），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 为属性间数据的依赖关系集合。

属性间的数据依赖将在后续章节讨论，本章中关系模式仅涉及关系名、属性名、域名、属性向域的映像 4 部分。

例如，在上面例子中，由于导师和研究生出自同一个域，所以要取不同的属性名，并在模式中定义属性向域的映像，即说明它们分别出自哪个域，如：

dom（导师）=dom（研究生）=姓名

关系模式通常可以简记为：

R（U）或 R（$A_1$，$A_2$，…，$A_n$）

其中，R 为关系名，$A_1$，$A_2$，…，$A_n$ 为属性名。而域名及属性向域的映像常常直接说明为属性的类型及长度。

关系是关系模式在某一个时刻的状态或内容。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。但实际上，人们常常把关系模式和关系都称为关系。

4. 关系数据库

在一个给定的应用领域中，反映这个应用的所有关系的集合构成一个关系数据库。

关系数据库也存在型和值之分，关系数据库的型也称为关系数据库模式，是对关系数据库的描述，它包括若干域的定义以及在这些域上定义的若干关系模式。关系数据库的值是这些关系模式在某一时刻对应的关系的集合，通常就称为关系数据库。

要特别注意区分数据库与表的概念，相关的表的集合称为数据库。有时为了方便，常常把每个表简称为数据库，但在概念上一定要注意两者的区别。

## 2.2 关系数据语言之一—— 关系代数

关系代数是一种抽象的查询语言，是关系数据操纵语言的一种传统表达方式，它是用对关系的运算来表达查询的。

任何一种运算都是将一定的运算符作用于一定的运算对象上，得到预期的运算结果。所以运算对象、运算符、运算结果是运算的三大要素。

关系代数的运算对象是关系，运算结果也是关系。关系代数用到的运算符包括：集合运算符、专门的关系运算符、比较运算符和逻辑运算符，如表 2-4 所示。

比较运算符和逻辑运算符是用来辅助专门的关系运算符进行操作的，所以关系代数的运算按运算符的不同主要分为传统的集合运算和专门的关系运算两类。其中传统的集合运算将关系看成元组的集合，其运算是从关系的“水平”方向，即行的角度来进行，而专门的关系运算不仅涉及行，而且还涉及列。

表 2-4 关系代数运算符

| 运算符 | | 含义 |
|---|---|---|
| 集合运算符 | ∪ | 并 |
| | − | 差 |
| | ∩ | 交 |
| 专门的关系运算符 | × | 广义笛卡尔积 |
| | σ | 选择 |
| | Π | 投影 |
| | ⋈ | 连接 |
| | ÷ | 除 |

续表

| 运 算 符 | | 含 义 |
|---|---|---|
| 比较运算符 | > | 大于 |
| | ≥ | 大于等于 |
| | < | 小于 |
| | ≤ | 小于等于 |
| | = | 等于 |
| | ≠ | 不等于 |
| 逻辑运算符 | ¬ | 非 |
| | ∧ | 与 |
| | ∨ | 或 |

为了叙述方便，引入一些记号。

（1）设关系模式为 $R(A_1, A_2, \cdots, A_n)$，它的一个关系设为 R。$t \in R$ 表示 t 是 R 的一个元组。$t[A_i]$ 则表示元组 t 中相应于属性 $A_i$ 的一个分量。

（2）若 $A=\{A_{i1}, A_{i2}, \cdots, A_{ik}\}$，其中 $A_{i1}, A_{i2}, \cdots, A_{ik}$ 是 $A_1, A_2, \cdots, A_n$ 中的一部分，则 A 称为属性列或域列。$\bar{A}$ 则表示 $\{A_1, A_2, \cdots, A_n\}$ 中去掉 $\{A_{i1}, A_{i2}, \cdots, A_{ik}\}$ 后剩余的属性组。$t[A]=(t[A_{i1}], t[A_{i2}], \cdots, t[A_{ik}])$ 表示元组 t 在属性列 A 上诸分量的集合。

（3）R 为 n 目关系，S 为 m 目关系。$t_r \in R$，$t_s \in S$。$t_r t_s$ 称为元组的连串（Concatenation）。它是一个（n+m）列的元组，前 n 个分量为 R 中的一个 n 元组，后 m 个分量为 S 中的一个 m 元组。

（4）给定一个关系 R（X，Z），X 和 Z 为属性组。定义当 $t[X]=x$ 时，x 在 R 中的象集（images set）为：

$$Z_x=\{t[z] \mid t \in R \wedge t[X]=x\}$$

它表示 R 中属性组 X 上值为 x 的诸元组在 Z 上分量值的集合。

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、广义笛卡尔积 4 种运算。

（1）并（union）

设关系 R 和关系 S 具有相同的目 n（即两个关系都有 n 个属性），且相应的属性取自同一个域，则关系 R 与关系 S 的并由属于 R 或属于 S 的元组组成。其结果仍为 n 目关系。记作：

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

（2）差（difference）

设关系 R 和关系 S 具有相同的目 n，且相应的属性取自同一个域，则关系 R 与关系 S 的差由属于 R 而不属于 S 的所有元组组成。其关系仍为 n 目关系。记作：

$$R-S=\{t \mid t \in R \wedge \neg t \in S\}$$

（3）交（intersection）

设关系 R 和关系 S 具有相同的目 n，且相应的属性取自同一个域，则关系 R 与关系 S