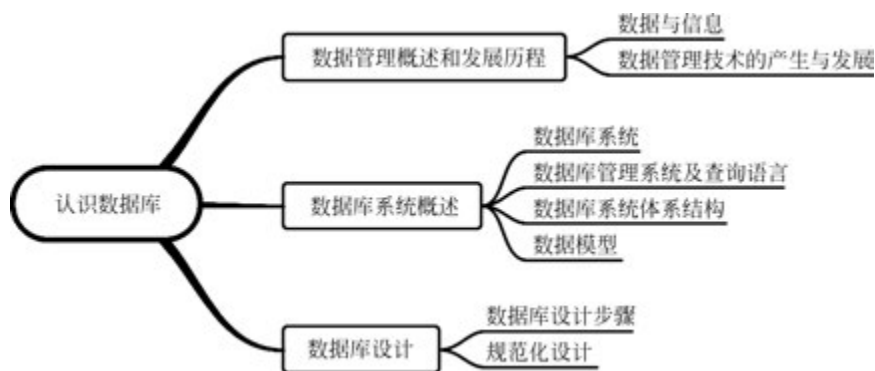


单元一 认识数据库

思维导图



内容导读

数据库技术是现代信息技术的重要组成部分,是计算机数据处理与信息管理的核心。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组织和存储的问题。数据库的建设规模、信息量的大小和使用频度等已经成为衡量社会信息化程度的重要标志。为了更好地掌握数据库技术,我们需要理解数据库的相关概念,掌握数据库系统的基本组成以及数据库的设计思路。

学习目标

- 了解数据管理技术的发展。
- 了解数据库系统的构成。
- 掌握 DBMS 的主要功能(DDL、DML、DQL、DCL)。
- 了解数据库的三级模式结构(外模式、模式、内模式)。
- 了解数据模型的相关概念。
- 掌握数据库设计的基本方法和步骤。

素质目标

在数字化浪潮中,数据库技术是国家信息安全的战略支撑。学习数据库时,需将个人发展融入国家需求,强化数据安全意识与自主可控能力。例如,在关键系统设计中,应优先采用国产数据库技术,减少对外依赖。同时注重数据加密与权限管控,保障敏感数据安全。这不仅是技术实践,更是维护国家数字主权的重要使命。未来从业者应以科技报国为己任,在设计中践行自主可控理念,助力我国信息产业高质量发展。

1.1 数据管理概述和发展历程

1.1.1 数据与信息

1. 数据

数据(data)是用来描述事物的符号记录,是信息的具体表现形式。在计算机系统中数据是以二进制 0、1 的形式表示。数据是数据库中存储的基本对象,它不仅包括数字、文字形式,而且包括图形、图像、声音、动画等多媒体数据。数据本身不包含特定意义,只是现实的反映;信息则赋予数据意义,可以用来指导人们的行动或决策。

2. 信息

信息(information)是客观事物经过加工处理后,以适合人们理解和使用的方式呈现出来的知识。信息可以用于决策、执行任务、进行预测等。假设有一个包含几千个数字的列表,这些数字就是数据。如果这个列表是某城市每个月的气温记录,那么这些数字就可以被处理和分析,得出每个月的平均气温、最高温度、最低温度等信息。这些信息对于了解气候变化、制订城市规划等非常有用。

在现代社会,信息和数据是驱动创新 and 发展的关键资源。从商业智能到政府决策,从科学研究到日常生活,信息和数据都扮演着越来越重要的角色。

3. 数据管理

数据管理是对数据进行收集、整理、组织、存储、分析、检索、分类、加工、打印报表、输出等一系列活动。数据管理是数据处理的基本环节,有效的数据管理不仅能够提高组织的决策质量,还能增强业务流程的效率,同时保障数据的安全性和合法性。

1.1.2 数据管理技术的产生与发展

数据管理技术是指对数据进行有效的分类、组织、编码、存储、检索和维护的一系列方法和技术。数据管理技术经历了人工管理、文件系统和数据库系统三个阶段,并逐步实现网络化、智能化和集成化。

1. 人工管理阶段

20 世纪 50 年代中期以前,计算机主要用于科学计算。那时没有存储数据的磁盘等硬件,也没有专门管理数据的软件,数据由计算或处理它的程序自行携带,程序设计依赖于数据表示。

人工管理阶段的特点如下。



微课 1-1 数据管理技术的产生和发展

- 数据不易长期保存。
- 应用程序管理数据。
- 数据不能共享。

2. 文件系统阶段

20 世纪 50 年代后期到 60 年代中期,计算机不仅用于科学计算,还大量用于数据管理。这一时期,不仅出现了磁鼓和磁盘等存储硬件,还出现了高级语言和操作系统等软件。程序和数据有了一定的独立性,并且有了程序文件和数据文件。但此时期的数据文件是为某一特定的应用服务的,修改了数据的逻辑结构就要修改相应的程序,反之亦然。文件系统阶段的特点如下。

- 数据可以长期保存。
- 由文件系统管理数据。
- 数据冗余大,共享性差。
- 数据独立性差。

3. 数据库系统阶段

从 20 世纪 60 年代后期开始,计算机信息管理的规模越来越大,随着网络的发展,数据共享的需求也日益增加,计算机软硬件功能越来越强,从而发展了数据库技术。特别是关系型数据库技术已经非常成熟,并且广泛应用于企事业各部门的信息管理中。数据库系统阶段的主要特点如下。

- 数据结构化: 用特定的数据模型来表示事物及事物之间的联系。
- 数据共享性高: 减少数据冗余,减少更新异常。
- 数据独立性强: 程序和数据相对独立。
- 数据粒度小: 粒度单位是记录中的数据项,粒度越小则处理速度就越快、越方便。
- 统一的管理和控制: 数据定义、操作和控制由数据库管理系统(database management system,DBMS)统一管理和控制。例如,MySQL、Access、Oracle 和 SQL Server 等数据库管理系统软件。
- 独立的数据操作界面: DBMS 提供管理平台,通过命令或界面(菜单、工具栏、对话框等)对数据库进行访问和处理。

4. 数据管理技术的发展

随着数据规模的爆炸式增长、计算机硬件的持续进步以及数据库应用的日益广泛,数据库技术正逐步迈向一个数据结构多样化、存储架构异构化的新时代。数字化快速发展,数据量和应用场景复杂度剧增,传统数据库技术难以满足需求,前沿技术顺势而生。

为有效应对这些挑战,数据库技术领域不断推陈出新,涌现出众多前沿技术,举例如下。

(1) Web 数据库技术: 旨在支持互联网应用中的动态数据交互,提升用户体验。

(2) 并行数据库技术: 通过采用大规模并行处理或对称多处理架构,显著提升数据库的处理能力和扩展性。

(3) 数据仓库与联机分析处理技术：专为高效数据分析和决策支持设计，助力企业洞察数据价值。

(4) 数据挖掘与商业智能技术：从海量数据中挖掘有价值的信息，为企业决策提供有力支持。

(5) 内容管理技术：专注于高效管理和存储非结构化数据，满足多样化数据管理需求。

(6) 海量数据管理技术：应对大规模数据的存储、查询和分析挑战，确保数据的高效处理。

(7) 云计算技术：提供弹性伸缩、高可用性和低成本的数据服务，推动数据库向云端迁移。

(8) 新型数据库架构：如时间序列数据库、图数据库、区块链数据库等，针对特定应用场景提供优化解决方案。

此外，人工智能与数据库的融合也成为当前的研究热点。例如，DeepSeek 等新兴 AI 技术正在探索将智能检索、推理与缓存能力集成到数据库中，以提升查询效果和降低使用成本。这种融合不仅提高了数据库的自动化和智能化水平，还为未来数据库的应用场景带来了更多可能性。

这些技术的发展展示了数据库领域在应对大数据、云计算、人工智能等新兴技术挑战时的创新能力和适应性，为未来数据库技术的发展指明了方向。

1.2 数据库系统概述

1.2.1 数据库系统

1. 数据库系统的含义

数据库系统(database system, DBS)是指在计算机系统中引入数据库后的系统,数据、数据库、数据库管理系统与操作数据库的应用开发工具、应用程序以及与数据库有关人员一起构成了一个完整的数据库系统,如图 1-1 所示。



微课 1-2 数据库系统

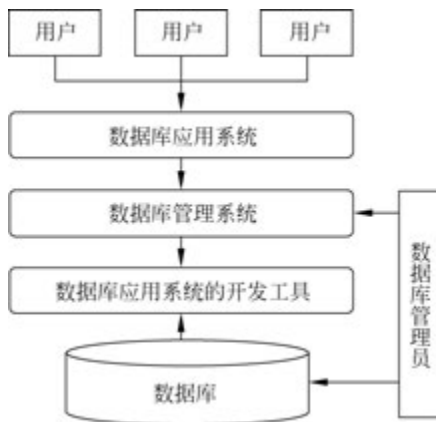


图 1-1 数据库系统示意图

2. 基本组成

(1) 数据库(database,DB)。数据库是指长期保存在计算机的存储设备上,按照某种数据模型组织起来的、可以被各种用户或应用共享的数据集合。

(2) 数据库管理系统。数据库管理系统是位于操作系统和用户之间的一种操作与管理数据库的大型软件,是数据库系统的核心。数据库管理系统主要负责建立、使用和维护数据库,对数据库进行统一的管理和控制,以保证数据库的安全性和完整性。

(3) 数据库应用系统(database application system,DBAS)。数据库应用系统是指利用数据库技术开发的用于解决特定领域问题的软件系统。这类系统通常会为终端用户打造可视化的操作界面,方便用户开展日常的数据处理相关工作。例如,企业的财务管理系统、人力资源管理系统、学校的选课系统,图书馆的图书管理系统,以及民航和铁路的售票系统等。

(4) 数据库应用系统的开发工具(development tools)。数据库应用系统的开发工具涵盖以下核心类别。

- 数据库设计工具:包括 PowerDesigner、ERWin 和 Office Visio 等,这些工具用于创建和优化数据库模型。
- 结构化查询语言(SQL):由数据库管理系统提供,用于访问和管理数据库。
- 数据库应用程序开发工具:涵盖 Java、Python、C# 和 VB.NET 等编程语言,用于开发数据库应用程序。
- 动态网页技术:包括 JSP、ASP.NET 和 PHP 等,这些技术用于构建交互式的 Web 应用。
- 数据访问接口:如 JDBC、ADO.NET 和 ODBC 等,它们允许应用程序与数据库进行通信。

这些工具共同支撑数据库应用系统从设计到实现的整个开发过程。

(5) 数据库用户(database user)。数据库系统中通常包括四种类型的用户,分别是数据库管理员、数据库设计师、应用程序员和终端用户。

- 数据库管理员(database administrator,DBA):DBA 负责整个数据库系统的建立、管理、运行、维护和监控等系统性工作,以及用户登记、存取数据的权限分配等服务性工作。
- 数据库设计师(database designer,DBD):根据数据库设计理论与方法,借助数据建模工具建立概念数据模型和逻辑数据模型,搜集和整理数据,使用数据库管理系统的数据库定义语言或操作界面建立相应的数据库。
- 应用程序员(application programmer):根据数据库设计和用户的功能需求,利用 Java、C# 等程序设计语言开发出功能完善、操作简便、满足用户需求的数据库应用系统,供终端用户使用。
- 终端用户(end user):使用数据库应用系统的最广泛群体,是数据库为之服务的对象。例如,学校学籍管理员、银行出纳员、窗口售票员、企业销售员、物资管理员等都是相应数据库系统的终端用户。

1.2.2 数据库管理系统及查询语言

1. 数据库管理系统的功能

DBMS 提供多种功能,可使多个应用程序和用户用不同的方法定义和操作数据,维护数据的完整性和安全性,进行多用户下的并发控制和恢复数据库等。DBMS 主要有以下几个方面的功能。

(1) 数据定义功能。DBMS 提供数据定义功能以实现定义数据的完整性约束、保密限制等约束。

(2) 数据操作功能。DBMS 提供数据操作功能以实现数据的插入、删除、更新、查询等操作。

(3) 数据控制功能。DBMS 提供数据控制功能以实现数据库的运行控制。对数据库的保护主要通过以下四个方面实现。

① 数据完整性的控制。保证数据库中数据及语义的正确性和有效性,防止任何造成数据错误的操作。

② 数据安全性的控制。防止未经授权的用户存取数据库中的数据,以避免数据的泄露、更改或破坏。

③ 数据库的恢复。在数据库一旦被破坏或数据不正确时,系统有能力把数据库恢复到正确的状态。

④ 数据库的并发控制。在多个用户同时对同一个数据进行操作时,系统应能加以控制和协调,防止破坏数据库中的数据。

2. 常见的关系数据库产品

基于关系模型的数据库管理系统现已日趋完善,商品化的关系型数据库管理系统(relational database management system,RDBMS)软件已经广泛应用于各行各业。

主要的 RDBMS 产品有瑞典的 MySQL AB 公司开发的 MySQL(目前属于甲骨文公司)、甲骨文公司的 Oracle、IBM 公司的 DB2 和微软的 SQL Server 等。本书选用的关系型数据库管理系统为 MySQL,MySQL 是当下非常流行的开源数据库管理系统,它具有快速、可靠和易于使用的特点。MySQL 具有跨平台的特性,可以在 Windows、UNIX、Linux 和 macOS 等平台使用。由于其开源免费、运营成本低,受到越来越多的公司青睐,如新浪、网易、淘宝等企业都使用 MySQL。常用数据库市场流行度排行如图 1-2 所示,Oracle、MySQL、Microsoft SQL Server 目前使用率居前三位。

3. 结构化查询语言

1) SQL 语言的分类

关系数据库的标准语言是 SQL(structured query language)。SQL 语言是一种专门用来与数据库通信的标准语言。SQL 语言集数据定义、数据操作、数据查询和数据控制

Rank	Rank			DBMS	Database Model	Score		
	Jan 2025	Dec 2024	Jan 2024			Jan 2025	Dec 2024	Jan 2024
1.	1.	1.	1.	Oracle	Relational, Multi-model	1258.76	-5.03	+11.27
2.	2.	2.	2.	MySQL	Relational, Multi-model	998.15	-5.61	-125.31
3.	3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	798.55	-7.14	-78.05
4.	4.	4.	4.	PostgreSQL	Relational, Multi-model	663.41	-2.97	+14.45
5.	5.	5.	5.	MongoDB	Document, Multi-model	402.50	+2.12	-14.98
6.	↑7.	↑9.	↑9.	Snowflake	Relational	153.90	+6.54	+27.98
7.	↓6.	↓6.	↓6.	Redis	Key-value, Multi-model	153.36	+3.08	-6.03
8.	8.	↓7.	↓7.	Elasticsearch	Multi-model	134.92	+2.60	-1.15
9.	9.	↓8.	↓8.	IBM Db2	Relational, Multi-model	122.97	+0.19	-9.43
10.	10.	↑11.	↑11.	SQLite	Relational	106.69	+4.97	-8.51

图 1-2 数据库流行度排行
(数据来源: DB-Engines 2024 年排名。)

功能于一体,充分体现了关系数据语言的特点。

(1) 数据定义语言(data definition language,DDL): 用于定义和管理数据库结构,包括创建、修改和删除数据库对象。常见的 DDL 命令有 CREATE、ALTER、DROP。

(2) 数据操作语言(data manipulation language,DML): 用于对数据库中的数据进行操作和管理,包括插入、更新、删除数据。常见的 DML 命令有 INSERT、UPDATE、DELETE。

(3) 数据查询语言(data query language,DQL): 用于从数据库中检索数据,即执行查询操作。常见的 DQL 命令有 SELECT。

(4) 数据控制语言(data control language,DCL): 用于管理数据库的安全性和权限,控制用户对数据库对象的访问权限。常见的 DCL 命令有 GRANT 和 REVOKE,用于授予和撤销用户权限。

2) SQL 语言的特点

SQL 语言具有多方面显著特点,这些特点使其成为数据库操作中极为重要且实用的工具,具体特点如下。

(1) 综合统一。SQL 是应用比较广泛的关系数据库语言,定义了操作关系数据库的标准语法,几乎所有的关系数据库管理系统都支持 SQL。SQL 语言风格统一,可以独立完成数据库生命周期中的全部活动,包括定义关系模式、数据操作、查询、数据库安全控制等一系列操作要求,这就为数据库应用系统开发提供了良好的环境。

(2) 高度非过程化。非关系数据模型的数据操纵语言是面向过程的语言,用其完成某项请求,必须指定存取路径。而用 SQL 语言进行数据操作,用户只需提出“做什么”,而不必指明“怎么做”,因此用户无须了解存取路径,存取路径的选择以及 SQL 语句的操作过程由系统自动完成。这不仅大大减轻了用户负担,而且有利于提高数据独立性。

(3) 面向集合的操作方式。SQL 语言采用集合操作方式,不仅查找结果可以是元组的集合,而且一次插入、删除、更新操作的对象也可以是元组的集合。

(4) 以同一种语法结构提供两种使用方式。SQL 语言既是自含式语言,又是嵌入式语言。作为自含式语言,它能够独立地用于联机交互的使用方式,用户可以在终端键盘上直接输入 SQL 命令对数据库进行操作。作为嵌入式语言,SQL 语句能够嵌入高级程序(例 Java、C)语言中,供程序员设计程序时使用。而在两种不同的使用方式下,SQL 语言的语法

结构基本上是一致的。这种以统一的语法结构为用户提供了极大的灵活性与方便性。

1.2.3 数据库系统体系结构

数据库系统的体系结构设计旨在通过合理的任务划分与数据管理机制,实现高效、安全、可扩展的数据服务。其核心架构可从以下两个视角分析。

(1) 交互架构:面向用户与应用程序的交互模式,如 C/S(client/server,客户端/服务器)模式和 B/S(browser/server,浏览器/服务器)模式,关注系统如何响应用户请求。

(2) 逻辑架构:面向数据组织与管理的三层模式结构(外模式、模式、内模式),确保数据的逻辑与物理独立性。

两种架构协同作用,共同支撑数据库系统的运行。以下从应用架构到数据管理逻辑,系统阐述其设计原理与实现方式。

1. 交互架构

(1) C/S 模式。C/S 模式中,客户端负责用户界面和交互,服务器处理数据存储与操作。客户端需要安装特定软件(如支持 MySQL 的 Python、Java 等工具)来连接服务器。客户端通过网络向服务器发送请求,服务器处理后返回结果。其优点是性能高,客户端分担计算任务,减轻服务器负担;缺点是维护成本高,客户端软件需定期更新。三层 C/S 模式将应用逻辑分离到中间层,客户端专注于用户界面,服务器处理数据存储与业务逻辑。三层 C/S 模式如图 1-3 所示。

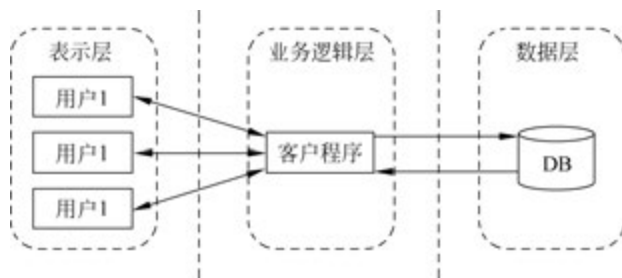


图 1-3 三层 C/S 模式

(2) B/S 模式。B/S 模式是基于 Web 的架构,即浏览器/服务器结构,客户端用 Web 浏览器作为用户界面。当前流行的三层 B/S 模式将应用程序功能分为三个层次,如图 1-4 所示。

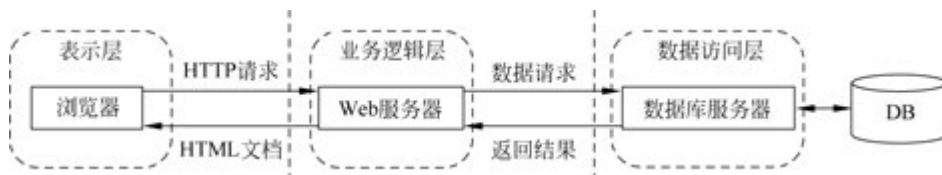


图 1-4 三层 B/S 模式

① 表示层：负责和用户直接交互，提供图形化界面，接收用户输入并转发请求。在 Web 应用中通常指 Web 浏览器，用到 HTML、CSS、JavaScript 和各种 Web 框架技术。

② 业务逻辑层：包含核心业务逻辑，处理数据验证、计算和处理等。接收表示层请求，执行逻辑处理并与数据访问层交互，再将结果返回表示层。常用服务器端编程语言如 Java、C#、Python 实现，常见框架有 Spring(Java)、.NET(C#)等。

③ 数据访问层：负责与数据库或其他数据存储机制交互，提供数据持久化服务，将业务逻辑层复杂查询转化为简单数据库操作，执行数据存取操作，可使用 SQL(用于 MySQL、Oracle、SQL Server 等关系型数据库)，或非关系型数据库如 MongoDB 等。

B/S 架构适用于需要跨平台访问、易于维护和扩展的 Web 应用程序。

2. 数据管理架构

从数据组织的视角来看，数据库系统采用外模式—模式—内模式的三层结构，这种结构设计的核心目标是保障数据的独立性，从而提高系统的灵活性和可维护性。

(1) 外模式。外模式(用户视图)是数据库用户能够访问的局部数据的逻辑结构和特征的描述，也称为子模式或用户模式。它面向特定的应用场景，为不同用户角色提供定制化的数据视图。例如，在选课管理系统中，教师的外模式可能仅包含学生成绩和课程安排；学生的外模式则显示个人选课信息和成绩。

外模式通过屏蔽底层数据的复杂性，实现用户操作的简化。同时，结合权限控制机制，外模式还能有效保障数据的安全性。这种设计使得用户无须关心底层数据的具体存储和管理细节，只需专注于他们需要的数据视图。

(2) 模式。模式(全局逻辑视图)是数据库中全体数据的逻辑结构和关系的抽象描述，是所有用户的公共数据视图。它定义了数据库中数据的整体结构，但不涉及物理存储细节，而是独立于具体的应用程序和开发工具。

例如，如果在“学生表”中增加一个“邮箱”字段，只需要修改外模式与模式之间的映像规则，而教师和学生看到的视图仍然可以保持不变。这种设计体现了逻辑独立性，即当模式的逻辑结构发生变化(如新增字段、修改表关系)时，外模式和应用程序无须调整。

(3) 内模式。内模式(物理存储视图)是数据库的物理存储结构的描述，它定义了数据在存储介质上的组织方式，例如数据的存储位置、文件格式、索引结构等。内模式的主要作用是优化数据的存储和访问性能。

内模式的设计直接影响数据库的读写性能和存储成本。通过模式—内模式映像，当内模式的物理存储方式变更时，模式和应用程序不受影响，从而实现物理独立性。

(4) 数据独立性。数据独立性是三层模式结构的核心优势，通过两层映像机制实现。

① 逻辑独立性：通过外模式—模式映像，当模式的逻辑结构发生变化时，外模式和应用程序无须调整。

② 物理独立性：通过模式—内模式映像，当内模式的物理存储方式变更时，模式和应用程序不受影响。

这种设计使得数据库系统在面对数据结构变更或存储优化时，能够最大限度地减少对现有应用的影响，从而提高系统的可维护性和扩展性。

综上所述,C/S、B/S 模式和三层模式结构从不同维度对数据库系统体系结构进行了描述,它们相互补充,共同帮助我们全面理解数据库系统的架构。在实际应用中,需要根据具体的需求和场景选择合适的体系结构。

1.2.4 数据模型

模型是对现实世界特征的模拟和抽象。提到模型,人们首先想到的是建筑模型、飞机模型等实体模型,这些模型帮助我们理解现实世界中的对象和系统并使之可视化。数据模型(data model)也是一种模型,它在数据库技术中扮演着关键角色,用于对现实世界的数据库特征进行抽象,以描述数据库的结构和语义。在实际应用中,数据模型通常分为三个层次:概念模型、逻辑模型和物理模型。



微课 1-3
数据模型

1. 概念模型

概念模型是对现实世界的认识和抽象描述,按用户的观点对实体及其联系建立概念化结构,用于信息世界的建模。概念模型不考虑在计算机和 DBMS 上的具体实现,常用的有 E-R 模型。

在 E-R 模型中可以通过一些数据来对现实世界进行抽象,具体说明如下。

(1) 实体(entity): 客观存在并可相互区分的事物。具有相同特征和性质的同一类实体的集合称为实体集。例如,学生李小明、张鹏都是实体,可以用实体集“学生”来表示。

(2) 属性(attribute): 实体或者联系所具有的某一特性,一个实体或者联系可由若干个属性来描述。例如,学生实体的属性有学号、姓名和性别等。属性由两部分组成,分别是属性名和属性值。例如,学号和姓名是属性名,而“2021001、张明”这些具体值是属性值。

(3) 联系(relationship): 实体集与实体集之间的联系,有一对一、一对多、多对多三种类型。例如,每位学生有一位学生证,学生和 student 证之间是一对一的联系;一位班级有多位学生,班级和学生是一对多的联系;一位学生可以选修多门课程,一门课程又可以被多位学生选修,学生和课程之间就形成了多对多的联系。

例如,在选课管理系统中,将学生、课程以及学生选课的客观事物抽象为“学生”实体、“课程”实体以及实体之间的“选课”联系,并派生出“成绩”属性。

将实体集及其联系用图表示称为实体—联系(entity-relationship, E-R)图。E-R 图提供了表示实体集、属性和联系的方法,用来描述现实世界的模型,其表示方法如下。

- 实体: 用矩形表示,将实体名写在框内。
- 属性: 用椭圆表示,将属性名写在框内,用无向线将实体与属性连接。
- 联系: 用菱形表示,将联系名写在框内,用无向线将相关的实体连接,并在连线旁标注联系类型(1:1、1:n 或 m:n)。如果联系也有属性,则把属性和联系也用无向边连接,如图 1-5、图 1-6 所示。