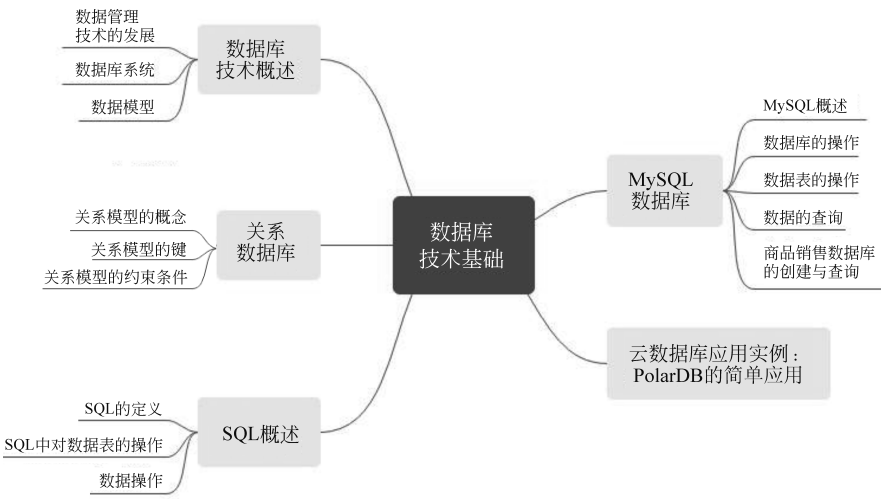
第5章

数据库技术基础

本章学习目标

- 了解数据管理技术的发展阶段。
- 理解数据库系统的概念。
- 熟悉常见的数据模型。
- 掌握关系模型的概念及约束条件。
- 了解 SQL 的基本概念。
- 熟练运用 SQL 进行数据库和数据表的基本操作。
- 掌握 MySQL 的基本用法。
- 能够操作简单的云数据库。

本章思维导图



5.1 数据库技术概述

5.1.1 数据管理技术的发展

数据管理是指对数据的组织、分类、编码、存储、检索和维护等环节的操作，是数据处

理的核心，随着计算机硬件、软件技术的不断发展，数据管理也经历了由低级到高级的发展过程，这个过程大致经历了人工管理、文件系统管理和数据库系统管理 3 个阶段。

1. 人工管理阶段

20 世纪 50 年代以前，当时计算机主要应用于数值计算。由于处理的数据量小，由用户直接管理(见图 5-1)，因此这一时期数据管理的主要特点如下。

(1) 数据与程序不可分割，数据依赖于特定的应用程序，没有专门的软件进行数据管理，数据的存储结构、存取方法和输入输出方式完全由程序员自行完成。

(2) 数据不保存，应用程序在执行时输入数据，程序结束时输出结果，随着处理过程的完成，数据与程序所占空间也被释放，数据无法重复使用，因此不能实现数据的共享。

(3) 各程序所用的数据彼此独立，数据缺乏逻辑组织和独立性，程序和程序之间存在大量的数据冗余。

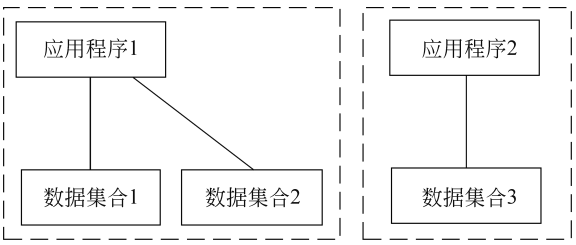


图 5-1　人工管理阶段示意图

2. 文件系统管理阶段

20 世纪 50 年代后期到 60 年代中期，硬件设备中出现了磁鼓、磁盘等直接存取数据的存储设备。在软件技术上，出现了操作系统和高级程序设计语言。在操作系统中，有了数据和文件管理的模块，负责把数据组织成相互独立的数据文件，系统可以按名存取文件，并可以实现对文件的修改、插入和删除。文件系统管理阶段示意图如图 5-2 所示。

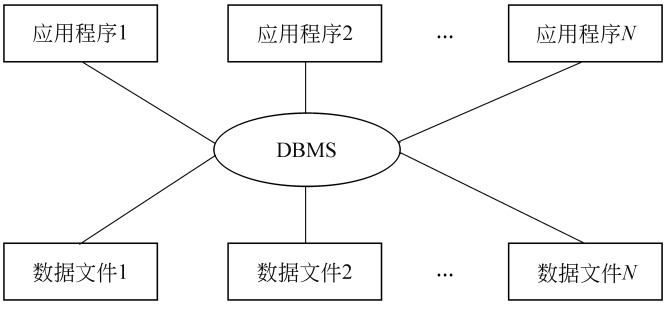


图 5-2　文件系统管理阶段示意图

这一时期的主要特点如下。

(1) 程序和数据分开存储，数据以文件的形式保存在外存储器上，这样程序和数据之

间有了一定的独立性。

（2）一个应用程序可使用多个数据文件，而一个数据文件也可以被多个应用程序所使用，从而实现了数据的共享。

但是，当要处理的数据量剧增时，文件系统的管理方法就暴露出如下问题：数据冗余性问题。由于每个应用程序都有对应的数据文件，因此有可能造成同样的数据在多个文件中重复存储。另外一个问题是数据不一致性。在对数据进行更新时，因此有可能造成同样的数据在不同文件中的更新不同步。

3. 数据库系统管理阶段

20 世纪 60 年代后期开始，由于计算机硬件、软件的快速发展，出现了数据库和对数据库进行统一管理的软件系统，这就是数据库管理系统。

数据库系统阶段具有以下特点。

（1）数据结构化。数据库中的数据按一定的数据模型进行组织，这样数据库系统不仅可以表示事物内部数据项之间的关系，也可以表示事物与事物之间的联系。

（2）数据冗余度小。数据以数据库文件的形式独立保存，数据结构中的每个数据元素在数据库中以记录的形式保存。这样，在建立数据库时，可以最大限度地减少数据的冗余。

（3）数据独立性高。数据库文件和应用程序之间彼此独立，应用程序的修改不影响数据、数据结构，以及存储位置的修改也不影响应用程序。

（4）数据共享性好。数据不面向特定的应用程序，而是面向整个系统组织管理相关数据，任何应用程序都可以使用数据库中所需要的其他数据。

（5）数据统一管理。数据由数据库管理系统进行管理。它负责建立数据库，维护数据库，提供安全性、数据一致性、完整性的控制，并提供多个用户同时使用的机制。

数据库系统管理阶段示意图如图 5-3 所示。

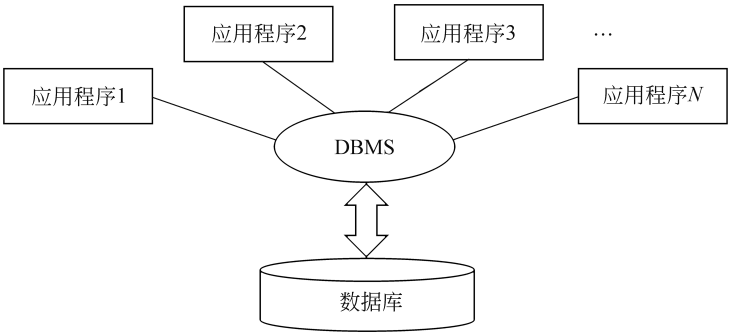


图 5-3　数据库系统管理阶段示意图

课程思政：遵纪守法，打造中国人自己的数据库

数据库的设计与维护需要具备职业道德规范。对于保密数据的使用，一定要按照国

家相关规定来处理，不能因为个人的利益而出卖国家数据机密。

随着国家“核高基”重大专项工作的实施，国产数据库的实力在不断增强，数据库的技术水平不断提升，各种功能日益完善，市场空间也变得越来越大。目前国产数据库有人大金仓、华为云 GaussDB、腾讯云 TcaplusDB 和阿里的云原生数据库等。

这里以华为云 GaussDB 为对象，简要介绍。

华为云 GaussDB 是华为自研数据库品牌，是华为基于外部电信与金融政企经验、华为内部流程 IT 与云底座深耕 10 年以上的数据库内核研发优化能力，从客户对高可用、高性能、安全可靠等诉求出发，结合云技术倾力打造的企业级分布式数据库。

GaussDB 是一个产品系列，在整体架构设计上，底层是分布式存储，中间是每个 DB 特有的数据结构，最外层则是各个生态的接口，体现了多模的设计理念，具体产品包括如下 3 种。

1. 关系型数据库

(1) 基于 openGauss 生态的分布式数据库 GaussDB(for openGauss)。

(2) 基于 MySQL 生态的分布式数据库 GaussDB(for MySQL)，100％兼容 MySQL。

2. 非关系型数据库

(1) 基于文档的分布式数据库 GaussDB(for Mongo)，100％兼容 MongoDB 的接口，解决了开源的协议风险，用 3 副本解决 9 副本的可靠性。

(2) 基于时序协议的分布式数据库 GaussDB(for Influx)，解决了开源主备的容量问题。

(3) 基于宽表 Cassandra 的 GaussDB(for Cassandra)，解决了 Java 中内存抖动(内存中大量对象被创建又被释放)的严重问题，数据强一致。

(4) 基于 KV 接口 GaussDB(for Redis)，当前 100％兼容 Redis，未来可以兼容更多的 KV 协议，数据强一致，超大容量 10T+，解决了开源主备脱节等问题。

3. 数据库工具及中间件

(1) 数据复制服务(Data Replication Service，DRS)。

(2) 数据库和应用迁移(Database and Application Migration UGO，UGO)。

(3) 分布式数据库中间件(Distributed Database Middleware，DDM)。

5.1.2 数据库系统

1. 数据库

数据库(DataBase，DB)，是指以文件的形式并按特定的组织方式将数据保存在存储介质中，因此，数据库中不仅包含了数据本身，也包含了数据之间的联系。

2. 数据库管理系统

数据库管理系统(Database Management System,DBMS),是实现对数据库进行管理的软件,它以统一的方式管理和维护数据库,并提供数据库接口供用户访问数据库。用户不必关心数据的结构和实现方法,发出命令完成具体的操作即可。常用的数据库管理软件有 Visual FoxPro、Access、SQL Server、MySQL、Oracle、DB2 等。

数据库管理系统一般具有如下功能。

1) 数据定义

数据定义通过数据定义语言(Data Definition Language,DDL)实现,用户通过它可以定义数据库,并定义和修改数据库中表的结构、数据完整性和其他约束规则等。

2) 数据操纵

数据操纵通过数据操纵语言(Data Manipulation Language,DML)实现。它是对数据库中的数据进行增加、修改、删除和查询等操作。

3) 数据库的建立和维护

数据库的建立包括数据库初始数据的输入,数据库的维护包括数据库的转储、恢复功能,还有数据库的性能分析功能等。

4) 运行控制

数据库的操作都要在控制程序的统一管理下进行,以保证数据的安全性、完整性以及多个用户对数据库的并发使用。

5) 数据字典

数据字典是描述数据的信息集合,是对数据库系统中要使用的所有数据元素的定义的集合。

3. 应用程序

应用程序是指系统开发人员使用数据库管理系统以及数据库资源开发的、应用于某一特定领域或某一实际问题的应用软件,如图书借阅管理系统、教务管理系统、药房管理系统、门诊预约管理系统等。

4. 数据库管理员

数据库管理员(Database Administrator,DBA),是专门从事管理和维护数据库管理系统的相关工作人员的统称,属于运维工程师的一个分支,主要负责业务数据库从设计、测试到部署交付的全生命周期管理。数据库管理员的核心目标是保证数据库管理系统的稳定性、安全性、完整性和高性能。

5. 数据库系统

数据库系统(Database System,DBS),是指由硬件、操作系统、数据库、数据库管理系统、应用程序、数据库管理员等组成的一个综合体。数据库系统构成示意图如图 5-4 所示。

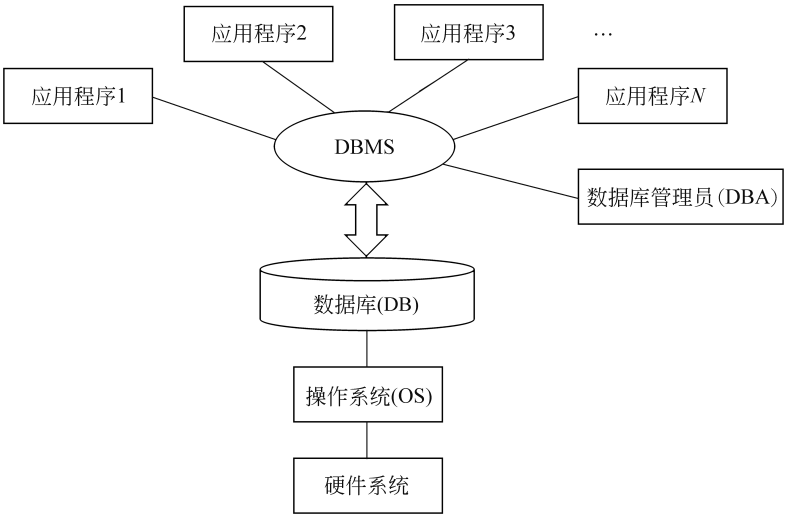


图 5-4　数据库系统构成示意图

5.1.3　数据模型

数据库中不仅包含数据本身，也包含数据和数据之间的联系。数据的组织是按特定的结构进行的，这种结构就是数据模型。

根据应用层次的不同，数据模型有 3 个类别：概念模型、逻辑模型和物理模型。

1. 概念模型

概念模型用来描述现实世界的事物。在设计数据库时，用概念模型抽象表示现实世界的各种事物及其联系。在现实世界中，通常把事物称为实体(entity)，如学生、课程、教师等。实体具有属性(attribute)，如学生的属性有班级、姓名、学号等。实体之间有联系(relationship)，如一位学生可以选修多门课程，一门课程可以被多人选修。

1) 实体之间的联系

实体之间的联系也称为实体集之间的联系，主要有 3 种类型：一对一、一对多和多对多。

- 一对一联系：实体集 A 中的一个实体最多与实体集 B 中的一个实体有联系，反之亦然。例如，一个学校只有一位校长，一个校长只能在一个学校任职。
- 一对多联系：实体集 A 中的一个实体与实体集 B 中的多个实体有联系，但实体集 B 中的一个实体至多和实体集 A 中的一个实体有联系。例如，一个班级有多名学生，而一名学生只属于一个班级。
- 多对多联系：实体集 A 中的一个实体与实体集 B 中的多个实体有联系，而实体集 B 中的一个实体也可以和实体集 A 中的多个实体有联系。例如前面提到的学生和课程之间就是多对多的联系。

2）E-R 图

概念模型常用如图 5-5 所示的图形表达，称为实体-关系模型，也叫 E-R 图，其中矩形表示实体，椭圆表示属性，菱形表示联系，连线表示所属关系，字母 M 和 N 表示多对多的联系。

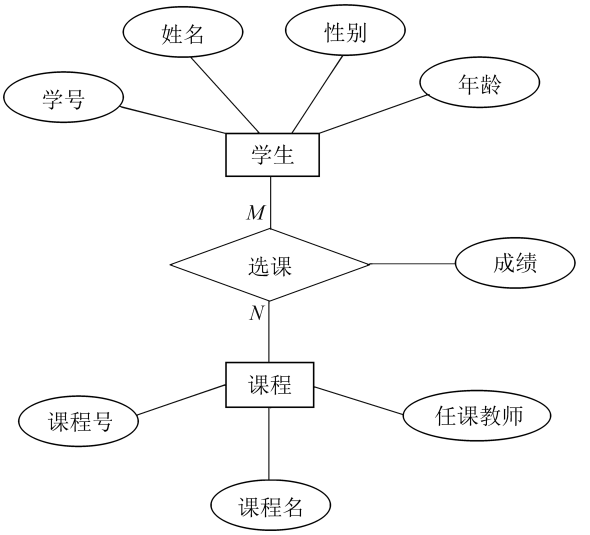


图 5-5　实体-关系模型图

2. 逻辑模型

表示数据之间逻辑关系的模型称为逻辑模型。目前，数据库管理系统所支持的数据逻辑模型有 3 种，即层次模型、网状模型和关系模型。

1）层次模型

层次模型是指用树形结构组织数据，可以表示数据之间的多级层次结构。

在树形结构中，各个实体用节点表示。整个树形结构中只有一个节点为最高节点，其余节点有且仅有一个父节点，相邻两层的上级节点和下级节点之间表示了一对多的联系。

现实世界中存在着大量的可以用层次结构表示的实体，例如国家的行政区域结构、单位的行政组织机构、家族的辈分关系等。

2）网状模型

网状模型中常用图的方式表示数据之间的关系，它突破了层次模型的两个限制：一是允许节点有多于一个的父节点；另一个是可以有一个以上的节点没有父节点。

网状模型可以表示多对多的联系，但数据结构的实现比较复杂。

3）关系模型

关系模型可以用二维表格的形式描述实体及实体之间的联系，在实际的关系模型中，操作的对象和操作的结果都用二维表表示，每个二维表都代表了一个关系。

在这 3 种数据模型中，关系模型的数据组织最简单、方便，因此，目前流行的数据库管理系统都是以关系模型为基础的，称为关系数据库管理系统。

3. 物理模型

物理模型用于描述数据在存储介质上的结构，如数据的存放顺序、文件的位置、分布等。每种逻辑模型在实现时，都对应相应的物理模型。物理模型不但由数据库管理系统决定，还与操作系统、计算机硬件有关。

5.2　关系数据库

5.2.1　关系模型的概念

采用关系模型组织数据的数据库被称为关系型数据库。在关系数据库中，对数据的操作是建立在一个或多个关系表格上的，通过对这些关联表的运算实现数据的处理。

1. 关系模型的定义

在关系模型中，可以使用二维表格形象地描述相关的数据和它们之间的关系，图 5-6 所示的读者信息表就是一个关系，关系名为 reader。

读者信息表

编号	姓名	年龄	性别	← 字段
0800001	黄青山	18	男	
0800002	李化	17	女	← 记录
0800003	周得鑫	19	女	
0800004	宋青	18	男	

图 5-6　关系模型的组成

在使用关系模型时，经常用到下面的术语。

1）字段

在上面的二维表中，垂直方向的每一列在具体的数据库文件中称为一个字段，也叫属性。每一个字段的名称为字段名，也叫属性名。

例如，读者信息表中有 4 个字段，字段名分别是“编号”“姓名”“年龄”和“性别”。

2）域

每个字段的取值范围称为域，例如，性别字段的取值域是{男，女}。

3）字段值

字段的取值称为字段值或属性值，例如，第二条记录的年龄字段值为 17。

4）表结构

一个完整的二维表由表结构和记录两部分组成。一个二维表中的第一行的所有字段的名称和属性的集合称为该表的结构。在创建表时，除了创建字段外，还应创建各个字段的数据类型、取值范围和宽度等，这些都称为字段的属性。

5）记录

二维表中从第二行起的每一行分别称为一条具体的记录，图 5-6 中的表由 4 个字

段 4 条记录组成。修改表时，要先分清是对表结构进行操作，还是对记录进行操作。

6）关系模式

关系模式是指对关系结构的描述，通常可以简记为如下格式。

关系名(字段 1，字段 2，字段 3，…，字段 *n*)

例如，图 5-6 的关系模式可以表示为：reader(学号，姓名，年龄，性别)。

2. 关系模型的特点

关系模型的结构简单，通常具有以下特点。

(1) 关系中的每一列不可再分。

(2) 同一个关系中不能出现相同的字段名。

(3) 关系中不允许有完全相同的记录。

(4) 关系中任意交换两行位置不影响数据的实际含义。

(5) 关系中任意交换两列位置也不影响数据的实际含义。

5.2.2 关系模型的键

键(Key)也称为**码**(Code)。在关系表中，常用的有候选键和主键。

1. 候选键(Candidate Key)

在一个关系表中可以唯一地区分一条记录的字段或字段的组合，称为候选键。

例如，在关系 reader 中，字段“编号”可以作为候选键。字段“编号”和“姓名”的组合也可以作为候选键。

一个关系表中可以有多个候选键，候选键也可以是字段的组合。

2. 主键(Primary Key)

如果一个关系表中有多个候选键，则可以从候选键中指定其中一个作为主键。

设置主键的目的是实现实体完整性的约束规则，由于主键的重要作用就是用来标识每条记录，这样，关系的实体完整性要求关系表中的记录在组成的主键上不允许出现两条记录的主键值相同，也就是说，既不能有空值，也不能有重复值。

例如，在图 5-6 的关系 reader 中，如果字段“编号”作为主键，那么该字段的值不能为空，也不能有两条记录的编号值相同。

3. 外键(Foreign Key)

如果表中的一个字段不是本表的主关键字或候选关键字，而是另外一个表的主关键字或候选关键字，则该字段称为外部关键字，简称外键。

例如，关系表 student (学号，姓名，性别，年龄，借书证号)的主键是学号。

关系表 score(学号，课程号，成绩)的主键是(学号，课程号)。

在关系表 score 中，主键是(学号，课程号)，其中“学号”不是 score 的主键，而是关系

表 student 的主键，因此，在关系表 score 中“学号”称为外键。

通过外键可以将两个表联系起来，其中以外键作为主键的表称为主表，例如，关系表 student。外键所在的表称为从表，例如，关系表 score。这样，两个表之间通过外键就可以建立起联系。

5.2.3 关系模型的约束条件

对数据库的数据进行录入时，为了保证数据库中数据的正确性和合理性，必须设置一定的条件，只有符合条件的数据，才允许进入数据库，这些限定条件称为约束条件。约束条件主要有实体完整性、参照完整性和用户定义完整性。

1. 实体完整性

实体完整性要求表中的记录都必须有主键，而作为主键的字段值必须是唯一的且是非空值。实体完整性就是保证关系中不能有重复的记录。

例如，关系 student（学号，姓名，性别，年龄，借书证号）的主键为“学号”，因此学号必须有值而且还不能重复。

2. 参照完整性

参照完整性约束规则是对相关联的主表和从表之间的约束，就是从表中的每一条记录，其外键的值必须是主表中存在的，因此，如果在两个表之间建立了关联关系，则对一个表进行的操作要影响到另一个表中的记录。这种制约表现在两方面：一是对从表添加和修改记录时受到的制约；二是对主表删除和修改时产生的级联反应。

对从表中的记录进行添加和修改操作时会受主表的制约影响。例如，如果在学生表和选修课之间通过学号建立关联，学生表是主表，选修课是从表，那么，在向从表中输入一条新记录时，系统要检查新记录的学号在主表中是否已存在，如果存在，则允许执行输入操作，否则拒绝输入；如果要修改从表中的学号，系统也要检查修改后的学号在主表中是否存在，如果不存在，系统同样会拒绝修改，这就是参照完整性约束。

对主表中的记录进行删除和修改操作时，会对从表产生影响。例如，如果删除主表中的一条记录，则从表中凡是与主表的主键值相同的记录也会被同时删除，这就是级联删除；如果修改主表中主键的值，则从表中相应记录的外键值也随之被修改，这就是级联更新。

3. 用户定义完整性

用户定义完整性是用户根据字段的意义设定的限制条件。例如，设定性别字段的取值范围为{男，女}。

关系的完整性可以在创建表时设定。当输入数据时，系统检查完整性约束条件，如果数据违反了约束条件，就不允许数据插入数据表中。