第 1 章

MySQL 数据库基础

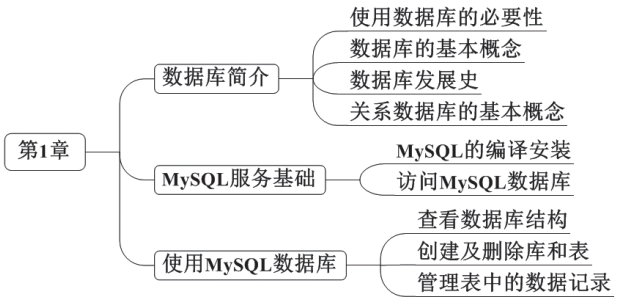
技能目标

- 理解数据库的基本概念
- 掌握 MySQL 的编译安装
- 会操作 MySQL 数据库

本章导读

21 世纪，人类迈入了“信息爆炸时代”，大量的数据、信息在不断产生，伴随而来的就是如何安全、有效地存储、检索和管理它们。对数据的有效存储、高效访问、方便共享和安全控制已经成为信息时代亟待解决的问题。

知识服务



1.1 数据库简介

1.1.1 使用数据库的必要性

使用数据库可以高效且条理分明地存储数据，使人们能够更加迅速、方便地管理数据。数据库具有以下特点：

- 可以结构化存储大量的数据信息，方便用户进行有效的检索和访问。
- 可以有效地保持数据信息的一致性、完整性，降低数据冗余。
- 可以满足应用的共享和安全方面的要求。

数据库技术是计算机科学的核心技术之一，具有完备的理论基础。对数据库基本概念的掌握，将有助于对数据库的理解。

1.1.2 数据库的基本概念

1. 数据

描述事物的符号记录称为数据（Data）。数据不仅仅包括数字，文字、图形、图像、声音、档案记录等都是数据。

在数据库中，数据是以“记录”的形式按统一的格式进行存储的，而不是杂乱无章的。相同格式和类型的数据统一存放在一起，而不会把“人”和“书”混在一起存储。这样，数据的存储就能够井然有序了。

如图 1.1 中存储的一行数据，在数据库中称为一条“记录”（Record）。每条记录中的每一个输入项称为“列”。图 1.1 中编号、姓名、性别、年龄、民族、专业都是列名。

	编号	姓名	性别	年龄	民族	专业
1	1	王明	男	22	汉	计算机科学与技术
2	2	赵玲	女	21	汉	信息管理
3	3	张雨彤	女	20	汉	音乐
4	4	李新	男	23	回	自动化
5	5	夏雪	女	22	汉	通信

图 1.1　数据记录

2. 数据库和数据库表

不同的记录组织在一起，就形成了数据库（Database，DB）的“表”（Table）。也可以说，表是用来存储具体数据的，如图 1.1 所示。那么数据库和表存在什么关系呢？简单地说，数据库就是表的集合。它是以一定的组织方式存储的相互有关的数据集合。例如，关系数据库的表由记录组成，记录由字段组成，字段由字符或数字组成。它可以供各种用户共享，具有最小冗余度和较高的数据独立性。它是统一管理的相关数据的集合。

通常，数据库并不是简单地存储这些数据的，还要表示它们之间的关系。例如，书和人是存在联系的，书的作者可能就是某个人，因此需要建立书与人的“关系”。这种关系也需要用数据库来表示，因此关系的描述也是数据库的一部分。

3. 数据库管理系统和数据库系统

数据库管理系统（Database Management System，DBMS）是实现对数据库资源有效组织、管理和存取的系统软件。它在操作系统的支持下，支持用户对数据库的各项操作。DBMS 主要包括以下功能：

- 数据库的建立和维护功能：包括建立数据库的结构和数据的录入与转换、数据库的转储与恢复、数据库的重组与性能监视等功能。
- 数据定义功能：包括定义全局数据结构、局部逻辑数据结构、存储结构、保密模式及信息格式等功能。保证存储在数据库中的数据正确、有效和相容，以防止不合语义的错误数据被输入或输出。
- 数据操纵功能：包括数据查询统计和数据更新两个方面。
- 数据库的运行管理功能：这是数据库管理系统的核心部分，包括并发控制、存取控制、数据库内部维护等功能。
- 通信功能：DBMS 其他软件系统之间的通信，如 Access 能与其他 Office 组件进行数据交换。

数据库系统（Database System，DBS）是一个人—机系统，一般由硬件、操作系统、数据库、DBMS、应用软件和数据库用户（包括数据库管理员）组成。用户可以通过 DBMS 操作数据库，也可以通过应用程序操作数据库。

应用程序是利用 DBMS，为解决某个具体的管理或数据处理的任务而编制的一系列命令的有序集合。如果应用程序比较完善，能够提供友好的人机界面，并编译成可执行文件发行，使得普通用户不需要具备计算机的专业知识，在较短时间内就学会使用，那么就称为数据库应用软件。

常用的数据库应用软件有人事管理、财务管理、图书管理等信息管理软件及各类信息咨询系统等。

数据库管理员（Database Administrator，DBA）负责数据库的更新和备份、数据库系统的维护、用户管理等工作，保证数据库系统的正常运行。DBA 一般由业务水平较高、资历较深的人员担任。

注意

数据库、数据库系统、数据库管理系统，甚至数据库表等名词，在日常讨论中通常不严格区别。遇到此情况时，可以根据具体情况，判断出实际所指的是什么。

1.1.3 数据库发展史

1. 数据库系统发展史

数据库技术的发展已经成为先进信息技术的重要组成部分，是现代计算机信息系统和计算机应用系统的基础和核心。数据库技术最初产生于 20 世纪 60 年代中期，根据数据模型的发展，可以划分为三个阶段：第一代的网状、层次数据库系统；第二代的关系数据库系统；第三代的以面向对象模型为主要特征的数据库系统。

（1）初级阶段——第一代数据库

自 20 世纪 60 年代起，第一代数据库系统问世。它们是层次模型与网状模型的数据库系统，为统一管理和共享数据提供了有力的支撑。在这个阶段中，数据库的代表是 1969 年 IBM 公司研制的层次模型的数据库管理系统——IMS（Information Management System，信息管理系统）和 20 世纪 70 年代美国数据系统语言协会（CODASYL）下属数据库任务组（DBTG）提议的网状模型。

（2）中级阶段——第二代数据库

20 世纪 70 年代初，第二代数据库——关系数据库开始出现。自 1970 年 IBM 研究员德加 • 考特阐述了关系模型的概念后，IBM 大力投入关系数据库的研究。关系数据库的底层实现起来比较容易，所以很快被采用，并进入了众多商业数据库的研发计划。Oracle 就是当时顺应关系数据模型的出现而成立的一家专做（关系）数据库的公司。20 世纪 80 年代初，IBM 公司的关系数据库系统 DB2 问世，而 Oracle 公司也将 Oracle 移植到桌面计算机上。这时，作为第二代数据库系统的关系数据库，开始逐步取代层次与网状模型的数据库，成为占主导地位的数据库，并成为行业主流。到目前为止，关系数据库系统仍占领数据库应用的主要地位。

关系数据库系统将结构化查询语言（Structured Query Language，SQL）作为数据定义语言（Data Definition Language，DDL）和数据操作语言（Data Manipulation Language，DML），它一诞生就成为关系数据库的标准语言。SQL 使得关系数据库中数据库表的查询可以通过简单的、声明性的方式进行，大大简化了程序员的工作。

关系数据库系统构筑在比较高的软件层次上，执行查询的效率普遍偏低。另外，严格的、标准的关系数据库是一个纯理论的模型，如果完全按照关系模型实现，会涉及很多方面的问题，其中一条就是效率不高。在现实环境中，考虑到商业运用的目的，数据库生产厂商各自加入了一些提高效率和提高可用性的功能，舍弃了一些不太现实

的约束。不同的数据库厂商在不同基础上的选择，导致了关系数据库系统向不同方向上的变迁。例如，在这个阶段中，Oracle 加入了“并行”的元素，并开始了向“关系—对象”型数据库的变迁。这样的变迁，也慢慢引出了新一代的数据库系统。

（3）高级阶段——第三代数据库

由于计算机应用的发展，计算机已从传统的科学计算、事务处理等领域，逐步扩展到工程设计、统计、人工智能、多媒体、分布式等领域，这些新的领域需要有新的数据库支撑，而传统关系数据库系统是以商业应用、事务处理为背景而发展起来的，并不完全适用于新领域的应用，因此，需要新的数据库系统，以满足不同领域的要求。

自 20 世纪 80 年代开始，各种适应不同领域的新型数据库系统不断涌现，如工程数据库、多媒体数据库、图形数据库、智能数据库、分布式数据库及面向对象数据库等，特别是面向对象数据库系统，由于其实用性强、适应面广而受到人们的青睐。20 世纪 90 年代后期，形成了多种数据库系统共同支撑应用的局面。当然，在商务应用方面，依然还是关系数据库占主流，不过，已经有一些新的元素被添加进主流商务数据库系统中。例如，Oracle 支持的“关系—对象”数据库模型。

随着科学技术的发展，计算机技术不断应用到各行各业。数据存储需求的不断膨胀，对未来的数据库技术将会有更高的要求。

2. 经典数据模型

数据是现实世界中“量”的抽象，而数据模型（Data Model）是数据特征的抽象。在数据库系统中，数据模型是它的核心与基础。数据模型表现为数据的结构、定义在其上的操作及约束条件。它从概念层次上描述了系统的静态特征、动态特征和约束条件，为数据库系统的信息表示与操作提供了一个抽象框架。

在 DBMS 的发展过程中，出现了网状模型、层次模型和关系模型三种经典的数据模型。由于受限于数学基础、编程技术和硬件条件，最初出现的层次模型和网状模型这两种模型与关系模型相比，与用户接口的上、中层部分更易于实现，所以，这在很长一段时间阻碍了关系模型的发展。

数据模型所描述的内容包括三方面：数据结构、数据操作和数据约束。下面简单介绍三种经典数据模型。

（1）网状模型

1）数据结构。在网状模型中，数据记录组织成图的形式，使用“数据结构图”进行抽象的分析和表示。如图 1.2 所示，该网状模型数据结构图表示的是银行客户、银行账户和银行支行三方面的一种复杂关系。

图 1.2 所示的网状模型抽象于图 1.3 所示的实际情况。它表示了下列几种基本关系：

- 一个银行客户可以拥有多个银行账号（一对多）。
- 一个银行账号也可以被多个客户所有（一对多）。
- 每个银行账户位于特定的银行支行（一对一）。

其中更蕴含着客户和银行之间多对多、多对一的关系。这种复杂的数据关系，在

网状模型上可以得到很好的支持。

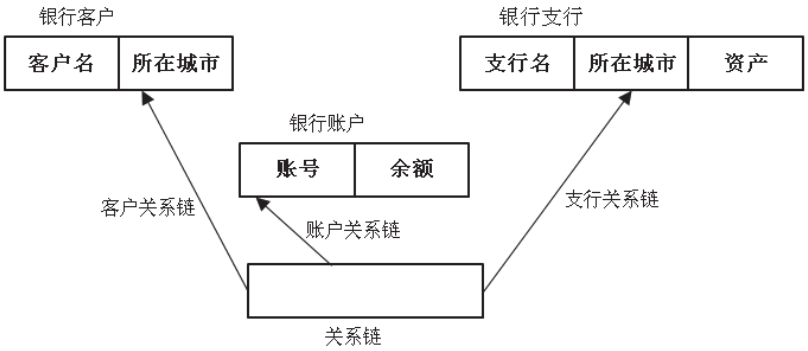


图 1.2　网状模型数据结构

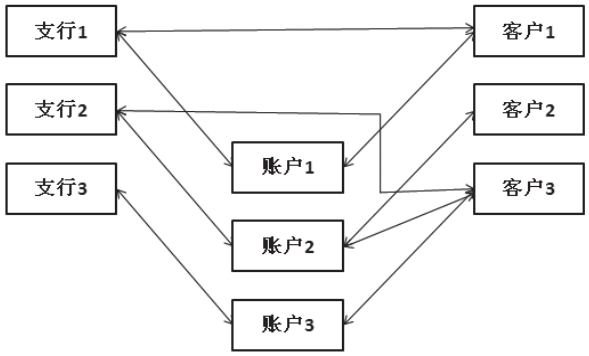


图 1.3　银行、客户、账户关系示例

网状模型适合表达复杂的数据关系的实现，也可以将数据冗余减小到最小。它的数据结构模型能直观反映现实中数据之间的联系。

2）数据操作。从数据结构的定义上不难看出，网状模型的数据操作是建立在关系链基础上的导航式的操作。

针对一个特定的网状模型系统的数据结构，有可能找到最优的查询算法。但是，一旦结构发生变化，就需要新的查询办法。网状模型以图论为基础，还无法得到一个通用的、高效的解决方案。所以，随着关系模型的飞速发展和广泛应用，网状模型已经暂时失去其重要性了。

3）数据约束。网状模型的数据约束是零散孤立的，或者分散在各个节点，或者集中成为一种关系链，这样容易导致不一致性或降低效率。所以，通常网状模型不具体实现数据约束，而由应用程序自身来实现数据约束。这样的情况也使得在网状模型基础上的开发变得困难重重。

（2）层次模型

1）数据结构。层次模型是网状模型的一个特例。在层次模型中，数据记录组织成

树的形式，使用“树结构图”进行抽象的分析和表示，适合一对多的关系模型。图 1.4 所示为企业组织结构的树结构图。

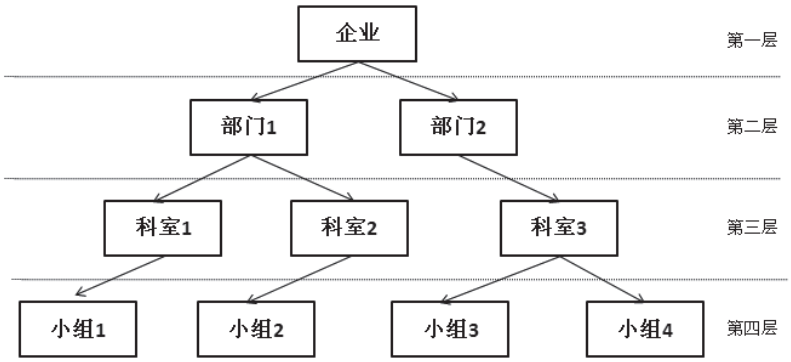


图 1.4　层次模型的树结构图

图 1.4 所示的层次模型的树结构图抽象于图 1.5 所示的实际情况。它包含了下列关系：

- 企业下辖多个部门（一对多）。
- 部门下辖多个科室（一对多）。
- 科室下辖多个小组（一对多）。

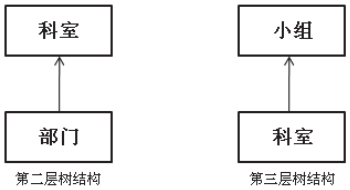


图 1.5　企业组织结构示例

相对于网状模型，层次模型禁止了多对多和多对一的关系，使得它的数据结构相对简单。

2）数据操作。在层次模型上的数据操作不可避免地具有网状模型的特点——导航性。但是，由于禁止了多对一和多对多的关系，因此数据操作相对网状模型而言简单了许多。这样的结构有利于提高数据的查询效率，但数据存取上还存在着必须导航的要求，因此，层次模型的数据库在数据操作上依然比较复杂。

3）数据约束。层次模型的数据约束与网状模型相似，由于结构的简化，去掉了网状模型中多对多和多对一的关系，数据约束处理的复杂性按级数下降，所以，层次模型的数据约束可以做到适当的系统实现，但很多还是依靠应用程序本身实现。

层次模型的实现技术比关系模型优越，比网状模型简单，所以一直独领风骚。它的代表是 IBM 公司的 IMS 系统。该系统曾是使用最早和最广的几个数据库之一，在

历史上曾是最大的数据库之一，由于它的开发者是最早开始处理并发、恢复、完整性和高效查询等问题的人，其中的一些技术和思想自然应用到 DB2 中，这是 DB2 长盛不衰的根源。DB2 有许多性能是非常优秀的，与这段历史不无关系。

（3）关系模型

1）数据结构。关系模型建立在关系代数的理论基础上，数据结构使用简单易懂的二维数据表，可以用简单的“实体 - 关系”（E-R）图来直接表示，E-R 图中包含了实体（数据对象）、关系和属性三个要素。

如图 1.6 所示，该图表示了银行客户和银行账户之间的关系。

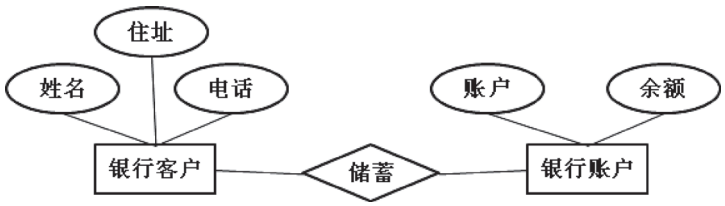


图 1.6　银行客户和银行账户之间的关系

- 实体：也称为实例，对应现实世界中可区别于其他对象的“事件”或“事物”，如银行客户、银行账户等。
- 实体集：具有相同类型及共享相同性质的实体集合。例如，银行所有客户的集合可以定义为“银行客户”实体集。
- 属性：实体所具有的某一特性，一个实体可以有多个属性。例如，“银行客户”实体集中的每个实体均具有姓名、住址、电话等属性。
- 联系：实体集之间的对应关系称为联系，也称为关系。例如，银行客户和银行账户之间存在“储蓄”的关系。

图 1.6 在关系数据库中可以简单地体现为下面三个表。

表 1-1 表示作为银行客户的实体集（客户编码为区分客户实体的唯一标识）。

表 1-1　银行客户表

客户编码	客户姓名	客户住址	客户电话
1	王明	中关村	010-87654321
2	赵玲	回龙观	010-12345678

表 1-2 表示作为银行账户的实体集（账户编码为区分账户实体的唯一标识）。

表 1-2　银行账户表

账户编码	账户	余额
1	23598273959029837	2000.23
2	27298275359025681	5000.55

表 1-3 表示建立银行账户和银行客户之间的储蓄关系（储蓄编码为区分各储蓄关系的唯一标识）。

表 1-3　储蓄关系表

储蓄编码	客户编码	账户编码
1	1	2
2	2	1

注意

虽然在银行客户看来，自己的账户和别人的账户完全不同，是唯一的。但是，银行内部往往还是会使用内部编码来区分管理和服务等不同业务。

2）数据操作。对于数据库的用户而言，关系模型使用从关系代数上抽象出来的数据库操作语言（DML）进行操作。结构化查询语言（SQL）就是其中最重要的一种，已经成为关系数据库的标准操作语言。它的特色是直接面向结果，简化操作步骤，使得数据库应用的设计变得非常简单易懂。

对于数据库的物理结构而言，关系数据库系统的数据结构简单、功能强、数据独立性高、理论基础坚实。严格的关系数据库以二维的数据库表作为基本数据结构，利用简单或复杂的索引技术实现查询算法，实现起来相对比较简单，也方便了预编译技术将 SQL 语言直接转化为有效的数据检索算法。

3）数据约束。关系模型的数据约束可以针对实体，也可以针对实体的属性，还可以针对关系，并可以在定义实体、实体属性和关系时全面实现。关系模型使用的数据定义语言（DDL）和关系模型在理论上对关系数据库核心实现的要求，使得数据约束可以很容易实现，但是它的效率并不高。

综合来说，关系模型相比网状模型和层次模型有更为坚实和完整的理论基础。相比层次模型和网状模型而言，关系模型与用户更靠近些，而网状模型和层次模型与底层实现的结合更紧密。这样的特色也更容易让关系模型成为商业数据库的选择。

3. 当今主流数据库介绍

在数据库技术日益发展的今天，主流数据库代表着成熟的数据库技术。了解常用数据库，就能知道数据库技术发展的程度，以及未来的大体方向。

（1）关系数据库

20 世纪 80 ～ 90 年代是关系数据库产品发展和竞争的时代。在市场逐渐淘汰了第一代数据库管理系统的大局面下，SQL Server、Oracle、IBM DB2、MySQL 等一批很有实力的关系数据库产品走到了主流商用数据库的位置。

1）SQL Server 简介

SQL Server 是 Microsoft 公司的数据库产品，在设计上大量利用了 Microsoft Windows 操作系统的底层结构，直接面向 Microsoft Windows，尤其是 Windows 系列服务器操作系统的用户。

Microsoft Windows 拥有众多的用户群，Microsoft 所有的产品都遵循统一的操作习惯。对数据库基本概念熟悉的 Windows 用户，可以很快地学会使用 SQL Server，上手比较容易。Windows 系统的易用性也让数据库管理员可以更容易、更方便、更轻松地进行管理。

Microsoft 公司针对市场的需求，不断扩展其性能，使得 SQL Server 在网络数据库服务和电子商务方面展示了强大的性能。

2）Oracle 简介

Oracle 公司成立于 1977 年，最初就是专门的数据库公司。

1998 年 9 月，Oracle 公司正式发布 Oracle 8i，“i”代表 Internet，这一版本中添加了大量为支持 Internet 而设计的特性。这一版本为数据库用户提供了全方位的 Java 支持。

在 2001 年 6 月的 Oracle Open World 大会中，Oracle 公司发布了 Oracle 9i，包含应用群集软件 Real Application Clusters（RAC）和商务智能（BI）功能。

2004 年 2 月，Oracle 公司发布了 Oracle 10g 版本，“g”代表 grid（网格）。这一版的最突出特性就是加入了网格计算的功能。

2013 年 6 月，Oracle 12c 正式发布，该版本提供了先进的技术堆栈管理、安全的数据库管理及企业级的服务管理，使企业能够快速实现私有云。

目前，Oracle 数据库成为世界上使用广泛的数据库系统之一。Oracle 公司在数据库领域一直处于领先地位，不仅数据库核心相当优秀，而且其相关的支持产品也相当完善和全面。Oracle 能适应 70 多种操作系统，这也是其他产品难以企及的优势。

3）DB2 简介

1970 年，IBM 公司的一位研究员德加考特发表论文，提出“关系模型”的概念。此后，他被称为“关系数据库之父”。IBM 公司投入巨资，开展包括“SystemR”和“SystemR*”项目在内的关系数据库技术的研究。13 年后，在“SystemR”和“SystemR*”项目的基础上，DB2 以 SystemR 为原型面世。

DB2 支持从 PC 到 UNIX，从中小型机到大型机，从 IBM 到非 IBM（HP 及 Sun UNIX 系统等）的各种操作系统平台。其中，服务器平台可以是 OS/400、AIX、OS/2、HP-UNIX、Sun Solaris 等操作系统，客户机平台可以是 OS/2 或 Windows、DOS、AIX、HP-UX、Sun Solaris 等操作系统。但是，DB2 服务器端的最佳运行环境还是 IBM 自己的操作系统平台 OS/400。

DB2 数据库核心又称为 DB2 通用服务器，可以运行于多种操作系统之上，它根据相应的平台环境做了调整和优化，以便达到较好的性能。由于 IBM 公司在商用服务器领域内的长期优势，在全球 500 强的企业中，超过 80% 的企业曾使用 DB2 作为数据库平台。

4）MySQL 简介

MySQL 也是一个关系型数据库管理系统，现已被 Oracle 公司收购。它与上述大型数据库相比，有不足之处，但是这丝毫没有减少它受欢迎的程度。

MySQL 运行于 Linux 操作系统之上，Apache 和 Nginx 作为 Web 服务器，MySQL 作为后台数据库，PHP/Perl/Python 作为脚本解释器。这四款软件都是免费或开源的，也就是说，企业可以不花一分钱（除人工外）就能建立起一个稳定、高速的网站系统，业内称为“LAMP”组合。因此，其以体积小、速度快、开源等特点，霸占了中小型网站相当大的市场。

（2）非关系数据库

非关系数据库也被称作 NoSQL（Not Only SQL），存储数据不以关系模型为依据，不需要固定的表格式。非关系型数据库作为关系数据库的一个补充，在日益快速发展的网站时代，发挥着高效率与高性能。

非关系型数据库的优点：

- 数据库高并发读写的需求。
- 对海量数据高效率存储与访问。
- 数据库的高扩展性与高可用性的需求。

常用的非关系数据库如 Memcached、Redis、MongoDB、HBase，我们将在后续课程中进行介绍。

1.1.4　关系数据库的基本概念

关系数据库系统是基于关系模型的数据库系统，是关系模型应用到数据库领域的实例化。它的基本概念来自于关系模型。

1. 关系数据库的基本结构

关系数据库使用的存储结构是多个二维表格，即反映事物及其联系的数据描述是以平面表格形式体现的。

在每个二维表中，每一行称为一条记录，用来描述一个对象的信息；每一列称为一个字段，用来描述对象的一个属性。数据表与数据库之间存在相应的关联，这些关联用来查询相关的数据。图 1.7 所示就是一个数据表。

关系数据库是由数据表之间的关联组成的。其中：

- 数据表通常是一个由行和列组成的二维表，每一个数据表分别说明数据库中某一特定的方面或部分的对象及其属性。
- 数据表中的行通常叫做记录或者元组，它代表众多具有相同属性的对象中的一个。
- 数据表中的列通常叫做字段或者属性，它代表相应数据库中存储对象的共有的属性。

2. 主键与外键

（1）主键

数据表中的每行记录都必须是唯一的，而不允许出现完全相同的记录，通过定义主键（主关键字，Primary Key）可以保证记录（实体）的唯一性。

键，即关键字，它是关系模型中一个非常重要的元素。

主键唯一标识表中的行数据，一个主键值对应一行数据。主键由一个或多个字段组成，其值具有唯一性，不允许取空值（NULL）。一个表只能有一个主键。

如果一个属性集能唯一地标识表的一行而又不含有多余的属性，那么这个属性集称为候选键。表中可以有多个候选键，但是只能有一个候选键可以选作表的主键，所有其他候选键称为备用键。例如，在图 1.7 所示的“学生信息统计表”中，“编号”“身份证号”或“姓名”“专业编号”都可以说是候选键，可以将“编号”定义为主键。

DB. stude. . . . 学生信息统计表

	编号	姓名	身份证号	性别	年龄	民族	专业编号
	1	王明	110626198801012811	男	22	汉	1
	2	赵玲	146225198906262638	女	21	汉	5
	3	张雨彤	130621199010033522	女	20	汉	3
	4	李新	110265198703051923	男	23	回	2
	5	夏雪	110215198403253562	女	22	汉	2
	6	王林	120026198602092813	男	24	满	4
	7	夏雪	130726199002162828	女	20	汉	5

图 1.7　学生信息统计表

（2）外键

一个关系数据库通常包含多个表，通过外键（Foreign Key）可以使这些表关联起来。

外键是用于建立和加强两个表数据之间的链接的一列或多列。通过将表中主键值的一列或多列添加到另一个表中，可创建两个表之间的链接。这个列就称为第二个表的外键。

例如，在图 1.8 所示的“专业名称表”中，字段“专业编号”是该表的主键，在图 1.7 所示的“学生信息统计表”中也有一个字段“专业编号”，则该字段称为“学生信息统计表”的外键。

DB. stude. . . bo. 专业名称表*

	专业编号	专业
	1	计算机科学与技术
	2	信息管理
	3	自动化
	4	音乐
	5	美术
	6	通信
	7	哲学

图 1.8　专业名称表

在图 1.7 和图 1.8 中，“专业名称表”就称为主表，“学生信息统计表”就称为从表。主表和从表总是成对出现的，相互之间以“外键”形成关联。

3. 数据完整性规则

为了维护数据库中的数据与现实世界的一致性，关系数据库的数据与更新操作必须遵守下列四类完整性规则。

（1）实体完整性规则

实体完整性规则要求关系中的元组在主键的属性上不能有空值。如果出现空值，那么主键值就起不到唯一标识元组的作用。

例如，在图 1.9 所示的“学生信息表”中，每个学生都有一个编号，用来唯一标识每个学生的信息记录，这个编号往往被设为该表的主键，以方便其他数据库表的关联应用。依照实体完整性规则，“编号”字段不允许为空。

DB. stude. . . . 学生信息统计表

编号	姓名	身份证号	性别	年龄	民族
1	王明	1106261988010…	男	22	汉
2	赵玲	1462251989062…	女	21	汉
3	张雨彤	1306211990100…	女	20	汉
4	李新	1102651987030…	男	23	回
5	夏雪	1102151984032…	女	22	汉
6	王林	1200261986020…	男	24	满

图 1.9　学生信息表

（2）域完整性规则

域完整性也称列完整性，指定一个数据集对某一个列是否有效或确定是否允许空值。

例如，在图 1.9 所示的“学生信息表”中，定义“性别”字段只能取值为“男”或“女”，这样该列就不会输入其他一些无效的值，如“female”“22”“苗族”等。

（3）引用完整性规则

如果两个表之间相互关联，那么引用完整性规则要求不允许引用不存在的元组。

例如，在图 1.9 所示的表中记录了所有的学生信息，在图 1.10 所示的表中记录了学生的考勤记录。

DB. student - dbo. 学生考勤表

序号	姓名	上课时间	下课时间
1	赵玲	2011-05-05 08:00:00.000	2011-05-05 17:00:00.000
2	夏雪	2011-05-09 09:00:00.000	2011-05-09 15:00:00.000
3	王菡	2011-05-09 08:30:00.000	2011-05-09 16:00:00.000

图 1.10　学生考勤表

学生“王菡”在“学生信息表”中不存在，但是在“学生考勤表”中，却有了她的出勤记录，这样的情况是不允许出现的。

（4）用户定义的完整性规则

用户定义的完整性规则是针对某一具体数据的约束条件，由应用环境决定。它反映了某一具体应用所涉及的数据必须满足的语义要求。系统提供定义和检验这类完整性的机制，以便用统一的系统方法进行处理，不再由应用程序承担这项工作。

1.2 MySQL 服务基础

MySQL 是一个真正的多线程、多用户的 SQL 数据库服务，凭借其高性能、高可靠和易于使用的特性，成为服务器领域中最受欢迎的开源数据库系统。在 2008 年以前，MySQL 项目由 MySQL AB 公司进行开发、发布和支持，之后历经 Sun 公司收购 MySQL AB 公司，Oracle 公司收购 Sun 公司的过程，目前 MySQL 项目由 Oracle 公司负责运营和维护。

本节将介绍 MySQL 的编译安装过程、服务控制方法，以及如何使用客户端工具访问 MySQL 数据库。

1.2.1 MySQL 的编译安装

为了确保 MySQL 数据库功能的完整性、可定制性，本小节将采用源代码编译的方式安装 MySQL 数据库系统。之前在学习 LAMP 时介绍过 MySQL 5.5 的编译安装，本节将选用 mysql-5.7.17.tar.gz，其官方站点为 https://dev.mysql.com/。

1. 准备工作

（1）如果 CentOS 7.3 已安装 mariadb 数据库，必须先将其卸载，然后再开始进行新下载的源码安装。

```
[root@www ~]# rpm -q mariadb
 mariadb-5.5.52-1.el7.x86_64

[root@www ~]# yum remove mariadb
[root@www ~]# yum install gcc gcc-c++ ncurses-devel
```

（2）MySQL 5.7 源码安装需要 cmake 编译安装，所以先下载最新稳定版本的 cmake 包并安装，官方下载网站：https://cmake.org/download/。

```
[root@www ~]# tar zxf  cmake-3.7.2.tar.gz
[root@www ~]# cd cmake-3.7.2
[root@www cmake-3.7.2]# ./configure
[root@www cmake-3.7.2]# gmake && gmake install
```

（3）MySQL5.7 需要 Boost 这个库，所以也需要下载安装，这里下载 1_59_0 版本，注意这个版本和 MySQL 的版本是相对应的。注意在 cmake 的时候不添加 -DWITH_BOOST 这个参数时它会报错告诉你需要 Boost 的哪个版本。

官方下载网站：http://sourceforge.net/projects/boost/files/boost/

```
[root@www ~]# tar zxf boost_1_59_0.tar.gz
[root@www ~]# mv boost_1_59_0/ /usr/local/boost
```

2. 源码编译及安装

（1）创建运行用户

为了加强数据库服务的权限控制，建议使用专门的运行用户，如 mysql。此用户不需要直接登录到系统，可以不创建宿主文件夹。

```
[root@www ~]# groupadd  mysql
[root@www ~]# useradd -M -s /sbin/nologin mysql -g mysql
```

（2）解包

将下载的 mysql 源码包解压，释放到 /usr/src 目录下，并切换到展开后的源码目录。

```
[root@www ~]# tar zxf mysql-5.7.17.tar.gz -C /usr/src
[root@www ~]# cd /usr/src/mysql-5.7.17
```

（3）配置

在内容丰富、结构庞大的企业网站平台中，可能会用到多种字符集的网页，相应地数据库系统也应该支持不同的字符集编码。在配置过程中，可以将默认使用的字符集设置为 utf8，并添加其他字符集的支持。

```
[root@www mysql-5.7.17]# cmake  -DCMAKE_INSTALL_PREFIX=/usr/local/mysql
-DSYSCONFDIR=/etc  -DDEFAULT_CHARSET=utf8 -DDEFAULT_COLLATION=utf8_general_ci
-DWITH_EXTRA_CHARSETS=all -DWITH_BOOST=/usr/local/boost
```

上述配置命令中，各选项的含义如下：

- DCMAKE_INSTALL_PREFIX：指定将 mysql 数据库程序安装到某目录下，如目录 /usr/local/ mysql。
- DSYSCONFDIR：指定初始化参数文件目录。
- DDEFAULT_CHARSET：指定默认使用的字符集编码，如 utf8。
- DDEFAULT_COLLATION：指定默认使用的字符集校对规则，utf8_general_ci 是适用于 UTF-8 字符集的通用规则。
- DWITH_EXTRA_CHARSETS：指定额外支持的其他字符集编码。
- DWITH_BOOST：指定 Boost 库的位置，5.7 版本必须添加这个参数。

（4）编译并安装

```
[root@www mysql-5.7.17]# make
[root@www mysql-5.7.17]# make install
```

3. 安装后的其他调整

（1）对数据库目录进行权限设置

```
[root@www ~]# chown -R mysql:mysql /usr/local/mysql
```

（2）建立配置文件

在 MySQL 源码目录中的 support-files 文件夹下，根据以下参考内容建立 MySQL 系统的 /etc/my.cnf 配置文件。

```
[root@www ~]# rm -rf /etc/my.cnf    // 如果原来 etc 文件夹下有 my.cnf 文件可以删除
[root@www ~]#
cp /usr/local/mysql/support-files/my-default.cnf /etc/my.cnf
```

（3）初始化数据库

为了能够正常使用 MySQL 数据库系统，应以运行用户 mysql 的身份执行初始化，指定数据存放目录等。注意最后一行会生成一个随机的临时密码，请牢记！

```
[root@www mysql-5.7.17]#
/usr/local/mysql/bin/mysqld --initialize --user=mysql --basedir=/usr/local/mysql --datadir=/usr/local/
        mysql/data
2017-02-16T08:19:09.867068Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is
        deprecated. Please use --explicit_defaults_for_timestamp server option (see documentation
        for more details).
2017-02-16T08:19:15.264290Z 0 [Warning] InnoDB: New log files created, LSN=45790
2017-02-16T08:19:15.887558Z 0 [Warning] InnoDB: Creating foreign key constraint system tables.
2017-02-16T08:19:16.032143Z 0 [Warning] No existing UUID has been found, so we assume that
        this is the first time that this server has been started. Generating a new UUID: 9b23c0fc-f420-
        11e6-8562-44a8424bf8c2.
2017-02-16T08:19:16.050948Z 0 [Warning] Gtid table is not ready to be used. Table 'mysql.gtid_
        executed' cannot be opened.
2017-02-16T08:19:16.052183Z 1 [Note] A temporary password is generated for root@localhost: ;kL;
        lnmbu9oA    // 初始 root 用户密码
```

（4）设置环境变量

为了方便在任何目录下使用 mysql 命令，需要在 /etc/profile 设置环境变量。

```
[root@www mysql-5.7.17]# echo "export PATH=$PATH:/usr/local/mysql/bin" >> /etc/ profile
[root@www mysql-5.7.17]# . /etc/profile          // 立即生效
```

4．添加系统服务

若希望添加 mysqld 系统服务，以便通过 systemctl 进行管理，可以直接使用源码包中提供的服务脚本。找到 support-files 文件夹下的 mysql.server 脚本文件，将其复制到 /etc/rc.d/init.d 目录下，并改名为 mysqld，然后再设置执行权限。

```
[root@www mysql-5.7.17]# cp support-files/mysql.server /etc/rc.d/init.d/ mysqld
[root@www mysql-5.7.17]# chmod +x /etc/rc.d/init.d/mysqld
```

将 MySQL 添加为 systemd 标准服务，方便以后使用“systemctl”命令进行管理。

```
[root@www ~]# vi /lib/systemd/system/mysqld.service
[Unit]
Description=mysqld
After=network.target

[Service]
Type=forking
ExecStart=/etc/rc.d/init.d/mysqld start
```

```
ExecReload=/etc/rc.d/init.d/mysqld restart
ExecStop=/etc/rc.d/init.d/mysqld stop

PrivateTmp=true

[Install]
WantedBy=multi-user.target
[root@www ~]# systemctl daemon-reload
[root@www ~]# systemctl enable mysqld
[root@www ~]# systemctl start mysqld
```

这样，以后就可以使用 systemctl 工具或直接调用 /etc/init.d/mysqld 脚本来控制 MySQL 数据库服务了。例如，若要启动 mysqld 服务，并查看其运行状态，可以执行以下操作。

```
[root@www ~]# systemctl start mysqld.service
[root@www ~]# systemctl status mysqld.service
mysqld.service - mysqld
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2017-02-19 19:43:37 CST; 2min 58s ago
  Process: 36550 ExecStart=/etc/rc.d/init.d/mysqld start (code=exited, status=0/SUCCESS)
 Main PID: 36555 (mysqld_safe)
   CGroup: /system.slice/mysqld.service
           ├─ 36555 /bin/sh /usr/local/mysql/bin/mysqld_safe --datad...
           └─ 36659 /usr/local/mysql/bin/mysqld --basedir=/usr/local...

Feb 19 19:43:34 www systemd[1]: Starting mysqld...
Feb 19 19:43:35 www mysqld[36550]: Starting MySQL.Logging to '/usr....
Feb 19 19:43:37 www mysqld[36550]: . SUCCESS!
Feb 19 19:43:37 www systemd[1]: Started mysqld.
Hint: Some lines were ellipsized, use -l to show in full.
 [root@www ~]# netstat -anpt | grep mysqld

tcp6       0      0 :::3306                 :::*                    LISTEN      133192/mysqld
```

MySQL 服务器默认通过 TCP 3306 端口提供服务。通过编辑 /etc/my.cnf 配置文件中 [mysqld] 配置段的“port = 3306”行，可以更改监听端口。

1.2.2　访问 MySQL 数据库

MySQL 数据库系统也是一个典型的 C/S（客户端 / 服务器）架构的应用，要访问 MySQL 数据库需要使用专门的客户端软件。在 Linux 系统中，最简单、易用的 MySQL 客户端软件是其自带的 mysql 命令工具。

1. 登录到 MySQL 服务器

经过安装后的初始化过程，MySQL 数据库的默认管理员用户名为“root”，密码为给定的随机密码。以 root 用户登录本机的 MySQL 数据库，可以执行以下操作。

```
[root@www ~]# mysql  -u root -p
Enter password:          // 根据提示输入上述随机的密码，然后修改密码。
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.7.17

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> set password =password('h@O123.com');
Query OK, 0 rows affected (0.00 sec)
```

MySQL 5.7 版本默认启用了密码增强插件 validate_password，新密码必须符合密码复杂性要求，如果在测试环境中，可以禁用此插件。

2. 执行 MySQL 操作语句

验证成功以后将会进入提示符为“mysql> ”的数据库操作环境，用户可以输入各种操作语句对数据库进行管理。每一条 MySQL 操作语句以分号“;”表示结束，输入时可以不区分大小写，但习惯上将 MySQL 语句中的关键字部分大写。

例如，以用户名 root 登录到“mysql>”环境后，执行“SHOW DATABASES;”语句可以查看当前数据库中有哪些库。

```
[root@www ~]# mysql -u root –p123456

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)

mysql>
```

3. 退出“mysql>”操作环境

在“mysql> ”操作环境中，执行“EXIT”或“QUIT”命令可以退出 mysql 命令工具，返回原来的 Shell 环境。

```
mysql> EXIT
Bye
[root@www ~]#
```

1.3 使用 MySQL 数据库

熟悉安装及访问 MySQL 数据库以后，接下来将学习使用 MySQL 数据库的基本操作，这也是在服务器运维工作中不可或缺的知识。本节中的所有数据库语句均在“mysql> ”操作环境中执行。

1.3.1 查看数据库结构

下面分别介绍查看数据库、表结构的相关操作语句。

1. 查看当前服务器中包含的库

SHOW DATABASES 语句：用于查看当前 MySQL 服务器中包含的库。经初始化后的 MySQL 服务器，默认建立了四个库：sys、mysql、information_schema 和 performance_schema（其中 mysql 库中包含了用户认证相关的表），执行以下操作可以进行查看。

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

2. 查看当前使用的库中包含的表

SHOW TABLES 语句：用于查看当前所在的库中包含的表。在操作之前，需要先使用 USE 语句切换到所使用的库。例如，执行以下操作可以显示 mysql 库中包含的所有表。

```
mysql> USE mysql;
Database changed
```

```
mysql> SHOW TABLES;
+--------------------------+
| Tables_in_mysql          |
+--------------------------+
| columns_priv             |
| db
| engine_cost              |
| event                    |
……        // 省略部分内容
| user                     |
+--------------------------+
31 rows in set (0.01 sec)
```

MySQL 数据库的数据文件存放在 /usr/local/mysql/data 目录下，每个数据库对应一个子目录，用于存储数据表文件。每一个数据表对应为三个文件，后缀名分别为“.frm”“.myd”和“.myi”。当然也有少数以 opt、csm、csv、ibd 结尾的。

3. 查看表的结构

DESCRIBE 语句：用于显示表的结构，即组成表的各字段（列）的信息。需要指定“库名 . 表名”作为参数；若只指定表名参数，则需先通过 USE 语句切换到目标库。例如，执行以下操作可以查看 mysql 库中的 user 表的结构，与直接执行“DESCRIBE mysql.user;”语句的效果相同。

```
mysql> USE mysql;
Database changed
mysql> DESCRIBE user;
+-----------------+-----------------+------+-----+---------+-------+
| Field           | Type            | Null | Key | Default | Extra |
+-----------------+-----------------+------+-----+---------+-------+
| Host            | char(60)        | NO   | PRI |         |       |
| User            | char(16)        | NO   | PRI |         |       |
| Select_priv     | enum('N','Y')   | NO   |     | N       |       |
……      // 省略部分内容
+-----------------+-----------------+------+-----+---------+-------+
45 rows in set (0.00 sec)
```

数据库目前标准的指令集是 SQL。SQL 是 Structured Query Language 的缩写，即结构化查询语言。它是 1974 年由 Boyce 和 Chamberlin 提出来的，1975—1979 年，IBM 公司研制的关系数据库管理系统原型 System R 实现了这种语言，经过多年的发展，SQL 语言得到了广泛的应用。

SQL 语言主要由以下几部分组成：

- DDL（Data Definition Language，数据定义语言）：用来建立数据库、数据库对象和定义其列，如 CREATE、ALTER、DROP。
- DML（Data Manipulation Language，数据操纵语言）：用来查询、插入、删

除和修改数据库中的数据，如 SELECT、INSERT、UPDATE、DELETE。

- DCL（Data Control Language，数据控制语言）：用来控制数据库组件的存取许可、存取权限等，如 COMMIT、ROLLBACK、GRANT、REVOKE。

1.3.2　创建及删除库和表

当需要为网站平台提供数据库服务时，如何操作来创建新的库呢？根据一份二维数据表格，如何在现有的库中创建表？对于废弃的数据表和库，又该如何进行删除？本小节主要解决这三个问题。

1. 创建新的库

CREATE DATABASE 语句：用于创建一个新的库，需指定数据库名称作为参数。例如，执行以下操作可以创建一个名为 auth 的库。

```
mysql> CREATE DATABASE auth;
Query OK, 1 row affected (0.01 sec)
```

刚创建的数据库是空的，其中不包含任何表，在 /usr/local/mysql/data 目录下会自动生成一个与新建的库名相同的空文件夹。

2. 创建新的表

CREATE TABLE 语句：用于在当前库中创建新的表，需指定数据表名称作为参数，并定义该表格所使用的各字段。基本格式如下所示：

```
CREATE TABLE 表名 ( 字段 1 名称 类型 , 字段 2 名称 类型 , …, PRIMARY KEY ( 主键名 ))
```

创建表之前，应先明确数据表格的结构、各字段的名称和类型等信息。例如，若要创建一个包含用户名、密码字串的用户验证表（表 1-4），应先分析表格结构。

表 1-4　用户验证表的内容

用户名	密码字串
zhangsan	*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
lisi	*2A032F7C5BA932872F0F045E0CF6B53CF702F2C5
wangwu	*FF8DAB1D945848EB3339D5E583B3FFF7F28AA631

上述表格中，“用户名”为不超过 16 个字符的字符串，且不能为空；“密码字串”为不超过 48 个字符的字符串（插入记录时使用 MySQL 的函数加密），默认值为空字串。考虑到字符集的兼容性，最好不要使用中文字段名，改用 user_name、user_passwd 分别表示用户名、密码字串。表格中不能拥有同名的用户，因此可将 user_name 作为主键。

针对上述分析结果，可以在 auth 库中按如下操作创建 users 表。其中，字段定义部分的 DEFAULT 语句用于设置默认的密码字串，PRIMARY 语句用于设置主键字段名。

```
mysql> USE auth;
mysql> CREATE TABLE users (user_name CHAR(16) NOT NULL, user_passwd CHAR(48)
    DEFAULT '', PRIMARY KEY (user_name));
Query OK, 0 rows affected (0.00 sec)
```

3. 删除一个数据表

DROP TABLE 语句：用于删除库中的表，需要指定“库名 . 表名”作为参数；若只指定表名参数，则需先通过执行“USE”语句切换到目标库。例如，执行以下操作可以删除 auth 库中的 users 表。

```
mysql> DROP TABLE auth.users;
Query OK, 0 rows affected (0.00 sec)
```


4. 删除一个数据库

DROP DATABASE 语句：用于删除指定的库，需要指定库名作为参数。例如，执行以下操作可以删除名为 auth 的库。

```
mysql> DROP DATABASE auth;
Query OK, 0 rows affected (0.01 sec)
```

1.3.3 管理表中的数据记录

参考上节中的步骤重新创建 auth 库和 users 表，下面将以 users 表为基础，学习向表中插入、查询、修改及删除数据记录的操作。

1. 插入数据记录

INSERT INTO 语句：用于向表中插入新的数据记录。语句格式如下所示：

```
INSERT INTO 表名 ( 字段 1, 字段 2,…) VALUES( 字段 1 的值 , 字段 2 的值 ,…)
```

执行以下操作将会向 auth 库中的 user 表插入一条记录：用户名为“zhangsan”，对应的密码为“123456”。注意：VALUES 部分的值应与前面指定的各字段逐一对应。

```
mysql> use auth;
Database changed
mysql> INSERT INTO users(user_name,user_passwd) VALUES('zhangsan', PASSWORD (
        '123456'));

Query OK, 1 row affected, 1 warning (0.19 sec)
```

在插入新的数据记录时，如果这条记录完整包括表中所有字段的值，则插入语句中指定字段的部分可以省略。例如，执行以下操作也可以向 auth 库中的 user 表插入一条新的记录：用户名为“lisi”，对应的密码为“654321”。

```
mysql> INSERT INTO users VALUES('lisi', PASSWORD('654321'));
```

```
Query OK, 1 row affected, 1 warning (0.19 sec)
```

2. 查询数据记录

SELECT 语句：用于从指定的表中查找符合条件的数据记录。MySQL 数据库支持标准的 SQL 查询语句，语句格式如下所示：

```
SELECT 字段名 1, 字段名 2,… FROM 表名 WHERE 条件表达式
```

表示所有字段时，可以使用通配符“*”，若要显示所有的数据记录则可以省略 WHERE 条件子句。例如，执行以下操作可以查看 auth 库中 users 表内的所有数据记录，其中密码字串因已加密，因此不会直接显示出实际的密码内容。

```
mysql> select * from auth.users;
+-----------+-------------------------------------------+
| user_name | user_passwd                               |
+-----------+-------------------------------------------+
| lisi      | *2A032F7C5BA932872F0F045E0CF6B53CF702F2C5 |
| zhangsan  | *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----------+-------------------------------------------+
2 rows in set (0.00 sec)
```

当需要根据特定的条件查找记录时，WHERE 条件子句则是必不可少的。例如，若要查找 users 表中用户名为“zhangsan”的记录，显示其中用户名、密码字段的信息，可以执行以下操作。

```
mysql> SELECT user_name,user_passwd FROM auth.users where user_name= 'zhangsan';
+-----------+-------------------------------------------+
| user_name | user_passwd                               |
+-----------+-------------------------------------------+
| zhangsan  | *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----------+-------------------------------------------+
1 row in set (0.00 sec)
```

3. 修改数据记录

UPDATE 语句：用于修改、更新表中的数据记录。语句格式如下所示：

```
UPDATE 表名 SET 字段名 1= 字段值 1[, 字段名 2= 字段值 2]  WHERE 条件表达式
```

执行以下操作可以修改 users 表中用户名为“lisi”的记录，将密码字串设为空值。验证记录内容可以发现 lisi 用户的密码串值已变为空白。

```
mysql> UPDATE auth.users SET user_passwd=PASSWORD('') WHERE user_name= 'lisi';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> select * from auth.users;
+-----------+-------------------------------------------+
| user_name | user_passwd                               |
```

```
+-----------+-------------------------------------------+
| lisi      |                                           |
| zhangsan  | *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----------+-------------------------------------------+
2 rows in set (0.00 sec)
```

若是在 Linux 命令行环境中执行，还可以使用 mysqladmin 工具来设置密码，但是会报一些警告。例如，执行以下操作并验证原密码后，可将数据库用户 root 的密码设置为“123457”。

```
[root@www ~]# mysqladmin -u root -p password '123457'
Enter password:
mysqladmin: [Warning] Using a password on the command line interface can be insecure.
Warning: Since password will be sent to server in plain text, use ssl connection to ensure password safety.
```


4. 删除数据记录

DELETE 语句：用于删除表中指定的数据记录。语句格式如下所示：

```
DELETE FROM 表名 WHERE 条件表达式
```

执行以下操作可以删除 users 表中用户名为“lisi”的数据记录，验证记录内容可以发现 lisi 用户的数据记录已经消失。

```
mysql> DELETE FROM auth.users WHERE user_name='lisi';
Query OK, 1 row affected (0.00 sec)
mysql> SELECT * FROM auth.users;
+-----------+-------------------------------------------+
| user_name | user_passwd                               |
+-----------+-------------------------------------------+
| zhangsan  | *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----------+-------------------------------------------+
1 row in set (0.00 sec)
```

在 MySQL 5.7 版本数据库服务器中，将 MySQL 库里面 user 表中的 password 字段修改为 authentication_string 字段，已经删除了一些空用户，提高了安全性。

```
mysql> SELECT user,host,authentication_string FROM mysql.user;
+-----------+-----------+-------------------------------------------+
| user      | host      | authentication_string                     |
+-----------+-----------+-------------------------------------------+
| root      | localhost | *F13F0EEE74714A1A9922D61FC15789AD75FE4958 |
| mysql.sys | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
+-----------+-----------+-------------------------------------------+
2 rows in set (0.00 sec)
```

本章总结

- 数据库由数据库表和其他数据对象组成。

- 经典数据模型有网状模型、层次模型和关系模型。
- 主键由一个或多个字段组成，其值具有唯一性，而且不允许取空值（NULL）。一个表只能有一个主键。
- 一个关系数据库通常包含多个表，可以通过外键将这些表关联起来。
- MySQL 是一个开源的 SQL 数据库软件，默认使用 TCP 3306 端口提供服务，配置文件是 /etc/my.cnf。
- MySQL 的基本管理操作包括查看数据库结构、创建及删除库和表、管理表中的数据记录。

本章作业

1．什么是主键、外键？

2．在编译安装 MySQL 数据库系统时，如何添加对多种字符集的支持？

3．已知 MySQL 服务器的 IP 地址为 192.168.1.241，监听的端口为 3456，授权访问的用户为 jerry，则使用 MySQL 客户端程序应如何进行连接？（提示：-P 选项指定端口；-h 选项指定目标主机地址）

4．新建 kgc 库，并在 kgc 库中创建一份 product 表（见表 1-5），录入数据并查询产品代号中包含“dev”的记录内容。

表 1-5　product 表的内容

产品代号	技术方向	培养目标
kgcdevj	Java 开发	精通 JavaEE 平台开发的软件工程师
kgcdevbd	大数据开发	精通大数据平台开发的软件工程师
kgctest	软件测试	精通系统和软件测试的测试工程师
kgcyun	云计算	精通自动化运维、云计算运维的网络工程师