第3章

RDBMS与SQL

3.1 关系数据库

1970年，IBM的关系数据库研究员，有“关系数据库之父”之称的埃德加·考特博士在《美国计算机学会通讯》刊物上发表了论文《大型共享数据库的关系模型》，文中首次提出了数据库“关系模型”的概念，奠定了关系模型的理论基础。20世纪70年代末，关系方法的理论研究和软件系统的研制均取得了很大成果，IBM公司的San Jose实验室在IBM 370系列机（图3-1）上研制的关系数据库实验系统System R历时6年获得成功。1981年，IBM公司又宣布具有System R全部特征的新的数据库产品SQL/DS问世。

图3-1 IBM 370系列机

由于关系模型简单明了，具有坚实的数学理论基础，所以SQL/DS一经推出就受到了学术界和产业界的高度重视和广泛响应，并很快成为数据库市场的主流。20世纪80年代以来，计算机厂商推出的数据库管理系统几乎都支持关系模型，数据库领域的研究工作也大都以关系模型为基础。ORACLE公司的Oracle，微软公司的SQL Server、Access，IBM公司的DB2，Sybase公司的Sybase，英孚美软件公司的Informix以及开源的MySQL等都是关系数据库。

关系数据库建立在关系模型（图3-2）基础上，借助集合代数等概念和方法来处理数据库中的数据。关系数据库同时也被组织成一组描述性表格，该表格实质是装载着数据项的特殊收集体，表格中的数据能以不同方式被存取，而不需要重新组织数据库

表格。

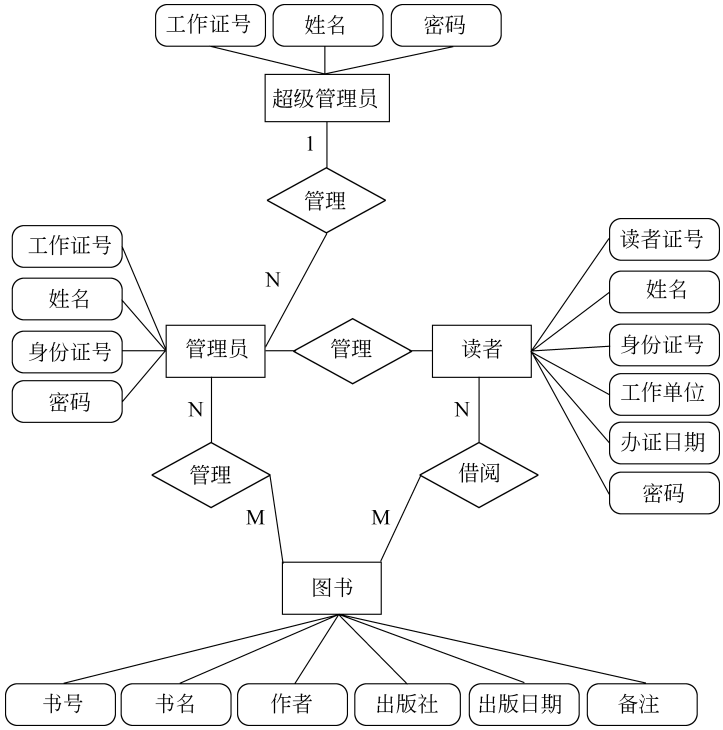

图 3-2　关系模型

(1) 关系数据库。在一个给定的应用领域中,所有实体及实体之间联系的集合构成一个关系数据库。

(2) 关系数据库的型与值。关系数据库的型称为关系数据库模式,是对关系数据库的描述,定义若干域,在这些域上定义若干关系模式。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。

① 单一的数据结构——关系。现实世界的实体以及实体间的各种联系均用关系来表示。

② 数据的逻辑结构——二维表。从用户角度看,关系模型中数据的逻辑结构是一张二维表。

关系模型的这种简单的数据结构能够表达丰富的语义,描述现实世界的实体以及实体间的各种关系。

3.2　RDBMS 的结构

关系数据库管理系统(RDBMS)由一套管理数据及操作数据的程序组成。要实现 RDBMS,至少应该建立下面 4 个组件:存储介质管理程序、内存管理程序、数据字典和查询语言。把这 4 个组件联合起来,就可以为 RDBMS 提供核心的数据管理服务和数据获

取服务。

3.2.1 存储介质管理程序

数据库系统会把数据持久地保存在磁盘或闪存盘驱动器中，其存储介质会直接与服务器或运行数据库服务的其他设备相连。比如，运行MySQL数据库[①]（图3-3为其标志）的笔记本电脑可以把数据持久地保存在本机的磁盘驱动器中。而对于大型企业来说，IT部门会搭建共享的存储空间。在这种情况下，会把整个大型磁盘阵列合起来当作一项存储资源，而数据库服务器可以从该磁盘阵列中读取数据，或是把数据保存到磁盘阵列之中。

图3-3 MySQL标志

无论使用哪种存储系统，RDBMS都需要记录每条数据的存储位置。使用磁盘及闪存盘等设备使得RDBMS的设计者能够改善数据的获取方式。

和基于文件的数据存储系统类似，RDBMS也要通过读取数据块及写入数据块的形式来运作，人们很容易就能在磁盘中创建并使用指向数据信息的索引。索引是一套包含定位信息的数据集，其中的定位信息会指明由数据库保存的那些数据块分别存储在磁盘中的位置。索引是根据数据中的某些属性编制出来的，例如，可以根据客户的ID或姓名来编制索引。每一条索引都会引用某个实体，并给出这个实体在磁盘或闪存中的存储位置，位于该位置的记录存放与本实体有关的信息。例如，“Smith，Jane 18277372”这条索引，可能是指磁盘的18277372这个位置上有个数据块，该数据块中存放了与Jane Smith有关的信息。

RDBMS存储管理程序还可以优化数据在磁盘中的排布方式，并对数据进行压缩，以节省存储空间。此外，它还能够对数据块进行复制，以防止由于磁盘中某个数据块损坏而造成数据丢失。

3.2.2 内存管理程序

RDBMS也要负责在内存中管理数据。一般来说，存储在数据库里的数据量要大于可用的内存量。因此，用户需要使用某份数据时，RDBMS的内存管理模块会将其读入，并一直保留在内存中，等到用户不再使用此数据，或是系统需要为其他数据腾出空间时，它还要负责从内存中删除该数据。由于从内存中读取数据的速度要比从磁盘中读取数据快好几个数量级，因此，RDBMS的总体性能很大程度上取决于内存管理程序能否有效地利用内存。

3.2.3 数据字典

数据字典是RDBMS的一部分，记录了与数据在数据库中的存储结构有关的信息

① MySQL由瑞典MySQL AB公司开发，现在属于ORACLE公司旗下产品，是最流行的关系数据库管理系统之一。MySQL使用的SQL语言是用于访问数据库的最常用标准化语言，由于其具有体积小、速度快、开放源码等特点，一般中小型网站的开发都选择MySQL作为网站数据库。

(图 3-4)。

数据字典包含多个层次的数据库结构信息,其中包括:

(1) 模式(schema)。

(2) 表(table)。

(3) 列(column)。

(4) 索引(index)。

(5) 约束(constraint)。

(6) 视图(view)。

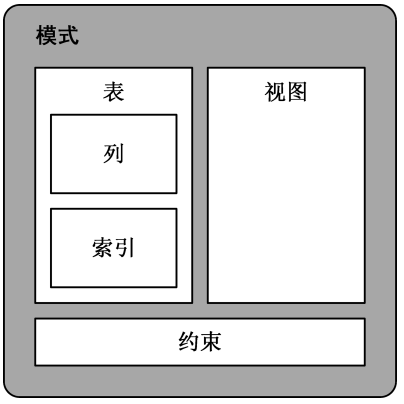

图 3-4 由数据字典管理的数据结构

模式由表、视图、索引以及所有与这套数据有关的其他结构所组成。一般来说,应该为每一种常见的数据使用方式单独创建一个模式。例如,应该给产品库存、应收账款、雇员及其福利分别创建一份模式。

表格是一种与实体有关的数据结构,而实体则用来描述一种与 RDBMS 支持的业务或操作相关联的真实事物或逻辑概念。例如,在一个描述人力资源数据的模式中,可能会出现雇员、经理及部门等实体。在一个描述库存数据的模式中,可能有仓库、产品及供应商等实体。

表格由列组成。列中含有单独的信息单元。一张存放雇员信息的表格可能包含的列有雇员姓名、街道地址、城市、省、邮政编码、出生日期及工资。每一列都会与一种数据类型相关联,该类型指出了本列能够存放什么样的数据。例如,表示雇员名字的列应该存放字符数据;表示出生日期的列应该是日期类型;表示工资的列应该是某种数字类型或货币类型。

由于索引是一种旨在改善 RDBMS 数据获取速度的数据结构,在一个存放雇员信息的表格中,可能会有一份根据雇员姓氏编制的索引,它引导我们按照雇员的姓氏快速地搜寻这张表。

所谓约束,是指一种规则,它可以进一步限制某列所能存放的数据值。与该列相关联的数据类型能够拦截类型不相符的错误数据。比如,如果程序不小心把某个数字写入雇员名字这一列,数据库就会拒绝这一操作。但是,对于存放工资的那一列来说,仅仅依靠数据类型并不能拦住某些负数,因为那些负数也是有效的数值或货币值。此时,可以给该列施加一条约束,例如规定薪水的值必须大于 0。一般来说,约束都是根据业务规则订立的,这些规则与数据所要表示的实体及操作有关。

视图是由一张或多张表格的相关列以及根据这些列算出的数值构成的,它可以限定用户所能看到的数据范围。比如,如果雇员表格中含有工资信息,可以根据该表格创建一份不含工资信息的视图。对于只需查询雇员姓名及住址的用户来说,可以使用这张视图而无须访问原来的表格。视图还可以把多张表格中的数据合并起来,比如,如果有一张表格包含了雇员的姓名,另一张表格详细描述了所有雇员在公司内的职位晋升状况,可以把这两张表格合并为一张视图。

3.3 结构化查询语言SQL

RDBMS的查询语言叫做SQL，它包含了能够执行两类操作的语句，一类是数据结构的定义操作，另一类是数据的处理操作，用于存取数据以及查询、更新和管理关系数据库系统。

SQL是高级的非过程化编程语言，允许用户在高层数据结构上工作。它不要求用户指定对数据的存放方法，也不需要用户了解具体的数据存放方式，所以具有完全不同于底层结构的不同数据库系统，可以使用相同的SQL作为数据输入与管理的接口。结构化查询语言的语句可以嵌套，这使它具有极大的灵活性和强大的功能。

SQL是1974年由博伊斯和钱伯林提出的，并首先在IBM公司研制的关系数据库系统System R上实现。由于它具有功能丰富、使用方便灵活、语言简洁易学等突出的优点，因此深受计算机工业界和计算机用户的欢迎。1980年10月，经美国国家标准局数据库委员会批准，SQL成为关系数据库语言的美国标准。同年，标准SQL公布，此后不久，国际标准化组织也作出了同样的决定。1979年，ORACLE公司首先提供商用的SQL，IBM公司在DB2和SQL/DS数据库系统中也实现了SQL。

SQL语言的核心部分相当于关系代数，但又具有关系代数没有的许多特点，如聚集、数据库更新等，它是一种综合的、通用的、功能极强的关系数据库语言。

SQL语言的特点如下。

(1) 数据描述、操纵、控制等功能一体化。

SQL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，录入数据，建立数据库，查询、更新、维护数据库，数据库重构，数据库安全性控制等一系列操作，为数据库应用系统开发提供了良好的环境。在数据库投入运行后，还可以根据需要随时修改模式，且不影响数据库的运行，从而使系统具有良好的可扩充性。

(2) 高度非过程化。

用SQL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语句的具体处理操作过程由系统自动完成。这不但大大减轻了用户负担，而且有利于提高数据独立性。

(3) 以同一种语法结构提供两种使用方式。

SQL有两种使用方式。一种是联机交互方式，即SQL能够独立地用于联机交互，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。另一种是嵌入到某种高级程序设计语言(如C、C#、Java语言等)中去使用。尽管使用方式不同，但所用语言的语法结构基本是一致的。SQL以统一的语法结构提供两种不同的操作方式，具有极大的灵活性与方便性。

(4) 语言简洁，易学易用。

尽管SQL的功能很强，但它十分简洁，完成数据定义、数据操纵、数据控制等核心功能只用了以下9个动词：CREATE、ALTER、DROP、SELECT、INSERT、UPDATE、DELETE、GRANT、REVOKE。SQL的语法接近英语口语，所以很容易学习和使用。

3.4 SQL语句的结构

SQL功能包括6部分，即数据定义、数据操纵、数据控制、数据查询、事务控制和指针控制。

3.4.1 数据定义

SQL能够定义数据库的三级模式结构，即外模式、全局模式和内模式结构。在SQL中，外模式又叫做视图，全局模式简称为模式，内模式由系统根据数据库模式自动实现。

数据定义语言(Data Definition Language，DDL)的语句包括动词CREATE、ALTER和DROP，可以在数据库中创建新表或修改、删除表，为表加入索引等。

在关系数据库的实现过程中，第一步是建立关系模式，定义基本表的结构，即确定该关系模式的属性组成，如每一属性的数据类型及数据可能的长度、是否允许为空值以及其他完整性约束条件。

SQL中的一些语句能够创建并删除集合、表格、视图、索引、约束以及其他数据结构，还有一些语句能够在表格中添加列、删除列，或是给表格设置读取权限及写入权限。

下面这条范例语句会创建一个模式。

```
CREATE SCHEMA humresc
```

下面这条范例语句可以创建一张表。

```
CREATE TABLE employees (
    emp_id int,
    emp_first_name varchar(25),
    emp_last_name varchar(25),
    emp_address varchar(50),
    emp_city varchar(50),
    emp_state varchar(2),
    emp_zip varchar(5),
    emp position_title varchar(30)
    )
```

上面的语句中并没有告诉计算机如何创建某个数据结构。也就是说，并没有命令计算机必须在某个特定的内存地址上面创建空闲的数据块，而是向RDBMS描述了所要创建的数据结构中应该包含什么样的数据。前面的语句创建了名为humreac的模式，接下来创建了一张包含8个列的表格，并将该表命名为employee。其中，varchar表示长度可变的字符类型，其后面的括号中所填的数字表示该列的最大长度。int表明emp_id这一列中的数据应该是整数类型。

3.4.2 数据操纵

有了数据库集合及相关表格后，就可以向其中添加数据并操作这些数据了。SQL的

数据操纵功能包括对基本表和视图的数据插入、删除和修改，以及强大的数据查询功能。

数据操纵语言（Data Manipulation Language，DML）的语句包括 INSERT（插入）、UPDATE（更新）、DELETE（删除）等。完成数据操纵的命令一般分为两种类型。

（1）数据检索（常称为查询）：寻找所需的具体数据。

（2）数据修改：插入、删除和更新数据。

下面这条 INSERT 语句可以向 employee 表格中插入数据。

```
INSERT INTO employee(emp_id, first_name, last_name)
    VALUES(1234, 'Jane', 'Smith')
```

这条语句会向表格中添加新行，该行的 emp_id 为 1234，first_name 为 Jane，last_name 为 Smith。在本表格中，这一行的其他列数值均为 NULL，这是一种特殊的数值，用来表示某行数据的某一列还没有指定具体的值。

通过更新语句和删除语句，用户可以修改现有各行内的数值，并移除表格中已有的数据行。

3.4.3 数据控制

SQL 的数据控制功能主要是对用户的访问权限加以控制，以保证系统的安全性。数据控制语言（Data Control Language，DCL）的语句通过 GRANT（授权）或 REVOKE（回收）实现权限控制，确定单个用户和用户组对数据库对象的访问权限。某些 RDBMS 可用 GRANT 或 REVOKE 语句控制对表单个列的访问权限。

3.4.4 数据查询

数据查询是数据库的核心操作。数据查询语言（Data Query Language，DQL）的语句也称为“数据检索语句”，用以从表中获得数据，确定数据怎样在应用程序里给出。保留字 SELECT 是 DQL（也是所有 SQL）用得最多的动词，其他 DQL 常用的保留字有 WHERE、ORDER BY、GROUP BY 和 HAVING。这些 DQL 保留字常与其他类型的 SQL 语句一起使用。

此外，事务控制语句能确保数据表的所有行及时更新。包括 COMMIT（提交）、SAVEPOINT（保存点）、ROLLBACK（回滚）语句。

指针控制语句，像 DECLARE CURSOR（声明指针）、FETCH INTO（存储过程）和 UPDATE WHERE CURRENT（更新指针位置）用于对一个或多个表单独行的操作。

SELECT 语句可以从数据库中读取数据。举例如下。

```
SELECT emp_id, first_name, last_name
    FROM employee
```

上面这条语句将会产生以下输出信息。

```
emp_id        first_name            last_name
-------------------------------------------------------------
1234          Jane                   Smith
```

SELECT、UPDATE及DELETE等数据操作语句可以表达非常复杂的操作过程，并且能够用相当复杂的逻辑来指定该操作针对的数据行。

3.5 关系数据库的ACID特征

数据库领域中的ACID是个首字母缩略词，它是指关系数据库管理系统的四项特征。

(1) A是指原子性(atomicity)，是指某个单元无法再继续细分的性质。比如从储蓄账户向支票账户转账就是一项事务，必须把事务中的每个步骤都执行完，整个事务才算完成，否则该事务就没有完成。数据库事务实际上就是一套不可分割的步骤。数据库在执行这些步骤时，必须将其视为一个无法分割的整体，如果其中某个步骤无法完成，那么整个单元内的所有步骤就都应视为没有完成。

(2) C是指一致性(consistency)，在关系数据库中，也称为严格一致性。换句话说，数据库事务绝对不会使数据库陷入那种数据完整性遭到破坏的状态。从储蓄账户向支票账户转账100美元，只会有两种结果，要么就是储蓄账户里少了100美元且支票账户里多了100美元，要么就是两个账户里的资金依然与刚开始执行事务时相同。数据库的一致性可以保证执行完转账操作之后只能产生这两种结果，绝对不会出现其他状况。

(3) I是指隔离性(isolation)，在执行完毕之前，受到隔离的事务对其他用户是不可见的。比如，在从银行的储蓄账户向支票账户转账的过程中，如果数据库正在从储蓄账户中扣款，但却还没把它打到支票账户里，那么此时就无法查询账户余额。数据库可以提供不同程度的隔离性。比如，在更新操作尚未彻底执行完时，数据库也可以把数据返回给用户，只不过此时所返回的数据并没有反映出该数据的最新值。

(4) D是指持久性(durability)。某个事务或操作一旦执行完毕，其效果就会保留下来，即便设备断电也不会受到影响。实际上，持久性就意味着数据会保存到磁盘、闪存盘或其他持久化的存储媒介之中。

关系数据库管理系统完全支持ACID事务，而NoSQL数据库有时也能提供某种程度的ACID事务。

3.6 关系数据库的三大范式

范式，即Normal Form，简称NF。设计关系数据库时，要想建立一个好的关系，必须使关系满足一定的约束条件，此约束就形成了规范。遵从不同的规范要求，设计出合理的关系数据库，这些不同的规范要求就被称为不同的范式。范式被分成几个等级，一级比一级要求严格，越高的范式数据库冗余越小。满足这些规范的数据库是简洁的、结构明晰的，同时，不会发生插入、删除和更新操作异常。

3.6.1 数据库范式分类

关系数据库的建立有六种范式，即第一范式(1NF)、第二范式(2NF)、第三范式(3NF)、巴斯-科德范式(BCNF)、第四范式(4NF)和第五范式(5NF，又称完美范式)。满

足最低要求的是第一范式(1NF)。在第一范式的基础上进一步满足更多规范要求的被称为第二范式(2NF),以此类推。

一般来说,数据库设计只需满足第三范式(3NF)就可以了。

3.6.2 第一范式(1NF)

第一范式强调每一列都是不可分割的原子数据项。

先用 Excel 模拟建立一个数据库的表(图 3-5),并在表中填入一些数据。

表 1 显然不符合第一范式。把表 1 修改一下,将“系名/系主任”列拆分成两列,这样,表 2 就遵循了第一范式(图 3-6)。

表 2 存在的问题如下。

(1) 比较严重的数据冗余,如姓名、系名、系主任列。

学号	姓名	系名/系主任	课程名称	分数
1	Ziph	信息系/何主任	Java	100
1	Ziph	信息系/何主任	C++	90
2	Marry	信息系/何主任	Java	99
2	Marry	信息系/何主任	Python	95
3	Jack	管理系/刘主任	会计	100
3	Jack	管理系/刘主任	酒店管理	88

图 3-5 数据库表 1

学号	姓名	系名	系主任	课程名称	分数
1	Ziph	信息系	何主任	Java	100
1	Ziph	信息系	何主任	C++	90
2	Marry	信息系	何主任	Java	99
2	Marry	信息系	何主任	Python	95
3	Jack	管理系	刘主任	会计	100
3	Jack	管理系	刘主任	酒店管理	88

图 3-6 数据库表 2

(2) 添加数据问题。当在数据表中添加一个新系和系主任时,比如在数据表中添加高主任管理化学系,添加后的数据表中就会多出高主任和化学系,而这两个数据并没有对应哪个学生,显然这是不合法的数据。

(3) 删除数据问题。如果 Jack 同学已经毕业多年,数据表中没有必要再保留 Jack 的相关数据。当在表 2 中删除 Jack 的相关数据后,刘主任和管理系以及会计和酒店管理专业都消失了,这显然是不合理的。

3.6.3 第二范式(2NF)

先来了解几个概念,包括函数的完全依赖、部分依赖和传递依赖。

函数依赖:即 A→B(符号→,指确定关系),如果通过 A 属性(或属性组)的值可以确定唯一 B 属性的值,则可以称 B 依赖于 A。例如,可以通过学号来确定姓名,可以通过学号和课程来确定该课程的分数等等。

(1) **完全函数依赖**。即 A→B,如果 A 是一个属性组,则 B 属性的确定需要依赖 A 属性组中的所有属性值。例如,分数的确定需要依赖学号和课程,而学号和课程可以称为一

个属性组。如果有学号没有课程，只知道是谁的分数，而不知道是哪一门课的分数。如果有课程没有学号，那只知道是哪门课程的分数，而不知道是谁的分数。所以该属性组的两个值是必不可少的。这就是完全函数依赖。

(2) **部分函数依赖**。即 A→B，如果 A 是一个属性组，则 B 属性的确定需要依赖 A 属性组中的部分属性值。例如，如果一个属性组中有两个属性值，它们分别是学号和课程名称。那姓名的确定只依赖这个属性组中的学号，与课程名称无关。简单来说，依赖于属性组中的部分成员即可成为部分函数依赖。

(3) **传递函数依赖**。即 A→B→C，传递函数依赖就是一个依赖的传递关系。通过确定 A 来确定 B，确定 B 之后就可以确定 C，三者的依赖关系就是 C 依赖于 B，B 依赖于 A。例如，可以通过学号来确定这位学生所在的系部，再通过系部来确定系主任是谁。而这三者的依赖关系就是一种传递函数依赖。

再来了解另一组概念，即码(候选码)、主属性码和非属性码。

(1) **码**。如果在一张表中，一个属性或属性组被其他所有属性所完全函数依赖，则称这个属性(或属性组)为该表的候选码，简称码。然而码又分为主属性码和非属性码。例如，没有学号和课程，无法确定分数，所以分数完全函数依赖于课程和学号。

(2) **主属性码**。主属性码也叫主码，即在所有候选码中挑选一个做主码，这里相当于主键。例如，分数完全函数依赖于课程和学号。该码属性组中的值就有课程、学号和分数，所以在三个候选码中挑选一个做主码，就可以挑选学号。

(3) **非属性码**。除主码属性组以外的属性叫做非属性码。例如，在分数完全函数依赖于课程和学号时，其中的学号已经被选为主码。那么就可以确定，除了学号以外，其他的属性值都是非属性码。也就是说，在这个完全函数依赖关系中，课程和分数是非属性码。

于是，在上述概念的基础上，就有第二范式的概念，即在 1NF 的基础上，非属性码的属性必须完全依赖于主码。或者说，在 1NF 的基础上消除非属性码的属性对主码的部分函数依赖。

还使用分数完全函数依赖于学号和课程这个函数依赖关系。此关系中非属性码为课程和分数，主码为学号。梳理清楚关系后，在 1NF 的基础上，非属性码的属性必须完全依赖于主码的第二范式。这就需要继续修改表结构了。遵循 1NF 和 2NF 的表结构如图 3-7 所示。

表3

学号	课程名称	分数
1	Java	100
1	C++	90
2	Java	99
2	Python	95
3	会计	100
3	酒店管理	88

表4

学号	姓名	系名	系主任
1	Ziph	信息系	何主任
2	Marry	信息系	何主任
3	Jack	管理系	刘主任

图 3-7　数据库表 3 和表 4

表 2 根据 1NF 和 2NF 拆分成了表 3 和表 4。这时候，表 3 中的分数就完全函数依赖

表3中的学号和课程。表4也挑选学号做主码。虽然解决了数据冗余问题，但是仅仅这样还不够，上述其他的两个问题，即数据删除和数据添加问题并没有解决。

3.6.4 第三范式(3NF)

第三范式是在2NF的基础上消除传递依赖。

上述数据表中有哪些传递依赖呢？表4中的传递依赖关系为：姓名→系名→系主任。该传递依赖关系为系主任传递依赖于姓名。为消除传递依赖，办法还是拆分表4为表5和表6(图3-8)。

表5

学号	姓名	系名
1	Ziph	信息系
2	Marry	信息系
3	Jack	管理系

表6

系名	系主任
信息系	何主任
信息系	何主任

图3-8 数据库表5和表6

把表4拆分成表5和表6后，再来分析添加和删除问题就会有不一样的结果。假设在数据表中添加高主任管理的化学系时，该数据只会添加到表6中，不会发生传递依赖而影响其他数据。假设Jack同学毕业了，要从表中删除Jack同学的相关数据，只要删除表4中的学号3数据和表5中的学号3数据即可，它们也没有传递依赖关系，同样不会影响其他数据。

3.6.5 范式的表设计

数据库的六大范式一级比一级要求严格，各种范式呈递次规范，越高的范式，数据库冗余越小。范式即是对数据库表设计的约束，约束越多，表设计就越复杂。表数据过于复杂，给后期数据库表的维护以及扩展、删除、备份等操作带来了一定的难度。所以，在实际开发中，只需要遵循数据库前面的三大范式即可，不需要额外扩展。

剖析三大范式的时候，最终版本的表结构就是表3＋表5＋表6。需要说明一个问题，这样设计表是可以的，但并不是很合理。因为建表时是有主键和外键约束的。在这三张表中，第一列默认为主键，其中主键为学号还可以接受，如果主键为系名，占用的空间就变大很多，在表的级联查询中会损耗性能。所以，一般设计表时需要主键约束，而其主键基本都是占用内容空间很小的数字。

【作　　业】

1. 1970年，在刊物《美国计算机学会通讯》上发表论文《大型共享数据库的关系模型》，首次提出数据库“关系模型”概念的是(　　)。

A. 埃德加·考特　　B. 冯·诺依曼

C. 埃德加·斯诺　　D. 艾伦·麦席森·图灵

2. 在非结构化数据库出现之前，数据库领域的研究工作大都以(　　)为基础。

A. 层次模型　　B. 关系模型　　C. 网状模型　　D. 文件模型

3. 下列数据库软件产品中，(　　)不属于关系数据库。

A. Oracle　　B. SQL Server　　C. MySQL　　D. NewSQL

4. 关系数据库也是一个被组织成一组拥有正式描述性的(　　)，其中的数据能以不同的方式被存取，而不需要重新组织数据库。

A. 标准　　B. 表格　　C. 文件　　D. 图形

5. 在关系数据库中，现实世界的实体以及实体间的各种联系均用(　　)来表示。

A. 等级　　B. 对象　　C. 动作　　D. 关系

6. 在关系数据库中，从用户角度看，关系模型中数据的逻辑结构是一张(　　)。

A. 箱线图　　B. 思维导图　　C. 二维表　　D. 立体图

7. 关系数据库管理系统(RDBMS)是一套管理数据及操作数据的程序，它至少应该实现存储介质管理程序、内存管理程序、(　　)和查询语言等4个组件。

A. 数据字典　　B. 图形界面　　C. 计算程序　　D. 优化程序

8. 大型企业的IT部门一般会搭建共享的存储空间。在这种情况下，会把整个大型(　　)合起来当作一项存储资源，而数据库服务器可以从其中读取数据，或是把数据保存到其中。

A. 软盘驱动器　　B. 磁盘阵列　　C. 闪存驱动器　　D. 磁鼓驱动器

9. MySQL是目前最流行的关系数据库管理系统之一。下列(　　)不属于MySQL的特点之一。

A. 速度快　　B. 非结构化　　C. 体积小　　D. 开放源码

10. 无论使用哪种存储系统，RDBMS都需要记录每条数据的存储位置。基于磁带的存储系统有个缺点，就是必须(　　)搜寻磁带，方能获取到待查询的数据。

A. 随机搜索　　B. 倒挡追溯　　C. 网络跟踪　　D. 从头至尾

11. (　　)是一套包含定位信息的数据集，其中的定位信息会指明由数据库保存的那些数据块分别存储在磁盘中的什么位置上。

A. 模块　　B. 程序　　C. 索引　　D. 数组

12. 作为RDBMS的一部分，(　　)记录了与数据在数据库中的存储结构有关的信息。

A. 数据字典　　B. 图形界面　　C. 计算程序　　D. 优化程序

13. 数据字典里面包含多个层次的数据库结构信息，但以下(　　)不属于其中之一。

A. 表(table)　　B. 类(class)　　C. 列(column)　　D. 索引(index)

14. (　　)是一种规则，它可以进一步限制某列所能存放的数据值。

A. 索引　　B. 秩序　　C. 纪律　　D. 约束

15. (　　)是由一张或多张表格的相关列以及根据这些列算出的数值所构成的，可以用来限定用户所能看到的数据范围。

A. 模块　　B. 程序　　C. 视图　　D. 数组

16. (　　)是一种特殊目的的编程语言，用于存取数据以及查询、更新和管理关系数据库系统。

A. 面向对象设计语言　　B. 结构化查询语言

C. 面向过程查询语言　　D. 表处理语言

17. 下列(　　)不属于SQL语句的功能之一。

A. 数据字典　　B. 数据定义　　C. 数据操纵　　D. 数据控制

18. ACID是个首字母缩略词，它是指关系数据库管理系统的四项特征，但下面(　　)

不属于这些特征之一。

A. 原子性　　B. 控制性　　C. 隔离性　　D. 持久性

19. 关系数据库建立有六种范式，即第一范式(1NF)、第二范式(2NF)、第三范式(3NF)、巴斯-科德范式(BCNF)、第四范式(4NF)和第五范式(5NF，又称完美范式)。一般来说，数据库设计只需满足(　　)就可以了。

A. 第一范式　　B. 第四范式　　C. 第六范式　　D. 第三范式

【实验与思考】 熟悉 RDBMS 与 SQL

1. 实验目的

(1) 熟悉关系模型和关系数据库的发展历史。

(2) 熟悉 RDBMS 结构，了解数据库 SQL 语言及其语句结构。

(3) 掌握关系数据库的 ACID 特征，了解关系数据库的三大范式。

2. 工具/准备工作

开始本实验之前，请认真阅读课程的相关内容。

需要准备一台带有浏览器，能够访问因特网的计算机。

3. 实验内容与步骤

(1) 请结合查阅相关文献资料，为“关系数据库”给出一个权威性的定义。

答：_____

_____

_____

这个定义的来源是：_____

(2) 请仔细阅读本章课文，熟悉关系数据模型，熟悉关系数据库。在此基础上，撰写短文，讨论“关系数据库管理系统(RDBMS)的 4 个必备组件”。

---------- 请将短文另外附纸粘贴于此 ----------

(3) 写出一条 SQL 数据操作语言的范例语句。

答：_____

_____

_____

(4) 写出一条 SQL 数据定义语言的范例语句。

答：_____

_____

_____

4. 实验总结

5. 实验评价(教师)