
视频讲解

第3章

关系数据库基本理论

关系数据库应用数学方法来处理数据库中的数据。最早将这类方法用于数据处理的是1962年CODASYL发表的“信息代数”,以及1968年David Child在IBM7090机上实现的集合论数据结构,但系统地、严格地提出关系模型的是美国IBM公司的E. F. Codd。

20世纪70年代末期,关系方法的理论研究和软件系统研制均取得了显著成果。IBM公司San Jose实验室在IBM370系列机器上成功研发了关系数据库实验系统System R。基于此,IBM于1981年推出了具备System R所有特性的数据库产品SQL/DS。与此同时,美国加州大学伯克利分校也开发了INGRES关系数据库实验系统,后来由INGRES公司将其商业化。

1976年后出现实验性及商品化系统System-R、Ingres、QBE等。关系数据库应用数学方法来处理数据库中的数据。由于具有数据结构简单、用户使用方便、功能强、数据独立性高和理论基础深等优点,20世纪70年代末以后问世的产品绝大多数是关系模型,并逐渐替代网状模型、层次模型数据库系统而成为主流数据库系统。目前关系数据库系统已经成熟,其产品发展表现在对多机种多操作系统的适应、查询语言标准化、提供多种开发工具、具有分布式功能、有较好的开放性以及提供多媒体管理、知识管理和工程管理的功能方面。

30多年来,关系数据库系统的研究和开发取得了辉煌的成就。关系数据库系统从实验室走向了社会,成为目前应用最广泛的数据库系统,大大促进了数据库应用领域的扩大和深入。

本章主要介绍关系模型的概念、关系模型的三类完整性、关系代数的基本运算,最后简要介绍关系演算。

3.1 关系模型概述

关系数据库系统是以关系模型为基础构建的数据库系统。第2章初步介绍了关系模型,了解了关系模型的基本术语。本章将较深入讲解关系模型,它是关系数据库系统的基础。

关系模型由以下三个基本部分组成:关系数据结构、关系操作以及关系中的完整性约束规则。关系数据结构简单清晰,在关系模型中,现实世界的实体及其之间的联系均用关系表示,从用户视角看,关系就是由行和列组成的二维表。

1. 关系操作

关系模型规定了关系操作的能力,但并不指定RDBMS(关系数据库管理系统)语言的具体语法,允许不同的RDBMS定义和开发各自的语言来实现这些操作。

1）基本的关系操作

关系数据库中的核心内容是关系，即二维表。对这样一张表的使用主要包括按照某些条件获取相应行、列的内容，或者通过表之间的联系获取两张表或多张表相应的行、列内容。关系操作的对象是关系，操作结果同样生成新的关系。

关系模型中常用的关系操作包括查询(Query)操作和更新操作[包括插入(Insert)、删除(Delete)、修改(Update)]两部分。

关系操作分为传统的集合运算和专门的关系运算两部分。传统的集合运算包括并(Union)、差(Except)、交(Intersection)、笛卡儿积。专门的关系运算包括选择(Select)、投影(Project)、连接(Join)、除(Divide)。

选择操作是指在关系中选择满足某些条件的元组(行)。

投影操作是在关系中选择若干属性列组成新的关系。投影之后不仅取消了原关系中的某些列，而且还可能取消某些元组。这是因为取消了某些属性列后，可能出现重复的行，应该取消这些完全相同的行。

连接操作是将不同的两个关系连接成为一个关系。对两个关系的连接其结果是一个包含原关系所有列的新关系。新关系中属性的名字是原有关系属性名加上原有关系名作为前缀。这种命名方法保证了新关系中属性名的唯一性，尽管原有不同关系中的属性可能是同名的。新关系中的元组是通过连接原有关系的元组而得到的。

2）关系数据语言的分类

早期的关系操作能力通常用代数方式或逻辑方式来描述，分别称为关系代数和关系演算。关系代数是用集合论中的关系运算来表达查询要求的方式。关系演算是以数理逻辑中的谓词演算来表达查询要求的方式。关系演算又可按谓词变元的基本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。若在关系演算中，谓词变元的基本对象是元组变量，则称之为元组关系演算；若谓词变元的基本对象是域变量，则称之为域关系演算。关系代数、元组关系演算和域关系演算三种语言在表达能力上是完全等价的。

随着关系模型的不断完善，关系模型另外还有一种介于关系代数和关系演算之间的语言——SQL(Structured Query Language，结构化查询语言)。SQL 不仅具有丰富的查询功能，而且具有数据定义和数据控制功能，是集查询语言、数据定义语言(DDL)、数据操纵语言(DML)和数据控制语言(DCL)于一体的关系数据库语言。它充分体现了关系数据语言的特点和优点，是关系数据的标准语言。

关系操作有三种不同的描述方式：关系代数、关系演算和结构化查询语言(SQL)，具体分类图如图 3-1 所示。

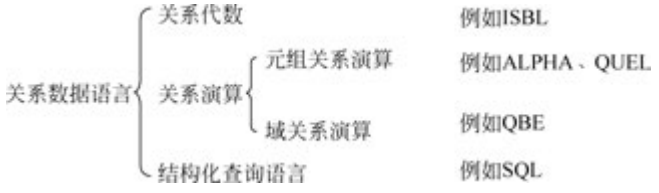

图 3-1　关系操作分类图

2. 关系中的完整性约束规则

为了维护数据库中数据与现实世界的一致性，对关系数据库的插入、删除和修改操作必须有一定的约束条件。数据的完整性约束是指在给定的数据模型中，数据及其联系所遵守的一

组通用的完整性规则，以确保数据库中数据的一致性和正确性。

例如，学生的学号必须唯一，学生的性别只能是男或女，学生的成绩为 0～100，所选修的课程必须是已经开设的课程等。

在关系数据模型中一般将数据完整性分为三类：

(1) 实体完整性。

(2) 参照完整性。

(3) 用户定义完整性。

其中实体完整性和参照完整性是关系数据模型必须满足的完整性约束条件。

3.2　关系数据模型

在关系数据模型中，数据的逻辑结构为满足一定条件的二维表。表具有固定的列数和任意的行数，在数学上称为“关系”，即将现实世界中的实体以及实体间的各种联系均用单一的关系来表示。

下面以学生-课程实体之间的联系为例，来说明现实世界中的实体以及它们之间的联系如何转换为二维表(即关系)，“学生-课程”关系框架如图 3-2 所示。

Sno	Sname	Sex	Age	Dept
学号	学生姓名	性别	年龄	系别

(a) “学生”关系

Cno	Cname	Ct
课程号	课程名	课时

(b) “课程”关系

Sno	Cno	Score
学号	课程号	成绩

(c) “选修课”关系

图 3-2　“学生-课程”关系框架

在层次模型和网状模型中，实体之间的联系是通过指针来实现的。而在关系数据模型中，实体之间的联系是通过二维表中公共属性值建立起来的联系来实现的。

从以上关系的框架中，可以很容易看出哪两个关系之间有联系。例如：

(1) “学生”关系和“选修课”关系有公共的属性“学号”，则表明这两个关系有联系。

(2) 而“课程”关系和“选修课”关系有公共的属性“课程号”，则表明这两个关系也有联系。

(3) 至于元组之间的联系，则与具体的数据有关。只有在公共属性上具有相同属性值的元组之间才有联系。

由上例可以看出，在一个关系中可以存放两类信息：

(1) 描述实体本身的信息。

(2) 描述实体(关系)之间的联系的信息。

所以，在建立关系模型时，只要把所有的实体及其属性用关系框架来表示，同时把实体之间的联系也用关系框架来表示，就可以得到一个关系模型。

3.2.1　关系数据结构和基本术语

关系数据模型有很多概念，如关系、元组、属性、关键字、域、分量、关系模式等，本节以图 3-3 为例对关系数据模型中涉及的其他概念进行解释。

1）候选键

若关系中的一个或多个属性的组合能唯一地标识一个元组，则称该属性或属性组合为候选键(Candidate Key)。每个关系中可以有多个候选键。

例如，在图 3-3(a)中，“学号”或“图书证号”都能唯一地标识一个元组，则“学号”和“图书证号”都可作为“学生”关系的候选键。在图 3-3(b)中，只有属性组“学号”和“课程号”才能唯一的标识一个元组，则候选键为(学号，课程号)。

学号	姓名	性别	年龄	图书证号	所在系
S3001	张明	男	22	B20050101	外语
S3002	李静	女	21	B20050102	外语
S4001	赵丽	女	21	B20050301	管理

(a)“学生”表

学号	课程号
S3001	C1
S3001	C2
S3001	C1
S4001	C3

(b)“选课”表

图 3-3 “学生-选课”关系举例

2）主键

如果一个关系中有多个候选键，可以从中选择一个作为查询、插入或删除元组的操作变量，被选用的候选键称为主关键字，或简称为主键(Primary Key)、关键字、主码。每个关系都有并且只有一个主键，通常用较小的属性组合作为主键。

例如，在图 3-3(a)中，选定“学号”作为数据操作的依据，则“学号”为主键。在图 3-3(b)中，主键为(学号，课程号)。

3）主属性

包含在任何候选键中的各属性称为主属性(Prime attribute)。

例如，在图 3-3(a)中，“学号”和“图书证号”为主属性。在图 3-3(b)中，“学号”和“课程号”为主属性。

4）非主属性

不包含在任何候选键中的属性称为非主属性(No primary attribute)。

例如，在图 3-3(a)中，“姓名”“性别”“年龄”“所在系”为非主属性。在图 3-3(b)中，无非主属性。

5）全键

在最极端的情况下，关系模式的所有属性是这个关系模式的候选键，称为全键(All-key)。例如，在图 3-3(b)中，属性组“学号”和“课程号”是“选课”表的候选键，称为全键。

6）外键

如果一个属性组不是所在关系的主键，但是其他关系的主键，则该属性组称为外键(Foreign Key)。例如，在图 3-3(b)中，“选课”表中的“学号”在“学生”表中为主键，所以“学号”

在“选课”表中为外键。

3.2.2　关系的数学定义

由于关系的概念来源于数学，关系数据模型是在集合代数的基础上建立起来的，所以有必要从数学的角度对关系的数据结构及其基本概念进行论述。

1. 域

定义 3.1　域(Domain)是一组具有相同数据类型的值的集合。

例如，自然数、整数、实数、{0,5}、大于 0 且小于 50 的自然数、长度小于 20 字节的字符串集合等，都可以是域。

2. 笛卡儿积

定义 3.2　给定一组域 $D_1,D_2,\cdots,D_n$，这些域中可以是相同的域。$D_1,D_2,\cdots,D_n$ 的笛卡儿积(Cartesian Product)为

$$D_1\times D_2\times\cdots\times D_n=\{(d_1,d_2,\cdots,d_n)\mid d_i\in D_i,i=1,\cdots,n\}$$

其中每个$(d_1,d_2,\cdots,d_n)$叫作一个 n 元组(n-tuple)或简称元组(Tuple)，元组中的每一个值 d_i 叫作一个分量(Component)。

这些域中可以存在相同的域，例如 D_1 和 D_2 可以是相同的域。

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

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

笛卡儿积可以表示为一个二维表。表中的每行对应一个元组，表中的每列的值来自一个域。例如，给出如下 3 个域：

D_1＝学生集合＝{李琳，王明，林丽}

D_2＝课程集合＝{英语，高数，政治}

D_3＝成绩集合＝{合格，不合格}

则 D_1、D_2、D_3 的笛卡儿积为：

$D_1\times D_2\times D_3$＝{(李琳，英语，合格)，(李琳，英语，不合格)，
(李琳，高数，合格)，(李琳，高数，不合格)，
(李琳，政治，合格)，(李琳，政治，不合格)，
(王明，英语，合格)，(王明，英语，不合格)，
(王明，高数，合格)，(王明，高数，不合格)，
(王明，政治，合格)，(王明，政治，不合格)，
(林丽，英语，合格)，(林丽，英语，不合格)，
(林丽，高数，合格)，(林丽，高数，不合格)，
(林丽，政治，合格)，(林丽，政治，不合格)}

$D_1\times D_2\times D_3$ 的基数　$M=3\times3\times2=18$

一共有 18 个元素，每个元素为一个元组，每个元组分别包含学生姓名、课程名称、成绩 3 个分量。这些元组构成了二维表的形式。

如果一个关系的元组个数是无限的，则称为无限关系；如果一个关系的元组个数是有限的，则称为有限关系。由于计算机存储系统的限制，一般不去处理无限关系，而只考虑有限关系。

注意：笛卡儿积不满足交换率，即笛卡儿积的元组有序。

3. 关系

定义 3.3 $D_1\times D_2\times\cdots\times D_n$ 的子集叫作在域 $D_1,D_2,\cdots,D_n$ 上的关系，可记作 $R(D_1,D_2,\cdots,D_n)$，R 为关系名，n 是关系的目或度(degree)。

关系(Relation)中的每个元素是关系中的元组，通常用 t 表示。

(1) 当 $n=1$ 时，称该关系为单元关系(Unary relation)。

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

无限关系在数据库系统中是无意义的，所以关系必须是笛卡儿积的有限子集。它对应一张二维表，表中的每行对应一个元组，表中的每列对应一个域。由于域可以相同，为了加以区分，必须给每列起一个名字，称为属性(Attribute)。n 目关系必有 n 个属性。由于笛卡儿积不满足交换律，根据定义，$(d_1,d_2,d_3,\cdots,d_n)\neq(d_2,d_1,d_3,\cdots,d_n)$。当给每列附加一个属性名后，关系元组的有序性便可取消，即 $(d_1,d_2,d_3,\cdots,d_n)=(d_2,d_1,d_3,\cdots,d_n)$。

一般来说，$D_1,D_2,\cdots,D_n$ 的笛卡儿积是没有实际语义的，只有它的某个子集才有实际含义。

例如上一个例子中，取笛卡儿积 $D_1\times D_2\times D_3$ 的子集如表 3-1 所示。

表 3-1 $D_1\times D_2\times D_3$ 的子集

姓　名	课　程	成　绩
李琳	英语	合格
李琳	高数	合格
李琳	政治	合格
王明	英语	合格
王明	高数	不合格
王明	政治	合格
林丽	英语	不合格
林丽	高数	合格
林丽	政治	合格

3.2.3 关系的性质

尽管关系与二维表格、传统的数据文件是非常类似的，但它们之间又有重要的区别。严格地说，关系是一种规范化了的二维表中行的集合，为了使相应的数据操作简化，在关系模型中，对关系作了种种限制，关系有以下性质：

(1) 列是同质的，即每一列中的分量是同一数据类型，来自同一个域，为元组的一个属性。

(2) 不同列的数据可以来自同一个域，称其中的每一列为一个属性，不同的属性要给予不同的属性名。

例如，表 3-2 中所示“职业”关系，“职业”与“兼职”是两个不同的属性，但它们取自同一个域，即职业＝{教师，工人，辅导员}。

表 3-2 “职业”关系

姓　名	职　业	兼　职
张伟	教师	辅导员
王丽	工人	教师
刘英	教师	辅导员

（3）尽管从逻辑上说，列的顺序无关紧要，但在数据库的设计和使用中，保持一致的列顺序有助于提高效率和用户体验。

例如，对表 3-3 所示关系 $T1$ 作列次序的交换时，无任何影响，交换后的关系 $T1$ 如表 3-4 所示。

表 3-3　关系 *T1*

姓　　名	性　　别
李伟	男
王月	女
刘洋	男

表 3-4　连同属性名一起交换的结果

性　　别	姓　　名
男	李伟
女	王月
男	刘洋

如果关系 $T1$ 作列次序的交换时，不交换属性名，只交换属性列中的值时，则得到如表 3-5 所示，此时，关系 $T1$ 出现错误。

表 3-5　不交换属性名的结果

姓　　名	性　　别
男	李伟
女	王月
男	刘洋

（4）任意两个元组的候选键不能相同。也就是表中的任意两行不能相同，即一个关系不能有相同的元组。因为数学上集合中没有相同的元素，而关系是元组的集合，所以作为集合元素的元组应该是唯一的。

（5）行的顺序无所谓，即行的次序可以任意交换。因为集合中的元素是无序的，所以作为集合元素的元组也是无序的。虽然逻辑上元组无序，但在物理存储上行的顺序可能会影响查询性能，根据关系的这个性质，可以改变元组的顺序使其具有某种排序，然后按照顺序查询数据，可以提高查询速度。

（6）每个分量必须取原子值，即每个分量应该是不可再分割的基本数据单元，是一个确定的值，而不是值的集合，也不能存在“表中有表”。满足此条件的关系称为规范化关系，否则称为非规范化关系。

例如，在表 3-6 中，籍贯含有省、市/县两项，出现了“表中有表”的现象，则为非规范化关系，而把籍贯分成省、市/县两列，将其规范化，如表 3-7 所示。

表 3-6　不规范的关系

姓　　名	籍　　贯	
	省	市/县
王洪	黑龙江	齐齐哈尔
李利	吉林	长春

表 3-7 规范后的关系

姓　　名	省	市/县
王洪	黑龙江	齐齐哈尔
李利	吉林	长春

3.3 关系模型的三类完整性

关系模型中有三类完整性约束：实体完整性、参照完整性和用户定义完整性。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，被称作关系的两个不变性，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。

3.3.1 实体完整性

规则 3.1 实体完整性规则 若属性(指一个或一组属性)A 是基本关系 R 的主属性，则 A 不能取空值。

在关系数据库中，空值(NULL)代表未知或未赋值的状态。

实体完整性规则是为了确保主键能够始终唯一标识关系中的每个元组，若取空值，便失去唯一元组功能。

例如，在关系模式学生(学号，姓名，性别，年龄，籍贯，专业名称)中，“学号”是主键。根据实体完整性约束规则，“学号”不能取空值。

在关系模式选课(学号，课程号，成绩)中，(学号，课程)为主键，所以这两个属性均不能取空值。

对于实体完整性规则说明如下：

(1) 实体完整性规则是针对基本关系而言的。一个基本表通常对应现实世界的一个实体集，如学生关系对应于学生的集合。

(2) 现实世界中的实体是可区分的，即它们具有某种唯一性标识，如每个学生都是独立的个体，是互不一样的。

(3) 相应地，关系模型中以主键作为唯一性标识。

(4) 主键中的属性即主属性不能取空值。如果主属性取空值，就说明存在某个不可标识的实体，即存在不可区分的实体，这与第(2)点相矛盾，因此这个规则称为实体完整性。

实体完整性是关系模型必须满足的完整性约束条件，关系数据库管理系统可以通过在关系中对属性设置主键来实现实体完整性(非主属性也可以说明是唯一的和非空值的)。

注意：实体完整性规则规定基本关系的所有主键的各属性都不能取空值，而不仅是主键整体不能取空值。

3.3.2 参照完整性

在关系模型中，实体及实体间的联系通过关系和外键约束来描述。这样就自然存在着关系与关系之间的相互引用和相互制约。

下面，通过 3 个例子来了解一下两个关系之间以及两个以上关系之间属性的引用。

【例 3-1】 学生实体和专业实体可以用下面的关系模式来表示：

学生(学号,姓名,性别,年龄,专业号)
专业(专业号,专业名)

这两个关系之间存在着属性的引用,即“学生”关系引用了“专业”关系的主键“专业号”。在这种情况下,“专业号”是“学生”关系中的外键,意味着“学生”关系中的“专业号”必须匹配“专业”关系中存在的某条记录的“专业号”。也就是说“学生”关系中的某个属性的取值需要参照“专业”关系的属性取值。

【例 3-2】 学生、课程、学生与课程之间的多对多联系可以如下 3 个关系表示:

学生(学号,姓名,性别,年龄,专业号)
课程(课程号,课程名,学分)
选课(学号,课程号,成绩)

这 3 个关系之间也存在着属性的引用,即“选课”关系引用了“学生”关系的主键“学号”和“课程”关系的主键“课程号”。这时“学号”和“课程号”属性是“选课”关系的外键。同样,“选课”关系中的“学号”值必须是确实存在的学生的学号,即“学生”关系中有该学生的记录;“选课”关系中的“课程号”值也必须是确定存在的课程的号,即“课程”关系中有该课程的记录。换句话说,“选课”关系中某些属性的取值需要参照其他关系的属性取值。

不仅两个或两个以上关系间可以存在引用关系,同一关系内部属性间也可能存在引用关系。

【例 3-3】 在学生(学号,姓名,性别,年龄,专业号,班长)关系中,“学号”属性是主键,“班长”属性引用了同一关系“学号”属性,即班长必须是另一个学生的学号,这意味着“班长”字段也是一个外键。

这 3 个例子说明关系与关系之间存在着相互引用、相互约束的情况。需要指出的是,外键并不一定要与相应的主键同名,如例 3-3 中“学生”关系的主键为“学号”,外键为“班长”。

参照完整性规则就是定义外键与主键之间的引用规则。

规则 3.2　参照完整性规则　若属性(或属性组)F 是基本关系 R 的外键,它与基本关系 S 的主键 K_s 相对应(基本关系 R 和 S 不一定是不同的关系),则对于 R 中每个元组在 F 上的值必须为:

(1) 或者取空值(F 的每个属性值均为空值)。

(2) 或者等于 S 中某个元组的主键值。

例如,在例 3-1 中,按照参照完整性规则,“学生”关系中每个元组的“专业号”属性只能取下面两类值:

(1) 空值,表示尚未给该学生分配专业。

(2) 非空值,这时该值必须是专业关系中某个元组的“专业号”值,表示该学生不可能被分配到一个不存在的专业中。

在例 3-2 中,按照参照完整性规则,“学号”和“课程号”属性也可以取两类值:空值或目标关系中已经存在的值。但由于“学号”和“课程号”是“选课”关系中的主属性,按照实体完整性规则,它们均不能取空值。所以“选课”关系中的“学号”和“课程号”属性实际上只能取相应被参照关系中已经存在的主键值。

在参照完整性规则中,R 与 S 可以是同一个关系。例如,在例 3-3 中,按照参照完整性规则,“班长”属性可以取两类值:

(1) 空值,表示该学生所在班级尚未选出班长。

(2) 非空值,这时该值必须是本关系中某个元组的学号值。

【例 3-4】 如图 3-4 所示，“学生”关系中某个学生(如 S1001 或 S2001)“所在系”属性的取值，必须在参照的“系别”关系中“所在系”主键的值中能够找到，否则表示把该学生分配到一个不存在的系中，显然不符合语义。如果某个学生(如 S2011)“所在系”取空值，则表示该学生尚未分配到任何一个系；否则，“学生”关系的“所在系”属性只能取“专业”关系中某个元组的“专业号”值。

学号	姓名	性别	年龄	所在系
S1001	王洪	男	22	计算机
S2001	刘明	男	21	信息
⋮	⋮	⋮	⋮	⋮
S2011	王丽	女	21	信息

(a) “学生”表

所在系	系地址
计算机	1号楼
信息	2号楼

(b) “系别”表

图 3-4 参照关系举例

除此之外，不同的关系数据库系统由于应用环境的不同，往往还需要一些特殊的约束条件，这就是用户定义完整性。

3.3.3 用户定义完整性

任何关系数据库系统都应该支持实体完整性和参照完整性。这是关系模型所要求的，也是关系模型固有的特性。

用户定义完整性就是针对某一具体关系数据库的唯一约束条件。它反映某一具体应用所涉及的数据必须满足的语义要求。关系模型应提供定义和检验这一类完整性的机制，以便用统一的系统的方法处理它们，而不是由应用程序来承担这一功能。

在用户定义完整性中最常见的是限定属性的取值范围，即对值域的约束，包括说明属性的数据类型、精度、取值范围、是否允许空值等。所以在用户定义完整性中最常见的是域完整性约束。

【例 3-5】 如图 3-4 所示，“学生”表中“学号”属性为字符型；“姓名”属性为字符型，不允许为空；“性别”属性为字符型，取值范围为(“男”“女”)；“年龄”属性为整型，取值范围(16～26)；“所在系”属性为字符型，允许为空。

3.4 关系代数的基本运算

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

关系运算的特点是集合操作方式，即操作的对象和结果都是集合，故这种操作方式称为一次一集合方式。相应地，非关系数据模型的数据操作方式则为一次一记录方式。

关系代数的运算对象是关系，运算结果亦为关系，关系代数运算符如表 3-8 所示。

表 3-8　关系代数运算符

运　算　符		含　义	运　算　符		含　义
集合运算符	∪	并	比较运算符	>	大于
	−	差		≥	大于或等于
	∩	交		<	小于
	×	笛卡儿积		≤	小于或等于
专门的关系运算符	σ	选择		=	等于
	π	投影		<>	不等于
	⋈	连接	逻辑运算符	¬	非
	÷	除		∧	与
				∨	或

关系代数的运算按运算符的不同可分为以下两类。

(1) 传统的集合运算。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”方向即行的角度进行。传统的集合运算包括并运算、交运算、差运算和笛卡儿积运算。

(2) 专门的关系运算。专门的关系运算除了把关系看成元组的集合，还通过运算表达了查询的要求。专门的关系运算不仅涉及行，还包括列的选择，这使得它不同于传统的集合运算。比较运算符和逻辑运算符用来辅助专门的关系运算符进行操作。专门的关系运算包括选择运算、投影运算、连接运算和除运算。

任何一种运算都是通过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。

3.4.1　传统的集合运算

传统的集合运算是二目运算，包括并(Union)、交(Intersection)、差(Except)、笛卡儿积(Extended Cartesian Product)4 种运算。

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，t 是元组变量，$t \in R$ 表示 t 是 R 的一个元组。

1. 并

设 R 和 S 都是 n 目关系，而且两者各对应属性的数据类型相同，则 R 和 S 的并定义为

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

$R \cup S$ 的结果仍为 n 目关系，由属于 R 或属于 S 的元组组成。

【例 3-6】 教务处印发选课表给各系填写该系学生选课情况。表格收回后需要综合为一个总表。这可以理解为关系的并运算。

2. 交

设 R 和 S 都是 n 目关系，而且两者各对应属性的数据类型相同，则 R 和 S 的交定义为

$$R \cap S = \{t \mid t \in R \wedge t \in S\}$$

$R \cap S$ 的结果仍为 n 目关系，由既属于 R 又属于 S 的元组组成。

【例 3-7】 根据某系学生名册和全校学生运动员名册产生该系学生运动员表格，这可以理解为关系的交运算。

3. 差

设 R 和 S 都是 n 目关系，而且两者各对应属性的数据类型相同，则 R 和 S 的差定义为

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

$R-S$ 的结果仍为 n 目关系，由属于 R 而不属于 S 的所有元组组成。

【例 3-8】 根据学生名册和学生运动员名册产生学生非运动员名册，这可以理解为关系的差运算。

4. 笛卡儿积

设 R 是 n 目关系，S 是 m 目关系，R 和 S 的笛卡儿积定义为

$$R \times S=\{\widehat{t_r t_s} \mid t_r \in R \wedge t_s \in S\}$$

$R \times S$ 是一个 $(m+n)$ 目关系，前 n 列是关系 R 的属性，后 m 列是关系 S 的属性。每个元组的前 n 个属性是关系 R 的一个元组，后 m 个属性是关系 S 的一个元组。若关系 R 有 p 个元组，关系 S 有 q 个元组，关系 $R \times S$ 有 $p \times q$ 个元组，且每个元组的属性为 $(m+n)$ 个。

【例 3-9】 两队游泳运动队均由 3 名队员组成。现作循环比赛，赛事表可看成是两队名单的笛卡儿乘积。

【例 3-10】 设有关系 R 和 S 如图 3-5，求 $R \cup S$，$R \cap S$，$R-S$，$R \times S$。

图 3-5(a)和图 3-5(b)分别为两个关系 R 和 S，图 3-5(c)为关系 R 和 S 的并，图 3-5(d)为关系 R 和 S 的交，图 3-5(e)为关系 R 和 S 的差，图 3-5(f)为关系 R 和 S 的笛卡儿积。

A	B	C
a_1	b_1	c_1
a_1	b_2	c_2
a_2	b_2	c_1

(a) R

A	B	C
a_1	b_2	c_2
a_1	b_3	c_2
a_2	b_2	c_1

(b) S

A	B	C
a_1	b_1	c_1
a_1	b_2	c_2
a_2	b_2	c_1
a_1	b_3	c_2

(c) $R \cup S$

A	B	C
a_1	b_2	c_2
a_2	b_2	c_1

(d) $R \cap S$

$R.A$	$R.B$	$R.C$	$S.A$	$S.B$	$S.C$
a_1	b_1	c_1	a_1	b_2	c_2
a_1	b_1	c_1	a_1	b_3	c_2
a_1	b_1	c_1	a_2	b_2	c_1
a_1	b_2	c_2	a_1	b_2	c_2
a_1	b_2	c_2	a_1	b_3	c_2
a_1	b_2	c_2	a_2	b_2	c_1
a_2	b_2	c_1	a_1	b_2	c_2
a_2	b_2	c_1	a_1	b_3	c_2
a_2	b_2	c_1	a_2	b_2	c_1

(f) $R \times S$

A	B	C
a_1	b_1	c_1

(e) R-S

图 3-5 传统集合运算举例

3.4.2 专门的关系运算

专门的关系运算包括选择、投影、连接、除运算等。为了方便叙述，先引入几个记号。

(1) 关系模式 $R(A_1, A_2, \cdots, A_n)$。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。$\overline{A}$ 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。

(3) R 为 n 目关系，S 为 m 目关系。$t_r \in R$，$t_s \in S$，$\widehat{t_r t_s}$ 称为元组的连接(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。当 $t[X]=x$ 时，x 在 R 中的像集(ImagesSet)定义为

$$Z_x=\{t[Z] \mid t \in R, t[X]=x\}$$

它表示 R 中属性组 X 上值为 x 的诸元组在 Z 上分量的集合。

例如，表 3-9 关系 R 中，

表 3-9　关系 *R*

X	*Z*
x_1	Z_1
x_1	Z_2
x_1	Z_3
x_2	Z_2
x_2	Z_3
x_3	Z_1
x_3	Z_3

x_1 在 R 中的像集 $Z_{x_1}=\{Z_1,Z_2,Z_3\}$，x_2 在 R 中的像集 $Z_{x_2}=\{Z_2,Z_3\}$，x_3 在 R 中的像集 $Z_{x_3}=\{Z_1,Z_3\}$。

下面给出这些专门的关系运算的定义。

1. 选择

选择(Selection)又称为限制(Restriction)，它是在关系 R 中选择满足给定条件的诸元组，形成一个新的关系。选择运算表示为：

$$\sigma_F(R)=\{t \mid t \in R \wedge F(t)='真'\}$$

其中 F 表示选择条件，它是一个逻辑表达式，取逻辑值为“真”或“假”。

逻辑表达式 F 的基本形式为：$X_1 \theta Y_1$

其中，θ 表示比较运算符，它可以是$>$、$\geqslant$、$<$、$\leqslant$、$=$或$<>$。X_1、Y_1 等是属性名，或为常量，或为简单函数；属性名也可以用它的序号来代替。在基本的选择条件上可以进一步进行逻辑运算，即进行求非($\neg$)、与($\wedge$)、或($\vee$)运算。

选择运算实际上是从关系 R 中选取使逻辑表达式 F 为真的元组。这是从行的角度进行的运算。

【例 3-11】 在图 3-6(a)所示的学生关系中，选择出“所在系”为“计算机”的元组构成新的关系。

$$\sigma_{所在系='计算机'}(学生)$$

结果如图 3-6(b)所示。

2. 投影

关系 R 上的投影(Projection)是从关系 R 中选择出若干属性列组成新的关系。分为两步：

(1) 选择出指定的属性，形成一个可能含有重复行的表。

(2) 删除重复行,形成新的关系。

投影运算表示为:

$$\pi_A(R)=\{t[A] \mid t \in R\}$$

其中,A 为 R 中属性列。

在关系二维表中投影是一种垂直操作,它针对二维表中的属性列。

【例 3-12】 在图 3-6(a)所示的学生关系中,选取姓名和所在系这两列构成新的关系。

$$\pi_{姓名,所在系}(学生)$$

结果如图 3-6(b)和图 3-6(c)所示。

学号	姓名	性别	年龄	所在系
S1001	王洪	男	22	计算机
S1002	赵南南	女	21	计算机
S2001	刘明	男	21	信息
S2002	李娟	女	20	信息

(a) "学生"表

学号	姓名	性别	年龄	所在系
S1001	王洪	男	22	计算机
S1002	赵南南	女	21	计算机

(b) 选择运算的结果

姓名	所在系
王洪	计算机
赵南南	计算机
刘明	信息
李娟	信息

(c) 投影运算的结果

图 3-6 选择和投影运算举例

3. 连接

连接(Join)也称为 θ 连接,它从两个关系的笛卡儿积中选取属性间满足一定条件的元组。记为

$$R \underset{A\theta B}{\bowtie} S=\{\widehat{t_r t_s} \mid t_r \in R \wedge t_s \in S \wedge t_r[A]\theta t_s[B]\}$$

其中 A 和 B 分别是关系 R 和 S 上度数相同且可比属性组。θ 为比较运算符。连接运算从 R 和 S 的笛卡儿积 $R\times S$ 中选取 R 关系在 A 属性组上的值与 S 关系在 B 属性组上值满足比较关系 θ 的元组。

在连接运算中有两种最常见的连接,一种是等值连接(Equi Join),另一种是自然连接(Natural Join)。

(1) 等值连接。将比较运算符 θ 为"="时的连接称为等值连接,其结果是从关系 R 和 S 的笛卡儿积中选取属性组 A 和 B 值相等的元组。记为

$$R \underset{A=B}{\bowtie} S=\{\widehat{t_r t_s} \mid t_r \in R \wedge t_s \in S \wedge t_r[A]=t_s[B]\}$$

(2) 自然连接。自然连接是一种特殊的等值连接。它要求两个关系中进行比较的分量必须是相同的属性组,并且在结果中把重复的属性列去掉。当关系 R 和 S 有相同的属性组 B,且该属性组的值相等时的连接称为自然连接。结果关系的属性集合为 R 的属性并上 S 减去属性 B 后的属性集合。R 和 S 的自然连接记为

$$R \bowtie S=\{t_r t_s \mid t_r \in R \wedge t_s \in S \wedge t_r[A]=t_s[B]\}$$

（3）自然连接与等值连接的区别。自然连接要求两个关系中进行比较的属性或属性组必须同名和相同值域，而等值连接只要求比较属性有相同的值域。

自然连接的结果中，同名的属性只保留一个。

一般的连接操作是从行的角度进行运算。但自然连接还需要取消重复列，所以是同时从行和列的角度进行运算。

【例 3-13】 设图 3-7(a)和图 3-7(b)分别为关系 R 和关系 S，图 3-7(c)为一般连接 $R \underset{C<E}{\bowtie} S$ 的结果，图 3-7(d)为等值连接 $R \underset{R.B=S.B}{\bowtie} S$，图 3-7(e)为自然连接 $R \bowtie S$ 的结果。

A	B	C
a_1	b_1	5
a_1	b_2	6
a_2	b_3	8
a_2	b_4	12

(a) R

B	E
b_1	3
b_2	7
b_3	10
b_3	2
b_5	2

(b) S

A	$R.B$	C	$S.B$	E
a_1	b_1	5	b_2	7
a_1	b_1	5	b_3	10
a_1	b_2	6	b_2	7
a_1	b_2	6	b_3	10
a_2	b_3	8	b_3	10

(c) $R \underset{C<E}{\bowtie} S$

A	$R.B$	C	$S.B$	E
a_1	b_1	5	b_1	3
a_1	b_2	6	b_2	7
a_2	b_3	8	b_3	10
a_2	b_3	8	b_3	2

(d) $R \underset{R.B=S.B}{\bowtie} S$

A	B	C	E
a_1	b_1	5	3
a_1	b_2	6	7
a_2	b_3	8	10
a_2	b_3	8	2

(e) $R \bowtie S$

图 3-7　连接运算举例

两个关系 R 与 S 在做自然连接时，选择两个关系在公共属性上值相等的元组构成新的关系。此时，关系 R 中某些元组有可能在 S 中不存在公共属性上值相等的元组，从而造成 R 中这些元组在操作时被舍弃了，同样，S 中的第 5 个元组被舍弃掉了。

如果把舍弃的元组也保存在结果关系中，而在其他属性上填空值（NULL），那么这种连接就叫作外连接（Outer Join）。如果只把左边关系 R 中要舍弃的元组保留就叫作左外连接（Left Outer Join 或 Left Join），如果只把右边关系 S 中要舍弃的元组保留就叫右外连接（Right Outer Join 或 Right Join）。

【例 3-14】 在图 3-8 中，图 3-8(a)是图 3-7 中的关系 R 和关系 S 的外连接，图 3-8(b)是左外连接，图 3-8(c)是右外连接。

下面举一个包括选择、投影、连接运算的综合例子。

【例 3-15】 在图 3-9(a)给出课程关系和选课关系中，求出选修外语课程学生的学号和成绩。

$$R_1 = \sigma_{\text{课程名}='\text{外语}'}(\text{课程})$$

$$R_2 = R_1 \bowtie_{\text{选课}}$$

$$R_3 = \pi_{\text{学号,成绩}}(R_2)$$

运算过程中产生关系 R_1 和 R_2 及最终得到的关系 R_3 如图 3-9(b)所示。

A	B	C	E
a_1	b_1	5	3
a_1	b_2	6	7
a_2	b_3	8	10
a_2	b_3	8	2
a_2	b_4	12	NULL
NULL	b_5	NULL	2

(a) 外连接

A	B	C	E
a_1	b_1	5	3
a_1	b_2	6	7
a_2	b_3	8	10
a_2	b_3	8	2
a_2	b_4	12	NULL

(b) 左外连接

A	B	C	E
a_1	b_1	5	3
a_1	b_2	6	7
a_2	b_3	8	10
a_2	b_3	8	2
NULL	b_5	NULL	2

(c) 右外连接

图 3-8　外连接运算举例

"课程"表

课程号	课程名	学时	学分
C1	外语	64	4
C2	数据库	48	2
C3	高数	48	3

"选课"表

学号	课程号	成绩
S1001	C1	80
S1001	C2	91
S1002	C3	NULL
S2001	C1	96

(a) "课程"表和"选课"表

R_1

课程号	课程名	学时	学分
C1	外语	64	4

R_2

课程号	课程名	学时	学分	学号	成绩
C1	外语	64	4	S1001	80
C1	外语	64	4	S2001	96

R_3

学号	成绩
S1001	80
S2001	96

(b) R_1、R_2和R_3

图 3-9　综合运算举例

4. 除运算

给定关系 $R(X,Y)$和 $S(Y,Z)$，其中 X、Y、Z 是属性组，R 中的 Y 与 S 中的 Y 可以有不同的属性名，但必须出自相同的域集。R 与 S 的除运算(Division)为

$$R \div S = \{t_r[X] \mid t_r \in R \wedge \pi_Y(S) \subseteq Y_x\}$$

$R \div S$ 是一个新关系 $P(X)$，P 是 R 中满足下列条件的元组在 X 属性列上的投影：元组在 X 上分量值 x 的像集 Y_x 包含 S 在 Y 上投影的集合。其中 Y_x 为值 x 在 R 中的像集，有 $x = t_r[X]$。

【例 3-16】 设关系 R、S 分别为图 3-10(a)和图 3-10(b)，$R \div S$ 的结果为图 3-10(c)。

在关系 R 中，A 可以取 4 个值$\{a_1, a_2, a_3, a_4\}$。其中，

a_1 的像集为$\{(b_1, c_2), (b_2, c_3), (b_2, c_1)\}$

a_2 的像集为$\{(b_3, c_7), (b_2, c_3)\}$

a_3 的像集为$\{(b_4, c_6)\}$

a_4 的像集为$\{(b_6, c_6)\}$

S 在(B,C)上的投影为$\{(b_1, c_2), (b_2, c_1), (b_2, c_3)\}$

所以只有 a_1 的像集$(B,C)_{a_1}$ 包含 S 在(B,C)属性组上的投影，即

$$R \div S = \{a_1\}$$

A	B	C
a_3	b_1	c_2
a_2	b_3	c_7
a_3	b_4	c_6
a_1	b_2	c_3
a_4	b_6	c_6
a_2	b_2	c_3
a_1	b_2	c_1

(a) R

B	C	D
b_1	c_2	d_1
b_2	c_1	d_1
b_2	c_3	d_2

(b) S

A
a_1

(c) $R \div S$

图 3-10　除运算举例

【例 3-17】 在如图 3-11(a)所示的选课关系中，求至少选修 C1、C2、C3 课程的学生学号。在学生关系中，学号的取值为{S1001，S1002，S2001}，其中：

学号 S1001 的像集为{C1，C2，C3}
学号 S1002 的像集为{C1，C4}
学号 S2001 的像集为{C1，C2，C3，C4}

所以只有学号 S1001 和学号 S2001 在课程号上的像集包含{C1，C2，C3}，即只有学号为 S1001 和 S2001 的学生至少选修 C1、C2、C3 课程，结果如图 3-11(c)所示。

学号	课程号
S1001	C1
S1002	C1
S2001	C1
S1001	C2
S2001	C2
S1001	C3
S2001	C3
S1002	C4
S2001	C4

(a) “选课”表

课程号
C1
C2
C3

(b) 选修课程号

学号
S1001
S2001

(c) 除结果

图 3-11　选课关系除运算举例

自然连接是关系代数中最重要的运算之一。利用自然连接、投影、选择可以有效地分割和组装关系。这是关系模型的 DML 具有各种优点的原因所在。

3.4.3　基本运算及其变换运算

本节介绍 8 种关系代数运算，其中并、差、笛卡儿积、选择和投影 5 种运算为基本运算，其他 3 种运算，即交、连接和除，均可以用这 5 种基本运算来表达。引用它们并不增加语言的能力，但可以简化表达。

两个关系的交运算表示为：

$$R \cap S = R - (R - S)$$

两个关系的自然连接运算表示为：

$$R \bowtie S = \pi_X(\sigma_{r[A_i]=s[B_j]}(R \times S))$$

两个关系的除运算表示为：

$$R \div S = \pi_X(R) - \pi_X((\pi_X(R) \times S) - R)$$

关系代数中，这些运算经有限次复合后形成的式子称为关系代数表达式。

3.5 关系演算

关系演算是以数理逻辑中的谓词演算为基础的，通过谓词形式来表示查询表达式。根据谓词变元的不同，可将关系演算分为元组关系演算和域关系演算，其代表语言分别为 ALPHA 和 QBE。元组关系演算语言的典型代表是 E. F. Codd 提出的 ALPHA 语言，这种语言虽然没有实际实现，但较有名气，INGRES 关系数据库上使用的 QUEL 语言，就是在 ALPHA 语言的基础上研制的。本节先介绍元组关系演算，然后简单介绍域关系演算。

3.5.1 元组关系演算语言——ALPHA

在关系运算中，用谓词公式来表达查询要求的方式称为关系演算。元组关系演算是以元组变量作为谓词变元的基本对象。由于用到谓词公式，必然涉及谓词变量和谓词合适公式(Well-formed Formila)的问题。

ALPHA 语言是以谓词公式来定义查询要求的。在谓词公式中存在客体变元，这里称为元组变量。元组变量是一个变量，其变化范围为某个命名的关系。

ALPHA 语言的基本格式是：

操作符　工作空间名(表达式)：操作条件

(1)“操作符”有 GET、PUT、HOLD、UPDATE、DELETE、DROP 等多种。

(2)“工作空间”是指内存空间，可以用一个字母表示，通常用 W 表示，也可以用别的字母表示。工作空间是用户与系统交互的空间，“表达式”则是指对数据的操作。

(3)“目标表”用于指定操作(如查询、更新等)出来的结果，它可以是关系名或属性名，一个操作语句可以同时对多个关系或多个属性进行操作。

(4)“操作条件”是用谓词公式表示的逻辑表达式，仅满足此条件的元组会被操作，此条件为可选项，若未指定则默认执行操作符对应的完整操作(如查询所有符合条件的记录)。此外，还可以在基本格式上添加排序要求、定额限制等附加条件。

下面以“学生管理”数据库为例，说明 ALPHA 语言的使用。

1. 数据查询

1) 简单查询

【例 3-18】 查询所有学生的数据。

```
GET  W(学生)
```

GET 语句的作用是把数据库中的数据读入内存空间 W，目标表为学生关系，代表查询出来的结果，即所有的学生。冒号后面的操作条件默认，表示无条件查询。

【例 3-19】 查询所有被选修的课程号。

```
GET  W (课程.课程号)
```

目标表为选课关系中的“课程号”属性，代表所有被选修的课程号码，查询结果会自动消去重复行。

2）条件查询

由冒号后面的逻辑表达式给出查询条件，在表达式中可以使用如下三类运算符。

- 比较运算符：＞、≥、＜、≤、＝(等于)、≠。
- 逻辑运算符：∧(与)、∨(或)、¬(非)。
- 表示执行次序的括号：()。

其中，比较运算符的优先级高于逻辑运算符，可以使用()改变它们的优先级。

【例 3-20】 查询计算机系工资低于 1000 元的教师的姓名和工资。

```
GET W (教师.教师姓名,教师.工资):教师.所在系 = '计算机'∧教师.工资＜1000
```

目标表为教师关系中的两个属性“教师姓名”和“工资”组成的属性列表。

3）排序查询

【例 3-21】 查询 S1001 同学所选课程号及成绩，并按成绩降序排列。

```
GET W (选课.课程号,选课.成绩):选课.学号 = 'S1001' DOWN 选课.成绩
```

DOWN 表示降序，UP 表示升序，以帮助理解排序的方向。

4）定额查询

【例 3-22】 查询一名男教师的教师编号和姓名。

```
GET W (1) (教师.教师编号,教师.姓名):教师.性别 = '男'
```

所谓的定额查询就是通过在 W 后面的括号中加上定额数量，但可以进一步解释定额查询的实际意义，例如在大数据集中限制返回结果的数量。这里(1)表示查询结果中男教师的个数，取出教师表中第一个男教师的教师号和姓名。排序和定额查询可以一起使用。

【例 3-23】 查询一名男教师的教师编号和姓名，并使他的年龄最小。

```
GET W (1) (教师.教师编号,教师.姓名):教师.性别 = '男' UP 教师.年龄
```

此语句的执行过程为：先查询所有男教师的教师编号和姓名，再按照年龄由小到大排序，然后找出第一位，也就是年龄最小的男教师。

5）带元组变量的查询

所谓的元组关系演算就是以元组变量作为谓词变元的基本对象，在关系演算的查询操作时，可以在相应的关系上定义元组变量。元组变量代表关系中的元组，其取值是在所定义的关系范围内变化，所以也称作范围变量(Range Variable)，一个关系可以设多个元组变量。

【例 3-24】 查询 S1001 同学所选课程号。

```
RANGE 选课 X
GET W (X.课程号):X.学号 = 'S1001'
```

使用 RANGE 来说明元组变量 X 为选课关系上的元组变量。如果关系的名字很长，使用起来不方便，这时可以设一个名字较短的元组变量来代替关系名，简化关系名，使操作更加方便。

6）带存在量词的查询

【例 3-25】 查询 S1001 同学所选课程名。

```
RANGE 选课 X
```

```
GET W(课程.课程名):∃X(课程.课程号 = X.课程号 ∧ X.学号 = 'S1001')
```

注意：操作条件中使用量词时必须用元组变量。

【例 3-26】 查询至少选修一门其学分为 4 的课程的学生的姓名。

```
RANGE  课程  CX
       选课  SCX
GET W(学生.姓名):∃SCX(SCX.学号 = S.学号 ∧ ∃CX(CX.课程号 = SCX.课程号 ∧ CX.学分 = 4))
```

此查询涉及 3 个关系，需要对课程关系和选课关系作用存在量词，所以用了两个元组变量。此语句的执行过程为：先查询学分为 4 的课程号，再根据找到的课程号在选课关系中查询其对应的学号，然后根据这些学号在学生关系中找到对应的学生姓名。

【例 3-27】 查询选修全部课程的学生姓名。

```
RANGE 课程 CX
      选课 SCX
GET W(学生.姓名):∀CX∃SCX(SCX.学号 = 学生.学号 ∧ CX.课程号 = SCX.课程号)
```

7）库函数查询

库函数也称集函数。用户在使用查询语言时，经常要作一些简单的运算。

如要统计某个关系中符合某一条件的元组数，或某些元组在某个属性上分量的和、平均值等。

在关系数据库语言中提供了有关这类运算的标准函数，增强了基本检索能力。

常用的库函数如表 3-10 所示。

表 3-10 常用库函数表

函 数 名 称	功 能
AVG	按列计算平均值
TOTAL	按列计算平均值
MAX	求一列中的最大值
MIN	求一列中的最小值
COUNT	按列值计算元组个数

【例 3-28】 求学号为 S1001 学生的平均分。

```
GET W(AVG(选课.成绩):学生.学号 = 'S1001'
```

【例 3-29】 求学校共有多少个系。

```
GET W(COUNT(学生.所在系))
```

COUNT 函数自动消去重复行，可计算字段“所在系”不同值的数目。

2. 数据更新

更新操作包括修改、插入和删除。

1）修改

修改操作使用 UPDATE 语句实现，具体操作分为以下三步：

（1）读数据。使用 HOLD 语句将要修改的元组从数据库中读到工作空间中。

（2）修改。利用宿主语言修改工作空间中元组的属性。

（3）送回。使用 UPDATE 语句将修改后的元组送回数据库中。

这里 HOLD 语句是带上并发控制的 GET 语句。

【例 3-30】 把刘伟教师转到信息系。

```
HOLD W(教师.所在系):教师.教师姓名 = '刘伟'
MOVE '信息' TO W.所在系
UPDATE W
```

在 ALPHA 语言中,不允许修改关系的主键,例如不能使用 UPDATE 语句修改教师关系中的教师编号。如果要修改主键,应该先使用删除操作删除该元组,再插入一条具有新主键值的元组。

2) 插入

插入操作使用 PUT 语句实现,具体操作分为以下两步:

(1) 建立新元组。利用宿主语言在工作空间中建立新元组。

(2) 写数据。使用 PUT 语句将元组写入指定的关系中。

【例 3-31】 在选课关系中插入一条选课记录('S1002','C3',85)。

```
MOVE 'S1002' TO W.学号
MOVE 'C3' TO W.课程号
MOVE 85 TO W.成绩
PUT W(选课)
```

PUT 语句的作用是把工作空间 W 中的数据写到数据库中,此例即把已经在工作空间建立的一条选课记录写入选课关系中。

注意:PUT 语句只能对一个关系进行操作,在插入操作时,拒绝接受主键相同的元组。

3) 删除

ALPHA 语言中的删除操作不但可以删除关系中的一些元组,还可以删除一个关系。

删除操作使用 DELETE 语句实现,具体操作分为以下两步:

(1) 读数据。使用 HOLD 语句将要删除的元组从数据库中读到工作空间中。

(2) 删除。使用 DELETE 语句删除该元组。

【例 3-32】 删除学号为 S2001 的学生的信息。

```
HOLD W(学生):学生.学号 = 'S2001'
DELETE W
```

【例 3-33】 删除全部学生的信息。

```
HOLD W(学生)
DELETE W
```

3.5.2　域关系演算

域关系演算是关系演算的另一种形式。域关系演算是以元组变量的分量(即域变量)作为谓词变元的基本对象。域关系演算语言的典型代表是 1975 年由 IBM 公司约克城高级研究试验室的 M. M. Zloof 提出的 QBE 语言,该语言于 1978 年在 IBM370 上实现。QBE 是 IBM 公司商品化关系数据库系统 DB2 的可选用户界面之一。QBE 是一个基于域演算的数据库语言。用户工作方式是在关系框架上填写简单的示例信息作为查询请求。所有操作符后跟圆点,域变量或样板的格式为'_字符串',其他类型的字串为常值。QBE 是 Query by Example 的缩写,也称为示例查询,它是一种很有特色的屏幕编辑语言,其特点如下:

(1) 以表格形式进行操作。每个操作都由一个或几个表格组成,每个表格都显示在终端的屏幕上,用户通过终端屏幕编辑程序以填写表格的方式构造查询要求,查询结果也以表格的形式显示出来,所以它具有直观和可对话的特点。

(2) 通过例子进行查询。通过使用一些实例,使该语言更易于被用户接受和掌握。

(3) 查询顺序自由。当有多个查询条件时,不要求使用者按照固定的思路和方式进行查询,使用更加方便。

使用 QBE 语言的步骤如下:

(1) 用户根据要求向系统申请一张或几张表格,显示在终端上。

(2) 用户在空白表格的左上角的一栏内输入关系名。

(3) 系统根据用户输入的关系名,将在第一行从左至右自动填写各个属性名。

(4) 用户在关系名或属性名下方的一格内填写相应的操作命令,操作命令包括:P.(打印或显示)、U.(修改)、I.(插入)、D.(删除)。如果要打印或显示整个元组时,应将"P"填在关系名的下方,如果只需打印或显示某一属性,应将"P"填在相应属性名的下方。

表格形式如表 3-11 所示。

表 3-11 表格形式

关系名	属性 1	属性 2	…	属性 *n*
操作命令	属性值或查询条件	属性值或查询条件	…	属性值或查询条件

QBE 功能分为数据定义、数据查询、数据维护和视图定义四类。

1. 数据定义

QBE 系统维护一个表,即关系目录表。它包含数据库中所有关系的名字、属性和说明信息。

关系框架的一般格式如表 3-12 所示。

表 3-12 关系框架的一般格式

关 系 名	属 性 名	属 性 名	属 性 名
操作命令 1	元组属性		
操作命令 2	或		
操作命令 3	查询要求信息		

【例 3-34】 建立关系 DEPARTMENT。

对首行各纵栏,从左到右填:

```
(命令) I. DEPARTMENT  I.
(第一属性名称) DNAME
(第二属性名称) DNUMBER
(第三属性名称) MGRSSN
(第四属性名称) STARTDT
```

然后按首列提示定义各属性:

```
KEY I.(是否主属性)
TYPE I.(数据类型)
DOMAIN I.(值域名称)
INVERSION I.(是否索引)
```

I. DEPARTMENT	I.	DNAME	DNUMBER	MGRSSN	STARTDT
KEY	I.	N	Y	N	N
TYPE	I.	CHAR	INTEGER	INTEGER	INTEGER
DOMAIN	I.	NAMES	NUMBERS	SSN	DATE
INVERSION	I.	N	Y	N	N

2. 数据查询

简单查询步骤示例如下。

【例 3-35】 求信息系全体学生的姓名。

操作步骤为：

(1) 用户提出要求。

(2) 屏幕显示空白表格。

(3) 用户在最左边一栏输入关系名 Student。

Student					

(4) 系统显示该关系的属性名。

Student	Sno	Sname	Ssex	Sage	Sdept

(5) 用户在上面构造查询条件。

Student	Sno	Sname	Ssex	Sage	Sdept
		P. T			IS

这里的 T 是示例元素，即域变量 QBE 要求示例元素下面一定要加下画线。IS 是查询条件，不用加下画线。P. 是操作符，表示打印(Print)，实际上是显示。

查询条件中可以使用比较运算符＞、≥、＜、≤、＝和≠。其中＝可以省略。

示例元素是这个域中可能的一个值，它不必是查询结果中的元素。例如要求信息系的学生，只要给出任意的一个学生名即可，而不必真是信息系的某个学生名。

对于例 3-35，也可如下构造查询要求：

Student	Sno	Sname	Ssex	Sage	Sdept
		P. 王莉			IS

这里的查询条件是 Sdept＝'IS'，其中“＝”被省略。

(6) 屏幕显示查询结果。

Student	Sno	Sname	Ssex	Sage	Sdept
		王莉 李勇			IS

根据用户的查询要求，求出了信息系的学生姓名。

【例 3-36】 查询全体学生的全部数据。

Student	Sno	Sname	Ssex	Sage	Sdept
	P. 06001	P. 王莉	P. 女	P. 20	P. CS

显示全部数据也可以简单地把 P. 操作符作用在关系名上。因此本查询也可以简单地表示如下：

Student	Sno	Sname	Ssex	Sage	Sdept
P.					

3. 数据维护

QBE 有 3 个数据维护操作：插入元组 I. 、删除元组 D. 、修改元组 U. 。

(1) 插入操作。插入操作符为“I. ”。新插入的元组必须具有键值，其他属性值可以为空。

【例 3-37】 把计算机系女生 06030，姓名为李红，年龄 20 岁存入数据库中。

Student	Sno	Sname	Ssex	Sage	Sdept
I.	06030	李红	女	20	CS

(2) 删除操作。删除操作符为“D. ”。

【例 3-38】 删除学生 06051。

Student	Sno	Sname	Ssex	Sage	Sdept
D.	06051				

由于 SC 关系和 Student 关系之间具有参照关系，为保证参照完整性，删除 06051 学生后，通常还应删除 06051 学生选修的全部课程。

SC	Sno	Cno	Grade
D.	06051		

(3) 修改操作。修改操作符为“U. ”。在 QBE 中，关系的主键不允许修改，如果需要修改某个元组的主键，只能先删除该元组，然后再插入新的主键的元组。

【例 3-39】 把 06001 学生的年龄改为 20 岁。

这是一个简单修改操作，不包含算术表达式，因此可以有两种表示方法。

① 将操作符“U. ”放在值上。

Student	Sno	Sname	Ssex	Sage	Sdept
	06001			U. 20	

② 将操作符“U. ”放在关系上。

Student	Sno	Sname	Ssex	Sage	Sdept
U.	06001			20	

这里，键 06001 标明要修改的元组。“U. ”标明所在的行是修改后的新值。由于主键是不

能修改的，所以即使在第二种写法中，系统也不会混淆要修改的属性。

【例 3-40】　把 06001 学生的年龄增加 1 岁。

这个修改操作涉及表达式，所以只能将操作符“U.”放在关系上。

Student	Sno	Sname	Ssex	Sage	Sdept
U.	06001 06001			19 19+1	

4. 视图定义

建立视图的方法类似于建立关系。

3.6　本章小结

关系数据库系统是目前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的。在数据库发展的历史上，关系模型是最重要的成就之一。本章内容是学习数据库原理知识的重点内容。

关系数据库系统和非关系数据库系统的区别是：关系数据库系统只有“表”这一种数据结构；而非关系数据库系统还有其他数据结构，以及对这些数据结构的操作。

本章系统讲解了关系数据库的重要概念，包括关系模型的数据结构、关系操作和关系的完整性约束。详细讲解了关系模型的三类完整性约束，包括实体完整性、参照完整性和用户定义完整性。最后介绍了用关系代数方式和关系逻辑方式来表达关系操作，即关系代数、元组关系演算和域关系演算。本章内容从具体到抽象，由浅入深，逐步讲解，便于加深读者的理解。

3.7　习题

1. 选择题

(1) 关系数据模型中，数据的逻辑结构是(　　)。

A. 树状结构　　B. 网状结构　　C. 二维表　　D. 图状结构

(2) 关系代数中的并运算要求两个关系具有(　　)。

A. 相同的行数　　B. 相同的列数

C. 相同的元组数　　D. 相同的属性名

(3) 在关系数据库中，用于表示实体间联系的是(　　)。

A. 主键　　B. 外键　　C. 索引　　D. 视图

(4) 下列(　　)操作不属于关系代数的基本运算。

A. 选择　　B. 投影　　C. 连接　　D. 排序

(5) 关系数据模型的完整性约束包括(　　)。

A. 实体完整性、参照完整性、用户定义完整性

B. 主键约束、外键约束、唯一性约束

C. 实体完整性、参照完整性、域完整性

D. 参照完整性、用户定义完整性、域完整性

(6) 在关系模式中，若属性(或属性组)能唯一标识一个元组，则称为(　　)。

A. 主键　　B. 外键　　C. 候选键　　D. 超键

(7) 自然连接是(　　)的一种特殊形式。

A. 笛卡儿积　　B. 差运算　　C. 等值连接　　D. 交运算

(8) 自然连接是一种特殊的等值连接。一般情况下，当对关系 R 和 S 使用自然连接时，要求 R 和 S 含有一个或多个共有的(　　)。

A. 元组　　B. 行　　C. 记录　　D. 属性

(9) 在关系代数的专门关系运算中，从表中选出满足某种条件的元组的操作称为(　　)。

A. 投影　　B. 连接　　C. 选择　　D. 扫描

(10) 在关系数据库中，用于描述现实世界实体及其联系的是(　　)。

A. 索引　　B. 视图　　C. 表　　D. 存储过程

2. 填空题

(1) 关系模型由关系数据结构、关系操作以及_____三部分组成。

(2) 关系代数的基本运算包括并、差、_____、选择和投影。

(3) 关系模型中，候选键是指能唯一标识一个元组的属性或_____。

(4) 实体完整性规则要求主键不能取_____值。

(5) 自然连接是等值连接的一种特殊形式，它要求比较的分量必须是相同的_____。

(6) 关系代数中的投影运算是在关系中选择若干_____列组成新的关系。

(7) 用户定义的完整性是针对某一具体关系数据库的_____约束条件。

(8) 关系代数中的差运算结果是属于第一个关系但不属于第二个关系的_____。

(9) 参照完整性规则要求外键的值必须等于被参照关系中某个主键的值或_____。

(10) 在关系数据库中，表由行和_____组成。

3. 简答题

(1) 简述关系模型的三个组成部分。

(2) 什么是实体完整性？它有什么作用？

(3) 简述 θ 连接、等值连接与自然连接的区别。

(4) 关系代数中有哪些基本运算？请列举并简要说明。

(5) 什么是用户定义完整性？它有什么作用？

4. 综合题

设有一个“工程项目”数据库，具体关系如表 3-13～表 3-16 所示。①供应商(供应商代码，供应商名，供应商状态，供应商所在城市)；②零件(零件代码，零件名称，零件颜色，零件重量)；③工程(工程号，工程名，工程项目所在城市)；④供应(供应商代码，零件代码，工程号，数量)。使用关系代数、ALPHA 语言、元组演算语言完成如下查询：

①求供应工程 J1 零件的供应商的代码。②求供应工程 J1 零件 P1 的供应商的代码。③求供应工程 J1 零件为“红”色的供应商的代码。④求没有使用天津供应商生产的“红”色零件的工程号。⑤求至少使用了供应商 S1 所供应的全部零件的工程号。

表 3-13　“供应商”表

供应商代码	供应商名	供应商状态	供应商所在城市
S1	精益	20	天津
S2	盛锡	10	北京
S3	东方红	30	北京
S4	金叶	10	天津
S5	泰达	20	上海

表 3-14　“零件”表

零件代码	零件名称	零件颜色	零件重量
P1	螺母	红	20
P2	螺栓	绿	12
P3	螺丝刀	蓝	18
P4	螺丝刀	红	18
P5	凸轮	蓝	16
P6	齿轮	红	23

表 3-15　“工程”表

工程号	工程名	工程项目所在城市
J1	三建	天津
J2	一汽	长春
J3	造船厂	北京
J4	机车厂	南京
J5	弹簧厂	上海

表 3-16　“供应”表

供应商代码	零件代码	工程号	数量
S1	P1	J1	200
S1	P1	J3	100
S1	P1	J4	700
S1	P2	J2	100
S2	P3	J1	400
S2	P3	J1	200
S2	P3	J3	500
S2	P3	J4	400
S2	P5	J2	400
S2	P5	J1	100
S3	P1	J1	200
S3	P3	J3	200
S4	P5	J4	100
S4	P6	J1	300
S4	P6	J3	200
S5	P2	J4	100
S5	P3	J1	200
S5	P6	J3	200
S5	P6	J4	500

习题 3