

目 录

第一章 概述	1 - 1
1.1 什么是数据库系统	1 - 1
1.2 数据库系统的特点	1 - 2
1.3 数据库系统的典型结构	1 - 6
1.4 数据语言	1 - 7
1.4.1 数据描述语言	1 - 7
1.4.2 数据操作语言	1 - 8
1.5 数据库管理系统	1 - 9
1.5.1 数据字典 (<i>Data dictionary</i>)	1 - 10
1.5.2 数据库管理员 (<i>DBA</i>)	1 - 11
1.5.3 用户访问数据库的过程	1 - 12
1.6 实体——联系方法	1 - 14
1.7 数据模型	1 - 17

第二章 存储结构	2 - 1
2.1 引言	2 - 1
2.1.1 文件的基本概念	2 - 1
2.1.2 数据库操作速度的估计	2 - 4
2.1.3 指示器 (<i>Pointer</i>)	2 - 5
2.1.4 关键字 (<i>Keys</i>)	2 - 5
2.1.5 钉定 (<i>Pinned</i>) 和未钉定的记录	2 - 6
2.1.6 文件结构概述	2 - 7
2.2 顺序文件	2 - 10
2.2.1 如何确定关键字值的顺序	2 - 10
2.2.2 顺序文件的存储组织	2 - 10
2.2.3 顺序文件的查找	2 - 11

2.3	随机结构之一——散列方法	2-12
2.3.1	散列方法的简要回顾	2-12
2.3.2	散列文件的设计	2-15
2.3.3	可扩充的散列	2-16
2.4	随机结构之二——索引结构	2-21
2.4.1	索引顺序文件	2-21
2.4.2	索引无序文件	2-21
2.4.3	索引的组织	2-22
2.4.4	索引文件的查找	2-25
2.5	<i>B</i> -树	2-25
2.5.1	二叉树	2-25
2.5.2	<i>B</i> -树	2-27
2.5.3	<i>B</i> +树	2-29
2.5.4	一个 <i>B</i> +树实例	2-31
2.6	变长记录文件	2-33
2.7	倒排文件	2-35
第三章	关系方法	3-1
3.1	关系及基本术语	3-1
3.2	关系运算	3-3
3.2.1	关系代数	3-3
3.2.2	元组关系演算	3-9
3.2.3	域关系演算	3-12
3.3	关于数据库的数据操作语言	3-14
3.3.1	基于关系代数的语言 <i>ISBL</i>	3-15
3.3.2	介于关系代数与演算之间的语言 <i>SEQUEL</i>	3-19
3.3.3	关于元组演算的语言 <i>QUEL</i>	3-27
3.3.4	基于域演算的语言 <i>QBE</i>	3-32
3.4	关系数据库的模式和子模式	3-37
3.4.1	源模式、目标模式及其物理映射	3-37
3.4.2	子模式、目标子模式及其映射	3-41

3.5	询问的优化·····	3-45
3.5.1	优化的一般策略·····	3-46
3.5.2	关系代数表达式的等价代换规则·····	3-47
3.5.3	关系代数表达式的优化算法·····	3-48
第四章	层次方法·····	4-1
4.1	一般概念·····	4-1
4.1.1	树·····	4-1
4.1.2	层次系统的数据模型·····	4-3
4.1.3	层次顺序与层次路径·····	4-5
4.1.4	层次系统的模式与子模式·····	4-7
4.2	IMS 系统的逻辑结构·····	4-8
4.2.1	IMS 的逻辑结构·····	4-8
4.2.2	IMS 的 DBD·····	4-9
4.2.3	IMS 的 PSB·····	4-12
4.3	IMS 的存储结构·····	4-14
4.3.1	HSAM·····	4-14
4.3.2	HISAM·····	4-15
4.3.3	HIDAM 和 HDAM·····	4-19
4.4	IMS 的数据子语言·····	4-25
4.4.1	子语言 DL/1·····	4-25
4.4.2	IMS 的应用程序·····	4-30
4.4.3	应用程序的运行·····	4-35
4.5	IMS 存储结构补充·····	4-36
4.5.1	辅数据集组·····	4-36
4.5.2	IMS 辅助索引·····	4-38
4.5.3	IMS 的逻辑数据库·····	4-40
第五章	DBTG 建议的网状模型的数据库系统·····	5-1
5.1	DBTG 系统的结构·····	5-1

5. 2	DBTG的数据模型	5 - 2
5.2.1	记录类型	5 - 2
5.2.2	络类型 (Set type)	5 - 3
5.2.3	络事件 (Set occurrence)	5 - 5
5.2.4	事物联系的DBTG表示法	5 - 7
5. 3	记录类型描述及其存储映射	5 - 10
5.3.1	DBTG句法使用的符号	5 - 10
5.3.2	记录类型的描述	5 - 11
5.3.3	记录类型的存储映射	5 - 13
5.3.4	记录类型举例	5 - 16
5. 4	络类型描述及其存储映射	5 - 17
5.4.1	络类型 (Set mode)	5 - 17
5.4.2	络次序 (Set order)	5 - 18
5.4.3	从记录类型性质的描述	5 - 23
5.4.4	络选择 (Set selection)	5 - 24
5.4.5	络类型举例	5 - 25
5. 5	模式数据描述	5 - 29
5. 6	子模式数据描述	5 - 29
5.6.1	子模式与模式的区别	5 - 30
5.6.2	子模式举例	5 - 30
5. 7	数据操作语言 (DML)	5 - 32
5.7.1	程序的运行	5 - 32
5.7.2	DML语句概述	5 - 33
5.7.3	应用程序举例	5 - 38

第六章 关系数据库的规范化理论

6. 1	关系模式的规范化概述	6 - 1
6. 2	关系数据库的设计理论	6 - 5
6.2.1	函数依赖 (Functional Dependency)	6 - 6
6.2.2	计算闭包	6 - 10

6.2.3	依赖集的复盖	6 - 12
6. 3	关系模式的分解	6 - 13
6.3.1	分解的定义	6 - 14
6.3.2	联接的不丢失性(<i>Lossless join</i>)	6 - 14
6.3.3	联接不丢失性的检验	6 - 14
6.3.4	分解对依赖的保持	6 - 16
6. 4	关系模式的规范化	6 - 17
6. 5	结果为 <i>BCNF</i> 的联接不丢失性分解	6 - 19
6. 6	多值依赖及第四范式	6 - 22

第三章 关系方法

3.1 关系及其基本术语

在1.7中用二维表描述了学生、课程的基本情况及学生与课程的联系(学生的学习情况)。在文件组织中,上述二维表格称为文件,表中每一行则称为一个记录,表头那一行称为记录格式,每一列中的各元素称为一个数据项。从而可知,文件与关系,记录与元组,记录类型与关系框架均很类似,后者是前者的一种抽象,前者是后者可行的实现。从概念的角度,关系中行和列的次序是无关重要的。且在一个关系中不允许重复元组存在。

关系数据库与文件系统名词对应情况:

关系(*relation*) —— 文件

元组(*tuple*) —— 记录

属性或域(*field*) —— 场

关系是实体集,元组是一个实体。

几个基本术语:

(1) 超关键字(*Super Key*)。在一个关系中,若某一属性集合的值对不同的元组是不同的,因而可唯一地标识元组,则称该属性集合为该关系的超关键字,如学生关系中的学号、年龄所组成的属性集合即为该关系的一个超关键字。

(2) 候选关键字(*candidate key*)与主关键字(*primary key*)。在一个关系中可能有多个属性都能唯一的标识元组。但在为关系组织物理文件时,通常选用一个作为插入、删除、检索元组的操作变量,则这个属性称为主关键字,而其它能唯一标识元组的属性称候选关键字。因而一个关系的主关键字只有一个,而候选关键字、超关键字可能有多个。

(3) 合成关键字(*Composite*)。多个属性组成关键字,如学号、课程号所组成的属性集合为合成关键字。

(4) 外来关键字(*foreign key*)。一个关系 R_1 的属性 A 作为另一关系 R_2 的候选关键字,则 A 为外来关键字。合成关键字和外来关键字提供了一种表示两个关系联系的方法。如属性学号、课程号都是外来关

键字，用来表示学生关系、课程关系与学习关系之间的联系。

我们借助于集合论的术语和符号给关系以严格定义。设有属性 $A_1, A_2, \dots, A_k$ ，它们分别在值域 $D_1, D_2, \dots, D_k$ 中取值，按照集合论的观点，这些值域构成一个笛卡尔乘积空间 $D = D_1 \times D_2 \times \dots \times D_k$ ， D 中的任意一个子集 D' 称为一个关系，记为 R 。其关系框架是属性 A_i 的一个有序集合，记为 $R(A_1, A_2, \dots, A_k)$ 。 D' 中的任一个点 $(t_1, t_2, \dots, t_k)$ 称 R 的一个元组。

其中 K 中为关系 R 的元 (arity)。当 $K=1, 2, 3$ 时，依次

NAME			AGE			SEX		
Li	21	M						
Wang								
Zhang	22	F						

= 表 NAS

NAME	AGE	SEX
Li	21	F
Li	21	M
Wang	21	F

NAME	AGE	SEX
Li	21	M
Wang	22	F
Zhang	21	F
Zhang	22	M

表 NAS			表 3.1		
NAME	AGE	SEX	NAME	AGE	SEX
Li	21	M	Li	21	M
Li	21	F	Li	21	F
Li	22	M	Li	22	M
Li	22	F	Li	22	F
Wang	21	M	Wang	21	M
Wang	21	F	Wang	21	F
Wang	22	M	Wang	22	M
Wang	22	F	Wang	22	F
Zhang	21	M	Zhang	21	M
Zhang	21	F	Zhang	21	F
Zhang	22	M	Zhang	22	M
Zhang	22	F	Zhang	22	F

称该关系为一元关系 (unary)，二元关系 (binary)，三元关系 (ternary)。因为关系是元组的集合，所以在关系中不包含重复元组。

例如， $NAME = \{Li, Wang, Zhang\}$ ， $AGE = \{21, 22\}$ ， $SEX = \{M, F\}$ ，笛卡尔乘积空间如表 3.1。表 NAS 展示的空间的两个子集 R_1 与 R_2 就是两个关系。我们把对应于同一关系框架的一些关系称为同类关系，均为同一乘积空间的子集。

3.2 关系运算

关系运算是设计关系数据库操作语言的基础，因为其中的一个询问往往表示成一个关系运算表达式。在以后将要讨论的关系模型中，数据及其联系都是用关系表示的，所以实现数据之间联系的运算也可由关系运算完成。我们首先研究三类关系运算（关系代数、元组关系演算和域关系演算）。

为叙述上的方便，对于关系 R 的属性 $A_1, A_2, \dots, A_n$ ，有时用属性名表示，有时用它们的序号表示，还有时把属性称为元组的分量。

设有同类关系 R 和 S ，若 R 的任何一个元组，必然是 S 的一个元组，则称关系 S 包含关系 R ，并记为： $S \supseteq R$ ，或 $R \subseteq S$ 。如果 $S \supseteq R$ ，同时存在 $S \subseteq R$ ，则称 R 等于 S ，并记为 $R = S$ 。

3.2.1 关系代数 (relational algebra)

关系代数运算的一部分是集合运算（如两个集合的和、差、交），另一部分是关系运算所特有的投影、选择、求商、连接等运算。我们将用定义集合的办法来定义关系代数运算，以便很容易从关系代数过渡到关系演算。在以下的一些定义中，将要用到集合论中的“属于”运算符 $\in$ 、 $\notin$ （不属于），算术比较运算符 $<$ 、 $\leq$ 、 $>$ 、 $\geq$ 、 $=$ 、 $\neq$ ，以及逻辑运算符 $\neg$ 、 $\wedge$ 、 $\vee$ ，现规定这些运算符的运算优先次序如下： $\in$ 、 $\notin$ 优先级最高，算术比较运算符次之，逻辑运算符为最低，且按 $\neg$ 、 $\wedge$ 、 $\vee$ 的先后次序进行。此外还要引入元组变量，即将关系中的元组视为变量时，则称之为关系的元组变量。

在关系代数中，有五种运算是最基本的，其他运算均可表为基本运算的合成式。

几种基本运算的定义如下：

(1) (union)。设有同类关系 R_1 、 R_2 ，定义二者的合并运算为：

$$R_1 \cup R_2 = \{ t \mid t \in R_1 \vee t \in R_2 \}$$

式中 $\cup$ 为合并运算符, t 为元组变量。该运算与集合论中的求和集运算相同, 合并运算后得到的关系是关系 R_1 与 R_2 元组的并集。显然, 它与 R_1 、 R_2 为同类关系 (图 3.2)。

(2) 求差 (difference)。设 R_1 、 R_2 为同类关系, 定义二者的求差运算为:

$$R_1 - R_2 = \{t \mid t \in R_1 \wedge t \notin R_2\}$$

A_1	A_2	A_3	A_1	A_2	A_3	A_2	A_3
b	2	d	a	3	c	2	d
b	3	b	b	2	d		
c	2	d	c	2	d		
d	3	b	e	5	f		
			g	6	f	3	b

关系 R_1 关系 R_2 关系 S_1

图 3.1 三个已知关系

A_1	A_2	A_3	A_1	A_2	A_3	A_1	A_2	A_3	A_1	A_2	A_3	A_1	A_2	A_3
b	2	d	b	3	b	b	2	d	g	6	f			
b	3	b	d	3	b	b	2	d	a	3	c			
c	2	d				b	3	b	b	2	d			
d	3	b				b	3	b	c	2	d			
a	3	c				c	2	d	b	3	b			
e	5	f				c	2	d	b	2	d			
g	6	f				d	3	b	c	2	d			
						d	3	b						
						d	3	b						

$R_1 \cup R_2$ $R_1 - R_2$ $R_1 \times S_1$ $\sigma_{[2] > 5 \vee [3] \neq f}(R_2)$
 $\pi_{3,2}(R_1)$

图 3.2 五种基本运算

式中 “-” 为求差运算符。即前者减去它与后者相同的那些元组, 其结果也是元组的集合, 且与 R_1 、 R_2 为同类关系, 显然, 它与集合论中的差运算相同 (3.2)。

(3) 乘积 (product)。设 R 为 K_1 元关系, S 为 K_2 元关系, 定义二者的乘积为

$$R \times S = \{t \mid t = \langle t^{k_1}, t^{k_2} \rangle \wedge t^{k_1} \in R \wedge t^{k_2} \in S\}$$

其中“ $\times$ ”为乘积运算符。二者的乘积可完全按照笛卡儿乘积去理解，即仍是元组的集合，且为一个 K_1+K_2 元的新关系。积的每个元组的前 K_1 个分量为 R 的一个元组，后 K_2 个分量为 S 的一个元组。用 R 的第 i 个元组与 S 的全部元组结合成 K_2 个元组，当 i 从1变到 R 元组数，就得到了新关系的全部元组（图3.2）。

(4) 投影 (projection)。设有 K 元关系 R ，其元组变量为 $t^k = \langle t_1, t_2, \dots, t_k \rangle$ ，那么关系 R 在其分量 $A_{j_1}, A_{j_2}, \dots, A_{j_n}$ ($n \leq K$, $j_1, j_2, \dots, j_n$ 为1到 K 之间互不相同的整数) 上的投影定义为

$$\Pi_{j_1, j_2, \dots, j_n}(R) = \{ t \mid t = \langle t_{j_1}, t_{j_2}, \dots, t_{j_n} \rangle \wedge \langle t_1, t_2, \dots, t_k \rangle \in R \}$$

其中 Π 为投影运算符。其涵义是按照 $j_1, j_2, \dots, j_n$ 的顺序从关系 R 中取下这 n 列，删除结果中的重复元组，构成一个以 $j_1, j_2, \dots, j_n$ 为列顺序的 n 元关系（图3.2）。为了书写方便，关系 $S(A_1, A_2, A_3, A_4)$ 在 A_4, A_1, A_3 等列上的投影可以记为 $\Pi_{A_4, A_1, A_3}(S)$ ，简记为 $\Pi_{4, 1, 3}(S)$ 亦可。

(5) 选择 (selection)。设 F 是一个命题公式，其运算对象是常量或元组的分量（分量可为分量名或分量序号，如第 i 个分量可记为 $[i]$ 或在无二义性情况下简记为 i ），运算符为算术比较运算符、逻辑运算符。如取 $F = [2] \geq 3 \vee [3] \neq [1]$ ，它表示这样的命题函数：元组的第2个分量大于等于3或第3个分量不等于第1个分量。那么关系 R 关于公式 F 的选择运算定义为

$$\sigma_F(R) = \{ t \mid t = \langle t_1, t_2, \dots, t_k \rangle \wedge t \in R \wedge F[t_1=t_1, t_2=t_2, \dots, t_k=t_k] = \text{True} \}$$

式中 σ 为选择运算符。 $\sigma_F(R)$ 表示从 R 中挑选满足 F 为真的那些行或元组所构成的，与 R 同类的关系（图3.2）。图3.1表示三个关系 R_1, R_2, S_1 ，有关该三个关系的五种基本代数运算则用图3.2表示。

以上定义的五种基本代数运算，均可写出实现算法。现给出合并运算的算法作为示范。

算法：合并运算。输入：同类关系 R_1, R_2 。输出： $R_1 \cup R_2$

方法：① 建立关系 $W = R_1 \cup R_2$ 的框架；

② $W_i = R_1$ 的全部元组 (n_1 个)；Counter := n_1 ；

- ③ $j := 0;$
- ④ $j := j + 1;$
- ⑤ 如果 $j > n_2$ (R_2 中的元组个数) 则转 ⑭;
- ⑥ $Y := R_2$ 的第 j 个元组;
- ⑦ $i := 0;$
- ⑧ $i := i + 1;$
- ⑨ 如果 $i > n_1$ 则转 ⑬;
- ⑩ $X := R_1$ 的第 i 个元组;
- ⑪ 如果 $X \neq Y$ 则转 ⑧;
- ⑫ 转 ④;
- ⑬ $W := WUY; Counter := Counter + 1;$ 转 ④;
- ⑭ 输出;

在一般文献中, 除前述五种基本运算外, 还介绍有其他一些代数运算, 现将较常见的四种叙述如下。

(6) 相交 (intersection)。设有同类关系 R_1, R_2 , 定义它们的相交运算为

$$R_1 \cap R_2 = \{t \mid t \in R_1 \wedge t \in R_2\}$$

其中 $\cap$ 为相交运算符。本运算是求关系 R_1 和 R_2 中所有的相同元组, 其操作结果仍然是一个与 R_1, R_2 同类的关系, 如图 3.4。显然它与集合论中的交集运算是相同的, 象集合运算一样, 这里也有 $R_1 \cap R_2 = R_1 - (R_1 - R_2)$ 。

(7) 连接 (join)。设有 K_1 元关系 R 和 K_2 元关系 S , 以及算术比较运算符 θ , 则关系 R, S 关于 R 的第 i 列和 S 的第 j 列的 θ -连接运算定义为:

$$R \bowtie_{i\theta j} S = \{t \mid t = \langle t^{h_1}, t^{h_2} \rangle \wedge t^{h_1} \in R \wedge t^{h_2} \in S \wedge t^{h_1}_{i\theta j} t^{h_2}_{j\theta j}\}$$

其中 $\bowtie_{i\theta j}$ 为连接运算符, “ $i\theta j$ ”表示按照 R 的第 i 列与 S 的第 j 列之间满足 θ 运算进行连接, t^{h_1} 表示 R 的元组变量 $t^{h_1} = \langle t^{h_1}_1, t^{h_1}_2, \dots, t^{h_1}_{K_1} \rangle$ 的第 i 个分量, t^{h_2} 表示 S 的元组变量 $t^{h_2} = \langle t^{h_2}_1, t^{h_2}_2, \dots, t^{h_2}_{K_2} \rangle$ 的第 j 个分量, 此运算就是用 R 的第 m ($m=1, \dots, m_1$) 个元组的第 i 个分量与 S 的第 j 列分量从头至尾作 θ 比较, 每当满足这一比较运算时, 就把 S 中该分量所在的元组接在 R 的第 m 元组的右边构成一行, 作为新关系的一个元组, 反之, 每当不满足这 θ 比较运算时就继续作该 j 列分量的下一次比较。这样,

从 $m=1$ 直到 $m=m_1$ 为止, 就得到了新关系的全部元组 (图 3.4)。

从乘积和 θ -连接的定义可知, 后者可看作一种有选择的乘积, 其选择条件即是 θ 运算。据此推得 θ -连接能用乘积和选择的合成形式表示:

$$R \bowtie_{\theta} S = \sigma_{\theta(i_1, \dots, i_n)} (R \times S)$$

当 θ 为 “=” 时, $R \bowtie_{\theta} S$ 称等连接 (Equijoin) (连接运算中的列序号也可代之以列名)。

(8) 自然连接 (natural join)。设 K_1 元关系 R 与 K_2 元关系 S 有相同的属性名 $B_1, B_2, \dots, B_n$, 它们在 R 中的列号为 $i_1, i_2, \dots, i_n$, 在 S 中的列号为 $j_1, j_2, \dots, j_n$, 今用 $\tilde{t}^{h_2}$ 表示关系 S 的元组变量 $t^{h_2} = \langle t_1^{h_2}, t_2^{h_2}, \dots, t_{h_2}^{h_2} \rangle$ 去掉分量 $t_{j_1}^{h_2}, t_{j_2}^{h_2}, \dots, t_{j_n}^{h_2}$ 后, 所组成的新元组变量, 则 R 与 S 的自然连接定义为

$$R \bowtie S = \{ t \mid t = \langle t^{h_1}, \tilde{t}^{h_2} \rangle \wedge t^{h_1} \in R \wedge \tilde{t}^{h_2} \in S \wedge t_{i_1}^{h_1} = t_{j_1}^{h_2} \wedge t_{i_2}^{h_1} = t_{j_2}^{h_2} \wedge \dots \wedge t_{i_n}^{h_1} = t_{j_n}^{h_2} \}$$

其中 “ $\bowtie$ ” 为自然连接运算符。此运算的直观解释是, 用 R 的第 K 个元组的 $i_1, i_2, \dots, i_n$ 列的分量依次与 S 的第 m 个元组的 $j_1, j_2, \dots, j_n$ 列的 n 个分量比较, 如果都相等, 则从 S 的第 m 个元组中去掉被比较的 n 个分量, 并把剩下的分量依原序接在 R 的第 K 元组的右边, 这样构成的一行即为自然连接的一个元组, 若这 n 个比较至少有一个不相等, 则不进行如此连接。照这种办法, 让 K, m 分别跑遍关系 R, S 的每个元组就得到了连接的全部元组, 构成 $K_1 + (K_2 - n)$ 元的新关系。

设从分量表 $1, 2, \dots, K_1, K_1+1, K_1+2, \dots, K_1+K_2$ 中去掉分量 $K_1+j_1, K_1+j_2, \dots, K_1+j_n$ 后, 所剩下的分量记为 $1, 2, \dots, K_1, m_1, m_2, \dots, m_{K_2-n}$, 则根据乘积、选择和投影的定义, 可把自然连接视为 R 与 S 的乘积经过选择再投影的结果。即可表示成三种运算的合成形式:

$$R \bowtie S = \Pi_{1, \dots, K_1, m_1, \dots, m_{K_2-n}} (\sigma_{(t_{i_1} = t_{j_1}) \wedge \dots \wedge (t_{i_n} = t_{j_n})} (R \times S))$$

(9) 求商 (Quotient)。设有 K_1 元关系 R 和 K_2 元关系 S , 且 $K_1 > K_2$, $S \neq \phi$ (表示无任何元组的空关系), 以及关系 S 的 K_2 个分量名与 R 的最后 K_2 个分量名相同, 则关系 R 关于 S 的求商定义为

$$R \div S = \{ t \mid t = \langle t_1^{h_1}, t_2^{h_2}, \dots, t_{h_1-h_2}^{h_1-h_2} \rangle \wedge \text{“如果 } t_2^{h_2} \in S, \text{ 则 } \langle t, t_2^{h_2} \rangle \in R \text{”} \}$$

其中“ $\div$ ”为求商运算符。求商应视为乘积的逆运算，即商 $\times S$ 所得到的全部元组均应属于 R 。故求商又称为除法 (division)，也可表示成基本运算的合成形式

$$R \div S = \Pi_{1, 2, \dots, h_1-h_2}(R) - \Pi_{1, 2, \dots, h_1-h_2}((\Pi_{1, 2, \dots, h_1-h_2}(R) \times S) - R)$$

下面给出补充代数运算的实例。设有图 3.1 的关系、 R_1 、 R_2 、 S_1 和图 3.3 的关系 G 、 H ，则上述四种补充代数运算的结果如图 3.4 所示。

A	B	C
a	4	b
b	5	c
d	3	a
e	4	d

关系 G

B	C
3	9
4	8
4	7

关系 H

图 3.3

A_1	A_2	A_3
b	2	d
c	2	d

A_1
b

$R_1 \div S_1$

$R_1 \cap R_2$

A	B	C	D
a	4	b	8
a	4	b	9
d	3	a	9
e	4	a	7
e	4	a	7

$G \bowtie H$

A	B	C	B	D
a	4	b	3	9
b	5	c	3	9
b	5	c	4	8
b	5	c	4	7
e	4	d	3	9

$G \bowtie H$
 $2 > 1$

A	B	C	B	D
a	4	b	4	8
a	4	b	4	7
d	3	a	3	9
e	4	d	4	8
e	4	d	4	7

$G \bowtie H$
 $2 = 1$

A	B	C	B	D
a	4	b	3	9
a	4	b	4	8
a	4	b	4	7
b	5	c	3	9
b	5	c	4	8
b	5	c	4	7
d	3	a	3	9
d	3	a	4	8
d	3	a	4	7
e	4	d	3	9
e	4	d	4	8
e	4	d	4	7

$G \times H$

图 3.4 补充代数运算的实例

现举例说明补充代数运算的算法。

算法： θ -连接。设关系 R_1 、 R_2 分别有 n_1 、 n_2 个元组， θ 为算术比较运算符。

输入： R_1 、 R_2 、 θ 、 i 、 j

输出： $R_1 \bowtie_{\theta} R_2$

方法：①建立 $W = R_1 \bowtie_{\theta} R_2$ 的框架； $C := 0$ ；

② $L := 0$ ；

③ $L := L + 1$ ；

④ 如果 $L > n_1$ 则转⑫；

⑤ $X := R_1$ 的第 L 个元组；

⑥ $K := 0$ ；

⑦ $K := K + 1$ ；

⑧ 如果 $K > n_2$ 则转③；

⑨ $Y := R_2$ 的第 K 个元组；

⑩ 如果 $X(i) \theta Y(j)$ 不成立则转⑦；

⑪ $W := W \cup (X // Y)$ ； $C := C + 1$ ；转⑦；

⑫ 输出；

其中 $X(i)$ 表示 X 的第 i 个分量， $X // Y$ 表示把 Y 串接在 X 的右边。

在关系代数运算中，我们把由五种基本代数运算经有限次复合而成的式子称为关系代数运算表达式。这种表达式的运算结果是一个关系。

3.2.2. 元组关系演算 (tuple relational calculus)

把谓词演算推广到关系运算中，就得到了关系演算。关系演算可分为元组关系演算和域关系演算两种。前者以元组为变量，简称元组演算，后者以域为变量，简称域演算。在元组演算中，用演算表达式： $\{t \mid \varphi(t)\}$ 表示关系，其中 $\varphi(t)$ 为公式， t 为 φ 中唯一的自由元组变量。本段将仿照谓词演算方法，递归地定义元组演算公式。

(1) 原子命题函数是公式，称为原子公式 (atom formulas)，它有下面三种形式：

(a) $R(t)$ 。 R 是关系名， t 是元组变量， $R(t)$ 表示这样一个命题函数：“ t 是关系 R 的元组”。于是关系 R 可用元组演算表达式 $\{t \mid R(t)\}$ 表示。

(b) $t[i] \theta c$ 或 $c \theta t[i]$ 。 $t[i]$ 表示元组变量 t 的第 i 个分量， C 是常量， θ 为算术比较运算符。 $t[i] \theta c$ 或 $c \theta t[i]$ 表示的命题函数是：“元组 t 的第 i 个分量与 C 之间满足 θ 运算”。如： $t(2) > 6$ ，表示“ t 的第 2 个分量大于 6”。

(c) $t[i] \theta u[j]$ 。 t, u 是两个元组变量， $t[i] \theta u[j]$ 表示的命题函数是：“元组 t 的第 i 个分量与元组 u 的第 j 个分量之间满足 θ 运算。”如 $t(2) \neq u(3)$ ，表示“ t 的第 2 个分量不等于 u 的第 3 个分量”。

原子公式中的元组变量，在自身的范围内是自由变量。

(2) 设 φ_1, φ_2 是公式，则 $\neg \varphi_1$ ， $\varphi_1 \wedge \varphi_2$ ， $\varphi_1 \vee \varphi_2$ 也都是公式，这三个运算依次定义为：“若 φ_1 为真，则 $\neg \varphi_1$ 为假，若 φ_1 为假，则 $\neg \varphi_1$ 为真”，“如果 φ_1, φ_2 同时为真，则 $\varphi_1 \wedge \varphi_2$ 亦为真，否则 $\varphi_1 \wedge \varphi_2$ 为假”，“若 φ_1, φ_2 之中有一个或两个为真，则 $\varphi_1 \vee \varphi_2$ 为真，否则 $\varphi_1 \vee \varphi_2$ 为假”。

(3) 设 φ 是公式， t 是 φ 中的某个元组变量，那么 $(\forall t)(\varphi)$ 、 $(\exists t)(\varphi)$ 都是公式。这里 $(\forall t)$ 是一个量词，其含义为“对所有的 t ”或“对任意一个 t ”； $(\forall t)(\varphi)$ 表示“对所有的 t ， φ 为真时， $(\forall t)(\varphi)$ 为真，否则为假”。如公式 $(\forall t)(\neg R(t))$ 表示这样的命题：“所有的元组 t 都不属于 R ，即 R 是一个空关系”。因此，当 $R = \phi$ 时， $(\forall t)(\neg R(t))$ 为真；当 $R \neq \phi$ 时， $(\forall t)(\neg R(t))$ 为假。符号 $(\exists t)$ 也是一个量词，其含义为“存在这样的 t ”或“至少有这样一个 t ”， $(\exists t)(\varphi)$ 表示“若有一个 t 使 φ 为真，则 $(\exists t)(\varphi)$ 为真，否则为假”。如公式 $(\exists t)(R(t))$ 表示这样的命题：“关系 R 至少有一个元组，即 $R \neq \phi$ ”。因此，如果 $R \neq \phi$ ，则 $(\exists t)(R(t))$ 为真；如果 $R = \phi$ ，则 $(\exists t)(R(t))$ 为假。

(4) 在元组演算的公式中，各种运算符的运算优先次序为：

- (a) 算术比较运算符最高，
- (b) 量词次之，且按 $\exists$ 、 $\forall$ 的先后次序进行，
- (c) 逻辑运算符优先级最低，且按 $\neg$ 、 $\wedge$ 、 $\vee$ 的先后次序进行，
- (d) 加括号时，括号中的运算优先。

(5) 元组演算的所有公式均按 (1)、(2)、(3)、(4) 所确定的规则经有限次复合求得，不再存在其它形式。由此可见，元组关系演算公式无非是由它的原子公式经过有限次的算术比较运算和 $\neg$ 、 $\wedge$ 、 $\vee$ 及量词而形成的复合公式。

下面列举几个元组演算的示例。设有如图 3.5 的关系 R 、 S 、 W 。现给出如下四个元组演算(图 3.6)：

$$(a) R_1 = \{t \mid R(t) \wedge t(3) \geq 4\}; \quad (b) R_2 = \{t \mid R(t) \wedge S(t)\};$$

$$(c) R_3 = \{t \mid (\exists u)(R(u) \wedge W(u) \wedge t(3) < u(1))\};$$

A_1	A_2	A_3
a	e	8
c	f	6
d	b	4
d	f	3

R

A_1	A_2	A_3
a	e	3
b	c	5
d	b	4
d	f	6

S

B_1	B_2
4	x
5	d

W

图 3.5 三个已知关系

$$(d) \{t \mid (\exists u)(\exists v)(R(u) \wedge W(v) \wedge u(2) = f \wedge t(1)$$

$$= u(3) \wedge t(2) = u(2) \wedge t(3) = u(1) \wedge t(4) = v(2))\}$$

A_1	A_2	A_3
a	e	8
c	f	6
d	b	4

R_1

A_1	A_2	A_3
a	e	8
d	b	4

R_2

A_1	A_2	A_3
d	b	4
d	f	3

R_3

A_3	A_2	A_1	B_2
6	f	c	x
6	f	c	d
3	f	d	x
3	f	d	d

R_4

图 3.6 关系演算的例子

关系代数运算均可等价地用元组关系演算表达式表示。等价是指等价双方运算表达式的结果关系相同。由于其它关系代数表达式都能表示成基本关系代数运算的合成形式。若能证明五种基本关系代数运算表达式可以等价地表示成某种元组演算表达式，则一切关系代数运算表达式就都可以等价地表示成元组演算表达式。

设 R 、 S 为两个关系，则 $(a) R \cup S$ 可等价地表示为 $\{t \mid R(t) \vee S(t)\}$

证明：因为 $R(t)$ 与 $t \in R$ 是两个等价命题， $S(t)$ 与 $t \in S$ 也是等价命题，所以用 $R(t)$ 、 $S(t)$ 代替关系代数表达式 $R \cup S = \{t \mid t \in R \vee t \in S\}$ 中的 $t \in R$ 、 $t \in S$ ，就得到了欲证结果。

(b) $R - S$ 等价于 $\{t \mid R(t) \wedge \neg S(t)\}$

证明：同上，从略。

(c) $R \times S$ 等价于

$$\begin{aligned} & \{t \mid (t^{k1})(t^{k2})(R(t^{k1}) \wedge S(t^{k2}) \wedge t[1] = \\ & t^{k1}[1] \wedge t[2] = t^{k1}[2] \wedge \dots \\ & \wedge t[k_1] = t^{k1}[k_1] \wedge t[k_1+1] = t^{k2}[1] \wedge t[k_1+2] = \\ & t^{k2}[2] \wedge \dots \wedge t[k_1+k_2] = t^{k2}[k_2])\} \end{aligned}$$

式中 R, S 依次为 k_1, k_2 元关系, t^{k1}, t^{k2} 表示 R, S 的元组变量。

证明：因为 $R \times S = \{t \mid t = \langle t^{k1}, t^{k2} \rangle \wedge t^{k1} \in R \wedge t^{k2} \in S\}$, 且 $t = \langle t^{k1}, t^{k2} \rangle$ 等价于 $t[1] = t^{k1}[1] \wedge t[2] = t^{k1}[2] \wedge \dots$
 $\wedge t[k_1] = t^{k1}[k_1] \wedge t[k_1+1] = t^{k2}[1] \wedge$
 $\wedge t[k_1+k_2] = t^{k2}[k_2]$,

以及 $t^{k1} \in R$ 等价于 $R(t^{k1})$, $t^{k2} \in S$ 等价于 $S(t^{k2})$, 经等价代换可得证结果。

(d) $\prod_{i_1, i_2, \dots, i_n} (R)$ 等价于

$$\begin{aligned} & \{t \mid (\exists u)(R(u) \wedge t[1] = u[i_1] \wedge t[2] = u[i_2] \wedge \dots \\ & \wedge t[n] = u[i_n])\} \end{aligned}$$

证明从略。

(e) $\sigma_F(R)$ 等价于 $\{t \mid R(t) \wedge F'\}$

其中 F' 是一个公式, 用 $t[i]$ 代替 F 中 t 的第 i 个分量即为 F' 。证明很简单, 无需赘述。

3.2.3 域关系演算 (domain relational calculus)

域关系演算类似于元组演算。所不同的是公式中的变量不是元组变量, 而是表示元组变量各个分量的域变量。域演算表达式的一般形式为:

$$\{t_1 t_2 \dots t_k \mid \varphi(t_1, t_2, \dots, t_k)\}$$

其中 $t_1, t_2, \dots, t_k$ 为元组变量 t 的各个分量, 统称为域变量, φ 是一个公式, 类似于元组演算公式。现模仿元组演算的方法递归地定义域演算公式如下:

(1) 原子命题函数是公式, 域演算也有三种形式的原子命题函数或原子公式:

(a) $R(t_1 t_2 \dots t_k)$, R 是 K 元关系, t_i 是域变量或常量。 $R(t_1 t_2 \dots t_k)$ 表示命题函数。 “以 $t_1, t_2, \dots, t_k$ 为分量的元组在关系 R 中”。