

# 第2章 关系数据库

## 1 绪论

关系模型具有严格的数学基础

应用数学方法处理数据库中的数据

奠定关系数据库理论基础的人——美国**IBM**公司的**E.F.Codd**。

模型的提出（**1970**年）——**E.F.Codd**的一篇论文

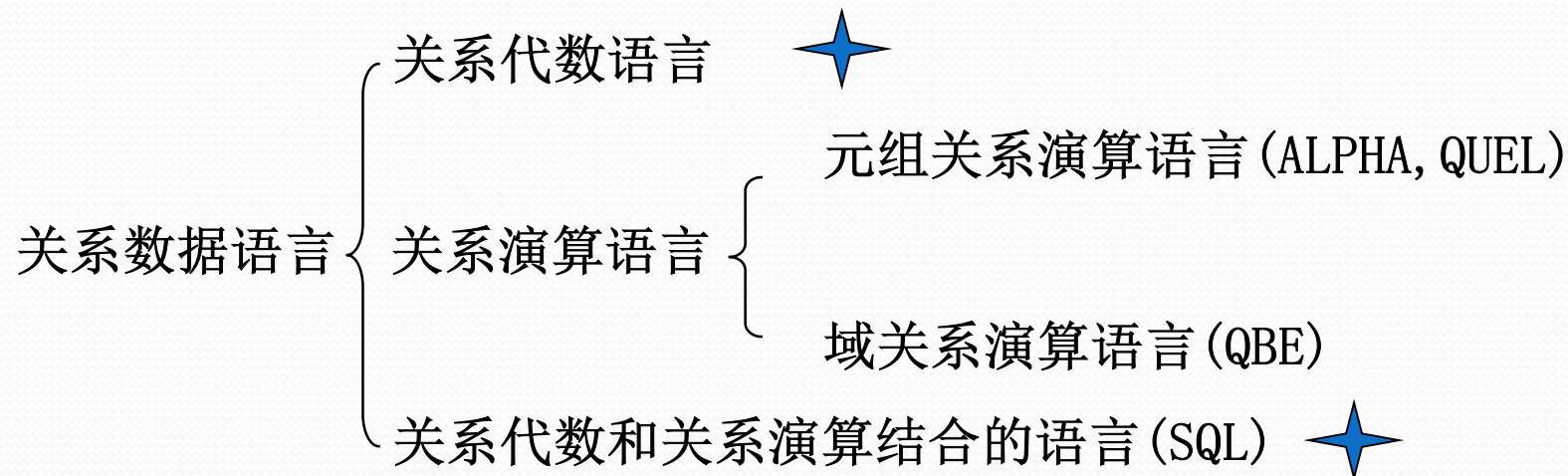
“**A Relational Model of Data for Shared Data Banks**”系统、严格的提出了关系模型，开创了数据库系统的新纪元。

- 20世纪70年代末关系方法的理论研究已经取得了很大的成果，其中有两大研究机构及其试验系统：
  - IBM公司的**System R**系统、
  - 美国加州大学伯克利分校的**INGRES**系统。
- 1981年关系数据库的**软件产品**就问世了。
- 代表性的商业数据库系统
  - Oracle, Informix(IBM收购), SQL Server, Sybase, DB2
  - Access, Foxpro, Foxbase



## ■ 关系模型的组成

- 关系数据结构（实体及实体间的联系均用二维表来表示）
- 关系操作（查询及增、删、改操作两大部分）



### 非过程式语言

- 关系的完整性（实体完整性，参照完整性，用户定义完整性）

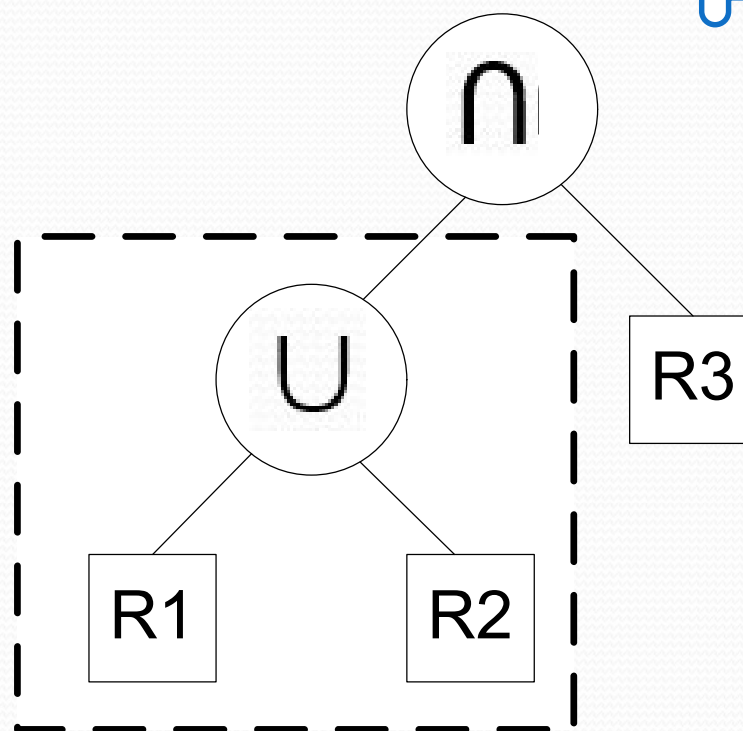
# 关系代数

$$(R1 \cup R2) \cap R3$$

运算的  
多次复合

多层迭代

代数算式的  
逻辑表达能力





# SQL

- S? ( Structured, 结构化) Query Language

**SELECT** 学号, 姓名

**FROM** 学生表

**WHERE** 学号 IN

(**SELECT** 学号

**FROM** 选修表

**WHERE** 课号 IN

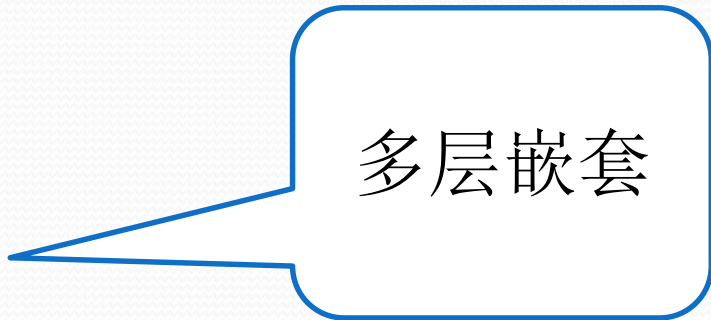
(**SELECT** 课号

**FROM** 课程表, 院系

**WHERE** 课程名 LIKE 'DATA%'

**AND** 课程表.开课院系编号=院系.院系编号

**AND** 院系.院系名称= '计算机' ))



多层嵌套

## 2 基本概念

### 2.1 域(domain)



——一组具有相同数据类型的值的集合。

例：整数，实数， $\leq 500$ 的整数，性别(男、女)、字符串。

### 2.2 笛卡尔积(Cartesian product)



#### 1. 定义

给定一组域 $D_1, D_2, \dots, D_n$ ，则其笛卡尔积为：

$$D_1 \times D_2 \times \dots \times D_n = \{(d_1, d_2, \dots, d_n) \mid d_j \in D_j, j=1, 2, \dots, n\}$$

#### 2. 说明

- 1)  $(d_1, \dots, d_n)$ 为集合中的一个元素，称为n元组(n-tuple)，简称元组。
- 2) 元组中每个值 $d_i$ 称为分量



3) 集合中元素无序

$$\{(a,1), (b,2), (c,3)\} = \{(b,2), (a,1), (c,3)\} \\ = \{(c,3), (b,2), (a,1)\}$$

4) 元组中分量有序

$(a, b, c) \neq (b, a, c)$  属性及其值的对应性。

5) 若干个域的笛卡尔积是一个二维表

例 设有三个域:

$D_1 = \text{男士集合} = \{\text{刘英}, \text{刘加加}\}$

$D_2 = \text{女士集合} = \{\text{白雪}, \text{白乘乘}\}$

$D = \text{儿童集合} = \{\text{刘学}, \text{刘一学}, \text{刘楚楚}\}$

则 $D_1, D_2, D_3$ 的笛卡尔积为如下一张二维表:

男 士	女 士	儿 童
刘英	白雪	刘学
刘英	白雪	刘一带
刘英	白雪	刘楚楚
刘英	白乘乘	刘学
刘英	白乘乘	刘一学
刘英	白乘乘	刘楚楚
刘加加	白雪	刘学
刘加加	白雪	刘一学
刘加加	白雪	刘楚楚
刘加加	白乘乘	刘学
刘加加	白乘乘	刘一学
刘加加	白乘乘	刘楚楚

基数

笛卡尔积基数 =  $2 \times 2 \times 3 = 12$  (12 个元组)

男士基数

女士基数

儿童基数



## 2.3 关系

### 1. 定义

$D_1 \times D_2 \times \dots \times D_n$  的任意子集称为在域  $D_1, D_2, \dots, D_n$  上的关系。

记为:  $R(D_1, D_2, \dots, D_n)$

### 2. 说明

1)  $R$  为**关系名**,  $n$  为关系的目或度(degree);

2) 关系是一张二维表;

3) 可多个**候选码**(candidate key);

4) 任选候选码之一为**主码**(primary key)。

主属性、  
非主属性、  
全码

例: 可从上表中取出一个有意义子集作为一个关系

男 士	女 士	儿 童
刘英	白雪	刘学
刘加加	白乘乘	刘一学
刘加加	白乘乘	刘楚楚



丈 夫	妻 子	孩 子
刘英	白雪	刘学
刘加加	白乘乘	刘一学

## 2.4 外码(foreign key)

——对于 $R_1$ 和 $R_2$ ,  $A_1, \dots, A_n$ 为其属性子集, 若 $A_1, A_2, \dots, A_n$ 不是 $R_1$ 的码, 但它是 $R_2$ 的码, 且两者相对应, 则称 $A_1, \dots, A_n$ 为 $R_1$ 的外码。

Student (XH, XM)

Course (KH, KM)

SC (XH, KH, CJ)      SC中的XH, KH均为外码。

## 2.5 关系模式(Relation Schema)

### 1. 定义

关系的描述:  $R(A_1, \dots, A_n)$

即:  $R(U, D, \text{DOM}, F)$

R: 关系名。

U: R中的属性名序列。

D: 域(取值范围)。

DOM: 属性到域的映象集(属性类型、长度)。

F: 属性间数据依赖关系。





## 2.6 关系数据库

1. 型：若干关系模式的集合(内含)。
2. 值：某一时刻每个关系模式对应的具体关系集(外延)。

## 2.7 视图(View)

View as table

## 2.8 关系的完整性

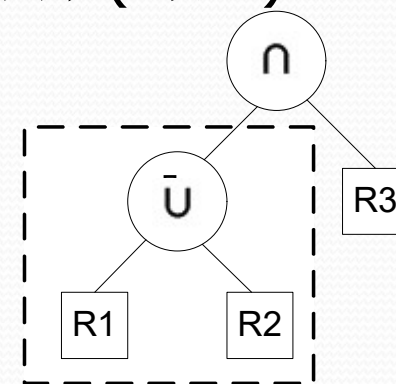
### 1. 实体完整性(Entity integrity)

——主码属性不能为空，不同元组的主码取值不能重复。

### 2. 参照完整性(Referential integrity)

——若关系 $R_1$ 中含有另一个关系 $R_2$ 中主码的属性组 $F$ (作为 $R_1$ 的外部KEY)，则对于 $R_1$ 的每个元组在 $F$ 上的值必须满足：

- 1) 空，或者
- 2) 等于 $R_2$ 中某个元组的主码值



例: EMPL(ENO, ENAME, DNO)

DEPT(DNO, DNAME)

则对于EMPL中每个DNO的值必须为:

- 取空值(说明该职工还未分配到某部门)

或者

- DEPT中某个元组的DNO值(不可能分配到一个不存在的部门)

关系的两个不变性: 实体完整性、参照完整性, 系统自动支持。

3. 用户定义完整性(user-defined integrity)

——用户定义的约束。

出租期限 $\leq 20$ 年,

工资 $\geq 4000$ 元



### 3 关系代数操作

$+$   $-$   $\times$   $\div$

数学——数据

#### 3.1 概述

1、含义：用对关系的运算来表达查询的一种传统方式。

2、分类：

1) 传统集合运算

并 ( $\cup$ )，交 ( $\cap$ )，差 ( $-$ )，笛卡尔积 ( $\times$ )

2) 专门的关系运算

投影 ( $\pi$ )，选择 ( $\sigma$ )，连接 ( $\bowtie$ )，除 ( $\div$ )

3、运算符

1) 集合运算符： $\cup$ 、 $\cap$ 、 $-$ 、 $\times$

2) 专门运算符： $\pi$ 、 $\sigma$ 、 $\bowtie$ 、 $\div$

3) 比较运算符： $>$ 、 $\geq$ 、 $<$ 、 $\leq$ 、 $=$ 、 $\neq$

4) 逻辑运算符： $\neg$ 、 $\wedge$ 、 $\vee$

5) 括号运算符： $( )$

符号书写区别明显

## 4、特殊记号

1) 设有关系模式  $R(A_1, A_2, \dots, A_i, \dots, A_n)$

则:

$t \in R$ :  $t$  是  $R$  的一个元组。

$t[A_i]$ : 元组  $t$  中相应属性  $A_i$  的一个分量。

Student

XH	XM	XB
2001 2007	于一 牛二	男 男

$\langle \dots \rangle \quad t$

“男”  $A_i$  (XB)

$t$  的一个分量



2) 设  $A = \{A_{i1}, A_{i2}, \dots, A_{ik}\}$ ,  $A_{i1}, A_{i2}, \dots, A_{ik}$  是  $A_1, A_2, \dots, A_n$  中的一部分, 则:

$A$ : 属性列或域列。

$\overline{A}$ :  $\{A_1, A_2, \dots, A_n\}$  中去掉  $(A_{i1}, A_{i2}, \dots, A_{ik})$  后剩余的属性组。

$t[A] = (t[A_{i1}], t[A_{i2}], \dots, t[A_{ik}])$ : 元组  $t$  在属性  $A$  上诸分量的集合

3) 设  $R$  为  $n$  目关系,  $S$  为  $m$  目关系,  $\widehat{t_r t_s}$  则:

$t_r \in R, t_s \in S$ :  $R$  和  $S$  的元组的连接 是一个  $n+m$  列元组; 前  $n$  个分量是  $R$  的一个  $n$  元组; 后  $m$  个分量是  $s$  的一个  $m$  元组, 又称元组的连串 (Concatenation)。

4) 设有关系 $R(X, Z)$ ， $X$ 、 $Z$ 为属性组，则：

当 $t[X] = x$ 时， $x$ 在 $R$ 中的象集 (image set) 为：

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

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： $R$ 为（学号，课程）设 $X$ 为学号，则 $Z$ 为课程，求 $x=1$ 的象集。

学号	课程
1	C语言
1	数据结构
1	数据库
2	C语言

课程
C语言
数据结构
数据库



- 关系改变后，学号1的象集？

学号	课程	成绩
1	C语言	85
1	数据结构	88
1	数据库	85
1	C语言	59
2	C语言	80

课程	成绩
C语言	85
数据结构	88
数据库	85
C语言	59

会员号甲	亲友团意见	会员号乙
1	同意	81
1	同意	82
1	不同意	83
1	不同意	84
2	同意	81
2	不同意	82



## 3.2 传统集合运算

### 1. 并 (union)

1) 定义：设有关两个n目关系R、S，则 $R \cup S$ 表示是由属于R或属于S的元组组成。

### 2) 特征

➤ 结果为n目关系： $R \cup S = \{t \mid t \in R \vee t \in S\}$ ;

➤ 参加运算的对象为两个关系;

从何而来?  
查询得来

➤ R、S属性同类（取自同一个域）;

➤ 相同元组取其一;

➤ 从“行”上取值。

3) 作用：将一个新元组集加入到原关系中去。

例: R

A	B	C
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>

S

A	B	C
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>1</sub>	b <sub>3</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>



则 $R \cup S$ 结果为:

R

A	B	C
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>

S

A	B	C
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>1</sub>	b <sub>3</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>

U

=

计算过程

A	B	C
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>
a <sub>1</sub>	b <sub>3</sub>	c <sub>2</sub>

## 2、交（intersection）

1) 定义：设有两个n目关系R、S， $R \cap S$ 是由既属于R同时又属于S的元组组成。

R				S			
A	B	C		A	B	C	
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>		a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>	
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>	$\cap$	a <sub>1</sub>	b <sub>3</sub>	c <sub>2</sub>	=
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>		a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>	

A	B	C
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>



## 2) 特征

- 结果为n目关系:  $R \cap S = \{t | t \in R \wedge t \in S\}$ ;
- 参加运算的为两个同目关系;
- R、S属性同类;
- 从“行”上取值。

3) 作用: 从两个关系中找出相同元组。

### 3、差（difference）

1) 定义：设有两个n目关系R、S，则R-S是由属于R不属于S的元组组成。

R				S			
A	B	C		A	B	C	
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>		a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>	
a <sub>1</sub>	b <sub>2</sub>	c <sub>2</sub>	—	a <sub>1</sub>	b <sub>3</sub>	c <sub>2</sub>	=
a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>		a <sub>2</sub>	b <sub>2</sub>	c <sub>1</sub>	

A	B	C
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>



## 2) 特征

- 结果为n目关系:  $R-S = \{t | t \in R \wedge \neg t \in S\}$ ;
- 参加运算的为两个同目关系;
- R、S同类;
- 从“行”上取值。

3) 作用: 从一个关系中删去某些元组。

## 4、笛卡尔积 (Cartesian product)

1) 定义: 设R为n目关系, S为m目关系, 则 $R \times S$ 是一个由R和S的所有元组连接在一起而组成的  $(n+m)$  列元组的集合。每一元组的前n个列是R的一个元组, 后m列是s的一个元组。

$R \times S$ 

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

开销? 内存!!  $\rightarrow$  I/O!!!



## 2) 特征

- 结果为 $(n+m)$ 目关系:  $R \times S = \widehat{\{t_r t_s \mid t_r \in R \wedge t_s \in S\}}$ ;
- 参加运算的是两个关系;
- $R$ 、 $S$ 不同类 (实际上);
- 从“行”上取值。

3) 作用: 将两个关系按元组连接组成一个新关系。

## 2.2.3 专门的关系运算

### 1、选择 (selection)

1) 定义：从指定关系R中选取满足条件的元组集的运算。

记作：  $\sigma_F(R) = \{t \mid t \in R \wedge F(t) = \text{‘真’}\}$

F：逻辑表达式（选择对象应满足的条件），一般表示为：

$X_1 \theta Y_1 [\phi X_2 \theta Y_2] \dots$

$X_1, Y_1$ ：属性 | 常量 | 简单函数 | 列顺序号

$\theta$ ：比较运算符：>、≥、<、≤、=、≠

$\phi$ ：逻辑运算符：¬、∧、∨

[ ]：任选标识符，其中内容可有可无

$A > 12, B = \text{‘田野’}, \sigma_3 = \text{‘8’} \wedge B \neq 5 (R)$



设:

student

SNO	NAME	SEX	AGE	DEPT
9901	于一	男	24	计算机
9903	牛二	男	23	动力
9904	白三	女	22	计算机
9902	马四	男	23	自控

SC

SNO	CNO	GRADE
9901	001	72
9902	001	70
9903	003	72
9904	002	85
9903	001	72

## Course

CNO	CNAME	CREDIT
001	DB	3
002	OS	3
003	C	2
004	AI	2



# 例1 查计算机系学生 student

SNO	NAME	SEX	AGE	DEPT
9901	于一	男	24	计算机
9903	牛二	男	23	动力
9904	白三	女	22	计算机
9902	马四	男	23	自控

$\sigma_{\text{DEPT} = \text{'计算机'}}(\text{Student})$

SNO	NAME	SEX	AGE	DEPT
9901	于一	男	24	计算机
9904	白三	女	22	计算机

## 例2 查“学分<3分”的课程

Course

CNO	CNAME	CREDIT
001	DB	3
002	OS	3
003	C	2
004	AI	2

$\sigma_{\text{CREDIT} < 3}(\text{Course})$

CNO	CNAME	CREDIT
003	C	2
004	AI	2



例3：查询成绩大于90分或者小于45分的选课记录

SC

SNO	CNO	GRADE
9901	001	72
9902	001	30
9903	003	72
9904	002	95
9903	001	72

查9902号~~和~~9904号学生的选课信息？

$\sigma_{SNO=9902 \vee SNO=9904} (SC)$

查选修了002号课程~~并且~~  
~~没有~~选修004号课程的学生选课信息？

$\sigma_{GRADE>90 \vee GRADE<45} (SC)$

SNO	CNO	GRADE
9902	001	30
9904	002	95

## 2) 说明

- 参加运算的是一个关系;
- 从行上取值。

## 2、投影 (projection)

1) 定义: 从指定关系R中选出若干属性列的运算。

记作:  $\pi_A(R) = \{t[A] | t \in R\}$

A: R中的若干属性列名或列顺序号。

注: 结果中**去掉重复元组**。





例3：查学生的姓名和年龄。

$\pi_{\text{NAME, AGE}}(\text{Student})$

或  $\pi_{2,4}(\text{Student})$

**student**

SNO	NAME	SEX	AGE	DEPT
9901	于一	男	24	计算机
9903	牛二	男	23	动力
9904	白三	女	22	计算机
9902	马四	男	23	自控

NAME	AGE
于一	24
牛二	23
白三	22
马四	23

例4：查招有学生的系有哪些

$\pi_{\text{DEPT}}$  (Student)  
或  $\pi_5$  (Student)

student

SNO	NAME	SEX	AGE	DEPT
9901	于一	男	24	计算机
9903	牛二	男	23	动力
9904	白三	女	22	计算机
9902	马四	男	23	自控

DEPT
计算机
动力
自控

去掉一个重复元组“计算机”

$\exists$ , 存在量词

2) 说明:

- 参加运算的只有一个关系;
- 从列上取值。



### 3、连接（Join）

1) 定义：从两个指定关系R和S中选取满足给定条件的元组连串的操作。

记为：

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

#### ■ 元组的连串（Concatenation）

■ 若  $r = (r_1, \dots, r_n)$ ,  $s = (s_1, \dots, s_m)$ , 则定义r与s的连串为：  
 $rs = (r_1, \dots, r_n, s_1, \dots, s_m)$



设有如下关系R和S:

R

A	B	C
a <sub>1</sub>	b <sub>1</sub>	5
a <sub>1</sub>	b <sub>2</sub>	6
a <sub>2</sub>	b <sub>3</sub>	8
a <sub>2</sub>	b <sub>4</sub>	12

S

B	E
b <sub>1</sub>	3
b <sub>2</sub>	7
b <sub>3</sub>	10
b <sub>3</sub>	2
b <sub>5</sub>	2

2) 说明

运算步骤:

①笛卡尔积 → ②取条件满足者



例5  $R \bowtie_{c \leq E} S$

笛卡尔积  
( $R \times S$ ):

	(R)		(S)	
A	B	C	B	E
$a_1$	$b_1$	5	$b_1$	3
$a_1$	$b_1$	5	$b_2$	7
$a_1$	$b_1$	5	$b_3$	10
$a_1$	$b_1$	5	$b_3$	2
$a_1$	$b_1$	5	$b_5$	2
$a_1$	$b_2$	6	$b_1$	3
		...	...	
$a_2$	$b_3$	8	$b_1$	3
		...	...	
$a_2$	$b_4$	12	$b_1$	3
$a_2$	$b_4$	12	$b_2$	7
		...	...	
$a_2$	$b_4$	12	$b_5$	2

取其中 $C < E$ 的元组:

A	B(R)	C	B(S)	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

参加运算的是两个关系;

- 参加运算的关系不一定同目;
- 从行上取值。

实际求解过程?

Nested loop

3) 等值连接 (equi-join)

$\theta$  仅为 “=” 的连接运算



#### 4) 自然连接 (Natural join)

一种**特殊的等值连接**，参加运算的指定关系R和S中用于比较的分量必须是相同的属性集，且结果中**去掉重复属性**的运算。

➤ 必须含公共属性

如R和S中的B。

➤ 运算步骤：

➤ 计算 $R \times S$ ；

➤ 选出比较值相等的元组；

➤ 去掉重复属性。

记为：  $R \bowtie S$

例6：上例中的RS结果：

·笛卡尔积同上

·取 $B(R) = B(S)$ 的元组

A	B(R)	C	B(S)	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

R

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

S

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



去掉重复属性B之一：

A	B	C	E
a <sub>1</sub>	b <sub>1</sub>	5	3
a <sub>1</sub>	b <sub>2</sub>	6	7
a <sub>2</sub>	b <sub>3</sub>	8	10
a <sub>2</sub>	b <sub>3</sub>	8	2

## 自然连接再举例

R

A	B	C
1	2	3
4	5	6
7	8	9
10	11	12

S

B	C	D
2	3	4
3	4	5
5	6	7
7	8	9
8	9	10

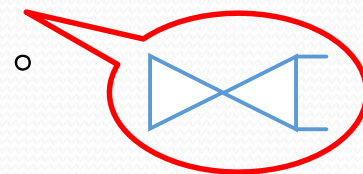
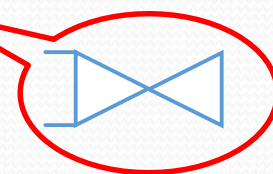
则  $R \bowtie S =$

A	B	C	D
1	2	3	4
4	5	6	7
7	8	9	10



## 补充概念：外连接、左外连接、右外连接

- 在连接运算中某些记录会由于在另一个关系中找到满足连接条件的元组（**悬浮元组, dangling tuple**）而被舍弃，如果把被舍弃的悬浮记录也保存在结果关系中，在其他属性上填空值（Null），则这种连接运算叫做外连接（outer join）。
- 左外连接（LEFT OUTER JOIN）：保存运算符左边的关系中要舍弃的记录（悬浮元组）。
- 右外连接（RIGHT OUTER JOIN）：保存运算符右边的关系中要舍弃的记录（悬浮元组）。



## 4、除 (division)

一般Y取同名属性组

### 1) 定义:

给定关系R (X, Y) 和S (Y, Z), 其中X、Y、Z为属性组, R中的Y与S中的Y可以有不同的属性名, 但必须具有相同的域集。R除S得到一个新关系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X上分量值x的象集 $Y_x$ 包含S在Y上投影的集合。

记作:

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

其中:

$Y_x$ : X在R中的象集,  $X = \text{tr} [X]$

$\forall$ , 全称量词



例6：有如下关系R、S：

R

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

S

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

$R \div S$

A
$a_1$

则  $R \div S = \{a_1\}$

**R****S** **$R \div S$** 

A	B	C	D				C	D	F			A	B
a1	b2	c3	d5				c3	d5	f3			a1	b2
a1	b2	c4	d6				c4	d6	f4				
a2	b4	c1	d3										
a3	b5	c2	d8										



R

S

 $R \div S$ 

A	B	C	D				C	D	F			A	B
a1	b2	c3	d5				c3	d5	f3			a1	b2
a1	b2	c4	d6				c4	d6	f4				
a2	b4	c1	d3										
a3	b5	c2	d8										

依据除法的定义，  
 $X = \{A, B\} \Rightarrow \{(a1, b2), (a2, b4), (a3, b5)\}$ ，  
 $Y = \{C, D\} \Rightarrow \{(c3, d5), (c4, d6)\}$ ，  
 $Z = \{F\} \Rightarrow \{f3, f4\}$ 。

R

S

 $R \div S$ 

A	B	C	D				C	D	F			A	B
a1	b2	c3	d5				c3	d5	f3			a1	b2
a1	b2	c4	d6				c4	d6	f4				
a2	b4	c1	d3										
a3	b5	c2	d8										

R在X上各个分量值的象集分别为：

- (a1,b2)的象集为{(c3,d5),(c4,d6)}
- (a2,b4)的象集为{(c1,d3)}
- (a3,b5)的象集为{(c2,d8)}
- S在Y上的投影为{(c3,d5),(c4,d6)}



R

S

 $R \div S$ 

A	B	C	D				C	D	F			A	B
a1	b2	c3	d5				c3	d5	f3			a1	b2
a1	b2	c4	d6				c4	d6	f4				
a2	b4	c1	d3										
a3	b5	c2	d8										

显然只有(a1,b2)的象集包含S在Y上的投影，所以 $R \div S = \{(a1, b2)\}$

实际应用中可考虑除运算表达式的构造方法，Z属性集可为空。

## ■ 5种基本运算

- 并、差、笛卡尔积、
- 投影、选择

## ■ 其它运算

- 交、连接、除
- 均可用5种基本运算来表达，引进它们并不增加语言的能力，但可以简化表达

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

$$\bullet R \bowtie S = \Pi_{\text{属性列表}}(\sigma_{\text{相同的属性列值相等}}(R) (R \times S))$$

$$\bullet R \div S = \Pi_X(R) - \Pi_X(\Pi_X(R) \times \Pi_Y(S) - R)$$



$\exists, \forall$ 

例:  $R \div S = \Pi_X(R) - \Pi_X(\Pi_X(R) \times \Pi_Y(S) - R)$

R

A	B	C	D
a	b	c	d
a	b	e	f
a	b	d	e
b	c	e	f
e	d	c	d
e	d	e	f

S

C	D
c	d
e	f

$\Pi_{AB}(R)$

A	B
a	b
b	c
e	d

$\Pi_{AB}(R) \times \Pi_{CD}(S)$

A	B	C	D
a	b	c	d
a	b	e	f
b	c	c	d
b	c	e	f
e	d	c	d
e	d	e	f

理解?

$\Pi_{AB}(R) \times \Pi_{CD}(S) - R$

A	B	C	D
b	c	c	d

$$R \div S = \begin{array}{|c|c|} \hline A & B \\ \hline a & b \\ b & c \\ e & d \\ \hline \end{array} - \begin{array}{|c|c|} \hline A & B \\ \hline b & c \\ \hline \end{array} = \begin{array}{|c|c|} \hline A & B \\ \hline a & b \\ e & d \\ \hline \end{array}$$

### 3.3 应用实例

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

例1：设有关系

教师（工作证号、姓名、性别、出生年份、职称、所在院系）  
 $TL(TNO, TNAME, TSEX, BYEAR, RANK, DEPT)$ ;

教学记录（工作证号、开课时间、课号、课时）  
 $CR(TNO, CTIME, CNO, CNUM)$ ;

$\sigma_{DEPT='计算机'}(TL)$  ,  $\sigma_{DEPT='自控'}(TL)$  ,  $\sigma_{RANK='5'}(TL)$  ,  
 $\sigma_{BYEAR>2000}(TL)$  ,

$\Pi_{TNO, TNAME, TSEX, DEPT} ( \sigma_{CNO='001'} ( ( ( \sigma_{DEPT='计算机'}(TL) \cup \sigma_{DEPT='自控'}(TL) ) \cap \sigma_{RANK='5'}(TL) - \sigma_{BYEAR>2000}(TL) ) \bowtie CR ) )$



例2：设有如下关系：

PER (PID, PNAME, AGE) 描述人的身份证号、姓名和年龄，

HOBBY (HNO, CONT) 描述各种爱好的编号和内容，

PH (PID, HNO) 描述人的各项爱好。

1) 解释下述关系代数表达式的含义

$\Pi_{\text{PER.PID, PNAME}}(\text{PER} \bowtie \delta_{\text{CONT}='微信聊天'}(\text{HOBBY} \bowtie \text{PH}))$

喜爱\*\*\*的人的\*\*\*

2) 写出查询没有任何兴趣爱好的人的身份证号和年龄的关系代数表达式。

$\Pi_{\text{PID, AGE}}(\text{PER}) - \Pi_{\text{PER.PID, AGE}}(\text{PER} \bowtie \text{PH})$

$\Pi_{\text{PID, AGE}}(\text{PER}) - \Pi_{\text{PER.PID, AGE}}(\text{PER} \bowtie \Pi_{\text{PID}}(\text{PH}))$

例3 设教学数据库有三个关系：

学生关系S（学号，姓名，性别，年龄，所在系）

$S(Sno, Sname, Ssex, Sage, Sdept)$

课程关系C（课程号，课程名，先修课，学分）

$C(Cno, Cname, Cpno, Ccredit)$

学习关系SC（学号，课程号，成绩）

$SC(Sno, Cno, Grade)$

下面用关系代数表达式来表达一些查询语句。

写这样语句的步骤→

- ①确定已知条件所在的关系
- ②确定输出内容所在的关系
- ③写出查询条件的表达



1) 查询学习课程号为C2的学生学号和成绩（涉及一个关系）。

$$\Pi_{\text{sno, grade}} \left( \sigma_{\text{cno}='c2'} (SC) \right)$$

2) 查询学习课程号为C2的学生学号和姓名（涉及两个关系）。

$$\Pi_{\text{sno, sname}} \left( \sigma_{\text{cno}='c2'} (S \bowtie SC) \right)$$

3) 查询选修了课程名为“数据库”的学生学号和姓名（涉及三个关系）。

$$\Pi_{\text{sno, sname}} \left( \sigma_{\text{cname}='数据库'} (S \bowtie SC \bowtie C) \right)$$

4) 查询选修了课程号为C1或C2的学生学号。

$$\Pi_{\text{sno}} \left( \sigma_{\text{cno}='c1' \vee \text{cno}='c2'} (SC) \right)$$

5) 查询至少选修了课程号为C2和C4的学生学号。

$$\Pi_{\text{sno}} \left( \sigma_{[1]=[4] \wedge [2]='C2' \wedge [5]='C4'} (SC \times SC) \right)$$

6) 查询不学C2课程的学生姓名、年龄。

$$\Pi_{\text{sname}, \text{sage}} (S) - \Pi_{\text{sname}, \text{sage}} \left( \sigma_{\text{cno}='C2'} (S \bowtie SC) \right)$$

所有学生          学了C2的学生

7) 查询选修了所有课程的学生姓名、年龄。

$$\Pi_{\text{sname}, \text{sage}} \left( S \bowtie \left( \Pi_{\text{sno}, \text{cno}} (SC) \div \Pi_{\text{cno}} (C) \right) \right)$$

求学了全部课程课学生学号

8) 查询所学课程包含了学生S2所学课程的学生学号。

$$\Pi_{\text{sno}, \text{cno}} (SC) \div \Pi_{\text{cno}} \left( \sigma_{\text{sno}='s2'} (SC) \right)$$



## 4 关系演算

- 以数理逻辑中的谓词演算为基础
- 元组关系演算，例ALPHA
- 域关系演算，例QBE

### 4.1 元组关系演算

- 形式化定义

$$\{ t \mid P(t) \}$$

表示所有使谓词 $P$ 为真的元组集合

- $t$ 为元组变量

如果元组变量前有“全称” ( $\forall$ ) 或“存在” ( $\exists$ ) 量词，则称其为约束元组变量，否则称为自由元组变量

- $P$ 是公式——由原子公式和运算符组成

## ■ 原子公式

- $t \in R$ , 写成  $R(t)$

- $t$  是关系  $R$  中的一个元组

- $t[i] \theta u[j]$

- $t[i]$  与  $u[j]$  为元组分量, 他们之间满足比较关系  $\theta$ , 元组  $t$  的第  $i$  个分量与元组  $u$  的第  $j$  个分量满足比较关系  $\theta$

- $t[i] \theta c$

- 分量  $t[i]$  与常量  $c$  之间满足比较关系  $\theta$

经典三问? ? ?



## ■ 公式的递归定义

- 原子公式是公式
- 若 $P$ 是公式, 那么 $\neg P$ 也是公式
- 若 $P_1, P_2$ 是公式, 则 $P_1 \wedge P_2, P_1 \vee P_2, \neg P_2$ 也是公式
  - 若 $P_1, P_2$ 同时为真, 则 $P_1 \wedge P_2$ 为真, 否则为假;
  - 若 $P_1, P_2$ 同时或有一个为真, 则 $P_1 \vee P_2$ 为真, 仅当同时为假, 则 $P_1 \vee P_2$ 为假;
  - 若 $P_1$ 为真, 则 $\neg P_1$ 为假。
- 如果 $P$ 是公式, 则 $\exists t (P)$ 也是公式
- 如果 $P$ 是公式, 则 $\forall t (P)$ 也是公式

## ■ 运算优先级(从高到低)

- 算术比较运算符最高
- 量词次之,  $\exists$ 高于 $\forall$
- 逻辑运算符: $\neg, \wedge, \vee$
- 括号优先

元组演算表达式举例：

$\{ t \mid S(t) \wedge t[A] > 2 \}$       S中A属性大于2的元组的集合

$\{ t \mid R(t) \wedge \neg S(t) \}$       R中不在S中出现的元组的集合

$\{ t \mid (\exists u) (S(t) \wedge R(u) \wedge t[C] < u[B]) \}$       S中满足下述条件的元组的集合：

C属性小于R中某个元组B属性的值。

$\{ t \mid (\forall u) (R(t) \wedge S(u) \wedge t[C] > u[A]) \}$       R中满足下述条件的元组的集合：

C属性大于S中每个元组A属性的值。



任何一个关系代数表达式都可等价地表示成元组关系演算表达式。关系代数中的5种基本运算用元组关系演算表示为：

$$R \cup S = \{t \mid R(t) \vee S(t)\}$$

对应于：  $R \cup S = \{t \mid t \in R(t) \vee t \in S(t)\}$

$$R - S = \{t \mid R(t) \wedge \neg S(t)\}$$

对应于：  $R - S = \{t \mid t \in R(t) \wedge \neg t \in S(t) \}$

$$R \times S = \left\{ t^{(r+s)} \mid (\exists u^{(r)})(\exists v^{(s)})(R(u) \wedge S(v) \wedge t[1] = u[1] \wedge \dots \wedge t[r] = u[r] \wedge t[r+1] = v[1] \wedge \dots \wedge t[r+s] = v[s]) \right\}$$

$$\Pi_{i_1, \dots, i_k}(R) = \left\{ t^{(k)} \mid (\exists u)(R(u) \wedge t[1] = u[i_1] \wedge \dots \wedge t[k] = u[i_k]) \right\}$$

$$\sigma_F(R) = \{t \mid R(t) \wedge F'\}, \text{ 其中 } F' \text{ 是 } F \text{ 的等价表示形式}$$



R

A	B	C
1	2	3
4	5	6
7	8	9

S

A	B	C
1	2	3
3	4	6
5	6	9

$$\{ t \mid S(t) \wedge t[A] > 2 \}$$

A	B	C
3	4	6
5	6	9

$$\{ t \mid R(t) \wedge \neg S(t) \}$$

A	B	C
4	5	6
7	8	9

u[B]

R

A	B	C
1	2	3
4	5	6
7	8	9

S

t[C]

A	B	C
1	2	3
3	4	6
5	6	9

$$\{ t \mid ( \exists u ) ( \textcolor{red}{S}(t) \wedge R(u) \wedge t[C] < u[B] ) \}$$

S中满足下述条件的元组的集合：C属性小于R中某个元组B属性的值。

A	B	C
1	2	3
3	4	6



**R**

A	B	C
1	2	3
4	5	6
7	8	9

$t[C]$

$u[A]$

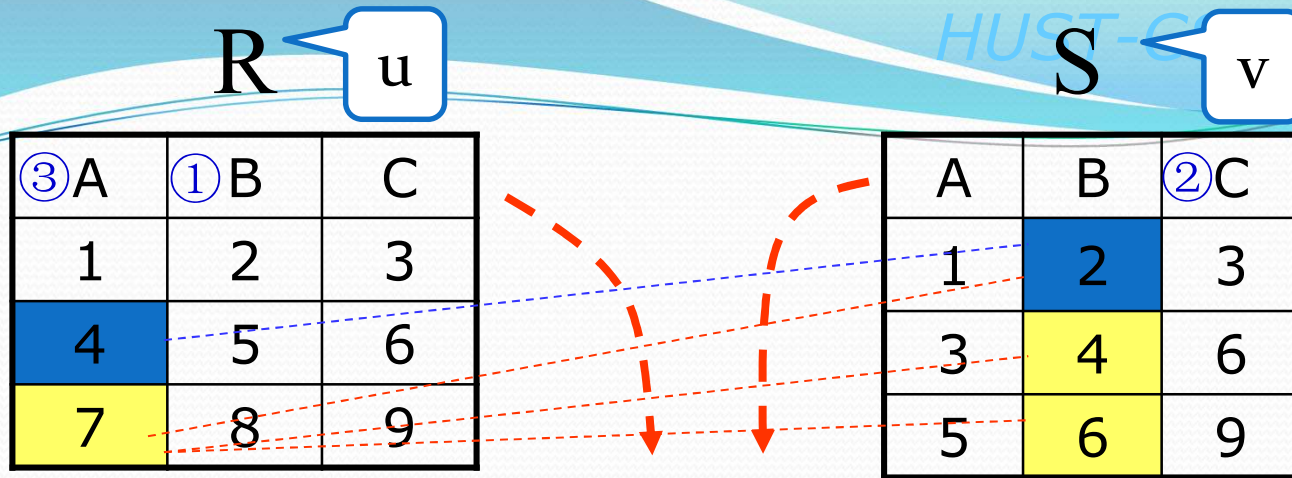
**S**

A	B	C
1	2	3
3	4	6
5	6	9

$$\{ t \mid (\forall u)( R(t) \wedge S(u) \wedge t[C] > u[A] ) \}$$

R中满足下述条件的元组的集合：C属性大于S中每个元组A属性的值。

A	B	C
4	5	6
7	8	9



$$\{ t \mid (\exists u)(\exists v)(R(u) \wedge S(v) \wedge u[A] > v[B])$$

$$\wedge t[A] = u[B] \wedge t[B] = v[C] \wedge t[C] = u[A] \}$$

新关系的A列从R的B列取值，新关系的B列从S的C列取值，新关系的C列从R的A列取值，只需满足条件对应元组的  $R[A] > S[B]$

R.B	S.C	R.A
5	3	4
8	3	7
8	6	7
8	9	7



- 表达式的安全性

- 元组关系演算有可能会产生无限关系，这样的表达式是不安全的

如  $\{t \mid \neg R(t)\}$ ，求所有不在  $R$  中的元组

无穷验证

- 引入公式  $P$  的域概念，用  $dom(P)$  表示

$dom(P)$  至少包含：显式出现在  $P$  中的值、在  $P$  中出现的关系的元组中出现的值（不必是最小集，但是有限集）。

- 如果出现在表达式  $\{t \mid P(t)\}$  结果中的所有值均来自  $dom(P)$ ，则称  $\{t \mid P(t)\}$  是安全的

（教材第2.5节P64页关于表达式安全的说明）

- (1) 如果 $t$ 使 $P(t)$ 为真, 则 $t$ 的每个分量是 $\text{dom}(P)$ 中的元素。
- (2) 对于 $P$ 中每个形如 $(\exists u)(W(u))$ 的子表达式, 若 $u$ 使 $W(u)$ 为真, 则 $u$ 的每个分量是 $\text{dom}(P)$ 中的元素。
- (3) 对于 $P$ 中每个形如 $(\forall u)(W(u))$ 的子表达式, 若 $u$ 使 $W(u)$ 为假, 则 $u$ 的每个分量是 $\text{dom}(P)$ 中的元素。换言之, 若 $u$ 每一分量不属于 $\text{dom}(P)$ , 则 $W(u)$ 为真。



$R$ 

A	B
A1	B1
A1	B2
A2	B3

 $\{ t \mid \neg (t \in R) \}$ 

A	B
A1	B3
A2	B1
A2	B2

$$\text{dom}(\neg (t \in R)) = \{ \{A1, A2\}, \{B1, B2, B3\} \}$$

## 4.2 域关系演算

### ■ 形式化定义

$$\{ x_1 x_2 \dots x_n \mid P (x_1, x_2, \dots, x_n) \}$$

$x_i$ 代表域变量,  $P$ 为由原子构成的公式

### ■ 原子公式

- $\langle x_1, x_2, \dots, x_n \rangle \in R, \text{ 记 } R(x_1, x_2, \dots, x_n)$

- $x_i$ 是域变量或域常量

- $x \theta y$

- 域变量 $x$ 与 $y$ 之间满足比较关系 $\theta$

- $x \theta c$

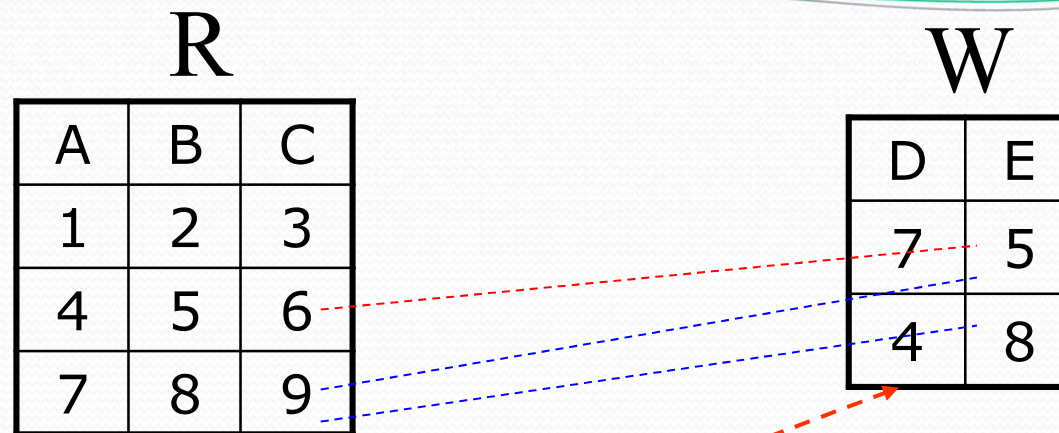
- 域变量 $x$ 与常量 $c$ 之间满足比较关系 $\theta$



R			S			W	
A	B	C	A	B	C	D	E
1	2	3	1	2	3	7	5
4	5	6	3	4	6	4	8
7	8	9	5	6	9		

$R1 = \{ x \ y \ z \mid R(x, y, z) \wedge x < 5 \wedge y > 3 \}$   
R中A列小于5并且B列大于3的元组的集合，  
列与R列对应不变。

A	B	C
4	5	6



$$R2 = \{ x \ y \ z \mid (\exists u) (\exists v) (R(z, x, u) \wedge W(y, v) \wedge u > v) \}$$

1、3列对应R的B、A列，2列对应W的D列，并且对应元组的  
C>E。

B	D	A
5	7	4
8	7	7
8	4	7



R			S		
A	B	C	A	B	C
1	2	3	1	2	3
4	5	6	3	4	6
7	8	9	5	6	9

$$R2 = \{xyz \mid R(x, y, z) \vee (S(x, y, z) \wedge y=4)\}$$

新关系与R列对应，其元组或属于R，或为S中B值为4的元组。

A	B	C
1	2	3
4	5	6
7	8	9
3	4	6

求R中3列小于8并且1列等于d的元组集合，  
域关系演算、关系代数、元组关系演算  
表达式

$$R_1 = \{XYZ | R(XYZ) \wedge Z < 8 \wedge X = d\}$$

等价于关系代数表达式

$$R_1 = \sigma_{A1='d' \wedge A3 < 8}(R)$$

等价于元组关系演算为

$$R_1 = \{t | R(t) \wedge t[1] = d \wedge t[3] < 8\}$$

A1	A2	A3
d	ce	5
d	bd	3
g	ef	7
d	cd	9

A1	A2	A3
d	ce	5
d	bd	3



归纳：

关系代数、元组关系演算和域关系演算都是抽象的查询语言，它们不是具体的**DBMS**中用户使用的语言。

但其数学基础、实现方式适合于数据操纵的逻辑算法描述和实现，是学习和理解SQL语言的基础。

这三种语言在表达能力上是完全等价的，且都是非过程化的。

## 慕课讨论题

- 举例说明关系参照完整性的含义
- 如下自然连接会出现什么问题，如何解决该问题。  
针对如下关系，希望列出所有同学信息及其选修课程情况，若采用自然连接，会出现什么情况，如何解决该问题。

Student:

<u>Sno</u>	Sname	Ssex	Sage	<u>Sdept</u>
200215121	李勇	男	20	CS
200215122	刘晨	女	19	IS
200215123	王敏	女	18	MA
200215125	张立	男	19	IS

SC:

Sno	Cno	Grade
200215121	1	92
200215121	2	85
200215121	3	88
200215122	2	90
200215122	3	80



思考题：回顾第一章列出的关于目前一些数据库的实现机制，里面提到了数据组织、存储的相关策略。

## openGauss架构 VS PostgreSQL架构 关键技术对比

- openGauss是衍生自PostgreSQL-XC，单机逻辑架构与PG接近。
- openGauss和PG在架构和关键技术上有根本性差异，尤其是存储引擎和优化器两大核心能力。

关键差异化因素		openGauss	PostgreSQL
运行时模型	执行模型	<b>线程池模型</b> ，高并发连接切换代价小、内存损耗小，执行效率高，一万并发连接比最优性能损耗<5%。	<b>进程模型</b> ，数据库进程通过共享内存实现通讯和数据共享。每个进程对应一个并发连接，存在切换性能损耗，导致多核扩展性问题。
事务处理	并发控制	<b>64位事务ID</b> ，使用CSN解决动态快照膨胀问题； <b>NUMA-Aware引擎优化改造解决“五把大锁”</b> 。	事务ID回卷，长期运行性能因为ID回收周期大幅波动；存在“五把大锁”的问题，导致事务执行效率和多处理器多核扩展性存在瓶颈。
	日志和检查点	<b>增量Checkpoint机制</b> ，实现性能波动<5%。	<b>全量checkpoint</b> ，性能短期波动>15%。
	<b>鲲鹏NUMA</b>	<b>NUMA改造、cache-line padding、原生spin-lock</b> 。	<b>NUMA多核能力弱，单机两路性能TPMC &lt;60w</b> 。
数据组织	多引擎	<b>行存、列存、内存引擎</b> ，在研DFV存储和原位更新。	仅支持行存。
SQL引擎	优化器	支持SQL Bypass, CBO吸收工行等企业场景优化能力。	支持CBO，复杂场景优化能力一般。
	SQL解析	ANSI/ISO标准SQL92、SQL99和SQL2003和企业扩展包。	ANSI/ISO标准SQL92、SQL99和SQL2003。

思考题：关系数据库的存储方案可分为行存储和列存储两种，行存储会把元组的整行数据保存在一起（一般是一个磁盘块内），列存储会把元组集合的相关性高的列保存在一起，相关性不高的列可能保存在不同的位置。你觉得采用行存储和列存储这两种方案，对于关系代数的执行性能有何影响？

## openGauss架构 VS PostgreSQL架构 关键技术对比

- openGauss是衍生自PostgreSQL-XC，单机逻辑架构与PG接近。
- openGauss和PG在架构和关键技术上有根本性差异，尤其是存储引擎和优化器两大核心能力。

关键差异化因素		openGauss	PostgreSQL
数据组织	多引擎	行存、列存、内存引擎，在研DFV存储和原位更新。	仅支持行存。
SQL引擎	优化器	支持SQL Bypass, CBO吸收工行等企业场景优化能力。	支持CBO，复杂场景优化能力一般。
	SQL解析	ANSI/ISO标准SQL92、SQL99和SQL2003和企业扩展包。	ANSI/ISO标准SQL92、SQL99和SQL2003。