# 1关系代数

## operators

表1 关系代数符号

|  |  |  |
| --- | --- | --- |
| 编号 | 运算符 | 单目/双目 |
| 1 | select: σ | 单目 |
| 2 | project: ∏ | 单目 |
| 3 | union: ∪ | 双目 |
| 4 | set difference: *–* | 双目 |
| 5 | Cartesian product: x | 双目 |
| 6 | rename: ρ | 单目 |
| 7 | Set intersection（集合交运算） | 双目 |
| 8 | Natural join（自然连接） | 双目 |
| 9 | Division （商） | 双目 |
| 10 | Assignment （赋值） | 双目 |
| 11 | Generalized Projection（广义投影） | 单目 |
| 12 | Aggregate Functions（聚集函数） | 单目 |
| 13 | Outer Join（外连接） | 双目 |

## select（选择）

*σp*(***r***) = {*t* | *t* ∈ *r* **and** *p(t)*}

*p* is a formula in propositional calculus consisting of **terms** connected by : ∧ (**and**), ∨ (**or**), ¬ (**not**)

★例子：

1. 选出物理系的所有老师

*σ* *dept\_name=“Physics”*(*instructor*)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

1. 选出物理系或者年薪资大于70000美元的老师；

*σ* *dept\_name=“Physics”* ∨ *salary>70000* (*instructor*)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

1. 选出除了物理和计算机学院之外的老师

*σ* *¬dept\_name= “Physics”* ∧¬ *dept\_name= “Comp. Sci.”* (*instructor*)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

## project （投影）



*A1, A2* are attribute names and *r* is a relation name

★例子：

1. 输出老师的ID,姓名和工资

∏*ID, name, salary* (*instructor*)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

1. 输出计算机系老师的ID,姓名和工资

∏*ID, name, salary* （*σ* *dept\_name=“Comp. Sci.”*(*instructor*)）

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

## Union Operation（集合并）

*r* ∪ *s* = {*t* | *t* ∈ *r* or *t* ∈ *s*}

1. *r,* *s* must have the *same* **arity（同元）** (same number of attributes)
2. The attribute domains must be **compatible（相容的）** (example: 2nd column of *r* deals with the same type of values as does the 2nd column of *s*) 关系r的第二列与s的第二列域一致
3. *r* ∪ *s* 是下面蓝色部分

★例子：

1. 查询出在计算机学院或者物理学院工作的老师的信息

*σ* *dept\_name=“Physics”*(*instructor*) ∪ *σ* *dept\_name=“Physics”*(*instructor*)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

1. to find all courses taught in the Fall 2018 semester, or in the Spring 2019 semester, or in both

∏*course\_id* (*σ* *semester=“Fall” Λ year=2018* (*section*))∪  
 ∏*course\_id* (*σ* *semester=“Spring” Λ year=2019* (*section*))

## Set Difference Operation（集合差）

*r – s* = {*t* | *t* ∈ *r* **and** t ∉ *s*}

* Set differences must be taken between **compatible** relations.

1. r and s must have the same arity（同元）
2. attribute domains of *r* and *s* must be compatible（相融）
3. *r-s*是下面的蓝色部分

★例子：

1. to find all courses taught in the Fall 2018 semester, but not in the Spring 2019 semester

∏*course\_id* (*σ* *semester=“Fall” Λ year=2018* (*section*)) −   
 ∏*course\_id* (*σ* *semester=“Spring” Λ year=2019* (*section*))

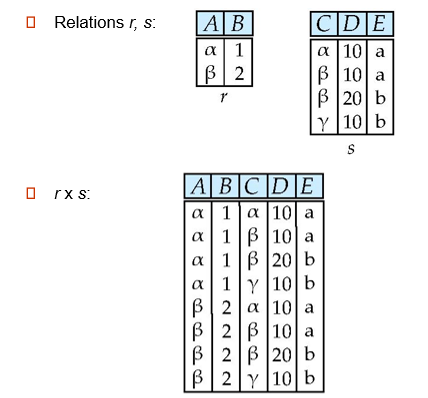
1. 找到所有在物理学院工作但是工资小于70000的所有老师。

*σ* *dept\_name=“Physics”* (*instructor*) –

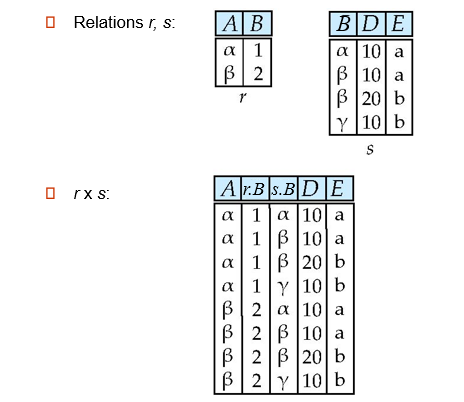
*σ* *salary>=70000* (*instructor*)

## Cartesian-Product Operation（笛卡儿积）

*r* x *s* = {*t q* | *t* ∈ *r* **and** *q* ∈ *s*}



当*r*和*s*具有相同的属性名时，笛卡儿积的结果中用关系名来区别不同的属性



★例子：

1. 选出计算机学院所有学生的选课信息

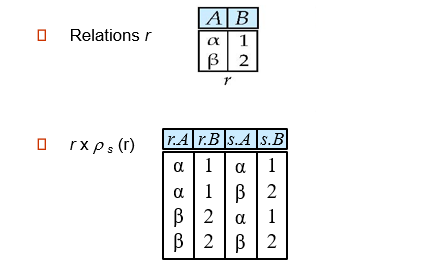
* （1）涉及哪些表？
* student×takes
* （2）如何表示计算机学院的学生？
* σdept\_name=‘计算机’（student×takes）
* （3）删除takes中，非这些学生选过的课程的信息
* σstudent.ID= takes.ID ( σdept\_name=‘计算机’（student×takes)）
* 有没有其他的表示方法？
* σdept\_name=‘计算机’ (σstudent.ID= takes.ID（student×takes)）

σstudent.ID= takes.ID （σdept\_name=‘计算机’（student）× takes）

## 更名

*ρ x* (*E*)

returns the expression *E* under the name *X*



★例子：

1. 查出大学里的最高工资。

∏salary(instructor)- ∏ instructor.salary(*σ* instructor.salary<d.salary(instructor× *ρd*(instructor)))

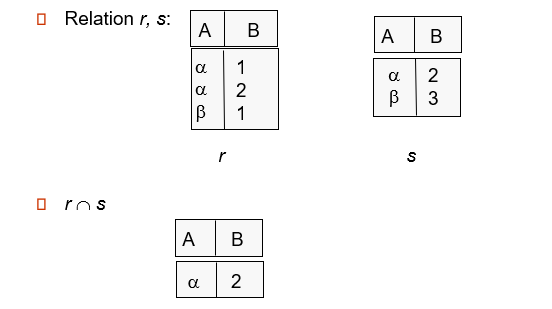
## Set-Intersection Operation（集合交）

*r* ∩ *s* = { *t* | *t* ∈ *r* **and** *t* ∈ *s* }

Assume:

* + *r*, *s* have the *same arity*
  + attributes of *r* and *s* are compatible

Note: *r* ∩ *s* = *r* – (*r* – *s*)



★例子：

1. to find all courses taught in the Fall 2018 semester, and in the Spring 2019 semester

∏*course\_id* (*σ* *semester=“Fall” Λ year=2018* (*section*)) ∩   
 ∏*course\_id* (*σ* *semester=“Spring” Λ year=2019* (*section*))

1. 找到计算机系且工资大于70000的所有老师的工号。

∏ *ID* (*σ* dept\_name=’Comp. Sci.’(instructor))∩∏ *ID* (*σ* instructor.salary>70000(instructor))

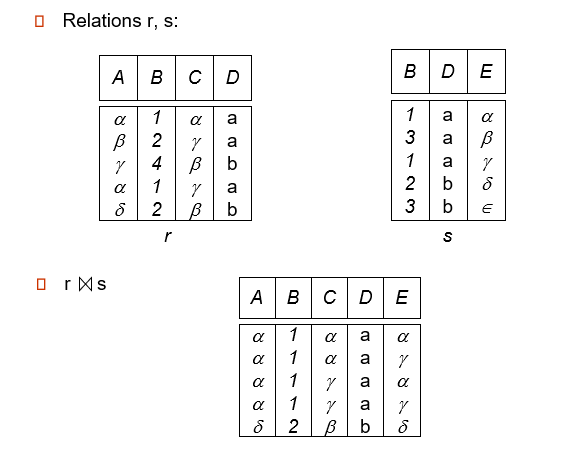
|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPR\_NAME | SALARY |
| 76766 | Crick | Bilogy | 72000 |
| 45565 | Katz | Comp.Sci. | 75000 |
| 12121 | Wu | Finance | 80000 |
| 98345 | Kim | Physics | 68200 |

## Natural-Join Operation(自然连接)

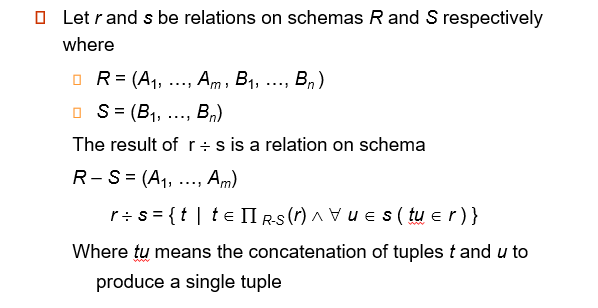
*R* = (*A, B, C, D*) *S* = (*E, B, D*)

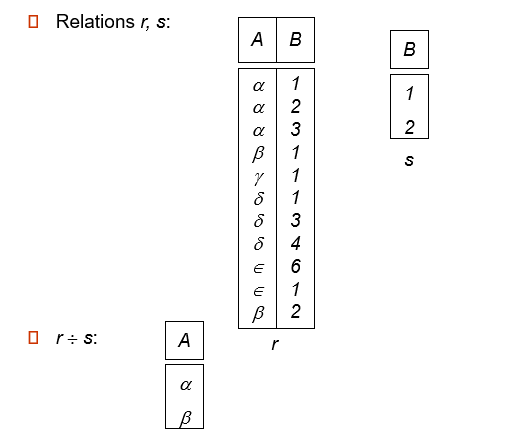
Result schema = (*A, B, C, D, E*)

* + *r* *s* is defined as:  
     ∏*r.A, r.B, r.C, r.D, s.E* (σ*r.B = s.B* ∧ *r.D = s.D* (*r*  x *s*))



## Division Operation（商）



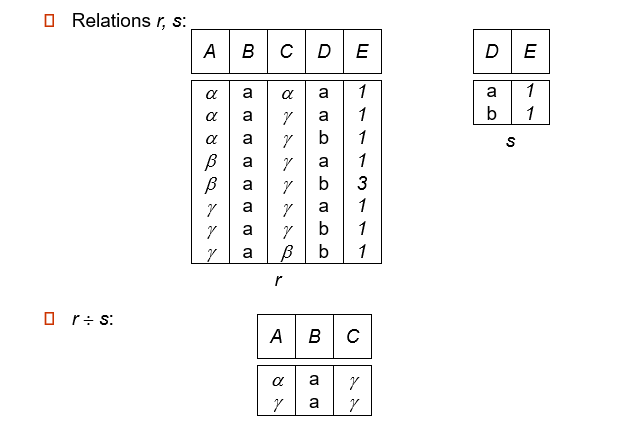


思路：

1 、结果应该只包含属性A;

2、可能包含的结果有α、β、γ、δ、ε。

3、候选结果分别于B中的1、2连接，然后判断其是否都存在在r中？如果都存在，该候选结果保留，否则删掉。可知，只有α、β满足条件，所以保留。



★例子：

1. 选出选了计算机系所有课程的学生的ID

∏*course\_id，ID* (takes)÷(∏*course\_id* (σ*department=“计算机”* (course)))

## Assignment Operation（赋值操作）

The assignment operation (←) provides a convenient way to express complex queries.

## Generalized Projection（广义投影）



* *E* is any relational-algebra expression
* Each of *F*1, *F*2, …, *Fn* are are arithmetic expressions involving constants and attributes in the schema of *E*.

★例子：

1. 每个学生当前还差多少学分可以毕业。假定最低学分要求是100学分。

∏*ID, 100–tot\_credit (student)*

## Aggregate Functions（聚集函数）

**avg**: average value

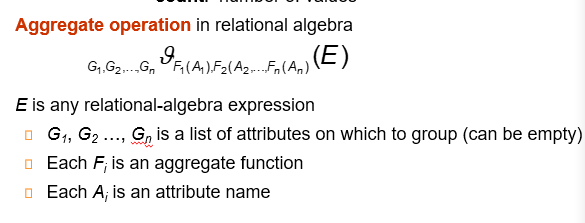
**min**: minimum value

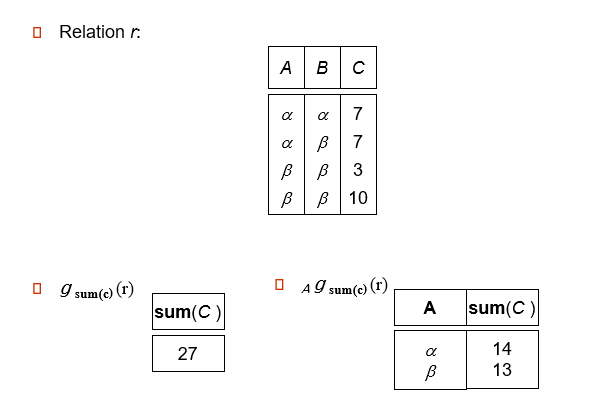
**max**: maximum value

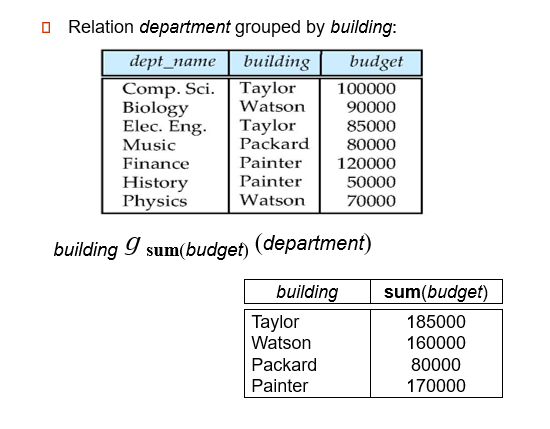
按什么属性聚集

**sum**: sum of values

**count**: number of values



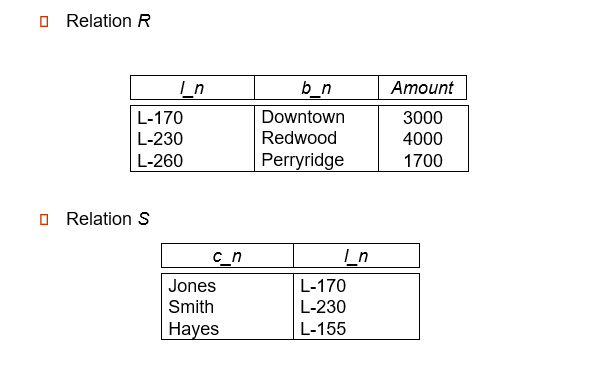


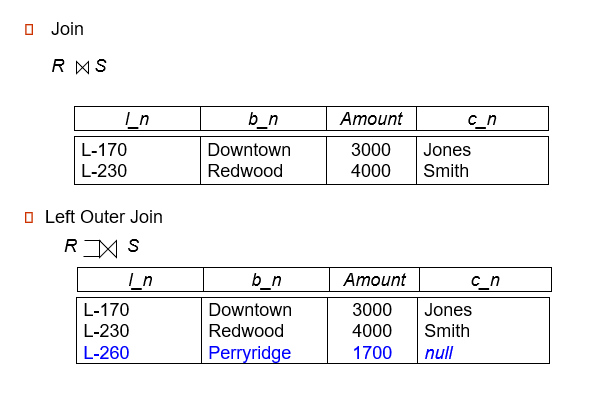


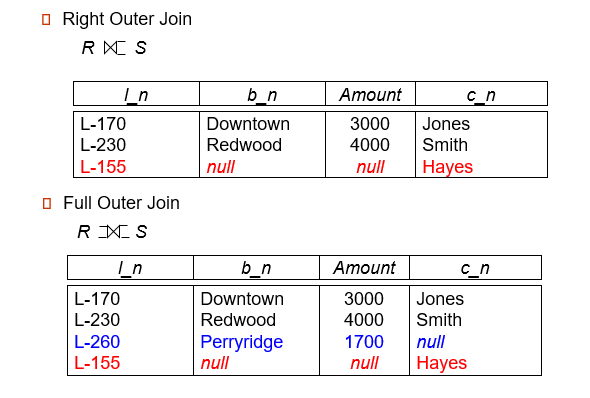
聚集结果是没有名字的，我们可以给上面的聚集结果起名字。



## Outer Join（外连接）







## Modification of the Database

* 1. Deletion
  2. Insertion

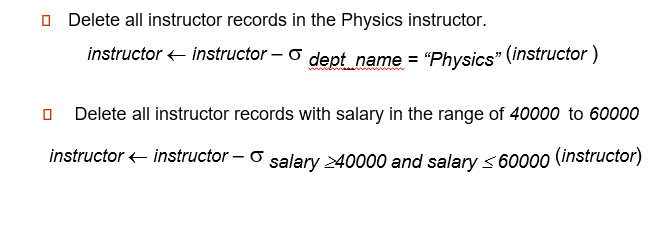
Deletion

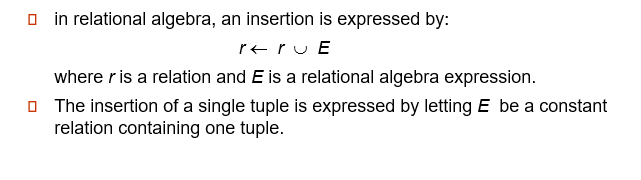
* 1. Updating

A deletion is expressed in relational algebra by:

*r* ← *r* – *E*

where *r* is a relation and *E* is a relational algebra query.





Insert

