数据库系统原理

引用中国人民大学信息学院原版PPT

华中科技大学计算机学院左琼修改版

School of Computer Science and Technology， HUST

2020

2.4 关系代数运算小结

- 5 种基本运算（并∪、差-、笛卡尔积×、选择σ、投影π）
- 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：
  - $R \cap S = R - (R - S) = S - (S - R)$
  - $R \bowtie S = \pi_{i1,\ldots,im}(\sigma_{R.A1=S.A1 \wedge R.A2=S.A2 \ldots \wedge R.Ak=S.Ak}(R \times S))$
  - $R \div S = \pi_{1,2,..r-s}(R) - \pi_{1,2,..r-s}((\pi_{1,2,..r-s}(R) \times S) - R)$
- 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。利用这些表达式可以实现对关系数据库的各种操作（插入、删除、修改、查询）。

2.4.3 附加运算*（了解*）

- **问题**　关系代数的基本运算足以表达任何查询，但使用不方便，写出的表达式太长。定义一些附加运算，不能增加关系代数功能，但能简化表达式。
- **更名运算**

  $\rho_{x(A1,A2,.....An)}(E)$

  将E的结果取名字x，且将属性名改为A1,...An
- **为什么需要更名运算?**

  一个关系代数表达式的**结果关系**没有名字；

  一个关系表可以多次参与到一个联系中，每次**参与角色不同**，如何区分?

更名运算示例*

- 示例　只用基本关系运算符，找出学生中最大年龄?
- 求解　利用集合的补集运算，求出所有非最大年龄集合的补集即可，反证思维

1）求出一个由非最大年龄构成的表

2）求所有学生年龄与上一步结果的差

如何求非最大年龄集合?

令所有学生集合为A，所有年龄集合为B，显然，A与B都在Student表中；

对于A中每个学生t，将其与B中所有年龄元组配对，判断：该学生年龄是否小于所有学生年龄集中的某一个。若是，则将其放在结果集中。

更名运算示例*

分析知， Student表两次以不同角色参与运算，为区别起见，令代表年龄的学生表为d。

S1： 首先将学生表与d表合并

$student \times \rho_d(student)$

S2： 在合并表中选择哪些年龄比所有年龄中某一个小的学生

$\pi_{student.age}(\sigma_{student.age<d.age}(student \times \rho_d(student)))$

S3： 求差（求问题的反面）

$\pi_{age}(student) - \pi_{student.age}(\sigma_{student.age<d.age}(student \times \rho_d(student)))$

注： 上面S1、S2步中，也可以用条件连接的选择功能，直接从学生表中将哪些年龄不是最大的学生选择出

$\rho_s(student) \underset{s.age<d.age}{\bowtie} \rho_d(student)$

思考题： 求出与“张三” 在同一个系的学生?

聚集运算*

- 问题示例　选课表每个学生会有多个选课记录，若要查询**所有**学生所选课程的平均分如何办?
- 解决办法　1）将选课表按照学号**分组**
  2）对分组后的表中**每个组**再调用AVG（）函数

SNO	CNO　GRADE
200215121	(1,　92) (2,　85) (3,　88)
200215122	(2,　90) (3,　80)
……	……

聚集运算*

- 聚集运算　将表按照属性分组，即将元组按照属性值重新组合
- 定义　$_{A1,...An}G(E)$
  - E是关系代数表达式，即是一张表
  - $A_1,...A_n$是用于分组的一组属性
  - 运算符G表示将表达式E按照$A_1,...A_n$分组
  - 同一组中所有元组在$A_1,...A_n$上值相同
  - 不同组元组在$A_1,...A_n$上值不同
- 示例　$_{sno}G(sc)$结果为上一页中表
  - $_{ssex}G(student \bowtie sc)$的结果为多少?
- 问题　聚集运算的结果是由多个分组组成的表，对每个分组可以定义聚集函数

聚集函数*

- 聚集函数　输入为集合，输出为单一值的函数。
  sum()，avg()，max(), min()，count()
- 应用　聚集函数往往和聚集运算组合使用
  对于聚集运算后的结果，对每个组再运用聚集函数处理

- 示例　表达式$_{sno}G_{count\ (cno)\ ,min(grade)}(sc)$的结果
  为每个学生所选课的个数和最低分
- 如何去除相同元素?
  distinct操作符，其含义是消除相同元素

e.g　$_{sno}G_{count\ (distinct\ cno)\ ,min(grade)}(sc)$

广义投影*

问题示例 学生表中只有年龄，若查询学生的出生年份如何办?

问题分析 出生年份可以通过年龄计算求得，因此，**扩展**投影运算，使投影属性可以是**派生属性**，即可以从表中经过运算得出。

广义投影 $\pi_{F1,...Fn}(E)$

F1,...Fn中是可以涉及常量、系统函数及E中属性的算数表达式

示例 $\pi_{sno,year()-age}$(student)结果为由(sno，year()-age）两列组成的表；

$\rho_{sno,\ birthday}(\pi_{sno,year()-age}$(student))结果为(sno, birthday)两列组成的表

数据库修改*

- 定义 对数据库的增、删、查的运算符，通过赋值完成
- 删除 r←r - E 其中：E是关系代数查询表达式

例如 sc ←sc - $\sigma_{sno=2}$ (sc)

从选课表中删除了2号学生的选课记录

- 插入 r←r ∪E
- 更新 r←$\pi_{F1,...Fn}$(r),使用广义投影可以改变元组中的值r←$\pi_{F1,...Fn}$(σ_P (r))∪(r-σ_P (r)),对r中的部分元组修改
- 注 上述定义的所有运算符来自于应用语义需求，每个运算符在后面的SQL语言中都有相应语句

视图操作*

视图操作　能够从已有的表集合中生成一个虚关系表的运算

出于安全性或出于方便性考虑，需要视图操作

定义　creat view *v* as E,将E的结果作为视图*v*

示例　creat view *cs-student* as ($\sigma_{sdept='cs'}$ (sc))

注意　视图定义运算不同于关系赋值运算。

执行赋值运算时，结果关系被计算并存储，

执行视图定义运算时，结果关系未被计算，

直到某个查询使用视图时才动态计算视图表

课堂练习

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

C (Cno, Cname, Cpno, Ccredit)

SC (sno, cno, grade)

1. 查询学习课程号为2的学生学号和成绩。

$\pi_{sno,grade}(\sigma_{cno='2'}(SC))$

2. 查询学习课程号为2的学生学号和姓名。

$\pi_{s.sno,sname}(\sigma_{cno='2'}(S \bowtie SC))$

3. 查询选修了“数据库”课程的学生学号和姓名。

$\pi_{s.sno,sname}(\sigma_{cname='数据库'}(S \bowtie SC \bowtie C))$

4. 查询至少选修了2号课程和4号课程的学生学号。

$\pi_{sno}(\sigma_{[1]=[4] \wedge [2]='2' \wedge [5]='4'}(SC \times SC))$

或：先建立一个临时关系K(Cno)，$\pi_{Sno,Cno}(SC) \div K$

Cno
2
4

课堂练习

S (Sno, Sname, Ssex, Sage, Sdept)
C (Cno, Cname, Cpno, Ccredit)
SC (sno, cno, grade)

5. 查询不学2号课程的学生姓名、年龄。

$\pi_{sname,sage}(S) - \pi_{sname,sage}(\sigma_{cno='2'}(S \bowtie SC))$

6. 查询选修了全部课程的学生学号和姓名。

$\pi_{Sno,Cno}(SC) \div \pi_{Cno}(C) \bowtie \pi_{Sno,Sname}(S)$

7. 查询至少选修了学号为95002的学生所修全部课程的学生学号。

$\pi_{Sno,Cno}(SC) \div \pi_{Cno}(\sigma_{Sno='95002'}(SC))$

关系代数课堂测试题

考虑下面关系数据库 员工 employee (ename, street, city)
工作 works (ename, cname, salary)
公司 company (cname, city)
经理 manages (pname, manager-name)

用关系代数表达式表示下面查询?

1）找出“first bank corporation”公司的所有员工姓名。
2）找出“first bank corporation”的所有员工姓名和居住城市。
3）找出“first bank corporation”的所有收入在10000元以上员工的姓名和居住城市。
4）找出所有居住地与工作的公司在同一个城市的员工姓名。
5）找出与其经理居住在同一个城市、同一街道的所有员工姓名。
6）找出不在“first bank corporation”工作的所有员工姓名。
7）找出比“small bank corporation”公司的所有员工收入都高的所有员工姓名。
8）假设公司可以在几个城市。找出位于“small bank corporation”所在的各个城市的所有公司。

第二章 关系数据库

2.5 关系演算（掌握与关系代数的差异*）

- 关系演算：以数理逻辑中的谓词演算为基础。
- 按谓词变元不同进行分类：

1.元组关系演算：

以元组变量作为谓词变元的基本对象

元组关系演算语言ALPHA

2.域关系演算：

以域变量作为谓词变元的基本对象

域关系演算语言QBE

2.5.1 元组关系演算*

- **用元组作为谓词变量的一种谓词演算方法。元组关系演算表达式的一般形式为：**

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

表示所有使 P(t) 为真的元组集合。

其中：

- **t —— 元组变量。表示一个元组，若 t 中有多个分量，表示为t[1]，t[2]，……；**
- **P(t) —— 由原子公式和运算符组成的复合公式。**

2.5.1 元组关系演算

❖原子公式有下列三种形式：

①R(s)　R是关系名，s是元组变量。含义：s是关系R的一个元组。

②s[i]θu[j]　s和u是元组变量，θ是算术比较运算符。含义：元组s的第i 个分量与元组u的第j个分量之间满足θ关系。

③s[i]θa　元组s的第i 个分量值与常量a之间满足θ关系。

❖运算符包括四类：

① 括号（）

② 算术运算符：＞　≥　＝　＜　≤　≠

③ 存在量词∃，全称量词∀

④ 逻辑运算符：¬　∧　∨

2.5.1 元组关系演算

公式的递归定义如下：

① 原子公式 P 是一个公式。其值为 P 的真、假值。

② 如果P_1和P_2是公式，那么$\neg P_1$、$P_1 \wedge P_2$、$P_1 \vee P_2$也是公式。其真假值遵循逻辑运算的一般原则。

③ 如果P是公式，那么 $(\exists t)P(t)$ 也是公式。设元组变量的域集$T = \{t_1, t_2, \ldots, t_n\}$，$(\exists t)P(t) \Leftrightarrow P(t_1) \vee P(t_2) \vee \ldots \vee P(t_n)$，至少存在一个元组 t_i 使得公式 P 为真，否则为假。

④ 如果P是公式，那么 $(\forall t)P(t)$ 也是公式。设元组变量的域集$T = \{t_1, t_2, \ldots, t_n\}$，$(\forall t)P(t) \Leftrightarrow P(t_1) \wedge P(t_2) \wedge \ldots \wedge P(t_n)$，所有元组 t_i 使得公式P为真时上式为真，否则为假。

t 在P中是自由变量，在 $(\exists t)P(t)$和$(\forall t)P(t)$中是约束变量，即：

自由元组变量——在一个公式中 t 未用∃ 、∀符号定义；

约束元组变量——在一个公式中 t 用∃ 、∀符号定义；

2.5.1 元组关系演算

- 例：设有两个关系 R 和 S，求表达式的值：

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

1）$R_1 = \{ t \mid S(t) \wedge t[1]>2 \}$

R_1

A	B	C
3	4	6
5	6	9

2）$R_2 = \{ t \mid R(t) \wedge \neg S(t) \}$

R_2

A	B	C
4	5	6
7	8	9

3）$R_3=\{ t \mid (\exists u)(S(t) \wedge R(u) \wedge t[3]< u[1]) \}$

R_3

A	B	C
1	2	3
3	4	6

2.5.1 元组关系演算

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

4）$R_4=\{ t \mid (\forall u)(R(t) \wedge S(u) \wedge t[3]>u[1])\}$

R_4

A	B	C
4	5	6
7	8	9

5）$R_5=\{ t \mid (\exists u)(\exists v)(R(u) \wedge S(v) \wedge u[1]>v[2] \wedge t[1]=u[2] \wedge t[2]=v[3] \wedge t[3]=u[1])\}$

R_5

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

应用

S (Sno, Sname, Ssex, Sage, Sdept)
C (Cno, Cname, Cpno, Ccredit)
SC (sno, cno, grade)

例1：查询CS系的学生信息

$\{ t \mid S(t) \wedge t[5]='CS' \}$

例2：查询学习课程号为2的学生学号。

$\{ t \mid (\exists u)(SC(u) \wedge u[2]='2' \wedge t[1]=u[1] \}$

例3：查询选修了“数据库”课程的学生学号。

$\{ t \mid (\exists u)(\exists v)(SC(u) \wedge C(v) \wedge v[1]=u[2] \wedge v[2]='数据库' \wedge t[1]=u[1]) \}$

例4：查询选修了全部课程的学生学号。

$\{ t \mid (\forall u)(C(u) \wedge (\exists v)(SC(v) \wedge v[2]=u[1] \wedge t[1]=v[1])) \}$

2.5.2 域关系演算*

- 域关系演算类似于元组关系演算，不同之处是用域变量代替元组变量的每一个分量，域变量的变化范围是某个值域而不是一个关系。域演算表达式形为：

$$\{ t_1 \ldots t_k \mid P(t_1,\ldots,t_k) \}$$

其中$P(t_1,\ldots,t_k)$是关于自由域变量$t_1,\ldots,t_k$的公式。

- 域关系演算的公式中也可使用∧、∨、¬等逻辑运算符，也可用 (∃ x) 和 (∀x) 形成新的公式，但变量 x 是域变量，不是元组变量。
- 域演算的原子公式：

① R(x1…xk)：R是一个k元关系，x_i是常量或域变量。含义：由$x_1,\ldots,x_k$组成的元组在关系R中。

② xθy：x，y是常量或域变量，但至少有一个是域变量，θ是算术比较符。含义：x 和 y 之间满足关系θ。

例：设有R、S和W三个关系，求表达式的值：

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

W

D	E
7	5
4	8

1）$R_1=\{xyz \mid R(xyz) \wedge x<5 \wedge y>3\}$

R_1

A	B	C
4	5	6

2）$R_2=\{xyz \mid R(xyz) \vee (S(xyz) \wedge y=4)\}$

R_2

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

3）$R_3=\{xyz \mid (\exists u)(\exists v)(R(zxu) \wedge W(yv) \wedge u>v)\}$

R_3

A	B	C
5	7	4
8	7	7
8	4	7

应用示例

S (Sno, Sname, Ssex, Sage, Sdept)
C (Cno, Cname, Cpno, Ccredit)
SC (sno, cno, grade)

例1：查询计算机系(IS)的全体学生。

$\{ abcde \mid S(abcde) \wedge e = 'CS' \}$

例2：查询年龄小于20岁的学生。

$\{ abcde \mid S(abcde) \wedge d < 20 \}$

例3：检索选修课程号为5的学生学号和姓名。

$\{ ab \mid (\exists u)(\exists v)(S(abcde) \wedge SC(uvw) \wedge a=u \wedge v='5') \}$

关系运算的安全性(★)

- 关系演算有可能会产生无限关系和无穷验证，这样的表达式是不安全的。例如：{ t | ¬R(t) }，(∀u)(ω(u))。
- 不产生无限关系和无穷验证的运算称为安全运算。其运算表达式称为安全表达式，所采取的措施称为安全限制。
- 在关系演算中，引入公式P的域概念，用DOM(P)表示。

  DOM(P) = 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）
- 满足下列条件时，称元组演算表达式{ t | P(t) }是安全的：
  - 出现在表达式{ t | P(t) }结果中的所有值均来自DOM(P)；
  - 对P中的每个形如(∃u)(ω(u)) 的子式，若u使ω(u) 为真，则 u 的每个分量必属于DOM(P)。
  - 对P中的每个形如(∀u)(ω(u)) 的子式，若u使ω(u) 为假，则 u 的每个分量必属于DOM(P)。

关系运算的安全性(★)

R

A	B
a1	b1
a2	b2

S

A	B
1	d
5	b
6	c
7	d

$R_1=\{ t \mid \neg R(t) \}$

DOM(P)={{a1 , a2} , {b1 , b2}}

R_1

A	B
a2	b1
a1	b2

$R_2=\{ t \mid (\exists u)(S(u) \wedge u[1]>3 \wedge t[1]=u[2] \}$

DOM(P)={{1,5,6,7,3} , {d,b,c}}

R_2

B
b
c
d

元组演算对基本关系操作的表示

并： $R \cup S \equiv \{ t \mid R(t) \vee S(t) \}$

差： $R - S \equiv \{ t \mid R(t) \wedge \neg S(t) \}$

笛卡儿积： $R \times S \equiv \{ t(r+s) \mid (\exists u)(\exists v)(R(u) \wedge S(v) \wedge t[1]=u[1] \wedge \ldots \wedge t[r]=u[r] \wedge t[r+1]=v[1] \wedge \ldots \wedge t[r+s]=v[s]) \}$

投影： $\pi_{i1,\ldots,im}(R) \equiv \{t(m) \mid (\exists u)R(u) \wedge t[1]=u[i_1] \wedge t[2]=u[i_2] \wedge \ldots \wedge t[m]=u[i_m]\}$

选择： $\sigma_F(R) \equiv \{ t \mid R(t) \wedge F' \}$ （F'是由F变化形成的谓词公式）

域演算对基本关系操作的表示

并： $R \cup S \equiv \{ x_1x_2\ldots x_n \mid R(x_1x_2\ldots x_n) \vee S(x_1x_2\ldots x_n) \}$

差： $R - S \equiv \{ x_1x_2\ldots x_n \mid R(x_1x_2\ldots x_n) \wedge \neg S(x_1x_2\ldots x_n) \}$

笛卡儿积： $R \times S \equiv \{ x_1x_2\ldots x_ny_1y_2\ldots y_m \mid R(x_1x_2\ldots x_n) \wedge S(y_1y_2\ldots y_m) \}$

投影： $\pi_{i1,\ldots,im}(R) \equiv \{ x_{i1}x_{i2}\ldots x_{im} \mid R(x_1x_2\ldots x_n) \}$

选择： $\sigma_F(R) \equiv \{ x_1x_2\ldots x_n \mid R(x_1x_2\ldots x_n) \wedge F' \}$

（F'是由F变化而形成的谓词公式）

2.6 小结

作业 P70 5，6（仅用关系代数完成）

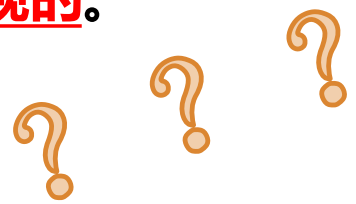
- 关系数据库系统与非关系数据库系统的区别：
  - 关系系统只有“表”这一种数据结构；
  - 非关系数据库系统还有其他数据结构，以及对这些数据结构的操作
- 关系模型概述
  - 关系数据结构及定义（关系，候选码，主码，外码，关系模式，关系数据库）。
  - 关系的完整性约束（实体完整性，参照完整性，用户定义的完整性）。
  - 关系代数（5个基本运算（并，差，笛卡儿积，选择，投影）以及交，连接，自然连接，除）
  - 关系演算*（元组关系演算，域关系演算，关系运算安全性DOM）

第一章作业问题

- 3.试述文件系统与数据库系统的区别与联系。

答案：

- 区别：从以下方面来区别：数据独立性、整体结构化、共享性高、冗余度低、数据控制能力（安全性、完整性、并发控制 和 恢复）
- 联系：
  - （1） 均为数据组织的管理技术；
  - （2） 均由数据管理软件管理数据，程序与数据之间用存取方法进行转换；
  - （3） 数据库系统是在文件系统的基础上发展而来的。
- 作业中的一种答案：
- 联系：文件系统和数据库系统均可以长期保存数据，都是计算机系统中管理数据库的软件。而数据库系统的组织和存储是通过操作系统中的文件系统来实现的。



第一章作业问题

一种答案：
应用程序对数据的访问需要通过DBMS，并不直接访问数据，所以数据和程序是独立的。

- 17. 什么叫数据与程序的物理独立性？什么叫数据与程序的逻辑独立性？为什么数据库系统具有数据与程序的独立性？

一种答案：
物理独立性：用户的应用程序与数据库中的数据的物理存储是相互独立的，数据的物理存储改变，程序不用改变
逻辑独立性：用户的应用程序与数据库的逻辑结构是相互独立的，数据的逻辑结构改变，程序不用改变

DB存储结构改变了，由DBA对模式／内模式映像做相应改变，可以使模式保持不变，从而应用程序也不必改变

有歧义

模式改变（增加新关系，新属性，修改属性类型）时，DBA对各个外模式／模式的映像做相应改变，可以使外模式保持不变，应用程序是依据数据的外模式编写的，从而应用程序不必修改。

2.4 关系代数运算小结

- 5 种基本运算（并∪、差-、笛卡尔积×、选择σ、投影π）
- 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：
  - $R \cap S = R - (R - S) = S - (S - R)$
  - $R \bowtie S = \pi_{i1,\ldots,im}(\sigma_{R.A1=S.A1 \wedge R.A2=S.A2 \ldots \wedge R.Ak=S.Ak}(R \times S))$
  - $R \div S = \pi_{1,2,..r-s}(R) - \pi_{1,2,..r-s}((\pi_{1,2,..r-s}(R) \times S) - R)$
- 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。利用这些表达式可以实现对关系数据库的各种操作（插入、删除、修改、查询）。

关系代数 vs. 关系演算

元组演算对基本关系操作的表示

并：$R \cup S \equiv \{ t \mid R(t) \vee S(t) \}$

差：$R - S \equiv \{ t \mid R(t) \wedge \neg S(t) \}$

$(\exists t)P(t)$
$(\forall t)P(t)$

域演算对基本关系操作的表示

并：$R \cup S \equiv \{ x_1 x_2 \ldots x_n \mid R(x_1 x_2 \ldots x_n) \vee S(x_1 x_2 \ldots x_n) \}$

差：$R - S \equiv \{ x_1 x_2 \ldots x_n \mid R(x_1 x_2 \ldots x_n) \wedge \neg S(x_1 x_2 \ldots x_n) \}$

……

$(\exists u)(\exists v)(R(zxu) \wedge W(yv) \wedge u>v$

课堂练习：

- 设R和S同为相容的k元关系，R有m个元组，S有n个元组，则关于R∩S，以下论述错误的是（　）

  - A. 等于R-(R-S)
  - B. 等于S-(S-R)
  - C. 最多有m个元组
  - D. 最少有0个元组

除

$\Pi_{姓名，课程}(R)\div S$

R

姓名	课程	成绩
张军	物理	88
王红	数学	80
张军	数学	90

÷

S

课程
数学
物理

X	Y

Y	Z

$\Pi_y(S)\subseteq Y_x$

$\Pi_y(S)$={课程}， Y_x = {课程}，
X={姓名，成绩}

R÷S = ?

{张军，88}={物理}，

{王红，80}={数学}，

{张军，90}={数学}

没有哪个X的象集包含了{物理，数学}，所以答案应该是空集！

关系代数随堂练习

考虑下面关系数据库 员工 employee (ename, street, city)
工作 works (ename, cname, salary)
公司 company (cname, city)
经理 manages (pname, manager-name)

用关系代数表达式表示下面查询?

1）找出“first bank corporation”公司的所有员工姓名。
2）找出“first bank corporation”的所有员工姓名和居住城市。
3）找出“first bank corporation”的所有收入在10000元以上员工的姓名和居住城市。
4）找出所有居住地与工作的公司在同一个城市的员工姓名。
5）找出与其经理居住在同一个城市、同一街道的所有员工姓名。
6）找出不在“first bank corporation”工作的所有员工姓名。
7）找出比“small bank corporation”公司的所有员工收入都高的所有员工姓名。
8）假设公司可以在几个城市部署分部。找出在“small bank corporation”公司所在的各个城市都有
分部的公司。
9）假设公司可以在几个城市。找出位于“small bank corporation”所在的各个城市的所有公司。

关系代数随堂练习

- 5）找出与其经理居住在同一个城市、同一街道的所有员工姓名。

$$\pi_{emp.ename}(\sigma_{cname=pname \wedge emp.city=b.city}(employee \bowtie works \bowtie \pi_{pname,b.city}(\sigma_{ename = manager\_name}(manages \bowtie \rho_b(employee)))))$$

- 6）找出不在“first bank corporation”工作的所有员工姓名。

$$\pi_{ename}(\sigma_{cname<>'FSB'}(works))$$

或

$$\pi_{ename}(works) - \pi_{ename}(\sigma_{cname='FSB'}(works))$$

关系代数随堂练习

- 7）找出比“small bank corporation”公司的所有员工收入都高的所有员工姓名。

$$\pi_{ename}\,(\sigma_{salary} > salary\,G_{max}\,(\sigma_{cname='SSB'}(works))\,(works))$$

关系代数随堂练习

8）假设公司可以在几个城市有分部。找出在“small bank corporation”公司所在的各个城市都有分部的公司。

$$company \div \pi_{city}(\sigma_{cname='SBC'}(company))$$

9）假设公司可以在几个城市。找出位于“small bank corporation”所在的各个城市的所有公司。

$$\pi_{cname}(company \bowtie \pi_{city}(\sigma_{cname='SBC'}(company)))$$