

Chapter 2

The Relational Model

(2021.8.30)

Ch2 The Relational Model

□ Data Model (数据模型)

1. is a set of definitions describing how real-world data is conceptually represented as computerized information.
2. It also describes the types of operations available to access and update this information.

□ Relational Model (关系模型)

□ Object-Relational Model (对象关系模型)

Ch2 The Relational Model

- ▶ **2.1 The CAP Database**
- ▶ **2.2 Naming the Parts of a Database**
- ▶ **2.3 Relational Rules**
- ▶ **2.4 Keys, Superkeys, and Null Values**
- ▶ **2.5 Relational Algebra**
- ▶ **2.6 Set-Theoretic Operations**
- ▶ **2.7 Native Relational Operations**
- ▶ **2.8 The Interdependence of Operations**
- ▶ **2.9 Illustrative Examples**
- ▶ **2.10 Other Relational Operations**

2.1 The CAP Database

□ **Example:** Figure 2.1 & 2.2

⌘ **CAP Database:** look at pg. 28, Figure 2.2

⌘ **Data Model:** look at pg.27, Figure 2.1

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

Example: Figure 2.1 & 2.2 (cont.)

AGENTS

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5

Example: Figure 2.1 & 2.2 (cont.)

PRODUCTS

<u>pid</u>	pname	city	quantity	price
P01	comb	Dallas	111400	0.50
p02	brush	Newark	203000	0.50
p03	razor	Duluth	150600	1.00
p04	pen	Duluth	125300	1.00
p05	pencil	Dallas	221400	1.00
p06	folder	Dallas	123100	2.00
p07	case	Newark	100500	1.00

ORDERS

<u>ordno</u>	month	cid	aid	pid	qty	dollars
1011	jan	c001	a01	p01	1000	450.00
1012	jan	c001	a01	p01	1000	450.00
1019	feb	c001	a02	p02	400	180.00
1017	feb	c001	a06	p03	600	540.00
1018	feb	c001	a03	p04	600	540.00
1023	mar	c001	a04	p05	500	450.00
1022	mar	c001	a05	p06	400	720.00
1025	apr	c001	a05	p07	800	720.00
1013	jan	c002	a03	p03	1000	880.00
1026	may	c002	a05	p03	800	704.00
1015	jan	c003	a03	p05	1200	1104.00
1014	jan	c003	a03	p05	1200	1104.00
1021	feb	c004	a06	p01	1000	460.00
1016	jan	c006	a01	p01	1000	500.00
1020	feb	c006	a03	p07	600	600.00
1024	mar	c006	a06	p01	800	400.00

2.1 The CAP Database

□ Note

⌘ More columns

- id, name
- discnt, percent, qty, month, qty
- city
- dollars

⌘ More rows

- no two rows in a table is

⌘ More tables

- unique identifier

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	
a05	Otasi	Duluth	
a06	Smith	Dallas	

<u>pid</u>	pname	city	quantity	price
P01	comb	Dallas	111400	0.50
p02	brush	Newark	203000	0.50
p03	razor	Duluth	150600	1.00
p04	pen	Duluth	125300	1.00
p05	pencil	Dallas	221400	1.00
p06	folder	Dallas	123100	2.00
p07	case	Newark	100500	1.00

2.2 Naming the Parts of a Database

□ Terminology (术语)

✧ Table (Relation)

- Old: file of records

✧ Column names (Attributes)

- Old: field names of records

✧ Rows (Tuples)

- Old: records of a file

✧ Table heading (Schema)

- Old: set of attributes

2.2 Naming the Parts of a Database

关系模型	关系数据库管理系统(SQL)	文件系统
Relation (关系)	Table (表)	File of Records (文件)
Attribute (属性)	Column (列)	Field (字段)
Tuple (元组)	Row (行)	Record (记录)
Schema (模式)	Table Heading (表头)	Type of Record (记录型)

2.2 Naming the Parts of a Database

table name

column name

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	2.00
c003	Allied	Dallas	5.00
c004	MCME	Duluth	8.00
c006	MCME	Kyoto	0.00

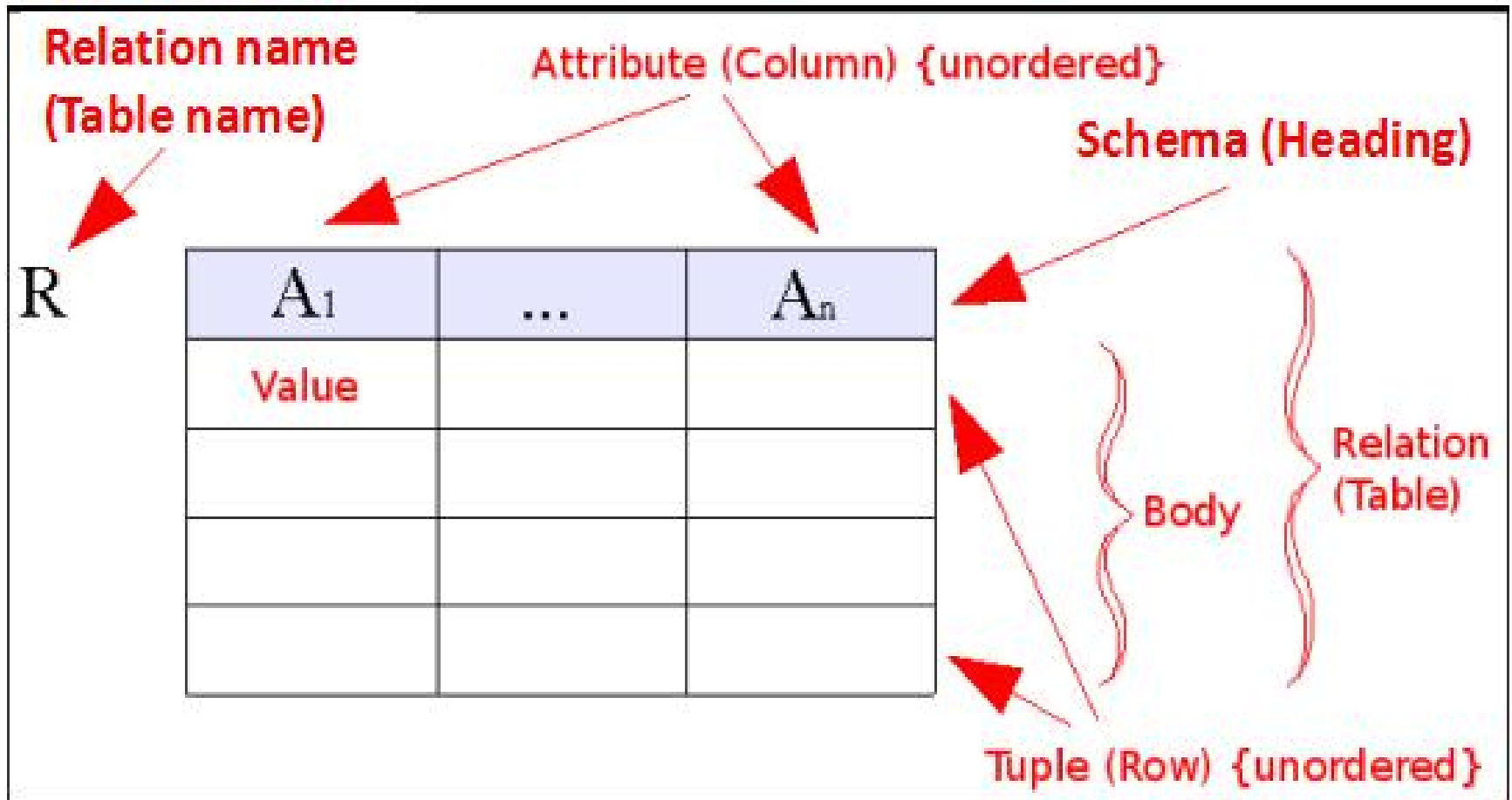
table heading

row

table

column

2.2 Naming the Parts of a Database



2.2 Naming the Parts of a Database

□ Definition

⌘ database

- a set of named tables, or relations.

**CAP = { CUSTOMERS, AGENTS,
PRODUCTS, ORDERS }**

⌘ Heading of table (schema)

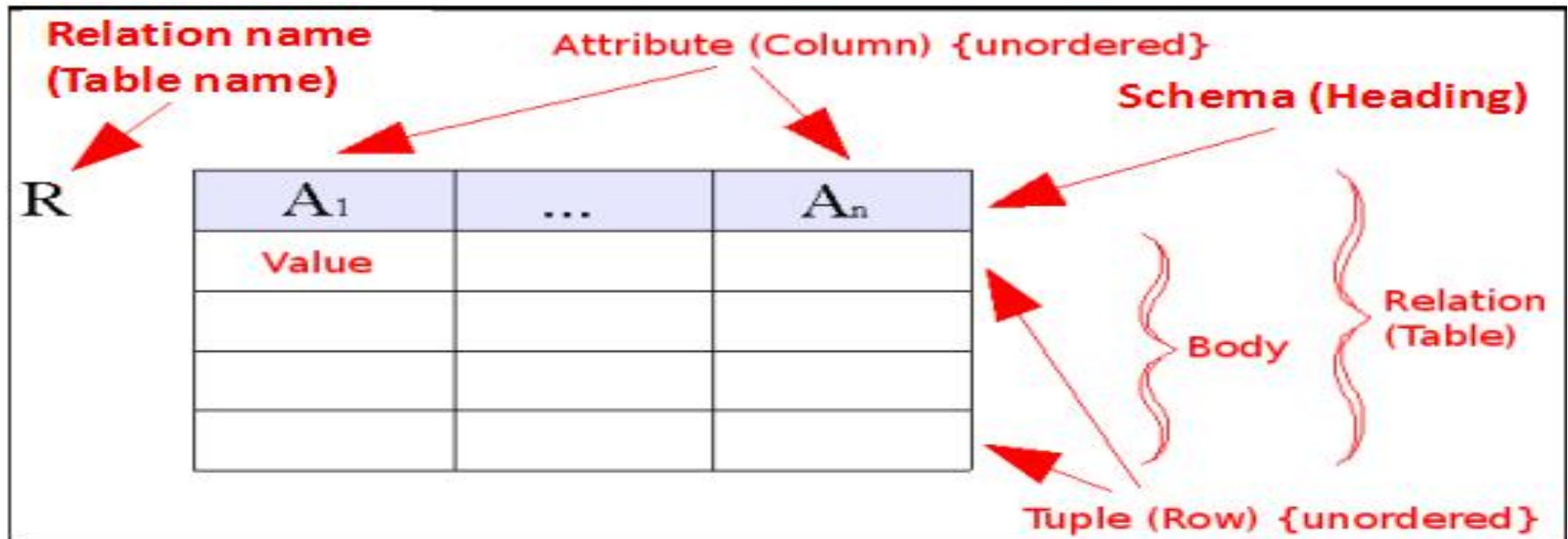
- a set of named columns

**Head(CUSTOMERS) =
{ cid, cname, city, discnt }**

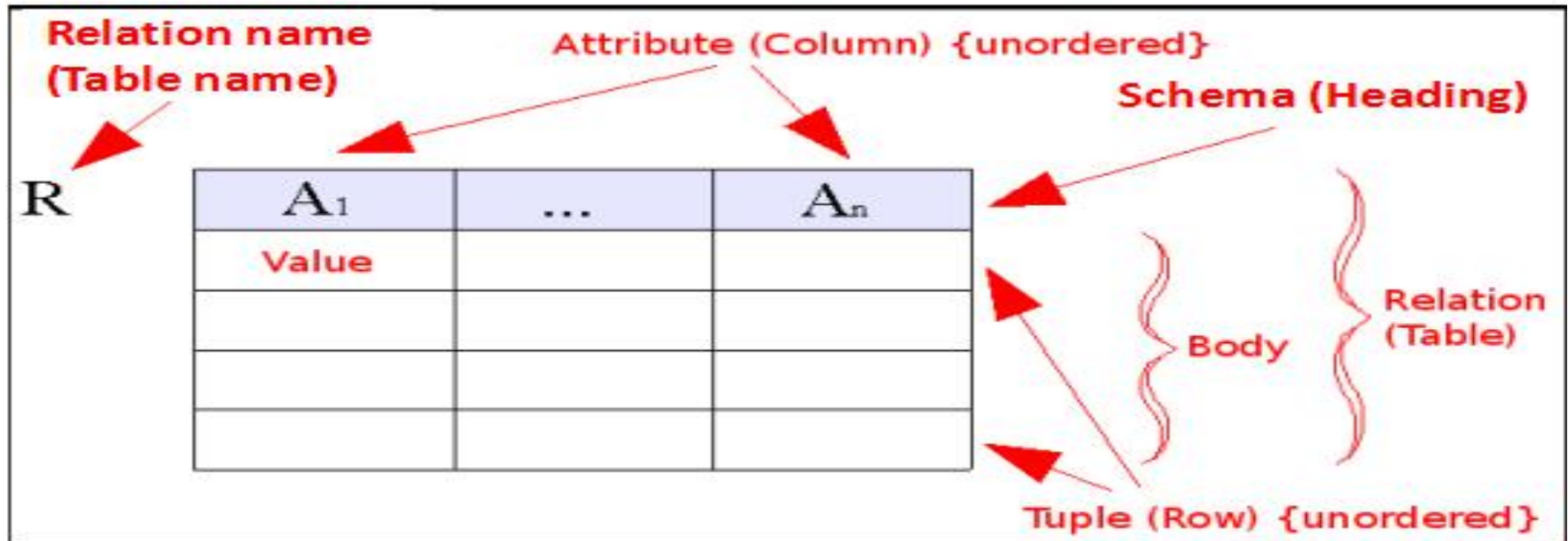
2.2 Naming the Parts of a Database

□ Note

- ✧ The number of rows changes frequently and rows are not remembered by users;
- ✧ the columns usually DON'T change in number, many names are remembered, and **USED TO POSE QUERIES.**



2.2 Naming the Parts of a Database



□ Program-Data Independence (数据独立性)

- when asked to make up a query to answer a question, query must still answer the question even if all the data changes.

http://en.wikipedia.org/wiki/Data_independence

2.2 Naming the Parts of a Database

□ **Column type** (也称 Domain, Datatype)

⌘ **Domain(city):** a set of city names

⌘ A table is **DECLARED** in SQL.

⌘ Columns have certain **TYPES** as in SQL:

- real, integer, char(13), date
- ...

2.2 Naming the Parts of a Database

□ Problems of Column type (cont.)

1. Most commercial database systems don't support types consisting of enumerated sets(as with city, month).

- *Integrity* (完整性)

2. Domain(city in CUSTOMERS) vs Domain(city in AGENTS) ?

- *particular type*

2.2 Naming the Parts of a Database

□ Relational Algebra (关系代数)

∞ **Domain of column is like an enumerated type.**

- **Domain(City)**

- **all the city names in the U.S.**

- **Domain(Discnt)**

- **all floats between 0.00 and 20.00.**

2.2 Naming the Parts of a Database

□ Cartesian Product (笛卡儿乘积)

✂ Set: $CID = \text{Domain}(cid)$, $CNAME = \text{Domain}(cname)$, $CITY = \text{Domain}(city)$, $DISCNT = \text{Domain}(discnt)$, then consider:

$CID \times CNAME \times CITY \times DISCNT$

consisting of all tuples: (w, x, y, z) , w in CID , x in $CNAME$, y in $CITY$, z in $DISCNT$

$CID \times CNAME \times CITY \times DISCNT =$

$\{(w, x, y, z) \mid w \in CID, x \in CNAME, y \in CITY, z \in DISCNT\}$

– A relation between these four domains is a subset of the Cartesian product.

$CUSTOMERS \subseteq CID \times CNAME \times CITY \times DISCNT$

2.2 Naming the Parts of a Database

- 设有一个由学号sno、姓名name和系别dept等三个属性所构成的学生关系S:

S	sno	name	dept
	1	张山	数学
	2	李英	数学
	3	王华	中文

- 在该关系中加入上述的三个学生元组，在关系代数中，上述的学生关系可以被表示为下面的集合:
- $S = \{ (1, \text{张三}, \text{数学}), (2, \text{李英}, \text{数学}), (3, \text{王华}, \text{中文}) \}$

2.2 Naming the Parts of a Database

S	sno	name	dept
	1	张山	数学
	2	李英	数学
	3	王华	中文

□ 假设只考虑上述的三个学生元组，那么：

Domain(sno) = {1, 2, 3}

Domain(name) = {张三, 李英, 王华}

Domain(dept) = {数学, 中文}

□ 将上述的三个值域进行笛卡尔乘积运算可得到如下的关系表W：

W

1	张山	数学
1	李英	数学
1	王华	数学
2	张山	数学
2	李英	数学
2	王华	数学
3	张山	数学
3	李英	数学
3	王华	数学

1	张山	中文
1	李英	中文
1	王华	中文
2	张山	中文
2	李英	中文
2	王华	中文
3	张山	中文
3	李英	中文
3	王华	中文

□ 原关系S (背景为黄色的三个元组所组成的集合)
显然只是笛卡尔乘积结果关系W的一个真子集。

2.3 Relational Rules

students

sid	Iname	fname	class	telephone
1	Jones	Allan	2	555-1234
2	Smith	John	3	555-4321
3	Brown	Harry	2	555-1122
5	White	Edward	3	555-3344

← **Relation Model**

students

sid	name		class	telephone	enrollment	
	Iname	fname			cno	major
1	Jones	Allan	2	555-1234	101	No
					108	Yes
2	Smith	John	3	555-4321	105	No
3	Brown	Harry	2	555-1122	101	Yes
					108	No
5	White	Edward	3	555-3344	102	No
					105	No

**Object-Relation
Model** →

2.3 Relational Rules

- ❑ **Rule 1. First Normal Form Rule**
- ❑ **Rule 2. Access Rows by Content Only Rule**
- ❑ **Rule 3. The Unique Row Rule**

2.3 Relational Rules

❑ Rule 1. First Normal Form Rule

⌘ columns that have *multi-valued attributes* (repeating fields) or have any *internal structure* (record) are not permitted.

students						
sid	name		class	telephone	enrollment	
	lname	fname			cno	major
1	Jones	Allan	2	555-1234	101	No
					108	Yes
2	Smith	John	3	555-4321	105	No
3	Brown	Harry	2	555-1122	101	Yes
					108	No
5	White	Edward	3	555-3344	102	No
					105	No

enrollment have multi-valued attr. & name have internal structure

Rule 1. First Normal Form Rule

- ❑ 对于含内部结构的属性**name**，可以将其所有的成员属性合并在一起，或者用其成员属性来代替属性**name**。
- ❑ 两种方案的区别仅在于从数据库中可以访问到的最小‘**name**’数据单位不同。

方案 1

sid	name	class	telephone
1	Jones Allan	2	555-1234
2	Smith John	3	555-4321
3	Brown Harry	2	555-1122
5	White Edward	3	555-3344

方案 2

sid	lname	fname	class	telephone
1	Jones	Allan	2	555-1234
2	Smith	John	3	555-4321
3	Brown	Harry	2	555-1122
5	White	Edward	3	555-3344

Rule 1. First Normal Form Rule

- ❑ In relational model, employees table can't have column "dependents" which contains multiple dependent's names (Figure 2.3)

eid	ename	position	dependents
e001	Smith, John	Agent	Michael J.
			Susan R.
e002	Andrews, David	Superintendent	David M. Jr.
e003	Jones, Franklin	Agent	Andrew K.
			Mark W.
			Louisa M.

Figure 2.3 An EMPLOYEES Table with a Multi-Valued dependents Attribute

In Object-Relational model, Figure 2.3 is OK!

Rule 1. First Normal Form Rule

□ 根据 rule 1 的要求，可以有三种模式设计修改方案：

- ① 在‘职工’表中设置多个 'dependents' 属性 (Figure 2.4)
- ② 只设置一个单值的 'dependents' 属性，然后通过多个不同的元组来记录一个职工的多个家属信息 (next slide)
- ③ 创建‘职工’和‘家属’两张关系表，之后可以通过表的join运算来获得Figure 2.3中的信息 (Figure 2.5)

但①和②都不是正确的模式设计方案！

eid	ename	position	dependents
e001	Smith, John	Agent	Michael J.
			Susan R.
e002	Andrews, David	Superintendent	David M. Jr.
e003	Jones, Franklin	Agent	Andrew K.
			Mark W.
			Louisa M.

Figure 2.3 An EMPLOYEES Table with a Multi-Valued dependents Attribute

eid	ename	position	dependent1	dependent2	...
e001	Smith, John	Agent	Michael J.	Susan R.	
e002	Andrews, David	Superintendent	David M. Jr.		
e003	Jones, Franklin	Agent	Andrew K.	Mark W.	...

Figure 2.4 An EMPLOYEES Table with Multiple Dependent Names on Each Employee

eid	ename	position	dependents
e001	Smith, John	Agent	Michael J.
			Susan R.
e002	Andrews, David	Superintendent	David M. Jr.
e003	Jones, Franklin	Agent	Andrew K.
			Mark W.
			Louisa M.

eid	ename	position	dependent
e001	Smith, John	Agent	Michael J.
e001	Smith, John	Agent	Susan R.
e002	Andrews, David	Superintendent	David M. Jr.
e003	Jones, Franklin	Agent	Andrew K.
e003	Jones, Franklin	Agent	Mark W.
e003	Jones, Franklin	Agent	Louisa M.

employee table with single-valued dependent attribute

Figure 2.5

EMPLOYEES

eid	ename	position
e001	Smith, John	Agent
e002	Andrews, David	Superintendent
e003	Jones, Franklin	Agent

DEPENDENTS

eid	dependent
e001	Michael J.
e001	Susan R.
e002	David M. Jr.
e003	Andrew K.
e003	Mark W.
e003	Louisa L.

方案③避免了前述的方案①和②的所有缺陷！

2.3 Relational Rules

□ Rule 2. Access Rows by Content Only Rule

⌘ can only retrieve rows by their content, the attribute values that exist in each row.

⌘ rule 2 implies:

- No order to the rows
- No order to the columns
- Disallows "pointers to rows" (RID - row ID)
- Most relational products break this rule by allowing users to **get at rows by RIDs**;
- new object-relational products **have refs as part of syntax**.

2.3 Relational Rules

□ Rule 3. The Unique Row Rule

✧ Two rows can't be same in all attributes at once.

✧ So that a relation is **an unordered SET of tuples**.

✧ But many products allow this for efficiency of load.

- There are even some tables where it is a good thing (*temperature readings in table, might repeat*).

□ But in the current Chapter, Chapter 2, we will assume that all these rules hold perfectly.

2.4 Keys, Superkeys, and Null Values

□ Idea of Keys

⌘ some set of columns in a table distinguishes rows.

⌘ [E.g] cid in CUSTOMERS, ordno in ORDERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

2.4 Keys, Superkeys, and Null Values

□ Key & Superkey

⌘ Superkey

- is a set of columns that has the uniqueness property
- [e.g.] A superkey for CUSTOMERS:
(cid, cname)

⌘ key

- is a minimal superkey: no subset of columns also has uniqueness property.
- [e.g.] A key for CUSTOMERS: cid

2.4 Keys, Superkeys, and Null Values

□ Def. 2.4.1 Table Key

✧ Given a table T , with $\text{Head}(T) = \{A_1, A_2, \dots, A_n\}$. A key for the table T , sometimes called a **candidate key** (候选关键字), is a set of attributes, $K = \{A_{i_1}, \dots, A_{i_k}\}$, with two properties:

- 1) If u, v are distinct tuples of T , then by designer intention $u[K] \neq v[K]$; that is, there will always exist at least one column, A_{im} , in the set of columns K such that $u[A_{im}] \neq v[A_{im}]$
- 2) No proper subset H of K has property 1

□ **Superkey**: a set of columns that fulfills property 1 but not necessarily property 2.

2.4 Keys, Superkeys, and Null Values

❏ Question: pg. 28, Figure 2.2, PRODUCTS

pid	pname	city	quantity	price
P01	comb	Dallas	111400	0.50
p02	brush	Newark	203000	0.50
p03	razor	Duluth	150600	1.00
p04	pen	Duluth	125300	1.00
p05	pencil	Dallas	221400	1.00
p06	folder	Dallas	123100	2.00
p07	case	Newark	100500	1.00

1) Is **pid** a key for PRODUCTS ? **pname** ? **city** ?

2) (**pid**, **city**) a key for PRODUCTS ? Is superkey ?

2.4 Keys, Superkeys, and Null Values

□ **Example 2.4.1.** Consider the table T:

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d1
a2	b1	c2	d1

- Assume intent of designer is that this table will remain with the same contents.
- Then can determine **keys** from content alone (this is a VERY UNUSUAL situation).

- 1) **No single column can be a key.**

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d1
a2	b1	c2	d1

-
- 原因：没有哪一个属性的取值能够满足关键字定义中的条件1！
 - 以属性‘A’为例，无法满足：“对表中的任意两个元组 t_1 和 t_2 ，都有 $t_1[A] \neq t_2[A]$ ”

- 1) No single column can be a key.
- 2) No set of columns can be a key if it contains D.

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d1
a2	b1	c2	d1

Because:

- the value of column D in row is same as each other
- If the set of **X** and **D** is a key (the set of X and D has property 1 of KEY), then the set of **X** has property 1 of KEY.

- 1) No single column can be a key.
- 2) No set of columns can be a key if it contains D.

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d1
a2	b1	c2	d1

- 3) Pairs: AB, AC, BC. Each of these pairs distinguishes all rows. Therefore all are keys.
-

- Is 'AB' a key?

A	B	C	D
a1	b1
a1	b2
a2	b1
a2	b2

- 1) No single column can be a key.
- 2) No set of columns can be a key if it contains D.

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d1
a2	b1	c2	d1

- 3) Pairs: AB, AC, BC. Each of these pairs distinguishes all rows. Therefore all are keys.
- 4) All other sets (*must have 3 or more elements*) contain one of these or else D as a proper subset.
- 5) Therefore no more keys.

2.4 Keys, Superkeys, and Null Values

- ❑ Various keys specified by intent of DBA are called **Candidate Keys**.
- ❑ A **Primary Key** (主关键字) is defined to be the **candidate key** chosen by the designer to identifies rows of T used in references by other tables.

2.4 Keys, Superkeys, and Null Values

□ Def. 2.4.3 Primary Key of a Table

- A primary key of a table T is the candidate key chosen by the database designer to uniquely identify specific rows of T.
 - **cid** is a primary key of **CUSTOMERS**
- maybe used in references by other tables.
 - **ORDERS** references **CUSTOMERS** by **cid**.

□ Th. 2.4.2. Every table T has at least one key.

□ Proof. Given a table T with $\text{Head}(T) = \{A_1 \dots A_n\}$.

– SET UP TO LOOP:

▪ Let attribute set S_1 be this entire set.

– LOOP:

▪ Now S_1 is a superkey for T; either S_1 is a Key for T or it has a proper subset S_2 , such that S_2 is also a superkey for T.

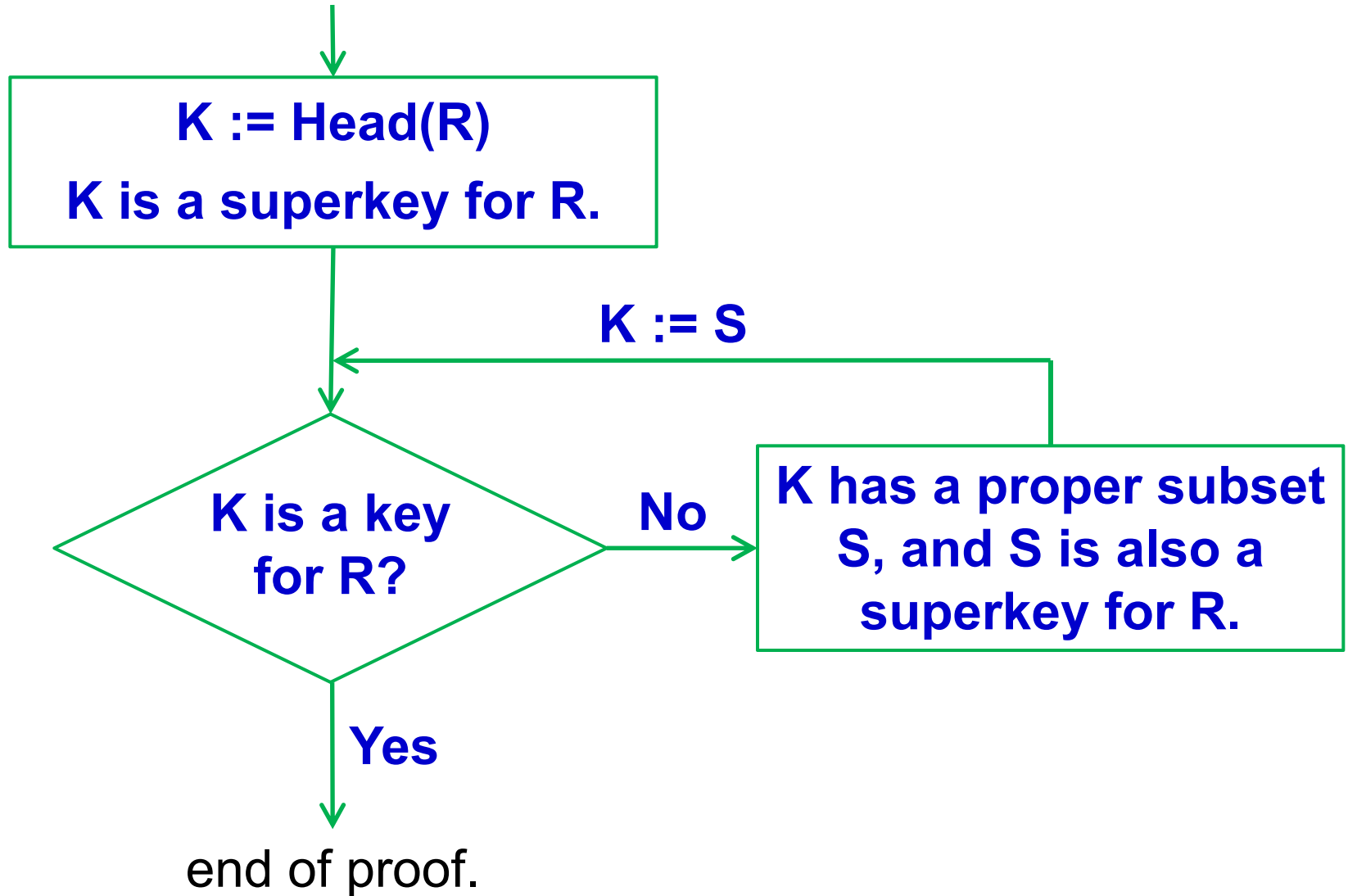
▪ Now, either S_2 is a Key or it has a proper subset S_3 , such that S_3 is also a superkey for T.

— But each successive element in the sequence S_1, S_2, S_3, \dots has a smaller number of columns (it is a PROPER subset), and we can never go to 0 (0 columns don't have unique values) or below.

— Thus this must be a finite sequence with a smallest set at the end S_n . That must be the key.

Proof.

input heading of relation R.



2.4 Keys, Superkeys, and Null Values

□ **Th. 2.4.2. Every table T has at least one key.**

□ **Proof.** Given a table T with $\text{Head}(T) = \{A_1 \dots A_n\}$.

令 $K := \text{Head}(T)$, 则 K 是表 T 的一个超键(superkey)

While (K 是表 T 的超键)

{

IF (存在 K 的一个真子集 S , 且 S 也是 T 的超键)

THEN { $K := S$; continue; }

ELSE break;

}

Return key K for T ;

2.4 Keys, Superkeys, and Null Values

❏ Null Values (空值)

⌘ A null value is placed in a field of a table when a specific value is either unknown or inappropriate.

- [e.g.] Insert a new agent:

(a12, Beowulf, *unknown*, *unknown*)

- Agent hasn't been assigned a percent commission or city yet (still in training, but want to have a record of him).

2.4 Keys, Superkeys, and Null Values

□ Null Values (cont.)

- ⌘ A null value can be used for either a numeric or character type. **BUT IT HAS A DIFFERENT VALUE FROM ANY REAL FIELD.** In particular, it is not zero (0) or the null string (").
- ⌘ It is handled specially by commercial databases.

AGENTS


<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5
a12	Beowulf	<i>unknown</i>	<i>unknown</i>

- ① Query all agents with **percent** < 6 ?
- ② Query all agents with **percent**<6 or **percent**>=6 ?
- ③ Query all agents with **city** <> 'New York' ?

AGENTS

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5
a12	Beowulf	<i>unknown</i>	<i>unknown</i>

① Query all agents with **percent** < 6 ?



<u>aid</u>	aname	city	percent
a05	Otasi	Duluth	5
a06	Smith	Dallas	5

AGENTS

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5
a12	Beowulf	<i>unknown</i>	<i>unknown</i>

② Query all agents with **percent** < 6 or **percent** >= 6 ?

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5

AGENTS

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5
a12	Beowulf	<i>unknown</i>	<i>unknown</i>

③ Query all agents with **city** <> 'New York' ?

<u>aid</u>	aname	city	percent
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a05	Otasi	Duluth	5
a06	Smith	Dallas	5

2.4 Keys, Superkeys, and Null Values

□ Rule 3. Entity Integrity Rule

(实体完整性)

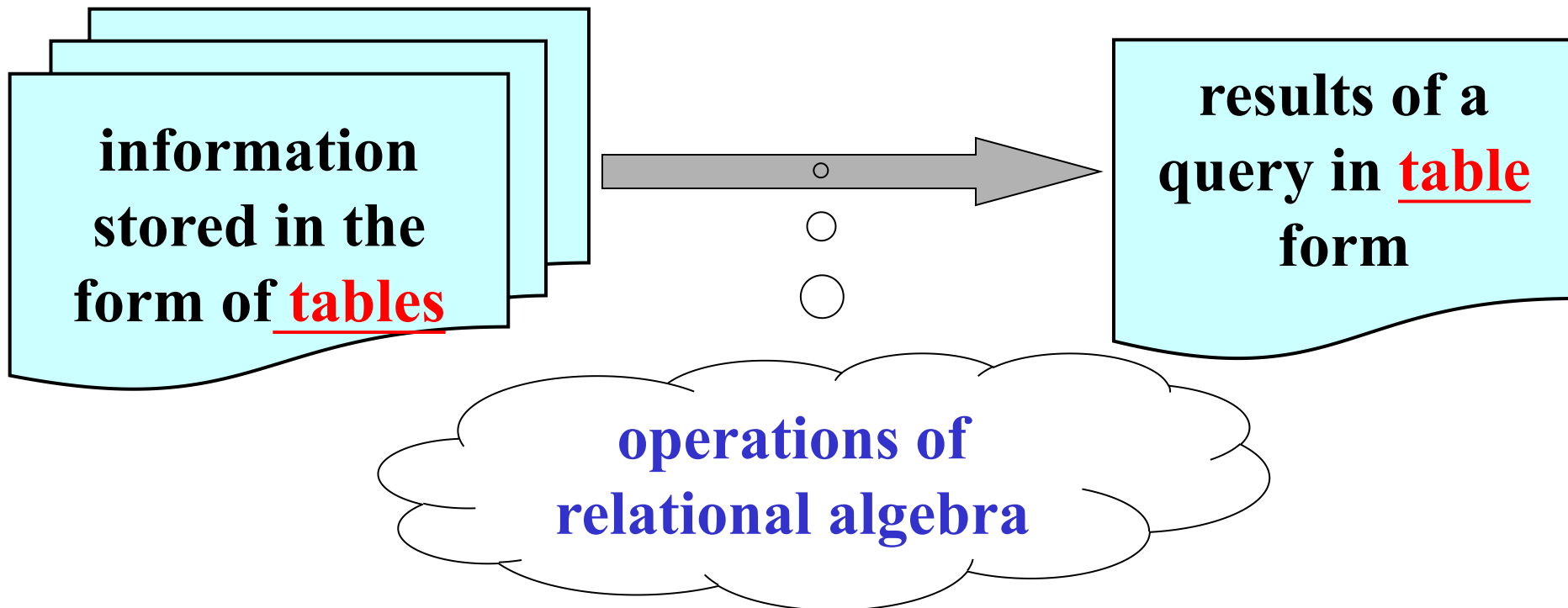
✧ No Column belonging to a primary key of a table T is allowed to take on **NULL** values for any row in T.

□ 与 “Rule 3. The Unique Row Rule” 是等价的。

2.5 Relational Algebra

□ Relational Algebra (关系代数)

✧ abstract language introduced by E. F. Codd



2.5 Relational Algebra

□ Fundamental Operations of Relational Algebra

✧ **two types of operations**

- set-theoretic operations

- depend on fact that table is a set of rows

- native relational operations

- depend on structure of table

□ set-theoretic operations

NAME	SYMBOL	FORM	EXAMPLE
UNION (并)	\cup	UNION	$R \cup S$
INTERSECTION (交)	\cap	INTERSECT	$R \cap S$
DIFFERENCE (差)	$-$	MINUS	$R - S$
PRODUCT (乘积)	\times	TIMES	$R \times S$

□ native relational operations

NAME	SYMBOL	FORM	EXAMPLE
PROJECT (投影)	R []	R []	R[A _{i1} ,...,A _{ik}]
	π		$\pi_{A_{i1}, \dots, A_{ik}}(R)$
SELECT (选择)	<i>R where C</i>	<i>R where C</i>	<i>R where A₁=5</i>
	σ	$\sigma_C(R)$	$\sigma_{A_1=5}(R)$
JOIN (联接)	\bowtie	JOIN	R \bowtie S
DIVISION (除法)	\div	DIVIDEBY	R \div S

2.6 Set-Theoretic Operations

□ Concepts in relational model

Def 2.6.1 Compatible Tables (相容表)

Def 2.6.2 Union, Intersection, Difference

Def 2.6.4 Product

Concepts

- 不论是哪一个关系（被访问的访问对象、访问结果，或者是访问过程中所生成的中间关系），我们都需要明确其关系模式和元组集合是什么。
- 以下图左所示的顾客关系**Customer**为例，其关系模式 **Head(Customer)={cid,cname,city,discnt}**；在关系代数中，该关系可以被看成是如下图右所示的元组集合：

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

{ <c001, TipTop, Duluth, 10.00>,
<c002, Basics, Dallas, 12.00>,
<c003, Allied, Dallas, 8.00>,
<c004, ACME, Duluth, 8.00>,
<c005, ACME, Kyoto, 0.00> }

2.6 Set-Theoretic Operations

□ Def. 2.6.1 Compatible Tables (相容表)

✧ Tables R and S are **compatible** if they have the same headings

✧ that is , if **Head(R)=Head(S)**, with attributes chosen from the same domains and with the same meanings.

Compatible Table (example 1)

R_1

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S_1

A	B	C
a1	b1	c1
a1	b1	c2
a1	b2	c3
a3	b2	c3

□ Tables R_1 and S_1 are *compatible* because:

✧ with the same number of columns

✧ each pair of columns (one in table R_1 , and another in table S_1) have:

- the same domains (character), and
- the same meanings, such as the same name of columns

Compatible Table (example 2)

R_2

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S_2

A	B	C
1	b1	c1
1	b1	c2
1	b2	c3
3	b2	c3

❑ Tables R_2 isn't *compatible* with table S_2 because:

- ✧ The domain of column **A** in table S_2 is integer, and the domain of column **A** in table R_2 is character, and that
- ✧ We can't find a pair of columns (**X**, **A**) (**X** is a column in table R_2 , and **A** is a column in table S_2) with the same domain (integer)

Compatible Table (example 3)

R_3

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S_3

A	B	D
a1	b1	c1
a1	b1	c2
a1	b2	c3
a3	b2	c3

❑ Tables R_3 isn't *compatible* with table S_3 because:

✧ We can't find a pair of columns(?, D) with the same meanings (? is a column in table R_3 , and D is a column in table S_3)

✧ In relational algebra, two columns are said to have the same meanings if they have the same name of column.

Compatible Table (example 4)

R_4

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S_4

A	B	C	D
a1	b1	c1	d1
a1	b1	c2	d1
a1	b2	c3	d1
a3	b2	c3	d1

❑ Tables R_4 isn't *compatible* with table S_4
because:

✧ The number of columns in table R_4 isn't
equal to the number of columns in table S_4

Compatible Table (example 5)

R_5

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S_5

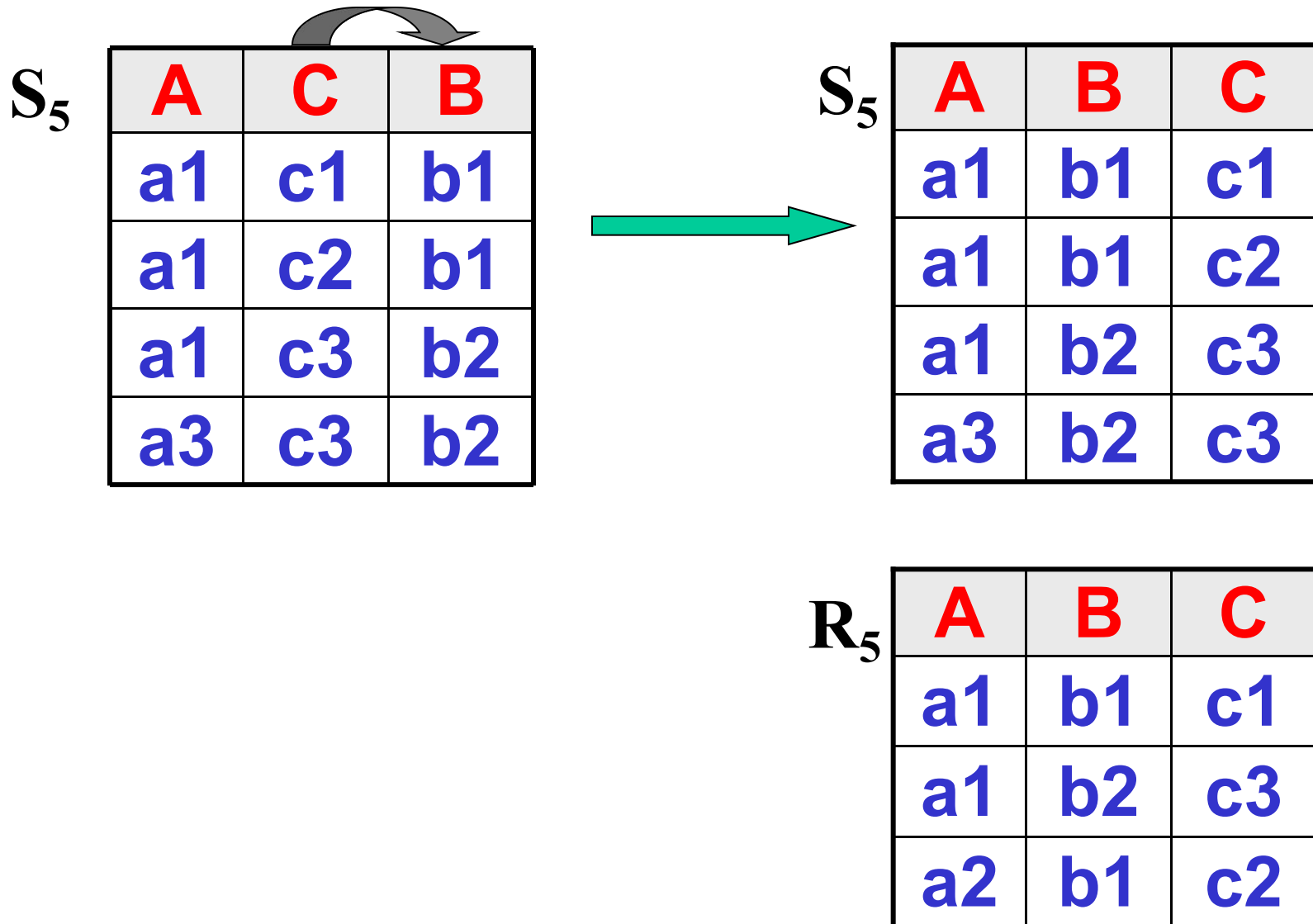
A	C	B
a1	c1	b1
a1	c2	b1
a1	c3	b2
a3	c3	b2

□ Tables R_5 and S_5 are *compatible* because:

✂ with the same headings of table

- We can move the column **B** to the left of the column **C** in table S_5 because **No order to the columns.**

Compatible Table (example 5)



2.6 Set-Theoretic Operations

□ Def 2.6.2 Union, Intersection, Difference

Let R and S be two compatible tables, where $\text{Head}(R) = \text{Head}(S)$

- $R \cup S$ is a table with the same heading as R (or S), for each row t in R or in S , t in $R \cup S$
- $R \cap S$ is a table with the same heading as R (or S), for each row t in R , if t appear in S , then t in $R \cap S$
- $R - S$ is a table with the same heading as R (or S), for each row t in R , if t don't appear in S , then t in $R - S$

[Define of Union] $R \cup S$

- 前提条件: $\text{Head}(R) = \text{Head}(S)$

- 结果关系:

- 1) $\text{Head}(R \cup S) = \text{Head}(R) = \text{Head}(S)$

- 2) The set of rows in the table $(R \cup S)$

```
T := R
```

```
For each row v in S
```

```
{
```

```
  If ( row v  $\notin$  R )
```

```
    Then ( add row v into T )
```

```
}
```

```
Return T
```

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

[Define of Intersection] $R \cap S$

- 前提条件: $\text{Head}(R) = \text{Head}(S)$

- 结果关系:

- 1) $\text{Head}(R \cap S) = \text{Head}(R) = \text{Head}(S)$

- 2) The set of rows in the table $(R \cap S)$

```
T := { }  
For each row u in R  
{  
    If ( row u  $\in$  S )  
    Then ( add row u into T )  
}  
Return T
```

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

[Define of Difference] **R – S**

- 前提条件: $\text{Head}(R) = \text{Head}(S)$

- 结果关系:

- 1) $\text{Head}(R - S) = \text{Head}(R) = \text{Head}(S)$

- 2) The set of rows in the table (R – S)

```
T := { }
```

```
For each row u in R
```

```
{
```

```
    If ( row u  $\notin$  S )
```

```
        Then ( add row u into T )
```

```
}
```

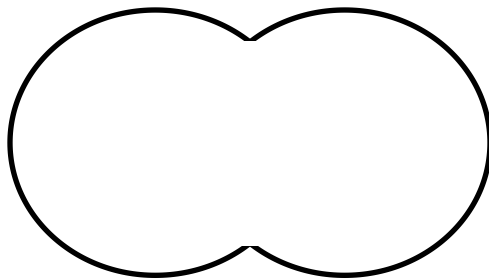
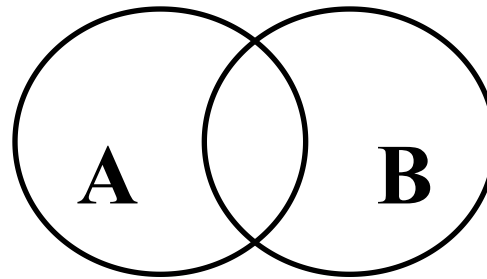
```
Return T
```

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

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

2.6 Set-Theoretic Operations

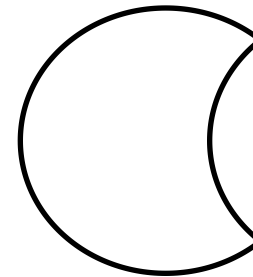
□ Example 2.6.2



$A \cup B$



$A \cap B$



$A - B$

2.6 Set-Theoretic Operations

□ The **UNION** and **INTERSECTION** operations are commutative (交换律) and associative (结合律).

1) $R \cup S = S \cup R$

2) $R \cap S = S \cap R$

3) $(R \cup S) \cup T = R \cup (S \cup T)$

4) $(R \cap S) \cap T = R \cap (S \cap T)$

2.6 Set-Theoretic Operations

□ The **DIFFERENCE** operation is not commutative

1) $R - S \neq S - R$

➤ $R - S = R - (R \cap S)$

➤ $S - R = S - (R \cap S)$

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

➤ $(R - S) \cap S = \Phi$

➤ $(R - S) \cap R = R - S$

➤ $(R - S) \cap (S - R) = \Phi$

R

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S

A	B	C
a1	b1	c1
a1	b1	c2
a1	b2	c3
a3	b2	c3

Exp. 2.6.1 The rows with blue background appear only in R, and the rows with yellow background appear only in S.

 $R \cup S$

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2
a1	b1	c2
a3	b2	c3

 $R \cap S$

A	B	C
a1	b1	c1
a1	b2	c3

 $R - S$

A	B	C
a2	b1	c2

 $S - R$

A	B	C
a1	b1	c2
a3	b2	c3

STUDENT (S)

FN	LN
Susan	Yao
Amy	Ford
Jimmy	Wang
Ramesh	Shah
John	Ford

INSTRUCTOR (I)

FN	LN
Ramesh	Shah
Francis	Johnson
Susan	Yao
John	Smith

S \cup I

FN	LN
Susan	Yao
Amy	Ford
Jimmy	Wang
Ramesh	Shah
John	Ford
Francis	Johnson
John	Smith

S \cap I

FN	LN
Susan	Yao
Ramesh	Shah

STUDENT (S)

FN	LN
Susan	Yao
Amy	Ford
Jimmy	Wang
Ramesh	Shah
John	Ford

INSTRUCTOR (I)

FN	LN
Ramesh	Shah
Francis	Johnson
Susan	Yao
John	Smith

S – I

FN	LN
Amy	Ford
Jimmy	Wang
John	Ford

I – S

FN	LN
Francis	Johnson
John	Smith

2.6 Set-Theoretic Operations

□ Def 2.6.3 Assignment, Alias

⌘ Table **R** : $R(A_1, A_2, \dots, A_n)$

⌘ 赋值运算:

$$S(B_1, B_2, \dots, B_n) := R(A_1, A_2, \dots, A_n)$$

⌘ Define a new table **S**, and

$$\text{Domain}(B_i) = \text{Domain}(A_i) \text{ for all } i (1 \leq i \leq n)$$

⌘ The content of the new table **S** is exactly the same as the old table **R**.

- for each row **t** in table **R**, **t** in table **S**, and
- for each row **t** in table **S**, **t** in table **R**

2.6 Set-Theoretic Operations

□ 表的赋值: $S := R$

- 简单的‘表的赋值’操作，可用于为关系 R 起一个别名 (**alias**)
- 或者理解为：定义了一个与关系 R ‘完全相同’的中间关系 S (相同的模式模式和元组集合)
- 通常用于 ‘关系自身的连接运算’ !

□ 赋值运算也可以用来保存计算的 ‘中间结果’

2.6 Set-Theoretic Operations

□ **Example 2.6.3:** $T := (R \cup S) - (R \cap S)$

✧ **We can define the table T by writing**

a) $T_1 := (R \cup S)$

b) $T_2 := (R \cap S)$

c) $T := T_1 - T_2$

R

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S

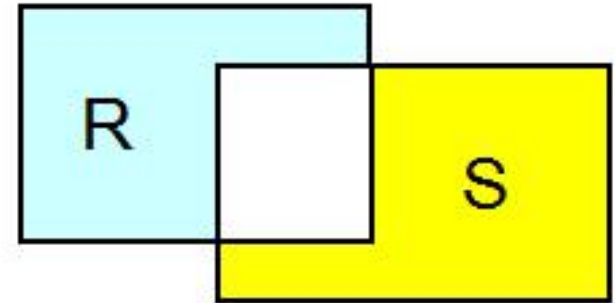
A	B	C
a1	b1	c1
a1	b1	c2
a1	b2	c3
a3	b2	c3

 $T_1 := R \cup S$

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2
a1	b1	c2
a3	b2	c3

 $T_2 := R \cap S$

A	B	C
a1	b1	c1
a1	b2	c3

Example 2.6.3**The picture of T
(blue&yellow)** **$T := T_1 - T_2$**

A	B	C
a2	b1	c2
a1	b1	c2
a3	b2	c3

2.6 Set-Theoretic Operations

□ Def 2.6.4 Product $R \times S$

The product of the tables R and S is a table T that

- if $\text{Head}(R)=\{A_1,\dots,A_n\}$, $\text{Head}(S)=\{B_1,\dots,B_m\}$, then

$$\text{Head}(T) = \{R.A_1, \dots, R.A_n, S.B_1, \dots, S.B_m\}$$

- t is a row in T if and only if

There are two rows u in R and v in S such that

$$t(R.A_i) = u(A_i) \text{ for } 1 \leq i \leq n, \text{ and}$$

$$t(S.B_k) = v(B_k) \text{ for } 1 \leq k \leq m$$

- 1) $\text{Head}(R) = \{ A_1, \dots, A_n \}, \text{Head}(S) = \{ B_1, \dots, B_m \}$
- 2) $\text{Head}(R \times S) = \{ A_1, \dots, A_n, B_1, \dots, B_m \}$
- 3) The set of rows in the table $R \times S$

For each row u in R

{ For each row v in S

{

Then we have a row t in the table $R \times S$

Which

{

$t[A_i] = u[A_i]$ for all $i, 1 \leq i \leq n$

$t[B_j] = v[B_j]$ for all $j, 1 \leq j \leq m$

}

}

}

$$R \times S = \{ (u, v) \mid u \in R \wedge v \in S \}$$

R

A	B	C
a1	b1	c1
a1	b2	c3
a2	b1	c2

S

B	C	D
b1	c1	d1
b1	c1	d3
b2	c2	d2
b1	c2	d4

Example 2.6.4

$R \times S$

R.A	R.B	R.C	S.B	S.C	S.D
a1	b1	c1	b1	c1	d1
a1	b1	c1	b1	c1	d3
a1	b1	c1	b2	c2	d2
a1	b1	c1	b1	c2	d4
a1	b2	c3	b1	c1	d1
a1	b2	c3	b1	c1	d3
a1	b2	c3	b2	c2	d2
a1	b2	c3	b1	c2	d4
a2	b1	c2	b1	c1	d1
a2	b1	c2	b1	c1	d3
a2	b1	c2	b2	c2	d2
a2	b1	c2	b1	c2	d4

$$T := R \times S$$

□ If the number of columns in table **R** is C_R , the number of columns in table **S** is C_S , then

- the number of columns in product of tables **R** and **S** is $(C_R + C_S)$

□ If the number of rows in table **R** is N_R , the number of rows in table **S** is N_S , then

- the number of rows in product of tables **R** and **S** is $(N_R \times N_S)$

2.7 Native Relational Operations

□ The Native Relational Operations of relational algebra are

✧ **projection (π)**

- select columns in table

✧ **selection (σ)**

- select rows in table

✧ **join (\Join)**

- combine a table with another table or itself

✧ **division (\div)**

- a complex operation of relational algebra

□ **Def 2.7.1 Projection:** $R [A_{i1}, A_{i2}, \dots, A_{ik}]$

⌘ **Head(R) = $\{A_1, A_2, \dots, A_n\}$**

⌘ **$A_{ij} \in \text{Head}(R)$ for all $1 \leq j \leq k$**

⌘ **the projection of R on attributes $A_{i1}, A_{i2}, \dots, A_{ik}$, is a table T that**

■ **$\text{Head}(T) = \{ A_{i1}, \dots, A_{ik} \}$**

■ **for every row r in the table R there will be a single row t in the table T such that**

➤ **$r[A_{ij}] = t[A_{ij}]$ for $1 \leq j \leq k$**

⌘ **can be writed as $\pi_{A_{i1}, \dots, A_{ik}} (R)$**

⌘ **cast out duplicate rows in the result of projection!**

Example 2.7.1

- list all customer names from the CUSTOMERS (is denoted by **C**) table in figure 2.2

C

CN := **C[cname]** or **CN** := $\pi_{\text{cname}}(\mathbf{C})$

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

CN

cname
TipTop
Basics
Allied
ACME

Example 2.7.1 (cont.)

C

<u>cid0</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

$\pi_{\text{city}}(\mathbf{C})$

city
Duluth
Dallas
Kyoto

❑ cast out duplicate rows '**Duluth**' and '**Dallas**' in the result of the projection of C on column 'city'.

Example 2.7.1 (cont.)

S

A	B	C
a1	b1	c1
a1	b1	c3
a2	b2	c2
a1	b2	c4

S[A]

A
a1
a2

S[B,A]

B	A
b1	a1
b2	a2
b2	a1

S[B,C]

B	C
b1	c1
b1	c3
b2	c2
b2	c4

cast out
duplicate
rows (a1)

cast out
duplicate
rows (b1, a1)

2.7 Native Relational Operations

□ Def 2.7.2 Selection

☞ Assume $\text{Head}(\mathbf{S}) = \{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n\}$, the selection operation on \mathbf{S} creates a new table, denoted by

$\mathbf{S} \text{ where } \mathbf{C}$ or $\sigma_{\mathbf{C}}(\mathbf{S})$

with the same set of attributes, and consisting of those rows of \mathbf{S} that obey the selection condition \mathbf{C} .

2.7 Native Relational Operations

□ **Def. The selection condition C**

1) C can be any comparison of the form

$$A_i \theta A_j \quad \text{or} \quad A_i \theta a$$

where A_i and A_j are attributes of S having the same domain, a is a constant from $\text{Domain}(A_i)$, and θ is one of the comparison operators $<$, $>$, $=$, \leq , \geq , \neq

2) If C and C' are conditions, then new conditions can be writing

$$C \text{ AND } C', \quad C \text{ OR } C', \quad \text{NOT } C$$

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

Example 2.7.2

□ Find all customers based in 'Kyoto'

T := CUSTOMERS where city = 'Kyoto'

T	<u>cid</u>	cname	city	discnt
	c006	ACME	Kyoto	0.00

<u>pid</u>	pname	city	quantity	price
P01	comb	Dallas	111400	0.50
p02	brush	Newark	203000	0.50
p03	razor	Duluth	150600	1.00
p04	pen	Duluth	125300	1.00
p05	pencil	Dallas	221400	1.00
p06	folder	Dallas	123100	2.00
p07	case	Newark	100500	1.00

❑ Find the products stored in 'Dallas' that cost more than \$0.50

PRODUCTS where city='Dallas' AND price>0.50

□ **Exp. 2.7.3: Retrieve all pairs of agents, both with a percentage commission of at least 6%, and both stationed in the same city.**

– **Step 1: All agents who have a percentage commission of at least 6%**

▪ **L := AGENTS where percent \geq 6**

▪ **M := AGENTS where percent \geq 6**

(M is the same table with L)

L

aid	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

Example 2.7.3

☞ **Step 2: Retrieve all pairs of agents, both with a percentage commission of at least 6%, and both stationed in the same city.**

L

<u>aid</u>	...	city	per.
a01	...	New York	6
a02	...	Newark	6
a03	...	Tokyo	7
a04	...	New York	6

M

<u>aid</u>	...	city	per.
a01	...	New York	6
a02	...	Newark	6
a03	...	Tokyo	7
a04	...	New York	6

PAIRS := (L × M) where L.city = M.city

L

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

M

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

PAIRS := (L × M) where L.city = M.city

<u>L.aid</u>	L.aname	L.city	L.percent	<u>M.aid</u>	M.aname	M.city	M.percent
a01	Smith	New York	6	a01	Smith	New York	6
a01	Smith	New York	6	a04	Gray	New York	6
a02	Jones	Newark	6	a02	Jones	Newark	6
a03	Brown	Tokyo	7	a03	Brown	Tokyo	7
a04	Gray	New York	6	a01	Smith	New York	6
a04	Gray	New York	6	a04	Gray	New York	6

L

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

M

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

(L × M) where L.city = M.city and L.aid < M.aid

<u>L.aid</u>	L.aname	L.city	L.percent	<u>M.aid</u>	M.aname	M.city	M.percent
a01	Smith	New York	6	a01	Smith	New York	6
a01	Smith	New York	6	a04	Gray	New York	6
a02	Jones	Newark	6	a02	Jones	Newark	6
a03	Brown	Tokyo	7	a03	Brown	Tokyo	7
a04	Gray	New York	6	a01	Smith	New York	6
a04	Gray	New York	6	a04	Gray	New York	6

Example 2.7.4

□ In example 2.7.3, can we replace P2 with P3 ?

P2 := (L×M) where L.city=M.city and L.aid<M.aid

P3 := (L×M) where L.city = 'New York' and
M.city = 'New York' and L.aid < M.aid

L

<u>aid</u>	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6

2.7 Native Relational Operations

□ Def 2.7.3 Precedence of Relational Operations

Precedence	Operators
Highest	PROJECT
	SELECT
	PRODUCT
	JOIN, DIVIDEBY
	INTERSECTION
Lowest	UNION, DIFFERENCE

Example 2.7.5 Complex Query

⌘ CUSTOMERS \rightarrow C

AGENTS \rightarrow A

⌘ PRODUCTS \rightarrow P

ORDERS \rightarrow O

[exp 2.7.5] Find all cities where we have either customers who have a discount of less than 10% or agents who make a commission of less than 6%.

$T_1 := (C \text{ where } \text{discnt} < 10) [\text{city}]$

$T_2 := (A \text{ where } \text{percent} < 6) [\text{city}]$

$T := T_1 \cup T_2$

[exp 2.7.5] Find all cities where we have either customers who have a discount of less than 10% or agents who make a commission of less than 6%.

$T_1 := (C \text{ where } \text{discnt} < 10) [\text{city}]$

$T_2 := (A \text{ where } \text{percent} < 6) [\text{city}]$

$T := T_1 \cup T_2$

另外一种错误的表示方法:

$R_1 := (C \text{ where } \text{discnt} < 10)$

$R_2 := (A \text{ where } \text{percent} < 6)$

$R := (R_1 \cup R_2) [\text{city}]$

其中的‘并’运算非法!

Complex Query

❑ Find cname and city of customers and aname of agents that the customer lives in the same city with agents.

$((C \times A) \text{ where } C.\text{city} = A.\text{city})[C.\text{cname}, C.\text{city}, A.\text{aname}]$

能否用如下的分步表示形式来表示该查询？

① $T := (C \times A) \text{ where } C.\text{city} = A.\text{city}$

② $R := T [C.\text{cname}, C.\text{city}, A.\text{aname}]$

可以，但不够严谨！

Complex Query

□ Find pid, month and qty of order whose customer's name is 'Allied'

((C x O) where C.cid=O.cid and C.cname='Allied')
[O.pid, O.month, O.qty]

or

((((C where cname='Allied') [cid]) x O)
where C.cid=O.cid) [O.pid,O.month,O.qty]

Complex Query

Find ordno of orders for customer, agent and product combinations that are all in the same city.

$((C[city, city] \times A[aid, city] \times P[pid, city] \times O)$
where $C.city = A.city$ and $P.city = A.city$
and $C.cid = O.cid$
and $A.aid = O.aid$
and $P.pid = O.pid) [O.ordno]$

Complex Query

- Find cid of customers who have a highest discount in all customers.

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

- Find cid of customers who have a highest discount in all customers.

CUSTOMERS

<u>cid</u>	cname	city	discent
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

【思路】要判断一个客户c的折扣是不是最高的，检查方法如下：

- ① 如果存在一个客户k，客户k的折扣大于客户c的折扣，那么客户c的折扣就不是最高的
- ② 如果对于所有的客户k来说，客户k的折扣都小于或等于客户c的折扣，那么客户c的折扣就是当前所有客户中最高的

- 要确定客户c的折扣是最高的(情况②)，需要遍历Customers表中的所有客户元组k，且都满足： $k.\text{discent} \leq c.\text{discent}$ (*difficult*)
- 如果仅仅是想确认客户c的折扣不是最高的(情况①)，只要找到一个客户元组k满足： $k.\text{discent} > c.\text{discent}$ (*easy*)

Complex Query

❑ Find cid of customers who have a highest discount in all customers.

1) Find all cid of customers

$R_1 := C [cid]$

2) Find cid of customers whose discount less than another customer. Let $S := C$, then

$R_2 := ((C \times S) \text{ where } C.\text{discnt} < S.\text{discnt})[C.cid]$

3) calculate the result of this query by DIFFERENCE operation.

$T := R_1 - R_2$

2.7 Native Relational Operations

□ The Native Relational Operations of relational algebra are

✧ **projection (π)**

- select columns in table

✧ **selection (σ)**

- select rows in table

✧ **join (\Join)**

- combine a table with another table or itself

✧ **division (\div)**

- a complex operation of relational algebra

2.7 Native Relational Operations

□ Def 2.7.4 Join (equijoin, natural join)

✧ Join 运算被称为 ‘联接’ 运算

✧ 又被称为 ‘等值联接’ (equijoin)

✧ 或者 ‘自然联接’ (natural join)

✧ 其作用是：根据两个关系中的同名属性的相等比较，来实现两个关系的合并

✧ 可以表示为 ‘ $R \text{ join } S$ ’ 或者 ‘ $R \bowtie S$ ’

✧ 其形式化定义如下

□ Def 2.7.4 Join

⌘ We have two tables **R** and **S**, with headings:

$$\text{Head}(R) = \{ A_1, \dots, A_n, B_1, \dots, B_k \}$$

$$\text{Head}(S) = \{ B_1, \dots, B_k, C_1, \dots, C_m \}$$

⌘ Then

$$\text{Head}(R \bowtie S) = \{ A_1, \dots, A_n, B_1, \dots, B_k, C_1, \dots, C_m \}$$

请注意，与笛卡尔乘积的结果关系模式的区别

$$\text{Head}(R \times S) = \{ A_1, \dots, A_n, R.B_1, \dots, R.B_k, S.B_1, \dots, S.B_k, C_1, \dots, C_m \}$$

□ Then

1) $\text{Head}(R \bowtie S) = \{ A_1, \dots, A_n, B_1, \dots, B_k, C_1, \dots, C_m \}$

2) The set of rows in the table $R \bowtie S$

For each row u in R

{

For each row v in S

{

If ($u[B_i] = v[B_i]$ for all i ($1 \leq i \leq k$))

Then we have a row t in the table $R \bowtie S$

Which {

$t[A_i] = u[A_i]$ for all i , $1 \leq i \leq n$

$t[C_i] = v[C_i]$ for all i , $1 \leq i \leq m$

$t[B_i] = u[B_i] = v[B_i]$ for all i , $1 \leq i \leq k$

}

}

}

Example 2.7.6

R

A	B1	B2
a1	b1	b1
a1	b2	b1
a2	b1	b2

S

B1	B2	C
b1	b1	c1
b1	b1	c2
b1	b2	c3
b2	b2	c4



$R \bowtie S$

A	B1	B2	C
a1	b1	b1	c1
a1	b1	b1	c2
a2	b1	b2	c3

□ 在这里， **$R \bowtie S$** 相当于是：

$((R \times S) \text{ where } R.B1 = S.B1 \text{ and } R.B2 = S.B2) [R.A, R.B1, R.B2, S.C]$

□ 可以仔细体会 **$R \bowtie S$** 与 **$R \times S$** 的区别

与join运算有关的一些等价变换规则

□ If $\text{Head}(R) \cap \text{Head}(S) = \Phi$, then $R \bowtie S = R \times S$

□ If $\text{Head}(R) = \text{Head}(S)$, then $R \bowtie S = R \cap S$

□ If $\text{Head}(R) \subseteq \text{Head}(S)$, then $R \bowtie S \subseteq S$

Complex Query

Find cname and city of customers and aname of agents that the customer lives in the same city with agents.

(基于笛卡尔乘积的查询表示方法)

$((C \times A) \text{ where } C.\text{city} = A.\text{city})[C.\text{cname}, C.\text{city}, A.\text{aname}]$

CUSTOMERS

cid	cname	city	discnt
-----	-------	------	--------

AGENTS

aid	aname	city	percent
-----	-------	------	---------

采用JOIN运算可以将该查询简化表示为:

$(C \bowtie A)[C.\text{cname}, C.\text{city}, A.\text{aname}]$

Complex Query

Find pid, month and qty of order whose customer's name is 'Allied'

$((C \times O) \text{ where } C.cid = O.cid \text{ and } C.cname = 'Allied') [O.pid, O.month, O.qty]$

or

$(((C \text{ where } cname = 'Allied') [cid]) \times O) \text{ where } C.cid = O.cid) [O.pid, O.month, O.qty]$

可简化表示为:

$((C \bowtie O) \text{ where } C.cname = 'Allied') [O.pid, O.month, O.qty]$

Complex Query

Find ordno of orders for customer, agent and product combinations that are all in the same city.

$((C[city] \times A[city] \times P[city] \times O)$
where $C.city = A.city$ and $P.city = A.city$
and $C.cid=O.cid$
and $A.aid=O.aid$
and $P.pid=O.pid$)
[O.ordno]

可简化表示为: $(C \bowtie A \bowtie P \bowtie O) [O.ordno]$

Example 2.7.7

Customers: C(cid, cname, city, discnt)

Agents: A(aid, aname, city, percent)

Products: P(pid, pname, city, quantity, price)

Orders: O(ordno, month, cid, aid, pid, qty, dollars)

□ **Example 2.7.7: Find the names of the customers who have ordered product 'p01'**

$((C \bowtie O) \text{ where pid} = \text{'p01'}) [C.cname]$

Example 2.7.8

Customers: C(cid, cname, city, discnt)

Agents: A(aid, aname, city, percent)

Products: P(pid, pname, city, quantity, price)

Orders: O(ordno, month, cid, aid, pid, qty, dollars)

□ **Example 2.7.8: Get names of customers who order at least one product costing \$0.50.**

T1 := ((O ∞ (P where price=0.50)[pid]) ∞ C)[cname]

T2 := ((O ∞ P where price=0.50) ∞ C)[cname]

Question: T1 is equal to T2 ? Why ?

Customers: C(cid, cname, city, discnt)

(思考题)

Agents: A(aid, aname, city, percent)

Products: P(pid, pname, city, quantity, price)

Orders: O(ordno, month, cid, aid, pid, qty, dollars)

□ **Example 2.7.8: Get names of customers who order at least one product costing \$0.50.**

T1 := ((O \bowtie (P where price=0.50)[pid]) \bowtie C) [cname]

T2 := ((O \bowtie (P where price=0.50)) \bowtie C) [cname]

T3 := (((O \bowtie P) \bowtie C) where price=0.50) [cname]

T4 := (((O \bowtie P[pid]) \bowtie C) where price=0.50) [cname]

Question: Which is correct ?

2.7 Native Relational Operations

□ The **Product** and **Join** operations are *commutative* and *associative*.

1) $R \times S = S \times R$

2) $R \bowtie S = S \bowtie R$

3) $(R \times S) \times T = R \times (S \times T)$

4) $(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$

2.7 Native Relational Operations

□ Def 2.7.5 The Division Operation

✧ Consider two tables R and S, where

$$\text{Head}(S) \subset \text{Head}(R)$$

✧ Specifically, assume that

$$\text{Head}(R) = \{ A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m \}$$

$$\text{Head}(S) = \{ B_1, B_2, \dots, B_m \}$$

- The table T is the result of the division $R \div S$ if
 - $\text{Head}(T) = \{ A_1, A_2, \dots, A_n \}$
 - T contains the largest possible set of rows t

2.7 Native Relational Operations

□ Def 2.7.5 $T := R \div S$

⌘ $\text{Head}(R) = \{ A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m \}$

⌘ $\text{Head}(S) = \{ B_1, B_2, \dots, B_m \}$

⌘ $\text{Head}(T) = \{ A_1, A_2, \dots, A_n \}$

⌘ T contains the largest possible set of rows t such that :

for each row s in S , we can find one row r in R , where

① $t(A_i) = r(A_i)$ for $1 \leq i \leq n$, and

② $s(B_j) = r(B_j)$ for $1 \leq j \leq m$

Difference between join and division

	$T = R \bowtie S$	$T = R \div S$
Head(R)	$\{A_1, \dots, A_n, B_1, \dots, B_k\}$	$\{A_1, \dots, A_n, B_1, \dots, B_m\}$
Head(S)	$\{B_1, \dots, B_k, C_1, \dots, C_m\}$	$\{B_1, \dots, B_m\}$
Head(T)	$\{A_1, \dots, A_n, B_1, \dots, B_k, C_1, C_2, \dots, C_m\}$	$\{A_1, A_2, \dots, A_n\}$

<p>Result of</p> <p>$T = R \bowtie S$</p>	<p>A row t is in the table $R \bowtie S$ if and only if</p> <p><u>there are two rows u in R and v in S, such that $u[B_i] = v[B_i]$ for all i ($1 \leq i \leq k$);</u></p> <p>then</p> <p>$t[A_i] = u[A_i]$ for all i, $1 \leq i \leq n$</p> <p>$t[C_i] = v[C_i]$ for all i, $1 \leq i \leq m$</p> <p>$t[B_i] = u[B_i] = v[B_i]$ for all i, $1 \leq i \leq k$</p>
<p>Result of</p> <p>$T = R \div S$</p>	<p>T contains <u>the largest possible set</u> of rows t such that</p> <p><u>for each row s in S, we can find one row r in R, where</u></p> <p><u>$t(A_i) = r(A_i)$ for $1 \leq i \leq n$</u> and</p> <p><u>$s(B_j) = r(B_j)$ for $1 \leq j \leq m$</u></p>

For each row **u** in **R**

{

For each row **v** in **S**

{

If (**u**[B_i] = **v**[B_i] for all i ($1 \leq i \leq k$)) Then

{

Generates a row **t** of **T** such that

t[A_i] = **u**[A_i] for all i , $1 \leq i \leq n$

t[C_i] = **v**[C_i] for all i , $1 \leq i \leq m$

t[B_i] = **u**[B_i] = **v**[B_i] for all i , $1 \leq i \leq k$

}

}

}

Result of $T=R \div S$

1. Assume a row **t** is in **T**, then:

for each row **s** in **S**

{

we can find a row **r** in **R**, and

{

t(A_i) = **r**(A_i) for $1 \leq i \leq n$

s(B_j) = **r**(B_j) for $1 \leq j \leq m$

}

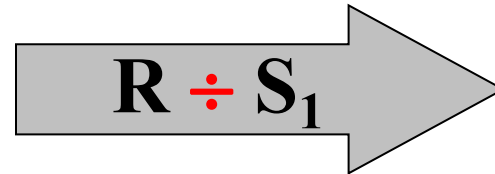
}

2. **T** contains the largest possible set of rows **t**

Example 2.7.9

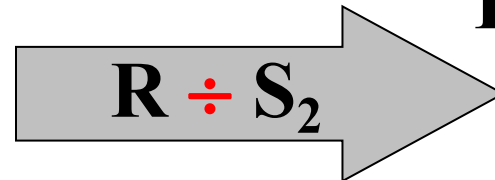
R		
A	B	C
a1	b1	c1
a2	b1	c1
a1	b2	c1
a1	b2	c2
a2	b1	c2
a1	b2	c3
a1	b2	c4
a1	b1	c5

S ₁	C
c1	



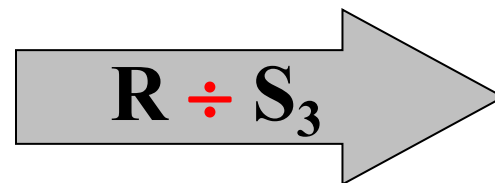
T ₁	A	B
	a1	b1
	a2	b1
	a1	b2

S ₂	C
c2	



T ₂	A	B
	a1	b2
	a2	b1

S ₃	C
c1	
c2	

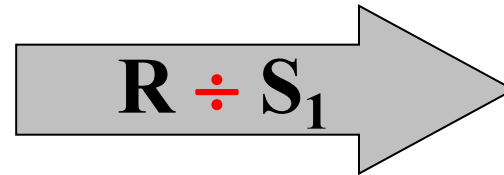


T ₃	A	B
	a1	b2
	a2	b1

Example 2.7.9 (cont.)

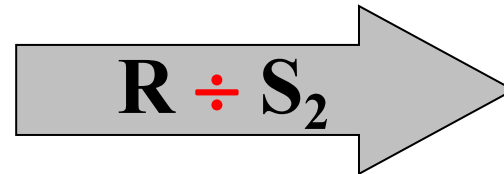
R		
A	B	C
a1	b1	c1
a2	b1	c1
a1	b2	c1
a1	b2	c2
a2	b1	c2
a1	b2	c3
a1	b2	c4
a1	b1	c5
a2	b2	c2

S ₁
C
c1



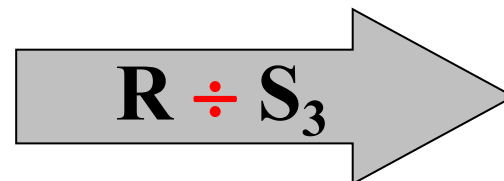
T ₁	
A	B
a1	b1
a2	b1
a1	b2

S ₂
C
c2



T ₂	
A	B
a1	b2
a2	b1
a2	b2

S ₃
C
c1
c2

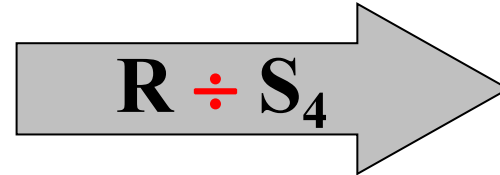


T ₃	
A	B
a1	b2
a2	b1

Example 2.7.9 (cont.)

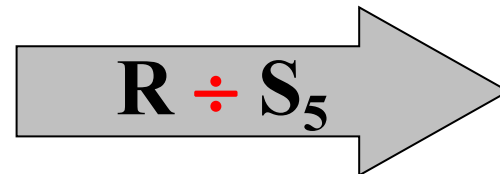
R		
A	B	C
a1	b1	c1
a2	b1	c1
a1	b2	c1
a1	b2	c2
a2	b1	c2
a1	b2	c3
a1	b2	c4
a1	b1	c5

S₄	C
	c1
	c2
	c3
	c4



T ₄	
A	B
a1?	b2

S ₅	C
	c1
	c2
	c3
	c4
	c5

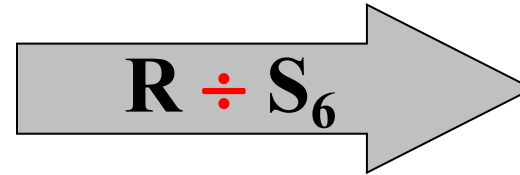


T ₅	
A	B
?	

Example 2.7.9 (cont.)

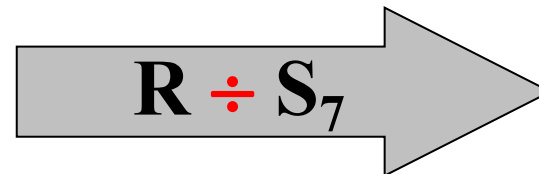
A	B	C
a1	b1	c1
a2	b1	c1
a1	b2	c1
a1	b2	c2
a2	b1	c2
a1	b2	c3
a1	b2	c4
a1	b1	c5

B	C
b1	c1



A
a1?
a2

B	C
b1	c1
b2	c1



A
a?

2.7 Native Relational Operations

□ Theorem 2.7.6

⌘ if $R = T \times S$, then

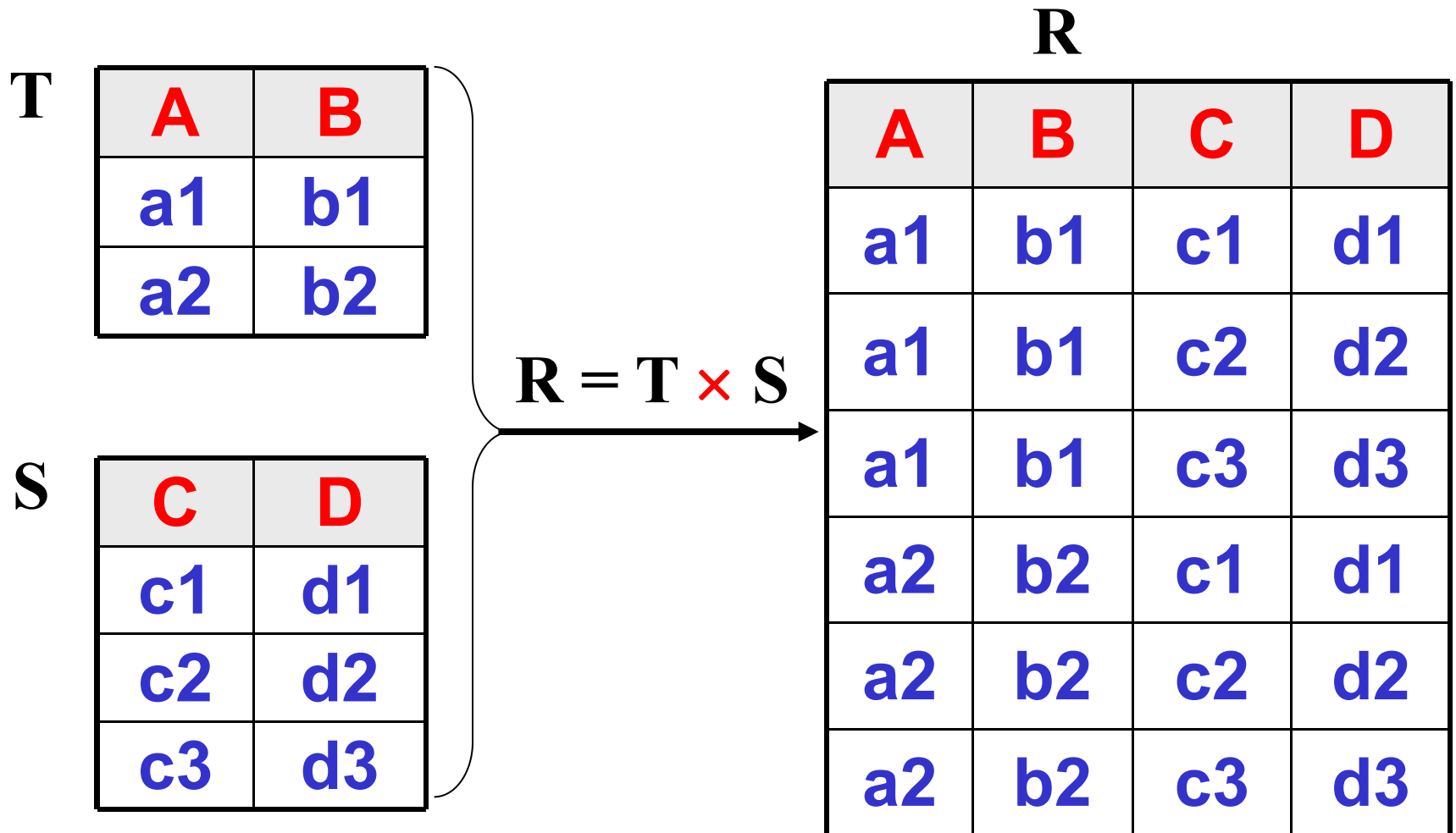
- $T = R \div S$

- $S = R \div T$

⌘ if $T = R \div S$, then

- $T \times S \subseteq R$

Example



Example

R

A	B	C	D
a1	b1	c1	d1
a1	b1	c2	d2
a1	b1	c3	d3
a1	b2	c1	d1
a2	b2	c1	d1
a2	b2	c2	d2
a2	b2	c3	d3
a2	b1	c2	d2

S

C	D
c1	d1
c2	d2
c3	d3

T = R ÷ S

A	B
a1	b1
a2	b2

R' = T × S

A	B	C	D
a1	b1	c1	d1
a1	b1	c2	d2
a1	b1	c3	d3
a2	b2	c1	d1
a2	b2	c2	d2
a2	b2	c3	d3

R' ⊆ R

2.7 Native Relational Operations

□ why we use division ?

✧ what kind of question does it answer about the data.

– Example

1. Get cids of customers who order products p01 ?
2. Get cids of customers who order products p01 and p02 ?
3. Get cids of customers who order all products ?

PRODUCTS (P)

<u>pid</u>	pname	...
p01	comb	...
p02	brush	...
p03	razor	...

ORDERS (O)

<u>ordno</u>	cid	aid	pid	...
1011	c001	...	p01	...
1012	c002	...	p01	...
1019	c001	...	p02	...
1017	c002	...	p02	...
1018	c004	...	p01	...
1023	c004	...	p02	...
1022	c002	...	p03	...
1025	c006	...	p03	...
1013	c003	...	p03	...

1. Get cids of customers who order products p01?
(O where pid='p01') [cid]

PRODUCTS (P)

<u>pid</u>	pname	...
p01	comb	...
p02	brush	...
p03	razor	...

ORDERS (O)

<u>ordno</u>	cid	aid	pid	...
1011	c001	...	p01	...
1012	c002	...	p01	...
1019	c001	...	p02	...
1017	c002	...	p02	...
1018	c004	...	p01	...
1023	c004	...	p02	...
1022	c002	...	p03	...
1025	c006	...	p03	...
1013	c003	...	p03	...

2. Get cids of customers who order products p01 and p02 ?

$(O \text{ where } pid='p01')[cid] \cap (O \text{ where } pid='p02')[cid]$

PRODUCTS (P)

<u>pid</u>	pname	...
p01	comb	...
p02	brush	...
p03	razor	...
⋮	⋮	⋮

ORDERS (O)

<u>ordno</u>	cid	aid	pid	...
1011	c001	...	p01	...
1012	c002	...	p01	...
1019	c001	...	p02	...
1017	c002	...	p02	...
1018	c004	...	p01	...
1023	c004	...	p02	...
1022	c002	...	p03	...
1025	c006	...	p03	...
1013	c003	...	p03	...
⋮	⋮	⋮	⋮	⋮

3. Get cids of customers who order *all products*?

2.7 Native Relational Operations

– Example 2.7.11

- Get names of customers who order all products ?

If a customer (*row c in CUSTOMERS*) order all products, then for each product (*row p in PRODUCTS*), we can find a row *o* in ORDERS such that

$$o.cid = c.cid \text{ and } o.pid = p.pid$$

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

PRODUCTS

<u>pid</u>	pname	city	price
p01	comb	Dallas	0.50
p02	brush	Newark	0.50
p03	razor	Duluth	1.00

ORDERS

<u>ordno</u>	month	cid	aid	pid	qty	dollar
1011	jan	c001	a01	p01	1000	450.0
1012	jan	c002	a01	p01	1000	440.0
1019	feb	c001	a02	p02	400	180.0
1017	feb	c002	a06	p02	600	264.0
1018	feb	c004	a03	p01	600	276.0
1023	mar	c004	a04	p02	1000	460.0
1022	mar	c002	a05	p03	400	352.0
1025	apr	c006	a05	p03	800	800.0
1013	jan	c003	a03	p03	1000	920.0

□ We can answer the request which use keyword 'all' by 'division' operation.

➤ **First:** get cids of customers who order all products.

$T := \text{ORDERS}[\text{cid}, \text{pid}] \div \text{PRODUCTS}[\text{pid}]$

1) **Why must project PRODUCTS on pid ?**

➤ columns of divisor must be subset of columns of dividend.

2) **Why must project ORDERS on cid, pid ?**

➤ We only try to find the value of cid in ORDERS the same for all pid.

➤ **Second:** get names of customers

$T_1 := ((T) \bowtie C) [\text{cname}]$

□ Get names of customers who order *all* products ?

➤ **First:** get cids of customers who order all products.

$$T := \text{ORDERS}[\text{cid}, \text{pid}] \div \text{PRODUCTS}[\text{pid}]$$

➤ **Second:** get names of customers

$$T_1 := ((T) \bowtie C) [\text{cname}]$$

合并之后得到如下的表示(T_1):

$$((\text{ORDERS}[\text{cid}, \text{pid}] \div \text{PRODUCTS}[\text{pid}]) \bowtie C)[\text{cname}]$$

□ **Question:** Can answer this request by writing?

$$T_2: \quad (O \bowtie C) [\text{cname}, \text{pid}] \div P[\text{pid}]$$

Example 2.7.11 (cont.)

❑ **Question:** Can answer this request by writing

$T_2 := (O \bowtie C) [cname, pid] \div P[pid] \quad ?$

CUSTOMERS

<u>cid</u>	cname	city	discnt
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

PRODUCTS

<u>pid</u>	pname	city	price
p01	comb	Dallas	0.50
p02	brush	Newark	0.50
p03	razor	Duluth	1.00

ORDERS

<u>ordno</u>	month	cid	aid	pid	qty	dollars
1011	jan	c001	a01	p01	1000	450.00
1012	jan	c002	a01	p01	1000	440.00
1019	feb	c001	a02	p02	400	180.00
1017	feb	c002	a06	p02	600	264.00
1018	feb	c004	a03	p01	600	276.00
1023	mar	c004	a04	p02	1000	460.00
1022	mar	c002	a05	p03	400	352.00
1025	apr	c006	a05	p03	800	800.00
1013	jan	c003	a03	p03	1000	920.00

$$T_1 := ((O[cid, pid] \div P[pid]) \bowtie C) [cname]$$

O[cid, pid]

cid	pid
c001	p01
c002	p01
c001	p02
c002	p02
c004	p01
c004	p02
c002	p03
c006	p03
c003	p03

÷

P[pid]

<u>pid</u>
p01
p02
p03

=

<u>cid</u>
c02

C

<u>cid</u>	cname	city	discont
c001	TipTop	Duluth	10.00
c002	Basics	Dallas	12.00
c003	Allied	Dallas	8.00
c004	ACME	Duluth	8.00
c006	ACME	Kyoto	0.00

O[cid, pid]

CUSTOMERS

 $(O_{\infty}C)[cname, pid]$

cid	pid
c001	p01
c002	p01
c001	p02
c002	p02
c004	p01
c004	p02
c002	p03
c006	p03
c003	p03

<u>cid</u>	cname	...
c001	TipTop	...
c002	Basics	...
c003	Allied	...
c004	ACME	...
c006	ACME	...

cname	pid
TipTop	p01
Basics	p01
TipTop	p02
Basics	p02
ACME	p01
ACME	p02
Basics	p03
ACME	p03
Allied	p03

$$T_2 := (O \infty C) [cname, pid] \div P[pid]$$

$(O \infty C)[cname, pid]$

cname	pid
TipTop	p01
Basics	p01
TipTop	p02
Basics	p02
ACME	p01
ACME	p02
Basics	p03
ACME	p03
Allied	p03

\div

$P[pid]$

<u>pid</u>
p01
p02
p03

$=$

$T_2[cname]$

<u>cname</u>
Basics?
ACME

□ Example 2.7.10

1) Get names of customers who order all products ordered by customer 'c006' ?

– Step 1: find all products ordered by customer 'c006'

$$R_1 := (\text{ORDERS where cid} = \text{'c006'})[\text{pid}]$$

– Step 2: find cid of customers who order all products ordered by customer 'c006'

$$R_2 := \text{ORDERS [cid, pid]} \div R_1$$

– Step 3: find cnames of customers

$$R_3 := (R_2 \bowtie C) [\text{cname}]$$

□ Example 2.7.10

- 2) Get pid of products ordered through all agents ?
 - 3) Get name of products ordered by all customers who live in Dallas ?
-

2) answer

$\text{ORDERS [aid, pid] } \div \text{AGENTS [aid]}$

3) answer

$R_1 := (\text{CUSTOMERS where city='Dallas'})[\text{cid}]$

$R_2 := \text{ORDERS [cid, pid] } \div R_1$

$R_3 := (\text{PRODUCTS } \bowtie R_2) [\text{pname}]$

□ Example 2.7.10

- 4) Get cids of customers who order all products priced at \$0.50.
- 5) Get cids of customers who order all products that anybody orders.

4) answer

$R_1 := (\text{PRODUCTS where price} = 0.50) [\text{pid}]$

$R_2 := \text{ORDERS} [\text{cid}, \text{pid}] \div R_1$

5) answer

$\text{ORDERS} [\text{cid}, \text{pid}] \div \text{ORDERS} [\text{pid}]$

Example of Operations of Relational Algebra

□ When & How to use

⌘ project / select

⌘ union / intersection / minus

⌘ product / join / division

□ Difference

- between product and join ?
- between join and division ?
- between minus and 'not equal' ?

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 1. Get aids of agents who do not supply product p02.

$A[aid] - (O \text{ where } pid = 'p02')[aid]$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 2. Get aids of agents who supply only product p02.

$O[aid] \text{ — } (O \text{ where } pid \neq 'p02')[aid]$

$? A[aid] \text{ — } (O \text{ where } pid \neq 'p02')[aid] \quad \times$

Review of exp1 & exp2

Exp 1. Get aids of agents who do not supply product p02.

$A[aid] \text{ — } (O \text{ where pid } = 'p02')[aid]$

Exp 2. Get aids of agents who supply only product p02.

$O[aid] \text{ — } (O \text{ where pid } \neq 'p02')[aid]$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 3. Get aids of agents who take orders on at least that set of products ordered by c004.

$O[aid, pid] \div (O \text{ where } cid = 'c004')[pid]$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 4. Get cids of customers who order p01 and p07.

$(O \text{ where } pid='p01')[cid] \cap (O \text{ where } pid='p07')[cid]$

? $(O \text{ where } pid = 'p01' \text{ and } pid = 'p07')[cid]$ ✗

$((O \text{ where } pid='p01') \cap (O \text{ where } pid='p07'))[cid]$ ✗

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 5. Get cids of customers who order p01 or p07.

□ Answer 1:

$(O \text{ where } pid='p01')[cid] \cup (O \text{ where } pid='p07')[cid]$

□ Answer 2:

$(O \text{ where } pid = 'p01' \text{ or } pid = 'p07')[cid]$

□ Answer 3:

$((O \text{ where } pid='p01') \cup (O \text{ where } pid='p07')) [cid]$

Review of exp4 & exp5

Exp 4. Get cids of customers who order p01 and p07.

$(O \text{ where pid='p01'})[cid] \cap (O \text{ where pid='p07'})[cid]$

? $(O \text{ where pid = 'p01' and pid = 'p07'})[cid]$ ✗

$((O \text{ where pid='p01'}) \cap (O \text{ where pid='p07'}))[cid]$ ✗

Exp 5. Get cids of customers who order p01 or p07.

❑ Answer 1:

$(O \text{ where pid='p01'})[cid] \cup (O \text{ where pid='p07'})[cid]$

❑ Answer 2:

$(O \text{ where pid = 'p01' or pid = 'p07'})[cid]$

❑ Answer 3:

$((O \text{ where pid='p01'}) \cup (O \text{ where pid='p07'})) [cid]$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 6. List all cities inhabited by customers who order product p02 or agents who place an order for p02.

$T_1 := ((O \text{ where } pid = 'p02') \text{ JOIN } C)[city]$

$T_2 := ((O \text{ where } pid = 'p02') \text{ JOIN } A)[city]$

$T := T_1 \text{ UNION } T_2$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 7. Get aids of agents who place an order for at least one customer that uses product p01.

((O where pid = 'p01')[cid] JOIN O) [aid]

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 8. Get aids of agents who place orders for all customers that uses product p01.

$O[cid, aid] \div (O \text{ where } pid = 'p01')[cid]$

请注意与前一个例子 Exp 7 的比较！

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 9. Retrieve product ids for all products that are not ordered by any customers living in a city beginning with the letter "D".

1) $T_1 := C \text{ where } city \geq 'D' \text{ and } city < 'E'$

2) $T_2 := P [pid] - (O \text{ JOIN } T_1) [pid]$

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

Exp 10. Retrieve cids of customers with the largest discounts.

□ Answer 1

1) **CY := C**

2) **T₁ := ((CY × C) where CY.discnt > C.discnt)[C.cid]**

3) **T₂ := C[cid] – T₁**

Example of Operations of Relational Algebra

Customers: **C**(cid, cname, city, discnt)

Exp 10. Retrieve cids of customers with the largest discounts.

□ **Answer 2 (by division)**

1) **CY := C**

2) **T₁(cyid, cid) :=**
((CY × C) where CY.discnt ≥ C.discnt)
[CY.cid, C.cid]

3) **T₂ := T₁ ÷ C[cid]**

不能少了 =

Exp 10. Retrieve cids of customers with the largest discounts.

❑ **Answer 1:** $CY := C$

1) $T_1 := ((CY \times C) \text{ where } CY.\text{discnt} > C.\text{discnt})[C.\text{cid}]$

2) $T_2 := C[\text{cid}] - T_1$

❑ **Answer 2:** $CY := C$

1) $T_1(\text{cyid}, \text{cid}) :=$
 $((CY \times C) \text{ where } CY.\text{discnt} \geq C.\text{discnt})$
 $[CY.\text{cid}, C.\text{cid}]$

1) $T_2 := T_1 \div C[\text{cid}]$

2.8 The Interdependence of Operations

□ **Theorem 2.8.1:** $A \cap B = A - (A - B)$

□ **Theorem 2.8.2**

$\text{Head}(R) = \{ A_1, \dots, A_n, B_1, \dots, B_k \}$

$\text{Head}(S) = \{ B_1, \dots, B_k, C_1, \dots, C_m \}$

$T_1 := R \times S$

$T_2 := (T_1)$ where $R.B_1 = S.B_1$ and ... and $R.B_k = S.B_k$

$T_3 := T_2[R.A_1, \dots, R.A_n, R.B_1, \dots, R.B_k, S.C_1, \dots, S.C_m]$

Then: $T_3 = R \bowtie S$

2.8 The Interdependence of Operations

□ Theorem 2.8.3

If $\text{Head}(R) = \{ A_1 \dots A_n B_1 \dots B_m \}$ and

$\text{Head}(S) = \{ B_1 \dots B_m \}$

$$R \div S = R[A_1 \dots A_n] - ((R[A_1 \dots A_n] \times S) - R) [A_1 \dots A_n]$$

1) $T_1 := R[A_1, \dots, A_n]$

2) $T_2 := T_1 \times S$

3) $T_3 := T_2 - R$

4) $T_4 := T_3 [A_1, \dots, A_n]$

5) $R \div S := T_1 - T_4$

‘除’运算的推导过程

1) $T_{\max} := \pi_{A_1 \dots A_n} (R)$

// T_{\max} 是最大可能的结果元组集合

2) $R_{\max} := T_{\max} \times S$

// R_{\max} 与关系 R 是同类关系

3) $T_1 := R_{\max} - R$

4) $T_2 := \pi_{A_1 \dots A_n} (T_1)$

// T_2 是关系 T_{\max} 中不满足除运算的结果要求的那些元组

// 理由：对于关系 T_2 中的一个元组 q ，至少能在关系 S 中找到一个元组 s ，使得由元组 q 和 s 所构成的元组 (q, s) 不在关系 R 中出现。

5) $R \div S := T_{\max} - T_2$

2.8 The Interdependence of Operations

- The set of basic relational operations
(union, difference, product, selection, projection)
- Relationally Complete

2.9 Illustrative Examples

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

- Exp 2.9.1: Get the names of customers who order at least one product priced at \$0.50.

$((P \text{ where price}=0.50)[pid] \bowtie O) \bowtie C) [cname]$

注意：Customers和Products表中都有同名的city属性，要防止它们对本次查询的干扰！

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

□ Exp 2.9.2: Find cids of all customers who don't place any order through agent a03.

在下述两个表达式中，请选出能够正确表达本次查询需求的那一个。

1) **C[cid] – (O where aid = 'a03') [cid]** ✓

2) **O[cid] – (O where aid = 'a03') [cid]** ✗

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

□ Exp 2.9.3: Retrieve customers who place orders only through agent a03.

在下述两个表达式中，请选出能够正确表达本次查询需求的那一个。

1) **O[cid] – (O where aid <> ‘a03’) [cid]** ✓

2) **C[cid] – (O where aid <> ‘a03’) [cid]** ✗

Customers: **C**(cid, cname, city, discont)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

□ Exp 2.9.4: Find products that have never been ordered by a customer based in New York through an agent based in Boston.

1) $T_1 := (C \text{ where city} = \text{'New York'}) [cid]$

2) $T_2 := (A \text{ where city} = \text{'Boston'}) [aid]$

3) $T_3 := (((T_1 \text{ join } O) \text{ join } T_2)$

4) $T_4 := P[pid] - T_3[pid]$

注意：中间每一步投影运算的重要性！

Customers: **C**(cid, cname, city, discnt)

Agents: **A**(aid, aname, city, percent)

Products: **P**(pid, pname, city, quantity, price)

Orders: **O**(ordno, month, cid, aid, pid, qty, dollars)

□ Exp 2.9.5: Get names of customers who order all products priced at \$0.50.

1) $T_1 := P \text{ where price} = 0.50$

2) $T_2 := O [cid, pid] \div T_1 [pid]$

3) $T_3 := (T_2 \text{ join } C) [cname]$

注意： 第二步‘除法’表达式的正确写法！

一组运用‘除法’运算的例子

- Exp 2.9.6: Get cids of customers who order all products that anybody orders.

$$O[cid, pid] \div O[pid]$$

- Exp 2.9.7: Get aids of agents who take orders on at least that set of products ordered by c004.

$$O[aid, pid] \div (O \text{ where } cid = 'c004')[pid]$$

□ Exp 2.9.8: Get cids of customers who order both products p01 and p07.

a) Answer 1:

① $T_1 := (O \text{ where } pid='p01') [cid]$

② $T_2 := (O \text{ where } pid='p07') [cid]$

③ $T_3 := T_1 \cap T_2$

b) Answer 2: (by division)

① $T_1 := P \text{ where } (pid='p01' \text{ or } pid='p07')$

② $T_2 := O [cid, pid] \div T_1 [pid]$

一组运用‘联接’运算的例子

□ Exp 2.9.9: Get cids of customers who place an order through at least one agent who places an order for product p03.

1) $T_1 := (O \text{ where pid} = \text{'p03'})[\text{aid}]$

2) $T_2 := (T_1 \bowtie O) [\text{cid}]$

□ Exp 2.9.10: Get cids of all customers who have the same discount as any customer in Dallas or Boston.

1) $T_1 := C \text{ where city} = \text{'Dallas'} \text{ or city} = \text{'Boston'}$

2) $T_2 := (T_1 [\text{discnt}] \text{ join } C) [\text{cid}]$

□ Exp 2.9.11: List pids of products that are ordered through agents who place orders for (possibly different) customers who order at least one product from an agent who has placed an order for customer c001.

- 1) Find aids of agents who has placed an order for customer c001: $T_1 := (O \text{ where } cid = 'c001')[aid]$
- 2) Find cids of customers who order at least one product from an agent in T_1 : $T_2 := (T_1 \bowtie O) [cid]$
- 3) Find aids of agents who place orders for customers in T_2 : $T_3 := (T_2 \bowtie O) [aid]$
- 4) Find pids of products that are ordered through agents in T_3 : $T_4 := (T_3 \bowtie O) [pid]$

2.9 Illustrative Examples

□ Exp 2.9.12: Get pids of products not ordered by any customer living in a city whose name begin with the letter D.

① $T_1 := (C \text{ where } \text{city} \geq 'D' \text{ and } \text{city} < 'E') [\text{cid}]$

② $T_2 := (T_1 \bowtie O) [\text{pid}]$

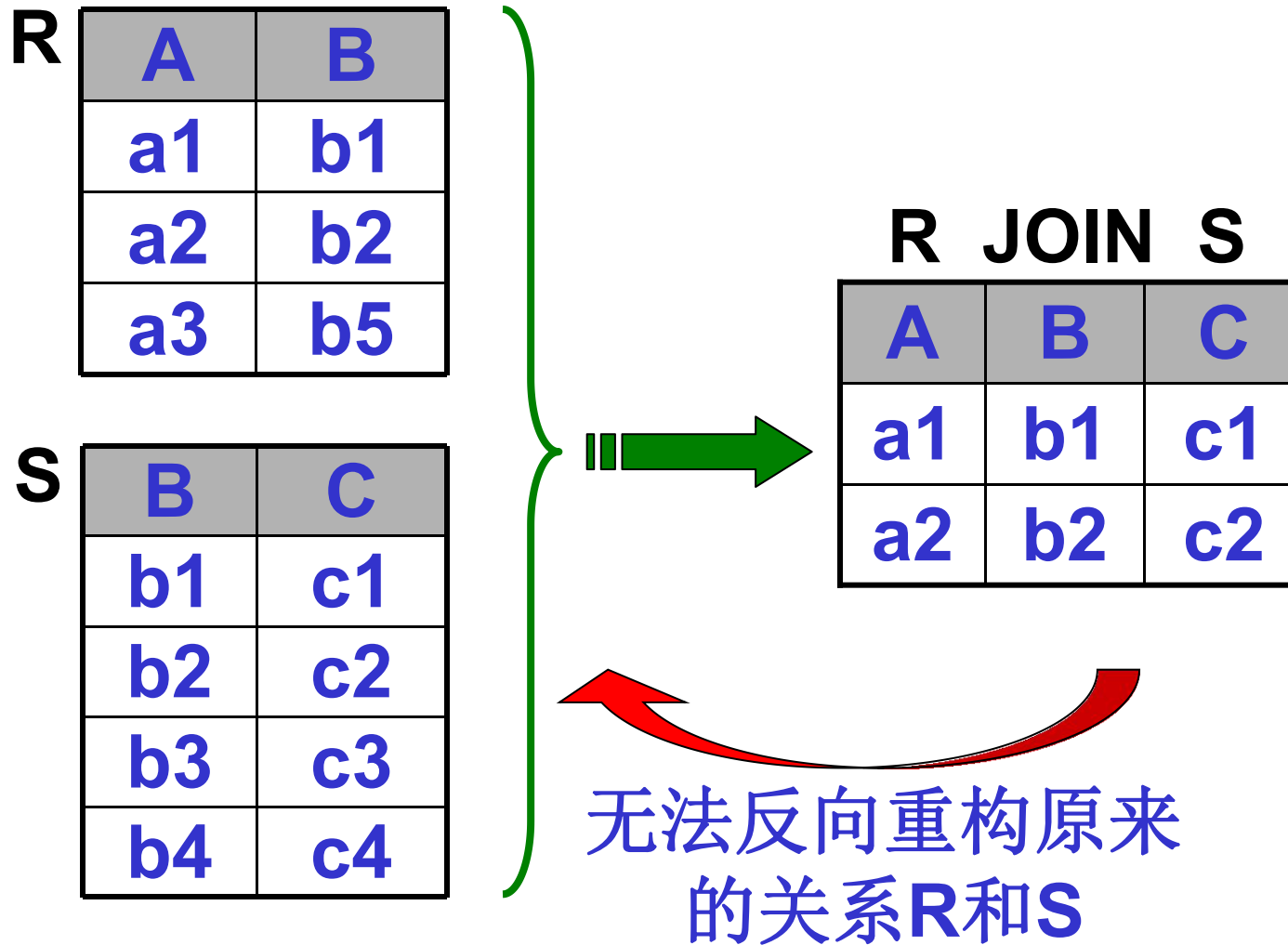
③ $T_3 := P [\text{pid}] - T_2$

2.10 Other Relational Operations

- ❑ **OUTER JOIN**（外联接）
- ❑ **LEFT OUTER JOIN**（左外联接）
- ❑ **RIGHT OUTER JOIN**（右外联接）

- ❑ **THETA JOIN**（ θ -联接）

2.10 Other Relational Operations



反向重构关系R: $(R \text{ left outer join } S) [A, B]$

R

A	B
a1	b1
a2	b2
a3	b5

S

B	C
b1	c1
b2	c2
b3	c3
b4	c4

R LEFT OUTER JOIN S

A	B	C
a1	b1	c1
a2	b2	c2
a3	b5	null

R RIGHT OUTER JOIN S

A	B	C
a1	b1	c1
a2	b2	c2
null	b3	c3
null	b4	c4

反向重构关系S: $(R \text{ right outer join } S) [B, C]$

$T = R \text{ Left Outer Join } S$

For each row u in R {

Found = false;

For each row v in S {

If ($u[B_i] = v[B_i]$ for all i ($1 \leq i \leq k$)) Then {

Found = true;

Generates a row t of T such that:

$t[A_i] = u[A_i]$ for all $1 \leq i \leq n$

$t[B_i] = u[B_i] = v[B_i]$ for all $1 \leq i \leq k$

$t[C_i] = v[C_i]$ for all $1 \leq i \leq m$

生成自然联接
的结果元组

}

}

If (Found == false) Then {

Generates a row t of T such that:

$t[A_i] = u[A_i]$ for all $1 \leq i \leq n$

$t[B_i] = u[B_i]$ for all $1 \leq i \leq k$

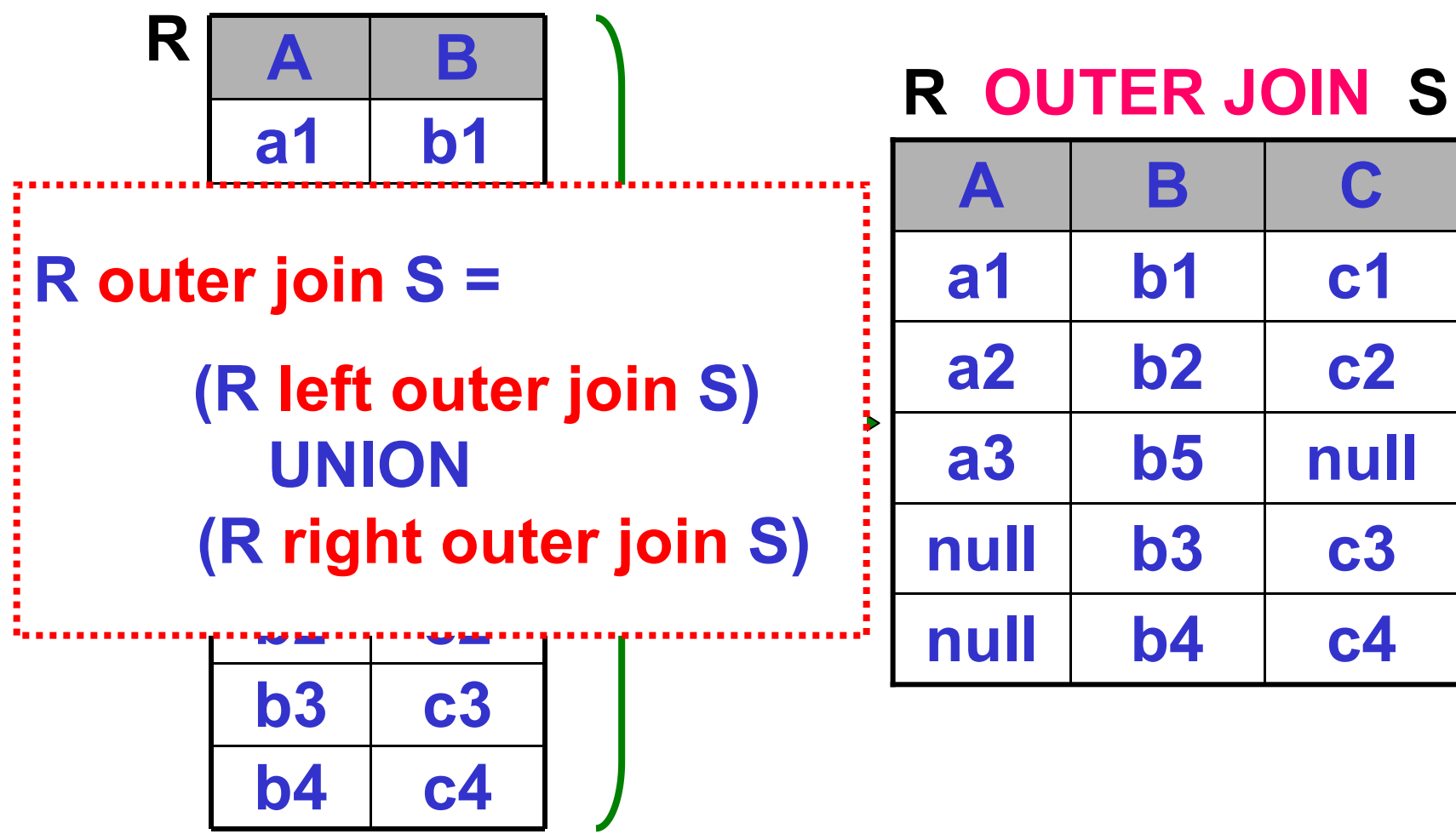
$t[C_i] = \text{NULL}$ for all $1 \leq i \leq m$

针对未参与自然
联接的元组 u ,
生成对应的外联
接结果元组

}

}

反向重构关系R: ((R outer join S) where A<>null)[A, B]



反向重构关系S: ((R outer join S) where C<>null)[B, C]

2.10 Other Relational Operations

R

A	B
a1	b1
a2	b2
a3	b5

S

B	C
b1	c1
b2	c2
b3	c3
b4	c4

R JOIN S

A	B	C
a1	b1	c1
a2	b2	c2

R OUTER JOIN S

A	B	C
a1	b1	c1
a2	b2	c2
a3	b5	null
null	b3	c3
null	b4	c4

R LEFT OUTER
JOIN S

A	B	C
a1	b1	c1
a2	b2	c2
a3	b5	null

R RIGHT OUTER
JOIN S

A	B	C
a1	b1	c1
a2	b2	c2
null	b3	c3
null	b4	c4

2.10 Other Relational Operations

□ Theta Join

$$R \bowtie_F S = (R \times S) \text{ where } F$$

- **Example 2.10.1:** Find all ordno values for orders whose order quantity exceeds the current quantity on hand for the product.

$$(O \bowtie_{O.pid = P.pid \text{ and } O.qty > P.quantity} P) [\text{ordno}]$$

Review of Operations of Relational Algebra

R

A	B	C	D
1	2	3	4
2	2	5	7
9	0	3	8

S

A	B	C	D
2	2	3	8
1	2	3	4
9	1	2	3

$R \cap S$

A	B	C	D
1	2	3	4

$R \cup S$

A	B	C	D
1	2	3	4
2	2	5	7
9	0	3	8
2	2	3	8
9	1	2	3

$R - S$

A	B	C	D
2	2	5	7
9	0	3	8

Review of Operations of Relational Algebra

$R \times S$

R.A	R.B	R.C	R.D	S.A	S.B	S.C	S.D
1	2	3	4	2	2	3	8
1	2	3	4	1	2	3	4
1	2	3	4	9	1	2	3
2	2	5	7	2	2	3	8
2	2	5	7	1	2	3	4
2	2	5	7	9	1	2	3
9	0	3	8	2	2	3	8
9	0	3	8	1	2	3	4
9	0	3	8	9	1	2	3

Review of Operations of Relational Algebra

R

A	B	C	D
1	2	3	4
1	4	8	3
2	4	2	6
1	1	4	7

S

D	E
5	1
6	4
7	3
6	8

$R \bowtie S$

A	B	C	D	E
2	4	2	6	4
2	4	2	6	8
1	1	4	7	3

Review of Operations of Relational Algebra

R

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

S_1

C	D
3	4
5	6

$R \div S_1$

A	B
1	2
7	8

S_2

C	D
3	4

$R \div S_2$

A	B
1	2
7	8

S_3

C	D
3	4
5	6
4	2

$R \div S_3$

A	B
1	2

Exercise

□ 课后练习

教材: 2.1, 2.2, 2.4, 2.5