CS2102 Lecture 2
Relational Algebra

Relational Algebra

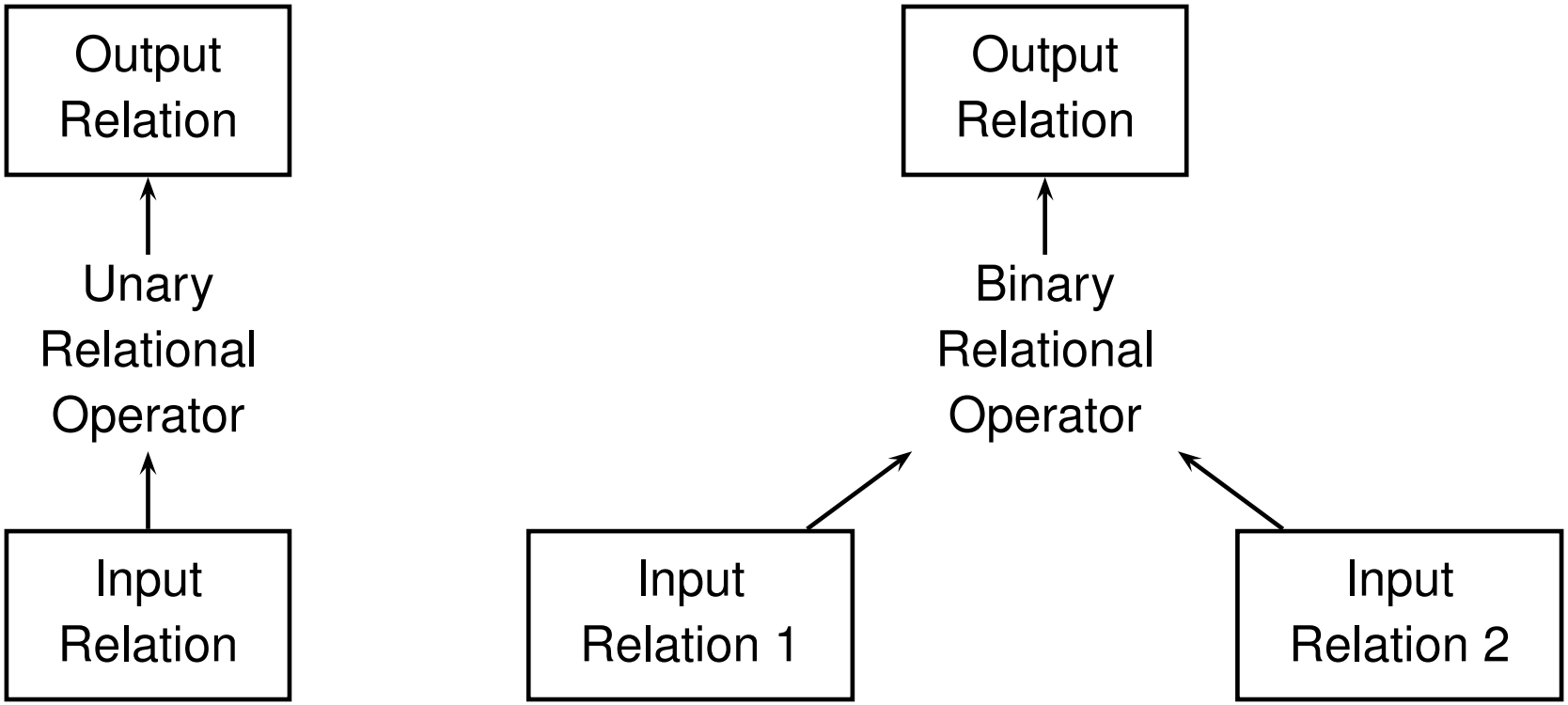
Stand firm in your refusal to remain conscious during algebra. In real life, I assure you, there is no such thing as algebra.

– Fran Lebowitz, Social Studies

- A formal language for asking queries on relations
- A query is composed of a collection of operators called relational operators
- Each operator takes one/two relations as input and computes an output relation
- Basic relational algebra operators
  - **Unary operators**: selection σ, projection π, renaming ρ
  - **Binary operators**: cross-product $\times$, union $\cup$, intersect $\cap$, difference $-$

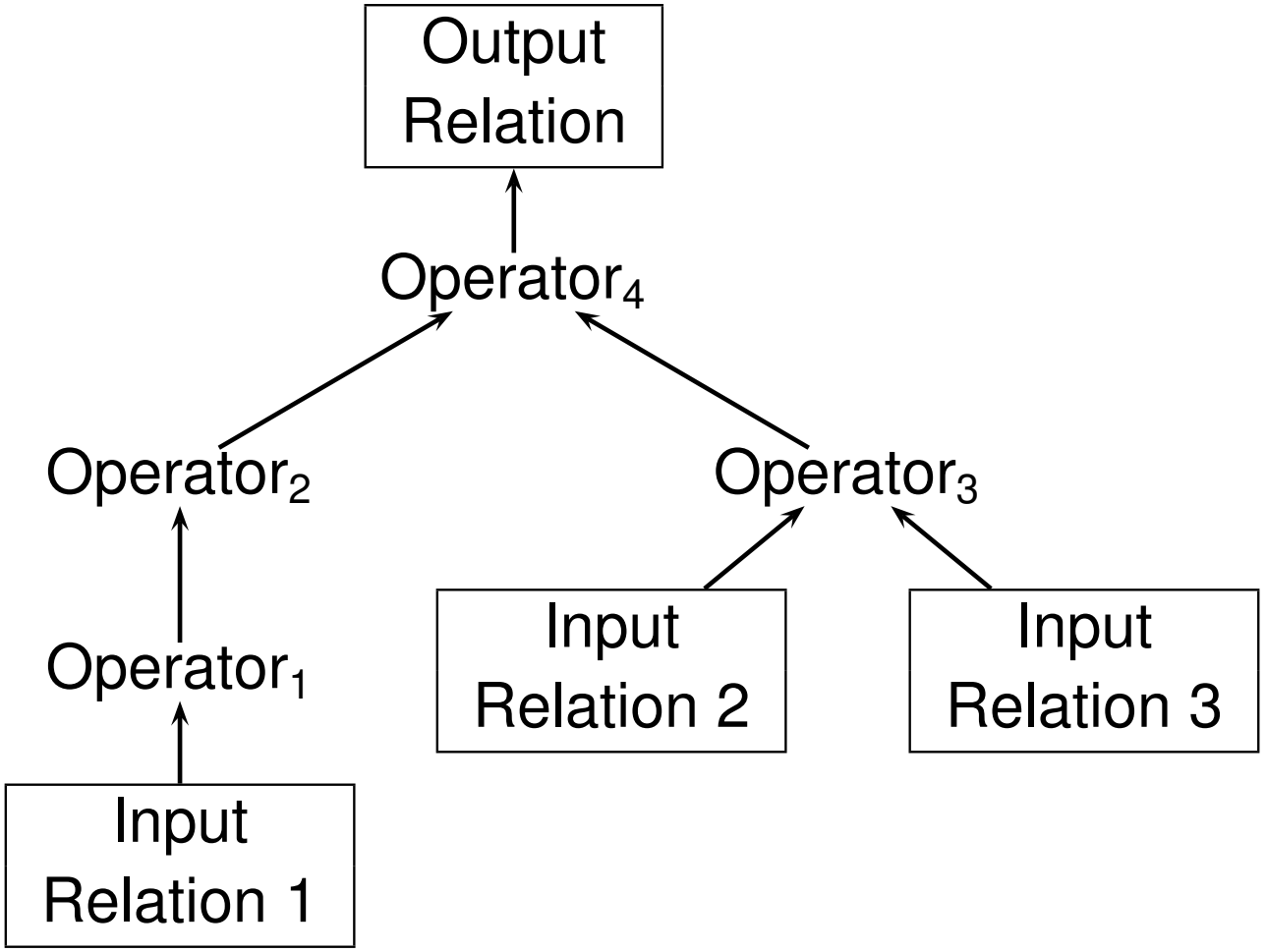
Closure Property

- Relations are *closed* under relational operators



Closure Property (cont.)

- Operators can be *composed* to form relational algebra expressions



Example Database

- Consider a database consisting of the following 6 relations:

Pizzas (pizza)
Contains (pizza, ingredient)
Restaurants (rname, area)
Sells (rname, pizza, price)
Customers (cname, area)
Likes (cname, pizza)

- Primary Key Constraints: primary keys are shown underlined
- Foreign Key Constraints:
  - Contains.pizza is a foreign key that refers to Pizzas.pizza
  - Sells.rname is a foreign key that refers to Restaurants.rname
  - Sells.pizza is a foreign key that refers to Pizzas.pizza
  - Likes.cname is a foreign key that refers to Customers.cname
  - Likes.pizza is a foreign key that refers to Pizzas.pizza

Example Database (cont.)

Pizzas

pizza
Diavola
Funghi
Hawaiian
Margherita
Marinara
Siciliana

Customers

cname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North

Restaurants

rname	area
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Contains

pizza	ingredient
Diavola	cheese
Diavola	chilli
Diavola	salami
Funghi	ham
Funghi	mushroom
Hawaiian	ham
Hawaiian	pineapple
Margherita	cheese
Margherita	tomato
Marinara	seafood
Siciliana	anchovies
Siciliana	capers
Siciliana	cheese

Sells

rname	pizza	price
Corleone Corner	Diavola	24
Corleone Corner	Hawaiian	25
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Lorenzo Tavern	Funghi	23
Mamma's Place	Marinara	22
Pizza King	Diavola	17
Pizza King	Hawaiian	21

Likes

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola

Selection: σ_c

- $\sigma_c(R)$ selects tuples from relation R that satisfy the *selection condition* c
  - For each $t \in R$, $t \in \sigma_c(R)$ iff c evaluates to **true** on t
  - The output table has the same schema as R
- **Example:** Find all restaurants, the pizzas that they sell and their prices, where the price is under $20

Sells

rname	pizza	price
Corleone Corner	Diavola	24
Corleone Corner	Hawaiian	25
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Lorenzo Tavern	Funghi	23
Mamma's Place	Marinara	22
Pizza King	Diavola	17
Pizza King	Hawaiian	21

$\sigma_{\text{price}<20}(\text{Sells})$

rname	pizza	price
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Pizza King	Diavola	17

Selection Conditions

- Selection condition is a boolean combination of *terms*
- A term is one of the following forms:
  1. attribute **op** constant
  2. attribute_1 **op** attribute_2
  3. term_1 **and** term_2
  4. term_1 **or** term_2
  5. **not** term_1
  6. (term_1)

$$\textbf{op} \in \{=, <>, <, \leq, >, \geq\}$$

- Operator precedence: (), *op*, not, and, or
- **Example**:

rname <> 'Pizza King' **or** price > 10 **and** pizza = 'Funghi'

is equivalent to

(rname <> 'Pizza King') **or** ((price > 10) **and** (pizza = 'Funghi'))

Selection Conditions (cont.)

- **Example:** Find all restaurants, the pizzas that they sell and their prices, where (1) the price is under $20 or the pizza is 'Marinara', and (2) the pizza is not 'Diavola'

Sells

rname	pizza	price
Corleone Corner	Diavola	24
Corleone Corner	Hawaiian	25
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Lorenzo Tavern	Funghi	23
Mamma's Place	Marinara	22
Pizza King	Diavola	17
Pizza King	Hawaiian	21

$\sigma_{(\text{price}<20 \textit{ or } \text{pizza}=\text{'Marinara'}) \textit{ and } \text{pizza}<>\text{'Diavola'}}(\text{Sells})$

rname	pizza	price
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Mamma's Place	Marinara	22

Selection Conditions with Null Values

- Result of a comparison operation involving *null* value is *unknown*
- Result of an arithmetic operation involving *null* value is *null*
- Example: Assume that the value of *x* is *null*
  - $x < 100$ evaluates to *unknown*
  - $x = null$ evaluates to *unknown*
  - $x <> null$ evaluates to *unknown*
  - $x + 20$ evaluates to *null*

Selection Conditions with Null Values (cont.)

- Three-valued logic system: *true*, *false*, & *unknown*

x	y	x AND y	x OR y	NOT x
FALSE	FALSE	FALSE	FALSE	
FALSE	UNKNOWN	FALSE	UNKNOWN	TRUE
FALSE	TRUE	FALSE	TRUE	
UNKNOWN	FALSE	FALSE	UNKNOWN	
UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	TRUE	
TRUE	FALSE	FALSE	TRUE	
TRUE	UNKNOWN	UNKNOWN	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE	

- For each $t \in R$, $t \in \sigma_c(R)$ iff c evaluates to **true** on t

Projection: π_ℓ

- $\pi_\ell(R)$ projects attributes given by a list ℓ of attributes from relation R
  - Each attribute in ℓ must be an attribute in R
  - The schema of the output table is determined by ℓ
- **Example:** Find all restaurants and the pizzas that they sell

Sells

rname	pizza	price
Corleone Corner	Diavola	24
Corleone Corner	Hawaiian	25
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Lorenzo Tavern	Funghi	23
Mamma's Place	Marinara	22
Pizza King	Diavola	17
Pizza King	Hawaiian	21

$\pi_{\text{rname,pizza}}(\text{Sells})$

rname	pizza
Corleone Corner	Diavola
Corleone Corner	Hawaiian
Corleone Corner	Margherita
Gambino Oven	Siciliana
Lorenzo Tavern	Funghi
Mamma's Place	Marinara
Pizza King	Diavola
Pizza King	Hawaiian

Projection: π_ℓ (cont.)

- Duplicate records are removed in the output relation
- **Example**: Find all pizza ingredients

Contains

pizza	ingredient
Diavola	cheese
Diavola	chilli
Diavola	salami
Funghi	ham
Funghi	mushroom
Hawaiian	ham
Hawaiian	pineapple
Margherita	cheese
Margherita	tomato
Marinara	seafood
Siciliana	anchovies
Siciliana	capers
Siciliana	cheese

$\pi_{\text{ingredient}}$(Contains)

ingredient
cheese
chilli
salami
ham
mushroom
pineapple
tomato
seafood
anchovies
capers

Renaming: ρ_ℓ

- $\rho_\ell(R)$ renames the attributes in R based on a list of attribute renamings ℓ
  - The schema of the output table is the same as R except that some attributes are renamed based on ℓ
- ℓ is a list of **attribute renamings** of the form $a_1 : b_1, \cdots, a_n : b_n$
  - Each renaming a_i:b_i renames attribute a_i to b_i
  - Each a_i must be an attribute in R
  - Each attribute in R can be renamed at most once
  - The order of the attribute renamings in ℓ does not matter
- Given $R(A, B, C)$, $\rho_{A:X,C:Z}(R)$ outputs a table with schema (X, B, Z)

Restaurants

rname	area
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

$\rho_{\text{area:region,rname:sname}}$(Restaurants)

sname	region
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Set Operators

- Union: $R \cup S$ returns a relation containing all tuples that occur in R or S (or both)
- Intersection: $R \cap S$ returns a relation containing all tuples that occur in both R and S
- Set-difference: $R - S$ returns a relation containing all tuples in R but not in S
- Union ($\cup$), intersection ($\cap$), and set-difference ($-$) operators require input relations to be union compatible

Union Compatibility

- Two relations are union compatible if
  1. they have the same number of attributes, and
  2. the corresponding attributes have the same domains
- Union compatible relations do not necessarily use the same attribute names

Union Compatibility (cont.)

- **Example**: Consider the following database:

  Student (sid:**integer**, dob:**date**, name:**text**)
  GradStudent (sid:**integer**, name:**text**)
  Report (reportid:**integer**, title:**text**, pubdate:**date**)

- Which of the following are valid set operations?
  1. Student $\cup$ GradStudent
  2. $\pi_{\text{sid,name}}$(Student) $\cup$ GradStudent
  3. Student $\cap$ Report
  4. $\pi_{\text{sid,name,dob}}$(Student) $-$ Report

Union Compatibility (cont.)

- The schema of the output of "R op S", where $op \in \{\cup, \cap, -\}$, is identical to the schema of R
- Assume **Restaurants**(rname:string, area:string) &

  Customers(cname:string, area:string)

Restaurants

rname	area
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Customers

cname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North

Customers ∪ Restaurants

cname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Restaurants ∪ Customers

rname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Set Operators: Examples

- **Example 1**: Find all pizzas that contain both cheese and chilli

Contains

pizza	ingredient
Diavola	cheese
Diavola	chilli
Diavola	salami
Funghi	ham
Funghi	mushroom
Hawaiian	ham
Hawaiian	pineapple
Margherita	cheese
Margherita	tomato
Marinara	seafood
Siciliana	anchovies
Siciliana	capers
Siciliana	cheese

$\pi_{pizza}(\sigma_{ingredient='cheese'}(\text{Contains}))$

pizza
Diavola
Margherita
Siciliana

$\pi_{pizza}(\sigma_{ingredient='chilli'}(\text{Contains}))$

pizza
Diavola

$\pi_{pizza}(\sigma_{ingredient='cheese'}(\text{Contains})) \cap \pi_{pizza}(\sigma_{ingredient='chilli'}(\text{Contains}))$

pizza
Diavola

Set Operators: Examples (cont.)

- **Example 2**: Find all pizzas that contain cheese but not chilli

Contains

pizza	ingredient
Diavola	cheese
Diavola	chilli
Diavola	salami
Funghi	ham
Funghi	mushroom
Hawaiian	ham
Hawaiian	pineapple
Margherita	cheese
Margherita	tomato
Marinara	seafood
Siciliana	anchovies
Siciliana	capers
Siciliana	cheese

$\pi_{pizza}(\sigma_{ingredient='cheese'}(\text{Contains}))$

pizza
Diavola
Margherita
Siciliana

$\pi_{pizza}(\sigma_{ingredient='chilli'}(\text{Contains}))$

pizza
Diavola

$\pi_{pizza}(\sigma_{ingredient='cheese'}(\text{Contains})) - \pi_{pizza}(\sigma_{ingredient='chilli'}(\text{Contains}))$

pizza
Margherita
Siciliana

Cross-Product: $\times$

- Consider relations $R(A, B, C)$ and $S(X, Y)$
- Cross-product: $R \times S$ outputs a relation with schema (A, B, C, X, Y) defined as follows:

$$R \times S = \{(a, b, c, x, y) \mid (a, b, c) \in R,\ (x, y) \in S\}$$

- Cross-product operation is also known as cartesian product

Cross-Product: Example

- Find all customer-restaurant pairs that are located in the central area

Customers

cname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North

Restaurants

rname	area
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

$\pi_{cname}(\sigma_{area='Central'}(\text{Customers}))$

cname
Moe
Ralph

$\pi_{rname}(\sigma_{area='Central'}(\text{Restaurants}))$

rname
Gambino Oven
Lorenzo Tavern

$\pi_{cname}(\sigma_{area='Central'}(\text{Customers})) \times \pi_{rname}(\sigma_{area='Central'}(\text{Restaurants}))$

cname	rname
Moe	Gambino Oven
Moe	Lorenzo Tavern
Ralph	Gambino Oven
Ralph	Lorenzo Tavern

Relational Algebra: Example

- Find customer pairs $(C1, C2)$ such that they like some common pizza and $C1 < C2$

Likes

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola

R

cname	cname2
Lisa	Maggie
Lisa	Moe
Maggie	Moe

$$R = \pi_{cname,cname2}(\sigma_{(pizza=pizza2)\ and\ (cname<cname2)}(Likes \times \rho_{cname:cname2,pizza:pizza2}(Likes)))$$

Relational Algebra: Example (cont.)

$$R' = Likes \times \rho_{cname:cname2,pizza:pizza2}(Likes)$$

Likes

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola

cname	pizza	cname2	pizza2
Homer	Hawaiian	Homer	Hawaiian
Homer	Hawaiian	Homer	Margherita
Homer	Hawaiian	Lisa	Funghi
Homer	Hawaiian	Maggie	Funghi
Homer	Hawaiian	Moe	Funghi
Homer	Hawaiian	Moe	Sciliana
Homer	Hawaiian	Ralph	Diavola
Homer	Margherita	Homer	Hawaiian
Homer	Margherita	Homer	Margherita
Homer	Margherita	Lisa	Funghi
Homer	Margherita	Maggie	Funghi
Homer	Margherita	Moe	Funghi
Homer	Margherita	Moe	Sciliana
Homer	Margherita	Ralph	Diavola
Lisa	Funghi	Homer	Hawaiian
Lisa	Funghi	Homer	Margherita
Lisa	Funghi	Lisa	Funghi
Lisa	Funghi	Maggie	Funghi
Lisa	Funghi	Moe	Funghi
Lisa	Funghi	Moe	Sciliana
Lisa	Funghi	Ralph	Diavola
Maggie	Funghi	Homer	Hawaiian
Maggie	Funghi	Homer	Margherita
Maggie	Funghi	Lisa	Funghi
Maggie	Funghi	Homer	Margherita
Maggie	Funghi	Moe	Funghi
Maggie	Funghi	Moe	Sciliana
Maggie	Funghi	Ralph	Diavola
Moe	Funghi	Homer	Hawaiian
Moe	Funghi	Homer	Margherita
Moe	Funghi	Lisa	Funghi
Moe	Funghi	Homer	Margherita
Moe	Funghi	Moe	Funghi
Moe	Funghi	Moe	Sciliana
Moe	Funghi	Ralph	Diavola
Moe	Sciliana	Homer	Hawaiian
Moe	Sciliana	Homer	Margherita
Moe	Sciliana	Lisa	Funghi
Moe	Sciliana	Homer	Margherita
Moe	Sciliana	Moe	Funghi
Moe	Sciliana	Moe	Sciliana
Moe	Sciliana	Ralph	Diavola
Ralph	Diavola	Homer	Hawaiian
Ralph	Diavola	Homer	Margherita
Ralph	Diavola	Lisa	Funghi
Ralph	Diavola	Maggie	Funghi
Ralph	Diavola	Moe	Funghi
Ralph	Diavola	Moe	Sciliana
Ralph	Diavola	Ralph	Diavola

R

cname	cname2
Lisa	Maggie
Lisa	Moe
Maggie	Moe

$R = \pi_{cname,cname2}(\sigma_c(R'))$, where
c is $(pizza = pizza2)$ *and* $(cname < cname2)$

Writing Relational Algebra Queries

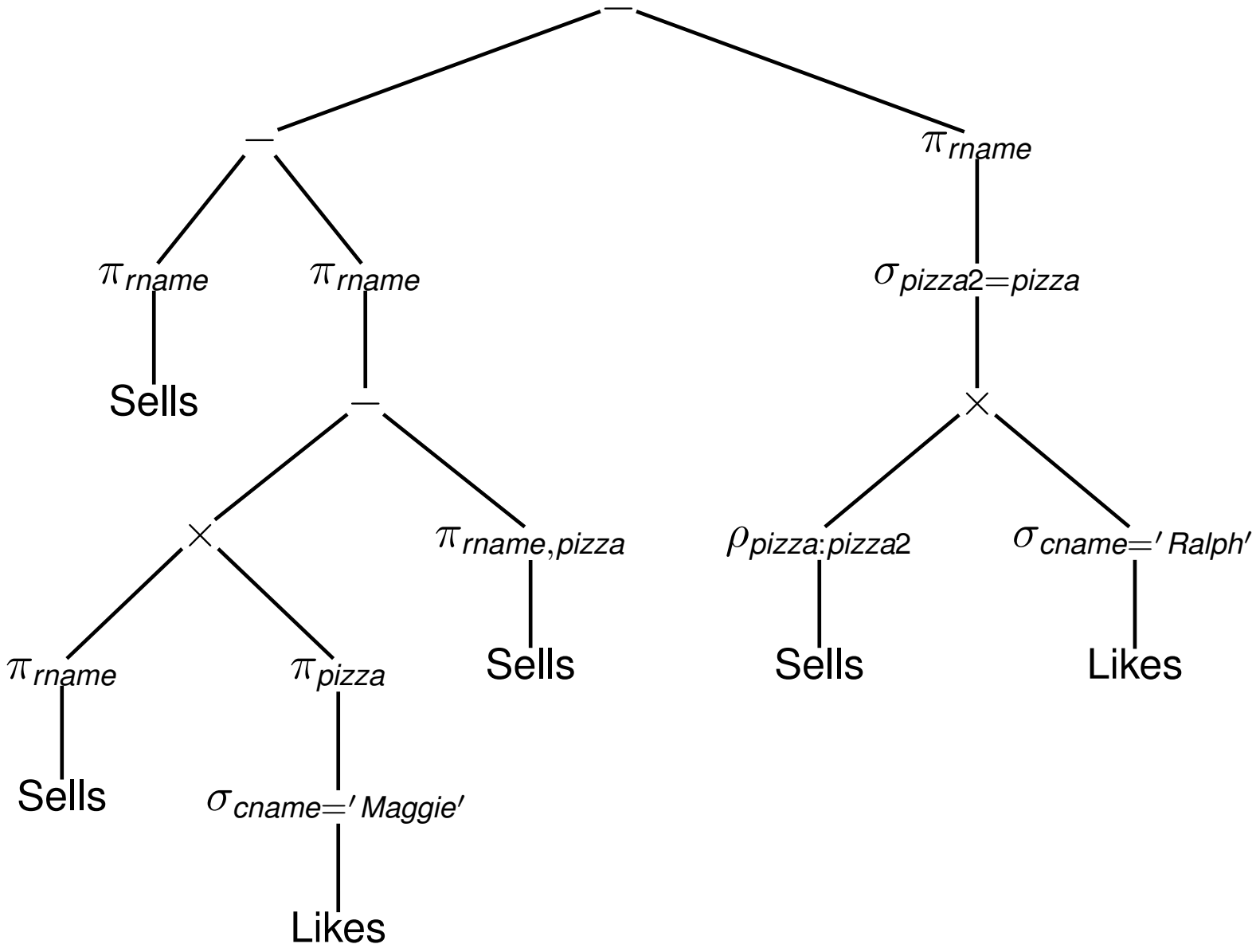
What does this relational algebra query compute?

$$\pi_{rname}(Sells) - \pi_{rname}((\pi_{rname}(Sells) \times (\pi_{pizza}(\sigma_{cname='Maggie'}(Likes)))) - \pi_{rname,pizza}(Sells)) - \pi_{rname}(\sigma_{pizza2=pizza}(\rho_{pizza:pizza2}(Sells) \times \sigma_{cname='Ralph'}(Likes)))$$

Writing Relational Algebra Queries (cont.)

- A complex relational algebra (RA) query presented as a single lengthy expression can be unreadable
- Two methods to improve readability of RA queries
  - Method 1: Operator trees
  - Method 2: Sequence of steps

Method 1: Operator Trees



Method 2: Sequence of steps

$$R_1 = \pi_{pizza}(\sigma_{cname='Maggie'}(\text{Likes}))$$
$$R_2 = \pi_{rname}(\text{Sells}) \times R_1$$
$$R_3 = \pi_{rname}(R_2 - \pi_{rname,pizza}(\text{Sells}))$$
$$R_4 = \pi_{rname}(\text{Sells}) - R_3$$
$$R_5 = \rho_{pizza:pizza5}(\sigma_{cname='Ralph'}(\text{Likes}))$$
$$R_6 = \pi_{rname}(\sigma_{pizza5=pizza}(\text{Sells} \times R_5))$$
$$\text{Answer} = R_4 - R_6$$

Join Operators in Relational Algebra

- A join operator combines cross-product, selection, and possibly projection operators
- More convenient to use than plain cross-product operator
- Join operators:
  - Inner join (aka join), $\bowtie_c$
  - Natural join, $\bowtie$
  - Left outer join (aka left join) $\rightarrow_c$
  - Right outer join (aka right join) $\leftarrow_c$
  - Full outer join (aka full join) $\leftrightarrow_c$
  - Natural left/right/full outer joins $\rightarrow$, $\leftarrow$, $\leftrightarrow$

Inner Join: $R \bowtie_c S$

The inner join of R and S is defined as

$$R \bowtie_c S = \sigma_c(R \times S)$$

Example: Find customer pairs $(C1, C2)$ such that they like some common pizza and $C1 < C2$

Likes

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola

$\pi_{cname,cname2}(Likes \bowtie_c \rho_{cname:cname2,pizza:pizza2}(Likes))$
where $c = (pizza = pizza2)$ *and* $(cname < cname2)$

cname	cname2
Lisa	Maggie
Lisa	Moe
Maggie	Moe

Natural Join: $R \bowtie S$

R

D	B	A
0	x	100
2	y	100
4	w	400
5	z	200
5	v	300

S

E	A	D	C
a	100	0	i
b	300	1	j
c	200	5	k

$R \bowtie_{(A=A2) \text{ and } (D=D2)} \rho_{A:A2,D:D2}(S)$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k

$\pi_{D,A,B,E,C}(R \bowtie_{(A=A2) \text{ and } (D=D2)} \rho_{A:A2,D:D2}(S))$

D	A	B	E	C
0	100	x	a	i
5	200	z	c	k

Natural Join: $R \bowtie S$ (cont.)

The natural join of R and S is defined as

$$R \bowtie S = \pi_{\ell}(R \bowtie_{c} \rho_{a_1:b_1,\cdots,a_n:b_n}(S))$$

where

- A = common attributes between R & S = $\{a_1, a_2, \cdots, a_n\}$
- c is "$(a_1 = b_1)$ *and* $\cdots$ *and* $(a_n = b_n)$"
- ℓ is the list of attributes in R that are also in A, followed by the list of attributes in R that are not in A, and the list of attributes in S that are not in A

Natural Join: Example

For each restaurant, find its name, area, and the pizzas it sells together with their prices

Restaurants

rname	area
Corleone Corner	North
Gambino Oven	Central
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

Sells

rname	pizza	price
Corleone Corner	Diavola	24
Corleone Corner	Hawaiian	25
Corleone Corner	Margherita	19
Gambino Oven	Siciliana	16
Lorenzo Tavern	Funghi	23
Mamma's Place	Marinara	22
Pizza King	Diavola	17
Pizza King	Hawaiian	21

Restaurants ⋈ Sells

rname	area	pizza	price
Corleone Corner	North	Diavola	24
Corleone Corner	North	Hawaiian	25
Corleone Corner	North	Margherita	19
Gambino Oven	Central	Siciliana	16
Lorenzo Tavern	Central	Funghi	23
Mamma's Place	South	Marinara	22
Pizza King	East	Diavola	17
Pizza King	East	Hawaiian	19

Dangling Tuples

R

D	B	A
0	x	100
2	y	100
4	w	400
5	z	200

S

E	A2	D2	C
a	100	0	i
b	300	1	j
c	200	5	k

$R \bowtie_{D=D2} S$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k

- A dangling tuple is a tuple in a join operand that does not participate in the join operation
- Let $attr(R)$ denote the list of attributes in the schema of R
  - Example: $attr(Sells)$ = rname,pizza,price
- We say that $t \in R$ is a dangling tuple in R w.r.t. $R \bowtie_c S$ if $t \notin \pi_{attr(R)}(R \bowtie_c S)$
- To preserve dangling tuples in the join result, use **outer joins**

Left/Right/Full Outer Joins

R

D	B	A
0	x	100
2	y	100
4	w	400
5	z	200

S

E	A2	D2	C
a	100	0	i
b	300	1	j
c	200	5	k

Inner join: $R \bowtie_{D=D2} S$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k

Left outer join: $R \rightarrow_{D=D2} S$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k
2	y	100	null	null	null	null
4	w	400	null	null	null	null

Right outer join: $R \leftarrow_{D=D2} S$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k
null	null	null	b	300	1	j

Full outer join: $R \leftrightarrow_{D=D2} S$

D	B	A	E	A2	D2	C
0	x	100	a	100	0	i
5	z	200	c	200	5	k
2	y	100	null	null	null	null
4	w	400	null	null	null	null
null	null	null	b	300	1	j

Left/Right/Full Outer Joins (cont.)

- Let $dangle(R \bowtie_c S)$ denote the set of dangling tuples in R w.r.t. $R \bowtie_c S$. Thus $dangle(R \bowtie_c S) \subseteq R$
- Let $null(R)$ denote a n-component tuple of null values, where n = arity of relation R
  - Example: $null(Sells)$ = (null,null,null)
- The left outer join of R and S is defined as

$$R \ \rightarrow_c \ S = (R \bowtie_c S) \cup (dangle(R \bowtie_c S) \times \{null(S)\})$$

- The right outer join of R and S is defined as

$$R \ \leftarrow_c \ S = (R \bowtie_c S) \cup (\{null(R)\} \times dangle(S \bowtie_c R))$$

- The full outer join of R and S is defined as

$$R \ \leftrightarrow_c \ S = (R \rightarrow_c S) \cup (\{null(R)\} \times dangle(S \bowtie_c R))$$

Natural Left/Right/Full Outer Joins

- Natural left outer join of R & S:
  $R \rightarrow S = \pi_{\ell}(R \rightarrow_c \rho_{a_1:b_1,\cdots,a_n:b_n}(S))$
- Natural right outer join of R & S:
  $R \leftarrow S = \pi_{\ell}(R \leftarrow_c \rho_{a_1:b_1,\cdots,a_n:b_n}(S))$
- Natural full outer join of R & S:
  $R \leftrightarrow S = \pi_{\ell}(R \leftrightarrow_c \rho_{a_1:b_1,\cdots,a_n:b_n}(S))$

  where
- A = common attributes between R & S = $\{a_1, a_2, \cdots, a_n\}$
- c is "$(a_1 = b_1)$ *and* $\cdots$ *and* $(a_n = b_n)$"
- ℓ is the list of attributes in R that are also in A, followed by the list of attributes in R that are not in A, and the list of attributes in S that are not in A

Natural Left Outer Join: Example

Find customers and the pizzas they like; include also customers who don't like any pizza

Customers

cname	area
Homer	West
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North

Likes

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola

cname	pizza
Homer	Hawaiian
Homer	Margherita
Lisa	Funghi
Maggie	Funghi
Moe	Funghi
Moe	Sciliana
Ralph	Diavola
Willie	null

$$\pi_{cname,pizza}(Customers \rightarrow Likes)$$

Natural Full Outer Join: Example

Find customer-restaurant pairs (C,R) where C and R are located in the same area. Include customers that are not co-located with any restaurant, and include restaurants that are not co-located with any customer

Customers

cname	area
Homer	North
Lisa	South
Maggie	East
Moe	Central
Ralph	Central
Willie	North

Restaurants

rname	area
Corleone Corner	West
Gambino Oven	East
Lorenzo Tavern	Central
Mamma's Place	South
Pizza King	East

cname	rname
Homer	null
Lisa	Mamma's Place
Maggie	Gambino Oven
Maggie	Pizza King
Moe	Lorenzo Tavern
Ralph	Lorenzo Tavern
Willie	null
null	Corleone Corner

$$\pi_{cname,rname}(Customers \leftrightarrow Restaurants)$$

Relational Algebra Expressions (RAEs)

- A relation is a RAE
- If R is a RAE, then $\sigma_p(R)$, $\pi_\ell(R)$, and $\rho_\ell(R)$ are also RAEs
- If R and S are RAEs, then $R \cup S$, $R \cap S$, $R - S$, and $R \times S$, $R \bowtie_c S$, $R \rightarrow_c S$, $R \leftarrow_c S$, $R \leftrightarrow_c S$, $R \bowtie S$, $R \rightarrow S$, $R \leftarrow S$ and $R \leftrightarrow S$ are also RAEs
- If R is a RAE, then (R) is also a RAE

Summary

- Relational algebra: formal language for querying relations
- Basic relational algebra operators
  - selection σ
  - projection π
  - renaming ρ
  - union $\cup$
  - intersect $\cap$
  - difference $-$
  - cross-product $\times$
- Join operators
  - Inner join (aka join), $\bowtie_c$
  - Left outer join (aka left join) $\rightarrow_c$
  - Right outer join (aka right join) $\leftarrow_c$
  - Full outer join (aka full join) $\leftrightarrow_c$
  - Natural inner/left/right/full joins ($\bowtie$, $\rightarrow$, $\leftarrow$, $\leftrightarrow$)