

### Relational Algebra

Chapter 4

### "Formal" Query Languages

- Foundation for commercial query languages like SQL
- Two types
  - Declarative: Relational Calculus
    - Describe what a user wants, rather than how to compute it.
  - Procedural : Relational Algebra
    - Operational, very useful for representing execution plans.
- Query Languages != programming languages!
  - QLs not expected to be "Turing complete".
  - QLs not intended to be used for complex calculations.
  - QLs support easy, efficient access to large data sets.

Understanding Algebra & Calculus is key to understanding SQL, query processing!

### Relational Algebra Preliminaries

- 'Query:
  - Input: Relational instances
  - Output: Relational instances!

Relational Algebra is "closed"

- Specified using the schemas.
  - May produce different results for different instances.
  - But schema of the result is fixed.
- The algebra assumes "set semantics" for relations. OK – this is a formal model!

## Relational Algebra

- Basic operations on relations:
  - Selection ( $\sigma$ ) Selects a subset of rows from relation.
  - Projection  $(\pi)$  Deletes unwanted columns from relation.
  - Cross-product (X) Allows us to combine two relations.
  - Set-difference ( ) Tuples in reln. 1, but not in reln. 2.
  - Union (U) Tuples in reln. 1 and in reln. 2.
- Additional operations (constructed from basic ops):
  - Intersection, Join, Division, Renaming
    - Not essential, but (very!) useful.
- Because algebra is closed, we can <u>compose</u> operators



 Retrieve rows that satisfy a logical condition

 $\sigma_{predicate}(relation)$ 

#### **Athlete**

aid	name	sport	country
1	Mary Lou Retton	gymnastics	USA
2	Jackie Joyner-Kersee	track	USA
3	Michael Phelps	swimming	USA
4	Johann Koss	skating	Norway
5	Natalie Coughlin	swimming	USA
6	Gabby Douglas	gymnastics	USA

#### Example:

 $\sigma_{sport='gymanstics'} \wedge country='_{USA'}$  (Athlete)

aid	name	sport	country
1	Mary Lou Retton	gymnastics	USA
6	Gabby Douglas	gymnastics	USA



 $\pi_{projectionlist}$  (Relation)

# Delete attributes that are not in projection list

- Projectionlist: a list of columns
- The result is a set (relational algebra uses set semantics)
- Remove duplicates!

#### **Athlete**

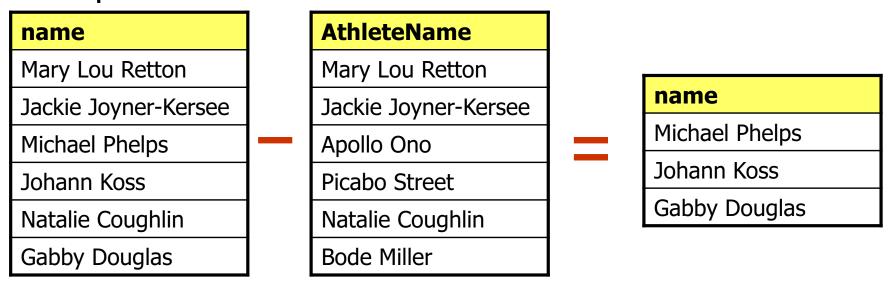
aid	name	sport	country
1	Mary Lou Retton	gymnastics	USA
2	Jackie Joyner-Kersee	track	USA
3	Michael Phelps	swimming	USA
4	Johann Koss	skating	Norway
5	Natalie Coughlin	swimming	USA
6	Paul Hamm	gymnastics	USA

$$\pi_{sport,country}(Athlete)$$

sport	country
gymnastics	USA
track	USA
swimming	USA
skating	Norway

# Set Operations: Union (u), Intersection(n), Set-Difference (-)

- Input: Two <u>union-compatible</u> relations
  - Same number and type of attributes, in same order
- Field names of result: uses the name from the FIRST input



Duplicates? Relational algebra uses set semantics. So no duplicates. Difference from SQL



### Cross-Product (Cartesian Product) X

- Result Schema
  - One field from both relations (Names inherited)
- If both input relations have a field with the same name, can use the rename operator p. (See 4.2.2 and 4.2.3)

in textbook)

#### **Athlete**

aid	name	sport	country
1	Mary Lou Retton	gymnastics	USA
2	Jackie Joyner-Kersee	track	USA

#### Venue

vid	venue
1	Los Angeles
2	Barcelona

aid	name	sport	country	vid	venue
1	Mary Lou Retton	gymnastics	USA	1	Los Angeles
2	Jackie Joyner-Kersee	track	USA	1	Los Angeles
1	Mary Lou Retton	gymnastics	USA	2	Barcelona
2	Jacki Joyner-Kersee	track	USA	2	Barcelona

## Derived Operators: Joins

- Most common way of combining information from two tables
- Conditional join (sometimes called a Θ-join)
  - Definition:  $R \bowtie_c S = \sigma_c(R \times S)$ , where *c* is a condition
- Equijoin
  - Join condition consists only of equalities
- Natural Join
  - Equijoin in which equalities are specified on all fields with the same name in R and S
- Despite equivalence, usually faster ways to evaluate joins than to compute cross-product!



### **Examples: Writing Queries in RA**

#### **Example Schema:**

Sailors (<u>sid</u>, sname, rating, age) Reserves (<u>sid</u>, <u>bid</u>, <u>day</u>) Boats (<u>bid</u>, bname, color)

#### **Sailors**

sid	sname	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

#### Reserves

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

#### **Boats**

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

## Find names of sailors who've reserved boat #103

Solution 1: (1) Extract reservations for boat ID 103 (2) Join with sailors and project on sname

Solution 2: Same as 1, but give temp names to intermediate results

Solution 3: (1) Join Reserves and Sailors

(2) Select on bid = 103

(3) project on sname

# Find names of sailors who've reserved a red boat

$$\pi_{sname}((\sigma_{color='red'}, Boats) \bowtie Reserves \bowtie Sailors)$$

#### An equivalent solution:

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'}Boats) \bowtie Res) \bowtie Sailors)$$

<u>Query optimizer</u> chooses from the (equivalent) expressions and chooses one for efficiency of evaluation.

# Find the names of sailors who've reserved at least one boat

 $\pi_{sname}$ (Reserves  $\bowtie$  Sailors)

Sailor appears in this intermediate relation only if there is at least one Reserves tuple with same sid.

## Derived Operators: Division

Useful for queries like:

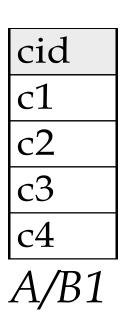
Find customers with accounts at all branches in Brooklyn.

- Let A have 2 fields, x and y; B have only field y:
  - $A/B = \{ \langle x \rangle \mid \forall \langle y \rangle \in B, \langle x, y \rangle \in A \}$
  - A/B contains all x tuples (customers) such that for <u>every</u>
     y tuple (branches in Brooklyn) in B, there is an <x,y>
     tuple in A
- In general, x and y can be lists of fields; y is the list of fields in B, and x ( )y is the list of fields of A.

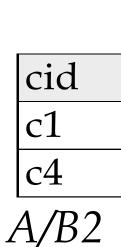
### Examples of Division A/B

cid	bname
c1	b1
c1	b2
c1	b3
c1	b4
c2	b1
c2	b2
c3	b2
c4	b2
c4	b4

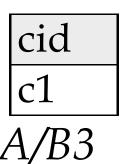
bname	
b2	
B1	



bname
b2
b4
<i>B</i> 2



bname
b1
b2
b4
<i>B3</i>



A

## Expressing A/B Using Basic Operators

- Can be equivalently expressed using basic operators
- Idea: For A/B, compute all x values that are not disqualified by some y value in B.
  - x value is disqualified if by attaching y value from B, we obtain an <x,y> tuple that is not in A

operator using basic operators?

Can you express this operator using basic 
$$\pi_{\chi}(A) - \pi_{\chi}((\pi_{\chi}(A) \times B) - A)$$

Disqualified x values

### **Examples of Division A/B**

cid	bname
c1	b1
c1	b2
c1	b3
c1	b4
c2	b1
c2	b2
c3	b2
c4	b2
c4	b4

bname
b2
b4
$\overline{B}$

cid	bname
<del>c1</del>	b2
<del>c</del> 1	b4
<del>c2</del>	<u>b2</u>
c2	b4
<del>-c3</del>	b2
c3	b4
<del>c4</del>	<u>b2</u>
<del>c4</del>	<del>b4</del>
$\pi_{\infty}$	$A)\times B$

cid	
c1	
c4	
A/B	

A

# Find names of sailors who've reserved a red OR green boat

- Identify all red or green boats, then
- find sailors who've reserved one of these boats:

$$\rho$$
 (Tempboats, ( $\sigma$  color = 'red'  $\vee$  color = 'green' Boats))
$$\pi_{sname}$$
(Tempboats  $\bowtie$  Reserves  $\bowtie$  Sailors)

#### Equivalent:

```
\rho(Tempboats, (\sigma_{color='red'}(Boats) \cup \sigma_{color='green'}(Boats)))
\pi_{sname}(Tempboats \bowtie Reserves \bowtie Sailors)
```

# Find names of sailors who've reserved a red AND green boat

- Does this work: Change ∨ to ∧ on previous slide?
- How about this?
  - Identify Sailor ids who've reserved red boats  $\rho(TempRed, \pi_{sid}((\sigma_{color='Red'}Boats) \bowtie Reserves))$
  - Identify Sailor ids who' ve reserved green boats  $\rho(TempGreen, \pi_{sid} \Big( (\sigma_{color='Green'} Boats) \bowtie Reserves \Big))$
  - Then use the intersection to get the names  $\pi_{sname}((TempRed \cap TempGreen) \bowtie Sailors)$

# Find the sids of sailors over age 20 who have not reserved a red boat

$$\pi_{sid}(\sigma_{age>20}Sailors) - \pi_{sid}((\sigma_{color='Red'}Boats) \bowtie Reserves)$$

## Find the names of sailors who've reserved all boats

 Uses division; schemas of the input relations must be carefully chosen:

```
\rho \ (Tempsids, (\pi_{sid,bid} Reserves) / (\pi_{bid} Boats))
\pi_{sname} (Tempsids \bowtie Sailors)
```



- Exercises 4.1, 4.3, 4.5
  - Only RA required for the last two.

## Find names of sailors who've reserved boat #103

Solution 1:

$$\pi_{sname}((\sigma_{bid=103} \text{Reserves}) \bowtie Sailors)$$

Solution 2:

$$\rho$$
 (Temp1,  $\sigma_{bid=103}$  Reserves)

 $\rho$  (Temp2, Temp1  $\bowtie$  Sailors)

 $\pi_{sname}$  (Temp2)

Solution 3:

$$\pi_{sname}(\sigma_{bid=103}(\text{Reserves} \bowtie Sailors))$$