Relational Algebra

Chapter 4

Division

- Not supported as a primitive operator, but useful for expressing queries like:

  *Find sailors who have reserved **all** boats.*

- Let *A* have 2 fields, *x* and *y*; *B* have only field *y*:
  - $A/B = \left\{ \langle x \rangle \mid \exists \langle x, y \rangle \in A \;\; \forall \langle y \rangle \in B \right\}$
  - i.e., ***A/B* contains all *x* tuples (sailors) such that for *every* *y* tuple (boat) in *B*, there is an *xy* tuple in *A*.**
  - *Or*: If the set of *y* values (boats) associated with an *x* value (sailor) in *A* contains all *y* values in *B*, the *x* value is in *A/B*.
- In general, *x* and *y* can be any lists of fields; *y* is the list of fields in *B*, and $x \cup y$ is the list of fields of *A*.

Examples of Division A/B

sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

A

pno
p2

B1

sno
s1
s2
s3
s4

A/B1

pno
p2
p4

B2

sno
s1
s4

A/B2

pno
p1
p2
p4

B3

sno
s1

A/B3

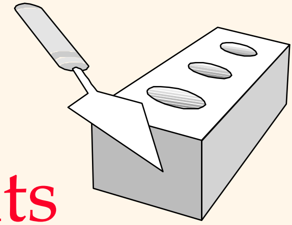
Expressing A/B Using Basic Operators

- Division is not essential op; just a useful shorthand.
  - (Also true of joins, but joins are so common that systems implement joins specially.)
- *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 *xy* tuple that is not in *A*.

Disqualified *x* values: $\pi_x((\pi_x(A) \times B) - A)$

A/B: $\pi_x(A)$ $-$ all disqualified tuples

Find names of sailors who've reserved boat #103



Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

<u>bid</u>	<u>Color</u>
101	Red
103	Green

Solution 1

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

Solution 2

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

Find names of sailors who've reserved boat #103

Sailors

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Boats

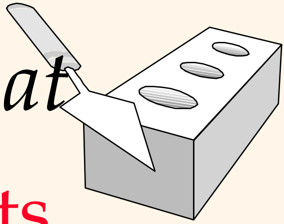
bid	Color
101	Red
103	Green

- Solution 3: $\rho\,(Temp1,\ \sigma_{bid=103}\,Reserves)$

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

$$\pi_{sname}\,(Temp2)$$

Find names of sailors who've reserved a red boat



Sailors

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Boats

bid	Color
101	Red
103	Green

Solution 1

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

Solution 2

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

Find names of sailors who've reserved a red boat

Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

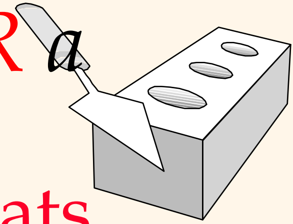
<u>bid</u>	<u>Color</u>
101	Red
103	Green

- A more efficient solution:

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

A query optimizer can find this!

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



Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

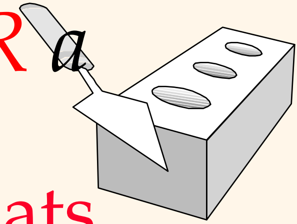
<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

<u>bid</u>	<u>Color</u>
101	Red
103	Green

$$\pi_{sname}((\sigma_{col='red' \vee col='green'} Boats) \bowtie \text{Reserves} \bowtie Sailors)$$

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



Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

<u>bid</u>	<u>Color</u>
101	Red
103	Green

- Using Union:

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

$$\rho(Tempgreen, \pi_{sname}((\sigma_{col='green'} Boats) \bowtie Reserves \bowtie Sailors))$$

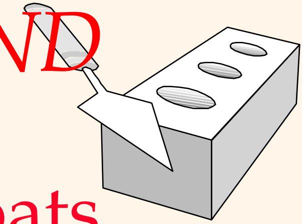
$$Tempred \cup Tempgreen$$

$$\rho(Tempred, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$$

$$\rho(Tempgreen, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$$

$$\pi_{sname}((Tempred \cup Tempgreen) \bowtie Sailors)$$

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



Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

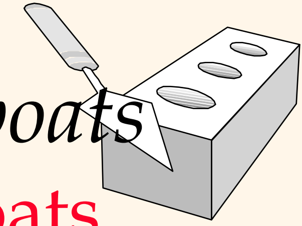
<u>bid</u>	<u>Color</u>
101	Red
103	Green

$$\rho(Tempred, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$$

$$\rho(Tempgreen, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$$

$$\pi_{sname}((Tempred \cap Tempgreen) \bowtie Sailors)$$

Find the names of sailors who've reserved all boats



Sailors

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Boats

bid	Color
101	Red
103	Green

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

$$\rho(Tempsids, (\pi_{sid,bid} Reserves) / (\pi_{bid} Boats))$$

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

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

Sailors

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Reserves

<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

Boats

<u>bid</u>	<u>Color</u>
101	Red
103	Green

$$\rho(Tempsids, (\pi_{sid,bid}\text{Reserves}) / (\pi_{bid}(\sigma_{Color='Green'}Boats)))$$

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

Example

- Employee (ID, person_name, street, city)
- Works (ID, company_name, salary)
- Company (company_name, city)

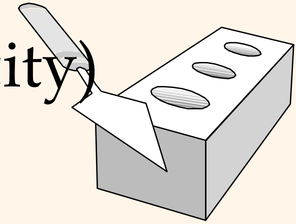
- Find the names of employees who live in Miami

$$\Pi_{person\_name}\ (\sigma_{city\ =\ \text{“Miami”}}\ (employee)\ )$$

- Find the names of employees whose salary is more than \$100K

$$\Pi_{person\_name}\ (\sigma_{salary\ >\ 100K}\ (employee \bowtie_{ID} works)\ )$$

Example

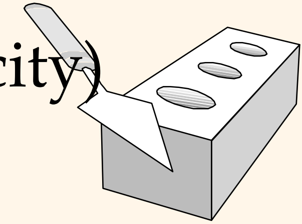


- Employee (ID, person_name, street, city)
- Works (ID, company_name, salary)
- Company (company_name, city)

- Find the names of employees who live in Miami and whose salary is more than $100K

$$\Pi_{person\_name}\left(\sigma_{city = \text{“Miami”} \text{ AND } salary > 100K}\left(employee \bowtie_{ID} works\right)\right)$$

Example



- Employee (ID, person_name, street, city)
- Works (ID, company_name, salary)
- Company (company_name, city)

- Find the ID and name of each employee who does not work for BigBank

$$\Pi_{ID,person\_name}(employee) - \Pi_{ID,person\_name}(employee \bowtie_{ID} (\sigma_{company\_name = \text{"BigBank"}}(works)))$$

Summary

- The relational model has rigorously defined query languages that are simple and powerful.
- Relational algebra is more operational; useful as internal representation for query evaluation plans.
- Several ways of expressing a given query; a query optimizer should choose the most efficient version.