Database Management Systems

Relational Algebra

M. Emre Gürsoy

Assistant Professor
Department of Computer Engineering
www.memregursoy.com

Introduction

- Relational query languages allow manipulation and retrieval of data from relational databases.
- Relational model supports several query languages.
- Two mathematical relational query languages that form the basis for "practical" languages (e.g., SQL):
  - Relational Algebra: More operational (procedural), useful for representing execution plans.
  - (Tuple) Relational Calculus: Lets users describe what they want, rather than how to compute it (non-operational, declarative).
- In this lecture, we will learn about **relational algebra**.
  - You can read about other query languages in the book.
    - Tuple relational calculus, domain relational calculus, query-by-example (QBE), ...

Relational Algebra

- Relational algebra contains a set of basic operators.
- These operators enable a user to specify **retrieval requests** (**queries**).
- Several operations can be composed (one inside the other) to express a more complex query.
- Queries are executed on relation states (instances).
  - Sailors reserve boats
  - S: sailors, R: reservations

R1

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

S1

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

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

RA Operators

- Projection - symbol: π (pi)
- Selection - symbol: σ (sigma)
- Union - symbol: $\cup$
- Intersection - symbol: $\cap$
- Difference (Set Difference) - symbol: – (minus)
- Cartesian Product (Cross Product) - symbol: **x**
- Rename - symbol: ρ (rho)
- Join - symbol: $\bowtie$ (bowtie)
- Division - symbol: **/** (slash)

- We won't cover aggregate functions (SUM, COUNT, MIN, MAX, AVG, etc.)

Projection

- Keep the attributes in the projection list, delete the rest.
- Resulting schema contains exactly the attributes in the projection list.
- By default, duplicate tuples are removed.

sname	rating
yuppy	9
lubber	8
guppy	5
rusty	10

$\pi_{sname,rating}(S2)$

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

age
35.0
55.5

$\pi_{age}(S2)$

Selection

- Select rows that satisfy a boolean selection condition.
- Resulting schema is identical to the schema of the input relation.

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

sid	sname	rating	age
28	yuppy	9	35.0
58	rusty	10	35.0

$$\sigma_{rating>8}(S2)$$

sname	rating
yuppy	9
rusty	10

$$\pi_{sname,rating}(\sigma_{rating>8}(S2))$$

Union

- Takes two union-compatible relations as input
  - Same number of attributes
  - Corresponding attributes have the same data type
- Computes set union – duplicates are removed
- What is the schema of the result?

S1

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

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0
44	guppy	5	35.0
28	yuppy	9	35.0

$S1 \cup S2$

Intersection and Difference

- Set operations, similar rules as **union**

S1

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

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

sid	sname	rating	age
22	dustin	7	45.0

$$S1 - S2$$

sid	sname	rating	age
31	lubber	8	55.5
58	rusty	10	35.0

$$S1 \cap S2$$

Cross Product

- Also known as **Cartesian product**
- Combine tuples from two relations:
  - R(A1, A2, . . ., An) x S(B1, B2, . . ., Bm)
- Result is a relation Q with n+m attributes
  - Q(A1, A2, . . ., An, B1, B2, . . ., Bm)
- The resulting relation state has all possible combinations (pairings) of tuples between R and S
  - If R has n_R tuples (denoted as |R| = n_R), and S has n_S tuples, then Q = R x S will have n_R * n_S tuples

- Do we need R and S to be union-compatible in order to perform a cross product?

Cross Product

- S1 x R1: Each row in S1 is paired with each row of R1

S1

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

R1

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

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Renaming

- Problem on the previous slide?
  - There are two potential sid columns.
  - Renaming operator **rho** can fix this!

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

$$\rho\,(D(1 \rightarrow sid1, 5 \rightarrow sid2),\ S1 \times R1)$$

Joins

- Combine multiple tables – important concept!
- You can think of a join as cross product followed by a selection:

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

- Result schema is the same as that of the cross product, but table typically has fewer tuples than cross product.
  - Tuples not satisfying the join condition are filtered out.

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	58	103	11/12/96

$$S1 \bowtie_{S1.sid < R1.sid} R1$$

Joins

- **Equi-join:** A special type of join (very common in practice) in which the join condition only contains equality.
- Result schema is similar to cross product, but includes only one copy of the attribute(s) for which equality is enforced.

sid	sname	rating	age	bid	day
22	dustin	7	45.0	101	10/10/96
58	rusty	10	35.0	103	11/12/96

$$S1 \bowtie_{sid} R1$$

Also allowed:

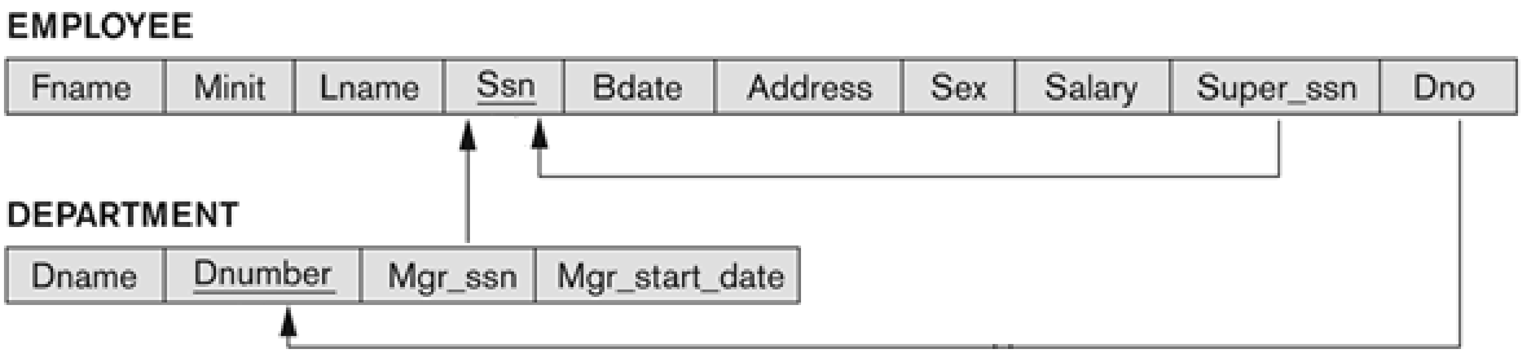
S1.sid = R1.sid

sid = sid

(empty): when column names agree and context is clear

Example

- For each department, print the deparment's name, number, and last name of its manager.



Fname	Minit	Lname	Ssn	...
Franklin	T	Wong	333445555	...
Jennifer	S	Wallace	987654321	...
James	E	Borg	888665555	...

Dname	Dnumber	Mgr_ssn	...
Research	5	333445555	...
Administration	4	987654321	...
Headquarters	1	888665555	...

$$\pi_{Dname,Dnumber,Lname}(\text{Department} \bowtie_{Mgr\_ssn=Ssn} \text{Employee})$$

Division

- Dividing A/B: Find those tuples in A that have all of B

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

pno
p2

B1

pno
p2
p4

B2

pno
p1
p2
p4

B3

sno
s1
s2
s3
s4

A/B1

sno
s1
s4

A/B2

sno
s1

A/B3

Exercises

- Consider the following schema:
  - Sailors (<u>sid</u>, sname, rating, age)
  - Boats (<u>bid</u>, bname, color)
  - Reserves (<u>sid, bid, day</u>)

Sailors

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

Boats

bid	bname	color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

Reserves

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

Exercises

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

- Find the names of sailors who have reserved boat # 103.

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

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

Both solutions are correct.
Which one is faster?

Exercises

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

- Find the names of sailors who have reserved a red boat.

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

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

p.s.: The difficulty of writing faster versions of queries grows rather quickly.. hence, query optimizers!

Exercises

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

- Find the names of sailors who have reserved a red boat or a green boat.

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

Boats

bid	bname	color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

Exercises

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

- Find the names of sailors who have reserved a red boat and a green boat (can be on two different days).

$$\underbrace{\pi_{sid}((\sigma_{color='red'}\text{Boats}) \bowtie \text{Reserves})}_{\text{sid's of sailors who reserved a red boat}} \bigcap \underbrace{\pi_{sid}((\sigma_{color='green'}\text{Boats}) \bowtie \text{Reserves})}_{\text{sid's of sailors who reserved a green boat}}$$

sid's of sailors who reserved both a red boat and a green boat

to retrieve their names, join with *Sailors* and project onto *sname*

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

Exercises

- Sailors (sid, sname, rating, age)
- Boats (bid, bname, color)
- Reserves (sid, bid, day)

- Find the names of sailors who have reserved all boats.
  - Uses division, but input schemas must be chosen carefully

$$\pi_{sname}((\pi_{sid,bid}(\text{Reserves})/\pi_{bid}(\text{Boats})) \bowtie \text{Sailors})$$

Exercises

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

- Find the names of sailors who have reserved all "Interlake" boats.

$$\pi_{sname}((\pi_{sid,bid}(\text{Reserves})/\pi_{bid}(\sigma_{bname='Interlake'}\text{Boats})) \bowtie \text{Sailors})$$

Boats

bid	bname	color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

Exercises

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

- Find the colors of boats reserved by a sailor named Albert.

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

Exercises

- Sailors (sid, sname, rating, age)
- Boats (bid, bname, color)
- Reserves (sid, bid, day)

- Find the names and id's of sailors who have **not** reserved a red boat.

$$\pi_{sname,sid}([\pi_{sid}(\text{Sailors}) - \pi_{sid}((\sigma_{color='red'}\text{Boats}) \bowtie \text{Reserves})] \bowtie \text{Sailors})$$

Exercises

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

- Find the sailor id's of sailors whose rating is better than some sailor called Bob.
  - There can be multiple sailors called “Bob” in the Sailors table
  - For a sailor to appear in the result, it is sufficient for them to have higher rating than any one of the Bobs

$$\pi_{S2.sid}(\sigma_{S2.rating>Sailors.rating}[\rho(\mathrm{S2}, \mathrm{Sailors}) \times (\sigma_{sname='Bob'}\mathrm{Sailors})])$$

Exercises

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

- Find the sailor id's of sailors whose rating is better than every sailor called Bob.

$$\pi_{sid}(\text{Sailors}) - \pi_{S2.sid}(\sigma_{S2.rating \leq Sailors.rating}[\rho(\text{S2}, \text{Sailors}) \times (\sigma_{sname='Bob'} \text{Sailors})])$$

Exercises

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

- Find the name and age of the oldest sailor.

$$\text{Let } A = \pi_{sid}(\text{Sailors})$$

$$\text{Let } B = \pi_{S2.sid}[\sigma_{S2.age<Sailors.age}(\rho(\text{S2}, \text{Sailors}) \times \text{Sailors})]$$

$$\text{The answer is: } \pi_{sname,age}[(A - B) \bowtie \text{Sailors}]$$