SQL and Relational Algebra

Dr. Bing Zhou

Basics of Relational Algebra

- Four types of operators:
  - Select/Show parts of a single relation: projection and selection.
  - Usual set operations (union, intersection, difference).
  - Combine the tuples of two relations, such as cartesian product and joins.
  - Renaming.

Projection

- The projection operator produces from a relation R a new relation containing only some of R's columns.

- "Delete" (i.e. not show) attributes not in projection list.
- Duplicates eliminated

- To obtain a relation containing only the columns $A_1, A_2, \ldots A_n$ of R

RA: $\pi\ A_1, A_2, \ldots A_n\ (R)$

SQL: **SELECT** $A_1, A_2, \ldots A_n$ **FROM** R;

Selection

- The selection operator applied to a relation R produces a new relation with a subset of R's tuples.
- The tuples in the resulting relation satisfy some condition C that involves the attributes of R.
  - with duplicate removal

RA: $\boldsymbol{\sigma}_{C}(R)$

SQL: **SELECT** * **FROM** R **WHERE** C;

- The WHERE clause of a SQL command corresponds to **σ**( ).

Union

- The union of two relations R and S is the set of tuples that are in R or in S or in both.
  - R and S must have identical sets of attributes and the types of the attributes must be the same.
  - The attributes of R and S must occur in the same order.

- What is the schema of the result ?

RA: R **U** S

SQL: (**SELECT** * **FROM** R)
UNION
(**SELECT** * **FROM** S);

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

$S1 \cup S2$

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

Intersection

- The intersection of two relations R and S is the set of tuples that are in both R and S.
- Same conditions hold on R and S as for the union operator.
  - R and S must have identical sets of attributes and the types of the attributes must be the same.
  - The attributes of R and S must occur in the same order.

RA: $R \cap S$

SQL: (**SELECT** * **FROM** R)
INTERSECT
(**SELECT** * **FROM** S);

Intersection

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

$S1 \cap S2$

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

Difference

- The difference of two relations R and S is the set of tuples that are in R but not in S.
- Same conditions hold on R and S as for the union operator.
  - R and S must have identical sets of attributes and the types of the attributes must be the same.
  - The attributes of R and S must occur in the same order.

RA: R - S

SQL: (**SELECT** * **FROM** R)
EXCEPT
(**SELECT** * **FROM** S);

- R − (R − S) = R ∩ S

Difference

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

$S1 - S2$

sid	sname	rating	age
22	dustin	7	45.0

Cartesian Product

- The Cartesian product (or cross-product or product) of two relations R and S is a the set of pairs that can be formed by pairing each tuple of R with each tuple of S.
  - The result is a relation whose schema is the schema for R followed by the schema for S.

RA: R **X** S

SQL: **SELECT** * **FROM** R , S ;

Cartesian Product

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

?

S1 x R1

(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

We rename attributes to avoid ambiguity or we prefix attribute with the name of the relation it belongs to.

Theta-Join

- The theta-join of two relations R and S is the set of tuples in the Cartesian product of R and S that satisfy some condition C.

RA: $R \underset{C}{\infty} S$

SQL:
```
SELECT *
FROM R , S
WHERE C;
```

- $R \underset{C}{\infty} S = \sigma_C (R \times S)$

Theta-Join

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

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

(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

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

Natural Join

- The natural join of two relations R and S is a set of pairs of tuples, one from R and one from S, that agree on whatever attributes are common to the schemas of R and S.
- The schema for the result contains the union of the attributes of R and S.
- Assume the schemas R(A,B, C) and S(B, C,D)

RA: R ∞ S

SQL: **SELECT** *
FROM R , S
WHERE R.B = S.B **AND** R.C = S.C;

Example: Natural Join

Sells(

bar,	beer,	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

Bars(

bar,	addr
Joe's	Maple St.
Sue's	River Rd.

)

BarInfo := Sells ⋈ Bars

Note: Bars.name has become Bars.bar to make the natural join "work."

BarInfo(

bar,	beer,	price,	addr
Joe's	Bud	2.50	Maple St.
Joe's	Milller	2.75	Maple St.
Sue's	Bud	2.50	River Rd.
Sue's	Coors	3.00	River Rd.

)

Operators Covered So far

- Remove parts of a single relation:
  - projection: $\pi_{A,B}(R)$ and `SELECT A, B FROM R`.
  - selection: $\sigma_C(R)$ and `SELECT * FROM R WHERE C`.
  - combining projection and selection:
    - $\pi_{A,B}(\sigma_C(R))$
    - `SELECT A, B FROM R WHERE C`.
- Set operations (R and S must have the same attributes, same attribute tyes, and same order of attributes):
  - union: $R \cup S$ and `(R) UNION (S)`.
  - intersection: $R \cap S$ and `(R) INTERSECT (S)`.
  - difference: $R - S$ and `(R) EXCEPT (S)`.
- Combine the tuples of two relations:
  - Cartesian product: $R \times S$ and `... FROM R, S ...`.
  - Theta-join: $R \bowtie_C S$ and `... FROM R, S WHERE C`.
  - Natural join: $R \bowtie S$; in SQL, list the conditions that the common attributes be equal in the `WHERE` clause.

Renaming

- If two relations have the same attribute, disambiguate the attributes by prefixing the attribute with the name of the relation it belongs to.
- How do we answer the query "Name pairs of students who live at the same address"? `Students(Name, Address)`
  - We need to take the cross-product of Students with itself?
  - How do we refer to the two "copies" of Students?
  - Use the rename operator.

RA: $\rho_{S\ (A1,A2,\ \ldots\ An)}(R)$: give R the name S; R has n attributes, which are called A1,A2, . . . ,An in S

SQL: Use the **AS** keyword in the **FROM** clause: Students AS Students1 renames Students to Students1

Renaming

Name pairs of students who live at the same address.

RA $\pi_{\texttt{S1.Name,S2.Name}}($
$\sigma_{\texttt{S1.Address = S2.Address}}(\rho_{\texttt{S1}}(\texttt{Students}) \times \rho_{\texttt{S2}}(\texttt{Students}))).$

SQL
```
SELECT S1.name, S2.name
FROM Students AS S1, Students AS S2
WHERE S1.address = S2.address;
```

- Are these correct ?
- No !!! the result includes tuples where a student is paired with himself/herself
- Solution: Add the condition S1.name <> S2.name.

Practicing Relational Algebra

Q1: Find names of sailors who have reserved boat #103

Reserves(sid, bid, day)

Sailors(sid, sname, rating, age)

- Solution 1:

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

- Solution 2 (more efficient)

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

- Solution 3 (using rename operator)

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

  $P(Temp2\ (Temp1 \infty Sailors))$

  $\pi_{sname}(Temp2)$

Q2: Find names of sailors who have reserved a red boat

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

- Solution 1:
  $\pi_{sname}((\sigma_{color = \text{'red'}} Boats) \infty Reserves \infty Sailors)$

- Solution 2 (more efficient)
  $\pi_{sname}(\pi_{sid} ((\pi_{bid}\sigma_{color = \text{'red'}} Boats) \infty Reserves) \infty Sailors)$

Q3: Find the colors of boats reserved by Lubber

Reserves(sid, bid, day)
Boats(bid, bname, color)

Sailors(sid, sname, rating, age)

- Solution:

$$\pi_{color}((\sigma_{sname = \text{'Lubber'}} Sailor) \infty Reserves \infty Boats)$$

Q4: Find the names of sailors who have reserved at least one boat

Reserves(sid, bid, day)
Boats(bid, bname, color)

Sailors(sid, sname, rating, age)

- Solution:

$$\pi_{sname}(Sailor \infty\ Reserves)$$

Q5: Find the names of sailors who have reserved a red or a green boat

Reserves(sid, bid, day)
Boats(bid, bname, color)

Sailors(sid, sname, rating, age)

- Solution:

$$\pi_{sname}(\sigma_{color='red' \text{ or } color='green'} \textbf{Boats} \infty \textbf{Reserves} \infty \textbf{Sailors})$$

Q6: Find the names of sailors who have reserved a red and a green boat

Reserves(sid, bid, day)
Boats(bid, bname, color)

Sailors(sid, sname, rating, age)

- Solution:

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

A ship cannot have TWO colors at the same time

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

Q7: Find the sids of sailors with age over 20 who have not reserved a red boat

Reserves(sid, bid, day)
Boats(bid, bname, color)

Sailors(sid, sname, rating, age)

Strategy ? ? ?

Find all sailors (sids) with age over 20
Find all sailors (sids) who have reserved a red boat
Take their set difference

- Solution:
  $\pi_{sid}\ (\sigma_{age>20}\ Sailors) - \pi_{sid}\ ((\sigma_{color='red'}\ Boats) \infty\ Reserves)$

Product (make, model, type)
PC (model, speed, ram, hd, price)
Laptop (model, speed, ram, hd, screen, price)
Printer (model, color, type, price)