Relational Algebra

Wednesday, April 17, 2019 11:03 AM

- SQL vs RA
  - (SQL: declare what we want; RA: express how to get it)
  - RA expression ⇔ query plan
  - Every DBMS implementations converts a SQL query to RA in order to execute it.
  - RA are often displayed using a tree.
- Set semantics - standard relational algebra
  Bag (allow duplicate) semantics - extended relational algebra
  (DB systems use; because it's more efficient.)
- Relational Algebra Operators

Union ∪, difference −
(intersection (not basic one))
Select σ
Projection π
Cartesian product ×, join ⋈
Rename ρ
} relational algebra
} extended RA

- Duplicate elimination δ
- Grouping and Aggregation γ
- Sorting τ

- $R1 \cap R2 = R1 - (R1 - R2)$

  $= R1 \bowtie R2$ when $R1, R2$ have same attributes

- $R1 \cup R2$: Bag Sem: if a appear twice in R_1, three times in R_2, then a appear 0 times in union.

- selection $\sigma_c(R)$

  e.g. $\sigma_{salary>10000}(Employee)$

- Projection $\Pi_{A1,\dots,An}(R)$

  e.g. $\Pi_{SSN, Name}(Employee) \rightarrow Answer(SSN, Name)$

  set semantics: will eliminate duplicate

  bag semantics: will not. (more efficient)

- Renaming $\rho_{B1,\dots,Bn}(R)$

  e.g. Given Employee(Name, SSN)

  $\rho_{N,S}(Employee) \rightarrow Answer(N, S)$

- Natural Join $R1 \bowtie R2$

  $R1 \bowtie R2 = \Pi_A(\sigma_\theta(R1 \times R2))$ → Cartesian product

$\bowtie_\theta$: check equality of all common attributes

Π_A : eliminates duplicate common attributes.

- Do projection after joined.

- When they don't share attribute:

  $R(A,B,C) \bowtie S(D,E)$

  $= R \times S$ (Simply join everything)

- $R(A,B) \bowtie S(A,B)$

  $= R \cap S$

- Theta join : are expensive

  $R1 \bowtie_\theta R2$, θ can be any condition.

- Equijoin

  $R1 \bowtie_\theta R2$, θ is equality

  e.g. $P \bowtie_{p.age = v.age} V$

* By "$\bowtie$", we often mean equijoin.

- Left outer join $P = \mathcal{J}$

  have everything on left.

- Groupby γ

  - there's no diff between having & where. they differ just by time

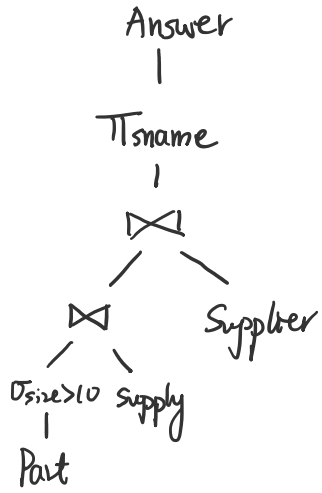
- σ after γ : having

  σ before γ : where

- Represent RA Queries as joins

  o e.g.

$\Pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part)))$




```
select z.sname
from Part x, Supply y, Supplier z
where x.psize>10
  and x.pno = y.pno
  and y.sno = z.sno
```

- Group by

```
select city, sum(quantity)
from sales
group by city
having count(*)>100
```

Answer

$\Pi_{city, q}$ temp2(city, q, c)

$\sigma_{c>100}$ temp(city, q, c)

$\gamma_{city, sum(quantity)\to q,\ count(*)\to c}$

sales(product, city, quantity)