Chapter # 08 — Relational Algebra. ①

Date:

* Unary Relational Operations:

① Select operation: Chooses a subset of tuples from a relation that satisfies a selection condition.

$\sigma_{<\text{selection condition}>}(R)$

Unary operator → applied to single relation.
Commutative. Degree → its number of attributes

② Project operation: Selects certain columns from table

$\pi_{<\text{attribute list}>}(R)$

Degree → equal to no. of attributes in <attribute list>
Removes duplications, is a set of distinct tuples called duplicate elimination.

If projection list is some super key of R i.e includes key of R, the resulting relation has a same number of tuples as R.

③ Sequences of operation & Rename Operation:

→ Inline expression: eg: $\pi_{Fname, Lname, salary}(\sigma_{DNO=5}(emp))$.
→ Sequence of operations:
eg: DEPS_EMP ← $\sigma_{DNO=5}$ (Emp)
RESULT ← $\pi_{Fname, Lname, salary}$ (DEPS_EMP).
→ Rename operators:
eg: TEMP ← $\sigma_{DNO=5}$ (emp)
R (First_name, Last_name, Salary) ← $\pi^{(Temp)}_{Fname, Lname, salary}$
Denoted by ρ

Date:

→ Renames both relation & attributes

$$\rho_{S(B1, B2, \ldots, Bn)}(R)$$

→ Renames relation only

$$\rho_S(R)$$

→ Renames attributes only

$$\rho_{(B1, B2, \ldots, Bn)}(R)$$

* Set Theory Operations:

~~"Union Compatible"~~ OR "Type Compatible"

Relations must have same type of tuples called "Type Compatible".

o— Union: R ∪ S. Tuples that are either in R or S or in both. Eliminate duplicate

eg: $Deps\_emps \leftarrow \sigma_{DNO=5}(Employee)$

$Result1 \leftarrow \pi_{ssn}(Deps\_emps)$

$Result2 \leftarrow \pi_{super\_ssn}(Deps\_emps)$

$Result \leftarrow Result1 \cup Result2$.

o— Intersection: R ∩ S. Tuples that are in R and S.

o— Set Difference: R − S. Tuples of R but not S.

$R \cup S = S \cup R$ and $R \cap S = S \cap R$

Union & intersection are commutative as well as associative.

$$R \cap S = ((R \cup S) - (R - S)) - (S - R)$$

Union all, intersection all, except all → do not eliminate duplicates.

A = {a, b, c} → (n)
B = {1, 2, 3} (m)

A × B = {(a,1), (a,2), (a,3), (b,1), (b,2), (b,3), (c,1), (c,2), (c,3)}
n+m attributes
n×m tuples

③

Date:

Cartesian Product (Cross Product) (Cross Join)

→ Binary set operations
→ But relations do not have to be Union compatible.
→ Produces a new element by combining every member (tuple) from one relation (set) with every member (tuple) from other relation (set)
→ Attributes → (n+m)
→ Tuples → (n × m)

eg: List of name of female emp's dependents.

FEMALE_EMPS ← $\sigma_{sex='F'}$ (Employee)

EMPNAMES ← $\pi_{Fname, Lname, ssn}$ (FEMALE_EMPS)

EMP_DEPENDENTS ← EMPNAMES × DEPENDENT

ACTUAL_DEPENDENTS ← $\sigma_{ssn=Essn}$ (EMP_DEPENDENTS)

RESULT ← $\pi_{Fname, Lname, Dependent\ name}$ (Actual_Dependent)

ACTUAL_Dependents ← EMPNAMES $\bowtie_{ssn=Essn}$ DEPENDENTS

* Binary Relational Operations:

① JOIN Operation.

→ Denoted by ⋈
→ Combine related tuples into single longer tuples from two relations

eg: Retreive name of manager of each department.

DEPT_MGR ← Department $\bowtie_{mgr\_ssn = ssn}$ Employee

Result ← $\pi_{dname, lname, fname}$ (Dept_mgr)

→ Attributes (n+m).

Date:

General join condition

<condition> And <condition> And ... And <condition>

$R \leftarrow (A_i \; \theta \; B_j) \rightarrow S \quad \theta \in \{=, >, <, \leq, \geq\}$

→ Join operation with general join condition is called Theta Join.

(2)

Equi Join:

→ Where the only comparision operator used is "="

Natural Join:

→ Denoted by *

→ Created to get rid of second attribute in Equi Join

→ Requires that two join attributes have same name in both relations

→ If attributes do not have same name, a Renaming operation is applied first

eg:- $Proj\_dept \leftarrow Project * P_{(dname, dnum, mgr\_ssn, mgr\_startdate)}(Department)$

here dnum is Join attribute.

Join selectivity: Ratio of expected size of join divided by maximum size $n \times m$.

Inner Join:

→ Combine data from two relations so that related information can be presented in single table

(3) Complete Set of Relational Algebra Operations:

$\{\sigma, \pi, \cup, \rho, -, \times\}$

→ $R \cap S \equiv (R \cup S) - ((R - S) \cup (S - R))$

→ $R \bowtie_{<condition>} S \equiv \sigma_{<condition>} (R \times S)$.

$\pi_{sid}^{(enroll-}(\pi_{sid}((\pi_{sid}(enrollment) \times \pi_{cid}(course)) - (enrollment)))$

(5)

S_1C_1 S_1E_2 S_2C_1 S_2C_2 S_3C_1 S_3C_2 − S_1C_1 S_2C_1 S_1C_2 S_3C_2

Date:

④ Division Operation: → for all, every

→ denoted by ÷ → A(x, y) / B(y) eg: E(sid, cid)/c(cid)

eg: Retrieve names of all employees who work on all the projects that 'JohnSmith' works on.

SMITH ← $\sigma_{Fname='John' \text{ And } Lname='Smith'}$ (Employee)

SMITH_PNOS ← π_{pno} (WORKS_ON ⋈ $_{Essn=ssn}$ SMITH)

SSN_PNOS ← $\pi_{Essn, Pno}$ (WORKS_ON)

SSNS(SSn) ← SSN_PNOS ÷ SMITH_PNOS

RESULT ← $\pi_{Fname, Lname}$ (SSNS * Employee)

* Additional Relational Operations:

① Generalized Projection:

→ Extends projection operation, allowing functions of attributes to be included in the projection list

eg:

Report ← P(ssn, net_salary, bons, tax)

($\pi_{ssn, salary-deduction, 2000*years\_service, 0.25*sal}$ (Employee))

② Aggregate Functions & Groupings:

<grouping attributes> F <functionlist> (R)

eg: $_{dno}$ F count(ssn), avg(salary) (Employee)

≈ select count(ssn), avg(salary) from employee group by dno.

NOTE: Duplicates are not eliminated & NULL values are not considered.