Procedural / non-procedural

Relational Algebra, Relational Calculus] SQL

- Relation → Relation
  Input → Output — (temporary table hold data asked by user,
- Language for express relational database queries → operations

R1, R2 > op → R3

Basic / Fundamental — Binary, Unary
Additional / derived — Binary, Unary

* Relational algebra work once on whole table.
Perform recursively on relation.

It fetch data from entire table.

Basic

σ	selection	Unary — only on 1 table
π	Projection	
ρ	Rename	

Binary

$\cup$	Union
$-$	SET DIFFERANCE
(X)	CARTESIAN Product

Derived — Binary

- Natural Join ($\bowtie$)
- left, right, full ($⟕$, $⟖$, $⟗$)
- Intersection ($\cap$)
- Division ($\div$)
- Assignment ($\leftarrow$)

SELECTION σ

FAST

$\sigma_{name='D'}$ (Student)

select ↓ not like sql query, sql clause where ~

~~$\sigma_{column\ condition}$~~ (table-name)

works on row

$\sigma_{age>17\ \wedge\ address="Delhi"}$ (Student)

∧ → ∨

$\sigma_{subject="DB"\ \wedge\ price=450\ \vee\ year>2010}$ (Books)

Projection π

(~~row~~) column works

$\pi_{rollno, name}$ (Student) — duplicate row eliminate

π_{column} (Table) — It will delete duplicate in column

Display rollno & name of student whose age is > 17

$\pi_{rollno, name}$ ($\sigma_{age>17}$ (STUDENT))

question always start from left side.

← Project always left side

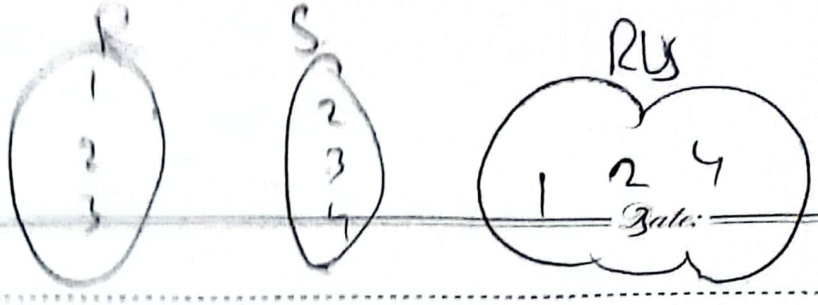
SET Operator

- Union — fundamental operator
- Set intersection — Derived operator
- Set difference — Fundamental operator

SET Operator → on two input, two relation must

→ be compatible [same no. of rows & columns; column are of same type

→ Duplicate tuple eliminated.



Notation RUS
In sql (SELECT * FROM R
UNION
SELECT * FROMS);

Union commulative
A∪B = B∪A

Remove Duplicate; π_{name}(STUDENT ∪ π_{name}(Employee)

(∩)

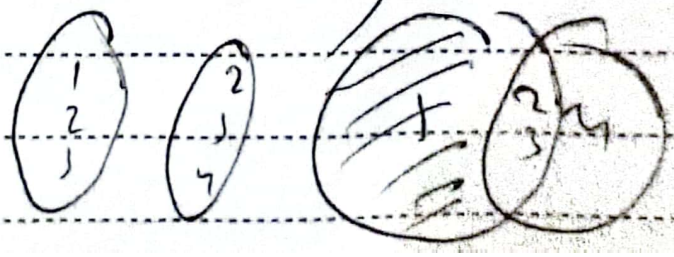
R∩S (SQL) (SELECT * FROM R
INTERSECT
SELECT * FROM S);

Find name of authy who have written a book & articles both.

π_{author}(BOOK) ∩ π_{author}(Articles)

SET Diffnt (Prent in R not in S)

R-S



In SQL (SELECT * FROM R) EXCEPT (SELECT * FROM S)

(STUDENT - Employee)

STUDENT	
1	A
2	B
3	C
4	D

Employee	
2	B
8	G
9	H

non commutative $A - B \neq B - A$

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

Intersection can be derived from SET Difference

Find the names of author who have written books but not articles

$\Pi_{author}(Books) - \Pi_{author}(Articles)$

~~Π_{name}~~ $* \Pi_{name}(\sigma_{age \geq 25}(User))$

$* \Pi_{*}(\sigma_{id > 2 \vee age = 21}(User))$

(cross join) — All pairs

X $R_1 \times R_2$ (Select * FROM R_1, R_2)

* Cartesian product → followed by select operation

* JOIN operator & its types

cartesian + selection — cartesian product or derive c

Join

⋈ $A \bowtie_c B = \sigma_c(A \times B)$

Inner join → tupple matchy condition

Outer join → condition match + other also

Inner Join

1) Theta / conditional Join

$A \bowtie_\theta B$ <, >, >=, ≠

2) Equi Join

type of theta join

use = equality

3) Natural Join

$A \bowtie B$

Based on common attribute — not common

Outer join

1) Left Outer Join ⟕

2) Right Outer Join ⟖

3) Full outer Join ⟗

Theta / Conditional Join

$A \bowtie_{\theta} B = \sigma_{\theta}(A \times B)$

θ → condition

>, ≥, =, ≠

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

$\sigma_{S1.sid < R1.sid}(S1 \times R1)$

Equi Join

$A \bowtie_{A1 = B1 \wedge \ldots \wedge A_n = B_n} B$

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

Natural Join

at least 1 common column

common ہو name or domain/type

بھی same ہو

$A \bowtie B$

Implicitly

condition کے لگا

same ہوگا

like equi join

E → show نہیں ہوگا 2 بار common

By default inner join

Date:

Natural Join = cartesian + Selection + Projection
product

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

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

Outer join

Left outer join courses ⟕ HOD

Right outer join $R_1 \bowtie R_2$

Full outer join $R_1 \bowtie R_2$

Rename operator → unary

$\rho_x(E)$ — relation; x → new name; ρ is use to rename operator.

$R \times R$ — ambiguity

In sql As

$R \times \rho_s(R)$

* Generalize Projection: Projection + Arithmetic operation

$\pi_{F_1, F_2 \ldots F_n}(E)$ — arithmetic operator

$\pi_{A, C, B}(r)$

$\pi_{2C-B}(r)$

* Aggregate function

avg, min, count, sum

ignore null value → entire relation → group

count not ignore

$_{G_1, G_2 \ldots G_n} g_{F_1(A_1), F_2(A_2) \ldots F_n(A_n)}(E)$

aggregate funt — column name — table

Date:

$g_{sum(c)}(r)$

group by branch name — $_{branchname}g_{sum(balance)}(account)$

↓ rename

$_{branchname}g_{sum(balance)\ as\ sumbal}(account)$

↓ we first rename as in it.

insertion

1) $\pi_{name}(\sigma_{address=lahore}(Student))$

2) $\pi_{name}(\sigma_{teachingsubj=DBMS}(Teacher))$

3) $\pi_{name}(\sigma_{studentName=Dunyamul}(Teacher \bowtie_{Teacher.tid=Student.tid} Student))$

Teacher Student

4) $T \leftarrow T \cup \{$ "Student" (xyz) $\}$

Deleted

5) $S \leftarrow S - (\sigma_{address=\text{"}pk\text{"}}(student))$

FAST

(Assignment operator)

convenient way to express

$t1 \leftarrow$ ~~~

$t2 \leftarrow$ ~~~

$R \leftarrow t1 - t2$

$temp1 \leftarrow \pi_x(A)$

$temp2 \leftarrow \pi_x((temp1 \times B) - R)$

$result \leftarrow temp1 - temp2$

1) $\pi_{*} (\sigma_{src = 'Dhn' \wedge dest = 'Lat'} (bus))$

2) π_{Pname} count (busy ⋈ booking ⋈ Pass)

$\pi_{Pname} (g_{count(*)} \text{ as } s1 (\text{booking} \bowtie \text{pass}))$

s1

3) $\pi_{bid}\ \sigma_{bus.passenger=2} \ldots Arr\ Time < 20/5/2021$ (passenger ⋈ bus schedule) date 29/5/2021

4) $\pi_{passName}$ (pass ⋈ booking)

2) π_{name} (passen ⋈ booking)

$\pi_{Pname} (\sigma_{bid = NULL} (passen ⟗ booking))$

d) $\sigma_{PBT\ Thr}$ (passenger ⋈ agent ⋈ booking)