Relational Algebra

DBMS Architecture

How does a SQL engine work ?

- SQL query → relational algebra plan
- Relational algebra plan → Optimized plan
- Execute each operator of the plan

Relational Algebra

- Five operators:
  - Union: $\cup$
  - Difference: -
  - Selection: σ
  - Projection: Π
  - Cartesian Product: $\times$
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural,equi-join, theta join, semi-join)
  - Renaming: ρ

1. Union and 2. Difference

- R1 ∪ R2
- Example:
  - ActiveEmployees ∪ RetiredEmployees

- R1 – R2
- Example:
  - AllEmployees -- RetiredEmployees

What about Intersection ?

- It is a derived operator
- R1 $\cap$ R2 = R1 – (R1 – R2)
- Also expressed as a join (will see later)
- Example
  - – UnionizedEmployees $\cap$ RetiredEmployees

3. Selection

- Returns all tuples which satisfy a condition
- Notation: $\sigma_c(R)$
- Examples
  - $\sigma_{\text{Salary > 40000}}$ (Employee)
  - $\sigma_{\text{name = “Smith”}}$ (Employee)
- The condition c can be =, <, $\leq$, >, $\geq$, <>

SSN	Name	Salary
1234545	John	200000
5423341	Smith	600000
4352342	Fred	500000

$\sigma_{\text{Salary > 40000}}$ (Employee)

SSN	Name	Salary
5423341	Smith	600000
4352342	Fred	500000

4. Projection

- Eliminates columns, then removes duplicates
- Notation: $\Pi_{A1,\ldots,An}(R)$
- Example: project social-security number and names:
  - $\Pi_{SSN, Name}$ (Employee)
  - Output schema: Answer(SSN, Name)

SSN	Name	Salary
1234545	John	200000
5423341	John	600000
4352342	John	200000

$\Pi_{\text{Name,Salary}}$ (Employee)

Name	Salary
John	20000
John	60000

5. Cartesian Product

- Each tuple in R1 with each tuple in R2
- Notation: R1 × R2
- Example:
  - Employee × Dependents
- Very rare in practice; mainly used to express joins

Cartesian Product Example

Employee

Name	SSN
John	999999999
Tony	777777777

Dependents

EmployeeSSN	Dname
999999999	Emily
777777777	Joe

Employee x Dependents

Name	SSN	EmployeeSSN	Dname
John	999999999	999999999	Emily
John	999999999	777777777	Joe
Tony	777777777	999999999	Emily
Tony	777777777	777777777	Joe

Renaming

- Changes the schema, not the instance
- Notation: $\rho_{B1,\ldots,Bn}$ (R)
- Example:
  - $\rho_{\text{LastName, SocSocNo}}$ (Employee)
  - Output schema: Answer(LastName, SocSocNo)

Renaming Example

Employee

Name	SSN
John	999999999
Tony	777777777

$$\rho_{LastName,\ SocSocNo}\ (\textbf{Employee})$$

LastName	SocSocNo
John	999999999
Tony	777777777

Natural Join

- Notation: R1 $|\times|$ R2

- Meaning: R1 $|\times|$ R2 = $\Pi_A(\sigma_C(R1 \times R2))$

- Where:
  - The selection σ_C checks equality of all common attributes
  - The projection eliminates the duplicate common attributes

Natural Join Example

Employee

Name	SSN
John	999999999
Tony	777777777

Dependents

SSN	Dname
999999999	Emily
777777777	Joe

Employee $\bowtie$ Dependents =

$\Pi_{\text{Name, SSN, Dname}}(\sigma_{\text{SSN=SSN2}}(\text{Employee x } \rho_{\text{SSN2, Dname}}(\text{Dependents}))$

Name	SSN	Dname
John	999999999	Emily
Tony	777777777	Joe

Natural Join

- R=

A	B
X	Y
X	Z
Y	Z
Z	V

S=

B	C
Z	U
V	W
Z	V

- R |×| S=

A	B	C
X	Z	U
X	Z	V
Y	Z	U
Y	Z	V
Z	V	W

Natural Join

- Given the schemas R(A, B, C, D), S(A, C, E), what is the schema of R |×| S ?

- Given R(A, B, C), S(D, E), what is R |×| S ?

- Given R(A, B), S(A, B), what is R |×| S ?

Theta Join

- A join that involves a predicate
- R1 $|\times|_{\theta}$ R2 = σ_{θ} (R1 $\times$ R2)
- Here θ can be any condition

Eq-join

- A theta join where θ is an equality
- R1 $|\times|_{A=B}$ R2 $= \sigma_{A=B}$ (R1 × R2)
- Example:
  - Employee $|\times|_{SSN=SSN}$ Dependents

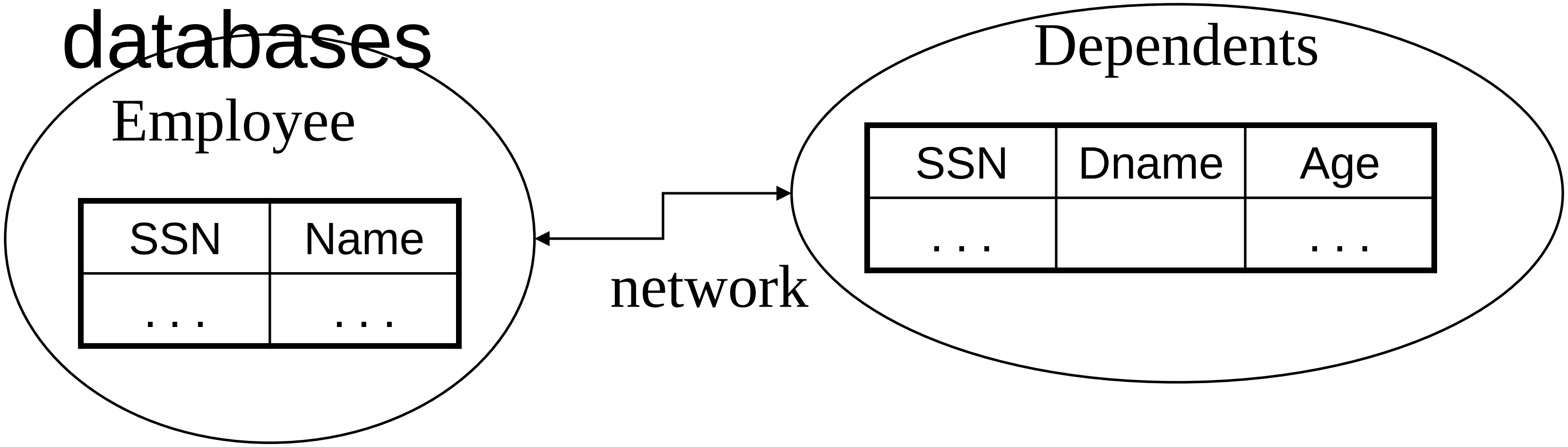
- Most useful join in practice

Semijoin

- R |× S = $\Pi_{A1,\ldots,An}$ (R |×| S)
- Where $A_1, \ldots, A_n$ are the attributes in R
- Example:
  - Employee |× Dependents

Semijoins in Distributed Databases

- Semijoins are used in distributed databases



SSN	Name
. . .	. . .

SSN	Dname	Age
. . .		. . .

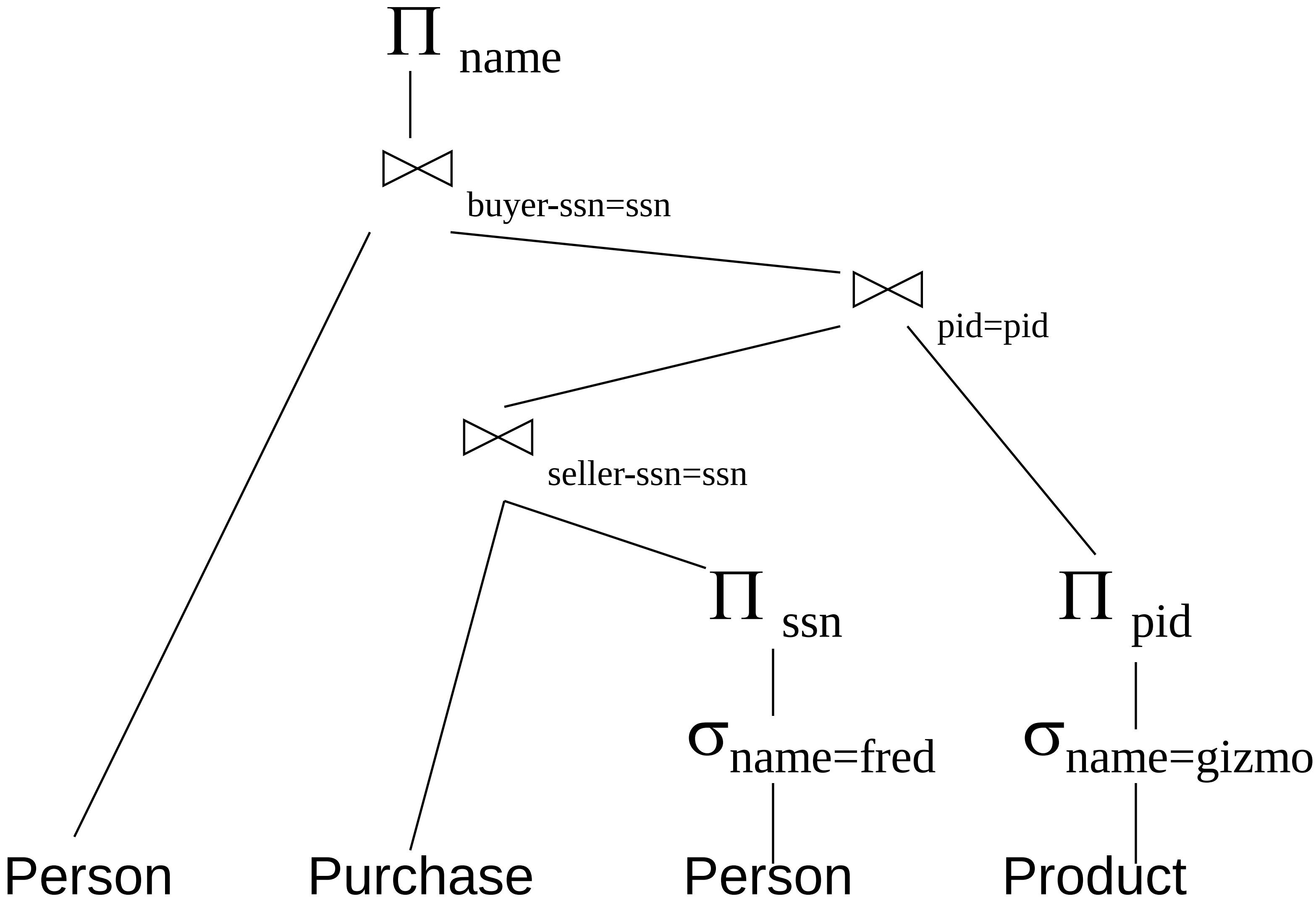
$$\boxed{\text{Employee} \ltimes_{ssn=ssn} (\sigma_{age>71}(\text{Dependents}))}$$

$T = \Pi_{SSN}\ \sigma_{age>71}(\text{Dependents})$

$R = \text{Employee} \ltimes T$

$\text{Answer} = R \bowtie \text{Dependents}$

Complex RA Expressions



Π_{name}

$\bowtie_{\text{buyer-ssn=ssn}}$

$\bowtie_{\text{pid=pid}}$

$\bowtie_{\text{seller-ssn=ssn}}$

Π_{ssn}

Π_{pid}

$\sigma_{\text{name=fred}}$

$\sigma_{\text{name=gizmo}}$

Person

Purchase

Person

Product

Operations on Bags

A **bag** = a set with repeated elements

All operations need to be defined carefully on bags

- $\{a,b,b,c\} \cup \{a,b,b,b,e,f,f\} = \{a,a,b,b,b,b,b,c,e,f,f\}$
- $\{a,b,b,b,c,c\} – \{b,c,c,c,d\} = \{a,b,b,d\}$
- $\sigma_C(R)$: preserve the number of occurrences
- $\Pi_A(R)$: no duplicate elimination
- Cartesian product, join: no duplicate elimination

Important ! Relational Engines work on bags, not sets !

Finally: RA has Limitations !

- Cannot compute “transitive closure”

Name1	Name2	Relationship
Fred	Mary	Father
Mary	Joe	Cousin
Mary	Bill	Spouse
Nancy	Lou	Sister

- Find all direct and indirect relatives of Fred
- Cannot express in RA !!! Need to write C program