Relational Algebra

Dr. Munesh Singh

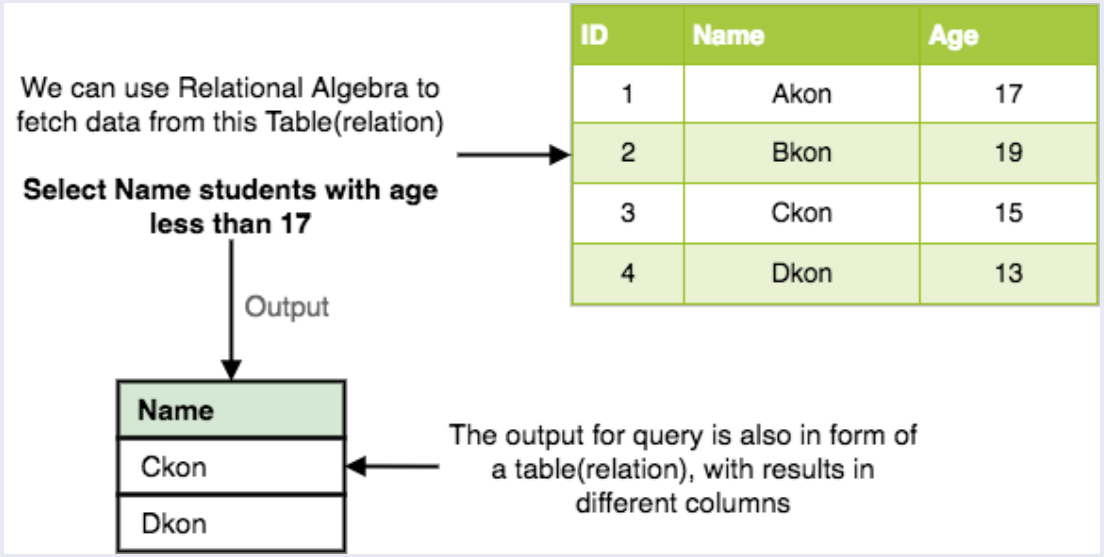
Indian Institute of Information Technology
Design and Manufacturing,
Kancheepuram
Chennai-600127

January 31, 2019

What is Relational Algebra?

Database

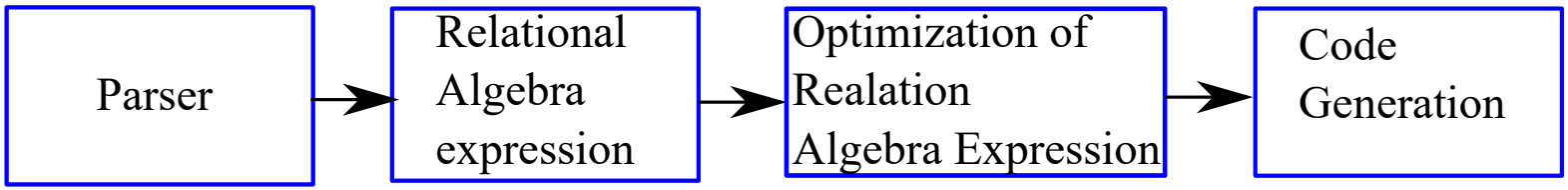
- Every database management system must define a query language to allow users to access the data stored in the database.
- Relational Algebra is a procedural query language used to query the database tables to access data in different ways.



Primary Operation

- The primary operations that we can perform using relational algebra are:

1. Select (σ)
2. Project (Π)
3. Union ($\cup$)
4. Set Different (-)
5. Intersection ($\cap$)
6. Cartesian Product (X)
7. Rename (ρ)



Basic Relational Algebra Operations

- Relational Algebra divides in various groups:
- Unary Relation Operations
  - SELECT
  - PROJECT
  - RENAME
- Relational Algebra Operation From Set Theory
  - UNION
  - INTERSECTION
  - DIFFERENCE
  - CARTESIAN PRODUCT
- Binary Relation Operations
  - JOIN
  - DIVISION

SELECT (σ)

- The SELECT operation is used for selecting a subset of the tuples according to a given selection condition
- Select operation selects tuples that satisfy a given predicate.
- $\sigma_p(r)$
- σ is the predicate
- **r** stands for relation which is the name of the table
- **p** is preposition logic

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Select all loan tuples where branch name is Hyderabad
- $\sigma_{bname=" Hyderabad"}(Loan)$
- Select all loan tuples where branch name is "Hyderabad" and loan amount greater than 5000

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Select all loan tuples where branch name is Hyderabad
- $\sigma_{bname=\text{"}Hyderabad\text{"}}(Loan)$
- Select all loan tuples where branch name is "Hyderabad" and loan amount greater than 5000
- $\sigma_{bname=\text{"}Hyderabad\text{"} \wedge amt>5000}(Loan)$

Projection (Π)

- The projection method defines a relation that contains a vertical subset of Relation.
- This helps to extract the values of specified attributes to eliminates duplicate values.
- $\Pi_{A1,A2,..An}(r)$
- Π is the predicate
- **r** stands for relation which is the name of the table
- **A1,A2,...An** is Attributes

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- List all loan number and the amount of the loan
- $\Pi_{Lno,amt}(Loan)$
- Find those customers who lives in "pune"

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- List all loan number and the amount of the loan
- $\Pi_{Lno,amt}(Loan)$
- Find those customers who lives in "pune"
- $\Pi_{cname}(\sigma_{city="pune"}(customer))$

Selection and Projection Query

Example

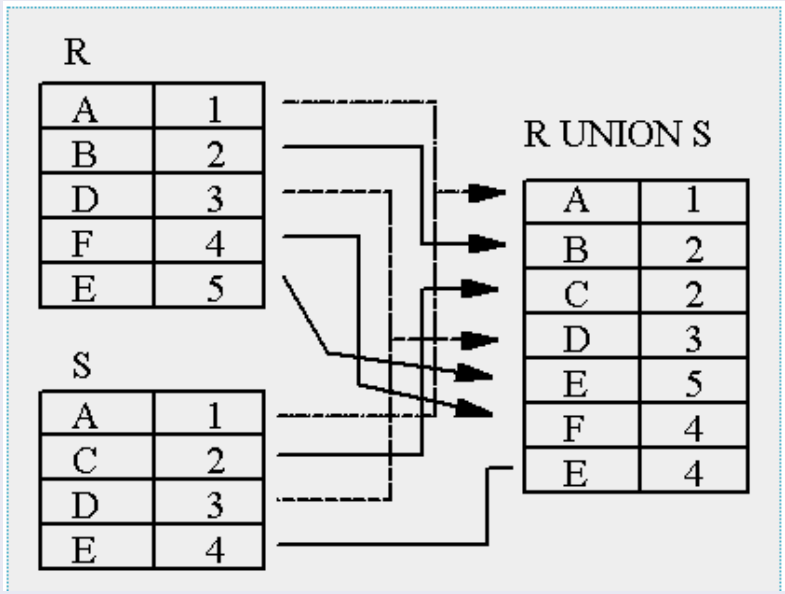
- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Find the account no where balance is less than 1000?
- Find those loan number which are from SBI kandagi branch with amount>1000
- Find branch name and branch city with assert more than 100000.

Union (∪)

- It includes all tuples that are in tables A or in B.
- For a union operation to be valid, the following conditions must hold:
  - R and S must be the same number of attributes
  - Attribute domain need to be compatible.
  - Duplicate tuples should be automatically removed



Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Find the name of all banks customers who have either an account or a loan or both
- $\Pi_{cname}(borrower)$
- $\Pi_{cname}(depositior)$
- $\Pi_{cname}(borrower) \cup \Pi_{cname}(depositior)$

Set Difference (-)

- The result of A - B, is a relation which includes all tuples that are in A but not in B.
- The two-operand relations A and B should be either compatible or Union compatible.
- It should be defined relation consisting of the tuples that are in relation A, but not in B.

R

A	1
B	2
D	3
F	4
E	5

S

A	1
C	2
D	3
E	4

R DIFFERENCE S

B	2
F	4
E	5

S DIFFERENCE R

C	2
E	4

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- List all customer names who are having an account but no loan from the bank
- $\Pi_{cname}(borrower)$
- $\Pi_{cname}(depositior)$
- $\Pi_{cname}(depositor) - \Pi_{cname}(borrower)$

Intersection (∩)

- Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.
- The attribute name of A has to match with the attribute name in B.

R

A	1
B	2
D	3
F	4
E	5

S

A	1
C	2
D	3
E	4

R INTERSECTION S

A	1
D	3

Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Find all customer who have a loan and account both on the bank
- $\Pi_{cname}(borrower)$
- $\Pi_{cname}(depositior)$
- $\Pi_{cname}(depositor) \cap \Pi_{cname}(borrower)$

Cartesian Product (X)

Example

- Find customer name who has taken the loan from "pune" branch
- $\Pi_{cname}(\sigma_{borrower.Lno=Loan.Lno \wedge bname="pune"}(borrower\ X\ loan))$

borrower

cname	Lno
c1	l1
c2	l2
c3	l2

loan

Lno	bname	amt
l1	pune	400000
l2	noida	600000

borrowerx loan

cname	borr. Lno	loan Lno	Bname	amt
c1	l1	l1	pune	400000
c1	l1	l2	noida	600000
c2	l2	l1	pune	400000
c2	l2	l2	noida	600000
c3	l2	l1	pune	400000
c3	l2	l2	noida	600000

Rename Operation (ρ)

- The results of relational algebra are also relations but without any name.
- The rename operation allows us to rename the output relation
- Denotes: $\rho_{newr}(oldr)$, $\rho_{newr(A1,A2,--An)}(oldr)$

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

Example

- Find out the maximum account balance in the bank
  $\Pi_{act.bal} - \Pi_{act.bal}(\sigma_{act.bal<a.bal}(account\ X\ \rho_a(account)))$

Acc

ano	bname	bal
a1	b1	100
a2	b2	500
a3	b3	400

Acc x p_a(Acc)

Acc.ano	Acc. bname	Acc.bal	a.ano	a.bname	a.bal
a1	b1	100	a1	b1	100
a1	b1	100	a2	b2	500
a1	b1	100	a3	b3	400
a2	b2	500	a1	b1	100
a2	b2	500	a2	b2	500
a2	b2	500	a3	b3	400
a3	b3	400	a1	b1	100
a3	b3	400	a2	b2	500
a3	b3	400	a3	b3	400

a

Join Operations

- Join operation is essentially a cartesian product followed by a selection criterion.
- JOIN operation also allows joining variously related tuples from different relations.
- Types of Joins:
  - **Inner join**(outcome/result contains only the matching tuple)
  - Theta join
  - EQUI join
  - Natural join
  - **Outer join**(The result will contain all the tuples from one or both of the relation)
  - Left Outer join
  - Right Outer join
  - Full Outer join

Theta Join $\bowtie_\theta$ & Equi Join

- $(S)\bowtie_\theta(R)$, where θ is a condition
- θ condition included comparative operators $=, >, <, >=, <=, \neg$
- I want to purchase both desktop and laptop but the price of desktop should be lower than laptop
- $\Pi_{lpbrand,deskbrand}((Desktop)\bowtie_{desktop.price<laptop.price}(Laptop))$

Desktop

desk brand	price
HP	50K
LG	40K
Acer	30K
Dell	70K

Laptop

lp brand	price
Dell	40k
Asus	50k
Mac	90k
Lenovo	30k

Natural Join ⋈

Student

sno	sname	sage
s1	n1	21
s2	n2	22
s3	n3	23
s4	n4	24

Course

sno	cname	fees
s1	c1	1000
s3	c3	3000
s5	c5	5000

sno	sname	sage	cname	fees
s1	n1	21	c1	1000
s3	n3	23	c3	3000

Left Outer Join

Student

sno	sname	sage
s1	n1	21
s2	n2	22
s3	n3	23
s4	n4	24

Course

sno	cname	fees
s1	c1	1000
s3	c3	3000
s5	c5	5000

sno	sname	sage	cname	fees
s1	n1	21	c1	1000
s2	n2	22	Null	Null
s3	n3	23	c3	3000
s4	n4	24	Null	Null

Right Outer Join

Student

sno	sname	sage
s1	n1	21
s2	n2	22
s3	n3	23
s4	n4	24

Course

sno	cname	fees
s1	c1	1000
s3	c3	3000
s5	c5	5000

sno	sname	sage	cname	fees
s1	n1	21	c1	1000
s3	n3	23	c3	3000
s5	Null	Null	c5	5000

Full Outer Join

Student

sno	sname	sage
s1	n1	21
s2	n2	22
s3	n3	23
s4	n4	24

Course

sno	cname	fees
s1	c1	1000
s3	c3	3000
s5	c5	5000

sno	sname	sage	cname	fees
s1	n1	21	c1	1000
s2	n2	22	Null	Null
s3	n3	23	c3	3000
s4	n4	24	Null	Null
s5	Null	Null	c5	5000

Division Operation

- The emp who works on all the project in company
- **Def:** if R(x)=R1(z) % R2(y) then relation R(x) is all the tuples t(x) in R1(z) that appears in R1 in combination with every tuple from R2(y), where Z=xUy.
- Result contain the attribute (R1-R2)

R1

sno	pno
s1	p1
s1	p3
s1	p2
s2	p1
s1	p4
s2	p2
s4	p1
s4	p4

R2

p.no
p1
p4

R1%R2=? **R**

s.no
s1
s4

Division Example

Example

- Account(ano,bname,bal)
- Branch(bname,bcity,asset)
- Customer(cnamr,cstreet,ccity)
- Loan(Lno,bname,amt)
- Depositor(cnamr,ano)
- Borrower(cname,cno)

Example

- Find all customers who have account at all branches of Delhi
- r1=$\Pi_{bname}(\sigma_{bcity=\text{"}Delhi\text{"}}(branch))$
- r2=$\Pi_{cname,bname}(depositor \bowtie account)$
- $r = r1\%r2$

Class Assignment

Use these Relations to solve relational algebra quiries

- **employees**(eid,name,salary,did,mdid)
- **projects**(pid,description)
- **workson**(eid,pid, hours)
- **departments**(did, location)

Quiries

- List the name of the project that have employees from the systems department working less than 5 hours. Pid is also the name of the project.
- List the name of employees with salary greater than their manager's salary

Class Assignment

Quiries

- List the name of employees working on all projects.
- List the name of employees making more than 100,000 rs and working on zero projects
- List the name of employees working on both projectX and projectY