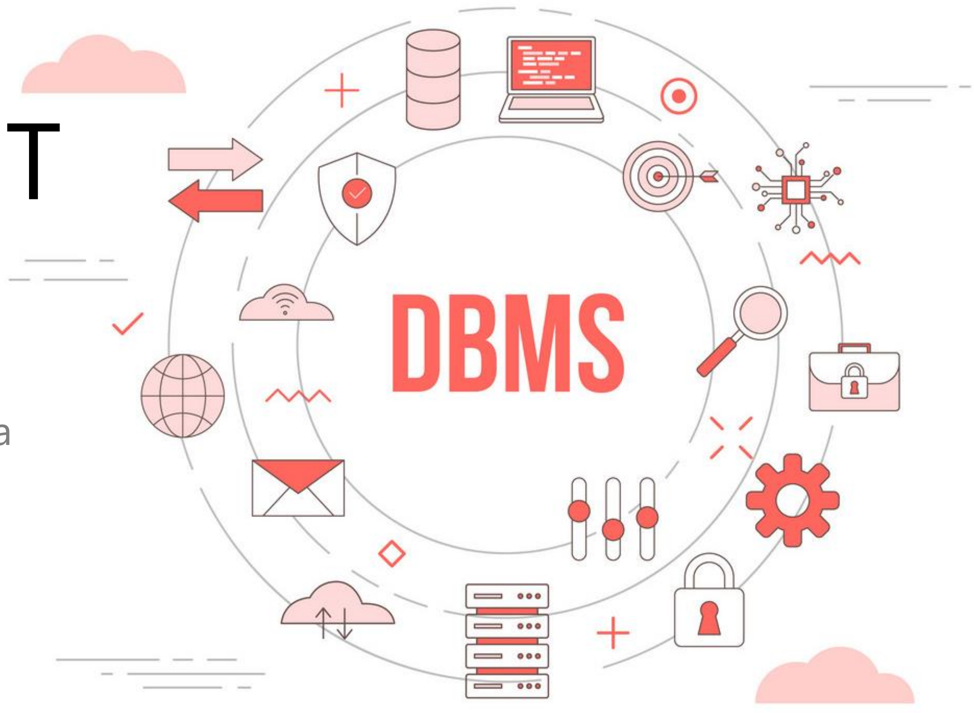
DATABASE MANAGEMENT SYSTEMS

Relational Algebra



Dr. Jay Sarraf
School of Computer Engineering
KIIT Deemed to be University

Overview

Edgar F. Codd introduced relationship algebra in 1970. (Father of DBMS).

It is often referred to as Procedural Query Language (PQL), since in PQL, a programmer or user must specify two things: "**What to Do**" and "**How to Do**."

- When we talk about relational algebra, we're talking about a procedural query language that accepts relation instances as input and outputs relation instances. It uses operators to carry out queries.

- An operator can be binary or unary. They create relations as an output and receive relations as an input. An application of recursive relational algebra is made to a relationship, and intermediate results are likewise regarded as relations.

$\pi \; \sigma \; \rho$

$\bowtie \; \cup \; \cap$

Relational Algebra Operators

Procedural vs Non Procedural

Procedural Language:

A series of instructions is used to represent the programme code in procedural languages. Both "**what to do**" and "**how to do**" must be specified by the user (step by step procedure). These directives are carried out in the correct order. These guidelines were created to address a specific issue.

Ex: FORTRAN, COBOL, ALGOL

Non Procedural Language:

The user simply has to indicate "**what to do**" and **not "how to do"** in non-procedural languages. It is sometimes referred to as a functional or applicative language. It entails building more complicated functionalities from the development of simpler functions.

Ex: SQL, PROLOG, LISP

Types of Operations in Relational Algebra

In Relational Algebra, we have two types of Operations.

Basic Operations & Derived Operations.
OR
Fundamental Operations & Secondary Operations

Basic / Fundamental Operators
Unary:- SELECT, PROJECT, RENAME
Binary:- UNION, SET DIFFERENCE, CARTESIAN PRODUCT

Derived / Secondary Operators
INTERSECTION, NATURAL JOIN, DIVISION, ASSIGNMENT

Cont…

Relational Schema

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR
S4	Dev	18	BBSR
S5	Eva	20	PURI

FACULTY

FID	FNAME	FAGE
F1	Prof A	40
F2	Prof B	38
F3	Prof C	50
F4	Prof D	38
F5	Prof E	40

SELECT Operator (**σ**)

The Selection Operator, represented by "sigma"**(σ)**, performs the select operation. In order to extract the tuples (rows) in the table when the specified criteria is met, this method is used.

The general syntax of select operator is: σ <selection−condition> (<relation name>)

Notation: σ p(r)
Where:
σ is used for selection prediction
r is used for relation
p is used as a propositional logic formula which may use connectors like:
AND OR and NOT. These relations can use as relational operators like =, ≠, ≥, <, >, ≤.

Cont…

Query:

Find the details of students whose age is ‘20’.

σ sage = 20 (STUDENTS)

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S3	Cyan	20	BBSR
S5	Eva	20	CITY

Query:
Find the details of students whose age is ‘20’ and are from city BBSR.

σ sage = 20 AND city= ‘BBSR’ (STUDENTS)

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S3	Cyan	20	BBSR

PROJECT Operator (π)

Projection Operator, denoted by "pi"(π), is responsible for project operation. Some characteristics (columns) from the table are retrieved using it. Because it divides the table vertically, it is sometimes referred to as vertical partitioning.

The general syntax of select operator is: **π** <attribute−list> (<relation name>)

Notation: π a(r)

Where:

π is used for projection

r is used for relation

a is used for attribute list.

Cont…

Query:
Find the students roll numbers and their respective names.

π sroll, sname (STUDENT)

SROLL	SNAME
S1	Avril
S2	Byril
S3	Cyan
S4	Dev
S5	Eva

Cont...

Composition of Relational Operators.

To respond to the complicated questions, relational algebra operators can be combined into an expression.

Query:

Find the names of students who live in BBSR.

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR
S4	Dev	18	BBSR
S5	Eva	20	PURI

$$\pi_{sname}(\sigma_{city = 'BBSR'}(STUDENT))$$

SNAME
Avril
Cyan
DEV

RENAME Operator (ρ)

The outcomes of relational algebra expressions do not have a name to which they may be referred.
The rename operation is marked by "Rho"(ρ). Its name implies that it is used to rename the output relation.

The general syntax of select operator is: ρ X (E)

Assuming E is a relational algebra expression with arity n.
The rename operation can also be expressed as ρ X(a1,a2,...an) (E)

Query:
Find the names of students who live in BBSR.

π sname (σ city = 'BBSR' (STUDENT))

can be written as:

ρ Student_Name *(σ city = 'BBSR' (STUDENT))*

π sname (Student_Name)

Cont…

Query:
Find the names of students who live in BBSR.

$$\pi_{sname}(\sigma_{city = 'BBSR'}(STUDENT))$$

can be written as:

$$\rho_{Roll, Name, Age, City}(STUDENT))$$

$$\pi_{Name}(\sigma_{City = 'BBSR'}(STUDENT))$$

Union Compatibility

Two or more tables(R1 U R2) are considered to be union-compatible if they have the same number of columns and their associated columns have the same or compatible domains.

Union (U) Operator

To merge data from two relations, use the union operation. It is represented by the symbol union( ).

R3 (c1,c2,... cn) is the union of two relations **R1(a1,a2,... an)** and **R2(b1,b2,... bn)** such that: domain(**ci**) = domain(**ai**) U domain(**bi**),1 $\leq$ i $\leq$ n

R1 U R2 is a relation that includes all tuples that are present in either R1 or R2 or both, but not duplicate tuples.

To perform the set operations such as UNION, DIFFERENCE and INTERSECTION, the relations need to be union compatible for the result to be a valid relation

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

π sroll, sname (STUDENTS) U π sroll, sname (ENROLLMENT)

SROLL	SNAME
S1	Avril
S2	Byril
S3	Cyan
S4	Dev
S5	Eva

Difference Operator (-)

Difference (-) Operator

It is represented with a (**-**) symbol.

R1 – R2 produces a relationship that includes all tuples in R1 but not in R2.

The name of R1's attribute must match the name of R2's attribute. R1 and R2's two-operand relations should be either compatible or Union compatible.

R3 (c1,c2,... cn) is the set difference of two relations **R1(a1,a2,... an)** and **R2(b1,b2,... bn)** such that: domain(**ci**) = domain(**ai**) - domain(**bi**),1 $\leq$ i $\leq$ n.

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

π sroll, sname (ENROLLMENT) - π sroll, sname (STUDENTS)

SROLL	SNAME
S4	Dev
S5	Eva

Cartesian Product Operator(×)

Cartesian Product Operator(×)

In DBMS, Cartesian Product is an operation that is used to integrate columns from two relations. When performed alone, a cartesian product is never a significant operation. It becomes meaningful, however, when it is followed by subsequent processes. It's also known as Cross Product or Cross Join.

The Cartesian product of two relations **R1(a1,a2,... an)** with cardinality **i** and **R2(b1,b2,... bm)** with cardinality **j** is a relation R3 with

- degree k = n + m,
- cardinality i*j and
- attributes (a1,a2,... an, b1,b2,... bm)

R1 x R2 is a relation that comprises all possible tuple combinations from R1 and R2. The Cartesian product may be used to integrate data from any two relationships.

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

STUDENTS x ENROLLMENT

Query:

Find the details of students who have taken enrolment in course 1.

σ cid = 'C1' AND student.sroll = enrolment.sroll (STUDENTS x ENROLLMENT)

Students. SROLL	Students. SNAME	Students. SAGE	Students. CITY	Enrollment .sroll	Enrollment .sname	FID	CID
S1	Avril	20	BBSR	S1	Avril	F1	C1

Cont…

STUDENTS x ENROLLMENT

Students. SROLL	Students. SNAME	Students. SAGE	Students. CITY	Enrollment .sroll	Enrollment .sname	FID	CID
S1	Avril	20	BBSR	S1	Avril	F1	C1
S1	Avril	20	BBSR	S2	Byril	F2	C2
S1	Avril	20	BBSR	S3	Cyan	F3	C3
S1	Avril	20	BBSR	S4	Dev	F4	C4
S1	Avril	20	BBSR	S5	Eva	F5	C5
S2	Byril	19	CTC	S1	Avril	F1	C1
S2	Byril	19	CTC	S2	Byril	F2	C2
S2	Byril	19	CTC	S3	Cyan	F3	C3
S2	Byril	19	CTC	S4	Dev	F4	C4
S2	Byril	19	CTC	S5	Eva	F5	C5
S3	Cyan	20	BBSR	S1	Avril	F1	C1
S3	Cyan	20	BBSR	S2	Byril	F2	C2
S3	Cyan	20	BBSR	S3	Cyan	F3	C3
S3	Cyan	20	BBSR	S4	Dev	F4	C4
S3	Cyan	20	BBSR	S5	Eva	F5	C5

Intersection Operator(∩)

Intersection Operator(∩)

The intersection operation is used to find rows that are shared by two relations. It is represented by the symbol (∩).

R1 ∩ R2 The result of this operation is a relation that includes all tuples that are in both R and S.

The name of R1's attribute must match the name of R2's attribute. R1 and R2's two-operand relations should be either compatible or Union compatible.

R3 (c1,c2,... cn) is the intersection of two relations **R1(a1,a2,... an)** and **R2(b1,b2,... bn)** such that: domain(**ci**) = domain(**ai**) ∩ domain(**bi**),1 $\leq$ i $\leq$ n.

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

π sroll, sname (STUDENTS) ∩ π sroll, sname (ENROLLMENT)

SROLL	SNAME
S1	Avril
S2	Byril
S3	Cyan

Join Operator(⋈)

JOIN Operator(⋈)

A Join operation merges related tuples from separate relations if and only if a specific join condition is met. It is denoted by (⋈).

The join operation creates a Cartesian product of its two parameters, then executes a selection requiring equality on the attributes that occur in both relations before removing the duplicate attributes.

Types of Join:

- **Inner Joins:**
  - Theta join
  - EQUI join
  - Natural join
- **Outer join:**
  - Left Outer Join
  - Right Outer Join
  - Full Outer Join

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

STUDENTS ⋈ ENROLLMENT

Query:

Find the details of students who have taken enrolment in course 1.

σ cid = 'C1' (STUDENTS ⋈ ENROLLMENT)

SROLL	SNAME	SAGE	CITY	FID	CID
S1	Avril	20	BBSR	F1	C1

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S4	Dev	F4	C4
S5	Eva	F5	C5

STUDENTS ⋈ ENROLLMENT

Query:

Find the NAME, ROLL AND AGE of students who have taken enrolment in course 1.

π sroll, sname,sage (σ cid = 'C1' (STUDENTS ⋈ ENROLLMENT))

SROLL	SNAME	SAGE
S1	Avril	20

DIVISION Operator(÷)

DIVISION Operator(÷)

The division operation generates a new relation by picking the Rows in one relation that match every row in another.

The division operation necessitates that we examine an entire relation at once. It is denoted by the division (÷) sign.

Division operator A÷B or A/B gives C, can be applied if and only if:

- Attributes of B are a valid subset of Attributes of A.
- The division operator will yield a relation with the properties = (All attributes of A – All Attributes of B)
- The relation provided by the division operator will return those tuples from relation A that are related with each tuple in relation B.

Cont…

STUDENTS

SROLL	SNAME	SAGE	CITY
S1	Avril	20	BBSR
S2	Byril	19	CTC
S3	Cyan	20	BBSR

ENROLLMENT

SROLL	SNAME	FID	CID
S1	Avril	F1	C1
S2	Byril	F2	C2
S3	Cyan	F3	C3
S1	Avril	F2	C2
S2	Byril	F2	C1
S1	Avril	F3	C3
S3	Cyan	F1	C1

COURSE

CID	CNAME
C1	DBMS
C2	OS
C3	FLA

Query:

Find the details of all students who have taken enrolment in all the courses.

(STUDENTS ⋈ ENROLLMENT)) ÷ π cid (COURSE))

Cont...

STUDENTS x ENROLLMENT

Students.SROLL	Students.SNAME	Students.SAGE	Students.CITY	Enrollment.sroll	Enrollment.sname	FID	CID
S1	**Avril**	**20**	**BBSR**	**S1**	**Avril**	**F1**	**C1**
~~S1~~	~~Avril~~	~~20~~	~~BBSR~~	~~S2~~	~~Byril~~	~~F2~~	~~C2~~
~~S1~~	~~Avril~~	~~20~~	~~BBSR~~	~~S3~~	~~Cyan~~	~~F3~~	~~C3~~
S1	**Avril**	**20**	**BBSR**	**S1**	**Avril**	**F2**	**C2**
~~S1~~	~~Avril~~	~~20~~	~~BBSR~~	~~S2~~	~~Byril~~	~~F2~~	~~C1~~
S1	**Avril**	**20**	**BBSR**	**S1**	**Avril**	**F3**	**C3**
~~S1~~	~~Avril~~	~~20~~	~~BBSR~~	~~S3~~	~~Cyan~~	~~F1~~	~~C1~~
~~S2~~	~~Byril~~	~~19~~	~~CTC~~	~~S1~~	~~Avril~~	~~F1~~	~~C1~~
S2	**Byril**	**19**	**CTC**	**S2**	**Byril**	**F2**	**C2**
~~S2~~	~~Byril~~	~~19~~	~~CTC~~	~~S3~~	~~Cyan~~	~~F3~~	~~C3~~
~~S2~~	~~Byril~~	~~19~~	~~CTC~~	~~S1~~	~~Avril~~	~~F2~~	~~C2~~
S2	**Byril**	**19**	**CTC**	**S2**	**Byril**	**F2**	**C1**
~~S2~~	~~Byril~~	~~19~~	~~CTC~~	~~S1~~	~~Avril~~	~~F3~~	~~C3~~
~~S2~~	~~Byril~~	~~19~~	~~CTC~~	~~S3~~	~~Cyan~~	~~F1~~	~~C1~~
~~S3~~	~~Cyan~~	~~20~~	~~BBSR~~	~~S1~~	~~Avril~~	~~F1~~	~~C1~~
~~S3~~	~~Cyan~~	~~20~~	~~BBSR~~	~~S2~~	~~Byril~~	~~F2~~	~~C2~~
S3	**Cyan**	**20**	**BBSR**	**S3**	**Cyan**	**F3**	**C3**
~~S3~~	~~Cyan~~	~~20~~	~~BBSR~~	~~S1~~	~~Avril~~	~~F2~~	~~C2~~
~~S3~~	~~Cyan~~	~~20~~	~~BBSR~~	~~S2~~	~~Byril~~	~~F2~~	~~C1~~
~~S3~~	~~Cyan~~	~~20~~	~~BBSR~~	~~S1~~	~~Avril~~	~~F3~~	~~C3~~
S3	**Cyan**	**20**	**BBSR**	**S3**	**Cyan**	**F1**	**C1**

Cont...

STUDENTS ⋈ ENROLLMENT

SROLL	SNAME	FID	CID	SAGE	CITY
S1	Avril	F1	C1	20	BBSR
S2	Byril	F2	C2	19	CTC
S3	Cyan	F3	C3	20	BBSR

COURSE

CID
C1
C2
C3

(STUDENTS ⋈ ENROLLMENT)) ÷ π cid (COURSE))

SROLL	SNAME	FID	SAGE	CITY
S1	Avril	F1	20	BBSR

Cont…

CUSTOMER

CName	AcNo
Avril	111
Byril	222
Cyan	333
Dev	444
Dev	555

ACCOUNT

AcNo	Bname	AcBalance
111	BBSR Main	10,000,00
222	CTC	1,00,000
333	Sambalpur	20,000
444	Kolkata Main	90,000
555	Salt Lake	1,00,000

BRANCH

Bname	BrCity
BBSR Main	Bhubaneswar
CTC	Cuttack
Sambalpur	Sambhalpur
Kolkata Main	Kolkata
Salt Lake	Kolkata

Query : Find all the customers who have an account at all the branches located in Kolkata.

π CName, Bname (Customer $\bowtie$ Account) $\div$ π Bname (σ BrCity='Kolkata'(Branch))

CNAME
Dev

Assignment Operator(←)

ASSIGNMENT Operator(←)

The assignment operation (←) makes it easy to describe sophisticated queries.
A temporary relation variable always uses assignment operator .

The result of the symbol on the right ← is allocated to the related variable on the symbol on the left ←.

A query may be expressed as a sequential program using the assignment operator, consisting of:

- a sequence of assignment,
- followed by an expression whose value is shown as a result of the query

Cont…

π CName, Bname (Customer ⋈ Account) ÷ π Bname (σ BrCity='Kolkata'(Branch))

Can be written using an assignment operataor

Temp1 ← π CName, Bname (Customer ⋈ Account)

Temp2 ← π Bname (σ BrCity='Kolkata'(Branch)

Tem1 ÷ Temp2 = **Results**

Generalized-projection

Generalized-projection

The projection operation is extended by the generalized-projection operation, which allows arithmetic functions to be utilised in the projection list. The generalized-projection formula is:

$\pi_{F1,F2...Fn}(E)$

Ex:Emp=(ssn, salary, deduction, years_service) be a relation.

A report may be required to show net_salary=salary-deduction, bonus=2000*years_service and tax=0.25*salary

REPORT ← ρ (ssn,net_salary,bonus,tax) (π ssn,salary−deduction, 2000∗years_service, 0.25∗salary (Emp))

Aggregate Functions(g)

Aggregate Functions(g)

Aggregate functions take a collection of values and return a single value as a result. NULL value will not participate in the aggregate functions. The general form of aggregate function is:

grouping_attribute **g** aggregate_functions (R)

Let Works = (emp_id, ename, salary, branch_name)

Query: Find the total sum of salaries of all the employees

Ans: g SUM(salary) (Works)

Query: Find the total sum of salaries of all the employees in each branch

Ans: branch_name g SUM(salary) (Works)

Query: Find the maximum salary for the employees at each branch, in addition to the sum of the salaries .

Ans: branch_name g SUM(salary),MAX(salary) (Works)

Query: Find the number of employees working

Ans: g COUNT(emp_id) (Works)