DBMS: Relational Algebra

Akhilesh Arya

Relational Models

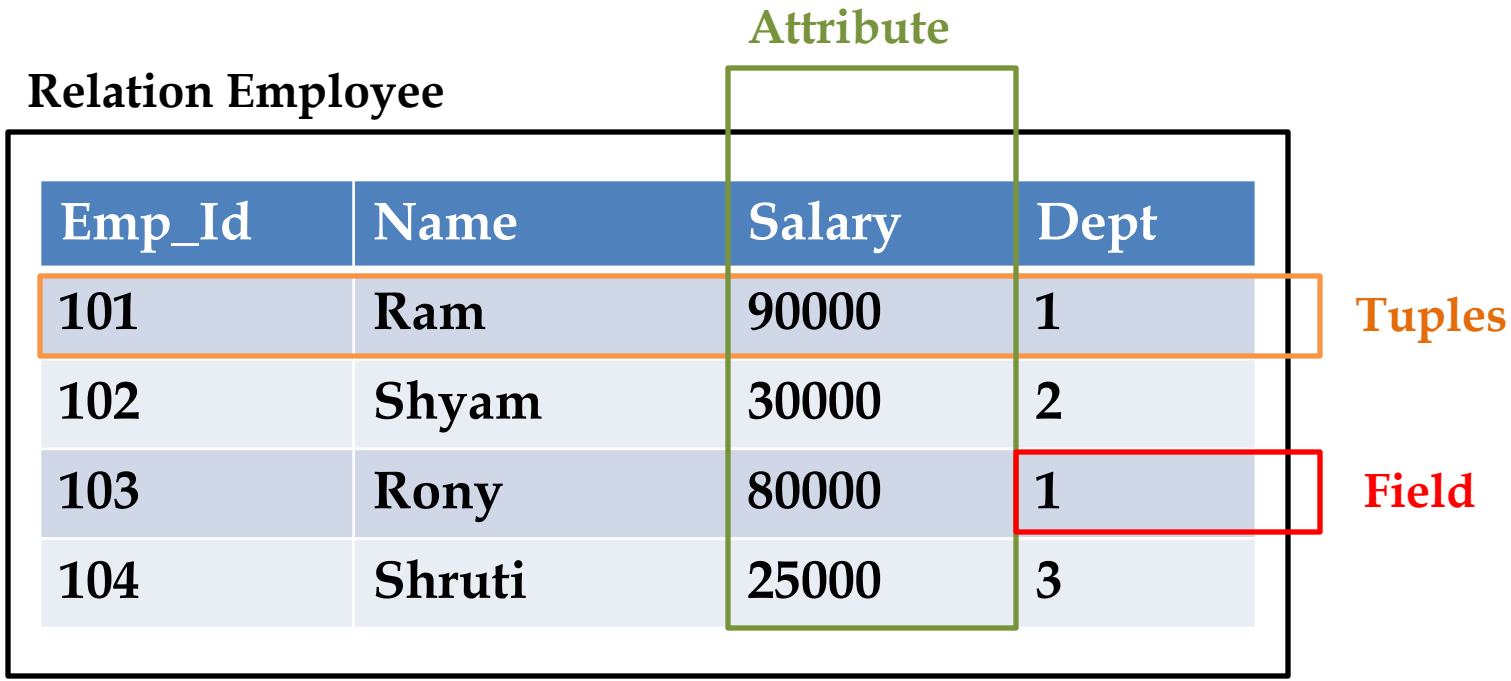
- Relational model can represent as a table with columns and rows.

Emp_Id	Name	Salary	Dept
101	Ram	9000	1
102	Shyam	10000	2
103	Rony	8000	1
104	Shruti	11000	3

Cont..

- **Relation:** is complete table that contains the inter-related data
- **Tuples:** the row in the relation is called as tuples
- **Attribute:** Each column in the relation is the attribute
- **Domain:** the set of all possible values that an attribute can have is the domain name
- **Degree or Arity:** total num of attributes in a relation

Example:

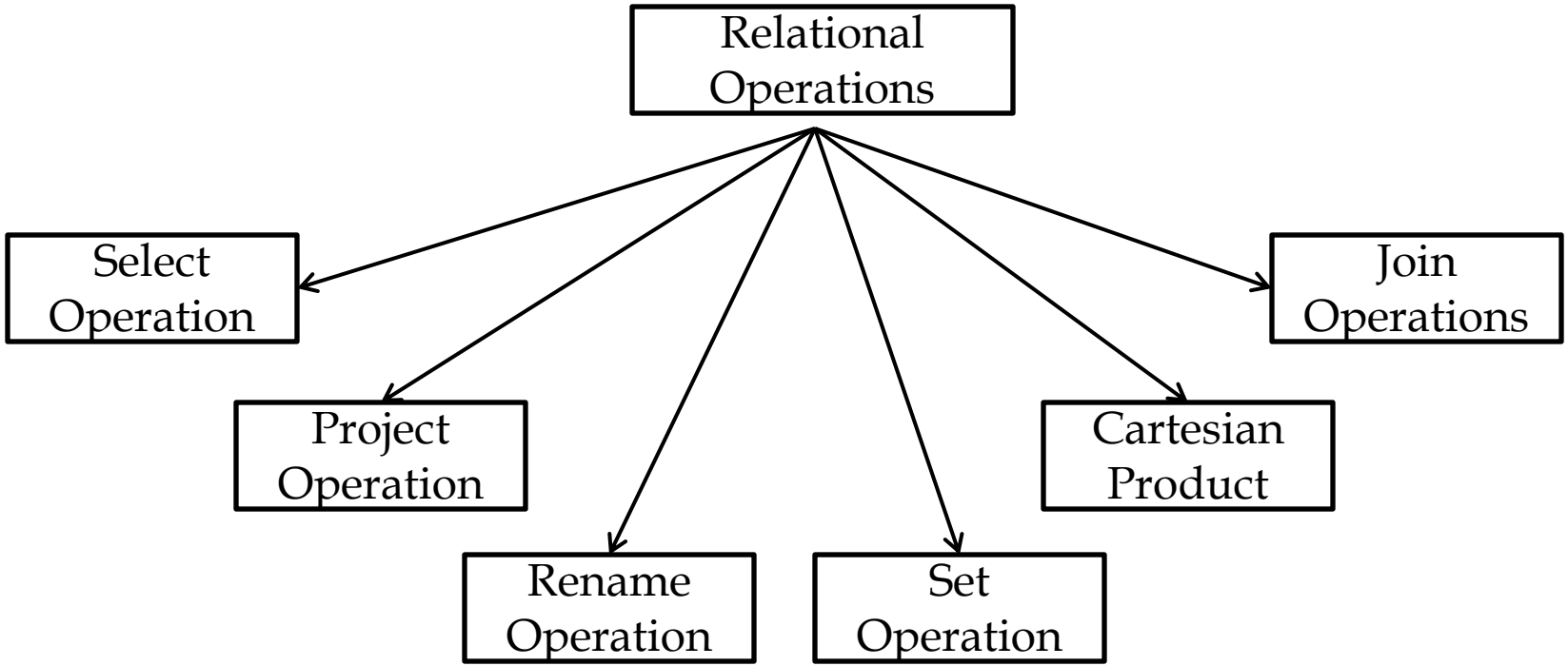


Domain of Emp_Id = All +ve num **Degree or arity = 4**

Relational Algebra

- Relational algebra is a procedural query language.
- It gives a step by step process to obtain the result of the query.
- It uses operators to perform queries
  - **Boolean:** AND, OR, and NOT
  - **Relational:** =, ≠, ≥, <, >, ≤

Types of Relational Operators



Select Operation

- Returns all tuples which satisfy a condition
- For those tuples that do not satisfy the conditions are discarded

$$\sigma_c(R)$$

- The condition c can be =, <, ≤, >, ≥, <>

Employee

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1
104	Shruti	25000	3

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

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
103	Rony	80000	1

Project Operation

- The project operation selects certain columns from the table and discards the other columns.

$$\Pi_{A1,\ldots,An}(R)$$

Employee

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1
104	Shruti	25000	3

$\Pi_{\text{Emp\_Id, Name}}$ (Employee)

Emp_Id	Name
101	Ram
102	Shyam
103	Rony
104	Shruti

Renaming Operation

- The resultant relation obtained from any relation algebra doe not have any name
- The rename (ρ) operator is used to assign names to such relations.

$$\rho_{Emp}(Employee)$$

- Now we can refer Employee relation with its new name Emp.

Set Operation

- UNION Operation
  - Will combine tuples of two union compatible relations
  - It eliminates duplicate elements and include tuple which are either in R1 or in R2

R1 U R2

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1
104	Shruti	25000	3

Employee Relation

$$\sigma_{Salary > 25000} (Employee) \cup \sigma_{Dept = 1} (Employee)$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1

$$\sigma_{\text{Salary} > 25000} (\text{Employee})$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
103	Rony	80000	1

$$\sigma_{\text{Dept} = 1} (\text{Employee})$$

$$\sigma_{Salary > 25000} (Employee) \cup \sigma_{Dept = 1} (Employee)$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1

Cont..

- Set Intersection
  - Will combine tuples of two set intersection compatible relations
  - It combines only common tuples from R1 and R2

$$R1 \cap R2$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1
104	Shruti	25000	3

Employee Relation

$$\sigma_{Salary > 25000}(Employee) \cap \sigma_{Dept = 1}(Employee)$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1

$$\sigma_{\text{Salary} > 25000} (\text{Employee})$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
103	Rony	80000	1

$$\sigma_{\text{Dept} = 1} (\text{Employee})$$

$$\sigma_{Salary > 25000}(Employee) \cap \sigma_{Dept = 1}(Employee)$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
103	Rony	80000	1

Cont..

- Set Difference
  - Will combine tuples of two set difference compatible relations
  - It allow us to find tuples that are in one relation but are not in another relation

R1 - R2

  - The set difference operation contains all tuples that are in R1 but not in R2.

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1
104	Shruti	25000	3

Employee Relation

$$\sigma_{Salary > 25000}(Employee) - \sigma_{Dept = 1}(Employee)$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
102	Shyam	30000	2
103	Rony	80000	1

$$\sigma_{\text{Salary} > 25000} (\text{Employee})$$

Emp_Id	Name	Salary	Dept
101	Ram	90000	1
103	Rony	80000	1

$$\sigma_{\text{Dept} = 1} (\text{Employee})$$

$$\sigma_{Salary > 25000}(Employee) - \sigma_{Dept = 1}(Employee)$$

Emp_Id	Name	Salary	Dept
102	Shyam	30000	2

Cartesion Product

- The cartesian product allows us to combine tuples from 2 relations in combinational fashion

$$R1 \times R2$$

- If the arity or R1 is ‘n’ and arity of R2 is m then arity of R1XR2 will be ‘n+m’
- If R1 has n_{r1} tuples and R2 has n_{r2} tuples then R1XR2 will have $n_{r1} * n_{r2}$ tuples

Emp_Id	Name
103	Rony
104	Shruti

Employee

Dept No.	Dname
203	Computer
204	Electronics

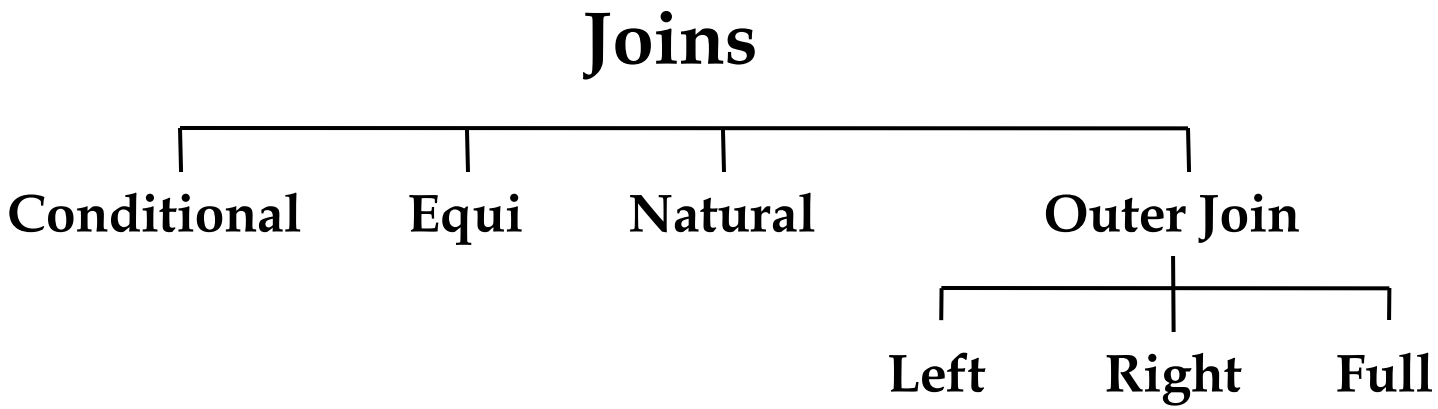
Department

Employee X Department

Emp_Id	Name	Dept No.	Dname
103	Rony	203	Computer
103	Rony	204	Electronics
104	Shruti	203	Computer
104	Shruti	204	Electronics

Join Operations

- Join operation can combine **cartesion** product and **selection condition** in a single operation.
- Join contains less number of tuples in comparison to cartesion product.



Conditional Join (Theta)

- It accept two relation and a condition and creates a new relation having tuples who are satisfying that condition

$$R1 \bowtie_c R2$$

- This is equivalent to

$$\sigma_c (R1 \times R2)$$

Equi Join

- Equi join case is a special case of conditional join where only condition of equality is used on the specific fields

Employee $\bowtie_{employee.deptno=department.deptno}$ Department

Natural Join

- Natural join is a special case of equi- join.
- Condition is checked on all the fields having the same name in two relations being joined

$$R1 \bowtie R2$$

Natural Join

Emp_Id	Name	P_Id
103	Rony	5
104	Shruti	6
105	Amit	Null

P_Id	Pname
4	P1
5	P2
6	P3

Employee $\bowtie$ Project

Emp_Id	Name	P_Id	Pname
103	Rony	5	P2
104	Shruti	6	P3

Division Operator

- Division operator is useful in the queries that include the phrases like “for all” or “every”

$$R1 \div R2$$

- Consider two instances A and B
- A has exactly 2 fields X and Y
- B has only one field that is Y with the same domain as in A
- Result of A/B will contain all the X values such that for every Y value in B, there is a tuple <x, y> in A.

EmpId	DeptNo
1976	1
1976	4
1976	3
1283	4
1283	3
2211	3

DeptNo
1
3
4

Result of A/B?

Outer Join

- Outer join deals with the missing values in the instance
- There are three types of Outer Join
  - Left Outer Join
  - Right Outer Join
  - Full Outer Join

Left Outer Join

Emp_Id	Name	P_Id
103	Rony	5
104	Shruti	6
105	Amit	Null

P_Id	Name
4	P1
5	P2
6	P3

Employee ⟕ Project

Emp_Id	Name	P_Id	Name
103	Rony	5	P2
104	Shruti	6	P3
105	Amit	Null	Null

Right Outer Join

Emp_Id	Name	P_Id
103	Rony	5
104	Shruti	6
105	Amit	Null

P_Id	Name
4	P1
5	P2
6	P3

Employee ⟖ Project

Emp_Id	Name	P_Id	Name
103	Rony	5	P2
104	Shruti	6	P3
Null	Null	4	P1

Full Outer Join

Emp_Id	Name	P_Id
103	Rony	5
104	Shruti	6
105	Amit	Null

P_Id	Name
4	P1
5	P2
6	P3

Employee ⟗ Project

Emp_Id	Name	P_Id	Name
103	Rony	5	P2
104	Shruti	6	P3
105	Amit	Null	Null
Null	Null	4	P1