RELATIONAL ALGEBRA

Contents

- What is Relational Algebra?
- Operations in Relational Algebra
- Extended Relational Algebra Operations
- Assignments

What is Relational Algebra?

- It is a *procedural query language*.
- Consists of set of operations.
- Takes one *(unary)* or two *(binary)* relations as input & produce a new relation as output.
- Introduced by *E. F. Codd* in *1970* as a basis for a database query languages.

What is a Query Language?

- □ Database language.
- □ Use for retrieving informations from database.
- □ Two types:-

  - Procedural

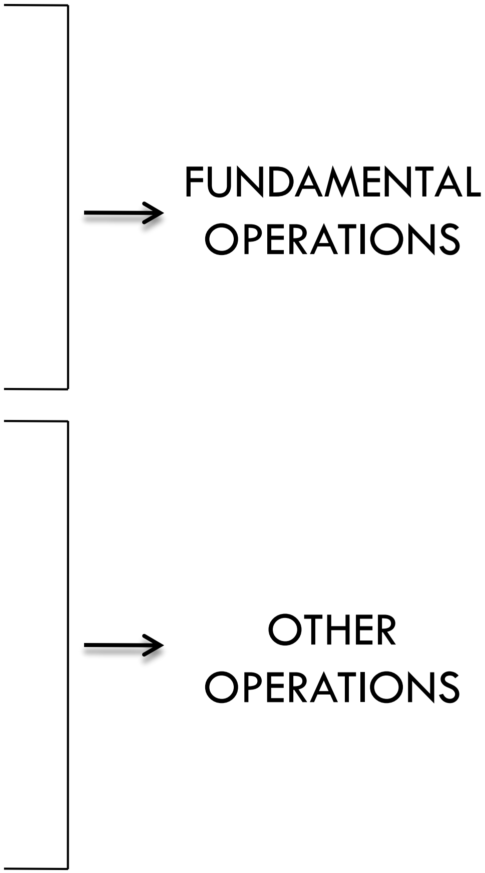
    Specifies what data are needed & how to get those data.

  - Non-procedural

    Specifies what data are needed except how to get those data.

Operations

- □ Select
- □ Project
- □ Union
- □ Set Difference
- □ Cartesian Product
- □ Rename
- □ Set Intersection
- □ Natural Join
- □ Semi Join
- □ Theta Join
- □ Assignment
- □ Outer Join
- □ Division

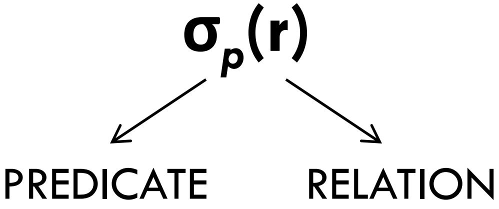


FUNDAMENTAL OPERATIONS

OTHER OPERATIONS

Select Operation (σ)

- Unary operation.
- Selects tuples *(rows)* that satisfy the given predicate *(condition)* from a relation *(table)*.



$\sigma_p(r)$

PREDICATE

RELATION

- There can be more than one predicate connected by connectors (*and* [**^**], *or* [**v**], *not* [¬]).
- Comparisons are performed using relational operators (**=, ≠, ≥, <, >, ≤**).

Examples

person(*name, age, weight*)

1. Find the details of all persons having age greater than equals to 34.

Ans: $\sigma_{age \geq 34}$(person)

2. Find the details of all persons having age greater than 34 & weight equals to 54.

Ans: $\sigma_{age > 34 \text{ ^ } weight = 54}$(person)

Illustrations

person

name	age	weight
Harry	34	80
Sally	28	64
George	29	70
Helena	54	54
Peter	34	80

$б_{age>=34}(person)$

name	age	weight
Harry	34	80
Helena	54	54
Peter	34	80

$б_{age>34 \wedge weight=54}(person)$

name	age	weight
Helena	54	54

Project Operation (Π)

- Unary operation.
- Selects specified attributes (*columns*) from a relation.

$$\Pi_{\underbrace{A1,\ A2,\ An}_{\text{ATTRIBUTES}}}\underset{\text{RELATION}}{(r)}$$

Example

employee(*name, age, salary*)

1. Find the name of all employees.

Ans: Π_{name}(employee)

employee

name	age	salary
Harry	34	80,000
Sally	28	90,000
George	29	70,000
Helena	54	54,280
Peter	34	40,000

Π_{name}(employee)

name
Harry
Sally
George
Helena
Peter

Combination of Select & Project Operations - Example

employee(*name, age, salary*)

1. Find the name of employees earning more than 80,000

Ans: $\Pi_{name}(\sigma_{salary>80000}(employee))$

employee

name	age	salary
Harry	34	80,000
Sally	28	90,000
George	29	70,000
Helena	54	54,280
Peter	34	40,000

б$_{salary>80,000}$(employee)

name	age	salary
Sally	28	90,000

$\prod_{name}$(б$_{salary>80,000}$(employee))

name
Sally

Union Operation (U)

- □ Binary operation.
- □ Returns the union of two compatible relations (say **r** & **s**).

r U s

where,

- **r** & **s** must have the same number of attributes.

- Attribute domains must be compatible.

- Duplicate tuples are automatically eliminated.

Example

course(*name*, *semester*, *teacher*)

1. Find the name of all courses taught in the 1st semester, the 2nd semester, or both.

Ans: $\Pi_{name}(\sigma_{semester=\text{“1st"}}(course))$

$\cup$

$\Pi_{name}(\sigma_{semester=\text{“2nd"}}(course))$

Illustration

course

name	semester	teacher
C1	1st	T1
C2	2nd	T2
C3	3rd	T3
C4	4th	T4
C1	2nd	T1

∏ $_{\text{name}}$(б$_{\text{semester=“1st"}}$(course))

name
C1

∏ $_{\text{name}}$(б$_{\text{semester=“2nd"}}$(course))

name
C2
C1

∏ $_{\text{name}}$(б$_{\text{semester=“1st"}}$(course)) U ∏$_{\text{name}}$(б$_{\text{semester=“2nd"}}$(course))

name
C1
C2

Set Difference (-)

- Binary operation.
- Returns the tuples which are present in first relation (**r**) but are not in the second relation (**s**).

r – s

- Like **union** the conditions are same for a valid **r – s** operation.

Example

course(*name*, *semester*, *teacher*)

1. Find the name of all courses taught in the 1st semester but not in 2nd semester.

Ans: $\Pi_{name}(\sigma_{semester=\text{“1st"}}(course))$

$-$

$\Pi_{name}(\sigma_{semester=\text{“2nd"}}(course))$

Illustration

course

name	semester	teacher
C1	1st	T1
C2	2nd	T2
C3	1st	T3
C4	4th	T4
C1	2nd	T1

$\prod_{\text{name}}(\text{б}_{\text{semester=“1 st"}}(\text{course}))$

name
C1
C3

$\prod_{\text{name}}(\text{б}_{\text{semester=“2nd"}}(\text{course}))$

name
C2
C1

$\prod_{\text{name}}(\text{б}_{\text{semester=“1 st"}}(\text{course})) - \prod_{\text{name}}(\text{б}_{\text{semester=“2nd"}}(\text{course}))$

name
C3

Cartesian Product (x)

- Binary operation.
- Combines information from any two relations (**r** & **s**).

r x s

- It defines a relation by concatenating every tuple of relation **r** with every tuple of relation **s.**

Illustration

person

name	age	weight
Harry	34	80
Sally	28	64
George	29	70

city

city
San Jose
Austin

person X city

name	age	weight	city
Harry	34	80	San Jose
Harry	34	80	Austin
Sally	28	64	San Jose
Sally	28	64	Austin
George	29	70	San Jose
George	29	70	Austin

Rename Operation (ρ)

□ Unary operation

$$\boldsymbol{\rho}_{\mathbf{a}\,/\,\mathbf{b}}\,(\mathbf{r})$$

where,

- **r** is a relation

- **a** and **b** are attributes

□ The result is identical to **r** except that the **b** field in the relation is renamed to an **a** field.

Example

employee

name	age	salary
Harry	34	80,000
Sally	28	90,000
George	29	70,000
Helena	54	54,280
Peter	34	40,000

$\rho_{ename\ /\ name}$(employee)

ename	age	salary
Harry	34	80,000
Sally	28	90,000
George	29	70,000
Helena	54	54,280
Peter	34	40,000

Set Intersection (∩)

- Binary operation.

$$\mathbf{r \cap s}$$

- Defines a relation consisting of the set of all tuples that are in both **r** and **s**.
- Like **union** the conditions are same for a valid **r ∩ s** operation.
- Expressed using basic operations:

$$\mathbf{r \cap s = r - (r - s)}$$

Example

course(*name, semester, teacher*)

1. Find the name of all courses taught in both the 1st & 2nd semesters.

Ans: $\Pi_{name}(\sigma_{semester=“1st"}(course))$

$\cap$

$\Pi_{name}(\sigma_{semester=“2nd"}(course))$

Illustration

course

name	semester	teacher
C1	1st	T1
C2	2nd	T2
C3	1st	T3
C4	4th	T4
C1	2nd	T1

$\prod_{name}$(б$_{semester=“1st"}$(course))

name
C1
C3

$\prod_{name}$(б$_{semester=“2nd"}$(course))

name
C2
C1

$\prod_{name}$(б$_{semester=“1st"}$(course)) $\cap$ $\prod_{name}$(б$_{semester=“2nd"}$(course))

name
C1

Natural Join (⋈)

- Binary operation.

  $$\mathbf{r} \bowtie \mathbf{s}$$

  where,

  - **r** & **s** are relations.

- The result is the set of all combinations of tuples in **r** and **s** that are equal on their common attribute names.

Illustration

t1

a	b
a1	b1
a2	b2
a3	b3

t2

b	c
b1	c1
b2	c2
b4	c4

$t1 \bowtie t2$

a	b	c
a1	b1	c1
a2	b2	c2

Semi Join (⋉)

- Similar to the *Natural Join.*

  $$\mathbf{r} \ltimes \mathbf{s}$$

  where,

  - **r** & **s** are relations.
- Outputs only those set of tuples in **r** for which there is a tuple in **s** that is equal on their common attribute names.

Illustration

t1

a	b
a1	b1
a2	b2
a3	b3

t2

b	c
b1	c1
b2	c2
b4	c4

t1 ⋉ t2

a	b
a1	b1
a2	b2

Theta Join (θ)

- Variant of the *Natural Join*.
- Combine a *selection* & a *Cartesian product* into a single operation.

$$\mathbf{r \bowtie_{\theta} s = \sigma_{\theta}(r\ x\ s)}$$

where,

- **r** & **s** are relations.

- ***θ*** is a predicate.

Assignment Operation

- Writing a relational algebra expression by assigning parts of it to temporary relation variables.
- Works like assignment in a programming language.
- Denoted by ←.

Illustration

course

name	semester	teacher
C1	1st	T1
C2	2nd	T2
C3	1st	T3
C4	4th	T4
C1	2nd	T1

$r1 \leftarrow \prod_{name}(\text{б}_{semester=\text{“1st"}}(course))$

name
C1
C3

$r2 \leftarrow \prod_{name}(\text{б}_{semester=\text{“2nd"}}(course))$

name
C2
C1

$r1 \cap r2$

name
C1

Outer Join

- Deals with missing information.
- Works in a similar fashion like *natural join*, but preserves those tuples that would be lost in the join by creating tuples in the result containing null values.
- Three types:

  - Left Outer Join

  - Right Outer Join

  - Full Outer Join

Left Outer Join (=⋈)

- Takes all the tuples in the left relation that did not match with any tuple in the right relation.
- Pads the tuples with null values for all other attributes from the right relation.
- Adds them to the result of the *natural join.*

r =⋈ s

where,

- **r** & **s** are relations.

Illustration

t1

a	b
a1	b1
a2	b2
a3	b3

t2

b	c
b1	c1
b2	c2
b4	c4

t1 =⋈ t2

a	b	c
a1	b1	c1
a2	b2	c2
a3	b3	*null*

Right Outer Join (⋈=)

□ Takes all the tuples in the right relation that did not match with any tuple in the left relation.

□ Pads the tuples with null values for all other attributes from the left relation.

□ Adds them to the result of the *natural join.*

r ⋈= s

where,

- **r** & **s** are relations.

Illustration

t1

a	b
a1	b1
a2	b2
a3	b3

t2

b	c
b1	c1
b2	c2
b4	c4

t1 ⋈= t2

a	b	c
a1	b1	c1
a2	b2	c2
null	b4	c4

Full Outer Join (=⋈=)

- Performs both left & right outer join operations.
- Pads the tuples with null values from the right relation that did not match with any from the left relation & vice versa.

Left Outer Join + Right Outer Join = Full Outer Join

r =⋈= s

where,

- **r** & **s** are relations.

Illustration

t1

a	**b**
a1	b1
a2	b2
a3	b3

t2

b	**c**
b1	c1
b2	c2
b4	c4

t1 =⋈= t2

a	**b**	**c**
a1	b1	c1
a2	b2	c2
a3	b3	*null*
null	b4	c4

Division Operation

- □ Binary operation.

  r / s

  where,

  - **r** is the *dividend* relation & **s** is the *divisor* relation.

  - All the attributes in **s** also appear in **r.**

  - **s** is not empty.

- □ The result consists of the restrictions of tuples in **r** to the attribute names unique to **r**, i.e., in the header of **r** but not in the header of **s**, for which it holds that all their combinations with tuples in **s** are present in **r.**

Illustration

completed

student	task
Fred	Database1
Fred	Database2
Fred	Compiler1
Eugene	Database1
Sara	Database1
Sara	Database2
Eugene	Compiler1

DBProject

task
Database1
Database2

completed / DBProject

student
Fred
Sara

If **DBProject** contains all the tasks of the Database project, then the result of the division above contains exactly the students who have completed both of the tasks in the Database project.

Extended Relational Algebra Operations

- Generalized Projection
- Aggregation

Generalized Projection

- Extension of *projection* operation.
- Allows *arithmetic operations* & *string functions* to be used in the projection list.

Example

employee(*name, age, salary*)

1. Find the name & monthly salary of all employees.

Ans: $\Pi_{name,\ salary/12}$(employee)

Aggregation ($\mathcal{G}$)

- □ Permits the use of aggregate functions (*sum*, *average*, *min*, *max*, *count*)

  - take a collection of values & return a single value as a result.

  $${}_{G1,\ldots,Gn}\mathcal{G}_{F1(A1),\ldots,Fm(Am)}(E)$$

  where,

  - E => Any relational algebra expression

  - G1,…,Gn => List of attributes on which to group

  - F1,…,Fm => Aggregate functions

  - A1,…,Am => Attribute names

Examples

instructor (*<u>id</u>*, *name*, *dept_name*, *salary*)

teaches (*id*, *course_id*, *sec_id*, *semester*, *year*)

1. Find out the sum of salaries of all instructors

   $\mathcal{G}_{\text{sum(salary)}}(\text{instructor})$

2. Find the total no. of instructors who teach a course in the Spring 2010 semester

   $\mathcal{G}_{\text{count-distinct(id)}}(\sigma_{\text{semester=“Spring” ^ year=2010}}(\text{teaches}))$

3. Find out the average salary of all instructors

   $\mathcal{G}_{\text{average(salary)}}(\text{instructor})$

4. Find out the average salary in each department

   ${}_{\text{dept\_name}}\mathcal{G}_{\text{average(salary)}}(\text{instructor})$

Assignments

1. **instructor (<u>id</u>, name, dept_name, salary)**

   teaches (id, course_id, sec_id, semester, year)

   Find the names of all instructors in the Physics department together with the course-id of all courses they taught.

Assignments (contd…)

2. **employee (emp_name, street, city)**

works (emp_name, comp_name, sal)

company (comp_name, city)

manages (emp_name, manager_name)

a) Find the names of all employees who work for 1st Bank Co-orporation.

b) Find the names & cities of residence of all employees who work for 1st Bank Co-orporation.

c) Find the names, street addresses & cities of all employees who work for 1st Bank Co-orporation & earn more than 10,000.

d) Find all employees in the DB who live in the same cities as the companies for which they work.

e) Find all employees in the DB who live in the same city & the same street as do their managers.

f) Find all employees in the DB who don’t work for 1st Bank Co-orporation.

Assignments (contd…)

3. **sailors (<u>sid</u>, sname, rating, age)**

reserved (sid, bid, day)

boats (<u>bid</u>, bname, color)

a) Find the names of the sailors who have reserved a red or green boat.

b) Find the names of the sailors with age over 20 who have not reserved a red boat.

c) Find the names of the sailor who have reserved all boats.

d) Find the color of the boat reserved by Smith.

Assignments (contd…)

4. **staff (<u>sno</u>, sname, pos, dob, sal, bno)**

branch (<u>bno</u>, street, city, pincode)

propertyforrent (<u>pno</u>, street, city, pincode, type, rooms, rent, ownerno, sno, bno)

a) List the names of all managers who earn more than 25000.

b) List the staff names who manage properties for rent in Kolkata.

c) List the names of staff who currently don't manage any properties.