Database Management Systems (BCSC-1003)

Topic: Relational Algebra in DBMS

Dr. Nikhil Govil

Assistant Professor, Dept. of CEA, GLA University, Mathura.

Relational Algebra

- Relational algebra is a widely used procedural query language.
- It takes instances of relations as input and yields instances of relations as output.
- It uses various operations to perform this action.
- Relational algebra mainly provides theoretical foundation for relational databases and SQL.

Basic SQL Relational Algebra Operations

Relational Algebra may be divided in various groups:

1. Relational Algebra Operations From Set Theory
    a. UNION (∪)
    b. INTERSECTION (∩)
    c. DIFFERENCE (–)

Basic SQL Relational Algebra Operations

2. Unary Relational Operations
   a. SELECT (σ)
   b. PROJECT (π)
   c. RENAME (ρ)

Basic SQL Relational Algebra Operations

3. Binary Relational Operations

I. JOIN (⋈)

Types of JOIN:

Various forms of join operation are:

A. Cross Join or Cartesian Product

B. Inner Joins:

(i). Theta join (ii). EQUI join (iii). Natural join

C. Outer join:

(i). Left Outer Join (ii). Right Outer Join

(iii). Full Outer Join

II. DIVISION (÷)

Relational Algebra Operations From Set Theory

UNION (∪)

UNION is symbolized by ∪ symbol. It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples. So, set A UNION set B would be expressed as:

$R = A \cup B$

Relational Algebra Operations From Set Theory

UNION (∪)

Exp.

Relation A

CNAME	CSTATUS
RAJAT	GOOD
RAHUL	EXCELLENT

Relation B

CNAME	CSTATUS
KARAN	POOR
RAJAT	GOOD

$R = A \cup B$

CNAME	CSTATUS
RAJAT	GOOD
RAHUL	EXCELLENT
KARAN	POOR

Relational Algebra Operations From Set Theory

INTERSECTION ($\cap$)

INTERSECTION is symbolized by $\cap$ symbol. 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.

$R = A \cap B$

Relational Algebra Operations From Set Theory

INTERSECTION ($\cap$)

Exp.

Relation A

CNAME	CSTATUS
RAJAT	GOOD
RAHUL	EXCELLENT

Relation B

CNAME	CSTATUS
KARAN	POOR
RAJAT	GOOD

$R = A \cap B$

CNAME	CSTATUS
RAJAT	GOOD

Relational Algebra Operations From Set Theory

DIFFERENCE (–)

DIFFERENCE is symbolized by – symbol. The result of A–B, is a relation which includes all tuples that are in A but not in B.

R = A – B

Relational Algebra Operations From Set Theory

DIFFERENCE (–)

Exp.

Relation A

CNAME	CSTATUS
RAJAT	GOOD
RAHUL	EXCELLENT

Relation B

CNAME	CSTATUS
KARAN	POOR
RAJAT	GOOD

R = A – B

CNAME	CSTATUS
RAHUL	EXCELLENT

Unary Relational Operations

a. SELECT (σ)

b. PROJECT (π)

c. RENAME (ρ)

SELECT Operator (σ)

- SELECT operation is used for selecting a subset of the tuples according to a given selection condition.

- Select Operator is denoted by sigma (σ).

- It is used as an expression to choose tuples which meet the selection condition.

- Select operator selects tuples that satisfy a given predicate.

SELECT Operator (σ)

Syntax:

$\sigma_{<Condition>}$ (Relation/Table name)

Properties:

1. It is like a WHERE clause in SQL.
2. Selection operator is commutative i.e.,

$$\sigma_{<condition\_1>} ((\sigma_{<condition\_2>} (R)) = \sigma_{<condition\_2>} ((\sigma_{<condition\_1>} (R))$$

SELECT Operator (σ)

Examples: STUDENT

ROLL_NO	NAME	AGE	COURSE
1	ABHISHEK	17	BTech
2	AMIT	16	BCA
3	AJEET	17	BTech
4	AKHIL	18	BTech
5	PRASHANT	17	BCA

Query 1: Find out the students of course ‘BTech’.

Syntax: $\sigma_{\text{course = 'BTech'}}$ (student)

Query2: Find the student(s) whose age is greater than 17.

Syntax: $\sigma_{\text{age} > 17}$ (student)

PROJECT or PROJECTION Operator (π)

- PROJECT Operator is denoted by pi (π).
- Project operation selects certain attributes discarding other attributes.
- It is also known as vertical partitioning since it partitions the relation or table vertically.
- Duplicate rows are automatically eliminated, as relation is a set.

PROJECT Operator (π)

Syntax:

$\pi_{\text{<Attribute list>}}$ (Relation/Table name)

Properties:

1. The degree of output relation (number of columns present) is equal to the number of attributes mentioned in the attribute list.
2. Projection operator does not obey commutative property i.e.

$$\pi_{\text{<list2>}} (\pi_{\text{<list1>}} (R)) \neq \pi_{\text{<list1>}} (\pi_{\text{<list2>}} (R))$$

PROJECT Operator (π)

Examples: STUDENT

ROLL_NO	NAME	AGE	COURSE
1	ABHISHEK	17	BTech
2	AMIT	16	BCA
3	AJEET	17	BTech
4	AKHIL	18	BTech
5	PRASHANT	17	BCA

Query 1: Find out the name and course from STUDENT relation.

Syntax: $\pi_{\text{name, course}}$ (student)

Query 2: Find the age from STUDENT relation.

Syntax: π_{age} (student)

AGE
17
16
18

RENAME Operator (ρ)

- The results of relational algebra are also relations but without any name.

- The RENAME operation allows us to rename the relation.

- It is denoted by rho (ρ).

RENAME Operator (ρ)

Syntax:

ρ (Relation2, Relation1)

Examples:

Query 1: Rename STUDENT relation to STUDENT1 relation.

Syntax: ρ (STUDENT1, STUDENT)

Query 2: Create a relation STUDENT_NAMES with RNO and NAME from STUDENT.

Syntax: ρ (STUDENT_NAMES, $\pi_{(RNO,\ NAME)}$ (STUDENT))

Binary Relational Algebra Operations

3. Binary Relational Operations

I. JOIN ($\bowtie$)

Types of JOIN:

Various forms of join operation are:

A. Cross Join or Cartesian Product

B. Inner Joins:

(i). Theta join (ii). EQUI join (iii). Natural join

C. Outer join:

(i). Left Outer Join (ii). Right Outer Join

(iii). Full Outer Join

II. DIVISION ($\div$)

Join (⋈)

- Join is a binary operation which allows us to combine join product and selection in one single statement.

- The goal of creating a join condition is that it helps us to combine the data from two or more DBMS tables.

- Various forms of join operation are: Cross Join, Inner Join & Outer Join.

Cross Join or Cartesian Product (×)

Cross Join or Cartesian Product is symbolized by × symbol. It is an operation used to merge columns from two relations. It is also called Cross Product.

$R = A \times B$

Cross Join or Cartesian Product (×)

Exp.

Relation A

NAME	AGE
KAJAL	32
ANIL	40

Relation B

JOB	LOCATION
DEVELOPER	CHENNAI
ANALYST	MUMBAI

$R = A \times B$

NAME	AGE	JOB	LOCATION
KAJAL	32	DEVELOPER	CHENNAI
KAJAL	32	ANALYST	MUMBAI
ANIL	40	DEVELOPER	CHENNAI
ANIL	40	ANALYST	MUMBAI

Inner Join

- INNER JOIN is used to return rows from both tables which satisfy the given condition.
- It is the most widely used join operation and can be considered as a default join-type.
- Inner Join further divided into three subtypes:

  (i). Theta join (ii). EQUI join (iii). Natural join

Inner Join: Theta Join

- THETA JOIN allows us to merge two tables based on the condition represented by theta.
- Theta joins work for all comparison operators.
- It is denoted by symbol θ.
- The general case of JOIN operation is called a Theta join.

Syntax: $\mathbf{A \bowtie_{\theta} B}$

R1 and R2 are relations having attributes (A_1, A_2, .., A_n) and (B_1, B_2, ..., B_n) such that the attributes don’t have anything in common, that is $R1 \cap R2 = \Phi$.

Inner Join: Theta Join

Example:

STUDENT

SID	NAME	STD
1001	AJAY	11
1002	JATIN	12

SUBJECT

CLASS	SUBJECT
11	MATH
11	HINDI
12	COMPUTER
12	SCIENCE

$$\text{STUDENT} \bowtie_{\text{Student.Std = Subject.Class}} \text{SUBJECT}$$

Output:

SID	NAME	STD	CLASS	SUBJECT
1001	AJAY	11	11	MATH
1001	AJAY	11	11	HINDI
1002	JATIN	12	12	COMPUTER
1002	JATIN	12	12	SCIENCE

Inner Join: Equi Join

- EQUI JOIN is done when a Theta join uses only the equivalence condition.

- When Theta join uses only equality comparison operator, it is said to be equi join.

- The previous example corresponds to equi join.

Inner Join: Natural Join

- NATURAL JOIN does not utilize any of the comparison operators.
- In this type of join, the attributes should have the same name and domain.
- In Natural Join, there should be at least one common attribute between two relations.
- It performs selection forming equality on those attributes which appear in both relations and eliminates the duplicate attributes.

Inner Join: Natural Join

Example:

R1

NUM	SQUARE
2	4
3	9

R2

NUM	CUBE
2	8
3	27

R1 $\bowtie$ R2

Output:

NUM	SQUARE	CUBE
2	4	8
3	9	27

Outer Join

- An OUTER JOIN doesn't require each record in the two join tables to have a matching record.
- In this type of join, the table retains each record even if no other matching record exists.
- Outer Join further divided into three subtypes:

  (i). Left Outer Join

  (ii). Right Outer Join

  (iii). Full Outer Join

Outer Join: Left Outer Join (⟕)

- LEFT OUTER JOIN returns all the rows from the table on the left even if no matching rows have been found in the table on the right.
- When no matching record found in the table on the right, NULL is returned.
- It is denoted by symbol (⟕).

Syntax: **A ⟕ B**

Outer Join: Left Outer Join (⟕)

Example:

R1

NUM	SQUARE
2	4
3	9
4	16

R2

NUM	CUBE
2	8
3	27
5	125

R1 ⟕ R2

Output:

NUM	SQUARE	CUBE
2	4	8
3	9	27
4	16	NULL

Outer Join: Right Outer Join (⋊)

- RIGHT OUTER JOIN returns all the columns from the table on the right even if no matching rows have been found in the table on the left.
- Where no matches have been found in the table on the left, NULL is returned. RIGHT OUTER JOIN is the opposite of LEFT OUTER JOIN.
- It is denoted by symbol (⋊).

Syntax: **A ⋊ B**

Outer Join: Right Outer Join (⟖)

Example:

R1

NUM	SQUARE
2	4
3	9
4	16

R2

NUM	CUBE
2	8
3	27
5	125

R1 ⟖ R2

Output:

NUM	CUBE	SQUARE
2	8	4
3	27	9
5	125	NULL

Outer Join: Full Outer Join (⟗)

- In a FULL OUTER JOIN , all tuples from both relations are included in the result, irrespective of the matching condition.
- It is denoted by symbol (⟗).

Syntax: **A ⟗ B**

Outer Join: Full Outer Join (⟗)

Example:

R1

NUM	SQUARE
2	4
3	9
4	16

R2

NUM	CUBE
2	8
3	27
5	125

R1 ⟗ R2

Output:

NUM	SQUARE	CUBE
2	4	8
3	9	27
4	16	NULL
5	NULL	125

<u>DIVISION (÷)</u>

Division operator A÷B can be applied if and only if:

- Attributes of B is proper subset of Attributes of A.
- The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B).
- The relation returned by division operator will return those tuples from relation A which are associated to every B’s tuple.

Syntax: **A ÷ B**

DIVISION (÷)

Example:

Relation A has 1 table as:

Sno	Pno
S1	P1
S1	P2
S1	P3
S1	P4
S1	P5
S1	P6
S2	P1
S2	P2
S3	P2
S4	P2
S4	P4
S4	P5

Relation B has 3 tables as:

Pno
P1

Pno
P2
P4

Pno
P1
P2
P3
P4
P5
P6

Then, A DIVIDE BY B gives the following resultant tables for all the three cases as follows:

Sno
S1
S2

Sno
S1
S4

Sno
S1

Exercise

Example 1: Consider the following schema:

SUPPLIER (Sid, S_name, S_addr)

PARTS (Pid, P_name, color)

CATALOG (sid, pid, cost)

Now, answer the following queries in Relational Algebra:

(a) Find the name of all the suppliers who supply Yellow parts.

(b) Find the Sid of suppliers who supply every parts.

(c) Find the Sid of suppliers who supply every red or green part.

Exercise

Solution:

(a) Find the name of all the suppliers who supply Yellow parts.

$$\pi_{S\_name}\ (SUPPLIER) \bowtie CATALOG \bowtie (\sigma_{color = \text{'Yellow'}}\ (PARTS))$$

(b) Find the Sid of suppliers who supply every parts.

$$\pi_{sid,\ pid}\ (CATALOG) \div \pi_{pid}\ (PARTS)$$

(c) Find the Sid of suppliers who supply every red or green part.

$$(\pi_{sid,\ pid}\ (CATALOG)) \div (\pi_{pid}\ (\sigma_{color = \text{'red'} \vee color = \text{'green'}}\ (PARTS)))$$

Exercise

Example 2: Consider a database that has the relation schema CR (StudentName, CourseName). An instance of the schema CR is as given below.

StudentName	CourseName
SA	CA
SA	CB
SA	CC
SB	CB
SB	CC
SC	CA
SC	CB
SC	CC
SD	CA
SD	CB
SD	CC
SD	CD
SE	CD
SE	CA
SE	CB
SF	CA
SF	CB
SF	CC

The following queries are made on the database.

$T1 \leftarrow \pi_{CourseName}(\sigma_{StudentName='SA'}(CR))$

$T2 \leftarrow CR \div T1$

The number of rows in T2 are ________.

(GATE 2017)

Exercise

Solution:

T1 will give:

CourseName
CA
CB
CC

T2$\leftarrow$CR$\div$T1 = All the tuples in CR which are matched with every tuple in T1:

StudentName
SA
SC
SD
SF

Reference Books

1. Elmasri and Navathe (2010), “Fudamentals of Database Systems”, 5th Edition, Addision Wesley.
2. Date C J,” An Introduction to Database Systems”, 8th Edition, Addision Wesley.
3. Korth, Silbertz and Sudarshan (1998), “Database Concepts”, 4th Edition, TMH.
4. M. Tamer Oezsu, Patrick Valduriez (2011). “Principles of Distributed Database Systems”, 2nd Edition, Prentice Hall.

Thank You !!!