Database Management System

Relational Algebra

Dr. Balasundaram A

VIT-Chennai

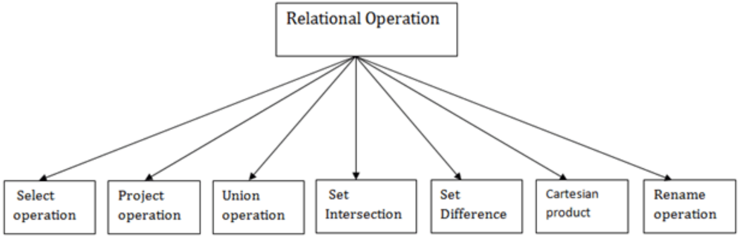
Relational Query Languages

- Relational database systems comes with a query language, which assists users to query the database instances.
- There are two kinds of query languages:
  1. **Relational Algebra**
  2. **Relational Calculus**

Relational Algebra

- Relational algebra is a widely used procedural query language.
- It collects instances of relations as input and gives occurrences of relations as output.
- It uses various operation to perform this action.
- Relational algebra operations are performed recursively on a relation.
- The output of these operations is a new relation, which might be formed from one or more input relations.
- Basic Relational Algebra Operations :
  1. **Unary Relational Operations** : SELECT(σ), PROJECT(Π), RENAME(ρ)
  2. **Binary Relational Operations** : JOIN($\bowtie$), DIVISION($/$)
  3. **Set Theory Operations** : UNION($\cup$), INTERSECTION($\cap$), DIFFERENCE($-$) AND CARTESIAN PRODUCT($\times$)

Relational Operations



SELECT(σ) Operation

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

Sigma(σ)Symbol denotes it. It is used as an expression to choose tuples which meet the selection condition.

Select operation selects tuples that satisfy a given predicate.

For Example : $\sigma_p(\mathbf{r})$:

- σ is a predicate
- p is a prepositional logic
- r stands for relation(Table)

SELECT(σ) Operation

Example-1:

$$\sigma_{\text{topic="Database"}}(\text{Tutorial})$$

Output : Selects tuples from Tutorial where topic = 'Database'.

Example-2:

$$\sigma_{\text{topic="Database" and author = "Balasundaram"}}(\text{CSE2004})$$

Output : Selects tuples from CSE2004 where the topic is '**Database**' and 'author' is **"Balasundaram"**.

Example-3:

$$\sigma_{\text{sales} > 50000}(\text{Customer})$$

Output :Selects tuples from Customers where sales is greater than 50000

Projection(Π) Operation

It eliminates all attributes of the input relation but those mentioned in the projection list. It defines a relation that contains a vertical subset of Relation.

It helps to extract the values of specified attributes to eliminates duplicate values.

(Π) symbol used to choose attributes from a relation.

This operation helps you to keep specific columns from a relation and discards the other columns.

Projection(Π) Operation

Example : Consider the following table.

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Here, the projection of **CustomerName** and **Status** from **Customer** Table will give :
$\Pi_{CustomerName,Status}$(Customer)

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

Examples of SELECT(σ) & PROJECTION (Π)

Notation: $\sigma_p(r)$

Where σ stands for selection predicate and r stands for relation, p is predicate logic formula which may use connectors like and, or, and not.

These terms may use relational operators like: =, ≠, ≥, <, >, ≤.

For example:

$\sigma_{\text{subject="database"}}$**(Books) :**

Selects tuples from Books where subject = 'database'.

$\sigma_{\text{subject="database" and price="450"}}$**(Books) :**

Selects tuples from Books where subject is 'database' and 'price' is 450.

$\sigma_{\text{subject="database" and price < "450" or year > "2010"}}$**(Books) :**

Selects tuples from Books where subject is 'database' and 'price' is 450 or those books published after 2010.

Examples of SELECT(σ) & PROJECTION (Π)

Notation: $\Pi_{A1, A2,..An}(r)$

Where $A_1, A_2, \ldots A_n$ are attribute names of relation r.

Duplicate rows are automatically eliminated, as relation is a set.

For example:

$\Pi_{subject,author}$**(Books)**

selects and projects columns named as subject and author from the relation Books.

Union (∪) Operation

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: A ∪ B.

The Following condition must hold:

- A and B must be the same number of attributes.
- Attribute domains need to be compatible.
- Duplicate tuples should be automatically removed.

Example : Union (∪) Operation

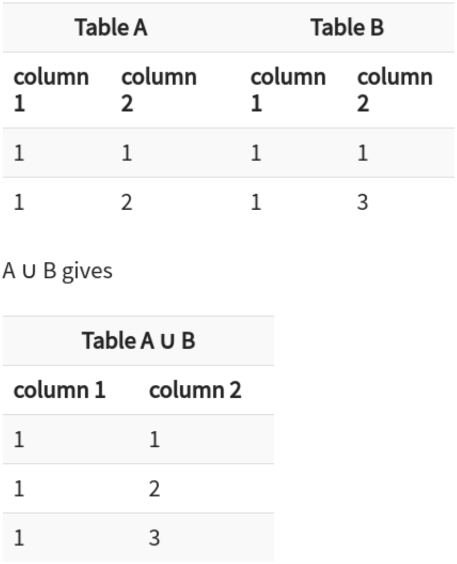


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

A ∪ B gives

Table A ∪ B	
column 1	column 2
1	1
1	2
1	3

Set Difference ($-$)

$-$ Symbol denotes it.

The result of A - B, is a relation which includes all tuples that are in A but not in B.

The Following Conditions must satisfy :

- The attribute name of A has to match with the attribute name 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.

Example : Set Difference ($-$)

For Example : Consider the Below Relations :

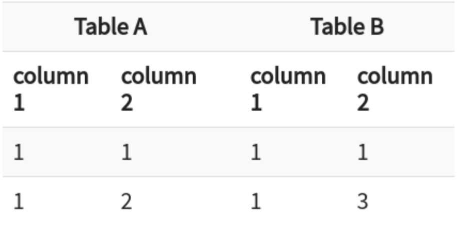


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

The Operations on Relations A $-$ B gives :

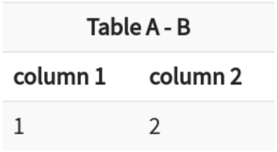


Table A - B	
column 1	column 2
1	2

Intersection ∩ Operation

An intersection is defined by the 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

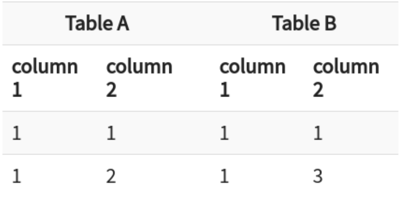


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

. The intersection of A ∩ B is :

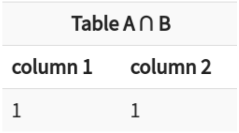


Table A ∩ B	
column 1	column 2
1	1

.

Cartesian Product (X) Operation

This type of operation is helpful to merge columns from two relations.

Generally, a Cartesian product is never a meaningful operation when it performs alone. However, it becomes meaningful when it is followed by other operations.

Example : Let us consider the below Relations A and B.
The $\sigma_{column2='1'}$(A X B) is :

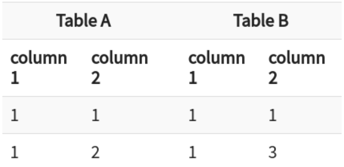
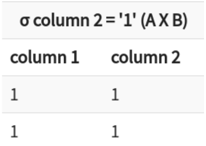


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3



σ column 2 = '1' (A X B)	
column 1	column 2
1	1
1	1

.

Join ($\bowtie$) Operation

Join operation is essentially a Cartesian product followed by a selection criterion.

Join operation denoted by $\bowtie$.

JOIN operation also allows joining variously related tuples from different relations.

Types of Join

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

Inner Join Operations

In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded.

1. **Theta Join** : The general case of JOIN operation is called a Theta join. It is denoted by symbol θ
   Example : $A \bowtie_{\theta} B$, $A \bowtie_{A.Column2 > B.Column2} B$.

2. **EQUI join** : When a theta join uses only equivalence condition, it becomes a equi join
   Example : $A \bowtie_{A.Column2 = B.Column2} B$.

3. **NATURAL JOIN** ($\bowtie$) : Natural join can only be performed if there is a common attribute (column) between the relations. The name and type of the attribute must be same.
   Example : $A \bowtie B$.

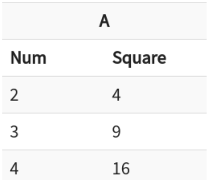
Outer Join ⟗

In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

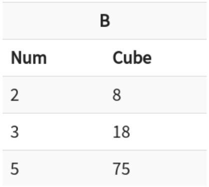
1. **Left Outer Join** (⟕) : In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.
2. **Right Outer Join**(⟖) : In the right outer join, operation allows keeping all tuple in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.
3. **Full Outer Join**(⟗) : In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

Join Operation Examples

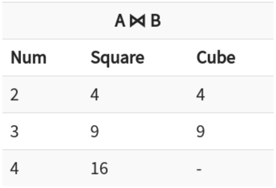
1. **Left Join** (A ⟕ B)



A	
Num	Square
2	4
3	9
4	16

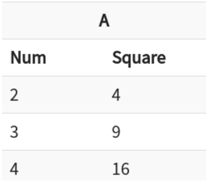


B	
Num	Cube
2	8
3	18
5	75

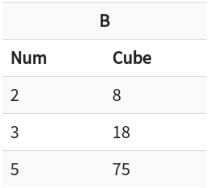


A ⋈ B		
Num	Square	Cube
2	4	4
3	9	9
4	16	-

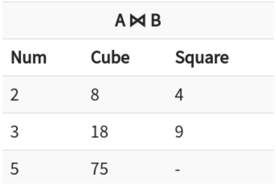
2. **Right Join** (A ⟖ B)



A	
Num	Square
2	4
3	9
4	16



B	
Num	Cube
2	8
3	18
5	75



A ⋈ B		
Num	Cube	Square
2	8	4
3	18	9
5	75	-

Join Operation Examples

3. **Full Join** (A ⟗ B)

A	
Num	**Square**
2	4
3	9
4	16

B	
Num	**Cube**
2	8
3	18
5	75

A ⋈ B		
Num	**Cube**	**Square**
2	4	8
3	9	18
4	16	-
5	-	75

Summary of Relational Algebra Operations

1. **Select(σ)** : It is used for selecting a subset of the tuples according to a given selection condition
2. **Projection(Π)** : The projection eliminates all attributes of the input relation but those mentioned in the projection list.
3. **Union($\cup$)** : It includes all tuples that are in tables A or in B.
4. **Intersection($\cap$)** : It is a relation which includes all tuples that are in A but not in B.
5. **Intersection($\cap$)** : It a relation consisting of a set of all tuple that are in both A and B.
6. **Cartesian Product(X)** : Iis helpful to merge columns from two relations.

Summary of Relational Algebra Operations

1. **Theta Join**(θ) : The general case of JOIN operation is called a Theta join
2. **EQUI Join** : When a theta join uses only equivalence condition, it becomes a equi join.
3. **Natural Join**($\bowtie$) : can only be performed if there is a common attribute (column) between the relations.
4. **Left Outer Join**(⟕) : operation allows keeping all tuple in the left relation.
5. **Right Outer Join**(⟖) : operation allows keeping all tuple in the right relation.
6. **Full Outer Join**(⟗) : all tuples from both relations are included in the result irrespective of the matching condition.

Thanks