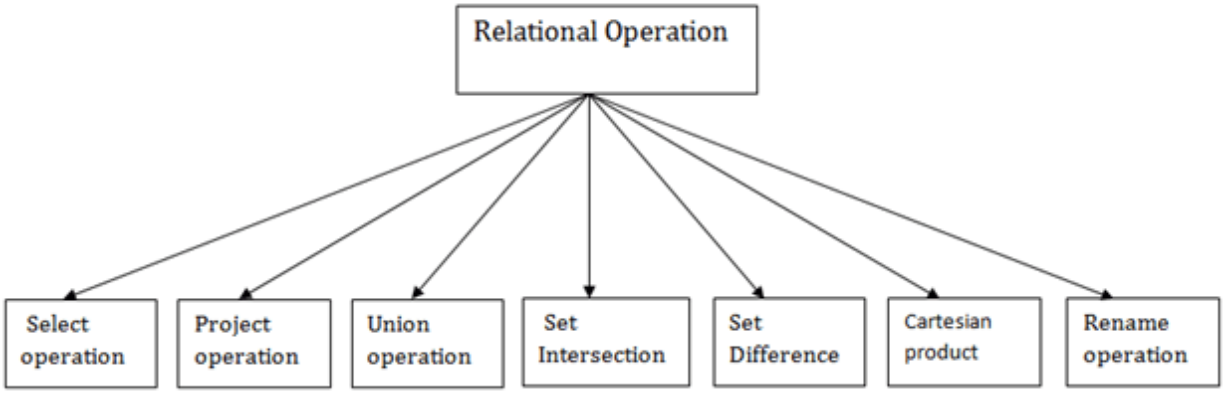
Relational Algebra

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.

Types of Relational operation



1. **Select Operation:**
   - o The select operation selects tuples that satisfy a given predicate.
   - o It is denoted by sigma (σ).
   - 1. 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 relational can use as relational operators like =, ≠, ≥, <, >, ≤.

For example: LOAN Relation

BRANCH_NAME	LOAN_NO	AMOUNT
Downtown	L-17	1000
Redwood	L-23	2000
Perryride	L-15	1500
Downtown	L-14	1500
Mianus	L-13	500
Roundhill	L-11	900
Perryride	L-16	1300

INPUT:

σ BRANCH_NAME="perryride" (LOAN)

Output:

BRANCH NAME	LOAN NO	AMOUNT
Perryride	L-15	1500
Perryride	L-16	1300

2. **Project Operation:**

This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.

It is denoted by Π.

Notation: Π A1, A2, An (r)
Where
A1, A2, A3 is used as an attribute name of relation r.

Example: CUSTOMER RELATION

NAME	STREET	CITY
Jones	Main	Harrison
Smith	North	Rye
Hays	Main	Harrison
Curry	North	Rye
Johnson	Alma	Brooklyn
Brooks	Senator	Brooklyn

Input:

Π NAME, CITY (CUSTOMER)

Output:

NAME	CITY
Johns	Harrison
Smith	Rye
Hays	Harrison
Curry	Rye
Johnson	Brooklyn
Brooks	Brooklyn

3. **Union Operation:**

Suppose there are two tuples R and S. The union operation contains all the tuples that are either in R or S or both in R & S.

It eliminates the duplicate tuples. It is denoted by ∪.

Notation: R ∪ S

A union operation must hold the following condition:

R and S must have the attribute of the same number.

Duplicate tuples are eliminated automatically.

Example:

DEPOSITOR RELATION

CUSTOMER NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

BORROW RELATION

CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-07

Input:

Π CUSTOMER_NAME (BORROW) ∪ Π CUSTOMER_NAME (DEPOSITOR

Output:

CUSTOMER NAME

Johnson

Smith

Hayes
Turner
Jones
Lindsay
Jackson
Curry
Williams
Mayes

4. **Set Intersection:**

   Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in both R & S.

   It is denoted by intersection ∩.

Notation: R ∩ S

Example: Using the above DEPOSITOR table and BORROW table

INPUT:

Π CUSTOMER_NAME (BORROW) ∩ Π CUSTOMER_NAME (DEPOSITOR)

OUTPUT:

CUSTOMER NAME

Smith
Jones

5. **Set Difference:**

Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in R but not in S.

It is denoted by intersection minus (-).

Notation: R - S

Example: Using the above DEPOSITOR table and BORROW table

Input:

Π CUSTOMER_NAME (BORROW) - Π CUSTOMER_NAME (DEPOSITOR)

Output:

CUSTOMER NAME

Jackson
Hayes
Williams
Curry

6. **Cartesian Product:**

The Cartesian product is used to combine each row in one table with each row in the other table. It is also known as a cross product.

It is denoted by X.

Notation: E X D

Example:

EMPLOYEE

EMP_ID	**EMP_NAME**	**EMP_DEPT**
1	Smith	A
2	Harry	C

3	John	B

DEPARTMENT

DEPT_NO	**DEPT_NAME**
A	Marketing
B	Sales
C	Legal

Input:

EMPLOYEE X DEPARTMENT

Output:

EMP_ID	**EMP_NAME**	**EMP_DEPT**	**DEPT_NO**	**DEPT_NAME**
1	Smith	A	A	Marketing
1	Smith	A	B	Sales
1	Smith	A	C	Legal
2	Harry	C	A	Marketing

7. **Rename Operation:**

The rename operation is used to rename the output relation. It is denoted by **rho** (ρ).

Example: We can use the rename operator to rename STUDENT relation to STUDENT1.

ρ(STUDENT1, STUDENT)

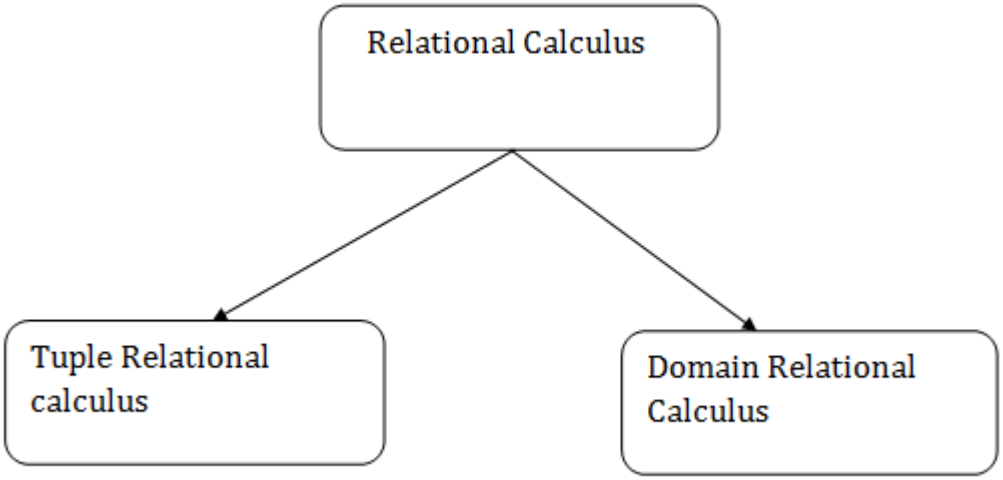
Note:Apart from these common operations Relational algebra can be used in Join operations.

Relational Calculus

Relational calculus is a non-procedural query language. In the non-procedural query language, the user is concerned with the details of how to obtain the end results.

- The relational calculus tells what to do but never explains how to do.

Types of Relational calculus:



1. Tuple Relational Calculus (TRC)

- The tuple relational calculus is specified to select the tuples in a relation. In TRC, filtering variable uses the tuples of a relation.

o The result of the relation can have one or more tuples.

Notation:

{T | P (T)} or {T | Condition (T)}

Where

T is the resulting tuples
P(T) is the condition used to fetch T.

For example:

{ T.name | Author(T) AND T.article = 'database' }

OUTPUT: This query selects the tuples from the AUTHOR relation. It returns a tuple with 'name' from Author who has written an article on 'database'.
TRC (tuple relation calculus) can be quantified. In TRC, we can use Existential (∃) and Universal Quantifiers (∀).

For example:

{ R| ∃T ∈ Authors(T.article='database' AND R.name=T.name)}

Output: This query will yield the same result as the previous one.

2. Domain Relational Calculus (DRC)

- The second form of relation is known as Domain relational calculus. In domain relational calculus, filtering variable uses the domain of attributes.

- Domain relational calculus uses the same operators as tuple calculus. It uses logical connectives ∧ (and), ∨ (or) and ¬ (not).

- It uses Existential (∃) and Universal Quantifiers (∀) to bind the variable

Notation:

{ a1, a2, a3, ..., an | P (a1, a2, a3, ... ,an)}

Where
a1, a2 are attributes **P** stands for formula built by inner attributes

For example:

{< article, page, subject > | ∈ javatpoint ∧ subject = 'database'}

Output: This query will yield the article, page, and subject from the relational javatpoint, where the subject is a database.