RELATIONAL ALGEBRA

- Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output.
- It uses operators to perform queries. An operator can be either **unary** or **binary**.
- They accept relations as their input and yield relations as their output.
- Relational algebra is performed recursively on a relation and intermediate results are also considered relations.

The fundamental operations of relational algebra are as follows −

- Select
- Project
- Union
- Set different
- Cartesian product
- Rename

Select Operation (σ)

- It selects tuples that satisfy the given predicate from a relation.
- **Notation** − $\sigma_p(r)$
- Where **σ** stands for selection predicate and **r** stands for relation. *p* is prepositional logic formula which may use connectors like **and, or,** and **not**. These terms may use relational operators like − =, ≠, ≥, < , >, ≤.

- **For example** −
- $\sigma_{subject = "database"}(Books)$

Output − Selects tuples from books where subject is 'database'.

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

Output − Selects tuples from books where subject is 'database' and 'price' is 450.

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

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

Project Operation (∏)

- It projects column(s) that satisfy a given predicate.
- Notation − $\prod_{A1, A2, An} (r)$
- Where A_1, A_2 , A_n are attribute names of relation **r**.
- Duplicate rows are automatically eliminated, as relation is a set.

For example −

- $\prod_{subject, author} (Books)$
- Selects and projects columns named as subject and author from the relation Books.

Union Operation (U)

- It performs binary union between two given relations and is defined as −

r ∪ s = { t | t ∈ r or t ∈ s} **Notation** − r U s

- Where **r** and **s** are either database relations or relation result set (temporary relation).
- For a union operation to be valid, the following conditions must hold −
- **r**, and **s** must have the same number of attributes.
- Attribute domains must be compatible.
- Duplicate tuples are automatically eliminated.
- $\prod_{author}$ (Books) ∪ $\prod_{author}$ (Articles)

Output − Projects the names of the authors who have either written a book or an article or both.

Set Difference (–)

- The result of set difference operation is tuples, which are present in one relation but are not in the second relation.
- **Notation** – **r** – **s**
- Finds all the tuples that are present in **r** but not in **s**.
- Π_{author} (Books) – Π_{author} (Articles)

Output – Provides the name of authors who have written books but not articles.

Cartesian Product (X)

- Combines information of two different relations into one.
- **Notation** – r X s
- Where **r** and **s** are relations and their output will be defined as –
- r X s = { q t | q ∈ r and t ∈ s}
- $\sigma_{\text{author = 'KORTH'}}$(Books X Articles)

Output – Yields a relation, which shows all the books and articles written by KORTH.

Rename Operation (ρ)

- The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'rename' operation is denoted with small Greek letter **rho** *ρ*.
- **Notation** − $\rho_x(E)$
- Where the result of expression **E** is saved with name of **x**.

Additional operations are −

- Set intersection
- Assignment
- Natural join

Relational Calculus

- In contrast to Relational Algebra, Relational Calculus is a non-procedural query language, that is, it tells what to do but never explains how to do it.
- Relational calculus exists in two forms –

a) Tuple Relational Calculas

b) Domain Relational Calculas

Tuple Relational Calculas

- Filtering variable ranges over tuples

Notation − {T | Condition}

- Returns all tuples T that satisfies a condition.
- **For example** −
- { T.name | Author(T) AND T.article = 'database' }

Output − Returns tuples with 'name' from Author who has written article on 'database'.

- TRC can be quantified. We can use Existential (∃) and Universal Quantifiers (∀).
- **For example** −
- { R| ∃T ∈ Authors(T.article='database' AND R.name=T.name)}
  Output − The above query will yield the same result as the previous one.

Domain Relational Calculus (DRC)

- In DRC, the filtering variable uses the domain of attributes instead of entire tuple values (as done in TRC, mentioned above).
- **Notation** –
- $\{ a_1, a_2, a_3, ..., a_n \mid P (a_1, a_2, a_3, ... ,a_n)\}$
- Where a1, a2 are attributes and **P** stands for formulae built by inner attributes.
- **For example** –
- {< article, page, subject > | ∈ KORTH ∧ subject = 'database'}
- **Output** – Yields Article, Page, and Subject from the relation KORTH, where subject is database.