**Relational Algebra**

Relational algebra is the basic set of operations for the relational model, its procedural query language ,these operations enable a user specify basic retrieval requests or queries such as insert update delete. The result of an operation is a new relation which may have been formed form one or more input relations. This property makes the algebra “closed” (all objects in relational algebra are relations).A sequence of relational algebra operations forms a relational algebra expression. The result of relational algebra expression is also relation that represents the result of a database query(retrieval request) .

**History of Origins of Algebra**

Muhamed ibn Musa AL-Khwarizmi worte a book tilted al-jabar about artimetic of variables ,Book was translated into latin its title (al-jabar) gave Algebra its name.

**Relational algebra consist of several groups of operations**

**Unary Relational Operations**

SELECT (symbol: σ (sigma))

PROJECT (symbol: Π (pi))

RENAME (symbol: ρ (rho))

**Relational Algebra Operations from set Theory**

UNIION (symbol: U )

INTERSECTION(symbol: (∩ )

DIFFERENCE(MINUS , symbol (-))

CARTESIAN PRODUCT(X)

**Binary Relational Operations**

Natural Join (⋈)  
Left, Right, Full outer join (⟕, ⟖, ⟗)  
 Intersection (∩)  
Division (÷)

## 

## Select Operator(σ)

## Select operator is denoted by sigma and it is used to find the tuples(rows) in a relation (or table) which satisfy the given condition Similar to where clause in SQL.

Syntax:

σ Condition/Predicate(Relation/Table name)

## Example

Table: CUSTOMER

---------------

Customer\_Id Customer\_Name Customer\_City

----------- ------------- -------------

C10100 Steve Agra

C10111 Raghu Agra

C10115 Chaitanya Noida

C10117 Ajeet Delhi

C10118 Carl Delhi

## Query

σ Customer\_City="Agra" (CUSTOMER)

## Output

Customer\_Id Customer\_Name Customer\_City

----------- ------------- -------------

C10100 Steve Agra

C10111 Raghu Agra

## Project Opreator(∏)

## It Is used to select desired columns (or attributes) from a table (a relation). Is similar with Select statements in sql.project creates a vertical partitioning .list of specified columns(attributes) is kept in each tuple, it removes any duplicate tuples this is because project opreation must be a set of tuples matitmatical sets do not allow duplicate elements.

## Syntax:

∏ column\_name1, column\_name2, ...., column\_nameN(table\_name)

## Query

∏ Customer\_Name, Customer\_City (CUSTOMER)

## Output

Customer\_Name Customer\_City

------------- -------------

Steve Agra

Raghu Agra

Chaitanya Noida

Ajeet Delhi

Carl Delhi

## Union Opreator (∪)

## It is used to select all the rows tuples from two tables(relations).The row (tuples ) that are present in both the tables will only appear once in the union set short you can say that there are no duplicates present after the union operation.

Table 1: COURSE

Course\_Id Student\_Name Student\_Id

--------- ------------ ----------

C101 Aditya S901

C104 Aditya S901

C106 Steve S911

C109 Paul S921

C115 Lucy S931

## 

Table 2: STUDENT

Student\_Id Student\_Name Student\_Age

------------ ---------- -----------

S901 Aditya 19

S911 Steve 18

S921 Paul 19

S931 Lucy 17

S941 Carl 16

S951 Rick 18

**Query:**

∏ Student\_Name (COURSE) ∪ ∏ Student\_Name (STUDENT)

**Output:**

Student\_Name

------------

Aditya

Carl

Paul

Lucy

Rick

Steve

## Intersection Opreator(∩)

## It is used to select common rows(tuples )from two tables(relations).Both tables should have same columns. Only those rows that are present in both tables will appear in the result set.

## Syntax:

table\_name1 ∩ table\_name2

**Query:**

∏ Student\_Name (COURSE) ∩ ∏ Student\_Name (STUDENT)

**Output:**

Student\_Name

------------

Aditya

Steve

Paul

Lucy

**Set Difference (-)**

Lets say we have two R1 and R2 we want to select all those tuples(rows) that are present in Relation R1 but not present Relation R2 this can be done using set difference R1-R2

Syntax:

table\_name1 - table\_name2

**Query**

∏ Student\_Name (STUDENT) - ∏ Student\_Name (COURSE)

**Output**

Student\_Name

------------

Carl

Rick

**Cartesian Product (X)**

Lets say we have two relations R1 and R2 then the cartesian product of these two relations (R1x r2) would combine each tuple of first relation R1 with the each tuple of second relation R2.

Syntax:

R1 X R2

Table 1: R

Col\_A Col\_B

----- ------

AA 100

BB 200

CC 300

Table 2: S

Col\_X Col\_Y

----- -----

XX 99

YY 11

ZZ 101

**Query:**  
Lets find the cartesian product of table R and S.

R X S

**Output:**

Col\_A Col\_B Col\_X Col\_Y

----- ------ ------ ------

AA 100 XX 99

AA 100 YY 11

AA 100 ZZ 101

BB 200 XX 99

BB 200 YY 11

BB 200 ZZ 101

CC 300 XX 99

CC 300 YY 11

CC 300 ZZ 101

## Rename(ρ)

## Can be used to rename a relation or an attribute of a relation

## ****Syntax:**** ρ(new\_relation\_name, old\_relation\_name

Table: CUSTOMER

Customer\_Id Customer\_Name Customer\_City

----------- ------------- -------------

C10100 Steve Agra

C10111 Raghu Agra

C10115 Chaitanya Noida

C10117 Ajeet Delhi

C10118 Carl Delhi

**Query:**

ρ(CUST\_NAMES, ∏(Customer\_Name)(CUSTOMER))

**Output:**

CUST\_NAMES

----------

Steve

Raghu

Chaitanya

Ajeet

Carl

## Relational Calculus

## Relational calculus is non-procedural query language that tells the system what data to be retrieved but doesn’t tell how to retrieve it. It creates a new relation which is specified in terms of variables that range over rows of the stored database relations (in tuple calculus) or over columns of the store relations(domain calculus). In a calculus expression there is no order of operations to specify how to retrieve the query results a calculus expression specifies only that information the result should contain.This is the main difference between relational algebra and calculus.

## Tuple Relational Calculus(TRC)

## It is used for selecting those tuples that satisfy those tuples that satisfy the given condition. It is based on specifying a number of tuple variables. Each tuple variable usually range over a particular database relation, meaning that the variable may take as its value any induvial tuple from that relation.

First\_Name Last\_Name Age

---------- --------- ----

Ajeet Singh 30

Chaitanya Singh 31

Rajeev Bhatia 27

Carl Pratap 28

## Syntax:

## {t | COND(t)} where t is a tuple variable and COND(t) is a conditional expression involving t.

## The result of such a query is the set of all tuples t that satisfy COND(t)

## Query

{ t.Last\_Name | Student(t) AND t.age > 30 }

## Output

Last\_Name

---------

Singh

## Domain Relational Calculus(DRC)

## Here the records are filtered based on the domains.

## Query

{< First\_Name, Age > | ∈ Student ∧ Age > 27}

## The symbols used for logical operators are: ∧ for AND, ∨ for OR and ┓ for NOT.

## Data Localization(Residency)

## Data localization is the act of storing data on any device that is physically present within the borders of a specific country where the data was generated.

## 