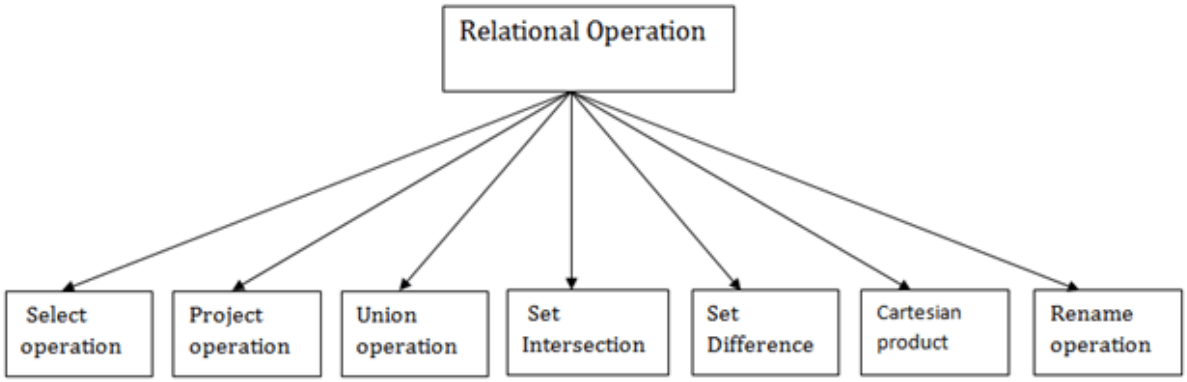
⇧ SCROLL TO TOP

# 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:

- The select operation selects tuples that satisfy a given predicate.
- It is denoted by sigma (σ).

> Notation: σ p(r)

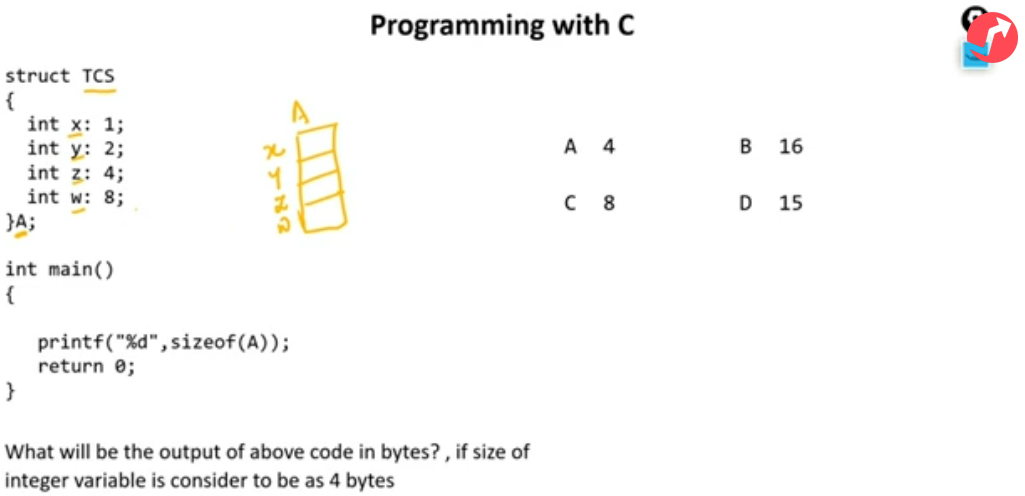
**Where:**

**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 ∏.

```
Notation: ∏ 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:**

> ∏ NAME, CITY (CUSTOMER)

**Output:**

| NAME | CITY |
|---|---|
| Jones | 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-17 |

**Input:**

> ∏ CUSTOMER_NAME (BORROW) ∪ ∏ 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:**

> ∏ CUSTOMER_NAME (BORROW) ∩ ∏ 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:**

> ∏ CUSTOMER_NAME (BORROW) - ∏ CUSTOMER_NAME (DEPOSITOR)

**Output:**

| CUSTOMER_NAME |
| --- |
| Jackson |
| Hayes |
| Willians |
| 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 |
| 2 | Harry | C | B | Sales |
| 2 | Harry | C | C | Legal |
| 3 | John | B | A | Marketing |
| 3 | John | B | B | Sales |
| 3 | John | B | C | Legal |

## 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.

← Prev Next →

## Feedback

- Send your Feedback to feedback@javatpoint.com

## Help Others, Please Share

## Learn Latest Tutorials

Splunk

SPSS

Swagger

Transact-SQL

Tumblr

ReactJS

Regex

Reinforcement Learning

R Programming

RxJS

React Native

Python Design Patterns

Python Pillow tutorial

Python Turtle tutorial

Keras

Python Pillow

Python Turtle

## Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

## Trending Technologies

Artificial Intelligence Tutorial

Artificial Intelligence

AWS Tutorial

AWS

Selenium tutorial

Selenium

Cloud Computing tutorial

Hadoop tutorial

Hadoop

ReactJS Tutorial

ReactJS

Cloud Computing

Data Science Tutorial

Data Science

Angular 7 Tutorial

Angular 7

Blockchain Tutorial

Blockchain

Git Tutorial

Git

Machine Learning Tutorial

Machine Learning

DevOps Tutorial

DevOps

## B.Tech / MCA

DBMS tutorial

DBMS

Data Structures tutorial

Data Structures

DAA tutorial

DAA