

Relational Algebra

Tutorial

November 19, 2025

Contents

Introduction

Selection ()

Projection ()

Rename ()

Set Operations

Cartesian Product (\times)

Join ()

Summary Table

End

Introduction

- ▶ Relational Algebra is a formal query language for manipulating relations.
- ▶ It forms the mathematical foundation behind SQL and query optimization.
- ▶ Each operation takes a relation as input and produces a new relation.

Purpose

Relational Algebra describes *what* data to retrieve, not *how* to retrieve it.

Selection ()

- ▶ Selects tuples that satisfy a given condition.
- ▶ Only rows are filtered; columns remain unchanged.
- ▶ General form: $\sigma_{\text{condition}}(R)$
- ▶ Selection is **commutative**.

Example

All cities in Japan:

$$\sigma_{\text{CountryCode}='JPN'}(\text{CITY})$$

Projection ()

- ▶ Selects specific attributes (columns).
- ▶ Removes duplicates automatically. (duplicate elimination)
- ▶ General form: $\pi_{\text{attribute list}}(R)$
- ▶ Not commutative.

Example

Names of American cities with population > 120000 :

$$\pi_{\text{Name}} (\sigma_{\text{Population} > 120000 \text{ AND } \text{CountryCode} = 'USA'}(\text{CITY}))$$

Rename ()

- ▶ Renames relations or attributes.
- ▶ Used when the same relation appears multiple times in a query.
- ▶ Essential for multi-step expressions and clarity.

General Forms

Relation rename:

$$R_{\text{new}} \leftarrow \rho_{\text{new}}(R)$$

Example (SQL Equivalent)

```
SELECT Branch AS Stream, Grade AS CGPA FROM Student_Details;
```

Set Operations

- ▶ Relations must be **union-compatible**.
- ▶ **Union:** $R \cup S$ (tuples in either R or S)
- ▶ **Intersection:** $R \cap S$ (tuples common to both)
- ▶ **Difference:** $R - S$ (tuples in R but not S)

Cartesian Product (\times)

- ▶ Combines every tuple of R with every tuple of S .
- ▶ Degree of result = attributes of R + attributes of S .
- ▶ Often used before a selection or join.

Example (SQL Cross Join)

```
SELECT Student.Name, Student.Age,  
StudentCourse.Course_ID  
FROM Student CROSS JOIN StudentCourse;
```

Relational Algebra Example

If $R = \{(1, A), (2, B)\}$ and $S = \{(X), (Y)\}$, then:

$$R \times S = \{(1, A, X), (1, A, Y), (2, B, X), (2, B, Y)\}$$

Join ()

- ▶ Equivalent to: Cartesian Product + Selection.
- ▶ Combines related tuples from two relations.
- ▶ General form: $R \bowtie_{\text{condition}} S$

Example

Cities located in Africa:

$$\pi_{\text{CITY.Name}} (\text{CITY} \bowtie_{\text{CITY.CountryCode} = \text{COUNTRY.Code}}$$
$$\sigma_{\text{Continent}='Africa'}(\text{COUNTRY})$$

Relational Algebra Summary

Operation	Purpose	Notation
Selection	Filter rows	$\sigma_{\text{cond}}(R)$
Projection	Select columns	$\pi_{\text{attrs}}(R)$
Rename	Rename relation/attrs	$\rho_{\text{name}}(R)$
Union	Tuples in R or S	$R \cup S$
Intersection	Tuples common to R,S	$R \cap S$
Difference	Tuples in R not in S	$R - S$
Cartesian Product	All row combinations	$R \times S$
Join	Combine related rows	$R \bowtie_{\text{cond}} S$

Operations in Relational Algebra

Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R .	$\sigma_{\langle \text{selection condition} \rangle}(R)$
PROJECT	Produces a new relation with only some of the attributes of R , and removes duplicate tuples.	$\pi_{\langle \text{attribute list} \rangle}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$, OR $R_1 \bowtie_{(\langle \text{join attributes 1} \rangle, \langle \text{join attributes 2} \rangle)} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\langle \text{join condition} \rangle} R_2$, OR $R_1 *_{(\langle \text{join attributes 1} \rangle, \langle \text{join attributes 2} \rangle)} R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2 ; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2 ; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2 ; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2 .	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Thank You!