Relational Algebra

|  |  |
| --- | --- |
| Set Operations | Union, intersection, set difference... |
| Relational Database Operations | Unary operations:   * Operate on a single relation * Select, project…   Binary operations:   * Operate on two tables. * Join... |

### Selection

The **SELECT** operation is used to identify the subset of tuples from a relation that satisfy a selection condition.

Formal notation: σ(select condition) (Relation)

A **cascade** / **sequence** of select operations can be combined into a single operation:

σ(condition 1) (σ(condition 2) (Relation\_2) AND σ(condition 3) (Relation\_3))

|  |
| --- |
| SELECT \* FROM Employees  WHERE dept\_no = 4  AND salary > 25000; |

### Projection

The **project** operation selects certain attributes from the table, while discarding others (vertical partition).

Formal notation: 𝝅first\_name, last\_name, salary (Employees)

Projection only returns distinct values.

|  |
| --- |
| SELECT first\_name, last\_name, salary  FROM Employees; |

SQL does not automatically project distinct values. The DISTINCT keyword is needed.

Combining select and project:

𝝅first\_name, last\_name, salary (σ(salary > 25000) (Employees))

### Intermediate Relations

Emps ← σ(salary > 25000) (Employees)

Result ← 𝝅first\_name, last\_name, salary (Emps)

### Union, Intersection & Difference

|  |  |  |
| --- | --- | --- |
| Set Operators | SQL |  |
| ∪  ∩  R - S | UNION  INTERSECT  EXCEPT | Union  Intersection  Difference |

### Joins

Relation\_1 ⋈(join condition) Relation\_2

|  |
| --- |
| SELECT Employees.name, job, Departments.name  FROM Employees, Departments  WHERE Employee.dept\_no = Department.dept\_no; |