

RDBMS

II. Relational Algebra

Relational algebra

- Operands: relations (tables)
- Closure: the result of any operation is another relation
- Complete: all combinations of operators allowed
- Unary operators (single operand):
sélection (σ), projection (π)
- Binary operators:
Cartesian product (\times), join (\bowtie), union (\cup), intersection (\cap), set difference ($-$), division ($/$)

Outline

- For each of these 8 operators:
 - ◆ the operation
 - ◆ syntax (notation)
 - ◆ semantics (expected result)
 - ◆ schema
 - ◆ some annotation
 - ◆ an example

Selection

σ

- Goal: only select some tuples (lines) of a relation

| Country | name | capital | population | surface |
|---------|---------|---------|------------|---------|
| | Austria | Vienna | 8 | 83 |
| | UK | London | 56 | 244 |
| | Switz. | Berne | 7 | 41 |

We wish to select only countries with a small surface :

small-country = σ [surface < 100] Country

| small-Country | name | capital | population | surface |
|----------------------|---------|---------|------------|---------|
| | Austria | Vienna | 8 | 83 |
| | UK | London | 56 | 244 |
| | Switz. | Berne | 7 | 41 |

Projection π

- Goal: only keep some attributes (columns) of a relation

| Country | name | capital | population | surface |
|---------|---------|---------|------------|---------|
| | Austria | Vienna | 8 | 83 |
| | UK | London | 56 | 244 |
| | Switz. | Berne | 7 | 41 |

We only want to keep name and capital attributes :

capitals = π [name, capital] Country

| capitals | name | capital | population | surface |
|-----------------|---------|---------|------------|---------|
| | Austria | Vienna | 8 | 83 |
| | UK | London | 56 | 244 |
| | Switz. | Berne | 7 | 41 |

5

Side-effect of projection

- Elimination of repeated tuples
 - A projection that does not preserve the primary key of a relation may produce identical tuples in its result
 - The result will only contain one instance of the tuple
 - In SQL, this is not the default behavior, use DISTINCT keyword to force this behavior

$R (B, C, D)$ $\pi (B, C) R$



three tuples

two tuples

6

Selection-projection

- We want the capitals of small Country:
 - ◆ $\text{small-Country} = \sigma [\text{surface} < 100] \text{Country}$
 - ◆ $\text{capitals} = \pi [\text{name}, \text{capital}] \text{small-Country}$

$\text{capital-small-Country} =$

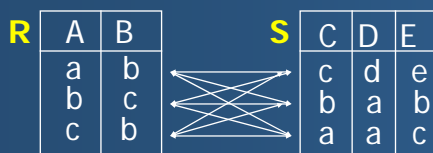
$\pi [\text{name}, \text{capital}] \sigma [\text{surface} < 100] \text{Country}$

| <u>name</u> | capital | population | surface |
|-------------|---------|------------|---------|
| Ireland | Dublin | 3 | 70 |
| Austria | Vienna | 8 | 83 |
| UK | London | 56 | 244 |
| Switz. | Berne | 7 | 41 |

(grey and beige parts eliminated) 7

Cartesian product \times

- Goal: construct all combinations of tuples of two relations (usually before a selection)
- syntax : $R \times S$
- example :



n tuples

m tuples

$R \times S$

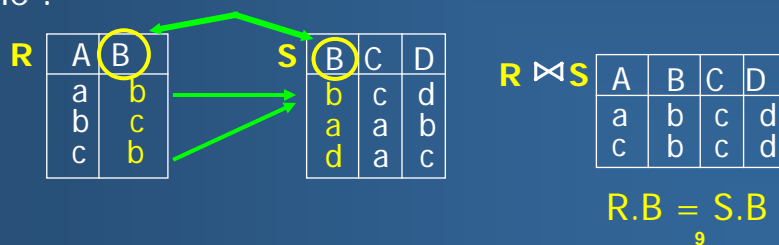
| A | B | C | D | E |
|---|---|---|---|---|
| a | b | c | d | e |
| a | b | b | a | b |
| a | b | c | c | e |
| b | c | c | b | a |
| b | c | c | b | a |
| b | c | c | b | a |
| c | b | b | b | a |
| c | b | a | a | c |

n x m tuples

Natural join



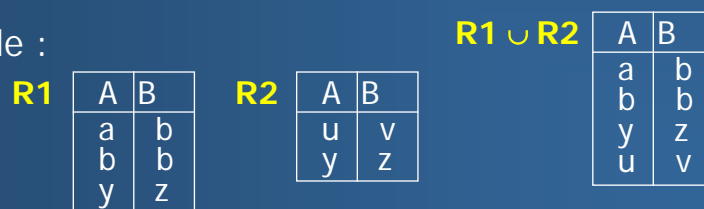
- Goal: create all **significant** combinations of the tuples of two relations
 - significant = bear the same value for the attribute on which the join is performed
- precondition: the two relations have an attribute of a the same type
- example :



Union



- binary operator
- syntax : $R \cup S$
- semantics : adds into a single relation the tuples (lines) of R and S
- schema : $\text{schema}(R \cup S) = \text{schema}(R) = \text{schema}(S)$
- precondition : $\text{schema}(R) = \text{schema}(S)$
- example :



Intersection \cap

- binary operator
- syntax : $R \cap S$
- semantics : selects tuples that belong to both R and S
- schema : schema ($R \cap S$) = schema (R) = schema (S)
- precondition : schema (R) = schema (S)
- example :

| R1 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>a</td><td>b</td></tr><tr><td>y</td><td>z</td></tr><tr><td>b</td><td>b</td></tr></tbody></table> | A | B | a | b | y | z | b | b | R2 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>u</td><td>v</td></tr><tr><td>y</td><td>z</td></tr></tbody></table> | A | B | u | v | y | z | R1 \cap R2 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>y</td><td>z</td></tr></tbody></table> | A | B | y | z |
|-----------|--|---|---|---|---|---|---|---|---|-----------|---|---|---|---|---|---|---|--------------------------------|--|---|---|---|---|
| A | B | | | | | | | | | | | | | | | | | | | | | | |
| a | b | | | | | | | | | | | | | | | | | | | | | | |
| y | z | | | | | | | | | | | | | | | | | | | | | | |
| b | b | | | | | | | | | | | | | | | | | | | | | | |
| A | B | | | | | | | | | | | | | | | | | | | | | | |
| u | v | | | | | | | | | | | | | | | | | | | | | | |
| y | z | | | | | | | | | | | | | | | | | | | | | | |
| A | B | | | | | | | | | | | | | | | | | | | | | | |
| y | z | | | | | | | | | | | | | | | | | | | | | | |

11

Set Difference -

- binary operator
- syntax : $R - S$
- semantics : selects tuples of R that are not in S
- schema : schema ($R - S$) = schema (R) = schema (S)
- precondition : schema (R) = schema (S)
- example :

| R1 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>a</td><td>b</td></tr><tr><td>y</td><td>z</td></tr><tr><td>b</td><td>b</td></tr></tbody></table> | A | B | a | b | y | z | b | b | R2 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>u</td><td>v</td></tr><tr><td>y</td><td>z</td></tr></tbody></table> | A | B | u | v | y | z | R1 - R2 | <table border="1"><thead><tr><th>A</th><th>B</th></tr></thead><tbody><tr><td>a</td><td>b</td></tr><tr><td>b</td><td>b</td></tr></tbody></table> | A | B | a | b | b | b |
|-----------|--|---|---|---|---|---|---|---|---|-----------|---|---|---|---|---|---|---|----------------|---|---|---|---|---|---|---|
| A | B | | | | | | | | | | | | | | | | | | | | | | | | |
| a | b | | | | | | | | | | | | | | | | | | | | | | | | |
| y | z | | | | | | | | | | | | | | | | | | | | | | | | |
| b | b | | | | | | | | | | | | | | | | | | | | | | | | |
| A | B | | | | | | | | | | | | | | | | | | | | | | | | |
| u | v | | | | | | | | | | | | | | | | | | | | | | | | |
| y | z | | | | | | | | | | | | | | | | | | | | | | | | |
| A | B | | | | | | | | | | | | | | | | | | | | | | | | |
| a | b | | | | | | | | | | | | | | | | | | | | | | | | |
| b | b | | | | | | | | | | | | | | | | | | | | | | | | |

12

Division /

- Goal: treat requests of the type «the ... such that ALL the...»
- let $R(A_1, \dots, A_n)$ and $V(A_1, \dots, A_m)$ with $n > m$ and A_1, \dots, A_m attributes of the same name in R and V
- $R/V = \{ \langle a_{m+1}, a_{m+2}, \dots, a_n \rangle / \forall \langle a_1, a_2, \dots, a_m \rangle \in V, \exists \langle a_1, a_2, \dots, a_m, a_{m+1}, a_{m+2}, \dots, a_n \rangle \in R \}$

- examples :

| A | B | C |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 2 | 0 |
| 1 | 2 | 1 |
| 1 | 3 | 0 |
| 2 | 1 | 1 |
| 2 | 3 | 3 |
| 3 | 1 | 1 |
| 3 | 2 | 0 |
| 3 | 2 | 1 |

| V | B | C | R/V | A |
|---|---|---|-----|---|
| 1 | 1 | 1 | 1 | 3 |
| 2 | 0 | | | |

| V' | B | C | R/V' | A |
|----|---|---|------|---|
| 1 | 1 | | 1 | 2 |
| | | | | 3 |

| V'' | B | C | R/V'' | A |
|-----|---|---|-------|---|
| 3 | 5 | | / | |

13

example

- R

| STUDENT | COURSE | PASSED |
|----------|--------|--------|
| Francois | RDB | yes |
| Francois | Prog | yes |
| Jacques | RDB | yes |
| Jacques | Math | yes |
| Pierre | Prog | yes |
| Pierre | RDB | no |

- V

| COURSE | PASSED |
|--------|--------|
| Prog | yes |
| RDB | yes |

- R/V

| STUDENT |
|----------|
| Francois |

14

Division

| certifications | PILOTE | APPAREIL |
|----------------|--------|----------|
| | Sierra | 737 |
| | Sierra | 757 |
| | Sierra | 747 |
| | Delta | 320 |
| | Delta | 757 |
| | Alpha | 737 |
| | Alpha | 757 |
| | Alpha | 747 |
| | Alpha | 320 |
| | India | 737 |

| avions | APPAREIL |
|--------|----------|
| | 320 |

certificationsA = certifications ÷ avions

| certificationsA | PILOTE |
|-----------------|--------|
| | Delta |
| | Alpha |

15

Division

| certifications | PILOTE | APPAREIL |
|----------------|--------|----------|
| | Sierra | 737 |
| | Sierra | 757 |
| | Sierra | 747 |
| | Delta | 320 |
| | Delta | 757 |
| | Alpha | 737 |
| | Alpha | 757 |
| | Alpha | 747 |
| | Alpha | 320 |
| | India | 737 |

| avions | APPAREIL |
|--------|----------|
| | 737 |
| | 757 |
| | 747 |

certificationsA = certifications ÷ avions

| certificationsA | PILOTE |
|-----------------|--------|
| | Sierra |
| | Alpha |

16

Examples of algebraic requests

- let us consider the following relations :

Journal (code-j, title, price, type, periodicity)

Depot (no-Depot, name-Depot, adress)

Delivery (no-Depot, code-j, date-deliv, quantity-delivered)

17

Satisfy these requests :

- What is the price of the journals ?
 π [price] Journal
- Give all known information on weekly journals.
 σ [periodicity = "weekly"] Journal
- Give the codes of the journals delivered in Paris.
 π [code-j] (σ [adress = "Paris"] Depot \bowtie Delivery)

18

Satisfy these requests :

- Give the number of the depots that receive several journals.

- π [no-Depot]
 (σ [code-j \neq code']
 (π [no-Depot, code'] α [code-j, code'] Delivery)
 ✕ π [no-Depot, code-j] Delivery)

- Note : α [code-j, code'] renames attribute code-j into code'