Chapter 3
Relational Database Languages: Relational Algebra

We first consider only *query* languages.

Relational Algebra: Queries are expressions over operators and relation names.

Relational Calculus: Queries are special formulas of first-order logic with free variables.

SQL: Combination from algebra and calculus and additional constructs. Widely used DML for relational databases.

QBE: Graphical query language.

Deductive Databases: Queries are logical rules.

Relational Database Languages: Comparison and Outlook

Remark:

- Relational Algebra and (safe) Relational Calculus have the same expressive power.
  For every expression of the algebra there is an equivalent expression in the calculus, and vice versa.
- A query language is called **relationally complete**, if it is (at least) as expressive as the relational algebra.
- These languages are compromises between efficiency and expressive power; they are not computationally complete (i.e., they cannot simulate a Turing Machine).
- They can be embedded into host languages (e.g. C++ or Java) or extended (PL/SQL), resulting in full computational completeness.
- Deductive Databases (Datalog) are more expressive than relational algebra and calculus.

3.1 Relational Algebra: Computations over Relations

Operations on Tuples – Overview Slide

Let $\mu \in \text{Tup}(\bar{X})$ where $\bar{X} = \{A_1, \ldots, A_k\}$.

(Formal definition of μ see Slide 61)

- For $\emptyset \subset \bar{Y} \subseteq \bar{X}$, the expression $\mu[\bar{Y}]$ denotes the **projection** of μ to $\bar{Y}$.

  Result: $\mu[\bar{Y}] \in \text{Tup}(\bar{Y})$ where $\mu[\bar{Y}](A) = \mu(A), A \in \bar{Y}$.

- A **selection condition** α (wrt. $\bar{X}$) is an expression of the form $A\,\theta\,B$ or $A\,\theta\,c$, or $c\,\theta\,A$ where $A, B \in \bar{X}$, dom(A) = dom(B), $c \in$ dom(A), and θ is a comparison operator on that domain like e.g. $\{=,\neq,\leq,<,\geq,>\}$.

  A tuple $\mu \in \text{Tup}(\bar{X})$ **satisfies** a selection condition α, if – according to α – $\mu(A)\,\theta\,\mu(B)$ or $\mu(A)\,\theta\,c$, or $c\,\theta\,\mu(A)$ holds.

  These (atomic) selection conditions can be combined to formulas by using $\wedge$, $\vee$, $\neg$, and $(,)$.

- For $\bar{Y} = \{B_1, \ldots, B_k\}$, the expression $\mu[A_1 \to B_1, \ldots, A_k \to B_k]$ denotes the **renaming** of μ.

  Result: $\mu[\ldots, A_i \to B_i, \ldots] \in \text{Tup}(\bar{Y})$ where $\mu[\ldots, A_i \to B_i, \ldots](B_i) = \mu(A_i)$ for $1 \leq i \leq k$.

Let $\mu \in \text{Tup}(\bar{X})$ where $\bar{X} = \{A_1, \ldots, A_k\}$.

Projection (Reduction to a subset of the attributes)

For $\emptyset \subset \bar{Y} \subseteq \bar{X}$, the expression $\mu[\bar{Y}]$ denotes the **projection** of μ to $\bar{Y}$.

Result: $\mu[\bar{Y}] \in \text{Tup}(\bar{Y})$ where $\mu[\bar{Y}](A) = \mu(A),\ A \in \bar{Y}$.

projection to a given set of attributes

Example 3.1

Consider the relation schema $R(\bar{X}) = Continent(name, area)$*:* $\bar{X} = [name,\ area]$

and the tuple $\mu = \boxed{\textit{name} \mapsto \text{“Asia”},\ \textit{area} \mapsto 4.50953e{+}07}$.

formally: $\mu(name) =$ *“Asia”*, $\mu(area) = 4.5E7$

projection attributes: Let $\bar{Y} = [name]$

Result: $\mu[name] = \boxed{\textit{name} \mapsto \text{“Asia”}}$ □

Again, $\mu \in \text{Tup}(\bar{X})$ where $\bar{X} = \{A_1, \ldots, A_k\}$.

Selection (only those tuples that satisfy some condition)

A **selection condition** α (wrt. $\bar{X}$) is an expression of the form $A\,\theta\,B$ or $A\,\theta\,c$, or $c\,\theta\,A$ where $A, B \in \bar{X}$, $\text{dom}(A) = \text{dom}(B)$, $c \in \text{dom}(A)$, and θ is a comparison operator on that domain like e.g. $\{=, \neq, \leq, <, \geq, >\}$.

A tuple $\mu \in \text{Tup}(\bar{X})$ **satisfies** a selection condition α, if – according to α – $\mu(A)\,\theta\,\mu(B)$ or $\mu(A)\,\theta\,c$, or $c\,\theta\,\mu(A)$ holds.

yes/no-selection of tuples (without changing the tuple)

Example 3.2

Consider again the relation schema $R(\bar{X}) = continent(name, area)$*:* $\bar{X} = [name,\ area]$*.*

Selection condition: $area > 20000000$*.*

Consider again the tuple $\mu =$ *name ↦ "Asia", area ↦ 4.50953e+07.*

formally: $\mu(name) =$ *"Asia",* $\mu(area) = 4.5E7$

check: $\mu(area) > 20000000$

Result: yes. □

These (atomic) selection conditions can be combined to formulas by using $\wedge$, $\vee$, $\neg$, and $(,)$.

Let $\mu \in \text{Tup}(\bar{X})$ where $\bar{X} = \{A_1, \ldots, A_k\}$.

Renaming (of attributes)

For $\bar{Y} = \{B_1, \ldots, B_k\}$, the expression $\mu[A_1 \to B_1, \ldots, A_k \to B_k]$ denotes the **renaming** of μ.

Result: $\mu[\ldots, A_i \to B_i, \ldots] \in \text{Tup}(\bar{Y})$ where $\mu[\ldots, A_i \to B_i, \ldots](B_i) = \mu(A_i)$ for $1 \leq i \leq k$.

renaming of attributes (without changing the tuple)

Example 3.3

Consider (for a tuple of the table $R(\bar{X}) = encompasses(country, continent, percent)$*):*

$\bar{X} = [country,\ continent,\ percent]$.

Consider the tuple $\mu =$ *country ↦ "R", continent ↦ "Asia", percent ↦ 80.*

formally: $\mu(country) =$ *"R",* $\mu(continent) =$ *"Asia",* $\mu(percent) = 80$

Renaming: $\bar{Y} = [code, name, percent]$

Result: a new tuple $\mu[country \to code,\ continent \to name,\ percent \to percent] =$ *code ↦ "R", name ↦ "Asia", percent ↦ 80 that now fits into the schema* $new\_encompasses(code, name, percent)$*.* □

The usefulness of renaming will become clear later ...

Expressions in the Relational Algebra

What is an algebra?

- An algebra consists of a "domain" (i.e., a set of "things"), and a set of operators.
- Operators map elements of the domain to other elements of the domain.
- Each of the operators has a "semantics", that is, a definition how the result of applying it to some input should look like.
- **Algebra expressions** are built over basic constants and operators (inductive definition).

Relational Algebra

- The "domain" consists of all relations (over arbitrary sets of attributes).
- The operators are then used for combining relations, and for describing computations - e.g., in SQL.
- **Relational algebra expressions** are defined inductively over relations and operators.
- Relational algebra expressions define queries against a relational database.

Inductive Definition of Expressions

Atomic Expressions - Base Cases of the Inductive Definition

- For an arbitrary attribute A and a constant $c \in$ dom(A), the **constant relation** $A : \{c\}$ is an algebra expression.

  Format: $[A]$
  Result relation: $\{\mu\}$ with $\mu = (A \mapsto c)$

A:{c}
A
c

- Given a database schema $\mathbf{R} = \{R_1(\bar{X}_1), \ldots, R_n(\bar{X}_n)\}$, every relation name R_i is an algebra expression.

  Format of R_i: $\bar{X}_i$
  Result relation (wrt. a given database state $\mathcal{S}$): the relation $\mathcal{S}(R_i)$ that is currently stored in the database.

Structural Induction: Applying an Operator

- takes one or more input relations $in_1, in_2, \ldots$
- produces a result relation out:
  - out has a **format**,
    depends on the formats of the input relations.
  - out is a relation, i.e., it contains some tuples,
    depends on the content of the input relations.

Base Operators

Let $\bar{X}, \bar{Y}$ formats and $r \in \text{Rel}(\bar{X})$ and $s \in \text{Rel}(\bar{Y})$ relations over $\bar{X}$ and $\bar{Y}$.

Union

Assume $r, s \in \text{Rel}(\bar{X})$.
Result format of $r \cup s$: $\bar{X}$
Result relation: $r \cup s = \{\mu \in \text{Tup}(\bar{X}) \mid \mu \in r \text{ or } \mu \in s\}$.

$r =$

A	B	C
a	b	c
d	a	f
c	b	d

$s =$

A	B	C
b	g	a
d	a	f

$r \cup s =$

A	B	C
a	b	c
d	a	f
c	b	d
b	g	a

Set Difference

Assume $r, s \in \text{Rel}(\bar{X})$.
Result format of $r \setminus s$: $\bar{X}$
Result relation: $r \setminus s = \{\mu \in r \mid \mu \notin s\}$.

$$r = \begin{array}{ccc} A & B & C \\ \hline a & b & c \\ d & a & f \\ c & b & d \end{array} \qquad s = \begin{array}{ccc} A & B & C \\ \hline b & g & a \\ d & a & f \end{array} \qquad r \setminus s = \begin{array}{ccc} A & B & C \\ \hline a & b & c \\ c & b & d \end{array}$$

Projection (Reduction to a subset of the attributes)

Assume $r \in \text{Rel}(\bar{X})$ and $\bar{Y} \subseteq \bar{X}$.
Result format of $\pi[\bar{Y}](r)$: $\bar{Y}$
Result relation: $\pi[\bar{Y}](r) = \{\mu[\bar{Y}] \mid \mu \in r\}$.

Example 3.4

Continent	
name	***area***
Europe	*10523000*
Africa	*30221500*
Asia	*44614500*
N. America	*24709000*
S. America	*17840000*
Australia	*9000000*

Let $\bar{Y} = [name]$

$\mu_1[name] = \boxed{\textit{name} \mapsto \text{“Europe”}}$
$\mu_2[name] = \boxed{\textit{name} \mapsto \text{“Africa”}}$
$\mu_3[name] = \boxed{\textit{name} \mapsto \text{“Asia”}}$
$\mu_4[name] = \boxed{\textit{name} \mapsto \text{“N.America”}}$
$\mu_4[name] = \boxed{\textit{name} \mapsto \text{“S.America”}}$
$\mu_5[name] = \boxed{\textit{name} \mapsto \text{“Australia”}}$

$\pi[name]$(***Continent***)
name
Europe
Africa
Asia
N.America
S.America
Australia

□

Selection (Reduction of number of tuples by a condition)

Assume $r \in$ Rel$(\bar{X})$ and a selection condition α over $\bar{X}$.

Result format of $\sigma[\alpha](r)$: $\bar{X}$
Result relation: $\sigma[\alpha](r) = \{\mu \in r \mid \mu$ satisfies $\alpha\}$.

Example 3.5

Continent	
name	**area**
Europe	*10523000*
Africa	*30221500*
Asia	*44614500*
N. America	*24709000*
S. America	*17840000*
Australia	*9000000*

Let $\alpha =$ *"area* > 20000000*"*

$\mu_1(area) > 20000000$ *?– no*
$\mu_2(area) > 20000000$ *?– yes*
$\mu_3(area) > 20000000$ *?– yes*
$\mu_4(area) > 20000000$ *?– yes*
$\mu_4(area) > 20000000$ *?– no*
$\mu_5(area) > 20000000$ *?– no*

σ[*area* $> 20E6$](***Continent***)	
name	**area**
Africa	*30221500*
Asia	*44614500*
N.America	*24709000*

□

Renaming (of attributes)

Assume $r \in$ Rel$(\bar{X})$ with $\bar{X} = [A_1, \ldots, A_k]$ and a renaming $[A_1 \to B_1, \ldots, A_k \to B_k]$.

Result format of $\rho[A_1 \to B_1, \ldots, A_k \to B_k](r)$: $[B_1, \ldots, B_k]$
Result relation: $\rho[A_1 \to B_1, \ldots, A_k \to B_k](r) = \{\mu[A_1 \to B_1, \ldots, A_k \to B_k] \mid \mu \in r\}$.

Example 3.6
Consider the renaming of the table $encompasses(country, continent, percent)$*:*

$\bar{X} = [country,\ continent,\ percent]$
Renaming: $\bar{Y} = [code, name, percent]$

$\rho[country \to code,\ continent \to name,\ percent \to percent]$(***encompasses***)		
code	**name**	**percent**
R	*Europe*	*20*
R	*Asia*	*80*
D	*Europe*	*100*
⋮	⋮	⋮

□

(Natural) Join (Combining two relations via common attributes)

Assume $r \in \text{Rel}(\bar{X})$ and $s \in \text{Rel}(\bar{Y})$ for arbitrary $\bar{X}, \bar{Y}$.

Convention: For $\bar{X} \cup \bar{Y}$, as a shorthand, write $\overline{XY}$.

for two tuples $\mu_1 = \boxed{v_1, \ldots, v_n}$ and $\mu_2 = \boxed{w_1, \ldots, w_m}$, $\mu_1\mu_2 := \boxed{v_1, \ldots, v_n, w_1, \ldots, w_m}$.

Result format of $r \bowtie s$: $\overline{XY}$.

Result relation: $r \bowtie s = \{\mu \in \text{Tup}(\overline{XY}) \mid \mu[\bar{X}] \in r \text{ and } \mu[\bar{Y}] \in s\}$.

Motivation

Simplest Case: $\bar{X} \cap \bar{Y} = \emptyset \Rightarrow$ Cartesian Product $r \bowtie s = r \times s$

$r \times s = \{\mu_1\mu_2 \in \text{Tup}(\overline{XY}) \mid \mu_1 \in r \text{ and } \mu_2 \in s\}$.

$r =$

A	B
1	2
4	5

$s =$

C	D
a	b
c	d
e	f

$r \bowtie s =$

A	B	C	D
1	2	a	b
1	2	c	d
1	2	e	f
4	5	a	b
4	5	c	d
4	5	e	f

Example 3.7 (Cartesian Product of Continent and Encompasses)

The cartesian product combines everything with everything, not only "meaningful" combinations:

Continent* $\times$ *encompasses				
name	***area***	***continent***	***country***	***percent***
Europe	*10523000*	*Europe*	*D*	*100*
Europe	*10523000*	*Europe*	*R*	*20*
Europe	*10523000*	*Asia*	*R*	*80*
Europe	*10523000*	*:*	*:*	*:*
Africa	*30221500*	*Europe*	*D*	*100*
Africa	*30221500*	*Europe*	*R*	*20*
Africa	*30221500*	*Asia*	*R*	*80*
Africa	*30221500*	*:*	*:*	*:*
Asia	*44614500*	*Europe*	*D*	*100*
Asia	*44614500*	*Europe*	*R*	*20*
Asia	*44614500*	*Asia*	*R*	*80*
Asia	*44614500*	*:*	*:*	*:*
:	*:*	*:*	*:*	*:*

Back to the Natural Join

General case $\bar{X} \cap \bar{Y} \neq \emptyset$: shared attribute names constrain the result relation.

Again the definition: $r \bowtie s = \{\mu \in \text{Tup}(\overline{XY}) \mid \mu[\bar{X}] \in r \text{ and } \mu[\bar{Y}] \in s\}$.

(Note: this implies that the tuples $\mu_1 := \mu[\bar{X}] \in r$ and $\mu_2 := \mu[\bar{Y}] \in s$ coincide in the shared attributes $\bar{X} \cap \bar{Y}$)

Example 3.8

Consider encompasses(country,continent,percent) and isMember(organization,country,type):

encompasses		
country	***continent***	***percent***
R	*Europe*	*20*
R	*Asia*	*80*
D	*Europe*	*100*
:	*:*	*:*

isMember		
organization	***country***	***type***
EU	*D*	*member*
UN	*D*	*member*
UN	*R*	*member*
:	*:*	*:*

$$encompasses \bowtie isMember = \{\mu \in \textit{Tup}(country, cont, perc, org, type) \mid$$
$$\mu[country, cont, perc] \in encompasses \text{ and } \mu[org, country, type] \in isMember\}$$

□

Example 3.8 (Continued)

$$encompasses \bowtie isMember = \{\mu \in \textit{Tup}(country, cont, perc, org, type) \mid$$
$$\mu[country, cont, perc] \in encompasses \text{ and } \mu[org, country, type] \in isMember\}$$

start with $(R, Europe, 20) \in encompasses$.
check which tuples in $isMember$ *match:*
- $(UN, R, member) \in isMember$ *matches:*
  result: $(R, Europe, 20, UN, member)$ *belongs to the result.*
  (some more matches ...)

continue with $(R, Asia, 80) \in encompasses$.
- $(UN, R, member) \in isMember$ *matches:*
  result: $(R, Asia, 80, UN, member)$ *belongs to the result.*
  (some more matches ...)

continue with $(D, Europe, 100) \in encompasses$.
- $(EU, D, member) \in isMember$ *matches:*
  result: $(D, Europe, 100, EU, member)$ *belongs to the result.*
- $(UN, D, member) \in isMember$ *matches:*
  result: $(D, Europe, 100, UN, member)$ *belongs to the result.*
  (some more matches ...)

□

Example 3.8 (Continued)
Result:

encompasses $\bowtie$ *isMember*				
country	***continent***	***percent***	***organization***	***type***
R	*Europe*	*20*	*UN*	*member*
R	*Europe*	*20*	*:*	*:*
R	*Asia*	*80*	*UN*	*member*
R	*Asia*	*80*	*:*	*:*
D	*Europe*	*100*	*UN*	*member*
D	*Europe*	*100*	*EU*	*member*
D	*Europe*	*100*	*:*	*:*
:	*:*	*:*	*:*	*:*

□

Example 3.9 (and Exercise)
Consider the expression

$$Continent \bowtie \rho[country \to code,\ continent \to name,\ percent \to percent](encompasses)$$

□

Functionalities of the Join

- Combining relations
- Selective functionality: only matching tuples survive
  (consider joining cities and organizations on headquarters)

Derived Operators

Intersection

Assume $r, s \in \text{Rel}(\bar{X})$.

Then, $r \cap s = \{\mu \in \text{Tup}(\bar{X}) \mid \mu \in r \text{ and } \mu \in s\}$.

Theorem 3.1
Intersection can be expressed by difference: $r \cap s = r \setminus (r \setminus s)$.

□

θ-Join

Combination of Cartesian Product and Selection:

Assume $r \in \text{Rel}(\bar{X})$, and $s \in \text{Rel}(\bar{Y})$, such that $\bar{X} \cap \bar{Y} = \emptyset$, and $A\,\theta\,B$ a selection condition.

$$r \bowtie_{A\theta B} s = \{\mu \in \text{Tup}(\overline{XY}) \mid \mu[\bar{X}] \in r,\ \mu[\bar{Y}] \in s \text{ and } \mu \text{ satisfies } A\theta B\} = \sigma[A\theta B](r \times s).$$

Equi-Join

θ-join that uses the "="-predicate.

Example 3.10 (and Exercise)

Consider again Example 3.7:

$Continent \bowtie encompasses \;=\; Continent \times encompasses$ *contained tuples that did not really make sense.*

$Continent \bowtie_{continent=name} encompasses$ *would be more useful.*

Furthermore, consider
$\pi[continent, area, code, percent](Continent \bowtie_{continent=name} encompasses)$*:*

- *removes the - now redundant - "name" column,*
- *is equivalent to the natural join* $(\rho[name \to continent](continent)) \bowtie encompasses.$ $\square$

Semi-Join

- recall: joins combine, but are also selective
- semi-join acts like a selection on a relation r:
  selection condition not given as a boolean formula on the attributes of r, but by "looking into" another relation (a subquery)

Assume $r \in \text{Rel}(\bar{X})$ and $s \in \text{Rel}(\bar{Y})$ such that $\bar{X} \cap \bar{Y} \neq \emptyset$.

Result format of $r \ltimes s$: $\bar{X}$
Result relation: $r \ltimes s = \pi[\bar{X}](r \bowtie s)$

The semi-join $r \ltimes s$ does *not* return the join, but checks which tuples of r "survive" the join with s (i.e., "which find a counterpart in s wrt. the shared attributes"):

- Used with subqueries: (main query) $\ltimes$ (subquery)
- $r \ltimes s \subseteq r$
- Used for optimizing the evaluation of joins (often in combination with indexes).

Semi-Join: Example

Give the names of all countries where a city with at least 1000000 inhabitants is located:

π[name]
$\ltimes$ Country.code=City.country
Country σ[population$>$1000000]
City

- Have a short look “inside” the subquery, but dont’ actually use it:
- look only if there is a big city in this country.
- “if the country code is in the set of country codes ...”:

π[name]
$\ltimes$ Country.code=City.country
Country π[country]   and put an index on the result set
σ[population$>$1000000]
City

Towards the Outer Join

- The (inner) join is the operator for combining relations

Example 3.11

- *Persons work in divisions of a company, tools are assigned to the divisions:*

Works	
Person	***Division***
John	*Production*
Bill	*Production*
John	*Research*
Mary	*Research*
Sue	*Sales*

Tools	
Division	***Tool***
Production	*hammer*
Research	*pen*
Research	*computer*
Admin.	*typewriter*

Works $\bowtie$ Tools		
Person	***Division***	***Tool***
John	*Production*	*hammer*
Bill	*Production*	*hammer*
John	*Research*	*pen*
John	*Research*	*computer*
Mary	*Research*	*pen*
Mary	*Research*	*computer*

- *join contains no tuple that describes Sue,*
- *join contains no tuple that describes the administration or sales division,*
- *join contains no tuple that shows that there is a typewriter.* □

Outer Join

Assume $r \in \text{Rel}(\bar{X})$ and $s \in \text{Rel}(\bar{Y})$.

Result format of $r \;⟗\; s$: $\overline{XY}$

The outer join extends the "inner" join with all tuples that have no counterpart in the other relation (filled with null values):

Example 3.12 (Outer Join)

Consider again Example 3.11

Works ⟗ Tools		
Person	***Division***	***Tool***
John	*Production*	*hammer*
Bill	*Production*	*hammer*
John	*Research*	*pen*
John	*Research*	*computer*
Mary	*Research*	*pen*
Mary	*Research*	*computer*
Sue	*Sales*	*NULL*
NULL	*Admin*	*typewriter*

Works ⋉ Tools	
Person	***Division***
John	*Production*
Bill	*Production*
John	*Research*
Mary	*Research*

Works ⋊ Tools	
Division	***Tool***
Production	*hammer*
Research	*pen*
Research	*computer*

Formally, the result relation $r \;⟗\; s$ is defined as follows:

$J = r \bowtie s$ — take the ("inner") join as base

$r_0 = r \setminus \pi[\bar{X}](J) = r \setminus (r \ltimes s)$ — r-tuples that "are missing"

$s_0 = s \setminus \pi[\bar{Y}](J) = s \setminus (r \rtimes s)$ — s-tuples that "are missing"

$\bar{Y}_0 = \bar{Y} \setminus \bar{X}$, $\bar{X}_0 = \bar{X} \setminus \bar{Y}$

Let $\mu_s \in \text{Tup}(\bar{Y}_0)$, $\mu_r \in \text{Tup}(\bar{X}_0)$ such that μ_s, μ_r consist only of *null* values

$$r \;⟗\; s = J \cup (r_0 \times \{\mu_s\}) \cup (s_0 \times \{\mu_r\}) \; .$$

Example 3.12 (Continued)

For the above example,

$J =$ *Works* $\bowtie$ *Tools*

$r_0 =$ *["Sue","Sales"]*, $s_0 =$ *["Admin","Typewriter"]*

$\bar{Y}_0 =$ *Tool*, $\bar{X}_0 =$ *Person*

$\mu_s =$

Tool
null

$\mu_r =$

Person
null

$r_0 \times \{\mu_s\} =$

Person	***Division***	***Tool***
Sue	*Sales*	*null*

$s_0 \times \{\mu_r\} =$

Person	***Division***	***Tool***
null	*Admin*	*Typewriter*

□

Left and Right Outer Join

Analogously to the (full) outer join:

- $r \sqsupset\!\bowtie s = J \cup (r_0 \times \{\mu_s\})$.
- $r \bowtie\!\sqsubset s = J \cup (s_0 \times \{\mu_r\})$.

Generalized Natural Join

Assume $r_i \subseteq \text{Tup}(\bar{X}_i)$.

Result format: $\cup_{i=1}^{n} \bar{X}_i$
Result relation: $\bowtie_{i=1}^{n} r_i = \{\mu \in \text{Tup}(\cup_{i=1}^{n} \bar{X}_i) \mid \mu[\bar{X}_i] \in r_i\}$

Exercise 3.1
Prove that the Generalized Natural Join *is well-defined, i.e., that the order how to join the* r_i *does not matter.*
Proceed as follows:

- *Show that the natural join is commutative,*
- *Show that the natural join is associative,*
- *... then complete the proof.*

□

Relational Division

Assume $r \in \text{Rel}(\bar{X})$ and $s \in \text{Rel}(\bar{Y})$ such that $\bar{Y} \subsetneq \bar{X}$.
Result format of $r \div s$: $\bar{Z} = \bar{X} \setminus \bar{Y}$.

The result relation $r \div s$ is specified as "all $\bar{Z}$-values that occur in $\pi[\bar{Z}](r)$, with the additional condition that they occur in r together with each of the $\bar{Y}$ values that occur in s".

Formally,

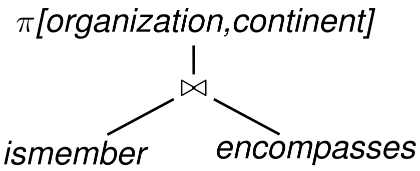
$$r \div s = \{\mu \in \text{Tup}(\bar{Z}) \mid \mu \in \pi[\bar{Z}](r) \wedge \{\mu\} \times s \subseteq r\} = \pi[\bar{Z}](r) \setminus \pi[\bar{Z}]((\pi[\bar{Z}](r) \times s) \setminus r).$$

- Simple observation: $\pi[\bar{Z}](r) \supseteq r \div s$.
  This constrains the set of possible results.
- Often, $\bar{Z}$ and $\bar{Y}$ correspond to the keys of relations that represent the instances of entity types.
- Exercise: the explicit "$\mu \in \pi[\bar{Z}](r)$" in the first characterization looks a bit redundant. Is it? – or why not?

Example 3.13 (Relational Division)

Compute those organizations that have at least one member on each continent:

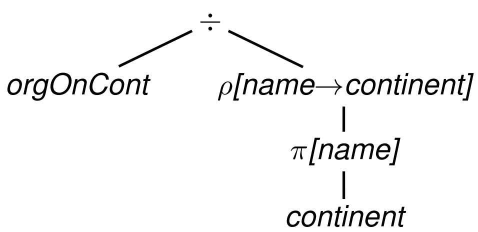
First step: which organizations have (some) member on which continents:




```
SELECT DISTINCT i.organization, e.continent
FROM ismember i, encompasses e
WHERE i.country=e.country
ORDER by 1
```

orgOnCont	
organization	**continent**
UN	Europe
UN	Asia
UN	N.America
UN	S.America
UN	Africa
UN	Australia
NATO	Europe
NATO	N.America
NATO	Asia
EU	Europe
:	:

Example 3.13 (Cont'd)



$r(\bar{X}),\ \ s(\bar{Y}),\ \ \bar{Z} := \bar{X} \setminus \bar{Y}$

$r \div s = \{\ \ \mu \in \mathit{Tup}(\bar{Z}) \mid$

$\qquad \mu \in \pi[\bar{Z}](r) \wedge \{\mu\} \times s \subseteq r\ \}$

$\bar{X}$ = [*organization, continent*]
$\bar{Y}$ = [*continent*]
Thus, $\bar{Z}$ = [*organization*].

orgOnCont	
organization	**continent**
UN	Europe
UN	Asia
UN	N.America
UN	S.America
UN	Africa
UN	Australia
NATO	Europe
NATO	N.America
NATO	Asia
EU	Europe
:	:

ρ[name→continent] (π[name](continent))
continent
Asia
Europe
Australia
N.America
S.America
Africa

- *UN: occurs with each continent in orgOnCont ⇒ belongs to the result.*
- *NATO: does not occur with each continent in orgOnCont ⇒ does not belong to the result.*
- *EU: does not occur with each continent in orgOnCont ⇒ does not belong to the result.*

Example 3.13 (Cont'd)
Consider again the formal algebraic characterization of the division:

$$r \div s = \{\mu \in \mathit{Tup}(\bar{Z}) \mid \mu \in \pi[\bar{Z}](r) \wedge \{\mu\} \times s \subseteq r\} = \pi[\bar{Z}](r) \setminus \pi[\bar{Z}]((\pi[\bar{Z}](r) \times s) \setminus r).$$

1. $r =$ *orgOnCont,* $s = \pi[\mathit{name}](\mathit{continent})$, $Z =$ *Country.*
2. $(\pi[\bar{Z}](r) \times s)$ *contains all tuples of organizations with each of the continents, e.g., (NATO,Europe), (NATO,Asia), (NATO,N.America), (NATO,S.America), (NATO,Africa), (NATO,Australia).*
3. $((\pi[\bar{Z}](r) \times s) \setminus r)$ *contains all such tuples which are not "valid", e.g., (NATO,Africa).*
4. *projecting this to the organizations yields all those organizations where a non-valid tuple has been generated in (2), i.e., that have no member on some continent (e.g., NATO).*
5. $\pi[\bar{Z}](r)$ *is the list of* all *organizations ...*
6. *... subtracting those computed in (4) yields those that have a member on* each *continent.* □

EXPRESSIONS

- inductively defined: combining expressions by operators

Example 3.14
The names of all cities where (i) headquarters of an organization are located, and (ii) that are capitals of a member country of this organization.

As a tree:

π*[City]*

$\cap$

π*[abbrev,city,prov,country]* — *Organization*

ρ*[capital→city]* — π*[abbrev,capital,prov,country]*

$\bowtie$

ρ*[organization→abbrev]* — *isMember*

ρ*[code→country]* — *Country*

□

Note that there are many equivalent expressions.

EXPRESSIONS IN THE RELATIONAL ALGEBRA AS QUERIES

Let $\mathbf{R} = \{R_1, \ldots, R_k\}$ a set of relation schemata of the form $R_i(\bar{X}_i)$. As already described, an **database state** to $\mathbf{R}$ is a **structure** $\mathcal{S}$ that maps every relation name R_i in $\mathbf{R}$ to a relation $\mathcal{S}(R_i) \subseteq \text{Tup}(\bar{X}_i)$

Every algebra expression Q defines a **query** against the state $\mathcal{S}$ of the database:

- For given $\mathbf{R}$, Q is assigned a **format** Σ_Q (the format of the answer).
- For every database state $\mathcal{S}$, $\mathcal{S}(Q) \subseteq \text{Tup}(\Sigma_Q)$ is a relation over Σ_Q, called the **answer set** for Q wrt. $\mathcal{S}$.
- $\mathcal{S}(Q)$ can be computed according to the inductive definition, starting with the innermost (atomic) subexpressions.
- Thus, the relational algebra has a **functional semantics**.

SUMMARY: INDUCTIVE DEFINITION OF EXPRESSIONS

Atomic Expressions

- For an arbitrary attribute A and a constant $a \in \text{dom}(A)$, the **constant relation** $A : \{a\}$ is an algebra expression.

  $\Sigma_{A:\{a\}} = [A]$ and $\mathcal{S}(A : \{a\}) = A : \{a\}$
- Every relation name R is an algebra expression.

  $\Sigma_R = \bar{X}$ and $\mathcal{S}(R) = \mathcal{S}(R)$.

Summary (Cont'd)

Compound Expressions

Assume algebra expressions Q_1, Q_2 that define Σ_{Q_1}, Σ_{Q_2}, $\mathcal{S}(Q_1)$, and $\mathcal{S}(Q_2)$.

Compound algebraic expressions are now formed by the following rules (corresponding to the algebra operators):

Union

If $\Sigma_{Q_1} = \Sigma_{Q_2}$, then $Q = (Q_1 \cup Q_2)$ is the **union** of Q_1 and Q_2.

$\Sigma_Q = \Sigma_{Q_1}$ and $\mathcal{S}(Q) = \mathcal{S}(Q_1) \cup \mathcal{S}(Q_2)$.

Difference

If $\Sigma_{Q_1} = \Sigma_{Q_2}$, then $Q = (Q_1 \setminus Q_2)$ is the **difference** of Q_1 and Q_2.

$\Sigma_Q = \Sigma_{Q_1}$ and $\mathcal{S}(Q) = \mathcal{S}(Q_1) \setminus \mathcal{S}(Q_2)$.

Inductive Definition of Expressions (Cont'd)

Selection

For a selection condition α over Σ_{Q_1}, $Q = \sigma[\alpha](Q_1)$ is the **selection** from Q_1 wrt. α.

$\Sigma_Q = \Sigma_{Q_1}$ and $\mathcal{S}(Q) = \sigma[\alpha](\mathcal{S}(Q_1))$.

Projection

For $\emptyset \neq \bar{Y} \subseteq \Sigma_{Q_1}$, $Q = \pi[\bar{Y}](Q_1)$ is the **projection** of Q_1 to the attributes in $\bar{Y}$.

$\Sigma_Q = \bar{Y}$ and $\mathcal{S}(Q) = \pi[\bar{Y}](\mathcal{S}(Q_1))$.

Natural Join

$Q = (Q_1 \bowtie Q_2)$ is the **(natural) join** of Q_1 and Q_2.

$\Sigma_Q = \Sigma_{Q_1} \cup \Sigma_{Q_2}$ and $\mathcal{S}(Q) = \mathcal{S}(Q_1) \bowtie \mathcal{S}(Q_2)$.

Renaming

For $\Sigma_{Q_1} = \{A_1, \ldots, A_k\}$ and $\{B_1, \ldots, B_k\}$ a set of attributes,
$Q = \rho[A_1 \to B_1, \ldots, A_k \to B_k](Q_1)$ is the **renaming** of Q_1

$\Sigma_Q = \{B_1, \ldots, B_k\}$ and $\mathcal{S}(Q) = \rho[A_1 \to B_1, \ldots, A_k \to B_k](\mathcal{S}(Q_1))$.

Example

Example 3.15

Professor(PNr, Name, Office), Course(CNr, Credits, CName)
teach(PNr, CNr), examine(PNr, CNr)

- *For each professor (name) determine the courses he gives (CName).*

$$\pi \text{ [Name, CName] ((Professor} \bowtie \text{teach)} \bowtie \text{Course)}$$

- *For each professor (name) determine the courses (CName) that he teaches, but that he does not examine.*

$$\begin{array}{l} \pi[\text{Name, CName}]((\\ (\pi[\text{Name, CNr}](\text{Professor} \bowtie \text{teach})) \\ \setminus \\ (\pi[\text{Name, CNr}](\text{Professor} \bowtie \text{examine})) \\ ) \bowtie \text{Course}) \end{array}$$

Simpler expression:

$$\pi \text{ [Name, CName] ((Professor} \bowtie \text{(teach} \setminus \text{examine))} \bowtie \text{Course)}$$ □

EQUIVALENCE OF EXPRESSIONS

Algebra expressions Q, Q' are called **equivalent**, $Q \equiv Q'$, if and only if for all structures $\mathcal{S}$, $\mathcal{S}(Q) = \mathcal{S}(Q')$.

Equivalence of expressions is the basis for **algebraic optimization**.

Let attr(α) the set of attributes that occur in a selection condition α, and $Q, Q_1, Q_2, \ldots$ expressions with formats X, X_1, ...

Projections

- $\bar{Z}, \bar{Y} \subseteq \bar{X} \Rightarrow \pi[\bar{Z}](\pi[\bar{Y}](Q)) \equiv \pi[\bar{Z} \cap \bar{Y}](Q)$.
- $\bar{Z} \subseteq \bar{Y} \subseteq \bar{X} \Rightarrow \pi[\bar{Z}](\pi[\bar{Y}](Q)) \equiv \pi[\bar{Z}](Q)$.

Selections

- $\sigma[\alpha_1](\sigma[\alpha_2](Q)) \equiv \sigma[\alpha_2](\sigma[\alpha_1](Q)) \equiv \sigma[\alpha_1 \wedge \alpha_2](Q))$.
- attr$(\alpha) \subseteq \bar{Y} \subseteq \bar{X} \Rightarrow \pi[\bar{Y}](\sigma[\alpha](Q)) \equiv \sigma[\alpha](\pi[\bar{Y}](Q))$.

Joins

- $Q_1 \bowtie Q_2 \equiv Q_2 \bowtie Q_1$.
- $(Q_1 \bowtie Q_2) \bowtie Q_3 \equiv Q_1 \bowtie (Q_2 \bowtie Q_3)$.

Equivalence of Expressions (Cont'd)

Joins and other Operations

- attr$(\alpha) \subseteq \bar{X}_1 \cap \bar{X}_2 \Rightarrow \sigma[\alpha](Q_1 \bowtie Q_2) \equiv \sigma[\alpha](Q_1) \bowtie \sigma[\alpha](Q_2)$.
- attr$(\alpha) \subseteq \bar{X}_1,$ attr$(\alpha) \cap \bar{X}_2 = \emptyset \Rightarrow \sigma[\alpha](Q_1 \bowtie Q_2) \equiv (\sigma[\alpha](Q_1)) \bowtie Q_2$.
- Assume $\bar{V} \subseteq \overline{X_1 X_2}$ and let $\bar{W} = \bar{X}_1 \cap \overline{V X_2}$, $\bar{U} = \bar{X}_2 \cap \overline{V X_1}$.
  Then, $\pi[\bar{V}](Q_1 \bowtie Q_2) \equiv \pi[\bar{V}](\pi[\bar{W}](Q_1) \bowtie \pi[\bar{U}](Q_2))$;
  (Note: unary operations bind stronger than binary operations)
- $\bar{X}_2 = \bar{X}_3 \Rightarrow Q_1 \bowtie (Q_2 \text{ op } Q_3) \equiv (Q_1 \bowtie Q_2) \text{ op } (Q_1 \bowtie Q_3)$ where op $\in \{\cup, \setminus\}$.
  (distributivity of $\bowtie$ wrt. $\cup$ and $\setminus$)
  Note the similarity to the arithmetic term algebra: $n \cdot (a \pm b) = (n \cdot a) \pm (n \times b)$

Exercise 3.2
Prove some of the equalities (use the definitions given on the "Base Operators" slide). □

Expressive Power of the Algebra

Transitive Closure

The transitive closure of a binary relation R, denoted by R^* is defined as follows:

$$\begin{aligned} R^1 &= R \\ R^{n+1} &= \{(a,b) \mid \text{there is an } s \text{ s.t. } (a,x) \in R^n \text{ and } (x,b) \in R\} \\ R^* &= \bigcup_{1..\infty} R^n \end{aligned}$$

Examples:

- child(x,y): child* = descendant
- flight connections
- flows_into of rivers in MONDIAL

Theorem 3.2
There is no expression of the relational algebra that computes the transitive closure of arbitrary binary relations r. □

Examples

Time to play. Perhaps postpone examples after comparison with SQL (next subsections)

Aspects

- join as “extending” operation (cartesian product – “all pairs of X and Y such that ...”)
- equijoin as “restricting” operation
- natural join/equijoin in many cases along key/foreign key relationships
- relational division (in case of queries of the style “return all X that are in a given relation with all Y such that ...”)