Chapter 4
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.

4.1 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 50)

- 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 $A\,\theta\,a$, or $a\,\theta\,A$ where $A, B \in \bar{X}$, $a \in \text{dom}(A)$, and θ is a comparison operator in $\{=,\neq,\leq,<,\geq,>\}$.

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

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 4.1

Consider (for a tuple of the table $R(\bar{X}) = continent(Name, Area)$*):* $\bar{X} = [Name,\ Area]$ *and the tuple* $\mu = \boxed{\text{"Asia", 4.50953e+07}}$.

formally: $\mu(Name) =$ *"Asia",* $\mu(Area) = 4.5E7$

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

Result: $\mu[Name] = \boxed{\text{"Asia"}}$ □

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

Selection

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

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

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

Example 4.2

Consider again (for a tuple of the table $R(\bar{X}) = continent(Name, Area)$): $\bar{X} = [Name,\ Area]$.

Selection condition: $Area > 10.000.000$.

Consider again the tuple $\mu =$ $\boxed{\text{“Asia”, 4.50953e+07}}$.

formally: $\mu(Name) =$ “Asia”, $\mu(Area) = 4.5E7$

check: $\mu(Area) > 10.000.000$

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

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 4.3

Consider (for a tuple of the table $R(\bar{X}) = encompasses(Country, Continent, Percent)$):

$\bar{X} = [Country,\ Continent,\ Percent]$.

Consider the tuple $\mu =$ $\boxed{\text{“R”, “Asia”, 80}}$.

formally: $\mu(Country) =$ “R”, $\mu(Continent) =$ “Asia”, $\mu(Percent) = 80$

Renaming: $\bar{Y} = [Code, Name, Percent]$

Result: a new tuple
$\mu[Country \to Code,\ Continent \to Name,\ Percent \to Percent] =$ $\boxed{\text{“R”, “Asia”, 80}}$ 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

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

  Format: $[A]$
  Result relation: $\{a\}$

A:$\{$**a**$\}$
A
a

- 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

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 4.4

Continent	
Name	***Area***
Europe	*9562489.6*
Africa	*3.02547e+07*
Asia	*4.50953e+07*
America	*3.9872e+07*
Australia	*8503474.56*

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

$\mu_1[Name] =$ *“Europe”*
$\mu_2[Name] =$ *“Africa”*
$\mu_3[Name] =$ *“Asia”*
$\mu_4[Name] =$ *“America”*
$\mu_5[Name] =$ *“Australia”*

$\pi[Name]$(***Continent***)
Name
Europe
Africa
Asia
America
Australia

□

Selection

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 4.5

Continent	
Name	***Area***
Europe	*9562489.6*
Africa	*3.02547e+07*
Asia	*4.50953e+07*
America	*3.9872e+07*
Australia	*8503474.56*

Let $\alpha =$ *"$Area > 10.000.000$"*

$\mu_1(Area) < 10.000.000$	*no*
$\mu_2(Area) > 10.000.000$	*yes*
$\mu_2(Area) > 10.000.000$	*yes*
$\mu_2(Area) > 10.000.000$	*yes*
$\mu_2(Area) < 10.000.000$	*no*

$\sigma[Area > 10E6]$(***Continent***)	
Name	***Area***
Africa	*3.02547e+07*
Asia	*4.50953e+07*
America	*3.9872e+07*

□

Renaming

Assume $r \in$ Rel$(\bar{X})$ with $X = [A_1, \ldots, A_k]$ 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 4.6
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*
$\vdots$	$\vdots$	$\vdots$

□

(Natural) Join

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

Convention: Instead of $\bar{X} \cup \bar{Y}$, we also 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 $r \bowtie s = r \times s$

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

$$r = \begin{array}{cc} A & B \\ \hline 1 & 2 \\ 4 & 5 \end{array} \qquad s = \begin{array}{cc} C & D \\ \hline a & b \\ c & d \\ e & f \end{array} \qquad r \bowtie s = \begin{array}{cccc} A & B & C & D \\ \hline 1 & 2 & a & b \\ 1 & 2 & c & d \\ 1 & 2 & e & f \\ 4 & 5 & a & b \\ 4 & 5 & c & d \\ 4 & 5 & e & f \end{array}$$

Example 4.7 (Cartesian Product of Continent and Encompasses)

Continent $\times$ ***encompasses***				
Name	***Area***	***Continent***	***Country***	***Percent***
Europe	*9562489.6*	*Europe*	*Germany*	*100*
Europe	*9562489.6*	*Europe*	*Russia*	*20*
Europe	*9562489.6*	*Asia*	*Russia*	*80*
Europe	*9562489.6*	*:*	*:*	*:*
Africa	*3.02547e+07*	*Europe*	*Germany*	*100*
Africa	*3.02547e+07*	*Europe*	*Russia*	*20*
Africa	*3.02547e+07*	*Asia*	*Russia*	*80*
Africa	*3.02547e+07*	*:*	*:*	*:*
Asia	*4.50953e+07*	*Europe*	*Germany*	*100*
Asia	*4.50953e+07*	*Europe*	*Russia*	*20*
Asia	*4.50953e+07*	*Asia*	*Russia*	*80*
Asia	*4.50953e+07*	*:*	*:*	*:*
:	*:*	*:*	*:*	*:*

□

Back to the Natural Join

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

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

Example 4.8

Consider encompasses(country,continent,percentage) and is_member(organization,country,type):

encompasses		
Country	***Continent***	***Percent***
R	*Europe*	*20*
R	*Asia*	*80*
D	*Europe*	*100*
:	*:*	*:*

is_member		
Organization	***Country***	***Type***
EU	*D*	*member*
UN	*D*	*member*
UN	*R*	*member*
:	*:*	*:*

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

□

Example 4.8 (Continued)

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

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

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

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

Example 4.8 (Continued)
Result:

encompasses $\times$ ***is_member***				
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 4.9 (and Exercise)
Consider the expression

$$continent \bowtie \rho[Country \to Code,\ Continent \to Name,\ Percent \to Percent](\textbf{\textit{encompasses}})$$

□

Derived Operators

Intersection

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

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

Theorem 4.1
Intersection can be expressed by Difference: $r \cap s = r \setminus (r \setminus s)$. □

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,

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

this implies that $\mu \in \pi(\bar{Z})(r)$

- Simple observation: $\pi(\bar{Z})(r) \supseteq r \div s$.
  This constrains the set of possible results.

Example 4.10 (Relational Division)
Compute all countries that belong both to Europe and to Asia:

enc	
country	**continent**
R	Asia
R	Europe
IND	Asia
D	Europe
TR	Asia
TR	Europe
ET	Africa
ET	Asia
CH	Europe
:	:

cts
continent
Asia
Europe

Compute $enc \div cts$:
$\bar{X} = [country, continent]$, $\bar{Y} = [continent]$
Thus, $\bar{Z} = [country]$.
Consider all values in $\pi[country](enc)$:
Start with "R" $\in \pi[country](enc)$:
  for "Asia" $\in cts$: *("R","Asia")* $\in enc$.
  for "Europe" $\in cts$: *("R","Europe")* $\in enc$.
OK. "R" belongs to the result.
Continue with "IND" $\in \pi[country](enc)$:
  for "Asia" $\in cts$: *("IND","Asia")* $\in enc$.
  for "Europe" $\in cts$: *("IND","Europe")* $\notin enc$.
"IND" does not belong to the result.
:
"TR" belongs to the result.
"ET" does not belong to the result.
"CH" does not belong to the result.

□

Example 4.10 (Cont(d))
Consider again Example 4.10 and the formal algebraic characterization of Division:

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

1. $r = belongs\_to$, $s = continent$, $Z = Country$.
2. $(\pi[Z]r \times s)$ *contains all tuples of countries with Europe and Asia, e.g., (Germany,Europe), (Germany,Asia), (Russia,Europe), (Russia,Asia)*
3. $((\pi[Z]r \times s) \setminus r)$ *contains all sich tuples which are not "valid", e.g., (Germany,Asia).*
4. *projecting this to the countries yields all those countries where a non-valid tuple has been generated in (2), i.e., which do not belong both to Europe and Asia.*
5. $\pi[Z]r$ *is the list of all countries ...*
6. *... subtracting those computed in (4) yields those that belong both to Europe and Asia.* □

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 4.11 (and Exercise)
Consider again Example 4.7:

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

$(Continent \times encompasses)_{continent=name}$ *would be more useful.*

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

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

SEVERAL EXTENSIONS OF THE JOIN

- Join is the operator for combining relations

Example 4.12
Consider a completely different database now for investigating joins.

- *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*
Administration	*typewriter*

□

Example 4.12 (Continued)
Consider the join of both tables:

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*

- *there is no tuple that describes Sue*
- *there is no tuple that describes the administration or sales division*
- *there is no tuple that shows that there is a typewriter*

□

Semi-Join

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

Result format: $\bar{X}$

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"):

Example 4.13

Consider again Example 4.12:

Works $\ltimes$ ***Tools***	
Person	***Division***
John	*Production*
Bill	*Production*
John	*Research*
Mary	*Research*

Works $\rtimes$ ***Tools***	
Division	***Tool***
Production	*hammer*
Research	*pen*
Research	*computer*

□

- Semijoins are e.g. used for optimizing the evaluation of multiple joins.
[see lecture on Database Theory]

Outer Join

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

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 4.14 (Outer Join)

Consider again Example 4.12

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*

□

Formally, the result relation 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"
$Y_0 = \bar{Y} \setminus \bar{X}$, $X_0 = \bar{X} \setminus \bar{Y}$
Let $\mu_1 \in \mathrm{Tup}(Y_0)$, $\mu_2 \in \mathrm{Tup}(X_0)$ such that μ_1, μ_2 consist only of *null* values

$$r ⟗ s = J \cup (r_0 \times \{\mu_1\}) \cup (s_0 \times \{\mu_2\}).$$

Example 4.14 (Continued)
For the above example,

$J = Works \bowtie Tools$
$r_0 = [$*"Sue","Sales"*$]$, $s_0 = [$*"Admin","Typewriter"*$]$
$Y_0 = Tool$, $X_0 = Person$

$\mu_1 =$

Tool
$null$

$\mu_2 =$

Person
$null$

$r_0 \times \{\mu_1\} =$

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

$s_0 \times \{\mu_2\} =$

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

□

Generalized Natural Join

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

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

Exercise 4.1
Prove that the natural join is commutative (which makes the generalized natural join well-defined):

$$\begin{aligned} \bowtie_{i=1}^{n} r_i &= ((\ldots((r_1 \bowtie r_2) \bowtie r_3) \bowtie \ldots) \bowtie r_n) \\ &= (r_1 \bowtie (r_2 \ldots (r_{n-1} \bowtie r_n) \ldots)) \end{aligned}$$

□

EXPRESSIONS

- inductively defined: combining expressions by operators

Example 4.15
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:

$$\pi[City]$$

$$\cap$$

$$\pi[Abbrev, City, Prov, Country] \qquad \rho[Capital \to City]$$

Organization $\qquad \pi[Abbrev, Capital, Prov, Country]$

$$\bowtie$$

$$\rho[Organization \to Abbrev] \qquad \rho[Code \to Country]$$

is_Member $\qquad$ *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 $Q = (Q_1 - Q_2)$ is the **difference** of Q_1 and Q_2.

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

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 $\bar{Y}$.

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

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

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, $\rho[A_1 \to B_1, \ldots, A_k \to B_k]Q_1$ is the **renaming** of Q_1

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

Example

Example 4.16
Professor(PNr, Name, Office), Course(CNr, Credits, CName)
teach(PNr, CNr), examine(PNr, CNr)

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

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

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

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

□

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, $\ldots$.

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 $V \subseteq \overline{X_1 X_2}$ and let $W = \bar{X}_1 \cap \overline{V X_2}$, $U = \bar{X}_2 \cap \overline{V X_1}$.
  Then, $\pi[V](Q_1 \bowtie Q_2) = \pi[V](\pi[W](Q_1) \bowtie \pi[U](Q_2))$;
- $\bar{X}_2 = \bar{X}_3 \Rightarrow Q_1 \bowtie (Q_2 \text{ op } Q_3) = (Q_1 \bowtie Q_2) \text{ op } (Q_1 \bowtie Q_3)$ where op $\in \{\cup, -\}$.

Exercise 4.2
Prove some of the equalities (use the definitions given on the "Base Operators" slide). □

Expressive Power of the Algebra

Theorem 4.2
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)

4.2 SQL

SQL: Structured (Standard) Query Language

Literature: A Guide to the SQL Standard, 3rd Edition, C.J. Date and H. Darwen, Addison-Wesley 1993

History: about 1974 as SEQUEL (IBM, System R)

Standardization:

SQL-86 and **SQL-89:** core language, based on existing implementations, including procedural extensions

SQL-92 (SQL2): some additions

SQL-99 (SQL3):

- active rules
- recursion
- objekt-relational and object-oriented concepts

Underlying Data Model

SQL uses the relational model:

- SQL relations are **multisets (bags)** of tuples (i.e., they can contain duplicates)
- Notions: Relation $\leadsto$ Table
  Tuple $\leadsto$ Row
  Attribute $\leadsto$ Column

The relational algebra serves as theoretical base for SQL as a query language.

The Language

Basic structure of SQL queries:

SELECT $A_1, \ldots, A_n$	(...corresponds to π in the algebra)
FROM $R_1, \ldots, R_m$	(...specifies the contributing relations)
WHERE F	(...corresponds to σ in the algebra)

corresponds to the algebra expression $\pi[A_1, \ldots, A_n](\sigma[F](r_1 \times \ldots \times r_m))$

- Note: cartesian product $\rightarrow$ aliasing

General Structure:

SELECT $A_1, \ldots, A_n$	list of attributes
FROM $R_1, \ldots, R_m$	list of relations
WHERE F	condition(s)
GROUP BY $B_1, \ldots, B_k$	list of grouping attributes
HAVING G	condition on groups
ORDER BY H	sort order

Subqueries correspond to subtrees in the relational algebra.

Constructing Queries

For each problem there are multiple possible equivalent queries in SQL (cf. Example 4.15). The choice is a matter of personal taste.

- analyze the problem "systematically":
  - collect all relations (in the FROM clause) that are needed
  - generate a suitable conjunctive WHERE clause

  $\Rightarrow$ leads to a single "broad" SFW query
  (cf. conjunctive queries, relational calculus)
- analyze the problem "top-down":
  - take the relations that directly contribute to the result in the (outer) FROM clause
  - do all further work in correlated subquery/-queries in the WHERE clause

  $\Rightarrow$ leads to a "main" part and nested subproblems
- decomposition of the problem into subproblems:
  - subproblems are solved by nested SFW queries that are combined in the FROM clause of a surrounding query

Comparison

SQL:

SELECT $A_1, \ldots, A_n$ FROM R_1 AS t_1, ..., R_m AS t_m WHERE F

- **equivalent expression in the relational algebra:**

$$\pi[A_1, \ldots, A_n](\sigma[F](r_1 \times \ldots \times r_m))$$

- **Algorithm (nested-loop):**
  FOR each tuple t_1 in relation R_1 DO
  FOR each tuple t_2 in relation R_2 DO
  :
  FOR each tuple t_n in relation R_n DO
  IF WHERE-clause is satisfied (after renaming $t_1, \ldots, t_n$) THEN
  evaluate the SELECT clause and generate the result tuple (projection).

Comparison

- The relational algebra has no notion of grouping and aggregate functions
- SQL has no clause that corresponds to relational division

Example 4.17
Consider again Example 4.10 (Slide 86).

The corresponding SQL formulation that implements division corresponds to the textual

"all countries that occur in $\pi[country](enc)$, with the additional condition that they occur in enc together with each of the $continent$ values that occur in cts",

or equivalent

"all countries c in $\pi[country](enc)$ such that there is no $continent$ value $cont$ in cts such that c does not occur together with $cont$ in enc": □

Example 4.17 (Continued)

"all countries c in $\pi[country](enc)$ such that there is no $continent$ value $cont$ in cts such that c does not occur together with $cont$ in enc":

SELECT enc1.country
FROM enc AS enc1 — *consider enc1.country="R" and enc1.country="D"*
WHERE NOT EXISTS — *correlated subquery*

((SELECT ct — *always*
FROM cts)

"Europe"
"Asia"

MINUS
(SELECT ct
FROM enc AS enc2
WHERE enc1.country = enc2.country
)

for "R":

"R"	"Asia"
"R"	"Europe"

for "D":

"D"	"Europe"

) — *remains: for "R": nothing $\leadsto$ "R" belongs to the result*
for D: "Asia" $\leadsto$ "D" does not belong to the result

Orthogonality

Full orthogonality means that an expression that results in a relation is allowed everywhere, where an input relation is allowed

- subqueries in the FROM clause
- subqueries in the WHERE clause
- combinations of set operations

But:

- Syntax of aggregation functions is not fully orthogonal:
  Not allowed: SUM(SELECT ...)

```
SELECT SUM(pop_biggest)
   FROM (SELECT country, MAX(population) AS pop_biggest
            FROM City
            GROUP BY country);
```

- The language OQL (Object Query Language) uses similar constructs and is fully orthogonal [see Lecture *Information Systems*].