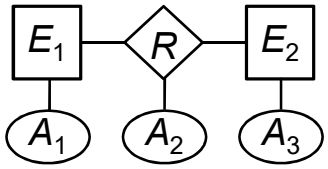
Transformation of an Entity-Relationship Diagram into Relation Schemas

The Big Picture (I)

Entity-Relationship Model

Entities, attributes, relationships

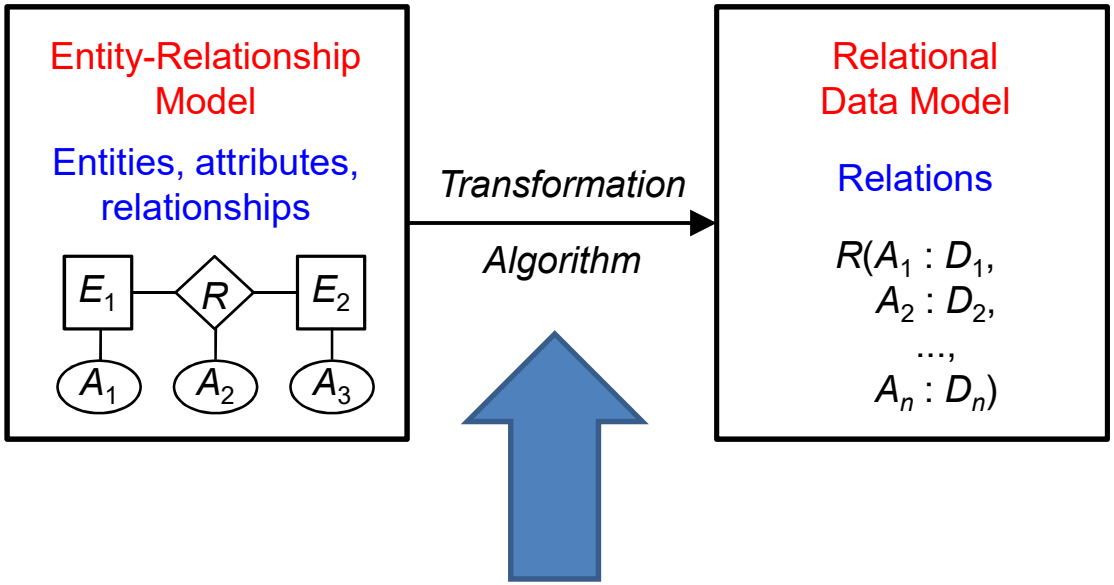


Relational Data Model

Relations

$R(A_1 : D_1,$
$A_2 : D_2,$
$\ldots,$
$A_n : D_n)$

The Big Picture (II)



Data Structures

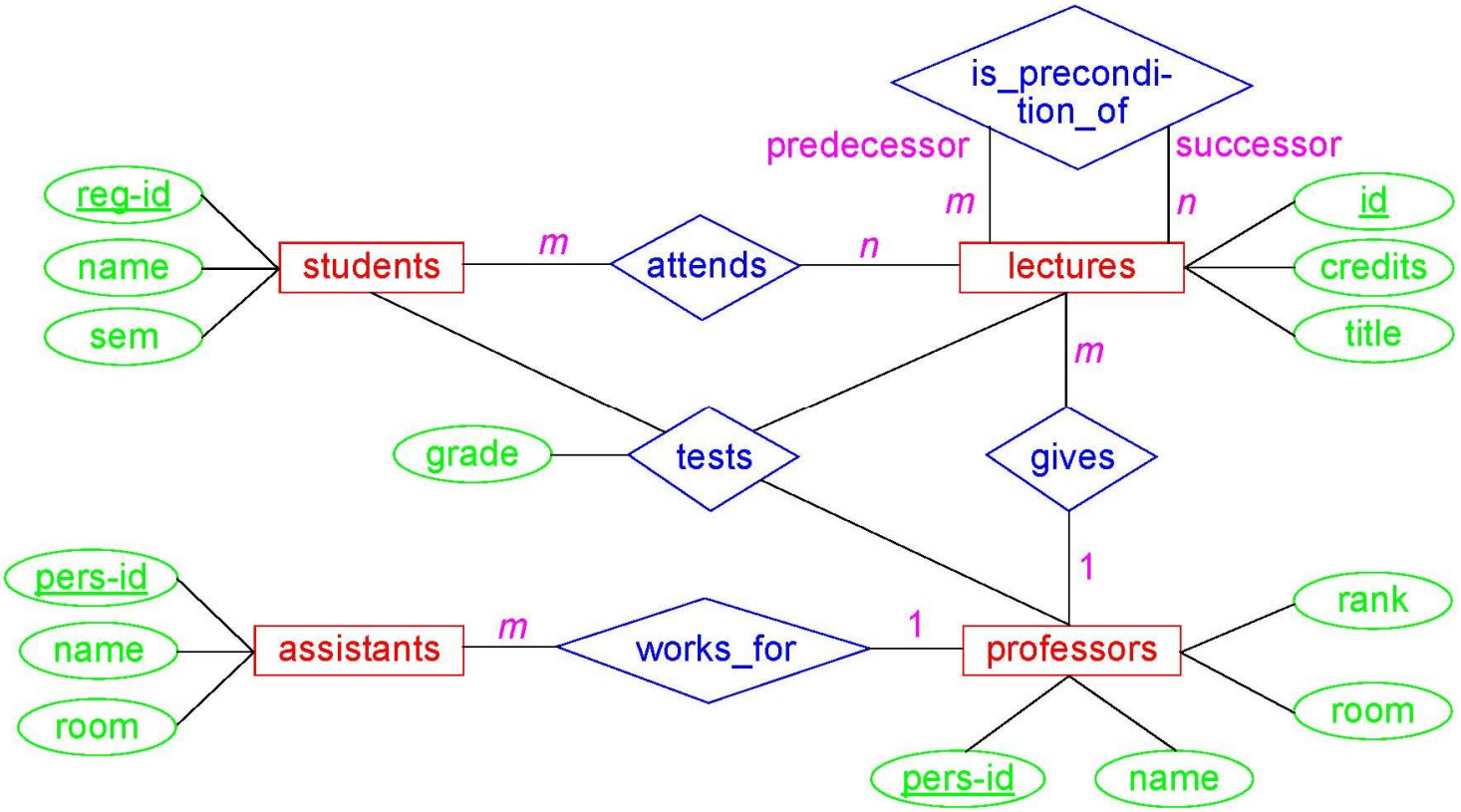
- Objectives
  - Transformation of an E-R diagram into a collection of relation schemas
  - Performance by a small number of relation schemas
  - Correct mapping of theoretical E-R concepts to practical relational database concepts

- Available data structures (structural components)
  - of the E-R model
    - entity sets
    - relationship sets (of different cardinalities)
    - attributes
  - of the relational model
    - relation schemas
    - attributes

Transformation of Strong Entity Sets (I)

- For each strong entity set *E* an independent relation schema *R* is created which comprises all simple attributes of *E*. From a composite attribute only the simple component attributes are taken.
- The attribute names of a relation schema usually correspond to the attribute names of the respective entity set. But names changes are allowed, of course.
- Each attribute name is extended by its domain, that is, data type
- We use the known syntax $R(A_1 : D_1, ..., A_n : D_n)$ where the A_i are attributes and the D_i are their domains
- The key of the entity set becomes the primary key of the relation schema and is underlined
- Each tuple in a relation on *R* corresponds to one entity of *E*
- Example: Conceptual university schema

Transformation of Strong Entity Sets (II)



students(<u>reg-id : *integer*</u>, name : *string*, sem : *integer*)

lectures(<u>id : *integer*</u>, credits : *integer*, title : *string*)

professors(<u>pers-id : *integer*</u>, name : *string*, rank : *string*, room : *integer*)

assistants(<u>pers-id : *integer*</u>, name : *string*, room : *integer*)

Transformation of Weak Entity Sets

- Step 1: For each weak entity set *W* with the respective strong entity set *E*, an independent relation schema *R* is created which comprises all simple attributes and all simple components of composite attributes of *W* as attributes of *R*.

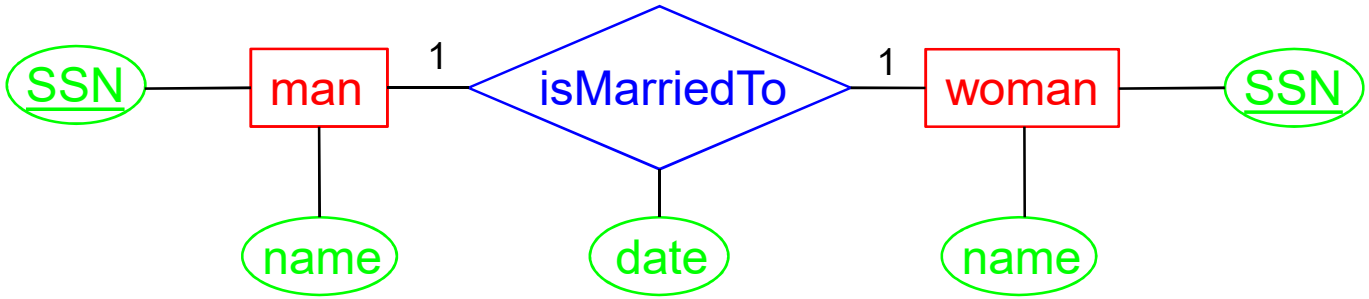
- Step 2: In addition, all primary key attributes of *E* are added to *R* as foreign key attributes. The primary key of *R* then arises from the combination of the primary key of *E* and the partial key of *W*, if the latter one exists.

- Example (separate from the conceptual university schema):

room(<u>rnumber : *integer*</u>, <u>bnumber : *integer*</u>)

Transformation of 1:1-Relationship Sets (I)

- For each binary 1:1-relationship set *R* let *S* and *T* be the relation schemas that correspond to the entity sets participating in *R*. One of the relation schemas, let us say *S*, is selected, and the primary key of *T* is added to *S* as foreign key. If an entity set, for example, *S*, totally participates in *R*, it is advantageous to add the primary key of *T* to *S*. In addition, all simple attributes and all simple components of composite attributes of *R* are taken as attributes of *S*.
- Example 1 (separate from the conceptual university schema)

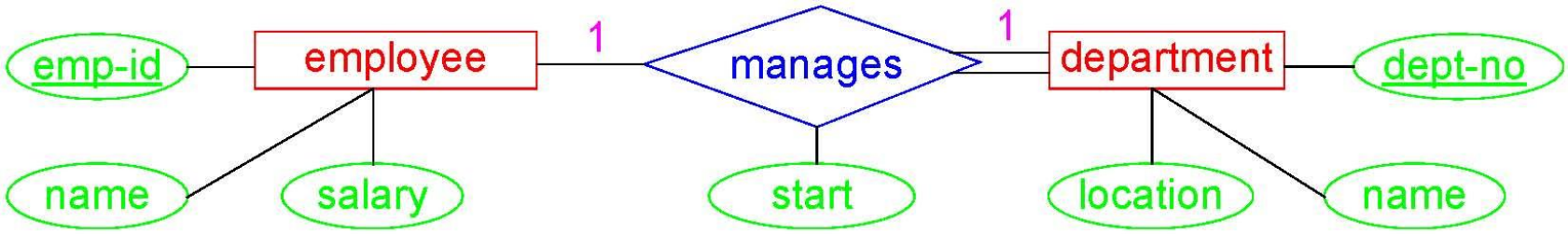


man(SSN : *integer*, name : *string*,
W_SSN : *integer*, date : *time*)
woman(SSN : *integer*, name : *string*)

man(SSN : *integer*, name : *string*)
woman(SSN : *integer*, name : *string*,
M_SSN : *integer*, date : *time*)

Transformation of 1:1-Relationship Sets (II)

- Example 2 (separate from the conceptual university schema)



department(<u>dept-no : *integer*</u>, name : *string*, location : *string*, emp-id: *integer*, start : *date*)

employee(<u>emp-id : *integer*</u>, name : *string*, salary : *double*)

- Each department is linked to exactly one employee
- No null values for attribute “emp-id”

Transformation of 1:*m*- and m:1 Relationship Sets

- For each binary 1:*m*-relationship set *R* let *S* be the relation schema which corresponds to the entity set participating in *R* on the *m*-side. Add to *S* as foreign key the primary key of relation schema *T*, which corresponds to the other entity set participating in *R*. The reason for this is that each entity on the *m*-side is associated with at most one entity on the 1-side of *R*. Furthermore, all simple attributes and all simple components of composite attributes of *R* are taken as attributes of *S*.

- Example (conceptual university schema)

  lectures(id : *integer*, credits : *integer*, title : *string*, held_by : *integer*)

  professors(pers-id : *integer*, name : *string*, room : *string*, rank : *string*)

  assistants(pers-id : *integer*, name : *string*, room : *string*, boss : *integer*)

- The names of attributes of a foreign key have to be changed sometimes in order to ensure the uniqueness of names in a schema

Transformation of *m*:*n*-Relationship Sets

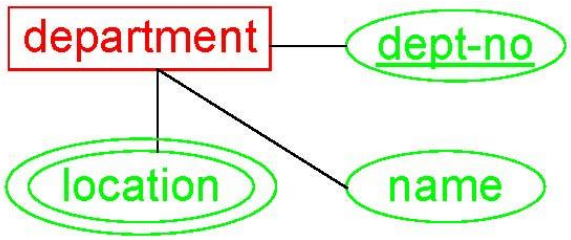
- For each binary *m*:*n*-relationship set *R* a *new* relation schema *S* is created. Add to *S* as foreign keys the primary keys of the relation schemas that correspond to the two entity sets participating in *R*. Their combination forms the primary key of *S*. Furthermore, all simple attributes and all simple components of composite attributes of *R* are taken as attributes of *S*.
- Example (conceptual university schema)

  attends(reg-id : *integer*, id : *integer*)

  is_precondition_of(predecessor : *integer*, successor : *integer*)

Transformation of Multivalued Attributes

- A new relation schema *R* is created for each multivalued attribute *A*. *R* comprises an attribute corresponding to *A* and as a foreign key the primary key *K* of the relation schema which corresponds to the entity set or relationship set containing *A* as an attribute. The primary key of *R* is the combination of *A* and *K*. If the multivalued attribute is composite, its simple components are added to *R*.
- Example (separate from the conceptual university schema)



department(dept-no : *string*, name : *string*)

dept-loc(location : *string*, dept-no : *string*)

dept-loc	location	dept_no
	Office 1	D3
	Office 4	D2
	Office 3	D3
	Office 2	D3

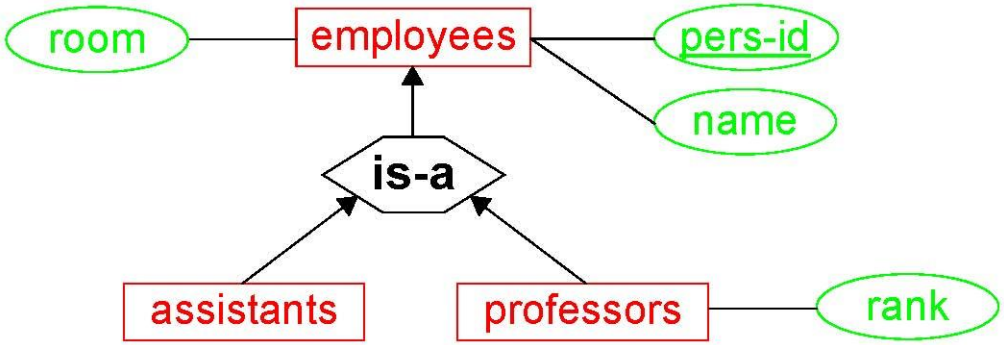
Transformation of *n*-ary Relationship Sets

- For each *n*-ary relationship set *R* with $n > 2$ a *new* relation schema *S* is created. Add to *S* as foreign keys the primary keys of the relation schemas corresponding to entity sets participating in *R*. Furthermore, all simple attributes and all simple components of composite attributes of *R* are taken as attributes of *S*. The primary key of *S* is the combination of all foreign keys.
- Example (conceptual university schema)

  tests(reg-id : *integer*, id : *integer*, pers-id : *integer*, grade : *integer*)

Transformation of Generalizations

- Generalizations are *not* represented by an own relation. The relationship is already expressed by the fact that the key of the common superclass is also used as the key of the specialized subclasses.
- Example:



employees(<u>pers-id : *integer*</u>,
name : *string*,
room : *integer*)

professors(<u>pers-id : *integer*</u>,
rank : *string*)

assistants(<u>pers-id : *integer*</u>)

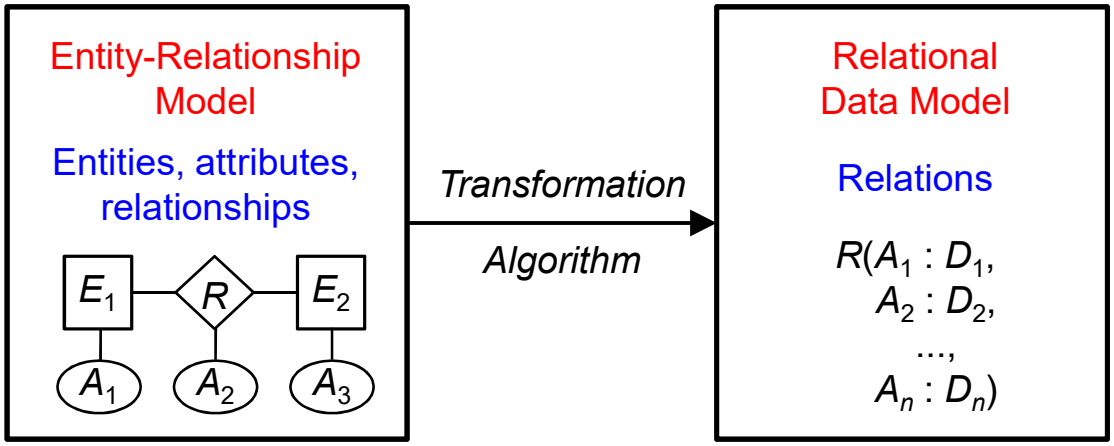
- Information about a professor is distributed to two tuples of two relations, namely to a tuple of the relation *employees* and to a tuple of the relation *professors*
- To obtain the complete information requires a connection of both relations and tuples, respectively (via a join). There is no inheritance in the relational data model.

Complete Conceptual University Schema

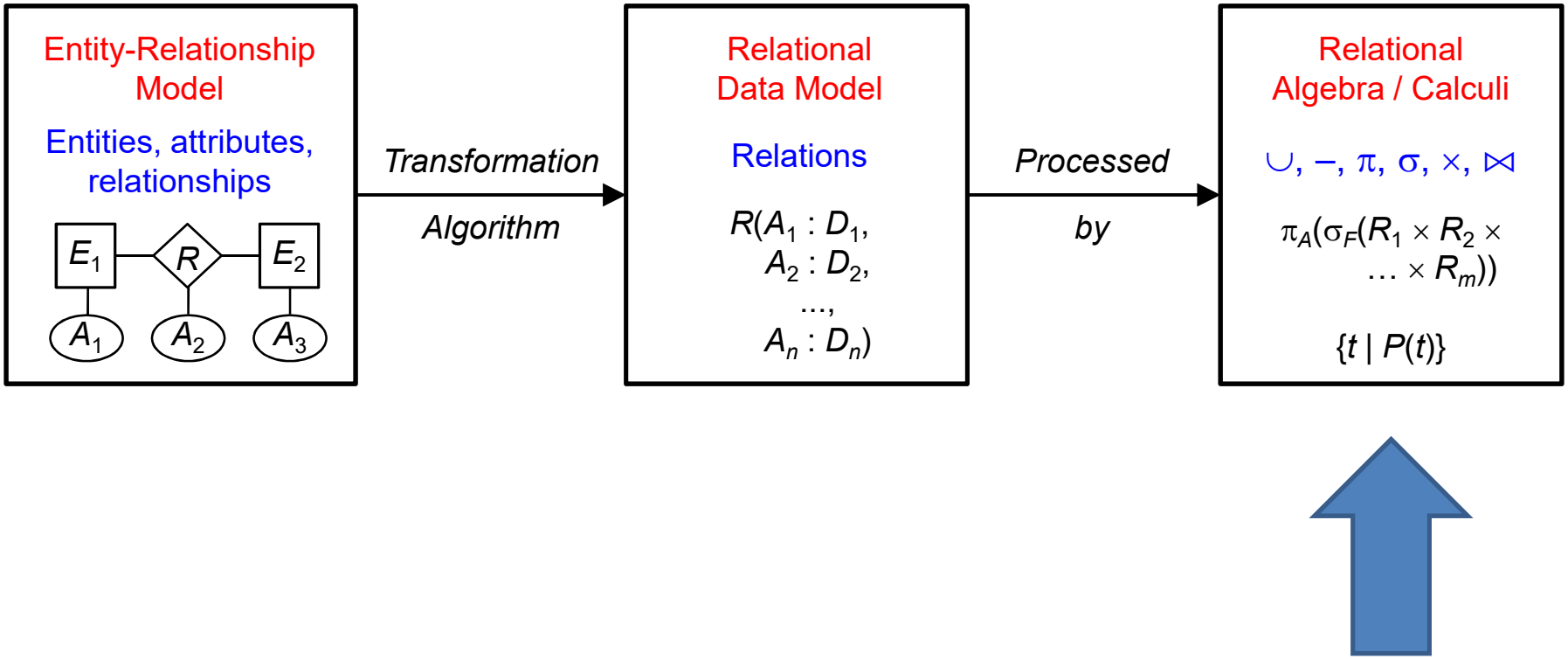
- students(reg-id : *integer*, name : *string*, sem : *integer*)
- lectures(id : *integer*, credits : *integer*, title : *string*, held_by : *integer*)
- professors(pers-id : *integer*, name : *string*, room : *integer*, rank : *string*)
- assistants(pers-id : *integer*, name : *string*, room : *integer*, boss : *integer*)
- attends(reg-id : *integer*, id : *integer*)
- is_precondition_of(predecessor : *integer*, successor : *integer*)
- tests(reg-id : *integer*, id : *integer*, pers-id : *integer*, grade : *integer*)

Relational Algebra

The Big Picture (I)



The Big Picture (II)



Introduction (I)

- So far: Structural description of the data of interest provided by means of a collection of relation schemas
- Questions now are:
  1. What can you do with relations?
  2. How are they queried?
  3. What are the main operations on relations to query them?
  4. How are these operations defined?
  5. How can they be used to extract information from a collection of relation schemas?
- Answer: Two formal query languages are provided for this purpose that cannot be used in practice
  - Relational calculi (discussed in a later lecture)
    - Tuple relational calculus
    - Domain relational calculus
  - Relational Algebra (discussed now)

Introduction (II)

- Relational Calculi
  - Declarative query languages that allow the user to specify *what* the data of interest are, *which* data one would like to retrieve, and which criteria these data have to fulfil
  - The user does not have to specify *how* a query has to be evaluated
- Relational Algebra (RA)
  - Procedural query language that requires from a user to specify *how* a query has to be evaluated
  - Step-by-step specification of the operations to be executed (execution plan, access plan)
  - Relational Algebra is important for the processing of SQL queries
- Only relations possible as input data and output data of RA operations
- Both languages are *closed*, i.e., the results of queries, which operate on relations, are again relations
  [Compare to: The set $\mathbb{N}$ of natural numbers is closed under addition but not under subtraction]

Toy University Database (I)

professors			
pers-id	name	rank	room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	052
2134	Augustinus	C3	309
2136	Curie	C4	036
2137	Kant	C4	007

students		
reg-id	name	sem
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	08
27550	Schopenhauer	06
28106	Carnap	03
29120	Theophastros	02
29555	Feuerbach	02

Toy University Database (II)

lectures			
id	title	credits	held_by
5001	foundations	4	2137
5041	ethics	4	2125
5043	epistemology	3	2126
5049	maieutics	2	2125
4052	logic	4	2125
5052	philosophy of science	3	2126
5216	bioethics	2	2126
5259	The Vienna Circle	2	2133
5022	faith and knowledge	2	2134
4630	The 3 Cutups	4	2137

is_precondition_of	
predecessor	successor
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

Toy University Database (III)

attends	
reg-id	id
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022
29555	5001

assistants			
pers-id	name	room	boss
3002	Platon	156	2125
3003	Aristoteles	199	2125
3004	Wittgenstein	101	2126
3005	Rhetikus	130	2127
3006	Newton	120	2127
3007	Spinoza	155	2134

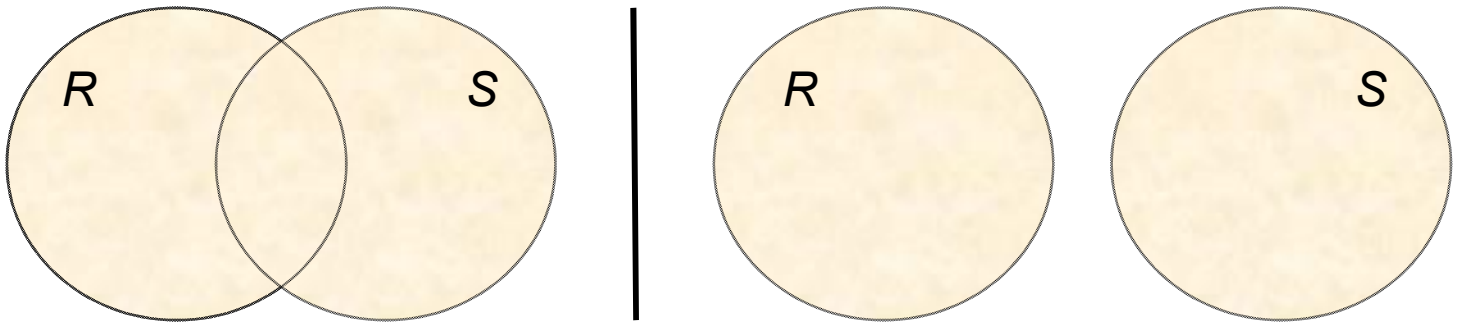
tests			
reg-id	id	pers-id	grade
28106	5001	2126	1
25403	5041	2125	2
27550	4630	2137	2

Overview of Basic Relational Algebra Operations

- The Relational Algebra is a *universal algebra* since it only has one data type, namely relation schemas
- The Relational Algebra comprises 5 (+1) basic operations:
  - Union ($\cup$)
  - Difference ($-$)
  - Cartesian product ($\times$)
  - Project (π)
  - Select (σ)
  - (Rename) (ρ)
- Other algebra operations are derived, that is, they can be expressed by an appropriate combination of the basic operations
- Given: Two relations $R(A_1 : C_1, A_2 : C_2, ..., A_r : C_r)$ and $S(B_1 : D_1, B_2 : D_2, ..., B_s : D_s)$ with arity (or degree) r and s
- R and S are schema compliant (identical except for renaming), if
  1. the equality $r = s$ holds
  2. there exists a permutation φ of the indices $\{1, ..., r\}$ such that $\forall\ 1 \le i \le r : C_i = D_{\varphi(i)}$

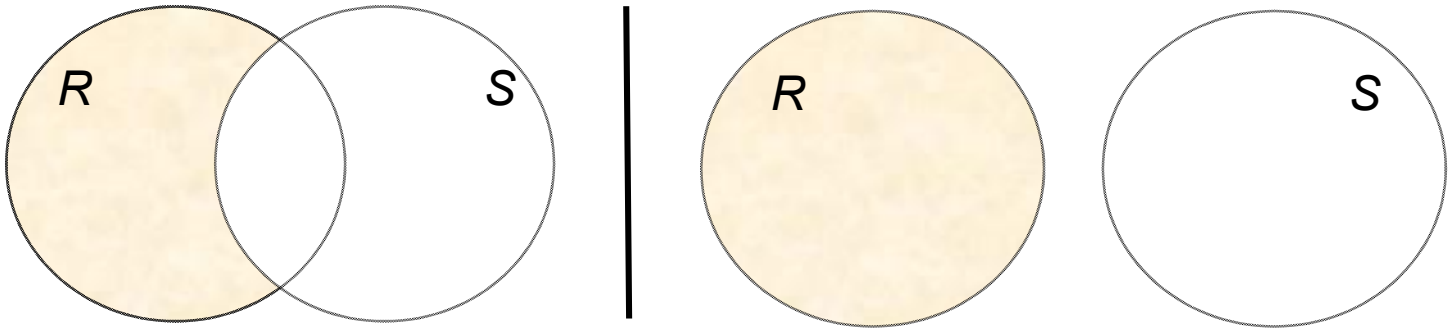
Union Operation

- At the schema level
  - Precondition is that R and S are schema compliant
  - Result schema is equal to R (or S), that is, there is no change
- At the data level
  - $R \cup S = \{t \mid t \in R \vee t \in S\}$
  - No duplicates in result due to set property
  - Let $|R|$ be the cardinality, that is, the number of tuples, of R; correspondingly $|S|$ is defined
  - Number of tuples of $R \cup S$ is $|R \cup S| = |R| + |S| - |R \cap S|$



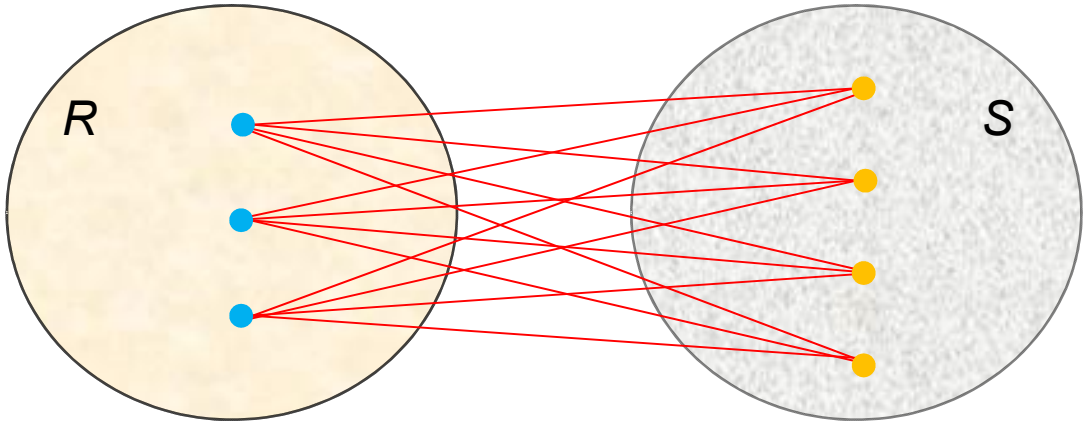
Difference Operation

- At the schema level
  - Precondition is that R and S are schema compliant
  - Result schema is equal to R (or S), that is, there is no change
- At the data level
  - $R - S = \{t \mid t \in R \wedge t \notin S\}$
  - No duplicates in result due to set property
  - Number of tuples of $R - S$ is $|R - S| = |R| - |R \cap S|$



Cartesian Product Operation

- At the schema level: Result relation schema T is the concatenation of the schemas (attribute lists) of R and S. That is, we get $T(A_1 : C_1, A_2 : C_2, ..., A_r : C_r, B_1 : D_1, B_2 : D_2, ..., B_s : D_s)$ with arity $r + s$.
- At the data level:
  - Tuple concatenation of two tuples $t = (v_1, ..., v_r) \in R$ and $u = (w_1, ..., w_s) \in S$ is defined as the tuple $t \circ u = (v_1, ..., v_r, w_1, ..., w_s) \in R \times S$ of arity $r + s$
  - $R \times S = \{t \circ u \mid t \in R, u \in S\}$
  - Number of tuples of $R \times S$ is $|R \times S| = |R| \cdot |S|$



Project Operation (I)

- Example: “Determine the room numbers of all professors”
- This is a query in *colloquial* language
- Two graphical illustrations of the desired result

professors			
pers-id	name	rank	room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	052
2134	Augustinus	C3	309
2136	Curie	C4	036
2137	Kant	C4	007

Choose columns of interest (“projection”)

$\pi_{room}(professors)$

professors			
pers-id	name	rank	room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	052
2134	Augustinus	C3	309
2136	Curie	C4	036
2137	Kant	C4	007

Delete columns of non-interest

Project Operation (II)

- At the schema level: Given $R(A_1 : C_1, ..., A_r : C_r)$, project R to $R'(A_{i_1}, \ldots, A_{i_n})$ where $A_{i_1}, \ldots, A_{i_n} \in \{A_1, ..., A_r\}$
- At the data level:
  - Let $t = (v_1, ..., v_r) \in R$ with $v_i \in C_i$ for $1 \leq i \leq r$ be a tuple. Then we define $t|_{A_{i_1}, ..., A_{i_n}} = (v_{i_1}, \ldots, v_{i_n})$ with $v_{i_j} \in C_{i_j}$ as the projection (restriction) of t to the attribute values of $A_{i_1}, \ldots, A_{i_n}$.
  - The projection π on R with respect to the attributes $A_{i_1}, \ldots, A_{i_n}$ is defined as $\pi_{A_{i_1}, ..., A_{i_n}}(R) = \{t|_{A_{i_1}, ..., A_{i_n}} \mid t = (v_1, ..., v_r) \in R\}$
- Elimination of duplicates due to the *set property* of relations
- Two tuples $s = (v_1, ..., v_n)$ and $t = (w_1, ..., w_m)$ are *equal* if, and only if, the following conditions hold:
  1. Both tuples have the same number of values, that is, $n = m$
  2. Corresponding values of both tuples have the same domain or type, that is, $\forall\, 1 \leq i \leq n : dom(v_i) = dom(w_i)$
  3. Both tuples are component-wise equal, that is, $\forall\, 1 \leq i \leq n : v_i = w_i$

Project Operation (III)

- Number of tuples of $\pi_{A_{i_1}, \ldots, A_{i_n}}(R)$ is $1 \leq |\pi_{A_{i_1}, \ldots, A_{i_n}}(R)| \leq |R|$
  - Number is equal to 1 if the projections of all tuples of R lead to the same projected tuple (requires that the key attributes do not belong to the projection attributes)
  - Number is equal to $|R|$ if the projections of all tuples of R lead to different tuples (this happens, for example, if the key attributes belong to the projection attributes)

- Projection operation implements a reduction of columns (“vertical reduction”)

Select Operation (I)

- Example: “Determine all professors that have the rank C4”
- Two graphical illustrations of the desired result

professors			
pers-id	name	rank	room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	052
2134	Augustinus	C3	309
2136	Curie	C4	036
2137	Kant	C4	007

professors			
pers-id	name	rank	room
2125	Sokrates	C4	226
2126	Russel	C4	232
~~2127~~	~~Kopernikus~~	~~C3~~	~~310~~
~~2133~~	~~Popper~~	~~C3~~	~~052~~
~~2134~~	~~Augustinus~~	~~C3~~	~~309~~
2136	Curie	C4	036
2137	Kant	C4	007

Choose rows of interest (“selection”)

$\sigma_{rank=\text{“C4”}}(professors)$

Delete rows of non-interest

Select Operation (II)

- At the schema level: New relation schema = old relation schema
- At the data level:
  - Selection of all tuples of a relation R that fulfil a given predicate or condition (that is, Boolean function) F
  - F is a *quantifier-free Boolean expression* (that is, no $\forall$ symbol and no $\exists$ symbol contained) which for a tuple $t \in R$ checks if F is fulfilled for the argument t, that is, if $F(t)$ yields the Boolean value *true*
  - Formula F is composed of
    - operands: constants or names of attributes
    - comparison operators: $=, \neq, <, \leq, >, \geq$
    - logical operators: $\wedge, \vee, \neg$
  - A selection σ with respect to a predicate F is defined as

    $\sigma_F(R) = \{t \in R \mid F(t)\}$

Select Operation (III)

- Number of tuples of $\sigma_F(R)$ is $0 \leq |\ \sigma_F(R)\ | \leq |R|$
  - Number is equal to 0 if no tuple of *R* fulfills the predicate *F*, that is, we obtain the empty table
  - Number is equal to $|R|$ if all tuples of *R* fulfill the predicate *F*, that is, $\sigma_F(R) = R$
- Selection operation implements a reduction of rows (“horizontal reduction”)