

Transformation of an n -ary relationship set

- 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 university database:
tests(reg-id, id, pers-id, grade)

Complete schema of the university database

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 : *string*, boss : *integer*)

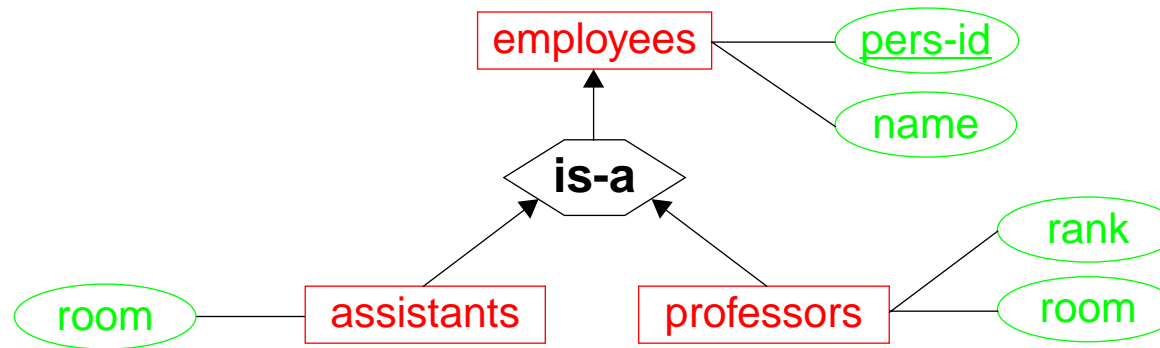
attends(reg-id : *integer*, id : *integer*)

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

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

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 key of the specialized subclasses.
- ❑ example:



employees(pers-id, name)

professors(pers-id, room, rank)

assistants(pers-id, room)

- ❑ information about a professor 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 (join). There is no inheritance in the relational data model.

3.4 Relational Algebra

Introduction

- so far: structural description by means of a database schema
- required: language for extracting information from the database
(likewise necessary, but later dealt with: **data manipulation language (DML)** with operations for inserting, changing and deleting information)
- two formal languages
 - **relational calculus** (tuple relational calculus, domain relational calculus): **declarative** language which allows to specify *which* data one would like to retrieve or which criteria these data have to fulfil, but not *how* a query has to be evaluated
 - **relational algebra: procedural** language which allows to specify *how* a query has to be evaluated (execution plan)
 - both languages are closed, i.e., the results of queries, which operate on relations, are again relations.
- universal algebra
 - given a set T („anchor of the algebra“)
 - given a set of operations $\{\sigma_1, \dots, \sigma_n\}$ of the form $\sigma_i: T^k \rightarrow T$

- ❑ relational algebra is a universal algebra
 - anchor is the set of all relations
 - 5 (+ 1) basic operations: **union**, **difference**, **Cartesian product**, **project**, **select**, (**rename**)
 - other algebra operations are derived, i.e., they can be expressed by the basic operations
- ❑ given: two relations $R(A_1 : C_1, A_2 : C_2, \dots, A_r : C_r)$ and $S(B_1 : D_1, B_2 : D_2, \dots, B_s : D_s)$ with arity r and s
- ❑ R and S are **schema compliant** (identical except for renaming), if $r = s$ holds and if there exists a permutation φ of the indices $\{1, \dots, r\}$, so that $\forall 1 \leq i \leq r : C_i = D_{\varphi(i)}$

The Union operation

- ❑ R and S are schema compliant
- ❑ $R \cup S = \{t \mid t \in R \vee t \in S\}$

The Difference operation

- ❑ R and S are schema compliant
- ❑ $R - S = \{t \mid t \in R \wedge t \notin S\}$

The Cartesian product operation

- ❑ result relation T has the attributes $\{A_1 : C_1, A_2 : C_2, \dots, A_r : C_r, B_1 : D_1, B_2 : D_2, \dots, B_s : D_s\}$ as the union of the attributes of R and S , T has arity $r + s$
- ❑ *tuple concatenation* of two tuples $t = (v_1, \dots, v_r)$ and $u = (w_1, \dots, w_s)$ is defined as the tuple $t \circ u = (v_1, \dots, v_r, w_1, \dots, w_s)$ of arity $r + s$.
- ❑ $R \times S = \{t \circ u \mid t \in R, u \in S\}$

The Project operation

- ❑ restriction of a relation schema to specified attributes
- ❑ correspondingly restriction of the tuples of the respective relation to these attributes
- ❑ elimination of possibly emerging duplicates (set property!)
- ❑ let $f: X \rightarrow Y$ be a function, and let $Z \subseteq X$
function restriction: $f|_Z = \{f(x) \mid x \in Z\}$
- ❑ Let R be a relation over the attribute set A , and let $B \subseteq A$. Using the alternative definition of a relation we obtain for the projection π :

$$\pi_B(R) = \{t|_B \mid t \in R\}$$

The **Select** operation

- ❑ new relation schema = old relation schema
- ❑ selection of all tuples of a relation R that fulfil a given predicate F
- ❑ F quantifier-free boolean expression which for a tuple $t \in R$ checks if F satisfies the condition for the argument t , i.e., 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 condition F is defined as
$$\sigma_F(R) = \{t \in R \mid F(t)\}$$

Renaming of relations and attributes („+1-operation“)

- ❑ sometimes necessary to use the same relation or the same attribute several times in a query
 - renaming of a relation or an attribute required

- ❑ $\rho_S(R)$: relation R is renamed into relation S
- ❑ $\rho_{B \leftarrow A}(R)$: attribute A of relation R is renamed into B
- ❑ The renaming operator differs from the other operators, since no new instance is created, but only the relation schema is changed.

Example queries based on the university database

- ❑ Which students have enrolled for more than 10 semesters?

$\sigma_{\text{sem} > 10}(\text{students})$

$\sigma_{\text{sem} > 10}(\text{students})$		
reg-id	name	sem
24002	Xenokrates	18
25403	Jonas	12

Example instance of our university database

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	Theophastrós	02
29555	Feuerbach	02

Example instance of our university database (*cont.*)

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

Example instance of our university database (*cont.*)

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

Example queries based on the university database (cont.)

- Which ranks do the professors have?

$\pi_{\text{rank}}(\text{professors})$

$\pi_{\text{rank}}(\text{professors})$
rank
C4
C3

- Output the personell ids and names of all professors and assistants.

$\pi_{\text{pers-id, name}}(\text{professors}) \cup \pi_{\text{pers-id, name}}(\text{assistants}) [= R]$

R	
pers-id	name
2125	Sokrates
2126	Russel
2127	Kopernikus

2133	Popper
2134	Augustinus
2136	Curie
2137	Kant
3002	Platon

3003	Aristoteles
3004	Wittgenstein
3005	Rhetikus
3006	Newton
3007	Spinoza

- Determine all students who so far have not passed any exam

$$\pi_{\text{reg-id}}(\text{students}) - \pi_{\text{reg-id}}(\text{tests}) [= R]$$

R
reg-id
24002
26120

26830
29120
29555

- professors × attends

professors × attends					
pers-id	name	rank	room	reg-id	id
2125	Sokrates	C4	226	26120	5001
...
2137	Kant	C4	007	29555	5001

- What are the predecessors of the predecessors of lecture 5216?

$$\pi_{V1.predecessor}(\sigma_{V2.successor=5216 \wedge V1.successor=V2.predecessor}(\rho_{V1}(is_precondition_of) \times \rho_{V2}(is_precondition_of)))$$

result: tuple with value 5001

Derived operators:

The Intersection operation

- R and S are schema compliant
- $R \cap S = R - (R - S)$

The Symmetrical Difference operation

- R and S are schema compliant
- $R \Delta S = (R - S) \cup (S - R)$
- union of those tuples of R and S that are only contained in one of both relations

The Quotient (Division) operation

- ❑ used in queries that contain an all-quantification
- ❑ $R(A_1, \dots, A_r)$ schema of a relation R , $S(B_1, \dots, B_s)$ schema of a relation S , $S \subseteq R$
 → result relation has schema $R - S$
- ❑ $R \div S = \pi_{R-S}(R) - \pi_{R-S}((\pi_{R-S}(R) \times S) - R)$
- ❑ definition using the following simplified assumptions: $r > s$, $S \neq \emptyset$, $A_r = B_s$, $A_{r-1} = B_{s-1}$,
 $A_{r-s+1} = B_1$
 → result relation has schema $\{A_1, \dots, A_{r-s}\}$
- ❑ result instance of the quotient:
 $R \div S = \{(a_1, \dots, a_{r-s}) \mid \forall (b_1, \dots, b_s) \in S : (a_1, \dots, a_{r-s}, b_1, \dots, b_s) \in R\}$

- ❑ example:

R	A	B	C	D
	a	b	c	d
	a	b	e	f
	b	c	e	f
	e	d	c	d
	e	d	e	f
	a	b	d	e

S	C	D
	c	d
	e	f

$R \div S$	A	B
	a	b
	e	d

The Natural Join operation

- assumption:
 - R has $m + k$ attributes $A_1, \dots, A_m, B_1, \dots, B_k$, and S has $n + k$ attributes $B_1, \dots, B_k, C_1, \dots, C_n$
 - $\Rightarrow R \bowtie S$ has the arity $m + n + k$
 - domains of the B_j are equal in R and in S
 - $\forall 1 \leq i \leq m \forall 1 \leq j \leq n : A_i \neq C_j$
- $R \bowtie S = \pi_{A_1, \dots, A_m, R.B_1, \dots, R.B_k, C_1, \dots, C_n}(\sigma_{R.B_1=S.B_1 \wedge \dots \wedge R.B_k=S.B_k}(R \times S))$
- join operator is associative and commutative

The Theta Join operation

- given: $R(A_1, \dots, A_r), S(B_1, \dots, B_s)$
- result schema has the $r + s$ attributes $A_1, \dots, A_r, B_1, \dots, B_s$
- $R \bowtie_F S = \sigma_F(R \times S)$
- a theta join of the form $R \bowtie_{R.A_i = S.B_j} S$ is also denoted as **equi-join**