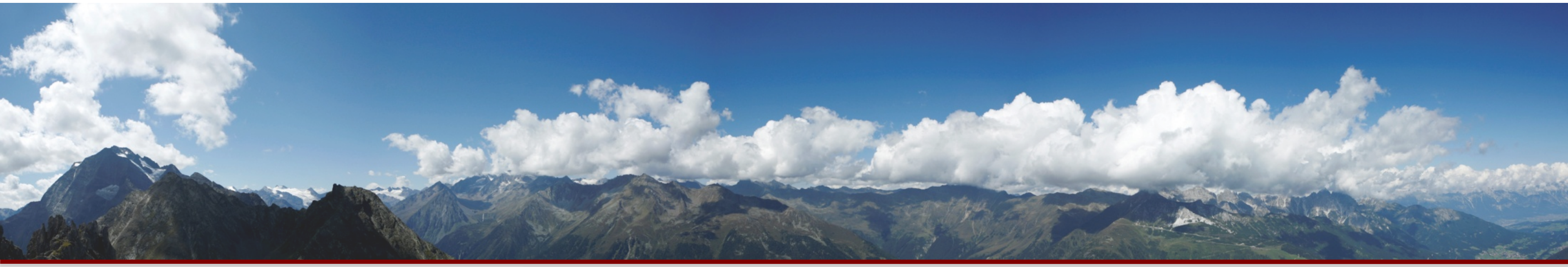
Introduction to modeling
WS 2015/16

Object-role modelling (ORM)

Slides for this part are based on Chapters 3-7 from Halpin, T. & Morgan, T. 2008, Information Modeling and Relational Databases, Second Edition (ISBN: 978-0-12-373568-3), published by Morgan Kaufmann Publishers.



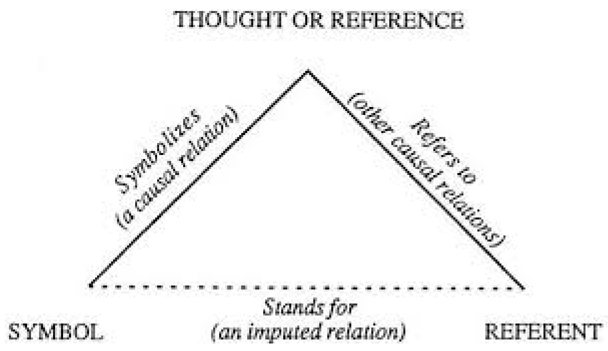
Where are we?

STI · INNSBRUCK

#	Title	Date
1	Introduction	07.10.2015
2	General concepts	21.10.2015
3	**ORM modeling**	**04.11.2015**
4	Relational modeling	18.11.2015
5	ER modeling	02.12.2015
6	OO modeling	16.12.2015
7	Services and process modeling	13.01.2016
10	Exam	27.01.2016

Object-role modelling (ORM)

- **Modeling Language/Notation**
- **Modeling Method**
- Three important principles:
  - Ogdens triangle: relationship between reality and the model
  - Natural language: the model must be expressed so that it can be understood by informed users
  - 100% principle: we can build an accurate enough model of reality



Conceptual Schema Design Procedure

Seven steps:

1. Transform familiar examples into elementary facts.
2. Draw the fact types, and apply a population check.
3. Check for entity types to be combined, and note any arithmetic derivations.
4. Add uniqueness constraints, and check the arity of fact types.
5. Add mandatory role constraints, and check for logical derivations.
6. Add value, set-comparison, and subtyping constraints.
7. Add other constraints and perform final checks.

Elementary facts

- A fact that can not be divided without losing meaning is called **elementary**
- Example:
  - Alice likes ice-cream and juice
- This fact is not elementary because it can be replaced by two elementary phrases
  - Alice likes ice-cream
  - As like juice

Which of these facts are elementary?

- John was born 11th of Oct 2012.
- John and Tom are brothers.
- John sings.
- John got 1 in Introduction To Modeling lecture last semester.
- Anna was hired as engineer on 01/01/2012.

Facts

- Consider the fact “John takes VO437244”
- There is information that is implied:
  - John is the name of a student
  - VO437244 is a course code
- What we mean is that

  student with name John

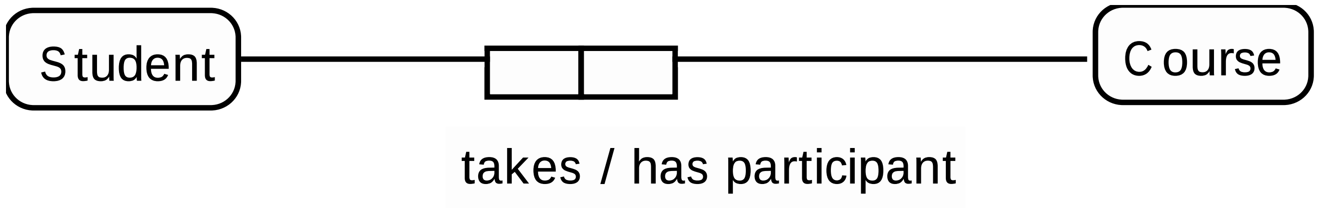
  takes the course with the course code VO437244

Facts (cont')

- Consider the fact:
  - "Student with name John takes the course with the course code VO437244"
- "student" and "course" are **concepts**
- "name" and "course code" are their **representations**
- "John" and "VO437244" are **instances** (data)

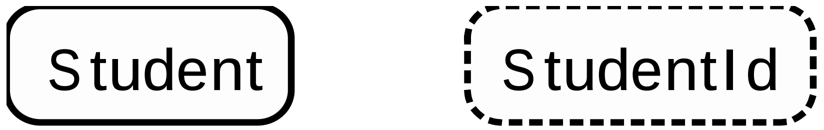
Facts and fact types

- The two sentences below have the same meaning:
  - "The student with name John takes the course with the course code VO437244"
  - "The course with course code VO437244 has participant the student with name John"
- Similar facts can be created by replacing instances:
  "The student with the name Anna takes the course with the course code VO437211"
- In ORM notation, fact types are represented as follows:



Concepts and representations

- Concepts and representations are depicted as solid and resp dashed rectangles with rounded corners
  - e.g. the concept Student and representation StudentId is depicted as follows:



- In ORM, the word **object types** is used as a generic term for concepts and representations

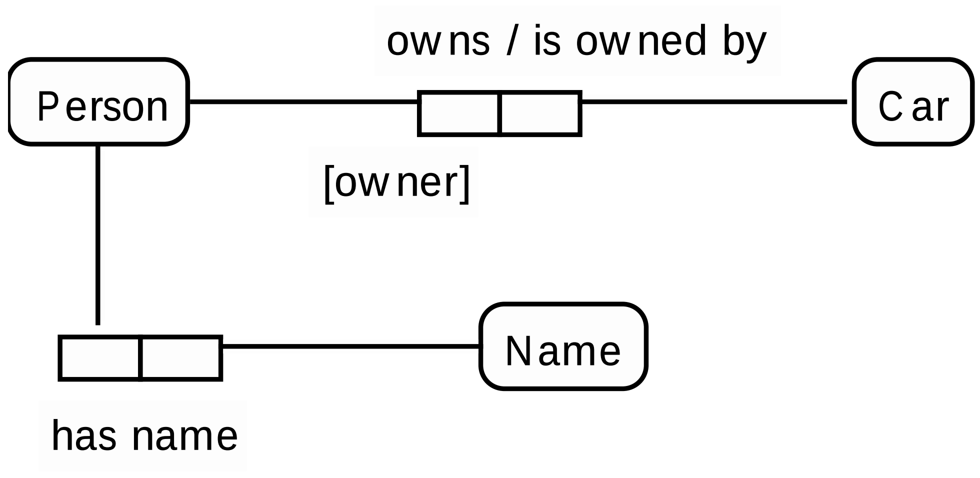
Arity

- Number of concepts in a fact is called fact **arity**
- Elementary sentence

  'Student with name John got on the course with the course code VO437244 grade 1“
  - contains three concepts: ”Student ",“Course“ and “Grade“ => the fact has arity 3
- Facts with arity 1 are called **unary**
- Facts with arity 2 are called **binary**
- Facts with arity 3 are called **ternary**
- One can construct facts with arbitrarily high arities – **n-aries**

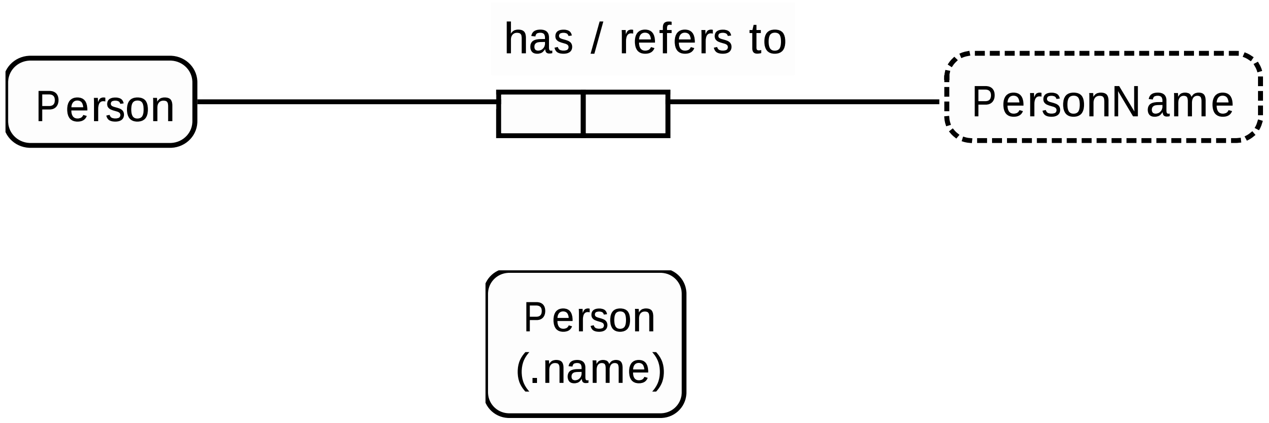
Roles and fact types

- *Roles* are relationship parts played by objects – depicted as *boxes*
  - Optionally, roles may be given a name, in which case the *role name* is displayed in square brackets next to the role box, e.g.



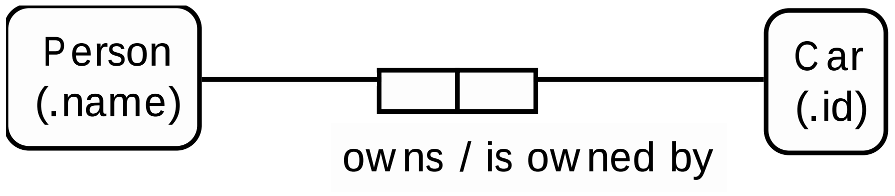
Reference Types

- Elementary facts are relationships between entities
- References are relationships between entities and values
  - Provide the bridge between the world of entities and the world of values



Fact tables

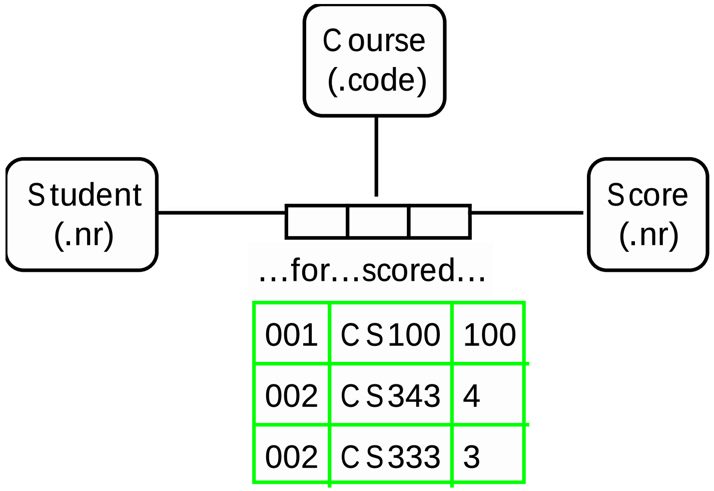
- A fact table is a table for displaying instances of a fact type



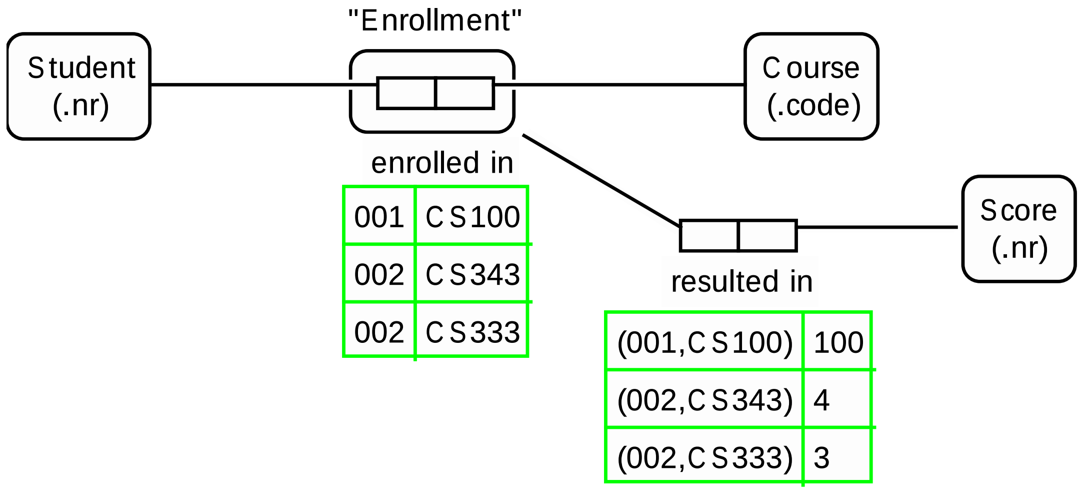
John	DI323
Ann	IN423
Mike	AN343

Objectification

- Treats a relationship between objects as an object itself



001	CS100	100
002	CS343	4
002	CS333	3

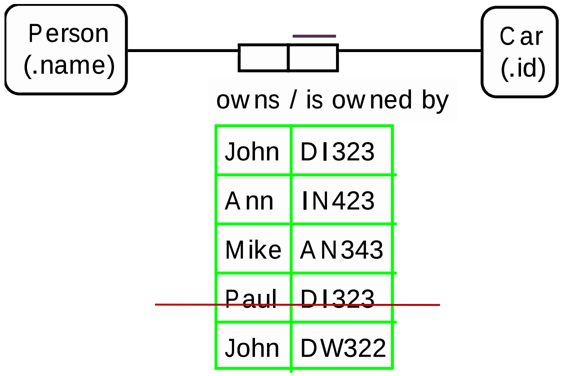


001	CS100
002	CS343
002	CS333

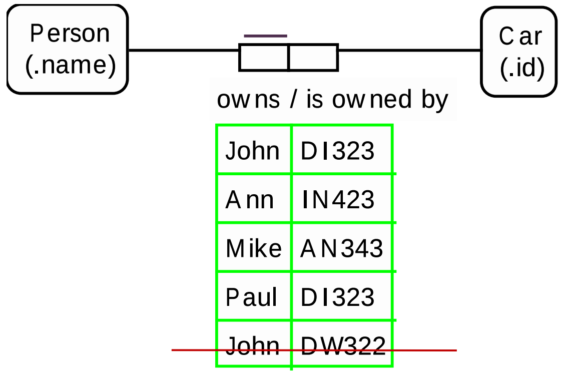
(001,CS100)	100
(002,CS343)	4
(002,CS333)	3

Uniqueness constraints

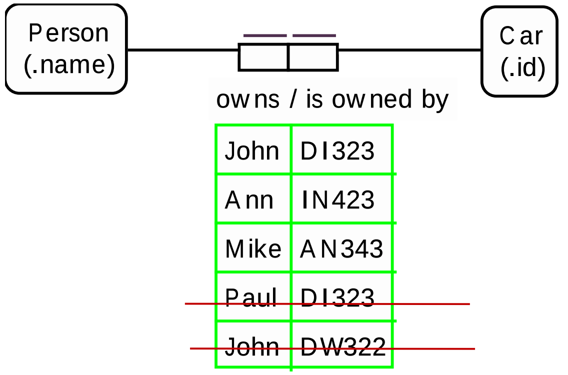
- Ensure entries in fact tables are unique
- A single role with a uniqueness *bar* means that entries in the associated column must be unique in that column (i.e., no duplicates are allowed in that column)



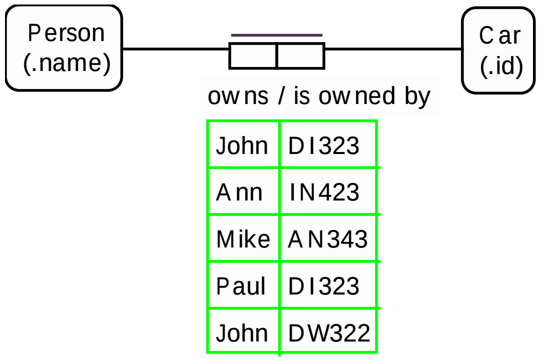
John	DI323
Ann	IN423
Mike	AN343
~~Paul~~	~~DI323~~
John	DW322



John	DI323
Ann	IN423
Mike	AN343
Paul	DI323
~~John~~	~~DW322~~

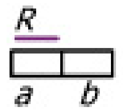


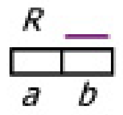
John	DI323
Ann	IN423
Mike	AN343
~~Paul~~	~~DI323~~
~~John~~	~~DW322~~

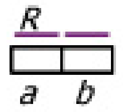


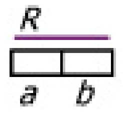
John	DI323
Ann	IN423
Mike	AN343
Paul	DI323
John	DW322

The four uniqueness constraint patterns for a binary

(a)  No duplicates are allowed in *a*'s column. Each *a* *R*'s at most one *b*.

(b)  No duplicates are allowed in *b*'s column. Each *b is R'd* by at most one *a*.

(c)  Both the foregoing constraints apply.

(d)  No duplicate (*a*, *b*) rows are allowed. Each *a* may *R* many *b* and vice versa.

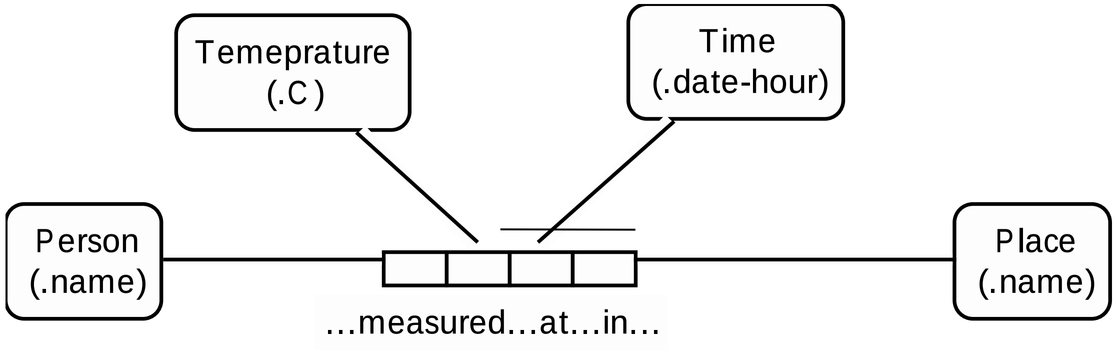
Uniqueness Constraints on Longer Fact Types

- Example: “John measured 10C at 10am in Innsbruck”
- Three main questions one always must ask:
  - What **concepts** do we have?
  - What **representation** do we use for the concepts?
  - Is the fact elementary in our UoD, and if it is not, how can it be split into elementary facts?

Uniqueness Constraints on Longer Fact Types (cont')

- Example: "John measured 10C at 11am in Innsbruck"
  - John is a person with representation person name
  - 10 is a temperature with representation degrees C
  - 11am is a time point with representation of time and date
  - Innsbruck is a is a place with representation place name
  - To determine whether the sentence is elementary, we use fact tables

Uniqueness Constraints on Longer Fact Types (cont')



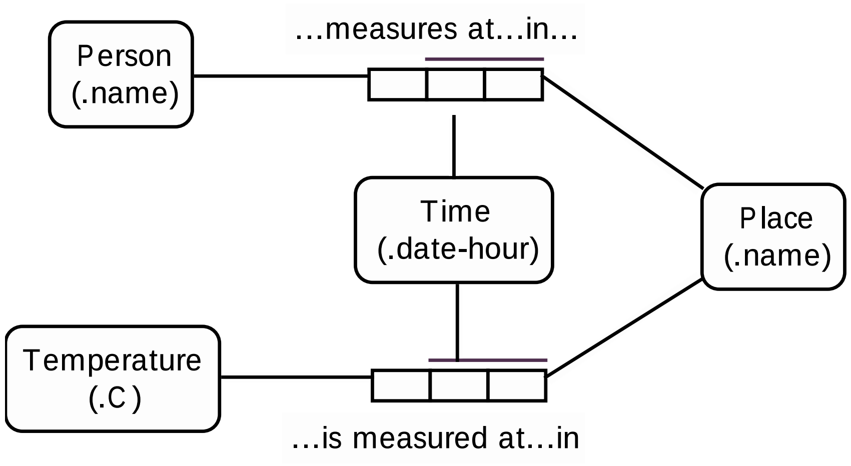
John	10	10.10 11am	Innsbruck
Alice	10	10.10 12pm	Vienna
~~Paul~~	~~4~~	~~10.10 11am~~	~~Innsbruck~~
John	5	11.10 11am	Innsbruck

- Which table entry needs to be removed?
- Where should the uniqueness constraint be placed?

“n-1 rule”

- A fact with arity n is never elementary if the corresponding fact table has a uniqueness constraint which is shorter than n-1
- If the shortest uniqueness constraint has length n-1, the fact is elementary
- Facts with arity 1 or 2 are always elementary

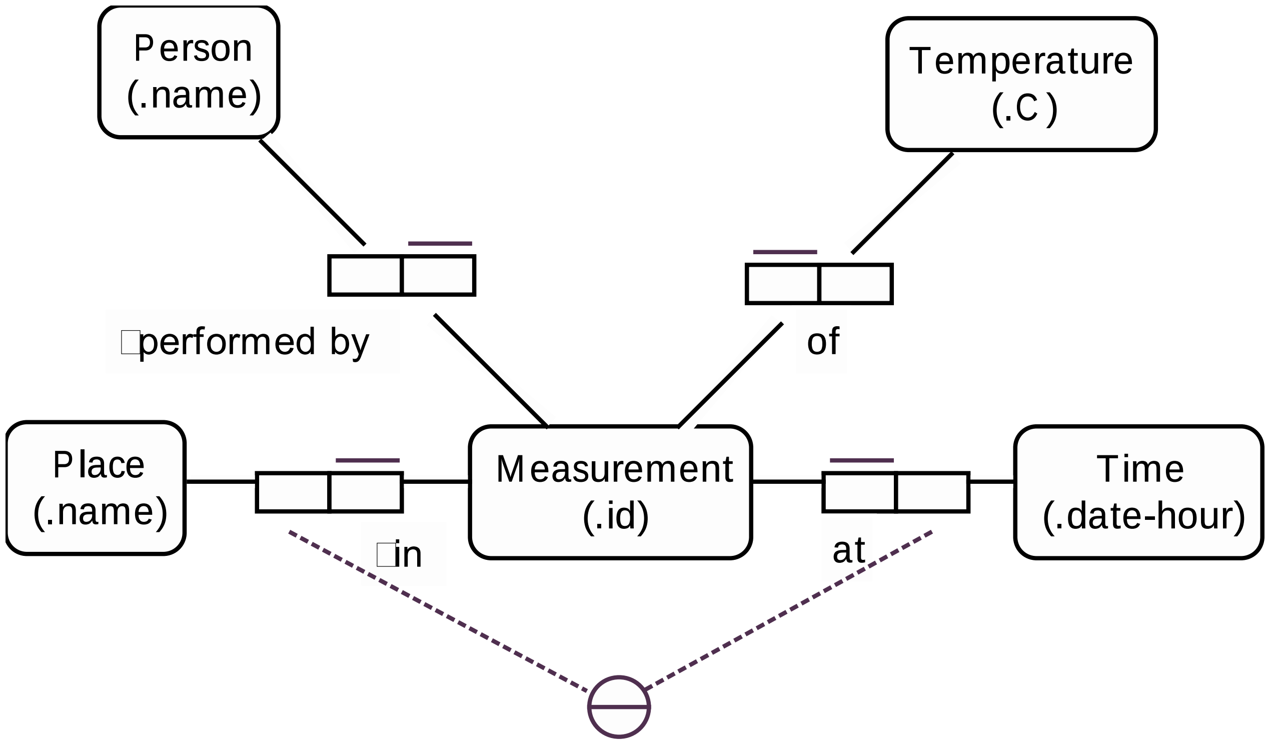
Uniqueness Constraints on Longer Fact Types (cont')



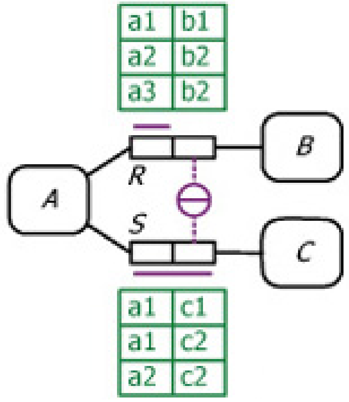
- The two facts have in common that Time and Place together determine a third value
- But does not imply that both temperature and Person instances are determined by the same combination of Time and Place

External Uniqueness Constraints

- Apply to roles from different predicates



Each *bc* refers to at most one *a*



a1	b1
a2	b2
a3	b2

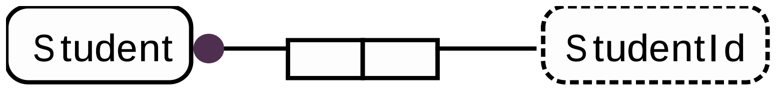
a1	c1
a1	c2
a2	c2

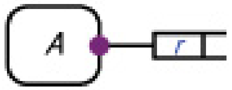
Populations of roles/concepts

- Population of roles
  pop(r) = population of role r
  = set of objects referenced in the column for *r*
- Role populations are used to determine *concept populations*
- Population of a concept A which plays roles r_1, r_2, ..., r_n is:
  $pop(A) = pop(r_1) \cup pop(r_2) \cup \ldots \cup pop(r_n)$
- Note: The population of a role / concept varies with time

Mandatory and Optional Roles

- A role is **mandatory** if and only if, for all states of the database, the role must be played by every member of the population of its object type; otherwise the role is **optional**
  - A mandatory role is also called a **total role**, since it is played by the total population of its object type
- To indicate explicitly that a role is mandatory, a mandatory role dot is added to the line that connects the role to its object type



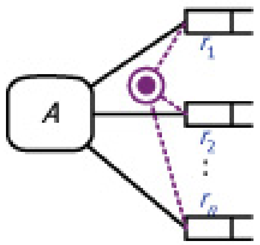


Role *r* is **mandatory** (for the population of *A*).

Each instance of type *A* that is recorded in the database is also recorded to play *r*.

pop(*r*) = pop(*A*)

Inclusive-or constraint



Inclusive-or (Disjunctive mandatory role) constraint

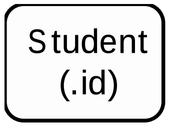
The inclusive disjunction of roles r_1 .. r_n is mandatory for *A*

i.e., each member of pop(*A*) plays r_1 **or** ... r_n (or all)

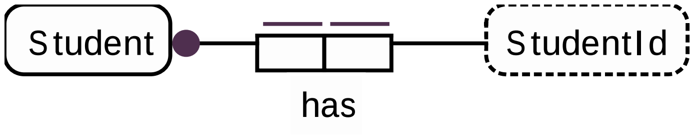
i.e., each member of pop(*A*) plays *at least one of* r_1, ..., r_n

Reference Schemes

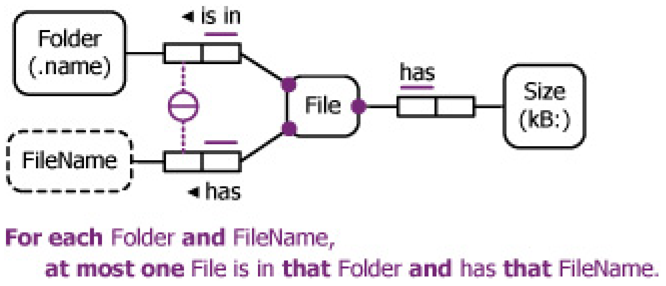
- Reference schemes are abbreviated as reference modes in parentheses.



- Reference schemes depicted explicitly—each appears as a mandatory, 1:1 reference type.

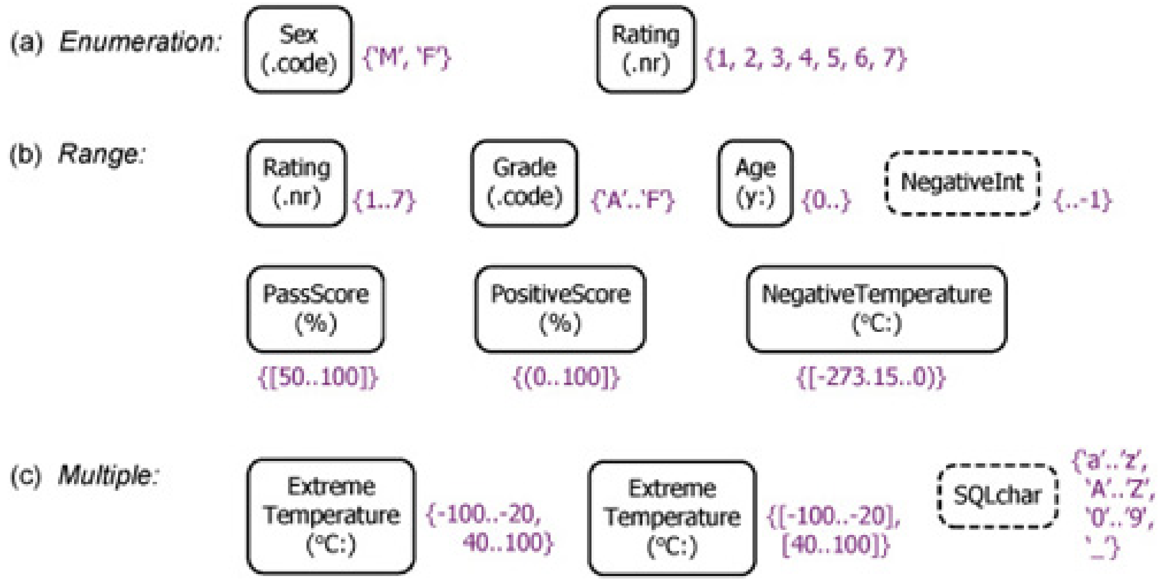


- Compound reference scheme


Folder
(.name)
◂is in
FileName
◂has
File
has
Size
(kB:)
For each Folder and FileName,
at most one File is in that Folder and has that FileName.

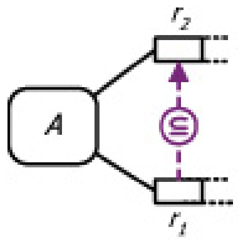
Value Constraints

- Limit the possible instances of a concept
- In practice: Specify a range of values that are legal representations - eg. by direct enumeration of values, specifying a lower and / or upper limit, etc



Material on this slide based on *Ch 6.3* in Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases*, Second Edition

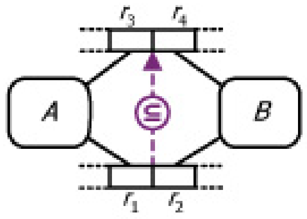
Subset, Equality, and Exclusion Constraints



For each database state:

$\text{pop}(r_1) \subseteq \text{pop}(r_2)$

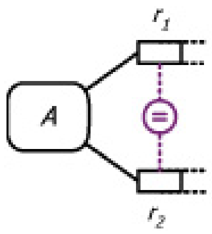
If *a* plays r_1 **then** *a* plays r_2



For each database state:

$\text{pop}(r_1, r_2) \subseteq \text{pop}(r_3, r_4)$

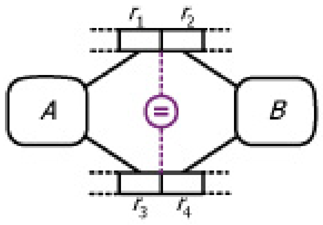
Each *ab* pair in $\text{pop}(r_1, r_2)$
is also in $\text{pop}(r_3, r_4)$



For each database state:

$\text{pop}(r_1) = \text{pop}(r_2)$

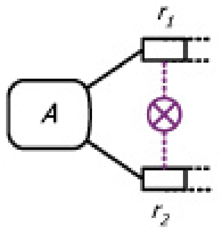
a plays r_1 **iff** *a* plays r_2



For each database state:

$\text{pop}(r_3, r_4) = \text{pop}(r_1, r_2)$

Each *ab* pair in $\text{pop}(r_3, r_4)$
is also in $\text{pop}(r_1, r_2)$
and vice versa



For each database state:

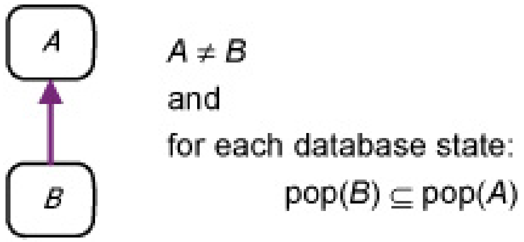
$\text{pop}(r_1) \cap \text{pop}(r_2) = \{\ \}$

No *a* plays **both** r_1 **and** r_2

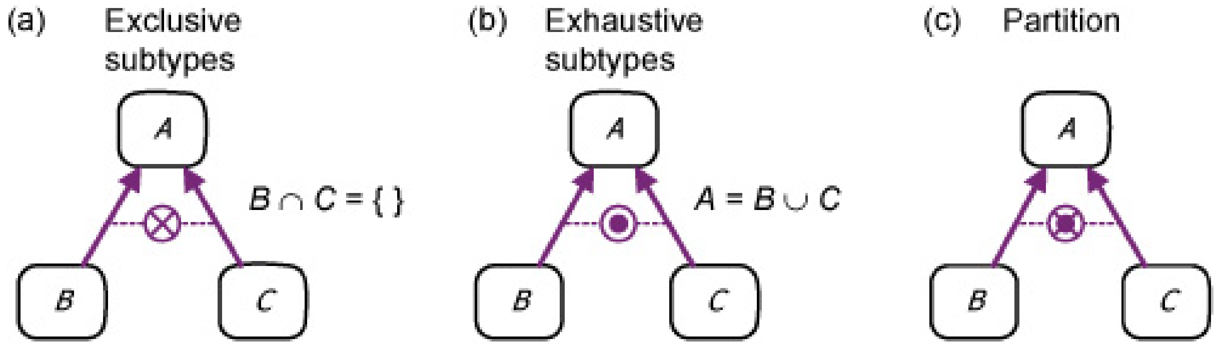
Material on this slide based on *Ch 6.4* in Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases*, Second Edition

Subtyping

- *B* is a proper subtype of A

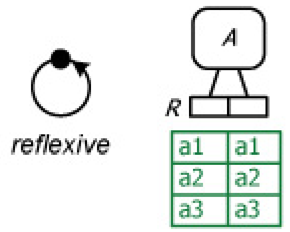


- *B* and *C* are mutually exclusive (a), exhaustive (b), or both (c)

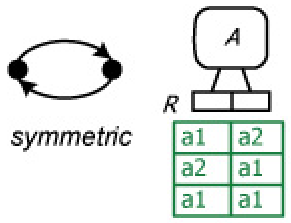


Material on this slide based on *Ch 6.5* in Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases*, Second Edition

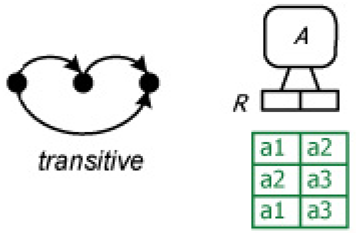
Ring constraints



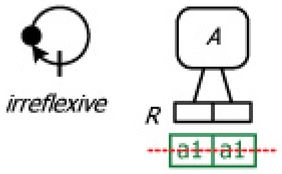
R is reflexive over its populations **iff**
for each *x* that plays a role of *R*, *xRx*



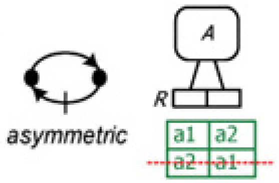
R is symmetric **iff**
for all *x*, *y*: *xRy* → *yRx*



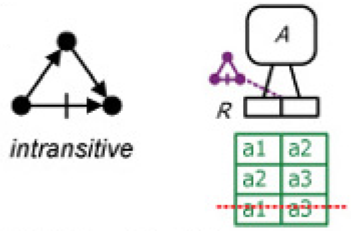
R is transitive **iff**
for all *x*, *y*: (*xRy* & *yRz*) → *xRz*



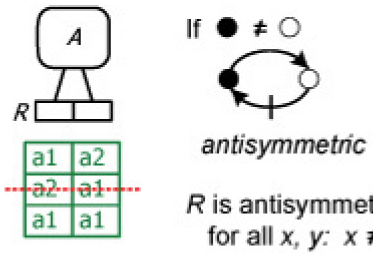
R is irreflexive **iff**
for all *x*, ~*xRx*



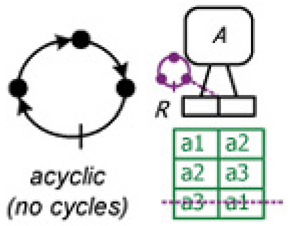
R is asymmetric **iff**
for all *x*, *y*: *xRy* → ~*yRx*



R is intransitive **iff**
for all *x*, *y*, *z*: *xRy* & *yRz* → ~*xRz*



R is antisymmetric **iff**
for all *x*, *y*: *x* ≠ *y* & *xRy* → ~*yRx*



Material on this slide based on *Ch 7.3* in Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases*, Second Edition

Implementing a Conceptual Schema

- Design the conceptual schema
- Annotate the conceptual schema with mapping choices as needed
- Map the design to a logical schema (e.g., relational or object-relational)
- Finesse the logical schema as needed (e.g., rename or reorder some columns)
- Generate the physical schema (e.g., in Microsoft SQL Server or IBM DB2)
- Create external schema(s) (e.g., forms, reports)
- Enforce security levels as needed
- Populate the database
- Issue queries and updates
- Update the schemas as needed

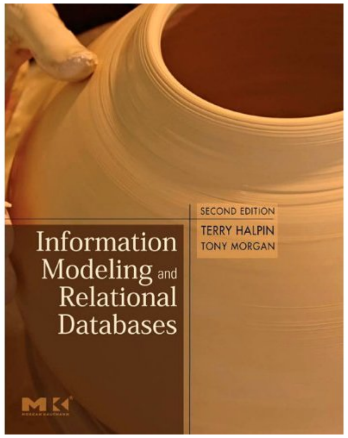
Material on this slide based on *Ch 11.1* in Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases*, Second Edition

Resources

- Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases, Second Edition* (ISBN: 978-0-12-373568-3), published by Morgan Kaufmann Publishers, an imprint of Elsevier.

- http://www.ormfoundation.org/
- http://www.orm.net/

- Tools: http://www.ormfoundation.org/files/10/default.aspx
  - ORM Lite: http://www.ormfoundation.org/files/folders/orm_lite/default.aspx



Next lecture

STI · INNSBRUCK

#	Title	Date
1	Introduction	07.10.2015
2	General concepts	21.10.2015
3	ORM modeling	04.11.2015
4	**Relational modeling**	**18.11.2015**
5	ER modeling	02.12.2015
6	OO modeling	16.12.2015
7	Services and process modeling	13.01.2016
10	Exam	27.01.2016

Questions?

STI · INNSBRUCK