Data and Process Modelling

3. Object-Role Modeling - CSDP Step 5

Marco Montali

KRDB Research Centre for Knowledge and Data
Faculty of Computer Science
Free University of Bozen-Bolzano

A.Y. 2014/2015

Mandatory Role Constraints

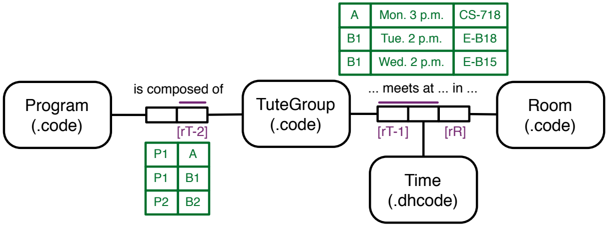
CSDP Step 5

Add mandatory role constraints, and check for logical derivations.

- Elicitation of mandatory role constraints (“at least one”).
  - Deep analysis of reference schemes.
  - Deep analysis of composite references.
- Check the obtained conceptual schema to find logical derivations.

Population vs Valuation

- Given an information base B and an object type T: population of T is the set of objects (instances) of T in B.
- The same definition applies to the population of a role: set of objects **referenced by** values in the column for that role.

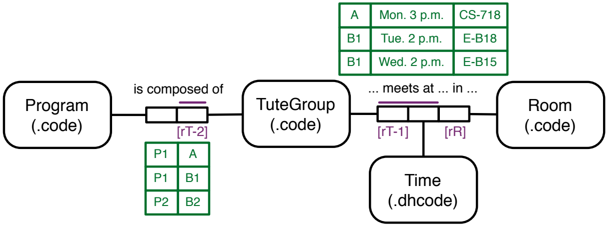


P1	A
P1	B1
P2	B2

A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

Population vs Valuation

- Given an information base B and an object type T: population of T is the set of objects (instances) of T in B.
- The same definition applies to the population of a role: set of objects **referenced by** values in the column for that role.



A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

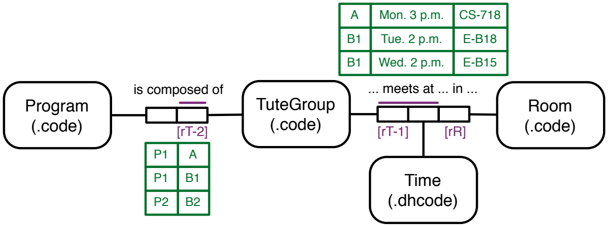
P1	A
P1	B1
P2	B2

- val(rT-1) = {'A','B1'}
- pop(rT-1) = {TuteGroup(.code) 'A', TuteGroup(.code) 'B1'}

Entity Type Population and Roles

- In general, each entity is stored in an information base because it plays at least one (referential) role (i.e., role in an elementary fact type).

Population of an entity type = union of the population of its roles



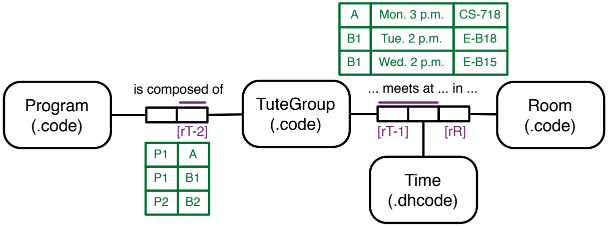
P1	A
P1	B1
P2	B2

A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

Entity Type Population and Roles

- In general, each entity is stored in an information base because it plays at least one (referential) role (i.e., role in an elementary fact type).

Population of an entity type = union of the population of its roles



A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

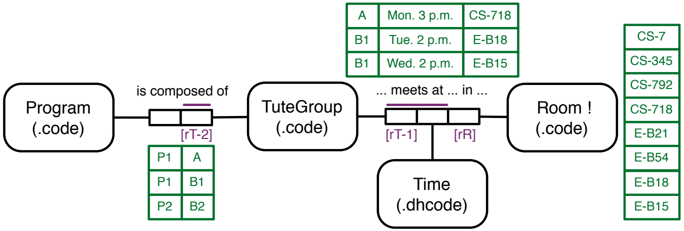
P1	A
P1	B1
P2	B2

- pop(TuteGroup) = pop(rT-1) $\bigcup$ pop(rT-2) = {A, B1} $\bigcup$ {A,B1,B2} = {A,B1,B2}

Independent Entity Type

Entity type whose possible fact roles are collectively optional.

- Intuition: the entity is important per sè.
- Graphically marked on the diagram with "!".
- Stored independently from its relationships with other entities.
- Can be associated to an object table (or reference table) listing its objects.



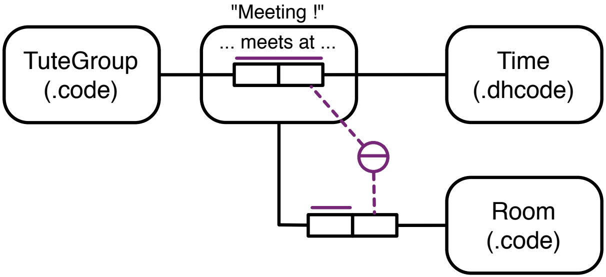
P1	A
P1	B1
P2	B2

A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

CS-7
CS-345
CS-792
CS-718
E-B21
E-B54
E-B18
E-B15

Objectified Association and Independence

- Objectified associations are often independent: we would like to first record the association (i.e., the corresponding reified entity) and then record other facts about it.

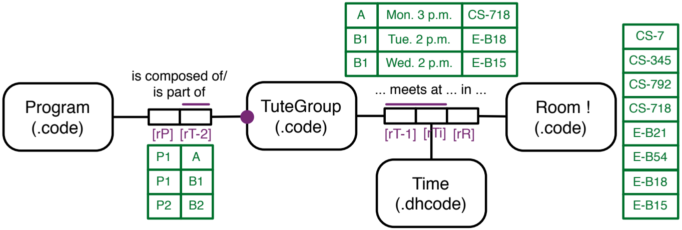


- First we record the Meeting, then where the Meeting is to be held.

Mandatory Role

For each information base, the role (r) is played by all members of the population of its attached entity type (A): $pop(r) = pop(A)$.

- Mandatory roles are represented adding a role dot to the connection line between the entity type and the role.



P1	A
P1	B1
P2	B2

A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

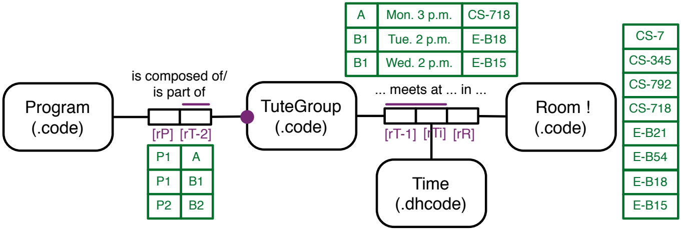
CS-7
CS-345
CS-792
CS-718
E-B21
E-B54
E-B18
E-B15

- Mandatory roles have an impact on the allowed updates!
  - A TuteGroup can be added only if its Program is already present.

Mandatory Role

For each information base, the role (r) is played by all members of the population of its attached entity type (A): $pop(r) = pop(A)$.

- Mandatory roles are represented adding a role dot to the connection line between the entity type and the role.



A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

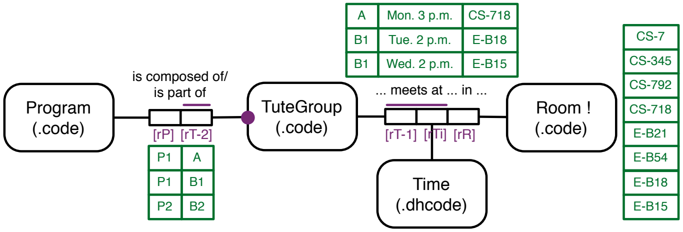
P1	A
P1	B1
P2	B2

CS-7
CS-345
CS-792
CS-718
E-B21
E-B54
E-B18
E-B15

- Mandatory roles have an impact on the allowed updates!
  - A TuteGroup can be added only if its Program is already present.
- If the role is not mandatory, it is called optional.

If two or more roles are played by the same object, they are optional unless explicitly marked mandatory.

Putting it All Together: Optional or Mandatory?



A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

P1	A
P1	B1
P2	B2

CS-7
CS-345
CS-792
CS-718
E-B21
E-B54
E-B18
E-B15

rT-2 is

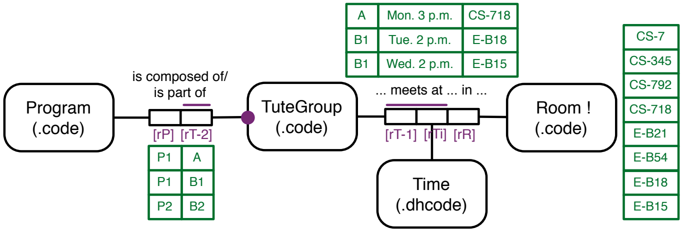
rT-1 is

rP is

rTi is

rR is

Putting it All Together: Optional or Mandatory?



rT-2 is mandatory (explicitly marked)

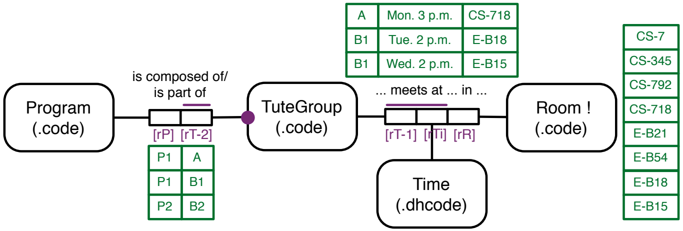
rT-1 is optional (two roles, rT-1 not marked as mandatory)

rP is mandatory (only one role)

rTi is mandatory (only one role)

rR is optional (independent entity type)

Putting it All Together: Optional or Mandatory?



rT-2 is mandatory (explicitly marked)

rT-1 is optional (two roles, rT-1 not marked as mandatory)

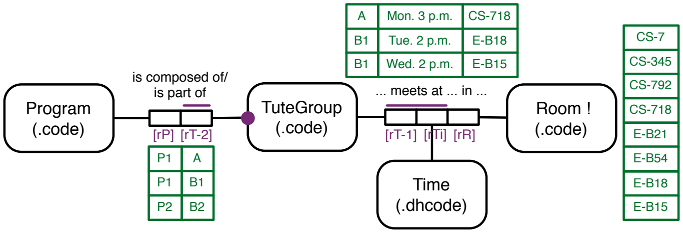
rP is mandatory (only one role)

rTi is mandatory (only one role)

rR is optional (independent entity type)

- These answers are only valid if the diagram is the definitive one!
- The really important mandatory roles are those marked explicitly!
- Don't mark the implied mandatory constraints, only those that deserve human attention!

Exactly one



P1	A
P1	B1
P2	B2

A	Mon. 3 p.m.	CS-718
B1	Tue. 2 p.m.	E-B18
B1	Wed. 2 p.m.	E-B15

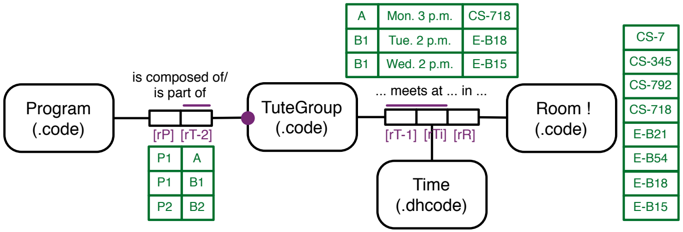
CS-7
CS-345
CS-792
CS-718
E-B21
E-B54
E-B18
E-B15

Verbalization of (some) constraints:

- **Each** TuteGroup is part of **at most one** Program.
- **Each** TuteGroup is part of **some** Program.

→ **Each** TuteGroup is part of **exactly one** Program.

Exactly one



Verbalization of (some) constraints:

- **Each** TuteGroup is part of **at most one** Program.
- **Each** TuteGroup is part of **some** Program.

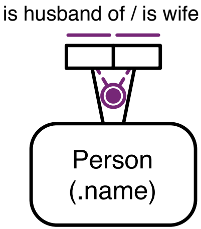
→ **Each** TuteGroup is part of **exactly one** Program.

- In general, 0..1 constraint + 1..* constraint → 1-1 constraint.
- 1-1 constraint is a bijection.

Disjunctive Mandatory Constraint

Mandatory constraint stating that the (inclusive) disjunction of two or more roles ($r_1, \ldots, r_n$) attached to an entity type (A) is mandatory: $pop(A) = \bigcup_{i \in \{1,\ldots,n\}} pop(r_i)$.

- Represented using a circle connected to the involved roles.
- We say that members of $pop(A)$ play r_1 or ... or r_n, i.e., that at least one of r_1, ..., r_n is played by each member of $pop(A)$.
- Example: supposing that the UoD is the one of "married people"...

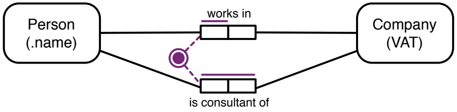


Explicit and Implicit Disjunctive Mandatory Constraints

- Remember: every non-independent entity type has a population = the population of all attached roles.
- Hence, there is an implicit disjunctive mandatory constraint spanning all such roles.
- This does not correspond to a “real” constraint reflecting reality $\rightarrow$ left implicit.
- Only relevant constraints must be added explicitly.

Explicit and Implicit Disjunctive Mandatory Constraints

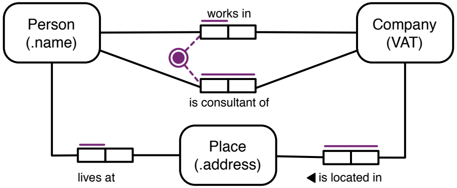
- Remember: every non-independent entity type has a population = the population of all attached roles.
- Hence, there is an implicit disjunctive mandatory constraint spanning all such roles.
- This does not correspond to a "real" constraint reflecting reality → left implicit.
- Only relevant constraints must be added explicitly.



Question: is this disjunctive mandatory constraint relevant for the domain?

Explicit and Implicit Disjunctive Mandatory Constraints

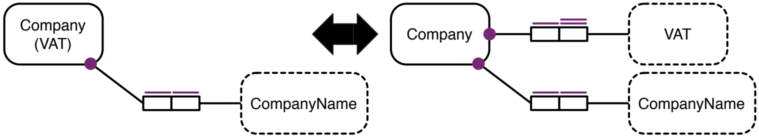
- Remember: every non-independent entity type has a population = the population of all attached roles.
- Hence, there is an implicit disjunctive mandatory constraint spanning all such roles.
- This does not correspond to a "real" constraint reflecting reality → left implicit.
- Only relevant constraints must be added explicitly.



Question: is this disjunctive mandatory constraint relevant for the domain? Yes!

Reference Scheme

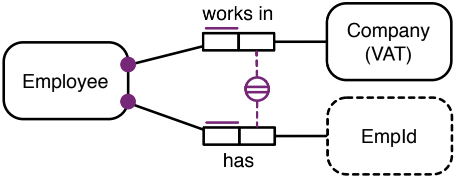
- Let us consider again the preferred reference scheme.
- We can now model it explicitly, and generalize it to cover composite reference schemes.
- A double bar represents the preferred reference scheme (or primary identifier) for the entity type attached to the adjacent role.



- This is a 1-1 reference scheme.
- In general, there are many possible reference schemes (synonyms) $\rightarrow$ choose one. . .

Compound Preferred Reference Scheme

- Preferred reference scheme that requires the combination of two or more values to identify the entity.
- Example: employee identified by the company in which she works + an internal code.



For each Company **and** EmpId, **at most** one Employee is working in **that** Company and **has** that EmpId.

How to Choose the "Best" Preferred Reference Scheme

- General strict guideline: must be an injection.
  - The scheme must 1:1 map each entity of the considered type to a tuple of one or more values (either directly or indirectly).
- Minimize the number of components.
- Prefer rigid identifiers (company names change, VAT codes don't).
- Favor stable identifiers, also considering the possibility of putting the current schema in relationship with a "bigger" schema.
- Avoid *compound disjunctive reference* if possible.
  - Formally, to be meaningful, a compound reference needs to be associated to a mandatory constraint over the involved roles together, but each of them can still be optional.
  - This is however messy.
  - And remember: in complex cases, it is also possible to introduce surrogate schemes...

Flattening, Nesting, Coreference

We want to model the following:

Every employee can write a review for a product, giving a certain score.

Flattening, Nesting, Coreference

We want to model the following:

Every employee can write a review for a product, giving a certain score.

Object types: Employee, Product, Score, Review ???
Core fact type: “write” or “write a review for”?

Flattening, Nesting, Coreference

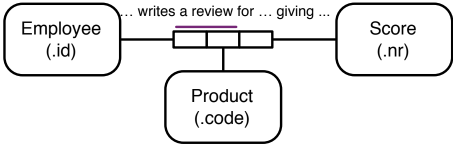
We want to model the following:

Every employee can write a review for a product, giving a certain score.

Object types: Employee, Product, Score, Review ???

Core fact type: “write” or “write a review for”?

Flattened



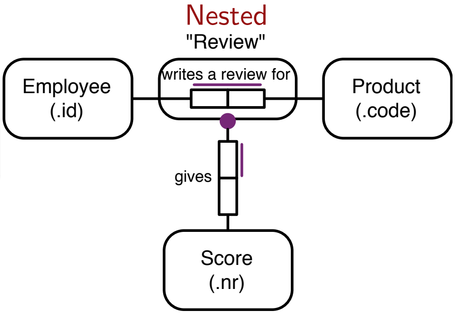
Flattening, Nesting, Coreference

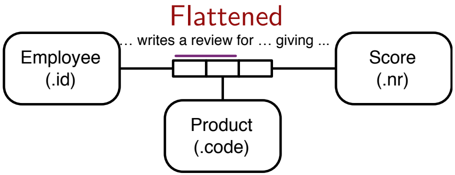
We want to model the following:

Every employee can write a review for a product, giving a certain score.

Object types: Employee, Product, Score, Review ???

Core fact type: “write” or “write a review for”?





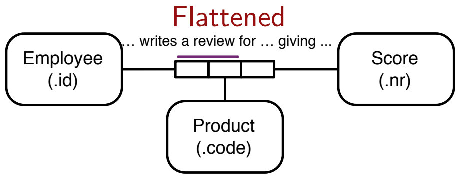
Flattening, Nesting, Coreference

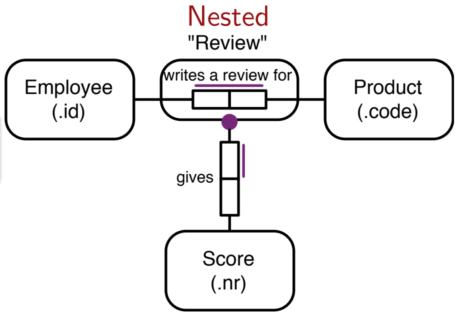
We want to model the following:

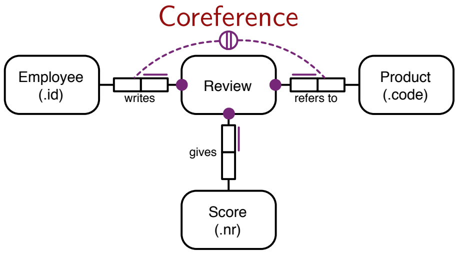
Every employee can write a review for a product, giving a certain score.

Object types: Employee, Product, Score, Review ???

Core fact type: "write" or "write a review for"?





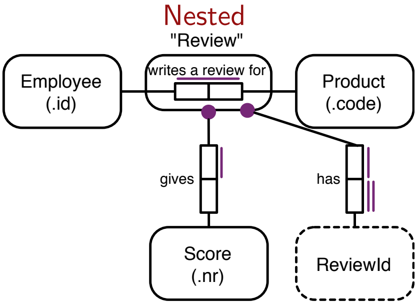


Nesting/Coreference with Explicit Id

We now want to use an internal id as preferred reference scheme for review.

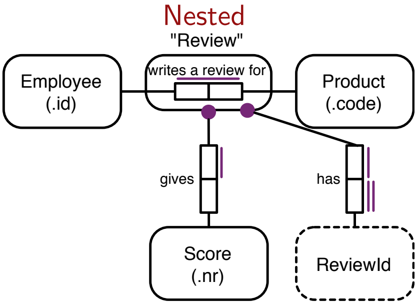
Nesting/Coreference with Explicit Id

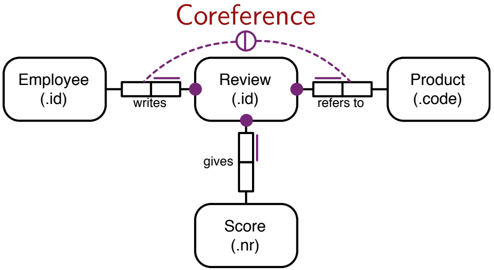
We now want to use an internal id as preferred reference scheme for review.



Nesting/Coreference with Explicit Id

We now want to use an internal id as preferred reference scheme for review.

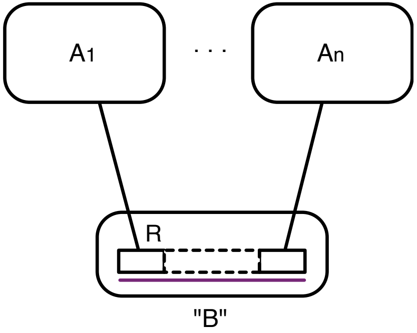




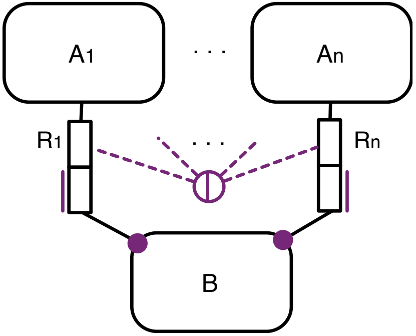
From Nesting to Coreference

General transformation:

Nesting



Coreference



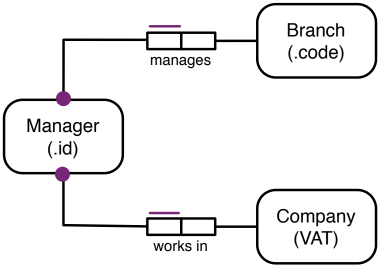
Logical Derivation Check

- Are there other important fact types that must be added, especially functional ones?
  - Functional fact type: binary fact type where at least one of the two roles has a UC.
- Are some fact types logically derivable from other fact types (thanks to constraints)?

Guideline

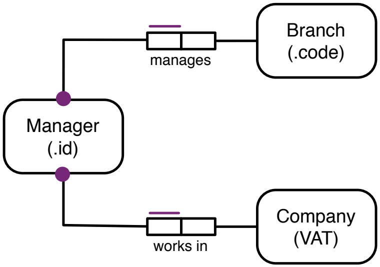
All constraints of a derived fact type are derivable.

Example

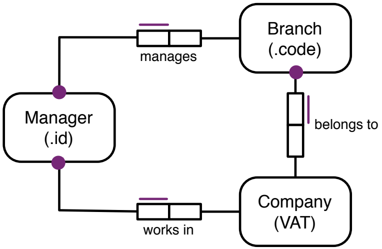


Is it possible/desirable to establish a relation between Branch and Company? Is it functional?

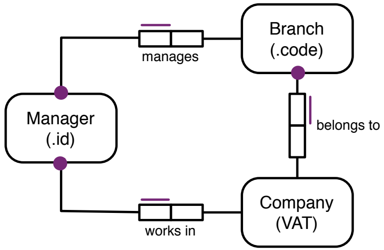
Example



Is it possible/desirable to establish a relation between Branch and Company? Is it functional?



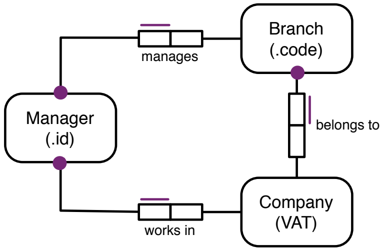
Example



Is it possible to derive one fact type from the other two?

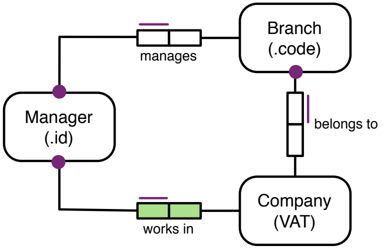
- Check if constraints are implied!
- Why is it possible for the fact type “works in” to be derivable?
- Because both the uniqueness and mandatory role are transitively implied.
- UC: implied because of the UCs on the chain.
- Mandatory constraint: implied because of the chain of mandatory constraints.

Example



- This shows that in principle "works in" is derivable. Still we need to check if semantically it is the case (domain experts).
    - Try to substitute "works in" with "is the primary contact of".
- In general, logical inference ensures that some relationship exists between Manage and Company, with 1-1 participation of the Manager. But we do not know *which*.

Example

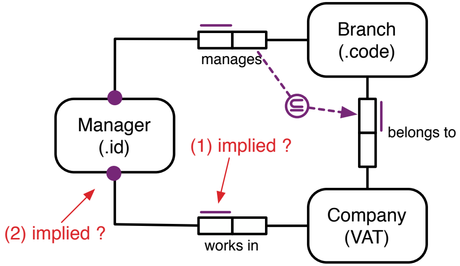


```
 Manager works in Company **iff**
Manager manages **some** Branch
**that** belongs to **that** Company
```

- This shows that in principle "works in" is derivable. Still we need to check if semantically it is the case (domain experts).
  - Try to substitute "works in" with "is the primary contact of".
- In general, logical inference ensures that some relationship exists between Manage and Company, with 1-1 participation of the Manager. But we do not know *which*.

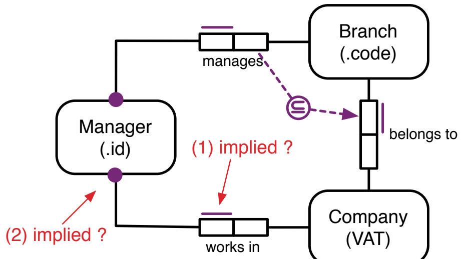
Are These Constraints Implied?

In any case, they must be shown (implied through logical inference. . .).



Are These Constraints Implied?

In any case, they must be shown (implied through logical inference. . .).

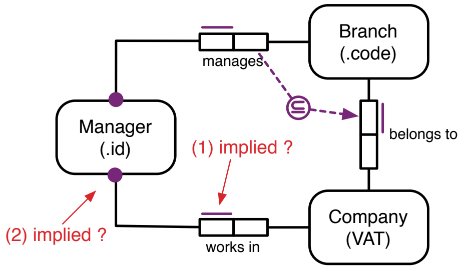


1. In principle yes (see previous slide).

Again, to check if "works in" is really the fact type that inherits these constraints, we must talk with domain experts!

Are These Constraints Implied?

In any case, they must be shown (implied through logical inference. . .).

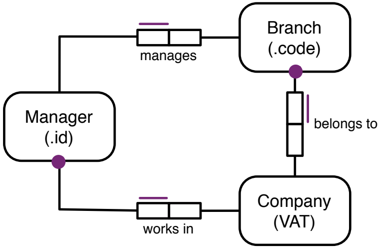


1. In principle yes (see previous slide).
2. In principle yes, thanks to the mandatory constraint from Manager to Branch and to the subset constraint expressing that each managed Branch always belongs to a Company.
    - The subset constraint would be implied if Branch had a mandatory participation in “belongs to”.

Again, to check if “works in” is really the fact type that inherits these constraints, we must talk with domain experts!

Example

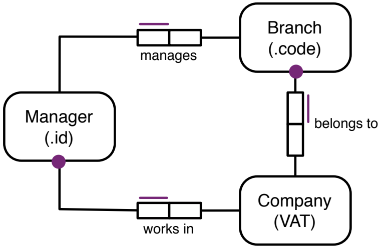
What happens if we do not have anymore the mandatory constraint between Manager and Branch?



- There exist managers that work in a Company but are not associated to any Branch.

Example

What happens if we do not have anymore the mandatory constraint between Manager and Branch?

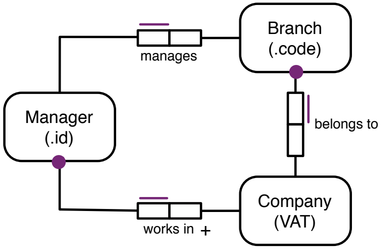


$+$ `Manager works in Company` **`if`** `Manager manages` **`some`** `Branch` **`that`** `belongs to` **`that`** `Company`

- There exist managers that work in a Company but are not associated to any Branch.
- The derivation rule only applies in one direction (is an implication).

Example

What happens if we do not have anymore the mandatory constraint between Manager and Branch?



`+ Manager works in Company` **`if`** `Manager manages` **`some`** `Branch` **`that`** `belongs to` **`that`** `Company`

- There exist managers that work in a Company but are not associated to any Branch.
- The derivation rule only applies in one direction (is an implication).
- The association between Manager and Company may be considered *semi-derived*.