

# Data and Process Modelling

## 3. Object-Role Modeling - CSDP Step 2

Marco Montali

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



# Fact Types and Sample Population

## CSDP Step 2

Draw the fact types and apply a population check.

1. Draw an **instance diagram** from the factual information obtained so far.
2. Generalize the instance diagram to a **conceptual schema diagram (structural schema)**.
3. Validate the correctness of the conceptual schema diagram with **sample population**  
→ **conceptual model** or **conceptual knowledge base**.

Remember: validation also involves issuing **conceptual queries** over the schema.

# Instance Diagram

---

<b>Person</b>	<b>Company</b>
G. Threepwood	MON5811
E. Marley	MEL1123
E. Marley	MON5811

---

# Instance Diagram

Person	Company
G. Threepwood	MON5811
E. Marley	MEL1123
E. Marley	MON5811

## CSDP Step 1

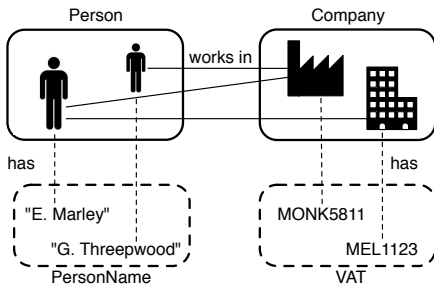
- The Person named 'G. Threepwood' *works in/employs* Company with VAT 'MON5811'.
- Person (.Name) 'E. Marley' *works in/employs* Company (VAT) 'MON5811'.
- ...

# Instance Diagram

Person	Company
G. Threepwood	MON5811
E. Marley	MEL1123
E. Marley	MON5811

## CSDP Step 1

- The Person named 'G. Threepwood' *works in/employs* Company with VAT 'MON5811'.
- Person (.Name) 'E. Marley' *works in/employs* Company (VAT) 'MON5811'.
- ...



# Conceptual Schema Diagram

- Abstraction of an instance diagram: individual objects are omitted in graphical elements.
- **Object type**: named, solid, soft rectangle.
- **Role** (object hole/relationship part): solid box.
  - ▶ Optional name in square brackets.
- **Predicate** of arity  $n$ :  $n$  contiguous role boxes.
  - ▶ One mandatory **reading** (default:left-to-right or up-to-down, otherwise arrow tip).
- **Participatory constraint**: **exactly one** line from an entity type to a role box.
  - ▶ The role can be played *only* by instances of the entity type.
- **Constraints** (see later. . .).

# Readings

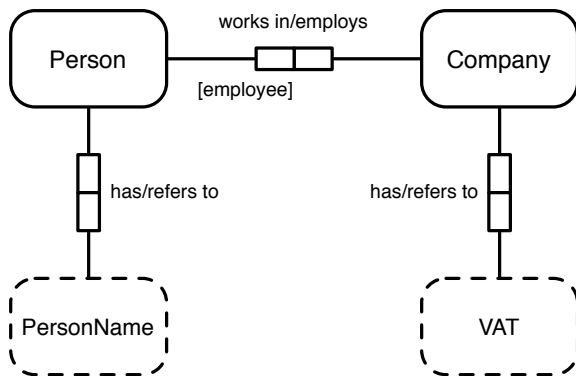
- Conventions:
  - ▶ Binary roles: optional inverse reading (separated from the mandatory one by '/').
  - ▶ N-ary roles ( $n > 2$ ): ellipsis '...' to represent object holes.
- How many (alias) readings for n-ary roles?
  - ▶ In general?
  - ▶ Displayed?
  - ▶ To easily query the schema?

# Readings

- Conventions:
  - ▶ Binary roles: optional inverse reading (separated from the mandatory one by '/').
  - ▶ N-ary roles ( $n > 2$ ): ellipsis '...' to represent object holes.
- How many (alias) readings for n-ary roles?
  - ▶ In general?  $n!$  (permutations)
  - ▶ Displayed? **1**
  - ▶ To easily query the schema?  $n$
- Guideline: define inverse reading for binary role, alias readings for n-ary roles only when needed.



# First Example



# Relationship Types and Reference Mode

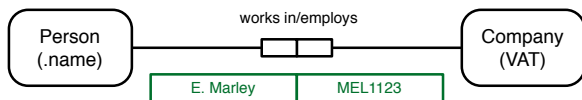
Types of relationship:

- **Elementary fact type**: relationship between entities.
- **Reference**: relationship between entities and values.
  - ▶ E.g.: The VAT number 'MEL1123' refers to some Company.
  - ▶ Also called **existential fact** (there exists a Company that has VAT number 'MEL1123').
  - ▶ Typically used for **preferred identification scheme**.
    - ★ **1:1 pattern**: every Company has a unique VAT number, every VAT number refers to a single Company.
    - ★ Compact representation using parentheses inside the entity type rounded rectangle.
    - ★ Fact tables can mention the referred values in place of the corresponding entity.

# Relationship Types and Reference Mode

Types of relationship:

- **Elementary fact type**: relationship between entities.
- **Reference**: relationship between entities and values.
  - ▶ E.g.: The VAT number 'MEL1123' refers to some Company.
  - ▶ Also called **existential fact** (there exists a Company that has VAT number 'MEL1123').
  - ▶ Typically used for **preferred identification scheme**.
    - ★ **1:1 pattern**: every Company has a unique VAT number, every VAT number refers to a single Company.
    - ★ Compact representation using parentheses inside the entity type rounded rectangle.
    - ★ Fact tables can mention the referred values in place of the corresponding entity.



# Reference Mode Types and Conversion to Value Types

- **Popular**: predefined typical reference modes.
- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
- **General**: other reference mode types.

# Reference Mode Types and Conversion to Value Types

- **Popular**: predefined typical reference modes.
  - ▶ Name, code, title, nr, #, id.
  - ▶ Dot notation: `Object_type(.ref_mode)`.
  - ▶ Conversion: `Object_type(.ref_mode) → Object_typeRef_mode`.



- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
- **General**: other reference mode types.

# Reference Mode Types and Conversion to Value Types

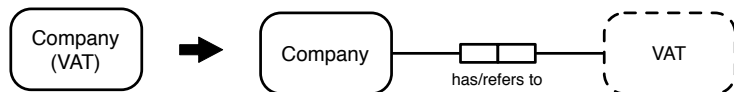
- **Popular**: predefined typical reference modes.
- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
  - ▶ Cm, m, kg, mile, USD, EUR, ...
  - ▶ Colon notation: `Object_type(:ref_mode)` or `Object_type(ref_mode:unit_type)` (unit type: mass, money, ...).
  - ▶ Conversion: `Object_type(ref_mode:)` → `ref_modeValue`.



- **General**: other reference mode types.

# Reference Mode Types and Conversion to Value Types

- **Popular**: predefined typical reference modes.
- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
- **General**: other reference mode types.
  - ▶ Examples: VAT, SSN, ISBN, URL, ...
  - ▶ Simple notation: `Object_type(ref_mode)`
  - ▶ Conversion: `Object_type(ref_mode) → ref_mode`.



# Knowledge Base Diagram

## Conceptual schema diagram + fact tables

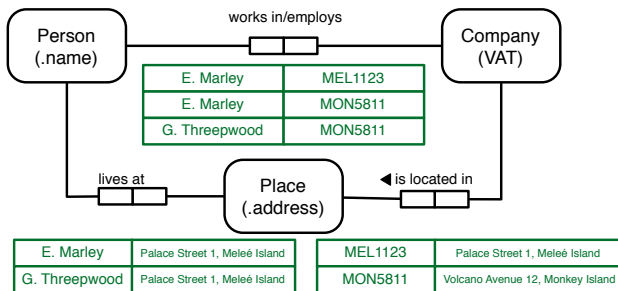
- **Fact table:** table with (*original*) instances of fact types.
  - ▶ For relationships: columns aligned to roles.
  - ▶ Values of reference modes identify entities.
- Why? Supports the validation of the conceptual schema diagram.
  - ▶ Identification of nonsensical diagrams.
  - ▶ Validation of constraints.
- Best practice: verbalize at least one fact from each fact table.



# Knowledge Base Diagram

## Conceptual schema diagram + fact tables

- **Fact table:** table with (*original*) instances of fact types.
  - ▶ For relationships: columns aligned to roles.
  - ▶ Values of reference modes identify entities.
- Why? Supports the validation of the conceptual schema diagram.
  - ▶ Identification of nonsensical diagrams.
  - ▶ Validation of constraints.
- Best practice: verbalize at least one fact from each fact table.

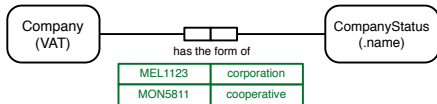


## Unary Fact Types

- Consider the possible types of companies: corporation, cooperative, ...
- Verbalization: Company (VAT) 'MEL1123' is a corporation.
- **Unary fact type**: only **one role** (being a corporation).

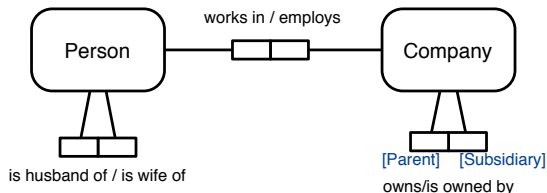


- Schema transformation: similar unaries can be factorized in a single binary.
  - ▶ “Status” object type.
  - ▶ Binary relationship between the object type and the “status” entity type.
  - ▶ Each unary becomes a value for the “status” object type.



# Heterogeneous vs Homogeneous Fact Types

- **Heterogeneous** fact type: involves distinct object types.
- **Homogeneous** fact type: all roles played by the same object type.
  - ▶ Binary homogeneous fact type: **ring** fact type.



# Reification

## Reification, Objectification, Nesting

The act of treating a relationship between objects as an object itself.

Corresponds to **nominalization** in linguistic: noun out of a verb phrase.

# Reification

## Reification, Objectification, Nesting

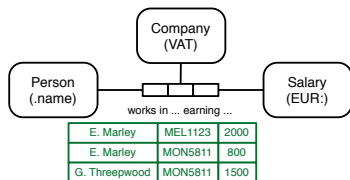
The act of treating a relationship between objects as an object itself.

Corresponds to **nominalization** in linguistic: noun out of a verb phrase.

- Person (.name) 'E. Marley' *works in* Company (VAT) 'MEL1123' *earning* a Salary (EUR:) of 2000.

vs

### Flattened



# Reification

## Reification, Objectification, Nesting

The act of treating a relationship between objects as an object itself.

Corresponds to **nominalization** in linguistic: noun out of a verb phrase.

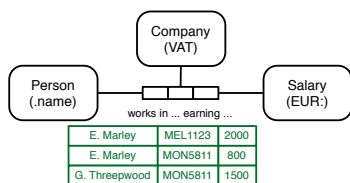
- Person (.name) 'E. Marley' *works in* Company (VAT) 'MEL1123' *earning* a Salary (EUR:) of 2000.

vs

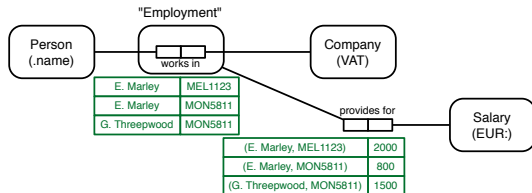
- Person (.name) 'E. Marley' *works in* Company (VAT) 'MEL1123'. **This Employment** *provides for* a Salary (EUR:) of 2000.

- ▶ Employment: reified object (name within "...").

### Flattened



### Nested

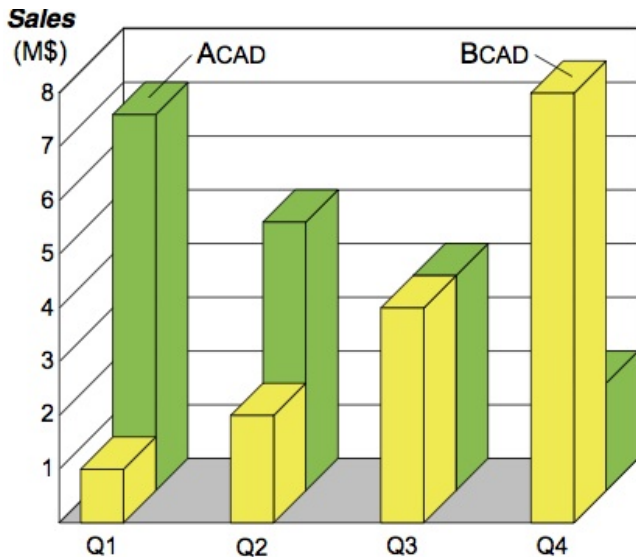


# Flattening vs Nesting

- The flattened and nested version are equivalent **only** when the role played by the reified association is *mandatory*.
  - ▶ E.g.: salary always known for each employment.
  - ▶ Why?
- We will detail these issues later on, also dealing with coreference.
- Which “form” to prefer? Modeler’s choice!
- Simple cases with mandatory objectified roles → prefer the flattened version.
- When the objectified association has optional roles, or plays many roles, → prefer the nested version (also for understandability).
  - ▶ Consider the case of “date of employment” in our example.

## Bar Chart Schematization

Try to schematize the following graphical report (from Halpin's book<sup>©</sup>).





# Pie Chart Schematization

Try to schematize the following graphical report (from Halpin's book<sup>©</sup>).

## US Federal Budget

