Database modeling with Merise

N.Benaouda

Email : nacbenenaouda@univ-setif.dz

nacbenaouda@gmail.com

Plan

- Introduction: Reminders
- Merise Method
  - ✓ Historical view
  - ✓Merise Principles
  - ✓Data Dictionary
  - ✓The conceptual schema
  - ✓The logical schema

Reminders

→Database

Structured set of coherent and lasting information, corresponding here to the activities of an organization.

→You have to model the data

- Analysis of information handled in the organization with formal representation of their nature and structuring.

→Implement the data

Use of a database management system (DBMS), with tools for data manipulation

Database Principles

- Data Modeling
- “Entity – Association” model
- “Relational” model

Data Modeling

- **Data**: Information corresponding to real elements.

  examples: a customer, a product, a supplier, a lot, a sale.

- **Entities**: Inventories of real items. Each entity is defined by a set of attributes.

  example: the product entity is described with number, name, type (vegetable or fruit), price.

- **Format**: The representation of each information. Basic data types are used (text, number, etc.) example: the name of a product is text.

- **Association**: Identification of dependencies between entities. example: a sale applies to a single product and a single customer, with an associated date and price.

Data Modeling Method

- The classic approach to an IT project includes the following steps:
- Analysis of the existing situation and needs.

- Creation of a series of models that allow all important aspects to be represented (Design).

- From the models, implementation of a database.

Data Modeling (continued)

- Once the analysis is completed, a series of models must be developed, based on the analysis document. These models will later allow us to implement a database, which will contain all the information necessary for the proper functioning of the computerized system.
- The creation of these models is done according to a certain method. We will base our course on the MERISE method (Method of Computer Study and Realization of Business Systems), which was developed during the 1970s (1978) at the instigation of the French Ministry of Industry.

Historical View of Merise Method(1)

- Merise is a methodology for information system development that was developed in France during the 1970s and 1980s.

“Merise" is an acronym for "Méthode d'Etude et de Réalisation Informatique pour les Systèmes d'Entreprise”, Or "Methodology for Information Systems Development and Realization for Enterprises" in English.

1. **Early 1970 :** Initiation of Merise by a group of French computer scientists and engineers.
2. **The 1980s:** Formalisation and documentation.
3. **1983 :** The first official version of the Merise.
4. **Key Concepts:** Merise introduced fundamental concepts, including data modeling using the Entity-Relationship Diagram (ERD), process modeling using data flow diagrams (DFD), and organizational modeling. The methodology emphasized the importance of a structured and organized approach to software development.

Historical View of Merise Method(2)

5. Widespread Adoption (1980s-1990s)

Merise gained significant traction in France and French-speaking countries during the late 1980s and early 1990s. It became the dominant methodology for information system development in these regions and was adopted by various organizations, both in the public and private sectors.

6. Evolution and Adaptation (1990s-Present)

Over time, Merise has evolved to adapt to changing technologies and methodologies in the software development field. It has incorporated object-oriented concepts and methodologies, reflecting advancements in the industry.

7. **Integration with UML (Unified Modeling Language)**

With the rise of UML as a standard modeling language in the software engineering community, efforts were made to integrate Merise concepts with UML. This integration aimed to enhance the applicability and effectiveness of Merise in a broader context.

Historical View of Merise Method(3)

8. **Continued Relevance (Present):** While newer methodologies and frameworks have emerged in the software engineering domain, Merise remains influential, particularly in French-speaking regions. Its foundational concepts and principles continue to be relevant and are often integrated into modern software development practices.

It's important to note that while Merise has had a significant impact on software development practices, particularly in French-speaking regions, the field of software engineering has evolved considerably since its inception, with newer methodologies and approaches gaining widespread adoption and influence globally.

Data Modeling (continued)

- ***The Data Dictionary (DD)***

It is a structure which brings together all the sections called **properties**, constituting the different **objects**, called **entities** of the future database.

We take stock of everything we want to put in the database and present it in a table.

Data Dictionnary

- Description

Property wording(1)	*Property Name (2)*	*Field type (3)*	*Dimensions (4)*
Customer number	*Cust_num*	*Number*	*#*
customer name	*Cust_name*	*Text*	*15*

Data Dictionnary(continued)

- **Property wording(1)** or attribute, characterizing information that is described in an unambiguous manner.
- **Property name(2)** property name abbreviated, unique name, without spaces and always respecting the same abbreviations. (Codification)
- **Field type(3)** field content. It can be:
  - ✓**Text** (which contains characters but there can also be additional numbers)
  - ✓**Numerical** (which will allow calculations)Date (or time)
  - ✓**Logical** (or boolean) example: Yes/no, True/False
- **Dimension** (4) number of characters for text

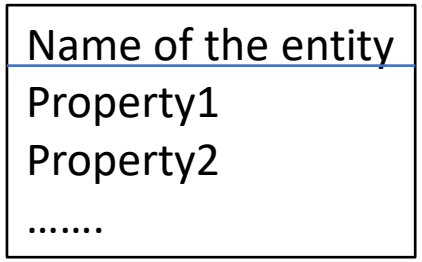
The conceptual data model(CDM)

- The conceptual data model (CDM) aims to formally write the data that will be used by the information system and **independently of the software** used. It is therefore an easily understandable representation of data, allowing the information system to be described using **entities**.

- The formalism used in this model is also known as the “Entity-Relationship Schema”. This formalism is based around 3 main concepts, **entities**, **relationships** and **properties**.

The concept of entity

- An entity is a representation of a material or immaterial element having a role in the system that we wish to describe.
- An entity makes it possible to model a set of concrete or abstract objects of the same nature.
- Formalism: An entity is characterized by its name and its properties. Represented by a rectangle, this rectangle is separated into two fields.
  - ✓ The top field contains the label.
  - ✓The bat field contains the list of properties of the entity.



Entity Examples

Client
Num_Client
Name
FirstName
Address
Postal_code
Locality

The Entity client

Occurrences of client

Client_3
003
Touati
Nihel
Constantine
25000
Locality3

Client_2
002
Filali
Omar
Alger
16000
Locality2

Client_1
001
Benali
Ali
Sétif
19000
Locality1

The entity(continued)

- Occurrence of an object

An occurrence is an individualized element belonging to the object. An entity can be:

- ✓A person or individual: STUDENT, CLIENT…
- ✓A concrete object: PRODUCT, TOOL, MACHINE...
- ✓An abstract object: ACCOUNT, TEACHING...
- ✓A place: DEPOSIT, WORKSHOP...
- ✓A documentary object: INVOICE, CONTRACT...

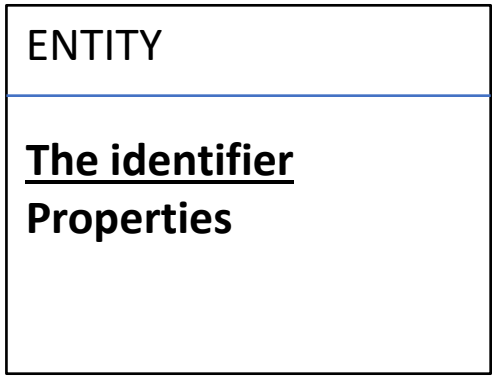
The Concept of Property

- A property is elementary data belonging to an object (Entity) or a relationship.
- Formalism: The name of the property is indicated inside the rectangle which represents the corresponding entity.
- Here are some examples of properties: For a Customer entity
  - ✓Client-name
  - ✓Client-Tel-No
  - ✓Client-Address
- A property is unique in a CDM; and cannot be attached to several different entities.

The term identifier

- An identifier is a set of properties (one or more) allowing one and only one entity to be designated.
- The original definition is as follows:

  The identifier is a special property of an object.
- Formalism: The property(ies) that constitute the identifier of an entity are underlined.
- The choice of a correct identifier is very important for modeling



The notion of relationship (association)

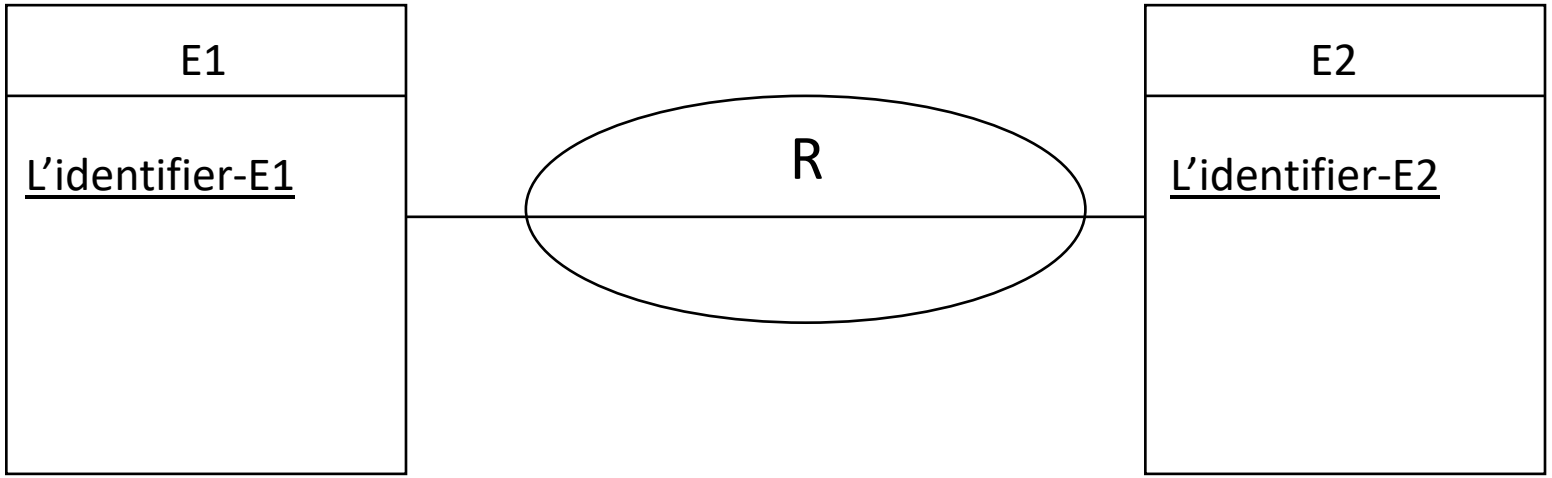
- A relationship describes a link between two or more entities. Each relationship has a noun, usually a verb. We distinguish several types of relationships depending on the number of participants:
- A recursive (or reflexive) relationship class relates the same entity.

  Example: The Marriage relationship connects two elements of the PERSON entity.
- A binary relationship connects two entities.

  Example: the relationship between the order and the customer is a binary or 2-dimensional relationship.
- A ternary relationship class connects three entities. Example: a car rental represents a relationship between a vehicle, a person and a date.
- An n-ary relationship class connects n entities.

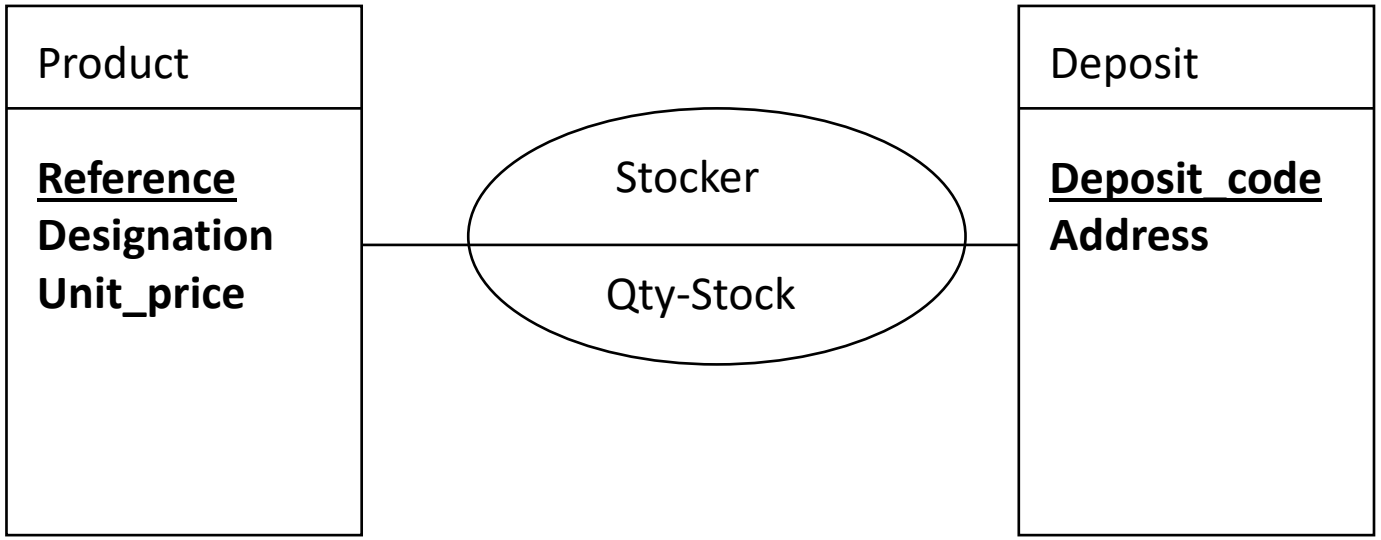
The notion of relationship (continued)

- Formalism: A relationship is represented by a hexagon (sometimes an ellipse) whose title R describes the type of relationship that connects the entities (usually a verb).



Relationship Examples

A product is stored in a deposit for a certain quantity: Qty-stock.



Occurrence of a relationship

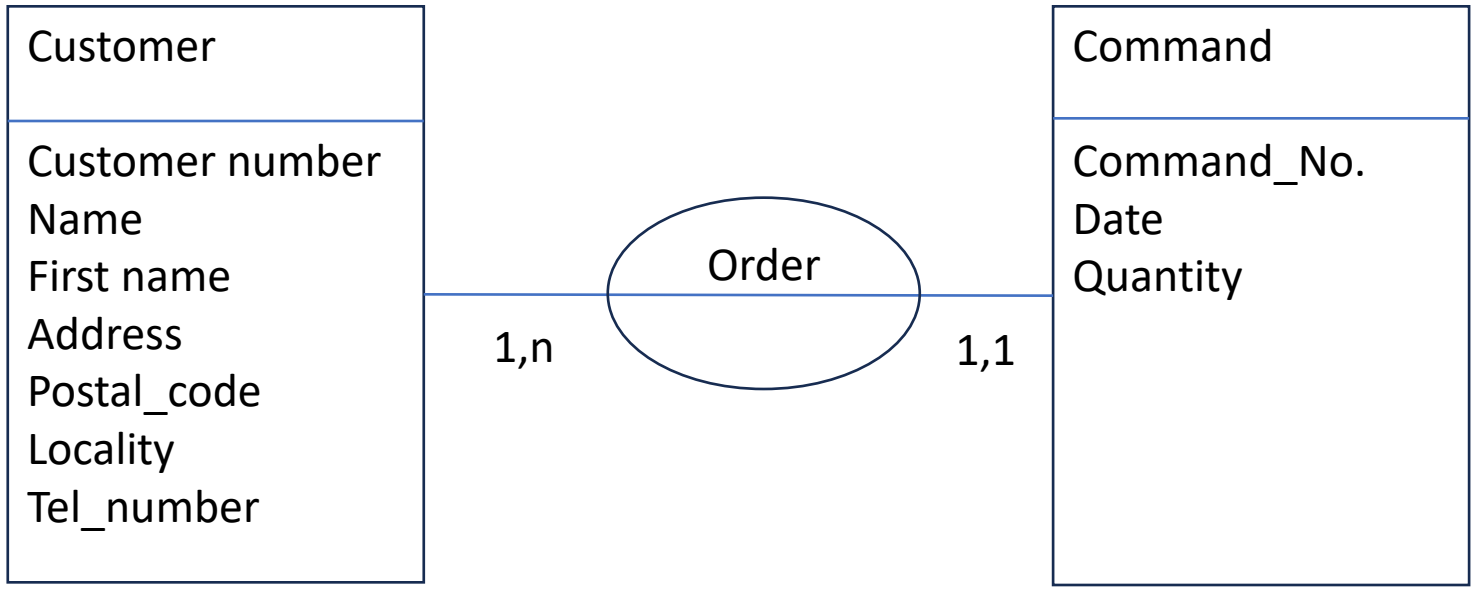
- Individualized relationship linking a single occurrence of the objects participating in this relationship.

- For each occurrence of a relationship, the identifier composed of the identifiers of the entities linked to the relationship must be unique.

Cardinalities of a Relationship

- The cardinalities allowing to characterize the link which exists between an entity and the relation to which it is connected. The cardinality of a relation is composed of a pair comprising a maximum limit and a minimum limit, an interval in which the cardinality of an entity can take its value:
  - ✓The minimum bound (usually 0 or 1) describes the minimum number of times an entity can participate in a relationship.
  - ✓The upper bound (usually 1 or N) describes the maximum number of times an entity can participate in the relationship.

Example1


Customer
Customer number
Name
First name
Address
Postal_code
Locality
Tel_number
Order
1,n
1,1
Command
Command_No.
Date
Quantity

Cardinalities (continued)

In the previous CDM,

- Between the Customer entity and the Order relationship, we have the following cardinalities:
  - ✓Minimum cardinality = 1, which means that each customer places at least one order.
  - ✓Maximum cardinality = n, which means that each customer can place several (n) orders.
- Between the Command entity and the Order relation, we find the following cardinalities:
  - ✓Minimum cardinality = 1, so each order is placed by at least one customer.
  - ✓Maximum cardinality = 1, each order is placed by a maximum of one customer.

  (In other words : each order is placed by one and only one customer.)

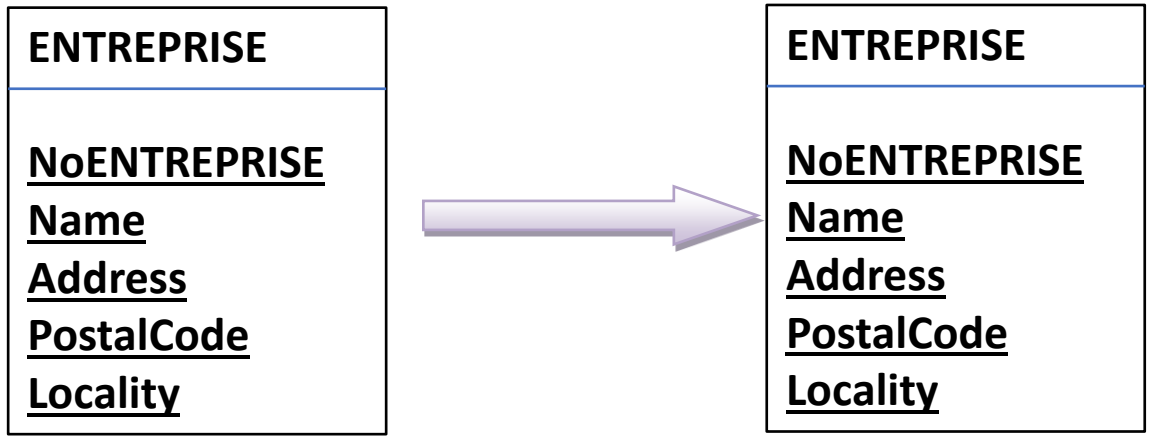
Cardinalities (continued)

- Example2: the Store binary relationship between two sets of entities PRODUCT and DEPOSIT.
- Cardinality 1-1: if and only if a product can only be stored in a single depot and a depot contains only one type of product.
- 1-N cardinality: a product can be stored in several repositories but where each repository contains only one type of product, or vice versa.
- M-N Cardinality: A product type can be stored in multiple repositories and a given repository can contain multiple product types.
- “More generally, the cardinalities of an entity in an association express the number of times that an occurrence of this entity can be involved in an occurrence of the association, at minimum and at maximum.”

Rules for Transformation from CDM to LDM

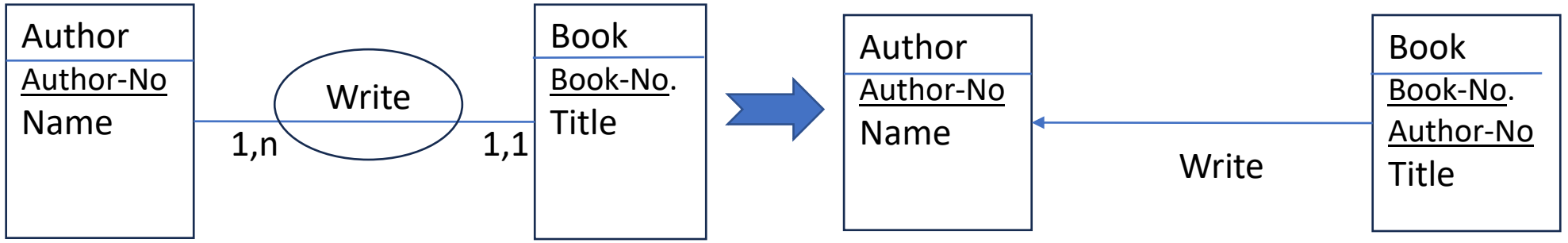
- Transforming entities

Any entity is transformed into a table. The entity's properties become the table's attributes. The entity identifier becomes the primary key of the table.



Entity Entreprise **Table Entreprise**

Transformation of binary relations of type1 (x,n) – (x,1)



Transformation of binary relations of type1 (x,n) – (x,1)

- In order to represent the relationship, we duplicate the primary key of the table based on the entity with cardinality (x,n) in the table based on the entity with cardinality (x,1). This attribute is called a foreign key. The two tables are linked by an arrow named according to the relationship, which points from the foreign key table to the table that contains the corresponding primary key.

Transformation of binary relations of type1 (x,n) – (x,1)

- The **Author_Num** attribute, which is the primary key of the Author table, becomes a foreign key in the Book table.

table.Book (Book_No, Title, #Author_Num)

means ➔ This is a foreign key.

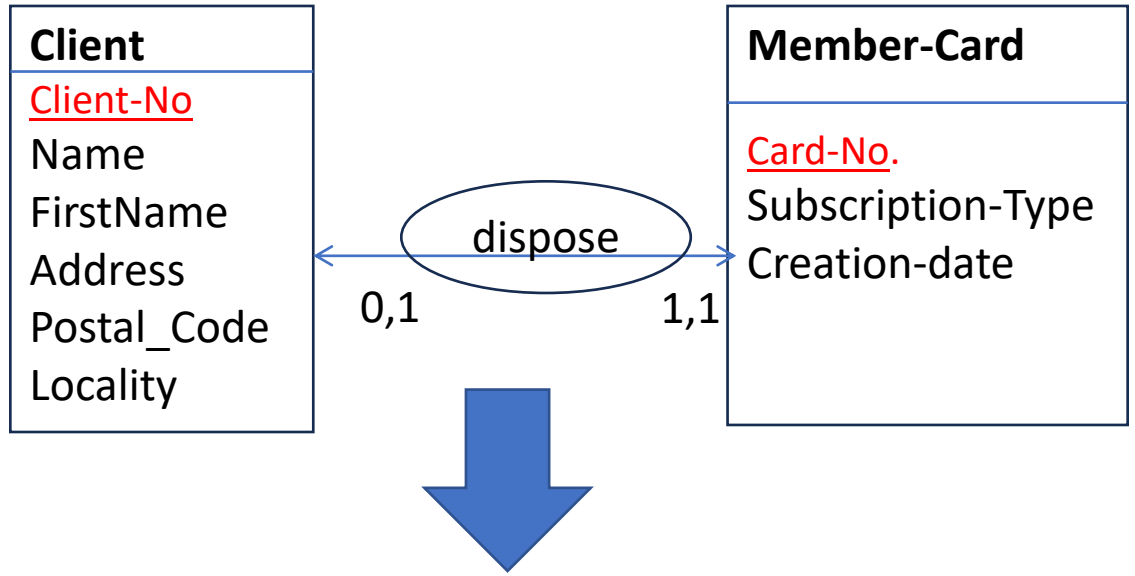
And the Author entity maintains its properties unchanged which become attributes.

Author(Author_No, Name)

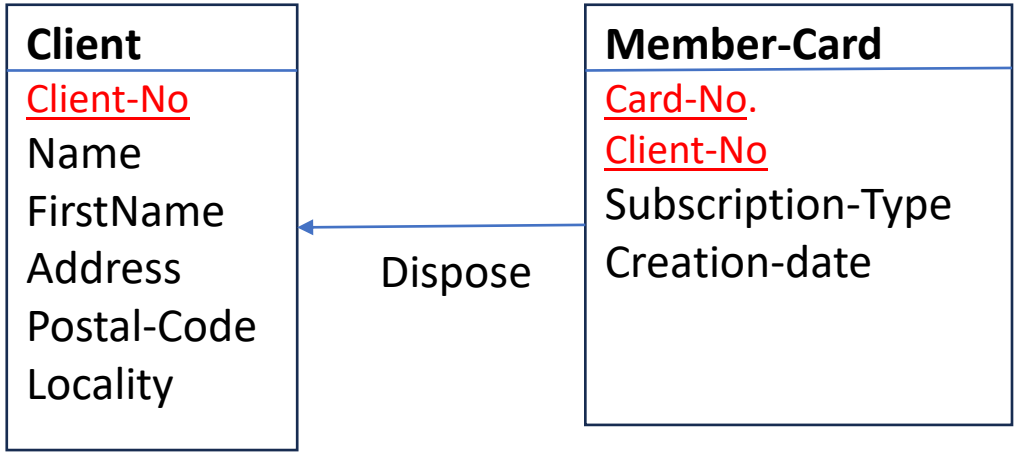
Transformation of binary relations of type (x,1) – (x,1)

1er cas: 0,1- 1,1

Entities



Tables

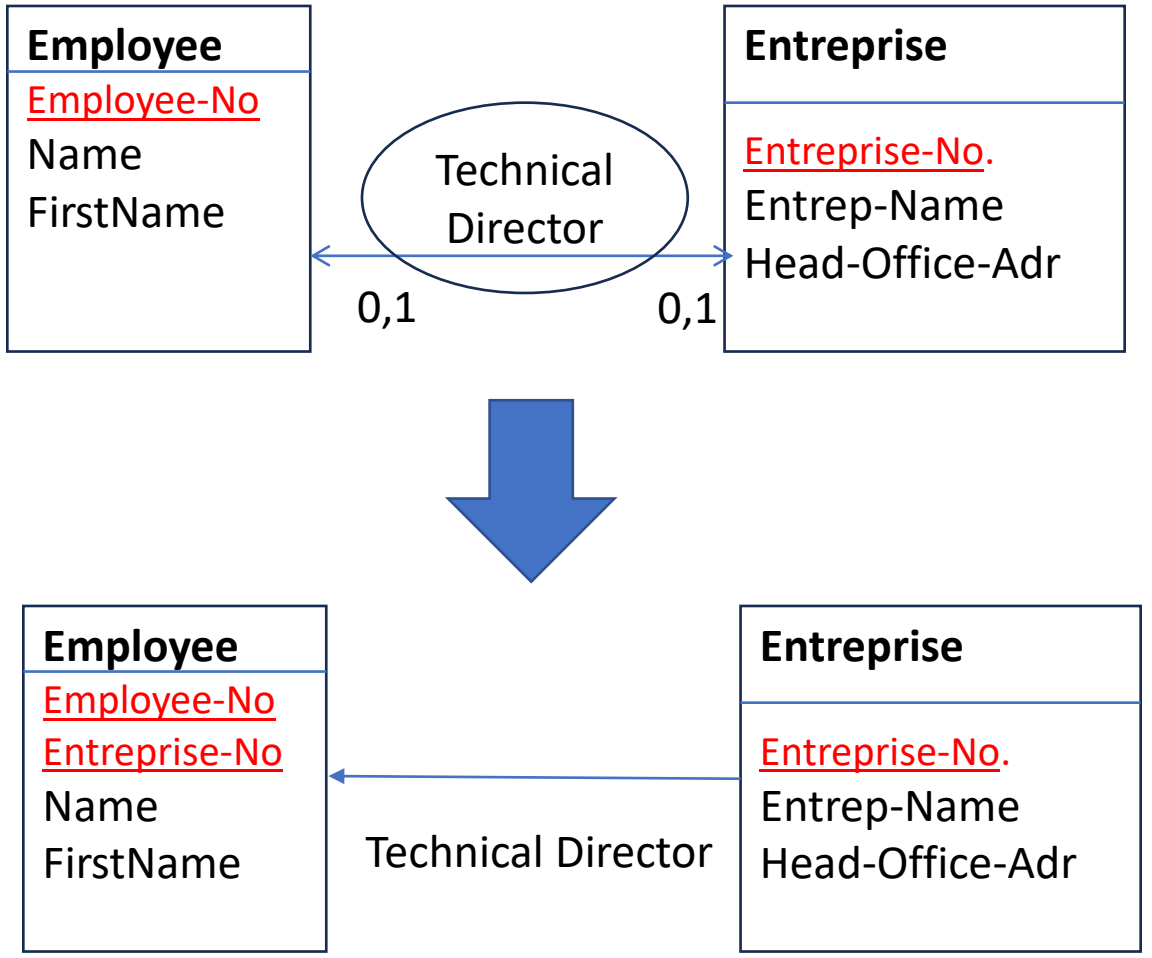


1st case: 0,1- 1,1(continued)

- Customer_No, which is the primary key of the Customer table, becomes a foreign key in the Member_Card table.

- Member_Card(Card_No, Subscription_Type,
  Creation_Date, #Client_No)

Transformation of binary relations of the type (x,1) – (x,1)
2nd case: 0,1- 0,1



2nd case: 0.1- 0.1 (continued)

- We duplicate the key of one of the tables in the other. When the relationship itself contains properties, these also become attributes of the table in which the foreign key was added.

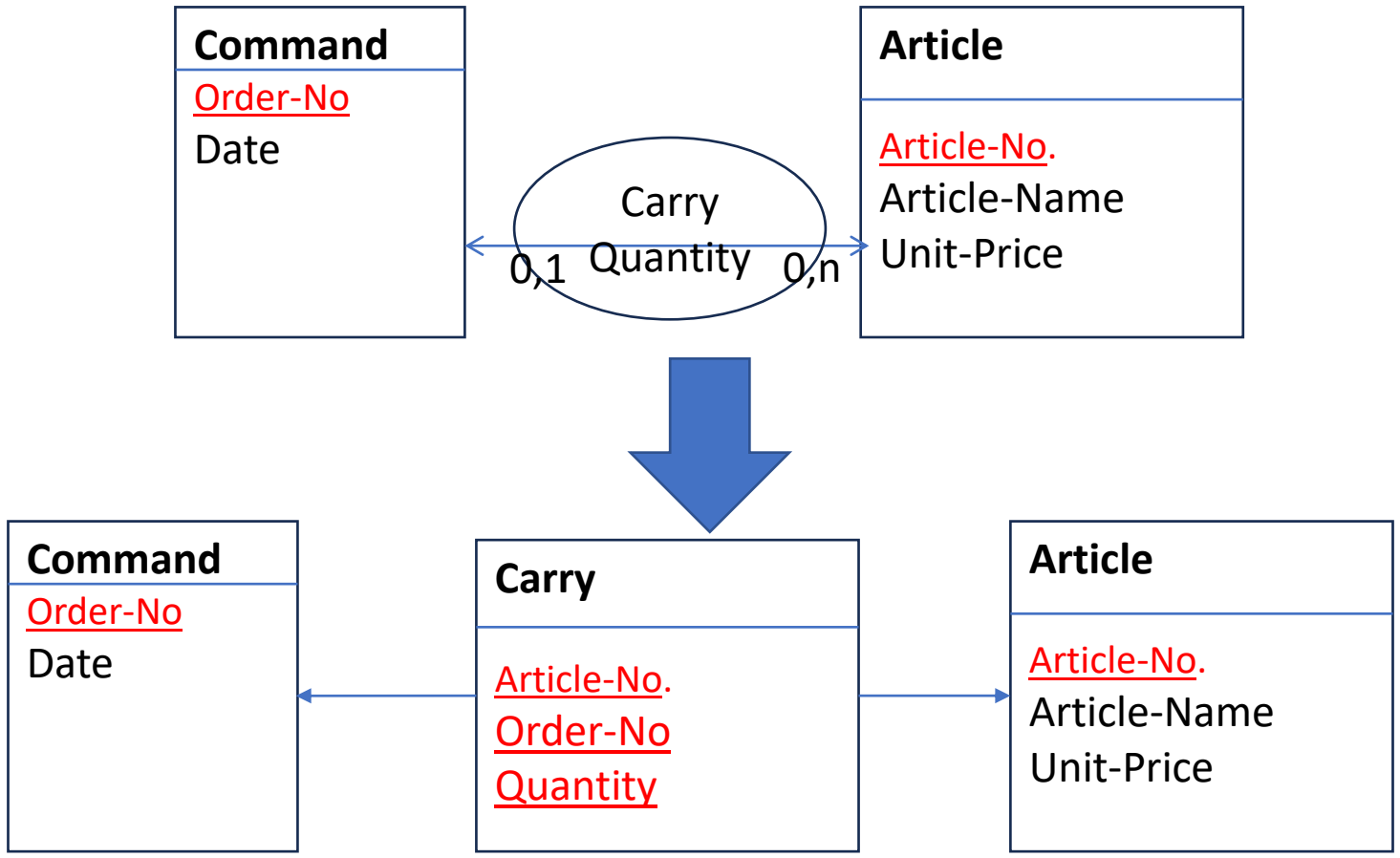
- Either we migrate the primary key from the Entreprise table into the Employee table, or we do the opposite.

Transformation of binary relations of the type (x,n) – (x,n)

- We create an additional table having as primary key a key composed of the primary keys of the 2 tables.

- When the relationship itself contains properties, these become attributes of the additional table. A relationship property that is underlined must belong to the compound primary key of the supplementary table.

(x,n) – (x,n) (Continued)



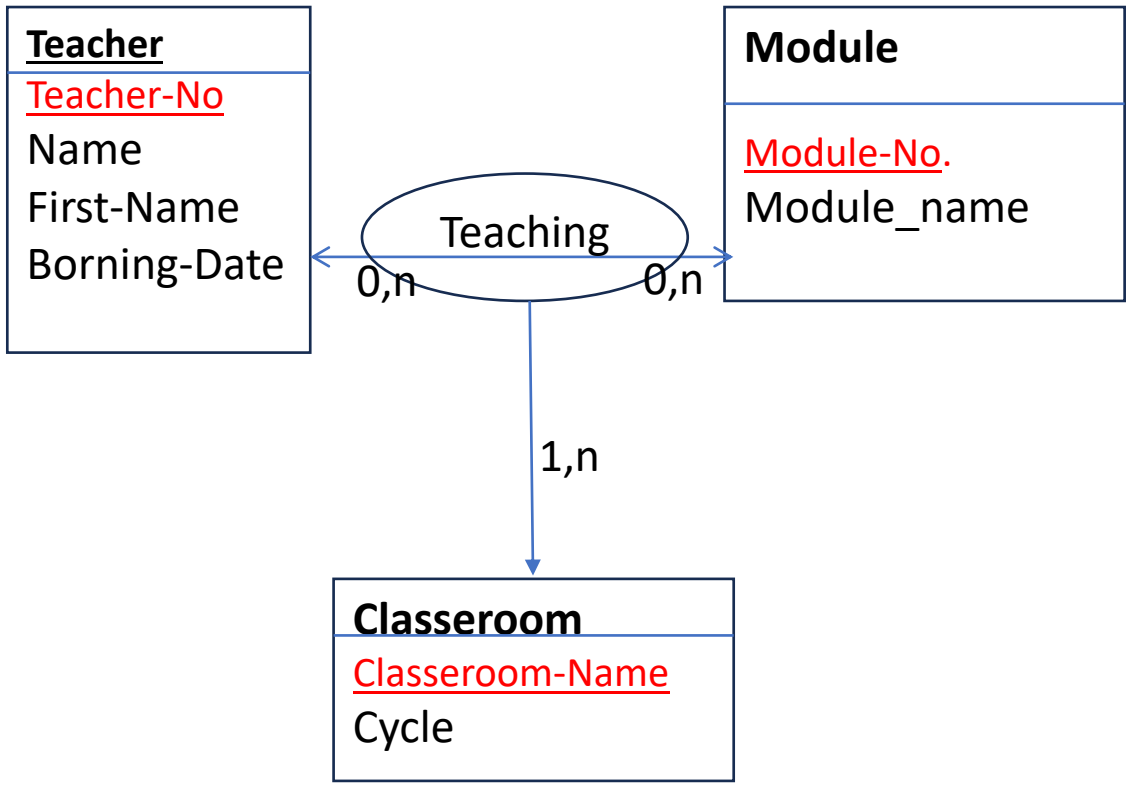
(x,n) – (x,n) (Continued)

- We create a Porter table, which contains as primary key a key composed of No-Order and Item_Code. It also contains the Quantity property from the Porter relation.

Transformation of ternary relation ships

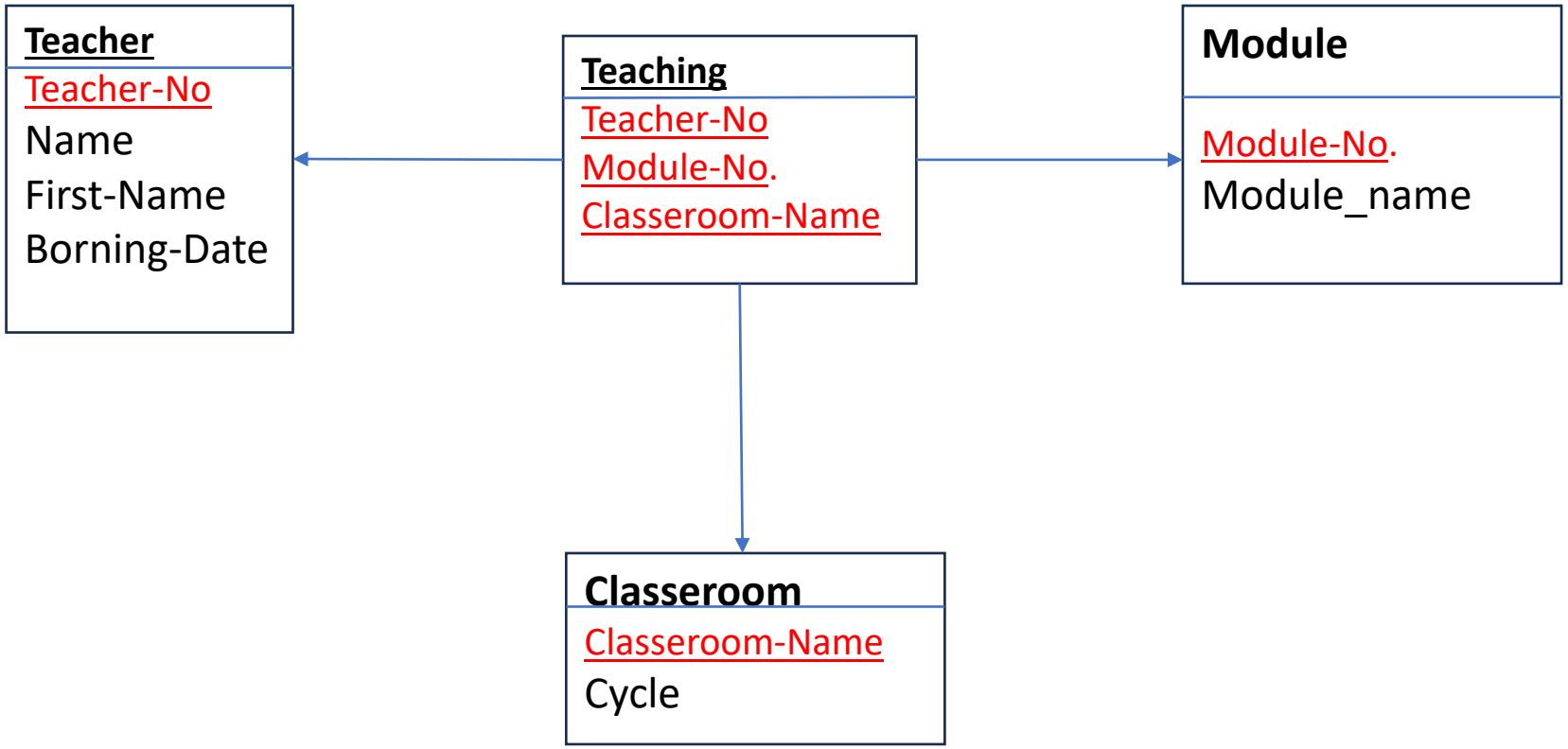
- We create an additional table having as primary key a key composed of the primary keys of all the linked tables. This rule applies independently of the different cardinalities. When the relationship itself contains properties, these become attributes of the additional table. A relationship property that is underlined must belong to the compound primary key of the supplementary table.

Transformation des relations ternaires



Transformation of ternary relationships

The Teaching table contains a key composed of Teacher-No, Module-Name and Classroom-Name.



Terminology

- Data Dictionary
- CMD (Conceptual Data Model)
- LDM (Logical Data Model)
- Relationship
- Association

مصطلحات

قاموس البيانات

نموذج البيانات المفاهيمي

نموذج البيانات المنطقي

علاقة

رابطة