Chapter 05: Logical Data Model (LDM)

Introduction

The LDM (Logical Data Model) represents an intermediate step between the conceptual model and the physical data model. It is the conceptual data model (CDM) enhanced with the definition of the logical organization of the data and optimized based on the operations to be performed on the data.

At this stage, the mode of data organization must be selected:

- Hierarchical model
- Network model
- Relational model
- Traditional file systems

The data organization mode presented in this course is the relational model.

1. Basic Concepts of the Relational Model

The relational model is a data organization model based on relationships.

- **Attribute**: An attribute is a property (a name) describing a piece of information stored in a database.

Examples of attributes: a person's age, a person's name, a social security number.

- **Domain**: The domain of an attribute is the set, finite or infinite, of its possible values.

For example, the attribute "social security number" has a domain consisting of all possible fifteen-digit combinations, while the attribute "name" has a domain consisting of all possible letter combinations (this type of combination is generally referred to as a character string or simply a string).

- **Relation**: A relation is a subset of the Cartesian product of n attribute domains ($n > 0$).

A relation is represented as a two-dimensional table where the *n* attributes correspond to the titles of the *n* columns.

- **Relation Schema**: A relation schema specifies the name of the relation and the list of attributes with their domains.

Format: *Relation Name (Primary Key, att1, att2, ..., attn)*.

Example: *Clients (ClientID, Name, FirstName, Address)*.

In a relational database management system, relations are represented as tables, attributes are referred to as fields, and each table contains records.

The table below illustrates an example of the relation *Person (Social Security Number, Name, FirstName)* and specifies its schema.

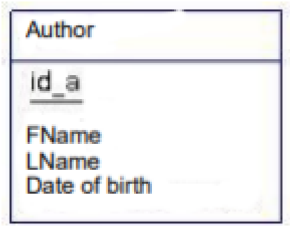
Here is the example relation table in English:

Social Security Number	Last Name	First Name
354338532	Hamza	Lamia
345353545	Hamza	Said
173354684	Hambli	Adem
973564213	Hambli	Sarra

- **Primary Key**: The primary key of a relation is the attribute that uniquely identifies a record in the relation. The primary key is underlined.
- **Foreign Key**: A foreign key in a relation consists of one or more attributes that serve as a primary key in another relation.
- **Relational Schema (Relational Model)**: A relational schema is composed of the set of all relation schemas.

2. Transformation from the CDM to the LDM (Conversion Rules)

2.1 Transformation of Entities: Each entity is converted into a relation. The properties of the entity become the attributes of the relation. The identifier of the entity becomes the primary key of the relation.



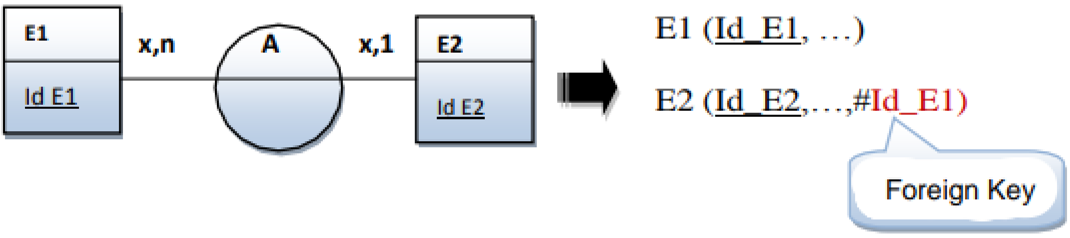
Author (id_a, Fname, LName, Date of birth)

2.2 Transformation of Associations

a) Transformation of Binary Associations of Type (x, n)-(x, 1)

In this case:

The primary key of the entity on the side of cardinality **(x, n)** is added as a foreign key in the relation corresponding to the entity on the side of cardinality **(x, 1)**.

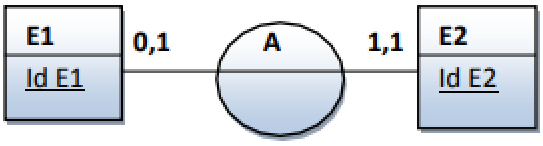


b) Transformation of Binary Associations of Type (x,1)-(x,1)

- **Case of a Binary Association (0,1)-(1,1):**

In this case:

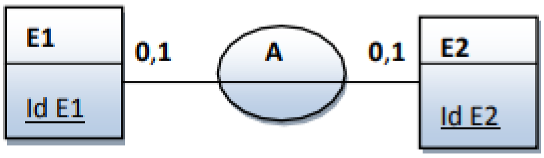
- The primary key of the entity on the side of cardinality **(O,1)** is added as a foreign key in the relation corresponding to the entity on the side of cardinality **(1,1)**.



E1 (Id_E1, …)

E2(Id_E2,…, #Id_E1)

➢ **Case of a binary association (0,1)-(0,1)**



E1 (Id_E1, …)

E2(Id_E2,…, #Id_E1)

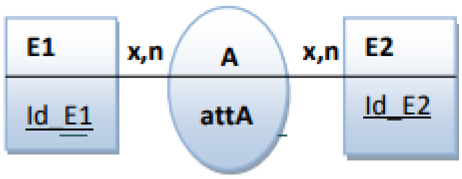
OR

E1 (Id_E1, …, #Id_E2)

E2(Id_E2,…)

c) Transformation of (x,n) (x,n) associations

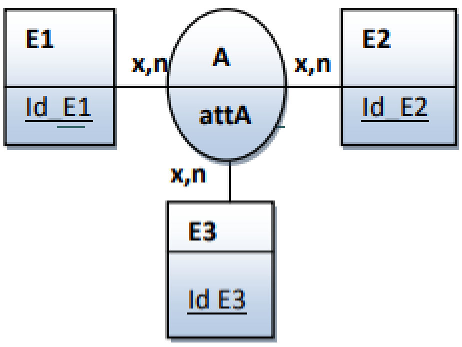
➢ **Transformation of binary associations**



E1 (Id_E1, …) E2(Id_E2,…)

A(#Id_E1, #Id_E2, attA)

➢ **Transformation of Ternary association:**



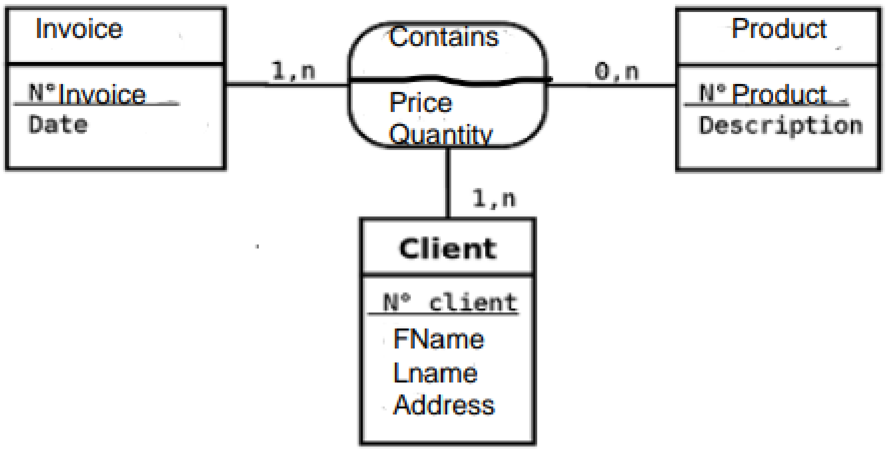
E1 (Id_E1, …)

E2(Id_E2,…)

E3(Id_E3,...)

A(#Id_E1,#Id_E2,#Id_E3, attA)

Example :



<u>Solution 1:</u>

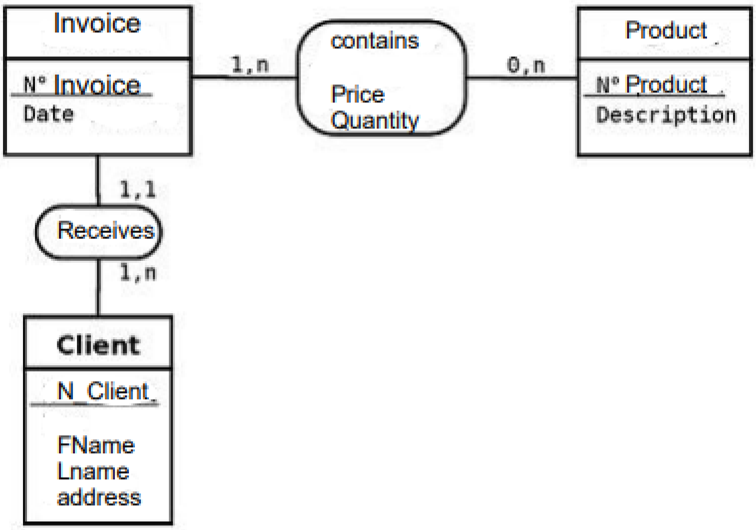
Invoice(<u>N_Invoice</u>, Date).
Product (<u>N_Product</u>, Description).
Client (<u>N_Client</u>, FName, Lname, Address).
Contains <u>(#N_Invoice, #N_Product, #N_Client</u>, Price, Quantity).

<u>Solution 2:</u>

The solution is to break the ternary association "Contains" into two binary associations.

It is always possible to eliminate an n-ary association (n>2) by reducing it to binary associations in the following way:

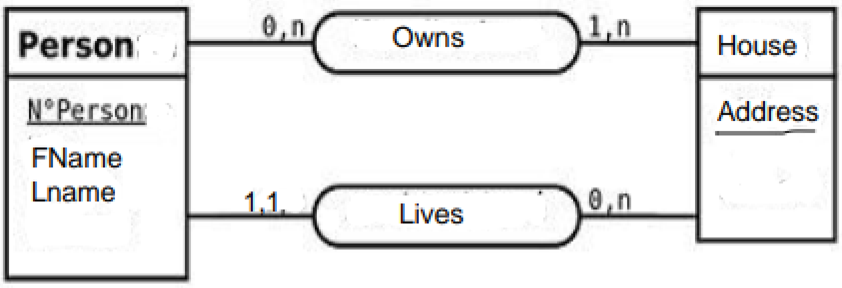
- Replace the n-ary association with an entity and assign it an identifier.
- Create binary associations between the new entity and all the entities from the collection of the former n-ary association.
- The cardinality of each created binary association is 1,1 on the side of the created entity (the one replacing the n-ary association), and 0,n or 1,n on the side of the entities from the collection of the former n-ary association.



Invoice(N_Invoice, Date, #N_Client).
Product (N_Product, Description).
Client (N_Client, FName, Lname, Address).
Contains (#N_Invoice, #N_Product, Price, Quantity).

d) Transformation of Multiple Relationships Between Two Entities

The general rules apply. Each of the associations is handled independently from the others, which may result in the addition of multiple references.



Person (N_Person, FName, Lname, #Address)

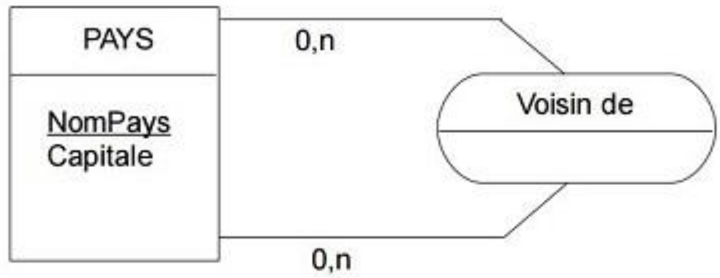
House (Address, PostalCode, City)

Owns (#N_Person, #Address)

e) The reflexive association

➔ The symmetric reflexive association (x, n) (x,n)

If A is related to B, then B is also related to A.



Indeed, if Belgium is **a neighbor of** France, then France is a neighbor of Belgium.

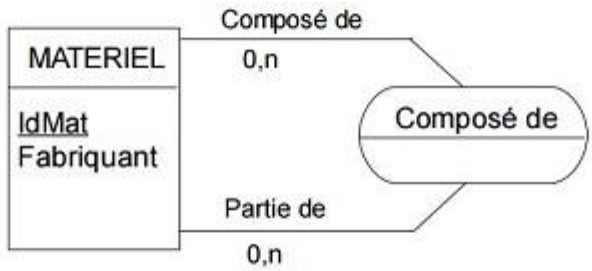
Transition to the MLD (Logical Data Model): The association becomes a table where the concatenated key is the ID of the first country and the ID of the second country, which is then renamed with the name of the association.

Neighbor_Of (<u>CountryName</u>, Neighbor_Of)

Renaming the key of the second **country** involved in the relationship is imperative, as we cannot have two fields with the same name in a table.

➔The non-symmetric reflexive association (x-n) (x-n)

If A is related to B, then B is in <u>reverse</u> relation with A.



Convention: Name the association on one line and the inverse association on the other.

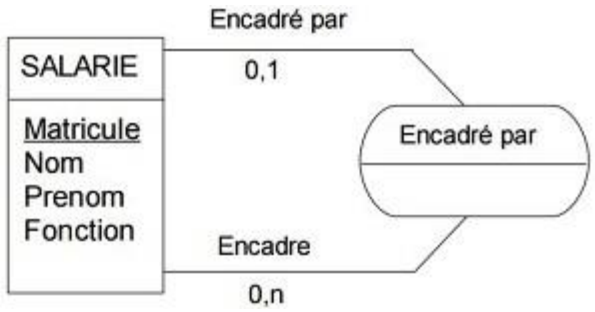
Indeed, a material can be **composed of** other materials, and some materials compose other materials.

Concrete is a material composed of cement, sand, fine stones, and water.

Transition to MLD: The association becomes a relation where the concatenated key is the material ID and the ID of one of the materials it is composed of, which is renamed with the name of the association.

Composed_of: (<u>idMat</u>, Composed_Of)

- ⇨ **Reflexive Association (X-n) (x-1)**
  A (1-n) association is necessarily asymmetric.



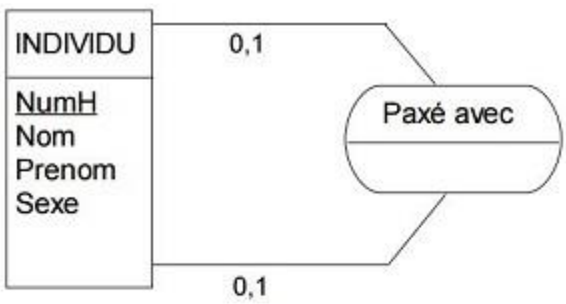
The same **employee** can simultaneously be a **superior** and have a **superior**.

Transition to the MLD (Logical Data Model): This is the case of a foreign key. We use the name of the association that labels the cardinality trait "Max 1."

Employee: (ID, Name, Surname, Role, Supervised_by)

We can observe that the CEO is an employee like the others, but he is not supervised by anyone. An occurrence that has a key representing the CEO and a foreign key representing another employee is therefore impossible. This is what the 0 in the branch 0,1 represents.

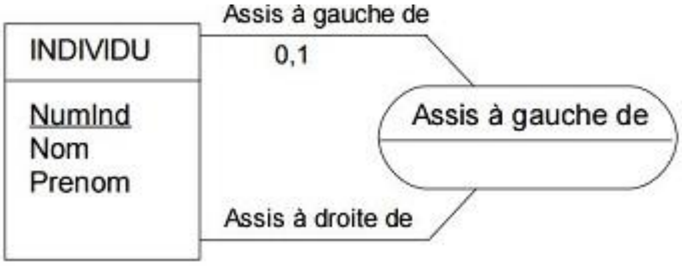
➔**The reflexive association (x-1) (x-1) symmetric**



Transition to MLD: This is the case of a foreign key, where the association name is used for the foreign key.

Individual (NumInd, LastName, FirstName, Gender, MarriedTo)

➔The reflexive association (x-1) (x-1) asymmetric



Transition to MLD: This is the case of a foreign key, where the association name is used for the foreign key.

Individual (<u>NumInd</u>, LastNameInd, FirstName, SitsToTheLeftOf)