Course Content

- History of the Database
- Relational Model
- SQL
  - Basic
  - Advanced
- PL/SQL
- Advanced Objects
- Optimization

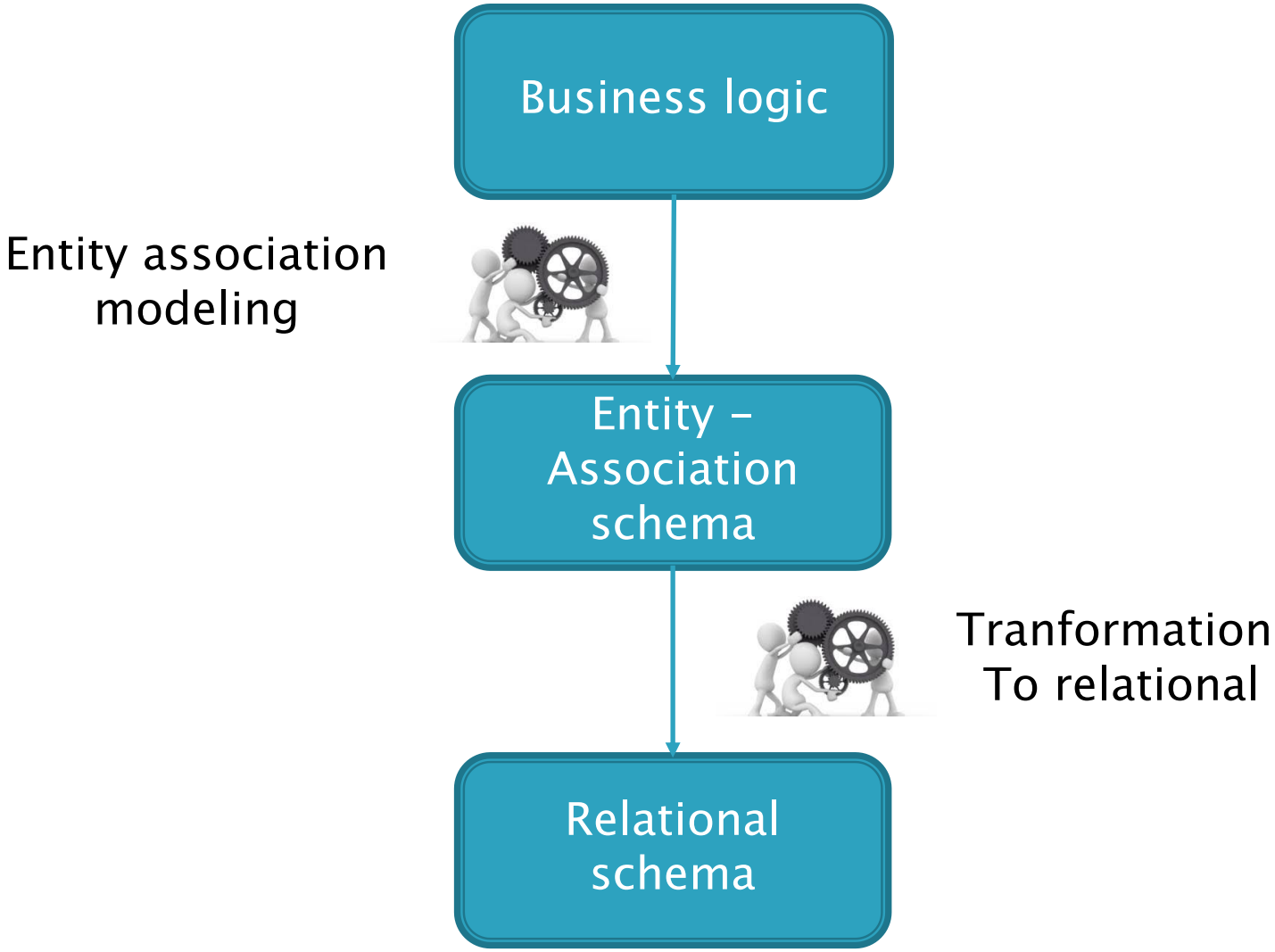
Relational schema

CCIN2P3

cnrs

in2p3

Concept


Business logic
Entity association modeling
Entity – Association schema
Tranformation To relational
Relational schema

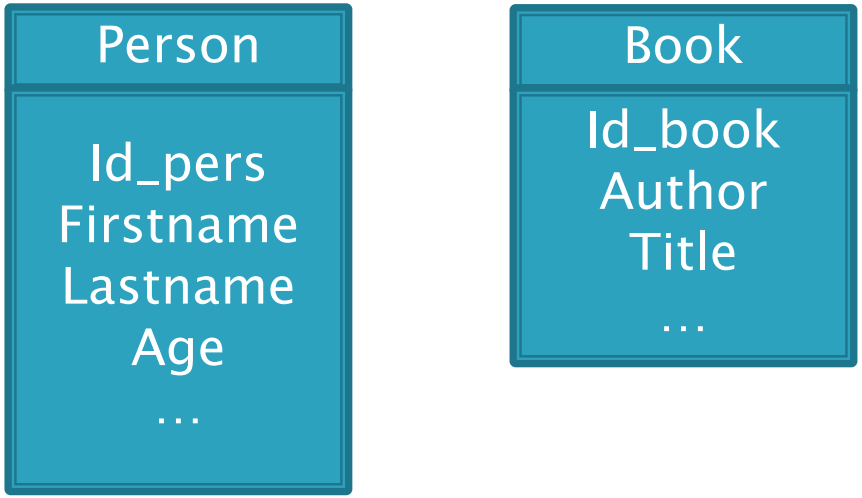
Entity-Association model

- The principle is based on a modelization based on entities and associations between them.
- An entity is an existing element or an unambiguously distinguishable identifiable concept.
  - An entity is an object, an event, a place, a person, ...
- A set / an entity class contains similar but distinguishable two by two.
  - examples: people, cars in a parking lot, a library of books ...
- Entities have properties (to distinguish them) called attributes.
- Entities in one set have the same kinds of attributes (with different values).
- Each attribute associates an entity with a value and a type (integers, reals, strings ...).
- Each Entity must be represented in a unique and unambiguous manner, to be distinguished from all others via an identifier.

Example

- The set of entities "person" has the attributes: name, surname, age, address ...
  - name associates a string of characters with the entity person.
  - age associates a short integer (not too high, come on! ;-)

- with the entity book.
  - "book" has the attributes: book number (4-digit integer) author (20-character string) title (50-character string) publisher (25-character strings)
  - "car" has the attributes: licence plate, brand, type, power, color

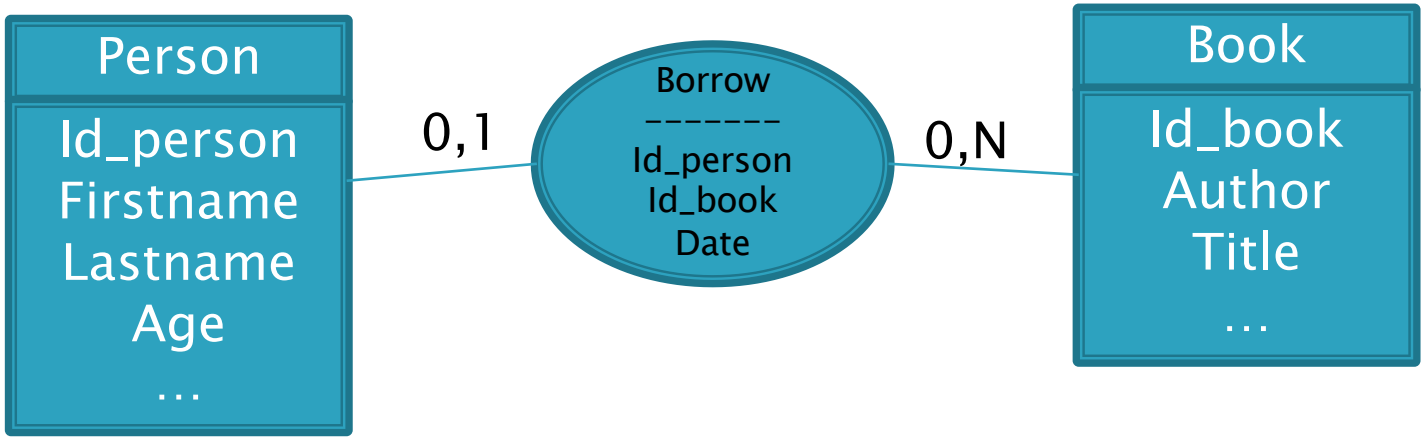


Associations

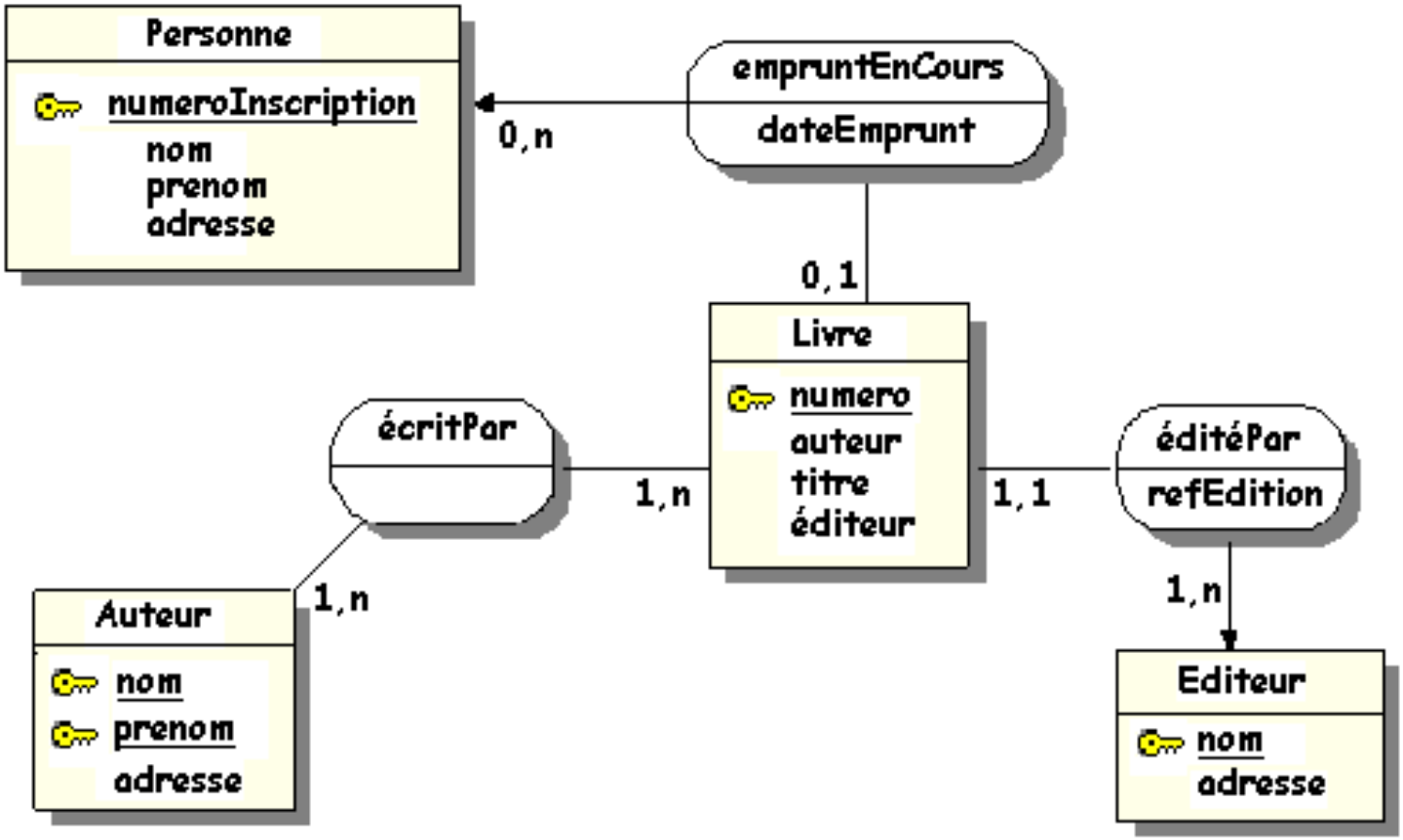
- An association class is a link between 2 or more entities classes.
- An association: It is a link between two or more entities.
- Associations :
  - From 0,1: an entity of A can be connected to none or only one entity of B
  - From 1,1: an entity of A is connected to a single entity of B
  - From 0 to several (0, N): an entity of A can be connected to none or several entities of B ...
  - From 1 to several (1, N): an entity of A may be connected to one or more entities of B

Attributes associated with an association

- When we determined a pair of entities (e1, e2) and we want to associate certain values with this pair, we associate an attribute to this association (as for an entity).
- By default, an association contains the entities identifiers that participate in the association
- When a person borrows a book we want to keep track of the loan date.
- This date would be meaningless in the Person entity, or in the Book entity, since there may be different loan dates for the same person, just as the book may be borrowed on different dates.



Example


Personne
numeroInscription
nom
prenom
adresse
0,n
empruntEnCours
dateEmprunt
0,1
Livre
numero
auteur
titre
éditeur
écritPar
1,n
1,1
éditéPar
refEdition
1,n
Auteur
nom
prenom
adresse
1,n
Editeur
nom
adresse

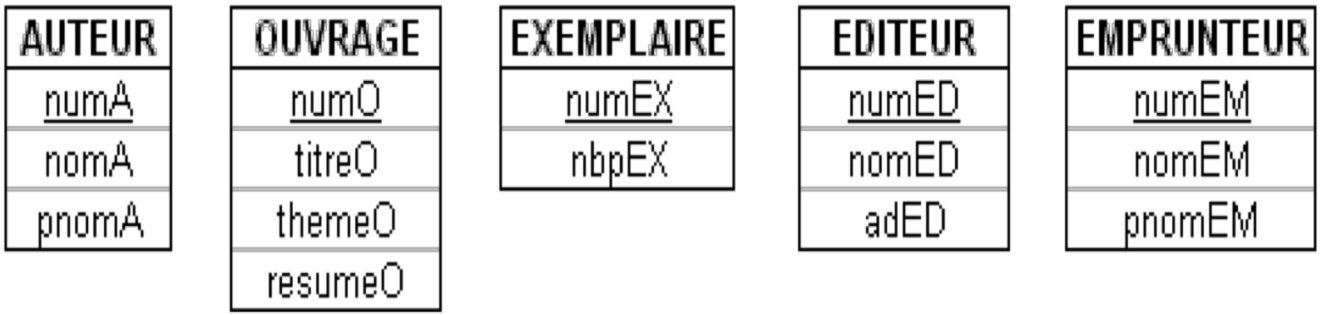
Summary

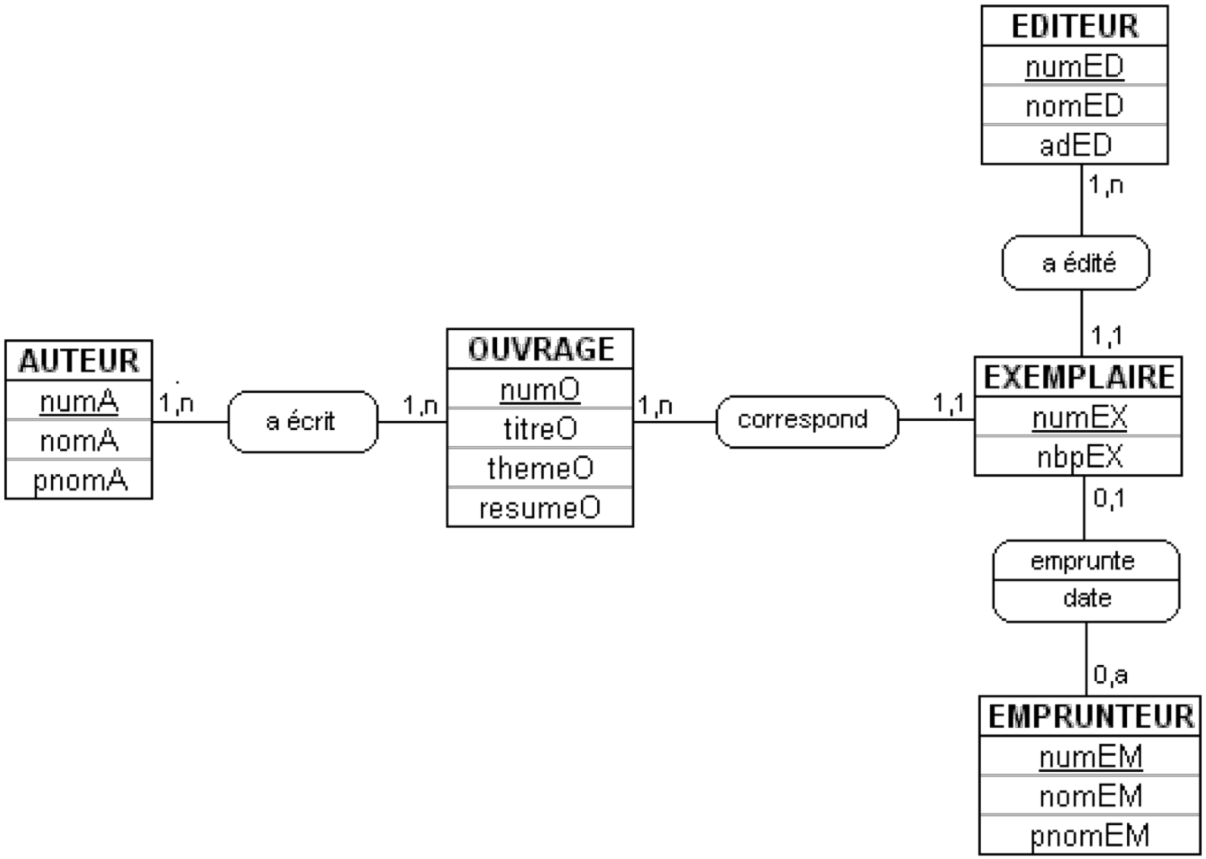
- Identify identities
- Associate an identifier to each identity.
- Associate the attributes with each identity.
- Define associations with cardinalities.

TD 1

- We want to computerize a library management and, among others, be able to know:
  - books written by the same writer / books written by a given author,
  - the topic of a book whose reference is known
  - the number of pages and the summary of a book
  - the publisher's information
- Note that a book may be available in multiple copies.
- We also want to know the borrowers, the books not returned to notify them.

Solution TD1





- Management of student internships in companies.
  - For each promotion we want to know: the supervisor(s) (from the company but also from the university) as well as their professional contact information, the date, time and room for the defense presentation as well as the grades received.

Solution TD2

ETUDIANT
numE
nomE
pnomE

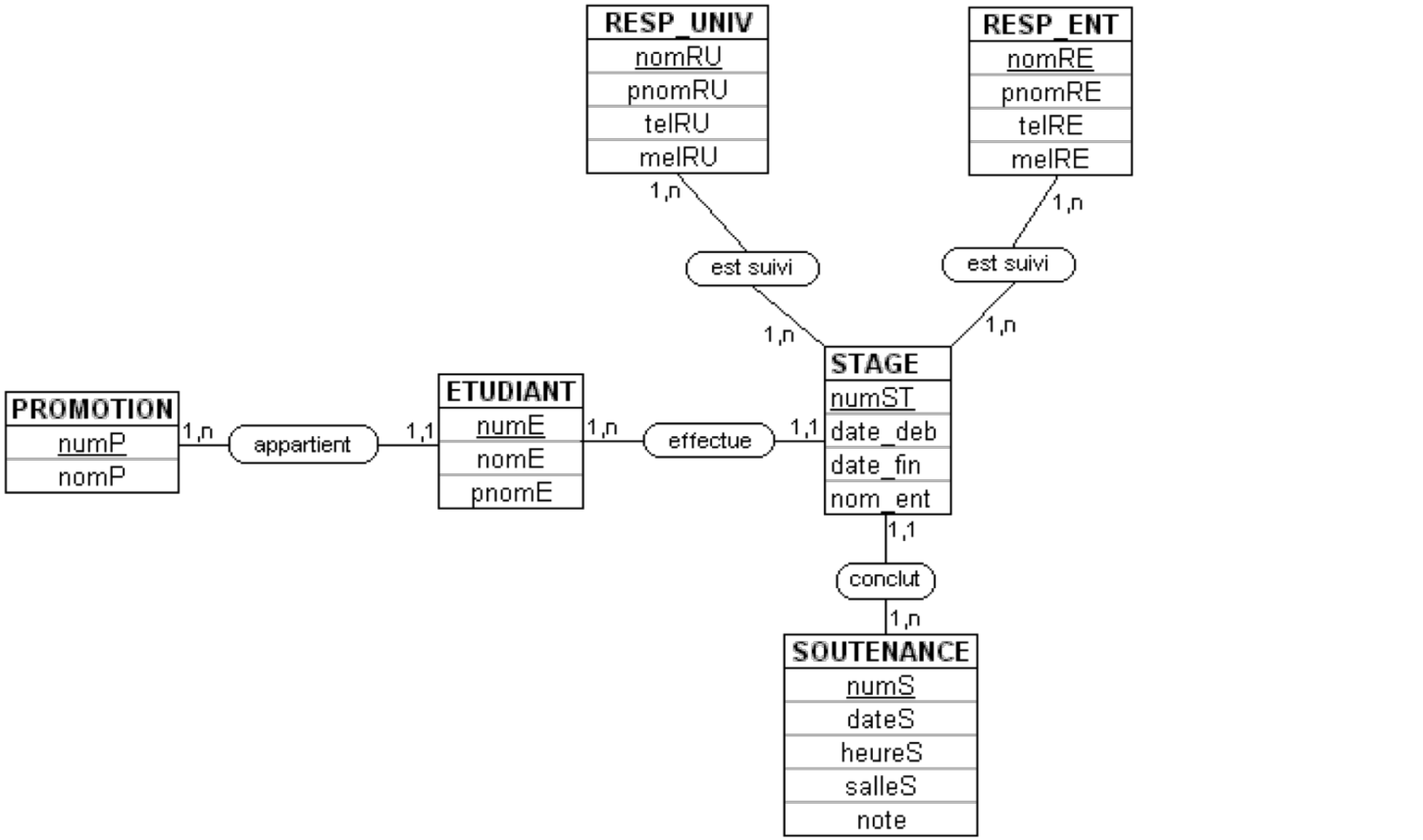
PROMOTION
numP
nomP

RESP_UNIV
nomRU
pnomRU
telRU
melRU

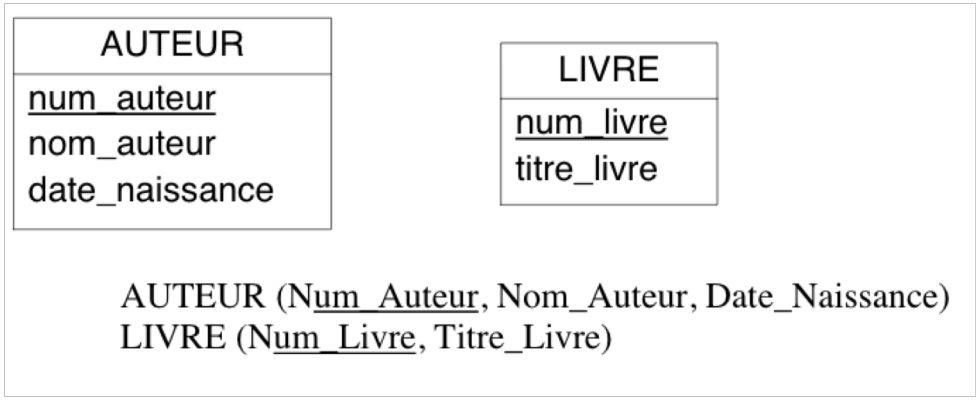
RESP_ENT
nomRE
pnomRE
telRE
melRE

SOUTENANCE
numS
dateS
heureS
salleS
note

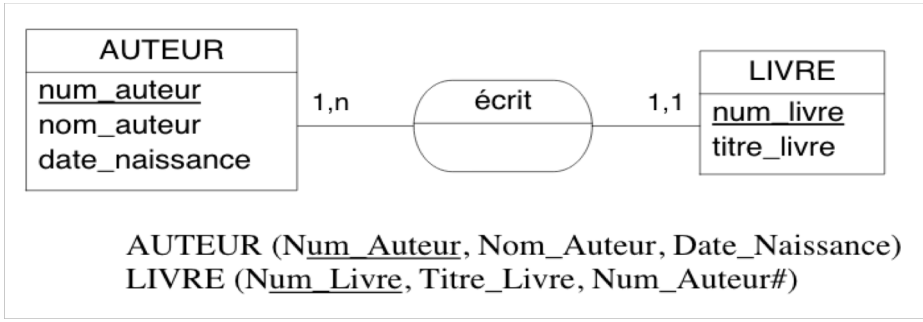
STAGE
numST
date_deb
date_fin
nom_ent



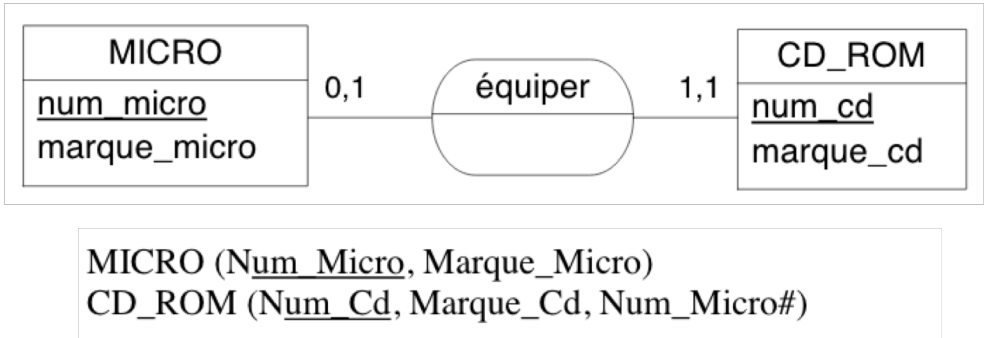
- **Step 1 : Entities**
  - Each entity of the EA model is transformed into a table
  - Entity properties become attributes of the table
  - Entity identifier becomes primary key



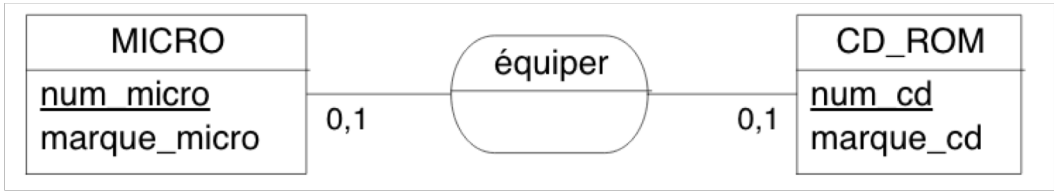
- Step 2 : Associations 1,1
  - When we find a cardinality 1,1, we add the target identifier to the source relation (it becomes a foreign key)
  - Association 1,1- 1,n



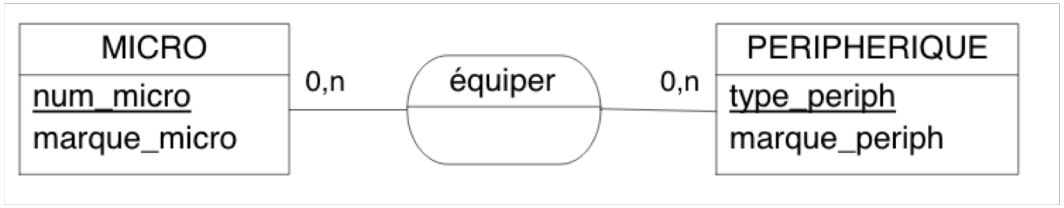
  - Association 0,1 -1,1



- Step 3 : Association 0,1 - 0,1 and x,n - y,n
  - The association becomes a relation in its own right whose key is the concatenation of the two entity identifiers concerned, including the possible association attributes.

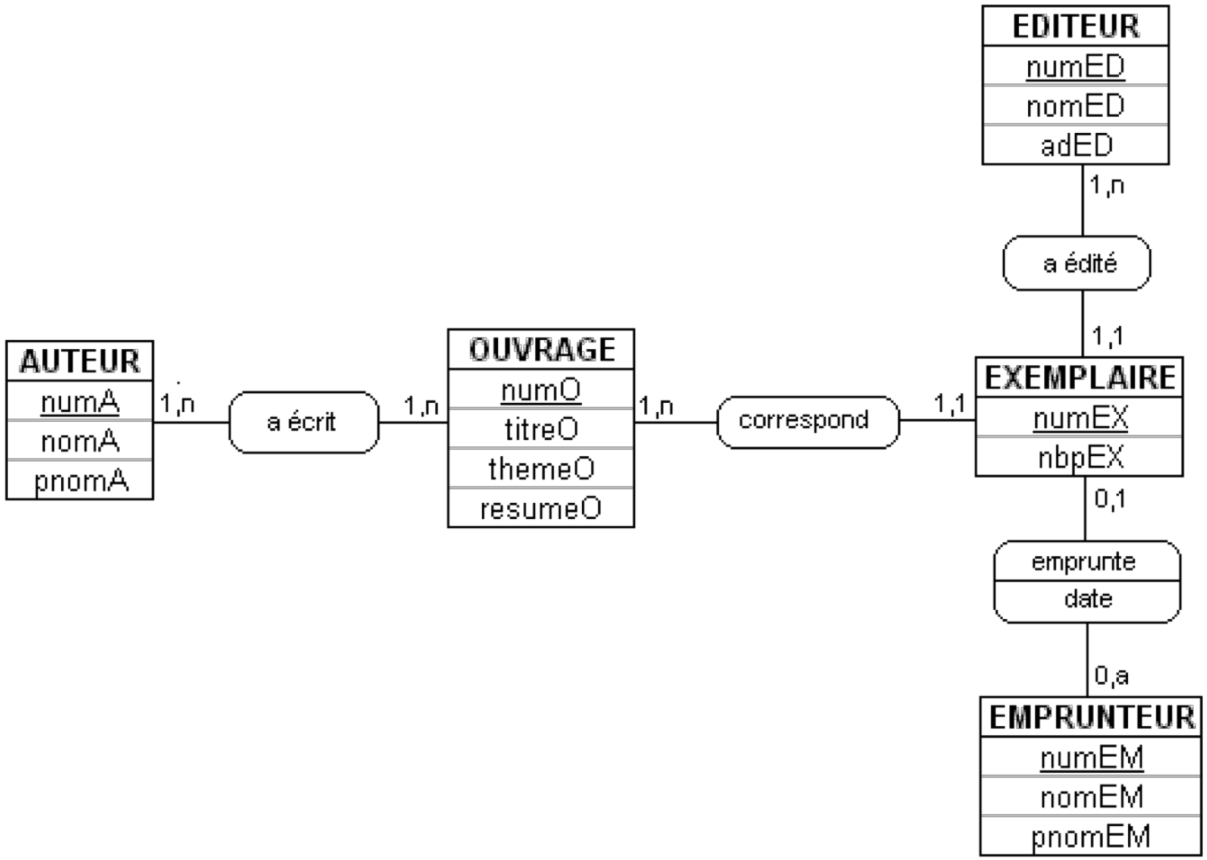


MICRO (Num_Micro, Marque_Micro)
CD_ROM (Num_Cd, Marque_Cd)
EQUIPER (Num_Micro#, Num_Cd#)



MICRO (Num_Micro, Marque_Micro)
PERIPHERIQUE (Type_Periph, Marque_Periph)
EQUIPER (Num_Micro#, Type_Periph#)

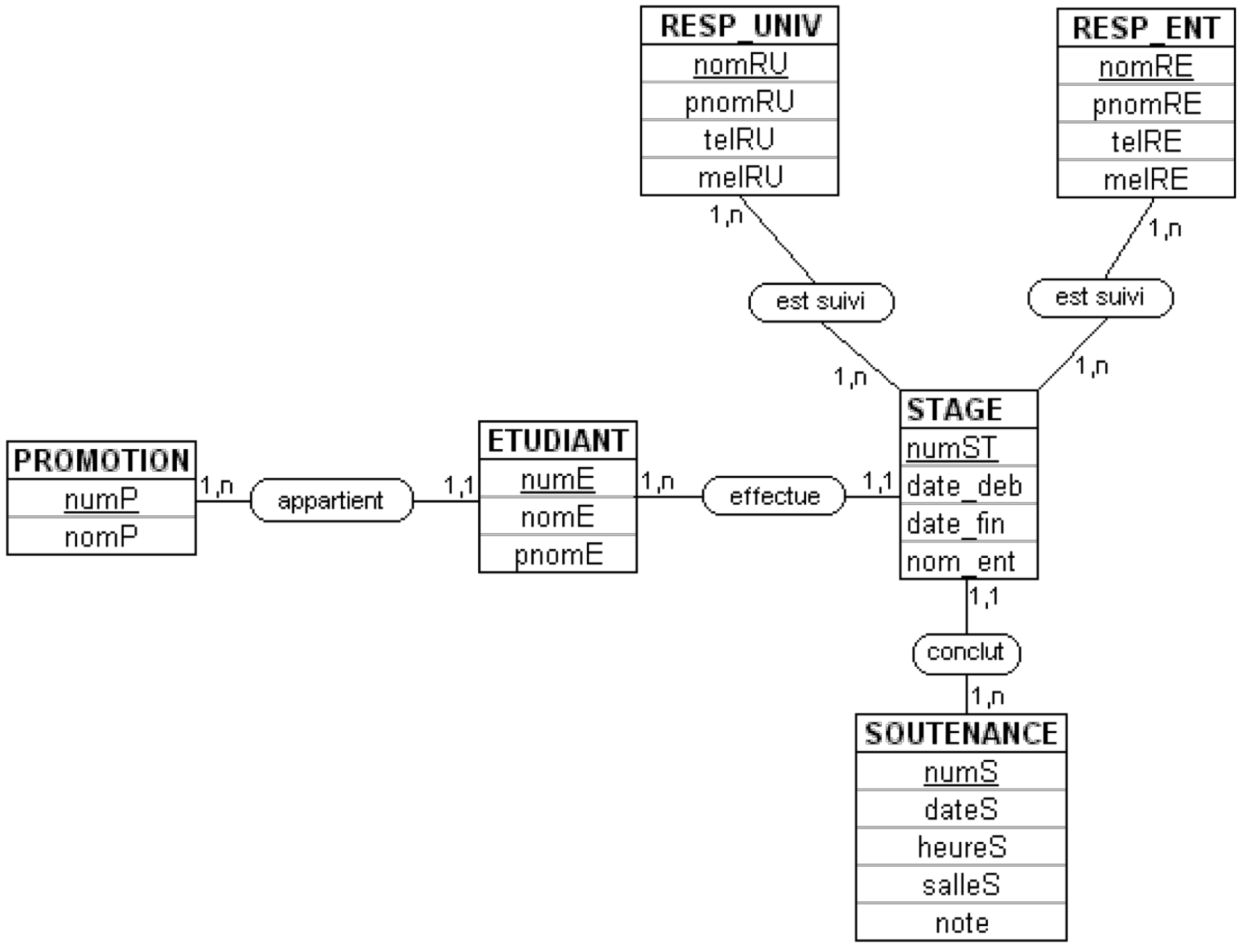
- Transform the EA model into a relational model :



Solution TD1

- Step 1 :
  - AUTEUR(numA, nomA, pnomA)
  - OUVRAGE(numO, titreO, themeO, resumeO)
  - EXEMPLAIRE(numEX, nbpEX)
  - EDITEUR(numED, nomED, adED)
  - EMPRUNTEUR(numEM, nomEM, pnomEM)
- Step 2 : associations 1,1
  - EXEMPLAIRE(numEX, nbpEX, #numO, #numED)
- Step 3: association 0,1 – 0,1
  - EMPRUNT(#numEX, #numEM)
- Step 4: association x,n – y,n
  - ECRITURE(#numO, #numA)
- Final solution:
  - AUTEUR(numA, nomA, pnomA)
OUVRAGE(numO, titreO, themeO, resumeO)
EXEMPLAIRE(numEX, nbpEX, #numO, #numED, #numEM)
EDITEUR(numED, nomED, adED)
EMPRUNTEUR(numEM, nomEM, pnomEM)
ECRITURE(#numA, #numO)
  - EMPRUNT(#numEX, #numEM)

- Transform the EA model into a relational model :



Solution TD2

- Step 1 :
  - PROMOTION(numP, nomP)
  - ETUDIANT(numE, nomE, pnomE)
  - STAGE(numST, date_deb, date_fin, nom_ent)
  - SOUTENANCE(numS, dateS, heureS, salleS, note)
  - RESP_UNIV(nomRU, pnomRU, telRU, melRU)
  - RESP_ENT(nomRE, pnomRE, telRE, melRE)
- Step 2 : associations 1,1
  - STAGE(numST, date_deb, date_fin, nom_ent,#numE,#numS)
  - ETUDIANT(numE, nomE, pnomE,#numP)
- Step 3:
  - SUIVI_UNIV(#nomRU, #numST)
  - SUIVI_ENT(#nomRE, #numST)
- Final solution:
  - PROMOTION(numP, nomP)
  - ETUDIANT(numE, nomE, pnomE,#numP)
  - STAGE(numST, date_deb, date_fin, nom_ent,#numE,#numS)
  - SOUTENANCE(numS, dateS, heureS, salleS, note)
  - RESP_UNIV(nomRU, pnomRU, telRU, melRU)
  - RESP_ENT(nomRE, pnomRE, telRE, melRE)
  - SUIVI_UNIV(# nomRU, #numST)
  - SUIVI_ENT(# nomRE, #numST)

Functional Dependencies

- The conceptual model is a good approach to design your relation model. It allows to build foundation of your business model.
- How to make sure our relation model is relevant ?
  - A poor design can lead to redundancy and related anomalies.
- How to come up with a better design by decomposing a relational schema ?
- Dr Ed. CODD introduced the concept of functional dependencies.
- A *functional dependency* (FD) is a relationship between two attributes
  - Example : between the PK and other non-key attributes within a table
  - AUTEUR(numA, nomA, pnomA)
  - DF : numA -> nomA numA -> pnomA

Functional Dependencies

- Properties:
  - Reflexivity X -> X ou (X,Y) -> X
    - cityname -> cityname
    - (cityname, department) -> cityname
  - Augmentation : If X -> Y then (X,Z) -> (Y,Z)
    - codezip -> cityname
    - (codezip, region) -> (cityname, region)
  - Transitivity : If X -> Y & if Y-> Z then X -> Z
    - codezip -> department & department -> region
  - Pseudo-transitivity : If X -> Y & if (T,Y) -> Z then (T,X) -> Z
    - codezip -> cityname & (department, cityname) -> region
  - Union : If X -> Y & If X -> Z then X -> (Y,Z)
    - codezip -> cityname & codezip -> department : codezip -> (cityname, department)
  - Decomposition : If X -> Y & If Z is included into Y then X -> Z
    - codezip -> (cityname, department)
      so codezip -> department & codezip -> cityname

Functional Dependencies

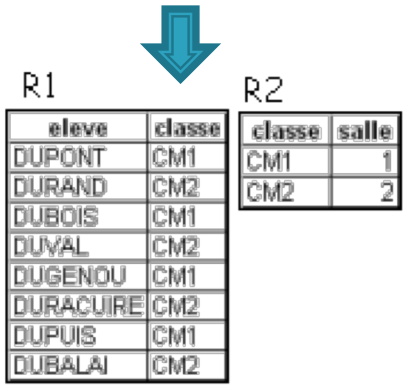
- What is a primary key ?
  - Let K be a set of attributes (possibly singleton) in a relation R
  - K is a superkey for relation R if K functionally determines all attributes of R.
  - K is a key for R if K is a superkey, but no proper subset of K is a superkey, called candidate key.
  - A primary key is a candidate key that has been selected as the means of identifying tuples in a relation.
    - CUSTOMER(fisrtname,lastname,codezip, region, department)
    - (fisrtname,lastname,codezip, region, department) is a superkey
    - (fisrtname,lastname,codezip) is a superkey
    - (fisrtname,lastname) is a superkey and a candidate key
- Thanks to the functional dependencies, Dr CODD developed the concept of Normalization.

- A method for :
  - Avoiding duplicate data
  - Running updates/deletes/inserts without inconsistency.
  - Optimizing the storage space
  - Decomposing your relation/ table without losing any data

- Example :

R

eleve	classe	salle
DUPONT	CM1	1
DURAND	CM2	2
DUBOIS	CM1	1
DUVAL	CM2	2
DUGENOU	CM1	1
DURACUIRE	CM2	2
DUPUIS	CM1	1
DUBALAI	CM2	2

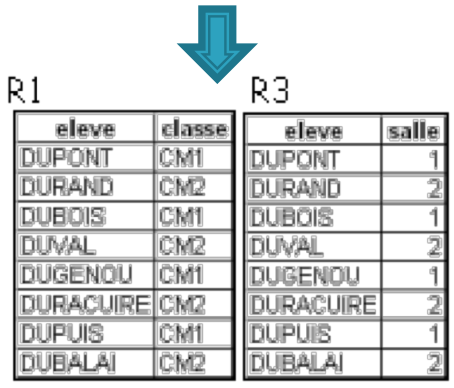


R1

eleve	classe
DUPONT	CM1
DURAND	CM2
DUBOIS	CM1
DUVAL	CM2
DUGENOU	CM1
DURACUIRE	CM2
DUPUIS	CM1
DUBALAI	CM2

R2

classe	salle
CM1	1
CM2	2

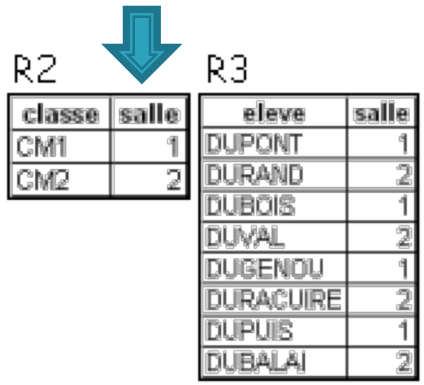


R1

eleve	classe
DUPONT	CM1
DURAND	CM2
DUBOIS	CM1
DUVAL	CM2
DUGENOU	CM1
DURACUIRE	CM2
DUPUIS	CM1
DUBALAI	CM2

R3

eleve	salle
DUPONT	1
DURAND	2
DUBOIS	1
DUVAL	2
DUGENOU	1
DURACUIRE	2
DUPUIS	1
DUBALAI	2



R2

classe	salle
CM1	1
CM2	2

R3

eleve	salle
DUPONT	1
DURAND	2
DUBOIS	1
DUVAL	2
DUGENOU	1
DURACUIRE	2
DUPUIS	1
DUBALAI	2

Normalization : 1FN

- 1st Normal Form : 1FN
  - Columns/attributes have a single value.
  - All the columns in a table should have unique names.
  - Values stored in a column should be of the same domain.
  - All the columns in a table should have unique names.

- Example :

IDEMP	FIRSTNAME	LASTNAME	CHILDS
1	john	Durand	Sophie, Julien, Yvan

NOT 1FN

IDEMP	FIRSTNAME	LASTNAME	CHILD
1	john	Durand	Sophie
1	john	Durand	Julien
1	john	Durand	Yvan

1FN

Normalization: 2FN

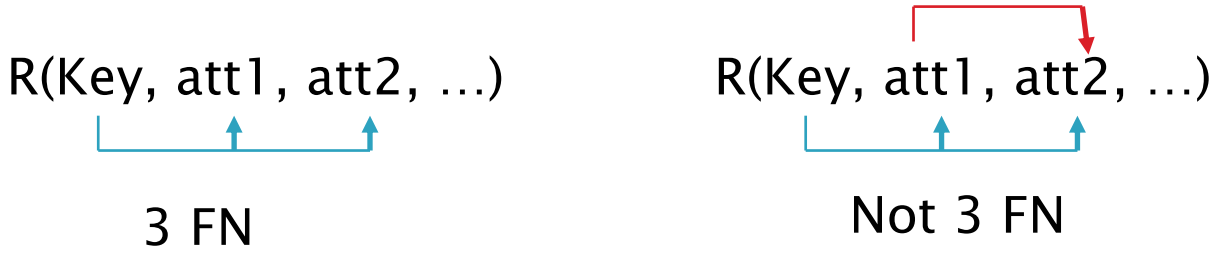
- 2nd Normal Form : 2 FN
  - The table is in the first Normal Form.
  - All attributes not belonging to the key does not depend on a part of the key.

- Example
  - LOCATION (codezip, city, department, region)
  - codezip -> department & codezip -> region

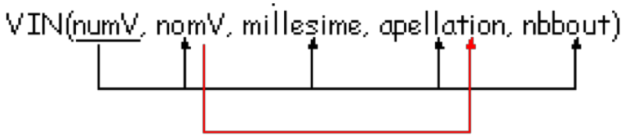
  - So department & region depends on codezip
  - LOCATION is not 2FN

  - The easyest solution is to remove columns city from the relation and create or move city to a new or an existing relation
    - LOCATION (codezip, department, region, id_city)
    - CITY(id_city, cityname)

- 3rd Normal Form : 3 FN
  - The table is in the second Normal Form.
  - All attributes not belonging to the key do not depend on an attribute which would not be in this key.
  - It means all attributes depend only on the key.



- Example :
  - VIN(<u>numV</u>, nomV, millesime, appellation, nbbottle)



2 FN

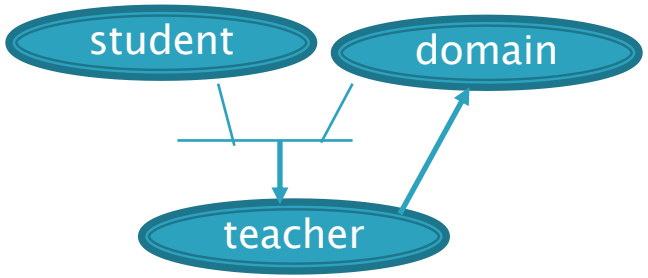
Not 3FN

  - VIN(<u>numV</u>, millesime, appellation, nbbottle, nomV)
  - ORIGINE (<u>nomV</u>, appellation)

Normalization : **Boyce-Codd** FN

- Boyce-Codd Form
  - The table is in the third Normal Form.
  - No dependency to a **super key**.
- Example

student	domain	teacher
DUPONT	Mathematics	COSINUS
DUPONT	Physics	TOURNESOL
DURAND	Mathematics	COSINUS
DURAND	Physics	EINSTEIN



  - What happens if the student DUPONT does study any more Physics we lose the teacher TOURNESOL !
- Decomposition

Teacher	domain
COSINUS	Mathematics
TOURNESOL	Physics
EINSTEIN	Physics

student	teacher
DUPONT	COSINUS
DUPONT	TOURNESOL
DURAND	COSINUS
DURAND	EINSTEIN

TD : Ex 1

- Here is a student relation :
  - STUDENT(ID, Name, Age, Club,Room)
  - ID -> Name, Age
  - ID -> Club
  - Club -> Room
- What is the NF ?

  2 FN
- Transform the relation in the 3rd FN ?

  Is not 3 FN due to Club -> Room

  STUDENT(ID, Name, Age, Club)
  Club(Club, Room)

TD: Ex 2

- Let the following relation :
  - orders (IdOrd, DateOrd, IdCust, AdrCust, IdItem,Price, Qty)

- What is the primary key ?

IdOrd → DateOrd, IdCust, AdrCust
IdOrd, IdItem → Price, Qty
IdCust → AdrCust
IdItem → Price

(IdOrd, IdItem) covers all attributes so it's a candidate key

- What normal from is it ?

orders is 1 FN : All attributes are unique and atomic
Order is not 2 FN :
IdItem -> Price
IdOrd → DateOrd, IdCust, AdrCust

- Transform it in Boyce-Code Form

Transformation to 2FN
orders (IdOrd, DateOrd, IdCust, AdrCust)
order-item (IdOrd, IdItem, Qty)
items(IdItem, Price)

NOT 3FN
IdCust -> AdrCust

Transformation to 3FN
Customer(IdCust, AdrCust)
orders (IdOrd, DateOrd, IdCust)
order-item (IdOrd, IdItem, Qty)
items(IdItem, Price)

IS BC NF