Base de données

Séance 2

1. MLD
2. Dépendance Fonctionnelle

Vérification et validation du modèle

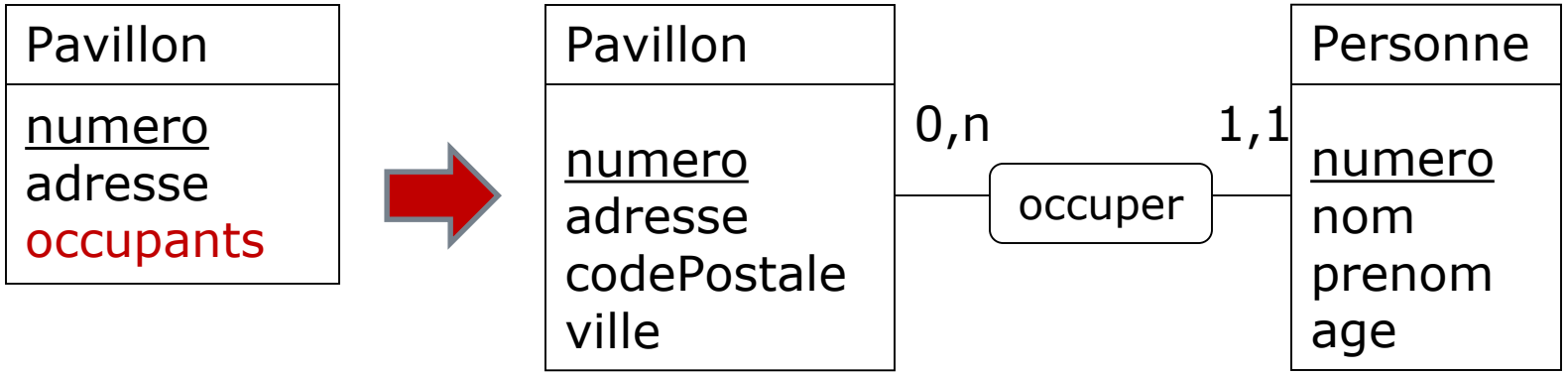
- Les entités et les propriétés doivent vérifier
  - L'intégrité sur les entités
  - L'intégrité référentielle
  - Chaque propriété doit être **élémentaire**
  - Chaque information ne doit apparaître **une seule fois** dans une entité donnée.
  - Chaque propriété doit prendre **une et une seule valeur** pour une occurrence donnée

Normalisation

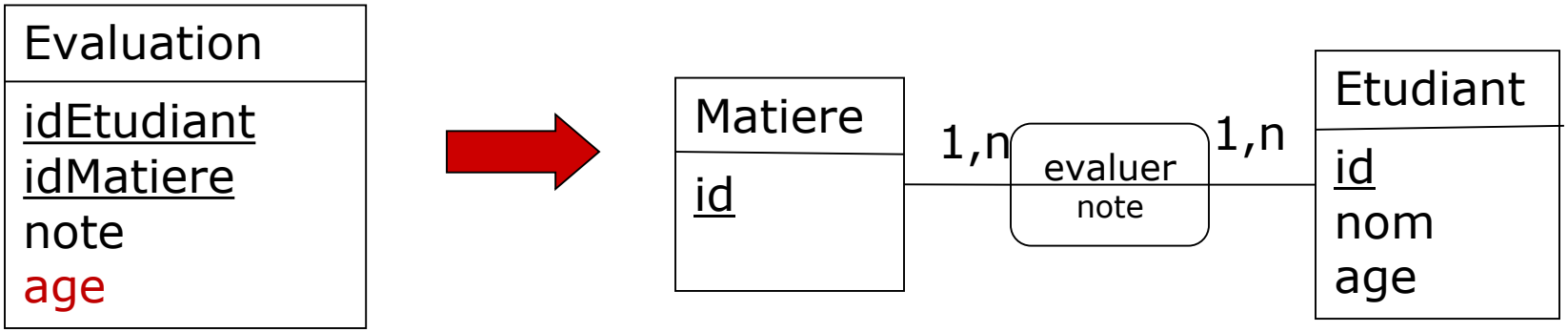
- But
  - Rendre le modèle le « plus propre possible ».
  - Limiter la redondance de données.

  => Augmenter la fiabilité et diminuer la maintenance.
- Les règles de normalisation
  - Semi-formelles (intuitives) sur le MCD
  - Formelles sur le MLD (séance prochaine)

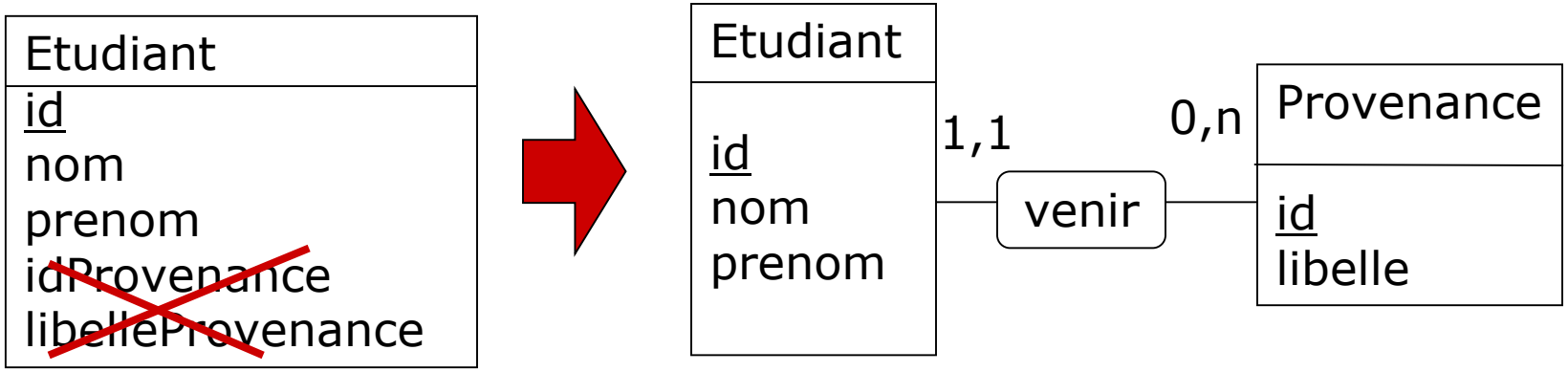
Propriété élémentaire



Intégrité référentielle



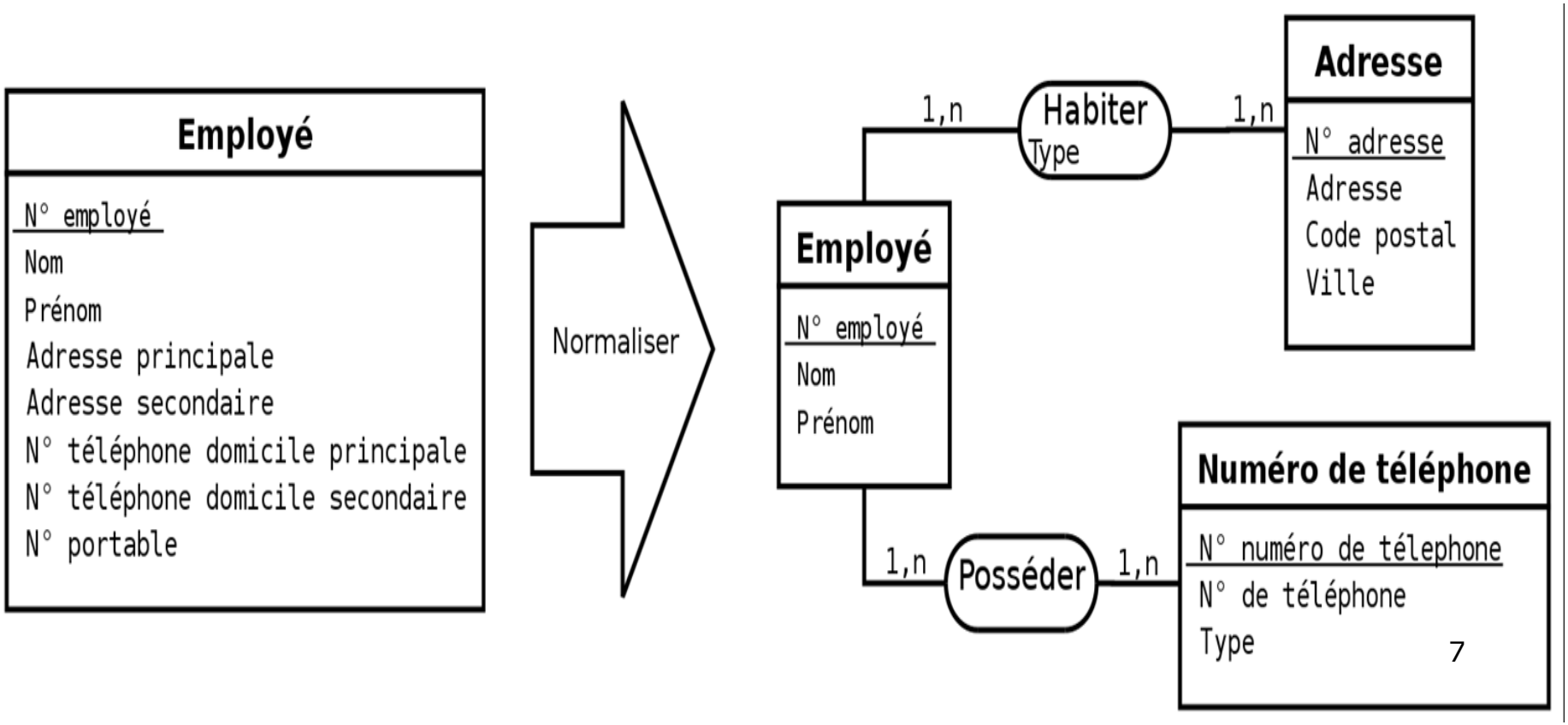
Intégrité référentielle



libelleProvenance ne dépend pas directement de la clé mais d'un autre attribut

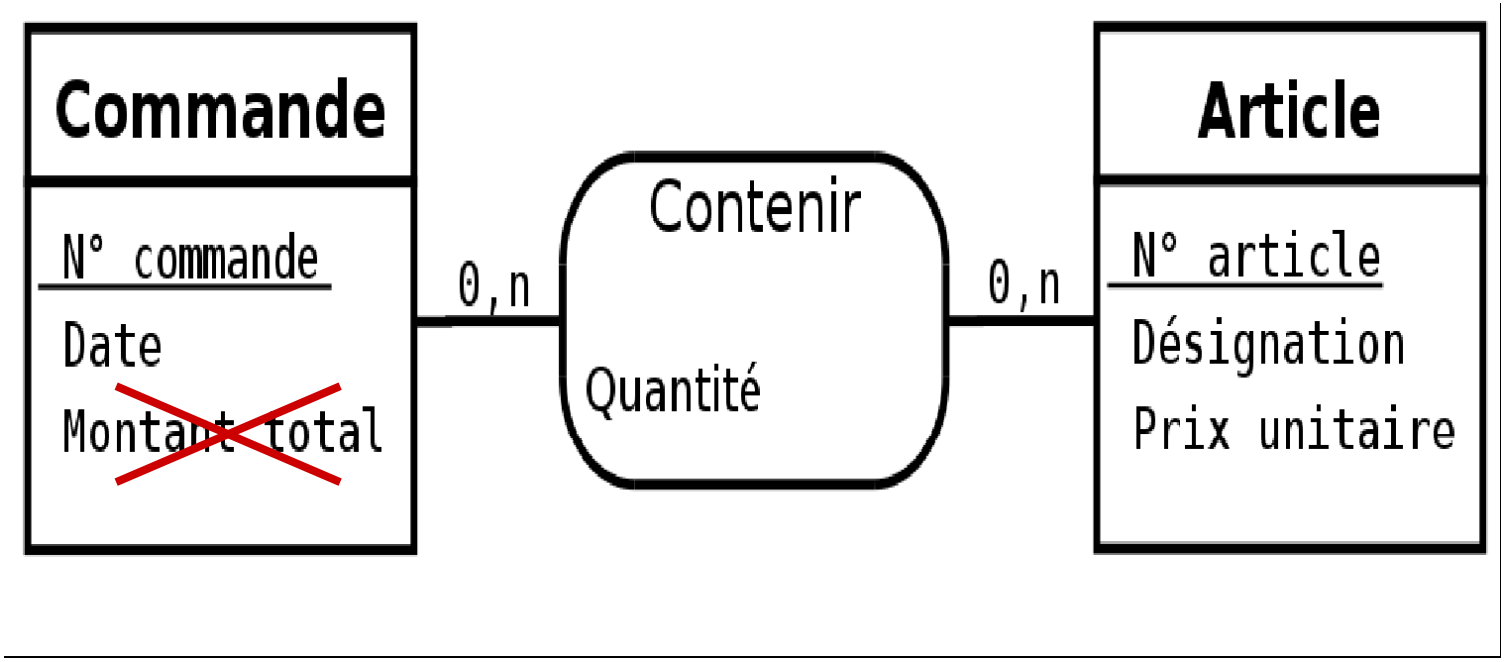
Attribut multiple

- Remplacer un attribut multiple en association et entité supplémentaire



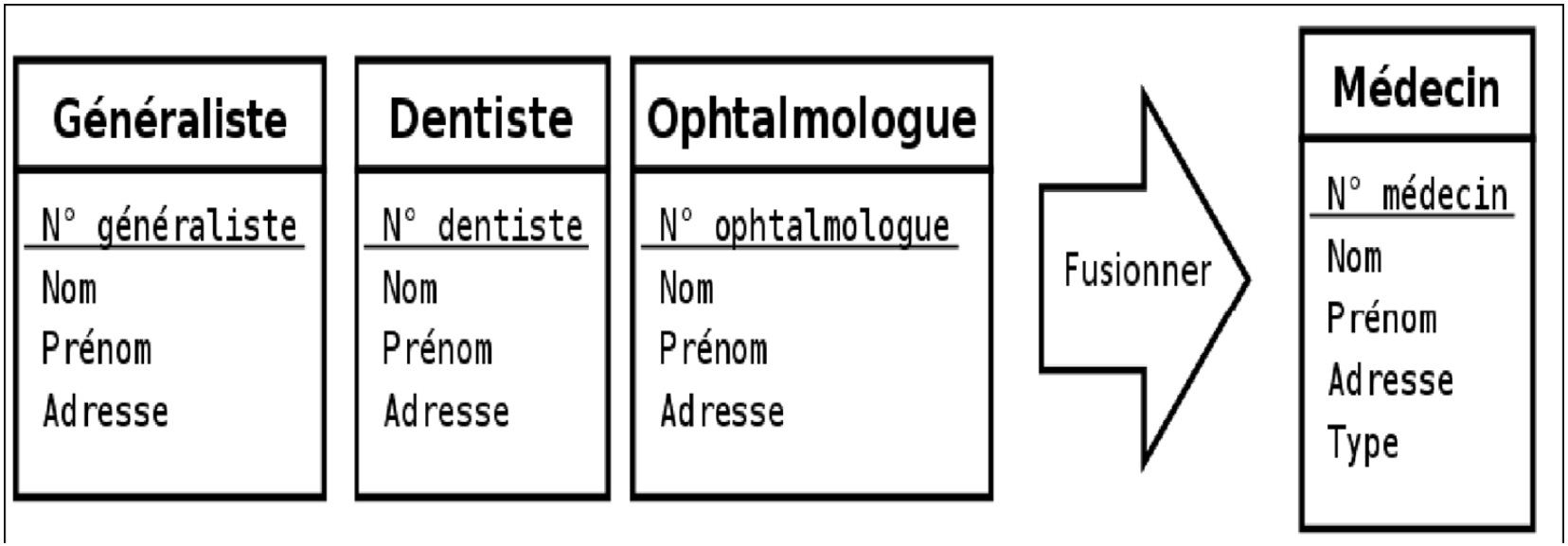
Attribut dérivé

- Il est déconseillé d'ajouter un attribut dérivé d'autres attributs

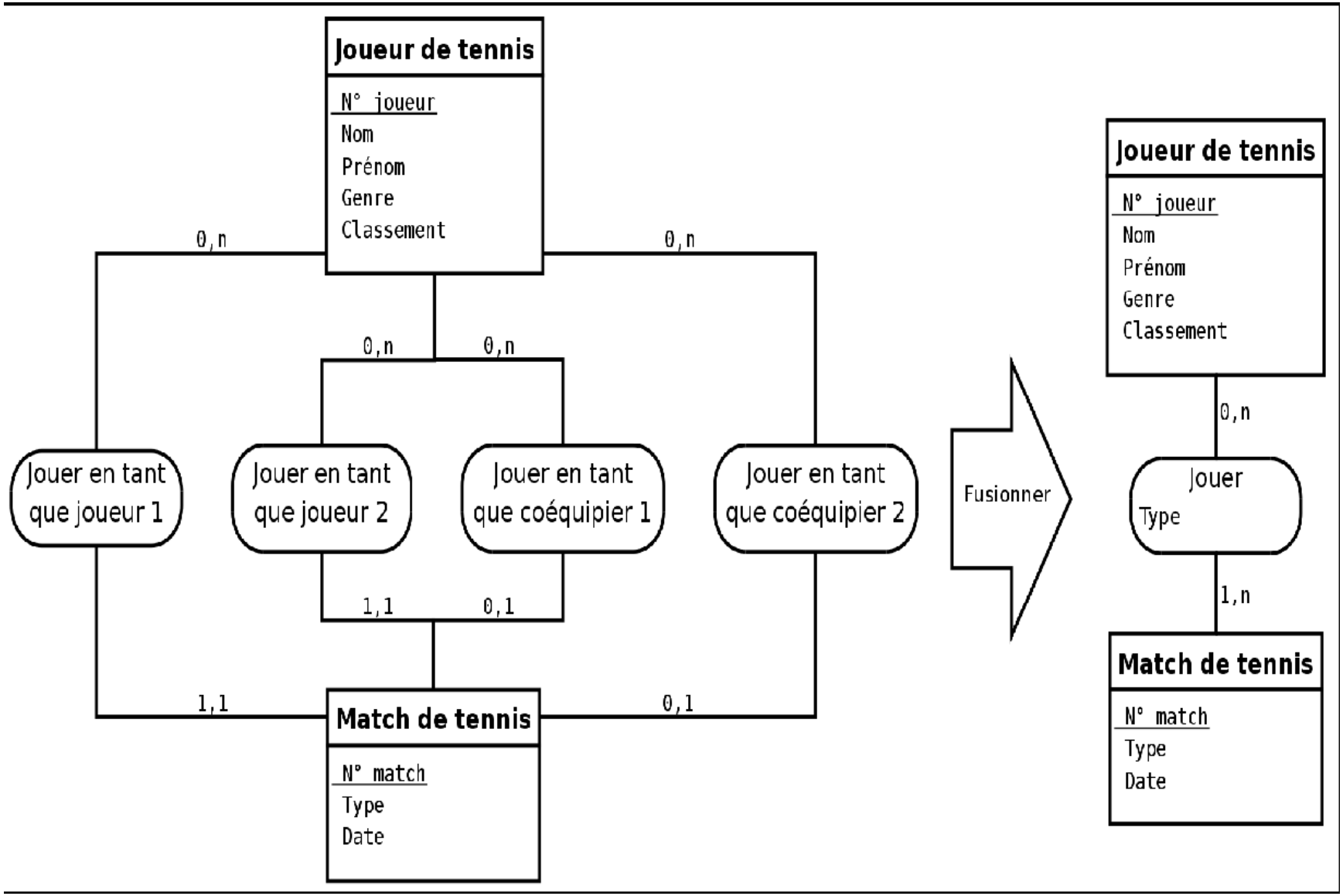


Fusion/suppression d'entités/associations (1)

- Il faut factoriser les entités/associations quand c'est possible

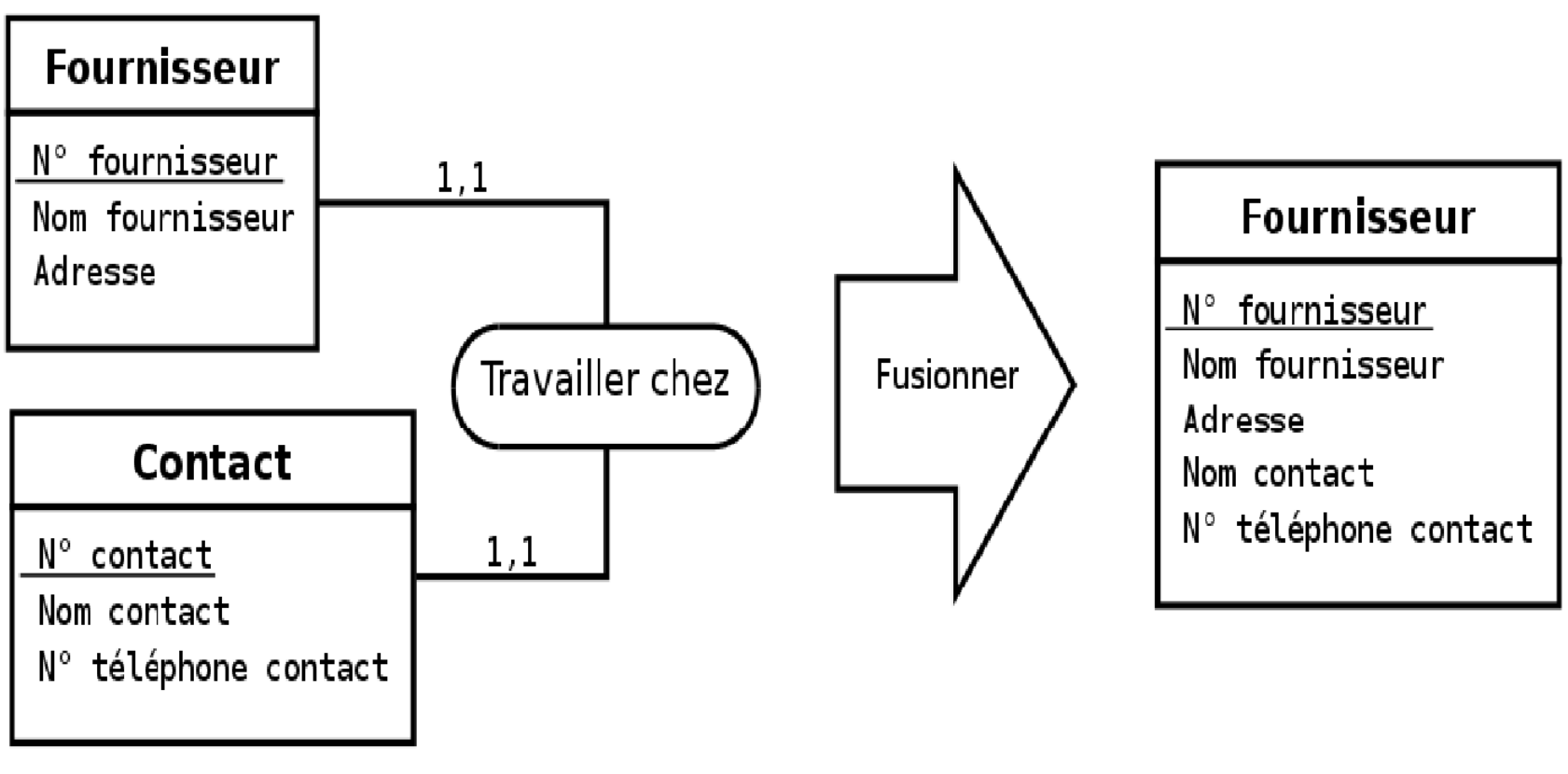


Fusion/suppression d'entités/associations (2)



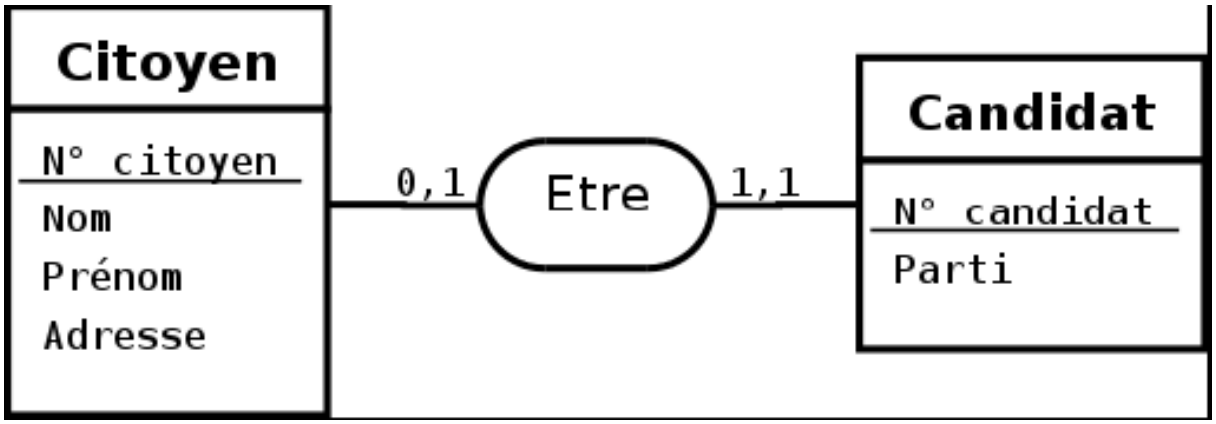
Fusion/suppression d'entités/associations (3)

- Lorsque les cardinalités d'une association sont toutes 1,1 c'est que l'association n'a pas lieu d'être

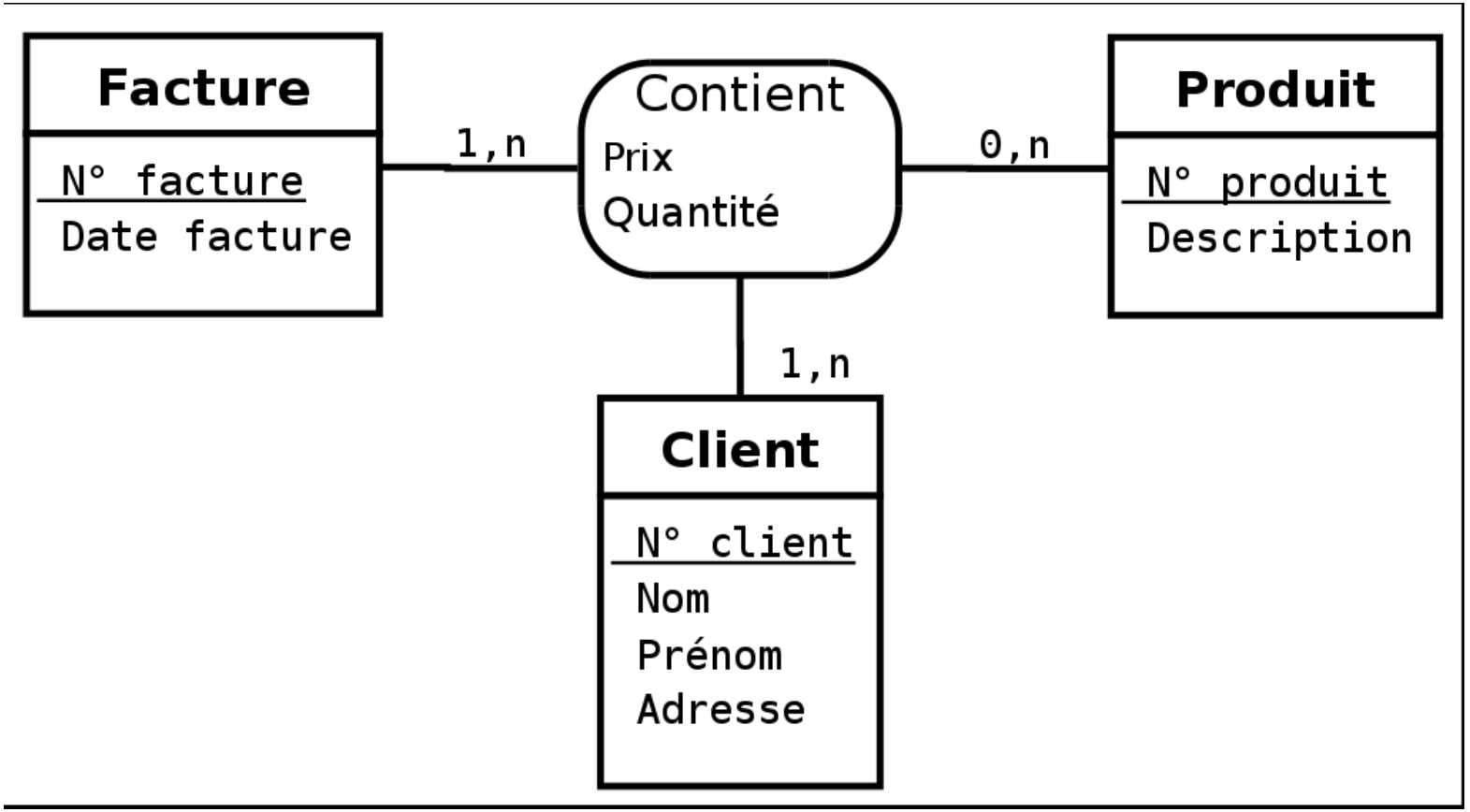


Fusion/suppression d'entités/associations (4)

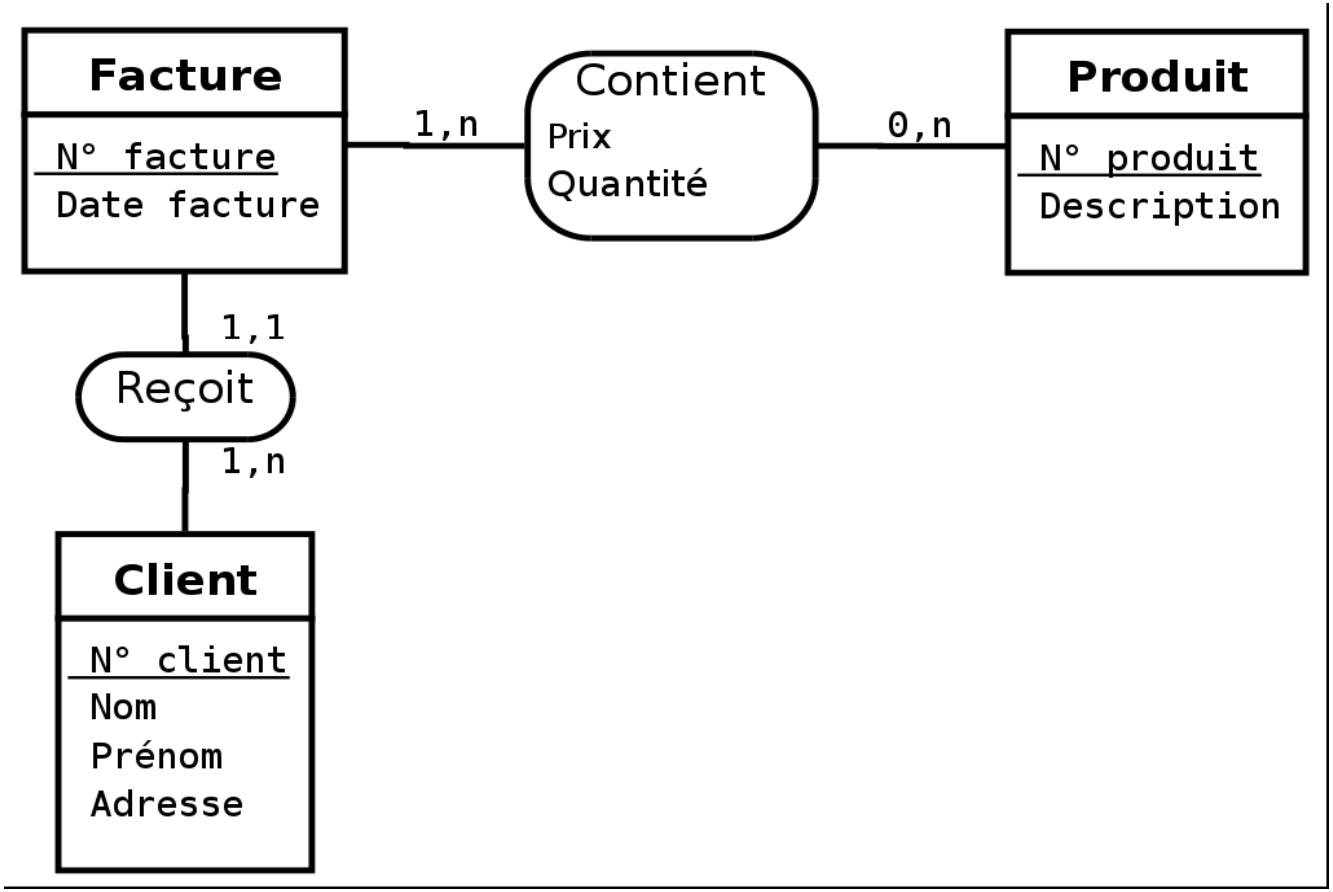
- Mais pas cela …



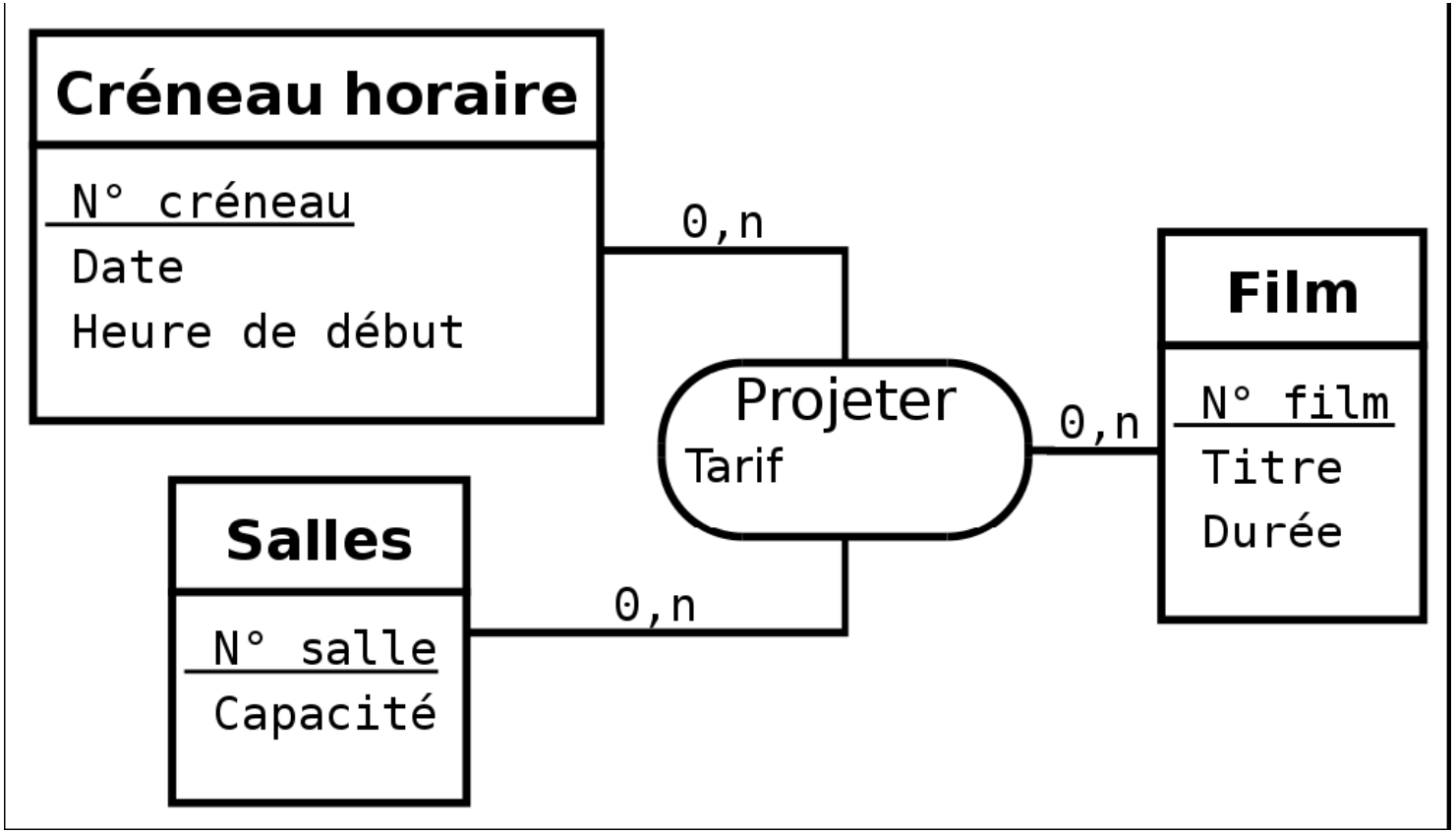
Association n-aire inappropriée



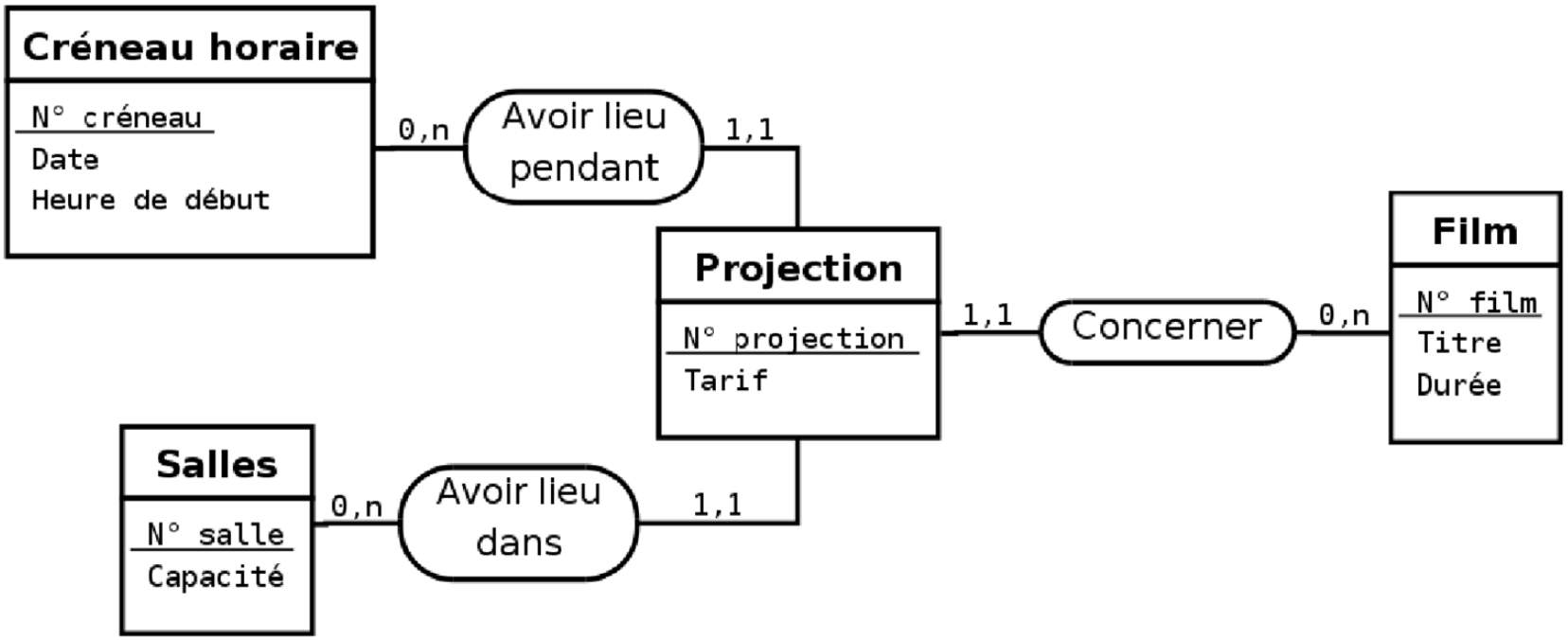
Corrigée en 2 associations binaires



Décomposition d'une association n-aire



Transformation en 1 entité et 3 associations binaires

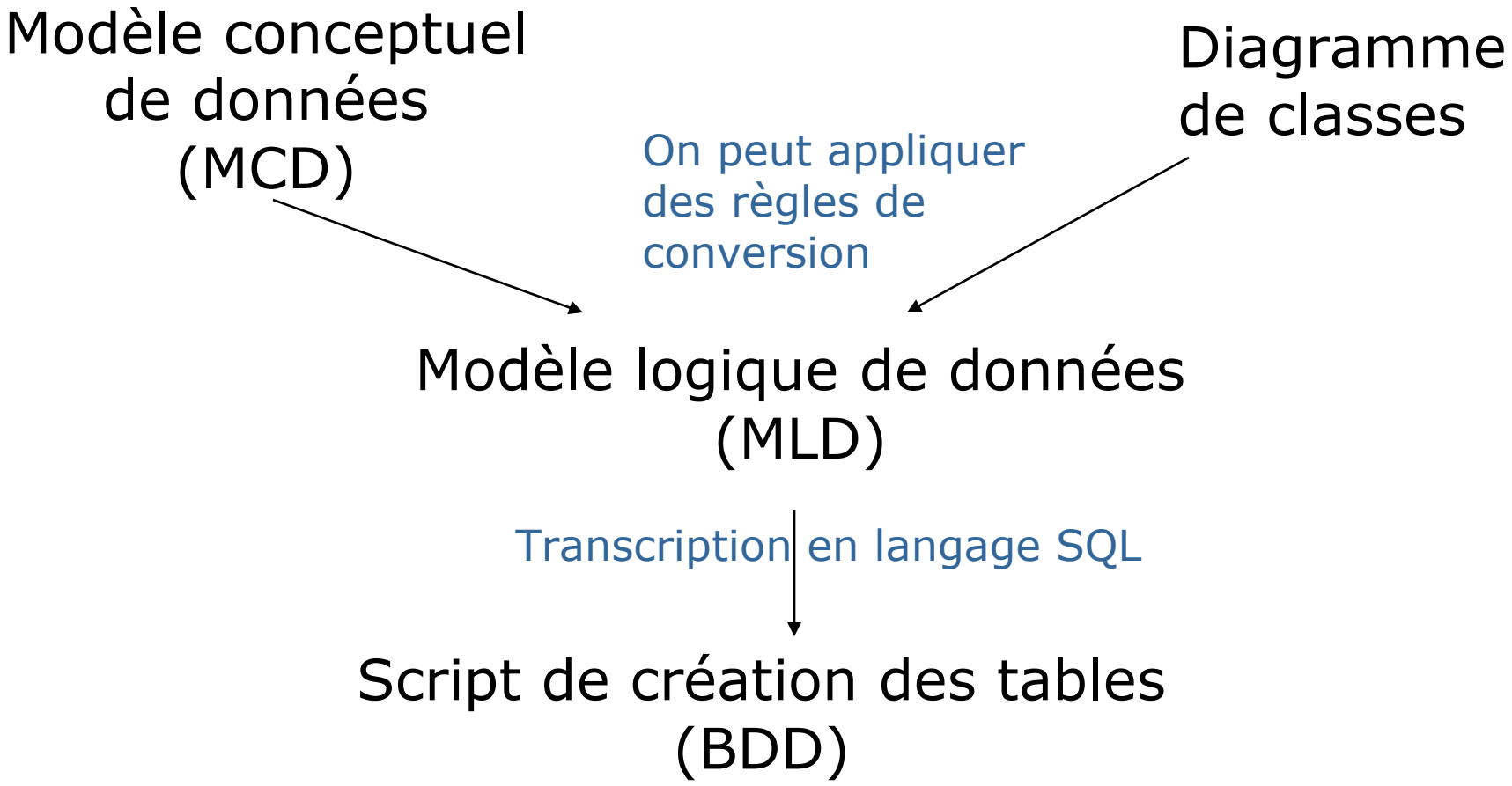


Extension du formalisme entités – associations

- But : de nouveaux opérateurs pour enrichir la modélisation et symboliser des situations plus complexes

- Introduction des concepts :
  - généralisation/spécialisation
    - représentation plus proche du monde réel
  - de nouvelles contraintes
    - inclusion, exclusion, ou exclusif, …

Du MCD à la base de données



Modèle logique de données (MLD)

- Reprend le contenu du MCD mais précise la structure et l'organisation des données telle qu'elles pourront être implémentées :
  - fichiers simples
  - **modèle relationnel**, …

- Modèle relationnel :
  - Schéma de relation :

    Entite(<u>id</u>, propriete1, propriete2, *#aid)*
  - Relation : <u>clé primaire</u>, attributs, *#clé étrangère*

Définitions

- **Clé candidate**
  - Une clé candidate d'une relation est un ensemble minimal des attributs de la relation dont les valeurs identifient à coup sûr une occurrence.
- **Clé primaire**
  - La clé primaire d'une relation est une de ses clés candidates.
  - Pour signaler la clé primaire, ses attributs sont généralement soulignés.
- **Clé étrangère**
  - Une clé étrangère dans une relation est formée d'un ou plusieurs attributs qui constituent une clé primaire dans une autre relation.

Règles de passage de MCD à MLD

- Règle 1 : Toute entité est représentée par une relation. Chaque attribut de l'entité devient un attribut de la relation. L'identifiant est conservé en tant que clé de la relation.

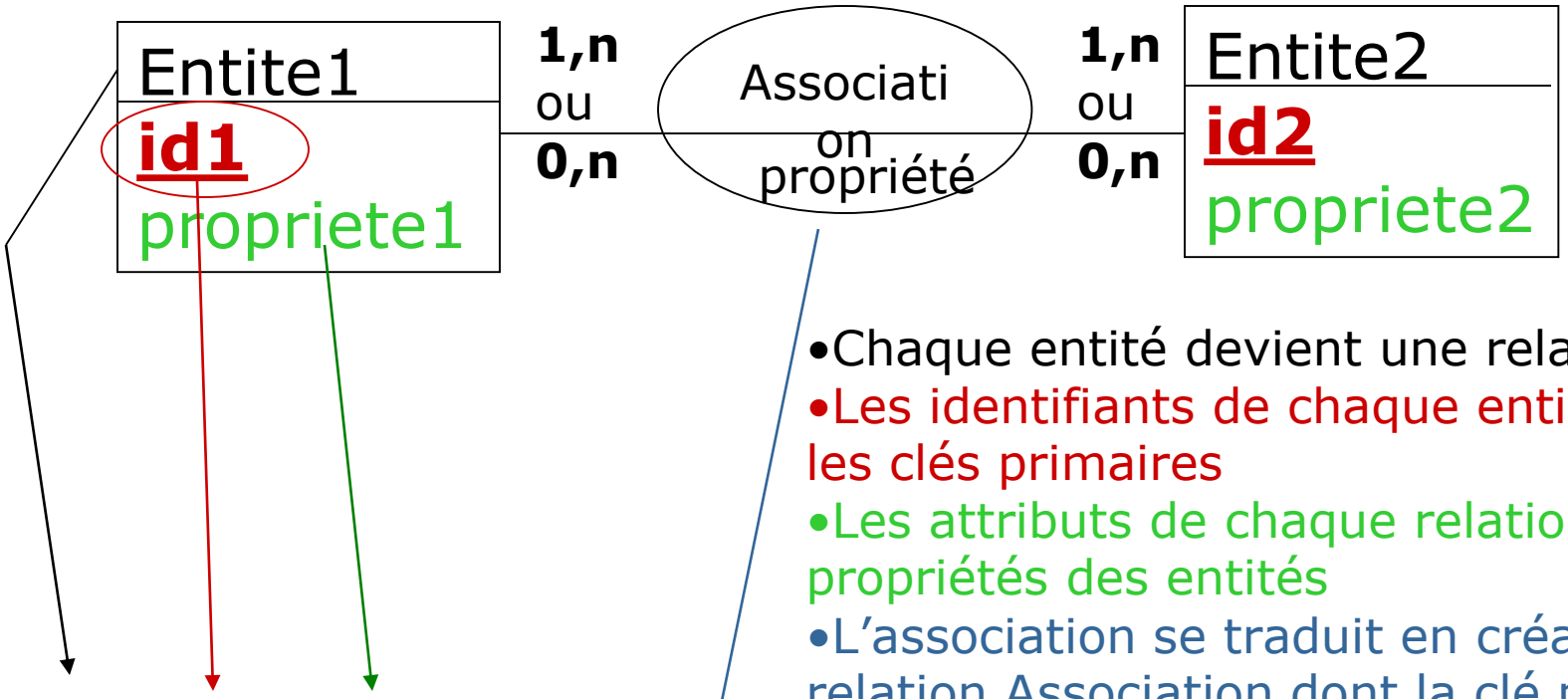
- Règle 2 : Toute association qui associe plus de deux entités (ternaire et au-delà) est représentée par une relation.

Règles de passage de MCD à MLD

- Règle 3 : Toute association binaire dont les cardinalités maximales sont n de chaque côté est une relation (relation dont les attributs sont les attributs clefs des entités qu'elle relie ainsi que les éventuels attributs propres à l'association).

- Règle 4 : Une association de type père - fils, cardinalité maximum à n d'un côté et à 1 de l'autre, n'est pas représentée par une relation. On indique les attributs clefs de l'entité père (côté (.,n)) dans le fils (côté (.,1)).

Relation matricielle ou n..n



- Chaque entité devient une relation
- Les identifiants de chaque entité sont les clés primaires
- Les attributs de chaque relation sont les propriétés des entités
- L'association se traduit en créant une relation Association dont la clé primaire est formée des identifiants de chaque entité participant à l'association- elles sont des clés étrangères

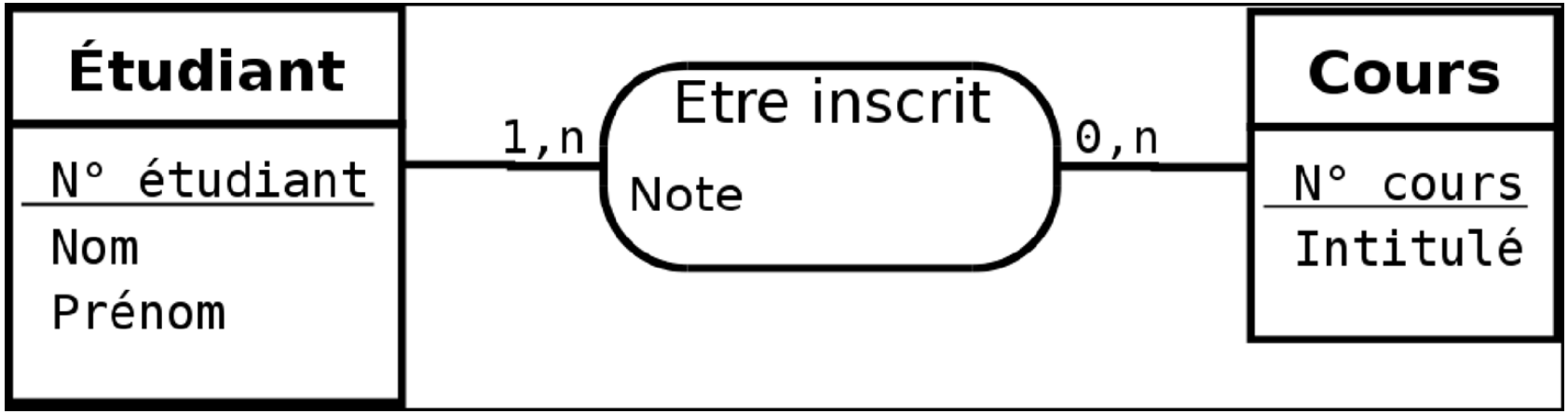
Entite1(id1, propriete1)

Entite2(id2, propriete2)

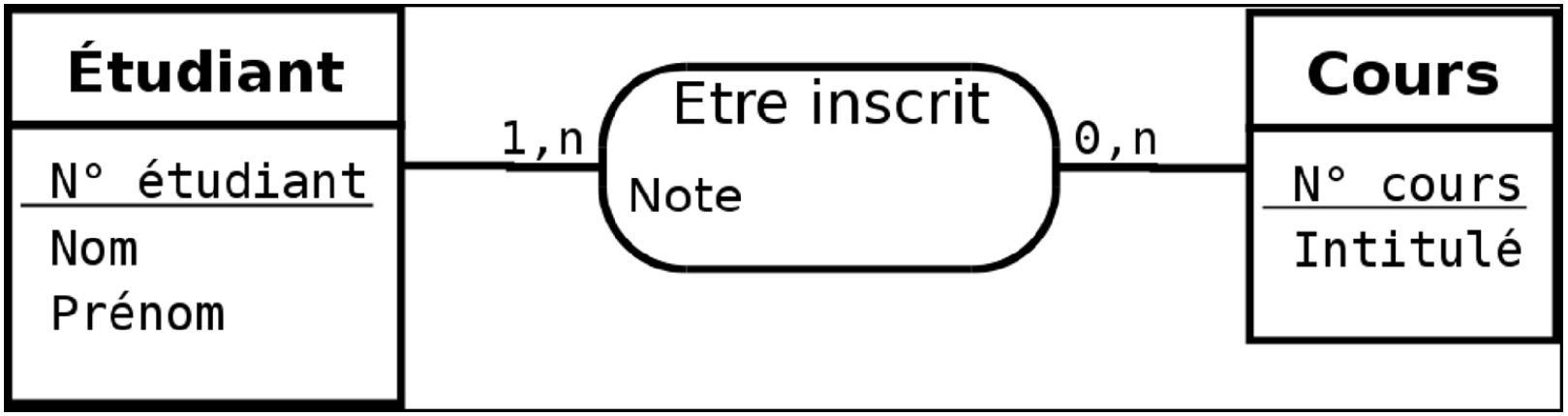
Association(*#id1,#id2*, propriete)

Exemple 1

- Un étudiant s'est inscrit en 1 ou plusieurs cours
- Un cours peut être inscrit par zéro ou plusieurs étudiants
- Pour chaque cours, un étudiant est évalué par une note

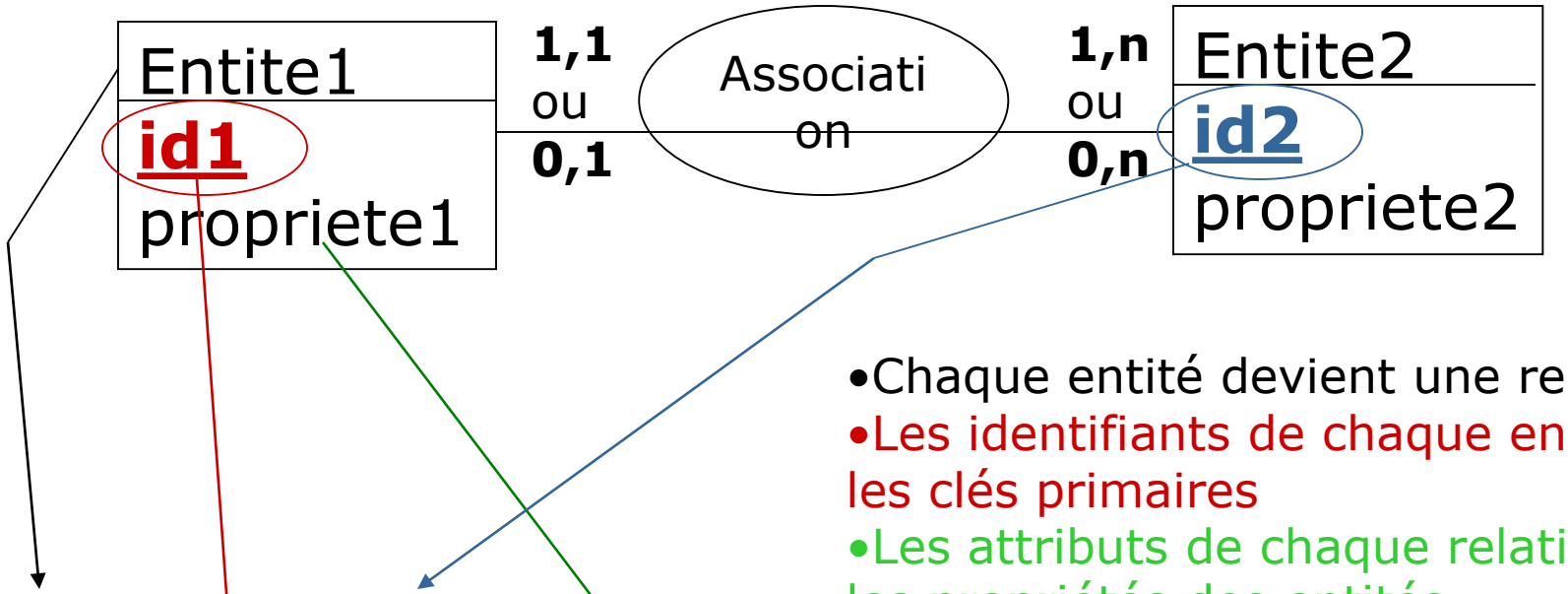


Exemple 1 : Traduction du MCD vers MLD



- Etudiant(numEtudiant, nom, prenom)
- Cours(numCours, intitule)
- Inscription(*#numEtudiant, #numCours*, note)

Relation père-fils



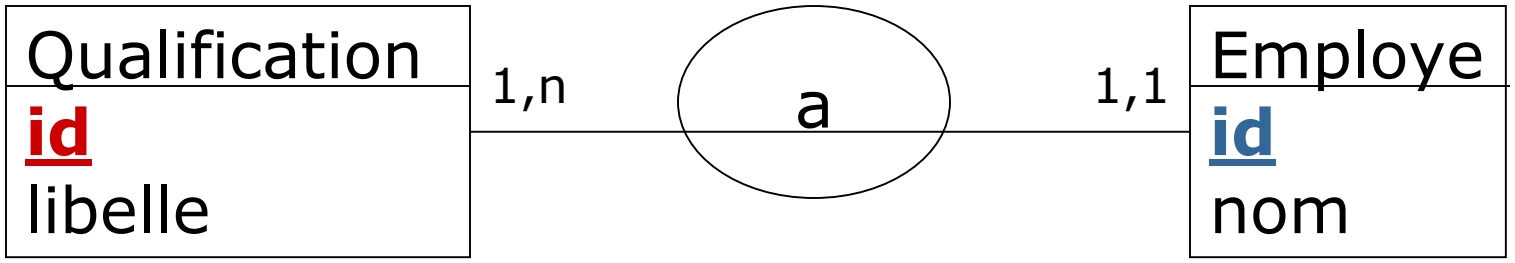
Entite1(id1, #*id2*, propriete1)

Entite2(id2, propriete2)

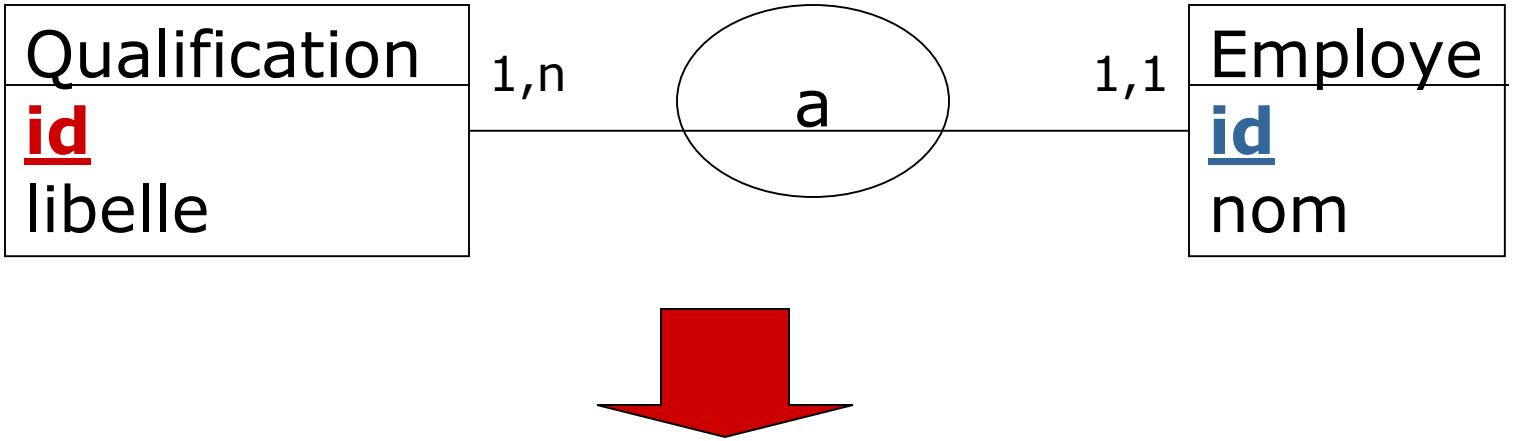
- Chaque entité devient une relation
- Les identifiants de chaque entité sont les clés primaires
- Les attributs de chaque relation sont les propriétés des entités
- L'association se traduit en rajoutant dans la relation Entite1 l'attribut correspondant à l'identifiant de Entite2, id2 devient donc clé étrangère

Exemple 2

- Un employé a une qualification
- Une qualification peut correspondre à plusieurs employés



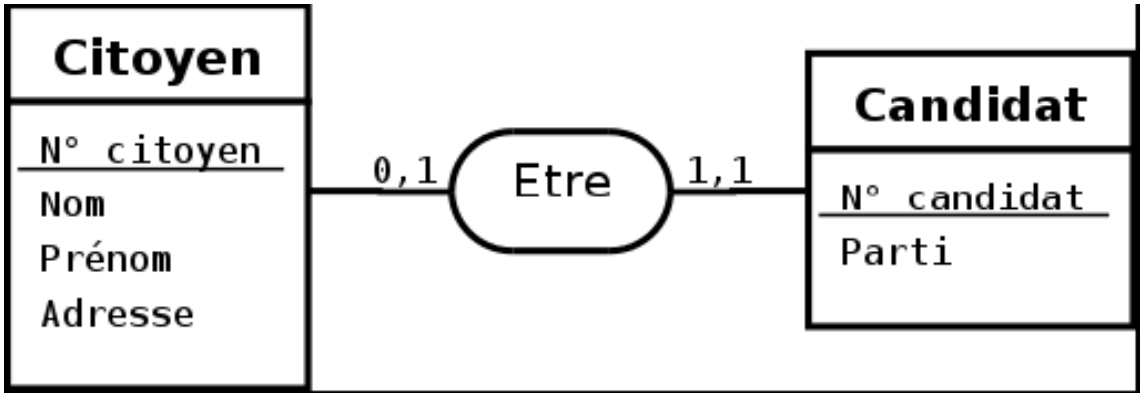
Exemple 2 : Traduction du MCD vers MLD



- Employe(id, nom, *#id_qualif*)
- Qualification(id, libelle)

Cas particuliers (1)

- Association (0,1) – (1,1) : on déclare une clé étrangère du côté (1,1) pour éviter des champs NULL



Citoyen(<u>numCitoyen</u>, nom, prenom, adresse)
Candidat(<u>numCandidat</u>, parti, *#numCitoyen*)

- Association (0,1) – (0,1) : on a le choix entre les deux relations pour placer la clé étrangère

Cas particuliers (2)

- Association réflexive : mêmes règles qu'une association binaire (avec renommage de la clé étrangère dans le cas d'une association père-fils)

- Exceptions : on pourra créer une relation supplémentaire pour une association (0,1)-(1,n) suivant la proportion d'occurrences de celle-ci, pour éviter une quantité trop importe de champs NULL

Base de données

Normalisation

Normalisation

- Dans la phase de conception sur le modèle entités-associations et dans le modèle relationnel pour éviter :
  - la redondance de données
  - la perte de données
  - les incohérences
  - l'effondrement de performance des traitements

- Dépendance fonctionnelle !

Relation en extension (table)

- Soit R la relation définie sur les ensembles A1, A2, …, An

$$R \subseteq A1 \times A2 \times \ldots \times An$$

- L'extension de R est l'ensemble de tuples de la relation à une instance donnée :

A1	A2	…	An
a_{11}	a_{21}	…	a_{n2}
a_{11}	a_{21}	…	a_{nk}
…	…	…	…
a_{12}	a_{2j}	…	a_{nk}

(a_{11}, a_{21}, …, a_{nk}) est un n-uplet (ou tuple)

Relation en intension (MLD)

- Schéma relationnel :

R(A1:D1, A2:D2,…, An:Dn)

avec :
- R: nom de la relation
- A1, A2, …, An : nom des attributs de la relation
- D1, D2,…, Dn : domaine des attributs
- et des contraintes d'intégrités éventuelles

Exemple

- Soit la relation de schéma

Voiture(<u>nv</u>, marque, type, puissance, couleur) :

NV	MARQUE	TYPE	PUISSANCE	COULEUR
AF234567	Renault	RME8	5	rouge
XN228756	Citroën	4ZX2	7	grise
DE998031	Peugeot	P307B	7	verte
ANA22201	Opel	OCA6	5	noir
FJ000075	Ford	FFI9	5	jaune
SI141481	Citroën	4ZX2	7	verte
BI151465	Renault	RME9	9	jaune

Définition

- **Dépendance fonctionnelle (DF):**
  - Soit R(A1,A2,… An) un schéma de relation, et X et Y des sous-ensembles de {A1, A2,… An}
  - On dit que X **détermine** Y ou que Y **dépend fonctionnellement** de X si, et seulement si, des valeurs identiques de X impliquent des valeurs identiques de Y
  - On le note : $X \rightarrow Y$

Exemple

- Ville(<u>numville</u>, nom, code_postal, population)

  - numville → nom
  - numville → code_postal
  - numville → population
  - code_postal → nom
  - nom → code_postal

Propriétés

- **Dépendances triviales (réflexivité)**
  - $A \rightarrow A$
  - $A, B \rightarrow A$
- Augmentation
  - $A \rightarrow B \Rightarrow A, C \rightarrow B$
- Transitivité
  - $A \rightarrow B$ et $B \rightarrow C \Rightarrow A \rightarrow C$

Définition

- **Dépendance fonctionnelle élémentaire**
  - Une DF élémentaire est une DF de la forme $X \rightarrow A$
    - où A est un attribut unique n'appartenant pas à X
    - et où il n'existe pas X' inclus au sens strict dans X (i.e. $X' \subset X$) tel que $X' \rightarrow A$

Exemple

- Soit la relation de schéma
  Reduction(<u>cru, client</u>, type, remise) :

CR	TYP	CLIENT	REMISE
U CHENAS	E A	C1	3%
MEDOC	A	C2	5%
JULIENAS	B	C	4
CHENAS	A	C2	4 %
		1	%

- **cru, client → remise** est élémentaire :
  - cru ~~→~~ remise : (CHENAS, A, C1, 3%) et (CHENAS, A, C2, 4%)
  - client ~~→~~ remise : (CHENAS, A, C1, 3%) et (JULIENAS, B, C1, 4%)

Exemple

- Soit la relation de schéma
  Reduction(<u>cru, client</u>, type, remise) :

CR	TYP	CLIENT	REMISE
U CHENAS	E A	C1	3%
MEDOC	A	C2	5%
JULIENAS	B	C	4
CHENAS	A	C2	4 % %

- **cru, client → type**
  - est une DF
  - mais n'est pas élémentaire : cru → type

Définitions

- **Dépendance fonctionnelle directe**
  - Une DF $X \rightarrow A$ est une DF directe s'il n'existe aucun attribut B tel que l'on puisse avoir $X \rightarrow B$ et $B \rightarrow A$

Exemple

- On reprend la relation de schéma

Voiture(<u>nv</u>, marque, type, puissance, couleur) :

NV	MARQUE	TYPE	PUISSANCE	COULEUR
AF234567	Renault	RME8	5	rouge
XN228756	Citroën	4ZX2	7	grise
DE998031	Peugeot	P307B	7	verte
ANA22201	Opel	OCA6	5	noir
FJ000075	Ford	FFI9	5	jaune
SI141481	Citroën	4ZX2	7	verte
BI151465	Renault	RME9	9	jaune

Exemple

- On reprend la relation de schéma

Voiture(<u>nv</u>, marque, type, puissance, couleur) :

NV	MARQUE	TYPE	PUISSANCE	COULEUR
AF234567	Renault	RME8	5	rouge
XN228756	Citroën	4ZX2	7	grise
DE998031	Peugeot	P307B	7	verte
ANA22201	Opel	OCA6	5	noir
FJ000075	Ford	FFI9	5	jaune
SI141481	Citroën	4ZX2	7	verte
BI151465	Renault	RME9	9	jaune

- On a NV → MARQUE, NV → TYPE, NV → PUISSANCE, NV → COULEUR
- mais aussi TYPE → MARQUE et TYPE → PUISSANCE
- donc NV → MARQUE, NV → PUISSANCE ne sont pas directes
- et NV → TYPE, NV → COULEUR sont directes