

# INTRODUCTION MERISE

# **INTRODUCTION À MERISE**

# MERISE

- Pour créer une base de données, nous utilisons couramment la méthode **Merise**.
- C'est une méthode d'analyse, de conception et de gestion de projet.
- Les Systèmes de Gestion de Base de Données (SGBD) sont apparus dans les années 60. Ils permettent de créer des bases de données, de mettre à jour leurs données, d'y rechercher des informations particulières et de les visualiser.
- Il existe de nombreux SGBD : PostgreSQL, MySQL, Oracle, Sybase, Access, MySQL,...  
Les SGBD utilisent généralement le langage SQL.

# MERISE

- Quand nous construisons directement les tables d'une base de données dans un logiciel de gestion des bases de données (PostgreSQL, Oracle, SQL Server, DB2, base, Access, MySQL, ...), nous sommes exposés à deux types de problème :
  - Nous ne savons pas toujours dans quelle table placer certaines colonnes (par exemple, l'adresse de livraison se met dans la table des clients ou dans la table des commandes ?).
  - Nous avons du mal à prévoir les tables de jonction intermédiaires (par exemple, la table des interprétations qui est indispensable entre les tables des films et la table des acteurs).
- Il est donc nécessaire de recourir à une étape préliminaire de conception.

# MERISE

- MERISE est une méthode française née dans les années 70, développée initialement par **Hubert Tardieu**. Elle fut ensuite mise en avant dans les années 80, à la demande du ministère de l'Industrie qui souhaitait une méthode de conception des SI.
- MERISE est donc une méthode d'analyse et de conception des SI basée sur le principe de la séparation des données et des traitements. Elle possède un certain nombre de **modèles** (ou **schémas**) qui sont répartis sur trois niveaux :
  - Le niveau **conceptuel** ;
  - Le niveau **logique ou organisationnel** ;
  - Le niveau **physique**.

# **MODÈLE CONCEPTUEL DE DONNÉES - MCD**

Avant de réfléchir au schéma relationnel d'une application, il est bon de modéliser la problématique à traiter d'un point de vue conceptuel et indépendamment du logiciel utilisé.

# MCD

- Il s'agit de l'élaboration du **modèle conceptuel des données** (MCD) qui est une représentation graphique et structurée des informations mémorisées par un SI. Le MCD est basé sur deux notions principales : les **entités** et les **associations**, d'où sa seconde appellation : le **schéma Entité/Association**.
- L'élaboration du MCD passe par les étapes suivantes :
  - La mise en place de **règles de gestion** (si celles-ci ne vous sont pas données) ;
  - L'élaboration du **dictionnaire des données** ;
  - La recherche des **dépendances fonctionnelles** entre ces données ;
  - L'élaboration du MCD (création des **entités** puis des **associations** puis ajout des **cardinalités**).

# DICTIONNAIRE DE DONNÉE

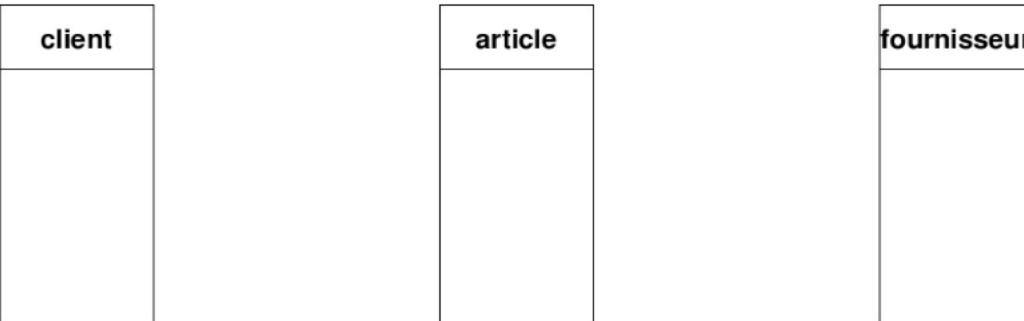
- Il est utile, voir indispensable, d'effectuer un inventaire brut des données qui seront stockées et/ou calculées par la base de données.
- Il faut donc différencier deux types de données, les données stockées et les données qui peuvent être déduites ou calculés à partir des données stockées.
- Ce document sera présenté sous forme de tableau de plusieurs colonnes, avec dans la première colonne le nom de la donnée, dans la seconde si cette donnée est calculée ou stockée et dans la troisième un commentaire descriptif de la donnée.

# DICTIONNAIRE DE DONNÉE

- C'est une étape intermédiaire qui peut avoir son importance, surtout si vous êtes plusieurs à travailler sur une même base de données, d'un volume important.
- Le dictionnaire des données est un document qui regroupe toutes les données que vous aurez à conserver dans votre base (et qui figureront donc dans le MCD). Pour chaque donnée, il indique :
- Le **code mnémonique** : il s'agit d'un libellé désignant une donnée (par exemple «*titre\_l*» pour le titre d'un livre) ;
- La **désignation** : il s'agit d'une mention décrivant ce à quoi la donnée correspond (par exemple «*titre du livre*») ;
- Le **type de donnée** :
  - **A ou Alphabétique** : lorsque la donnée est uniquement composée de caractères alphabétiques (de 'A' à 'Z' et de 'a' à 'z'),
  - **N ou Numérique** : lorsque la donnée est composée uniquement de nombres (entiers ou réels),
  - **AN ou Alphanumérique** : lorsque la donnée peut être composée à la fois de caractères alphabétiques et numériques,
  - **Date** : lorsque la donnée est une date (au format AAAA-MM-JJ),
  - **Booléen** : Vrai ou Faux ;
- La **taille** : elle s'exprime en nombre de caractères ou de chiffres. Dans le cas d'une date au format AAAA-JJ-MM, on compte également le nombre de caractères, soit 10 caractères. Pour ce qui est du type booléen, nul besoin de préciser la taille (ceci dépend de l'implémentation du SGBDR) ;
- Parfois des **remarques** ou **observations** complémentaires (par exemple si une donnée est strictement supérieure à 0, etc.).

# SCHÉMA ENTITÉ-ASSOCIATION

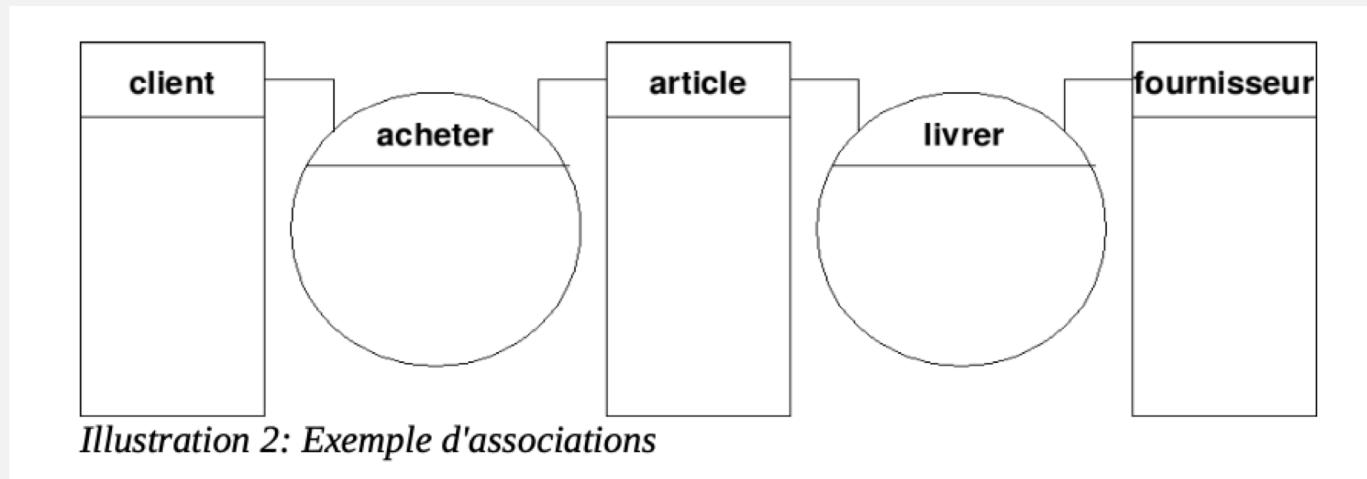
- Une **entité** est une population d'individus n'ayant que des caractéristiques comparables. Par exemple, dans une entreprise qui achète et vend des produits, nous avons l'entité *client*, l'entité *article* et l'entité *fournisseurs*



*Illustration 1: Exemple d'entités*

# SCHÉMA ENTITÉ-ASSOCIATION

- Une **association** est un lien entre plusieurs entités. Dans notre exemple illustration2, l'association *acheter* fait le lien entre les entités articles et clients, tandis que l'association *livrer* fait le lien entre les entités articles et fournisseurs. Nous avons souvent intérêt à distinguer deux verbes pour décrire l'association, l'un dans un sens l'autre dans l'autre sens. . .



# SCHÉMA ENTITÉ-ASSOCIATION

- Un **attribut** est une propriété d'une entité ou d'une association. Toujours dans notre exemple, le *prix unitaire* est un attribut de l'entité article, le *nom du client* est un attribut de l'entité client. La *quantité commandée* est un attribut de l'association acheter, la *date de livraison* est un attribut de l'association livrer.

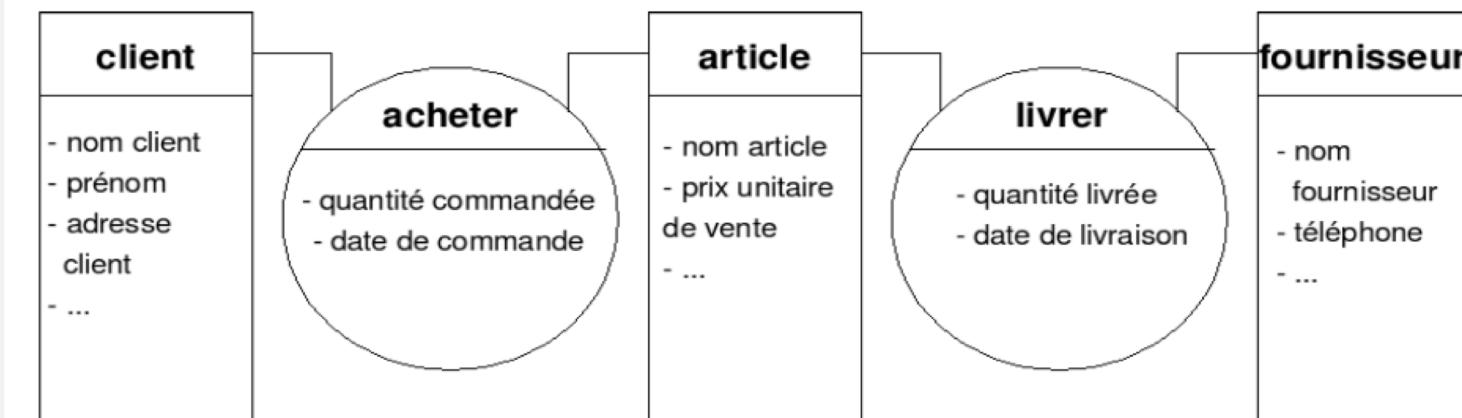
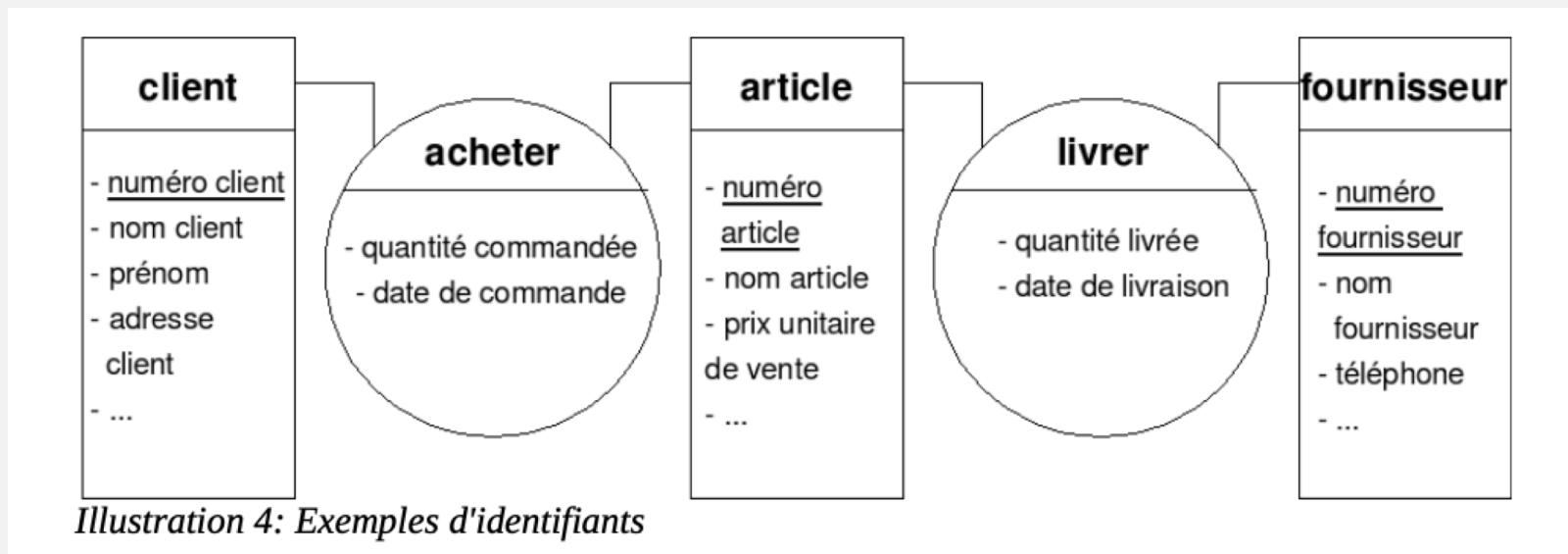


Illustration 3: Exemples d'attributs

# SCHÉMA ENTITÉ-ASSOCIATION

- Chaque individu d'une entité doit être identifiable de manière unique. C'est pourquoi les entités doivent posséder un attribut sans doublon : **l'identifiant**. Le *numéro de client* est l'identifiant de l'entité client.



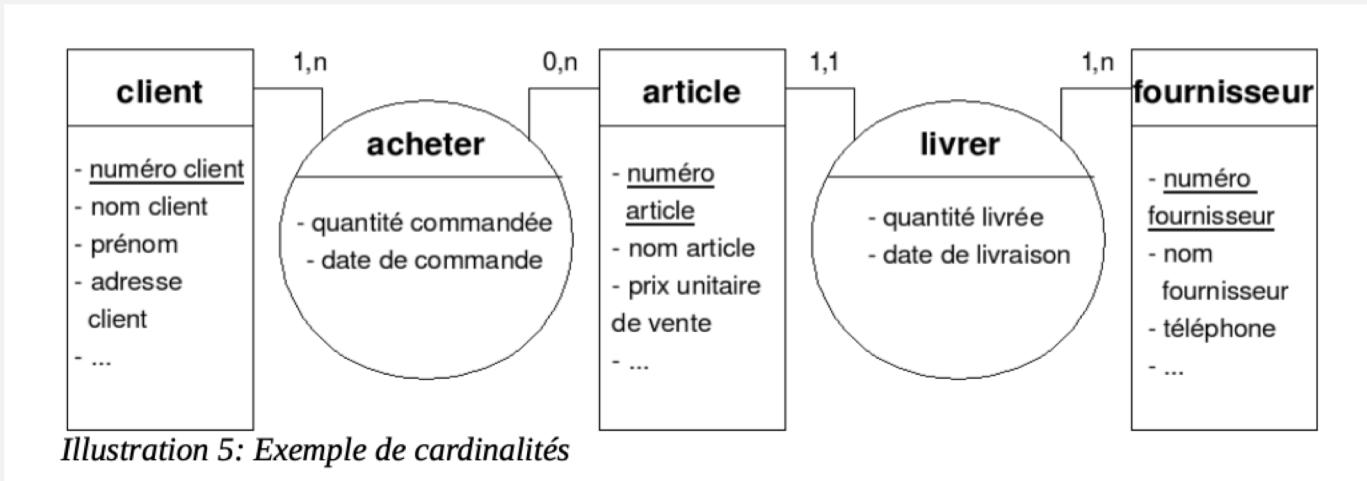
# SCHÉMA ENTITÉ-ASSOCIATION

## Remarques :

- Un attribut ne doit pas figurer dans deux entités ou associations différentes (donc il faut spécialiser l'attribut *nom* en *nom du client*, *nom du produit* et *nom du fournisseur*) ;
- Une entité possède au moins un attribut (son identifiant) ;
- Une association peut ne pas posséder d'attribut.

# SCHÉMA ENTITÉ-ASSOCIATION

- La **cardinalité** d'un lien entre une entité et une association est le minimum et le maximum de fois qu'un individu de l'entité peut être concerné par l'association. Un client a au moins acheté un article et peut acheter n articles (n étant indéterminé), tandis qu'un article peut avoir été acheté entre 0 et n fois (même si ce n'est pas le même n) et n'est livré que par 1 fournisseur.



# CAS PARTICULIERS

## Exemple d'association binaire réflexive :

- Un employé est dirigé par un employé (sauf le directeur général) et un employé peut diriger plusieurs employés. Pour distinguer les deux liaisons, nous utilisons la notion de rôle avec l'ajout d'étiquettes (ici *supérieur*, *personnel*).

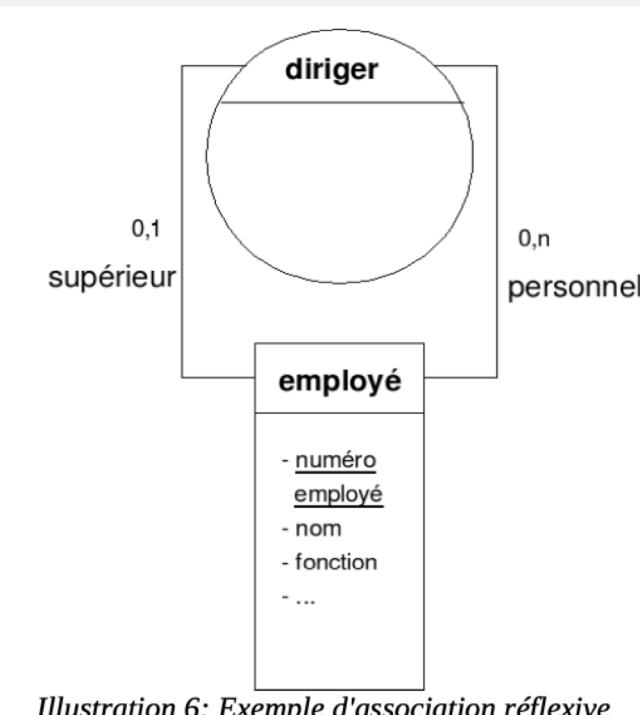
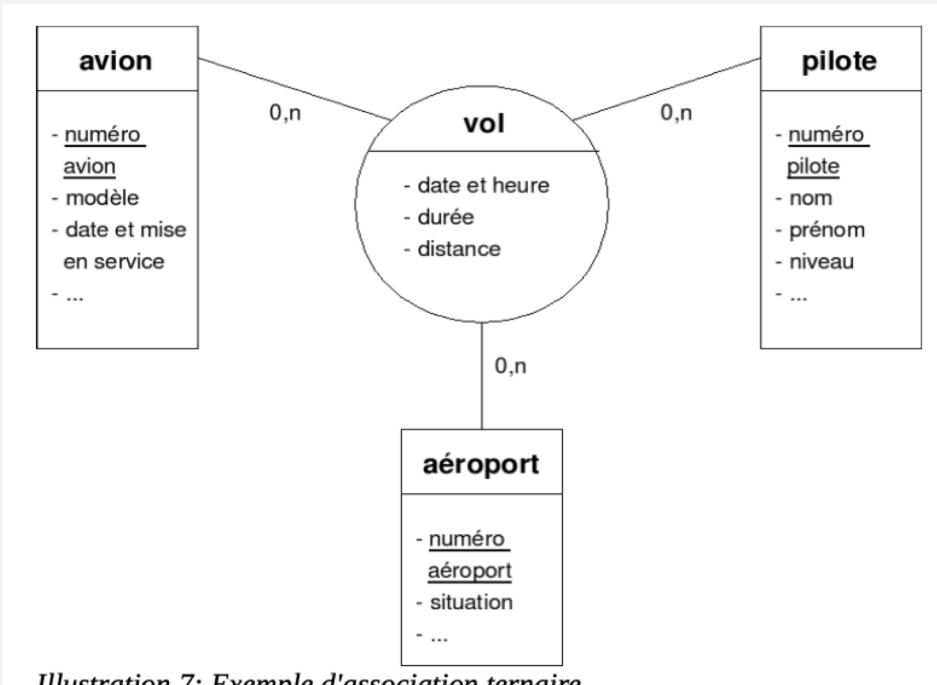


Illustration 6: Exemple d'association réflexive

# CAS PARTICULIERS

## Exemple d'association entre trois entités (association ternaire) :

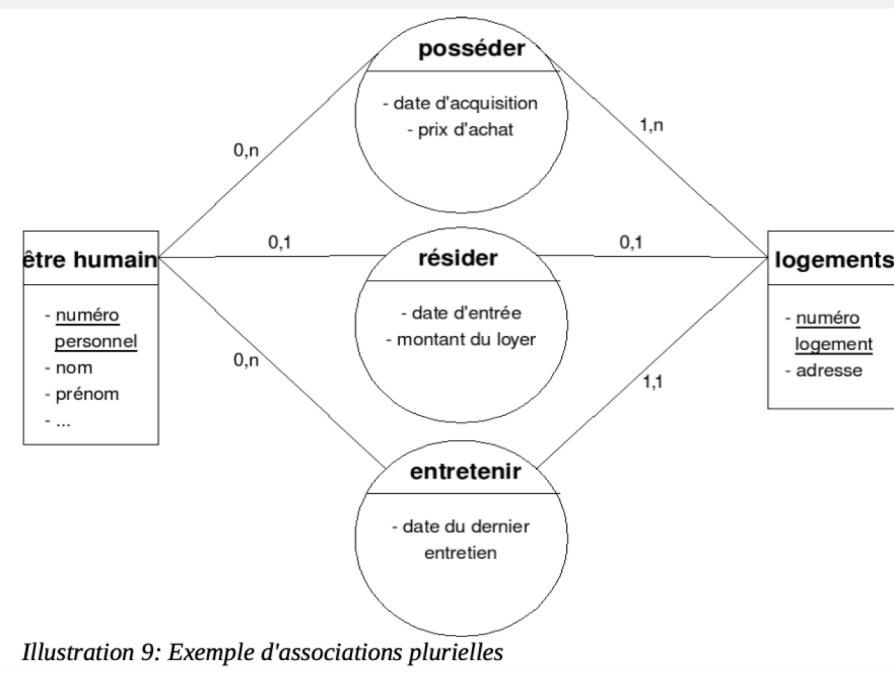
- Dans ce cas, un avion, un pilote ou un aéroport peuvent être utilisés 0 ou plusieurs fois par l'ensemble des vols (d'où les cardinalités).



# CAS PARTICULIERS

Exemple de plusieurs associations entre deux mêmes entités :

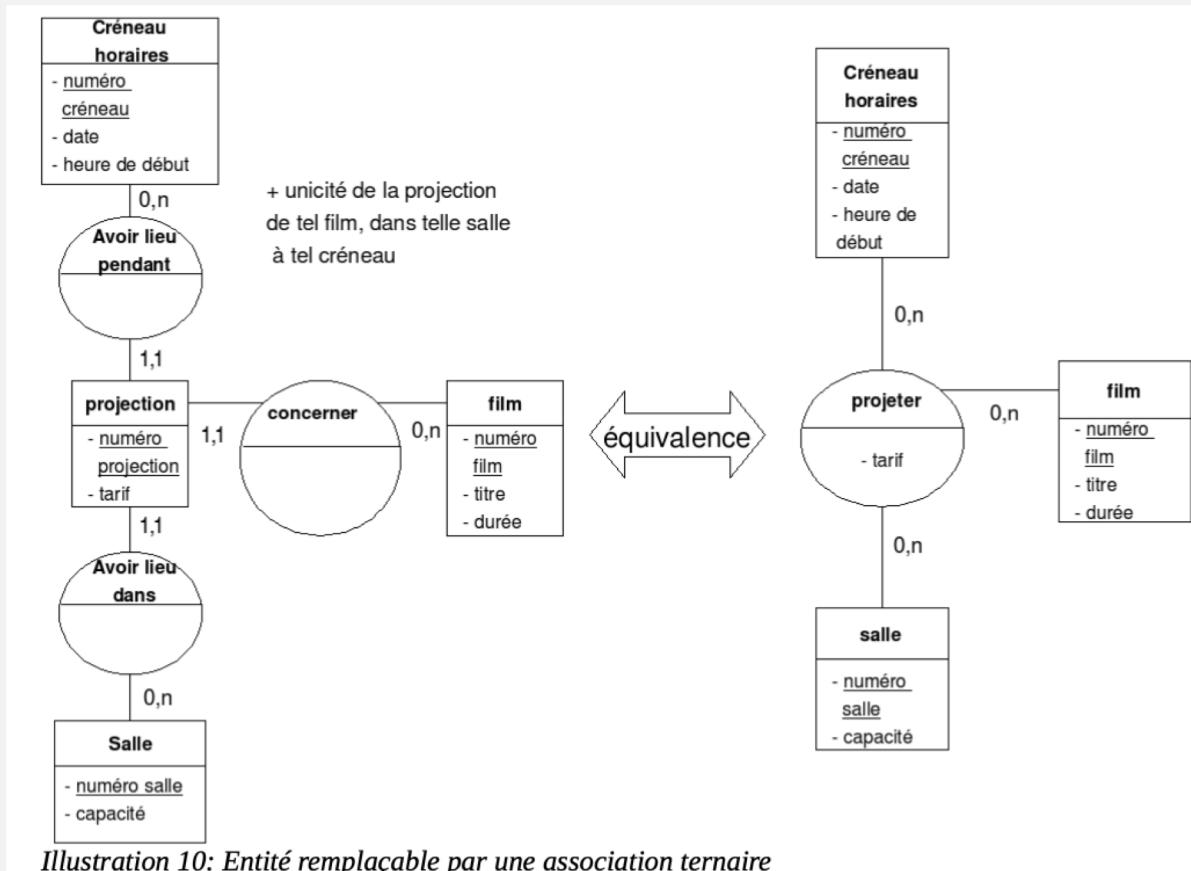
- Dans ce cas, un être humain peut être propriétaire, locataire et/ou chargé de l'entretien. Nous supposons qu'un être humain ne loue qu'un logement au maximum, qu'un logement n'est occupé que par une personne au maximum et qu'un logement est entretenu par une et une seule personne.



# RÈGLES DE NORMALISATION

## I- Normalisation des entités

- Toutes les entités qui sont remplaçables par une association doivent être remplacées

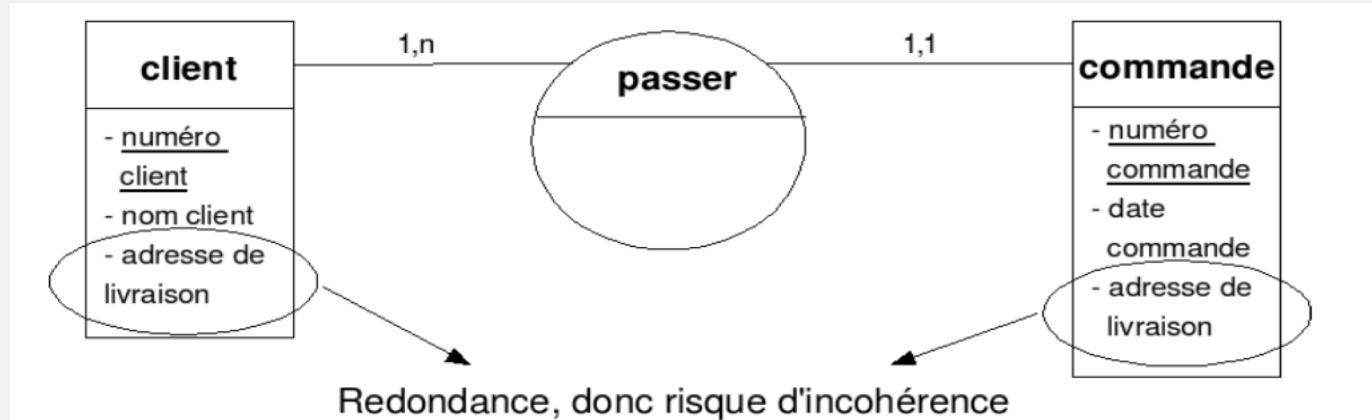
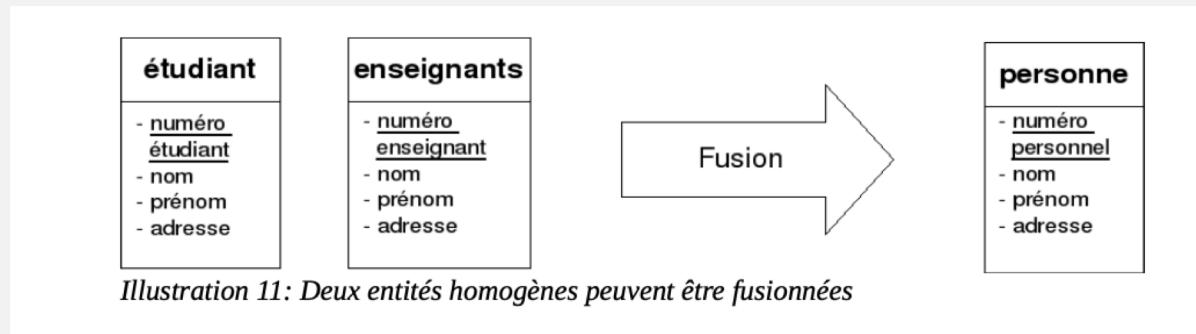


# RÈGLES DE NORMALISATION

## 2 - Normalisation des noms

- Le nom d'une entités d'une association ou d'un attribut doit être unique.

**Remarque :** lorsqu'il reste plusieurs fois le même nom, c'est parfois symptomatique d'une modélisation qui n'est pas terminée ou le signe d'une redondance.



# RÈGLES DE NORMALISATION

## 3 - Normalisation des attributs

- Il faut remplacer les attributs en plusieurs exemplaires en une association supplémentaire de cardinalités maximales n et il ne faut pas ajouter d'attribut calculable à partir d'autres attributs.
- En effet, d'une part, les attributs en plusieurs exemplaires posent des problèmes d'évolutivité du modèle (comment faire si un employé a deux adresses secondaires ?) et d'autre part, les attributs calculables induisent un risque d'incohérence entre les valeurs des attributs de base et celles des attributs calculés.

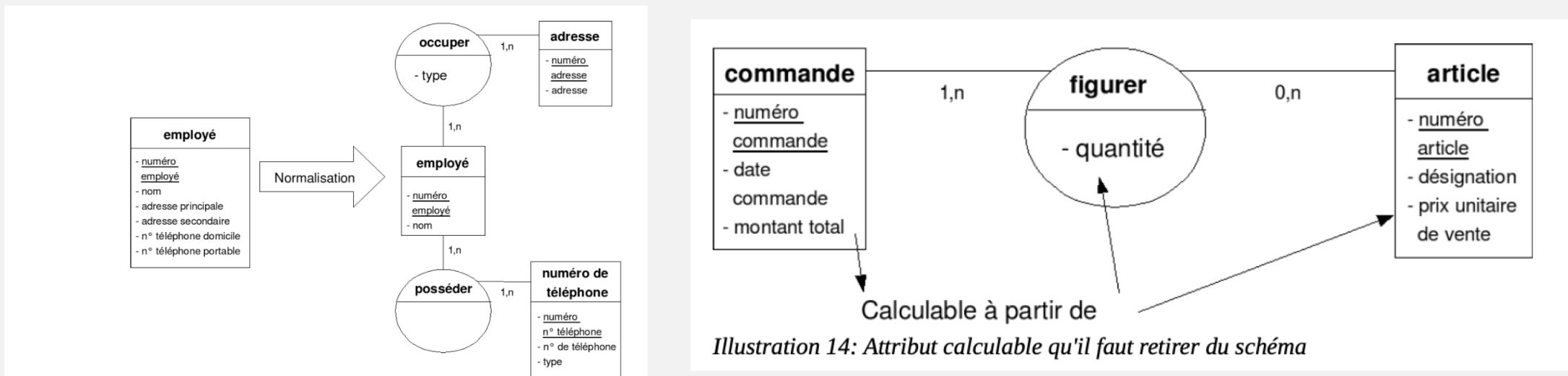


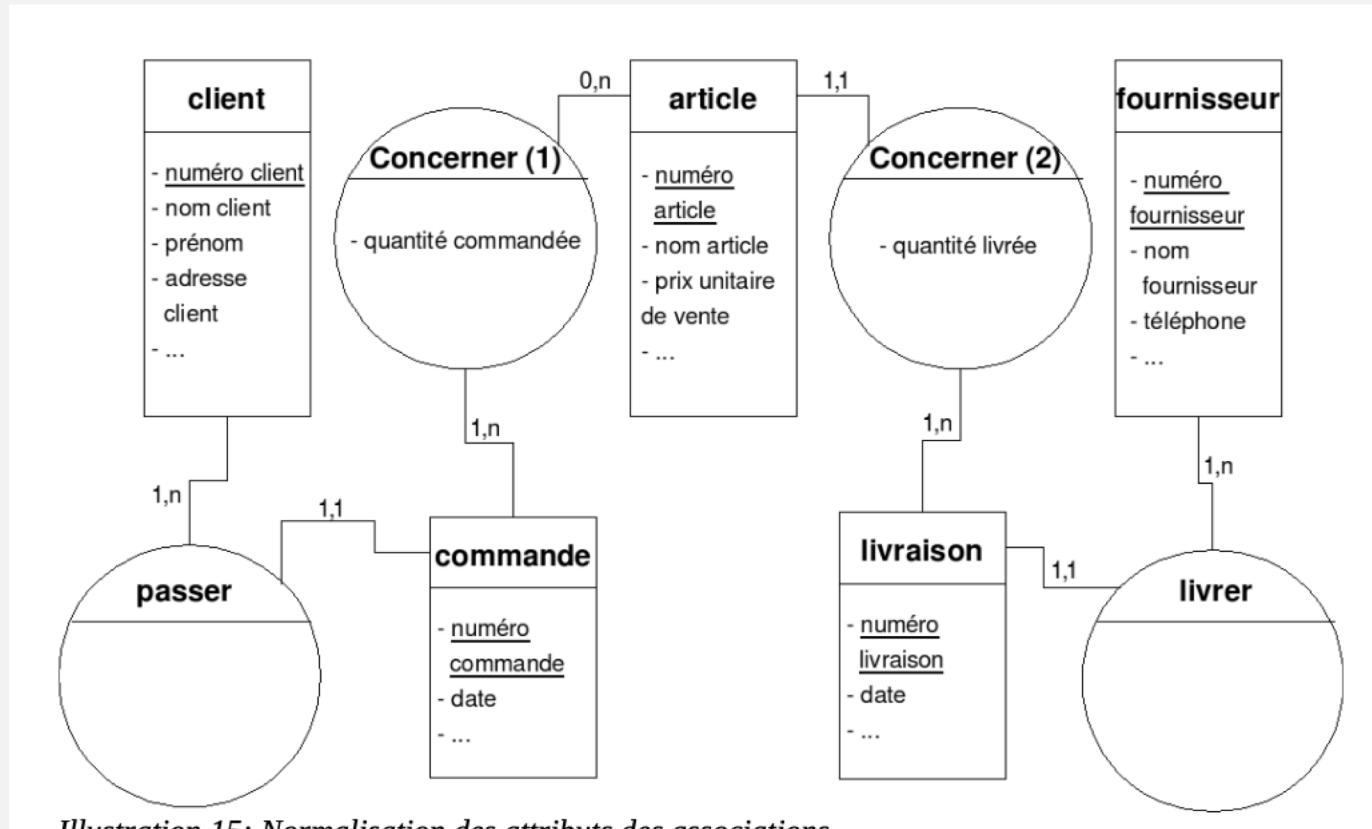
Illustration 14: Attribut calculable qu'il faut retirer du schéma

# RÈGLES DE NORMALISATION

## 4 - Normalisation des attributs des associations

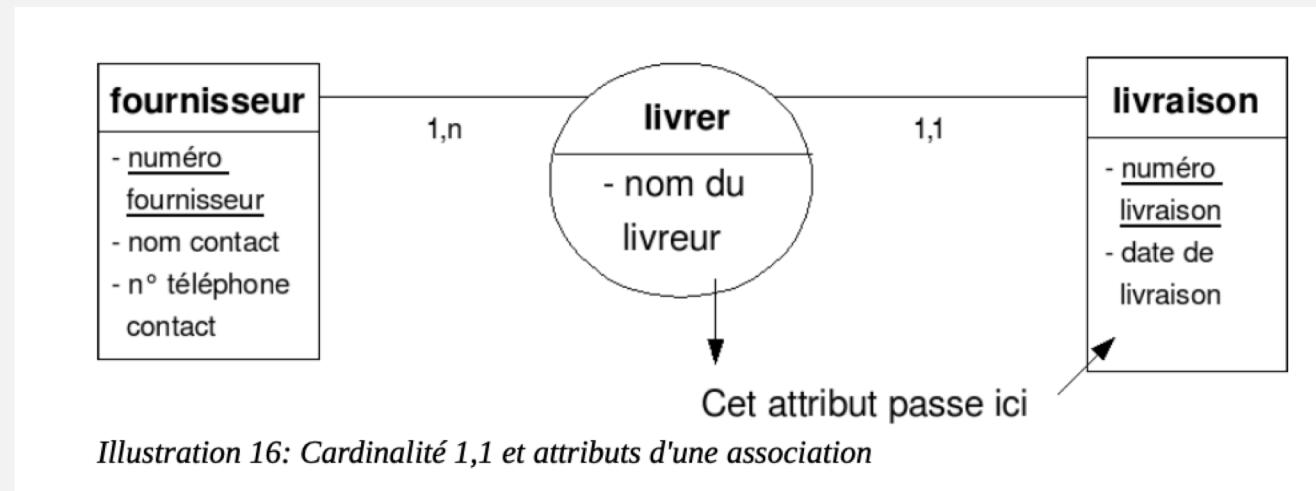
- Les attributs d'une association doivent dépendre directement des identifiants de toutes les entités en association.
- Par exemple, la quantité commandée dépend à la fois du numéro de client et du numéro d'article, par contre la date de commande non. En effet la date dépend du client et non d'un des articles commandés. Il faut donc faire une entité commandes à part, idem pour les livraisons.

# RÈGLES DE NORMALISATION



# RÈGLES DE NORMALISATION

Autre conséquence de la normalisation des attributs des associations : une entité avec une cardinalité de 1,1 ou 0,1 aspire les attributs de l'association



# RÈGLES DE NORMALISATION

Il faut éliminer les associations fantômes et les associations redondantes ou en plusieurs exemplaires.

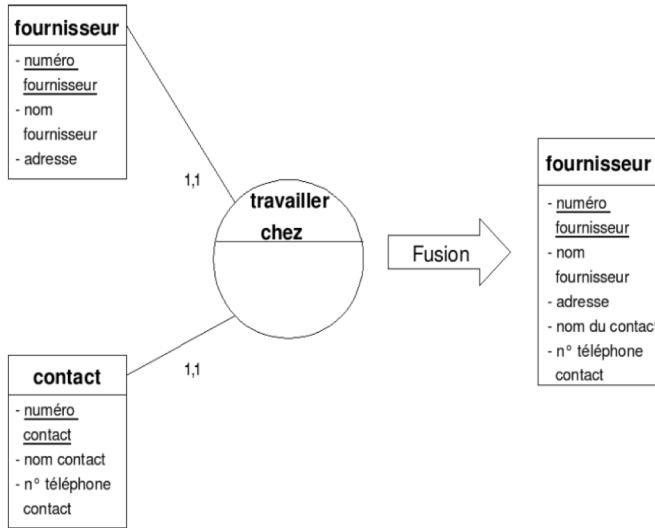


Illustration 17: Les cardinalités sont toutes 1,1 donc c'est une association fantôme

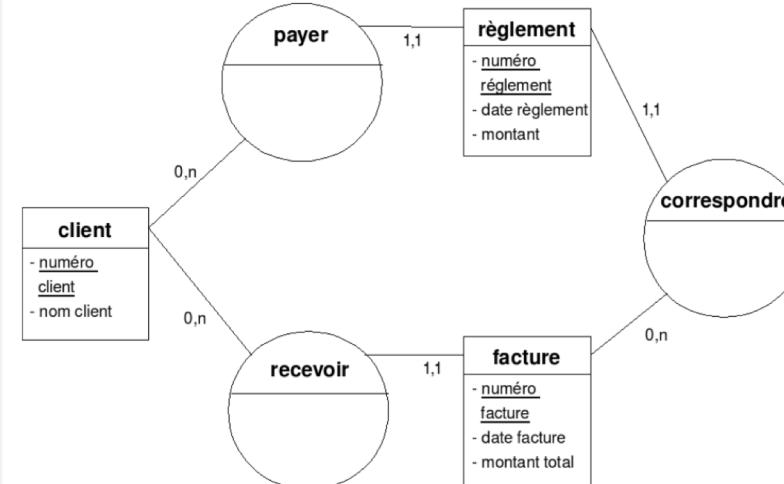


Illustration 18: Si un client ne peut pas régler la facture d'un autre client, alors l'association payer est inutile et doit être supprimée (dans le cas contraire, l'association payer doit être maintenue)

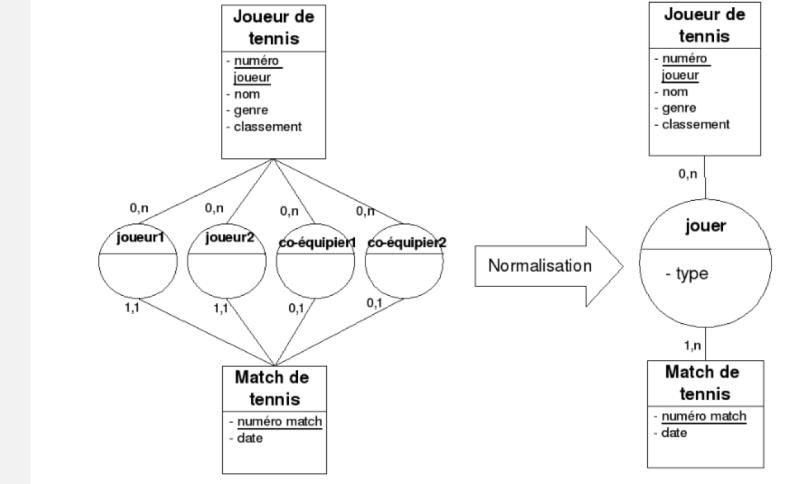


Illustration 19: Une association suffit pour remplacer les 4 associations

# RÈGLES DE NORMALISATION

## 5 - Normalisation des cardinalités

- Une cardinalité minimale est toujours 0 ou 1 (et pas 2, 3 ou n) et une cardinalité maximale est toujours 1 ou n (et pas 2, 3...).
  - Cela signifie que si une cardinalité maximale est connue et vaut 2, 3 ou plus (un nombre limité d'emprunts dans une bibliothèque par exemple), alors nous considérons quand même qu'elle est indéterminé, et vaut n. Cela se justifie par le fait que même si nous connaissons n au moment de la conception, il se peut que cette valeur évolue au cours du temps. Il vaut donc mieux considérer n comme une inconnue dès le départ.
  - Cela signifie également que nous ne modélisons pas les cardinalités minimales qui valent plus de 1 car ce genre de valeur est aussi mené à évoluer. Par ailleurs, avec une cardinalité maximale de 0, l'association n'aurait aucune signification.

# RÈGLES DE NORMALISATION

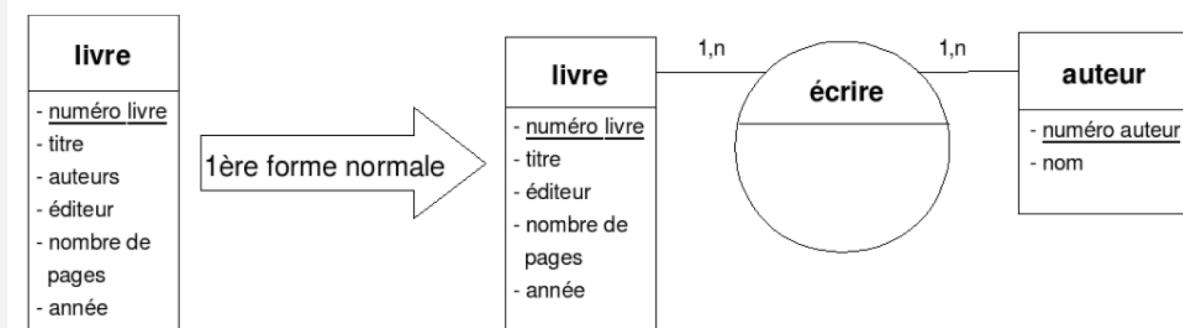


Illustration 20: Application de la première forme normale : il peut y avoir plusieurs auteurs pour un livre donné.

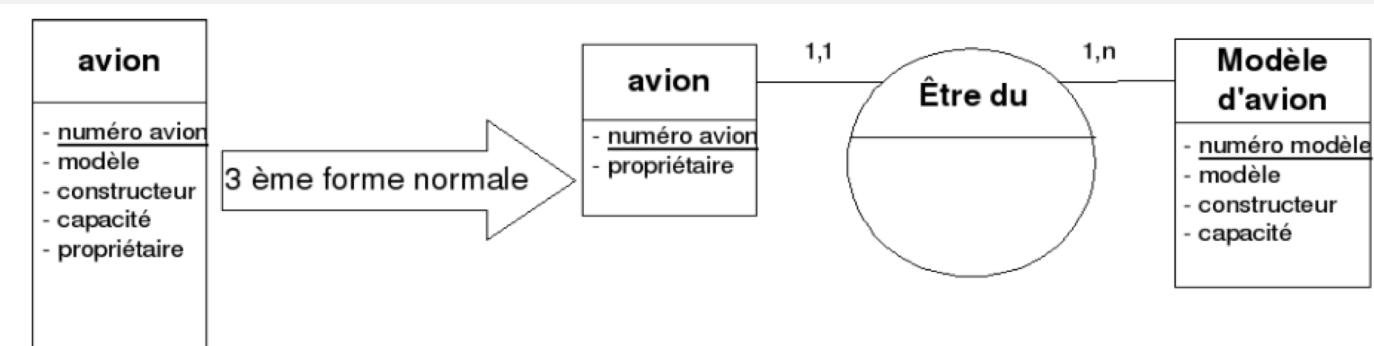


Illustration 21: Application de la troisième forme normale

# MÉTHODOLOGIE

**Face à un problème procéder ainsi :**

- Identifier les entités en présence ;
- Lister leurs attributs ;
- Ajouter les identifiants (numéro arbitraire et auto-incrémenté) ;
- Etablir les associations binaires entre les entités ;
- Lister leurs attributs ;
- Calculer les cardinalités ;
- Vérifier les règles de normalisation et en particulier, la normalisation des entités.
- Effectuer les corrections nécessaires.

# **MODÈLE LOGIQUE DE DONNÉES-MLD**

Maintenant que le MCD est établit, nous pouvons le traduire en différents systèmes logiques, comme les systèmes par fichiers ou les bases de données.

# CONVERSION MCD => MLD

Pour traduire un MCD en un MLDR, il suffit d'appliquer cinq règles.

**Notations :** Nous disons qu'une association entre deux entités (éventuellement réflexive) est de type :

- **I : I** => si les deux cardinalités sont 0,1 ou 1,1.
- **I : n** => si une des deux cardinalités est 0,n ou 1,n, l'autre cardinalité est 0,1 ou 1,1
- **n : m** (plusieurs à plusieurs) => si les deux cardinalités sont 0,n ou 1,n.

	0,1	1,1	0,n	1,n
0,1	1:1	1:1	1:n	1:n
1,1	1:1	1:1	1:n	1:n
0,n	1:n	1:n	n:m	n:m
1,n	1:n	1:n	n:m	n:m

## **CONVERSION MCD => MLD**

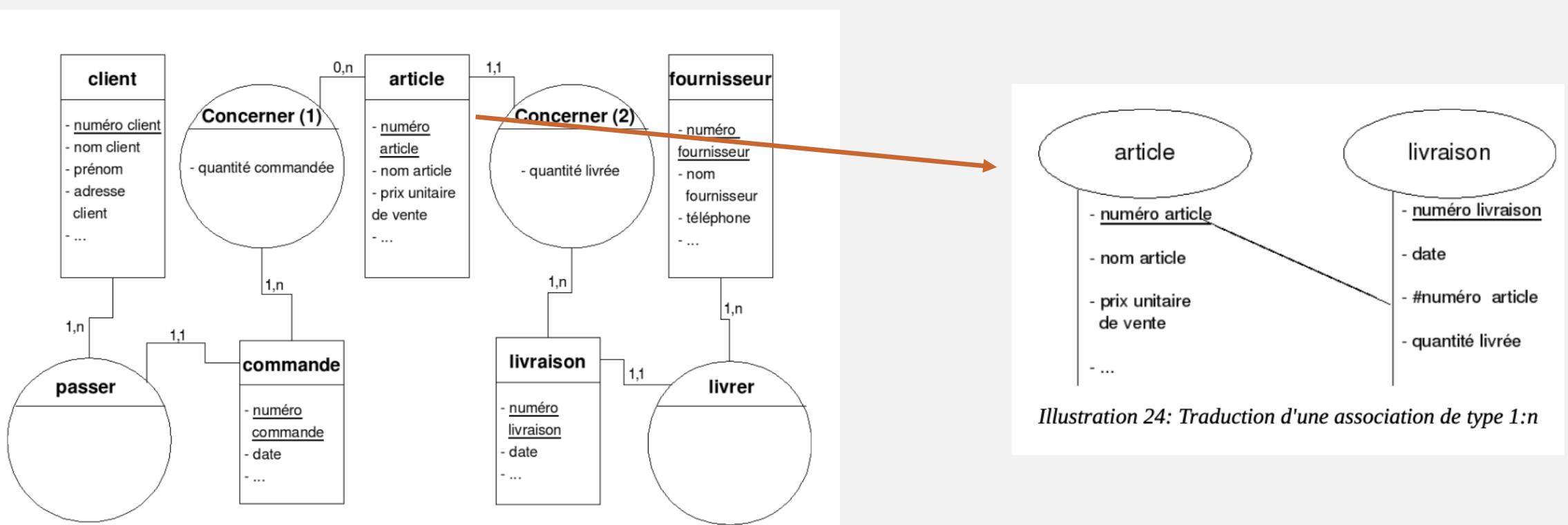
### **Règle 1**

- Toute entité devient une table dans laquelle les attributs sont ceux de l'entité. L'identifiant de l'entité constitue alors la clé primaire de la table. Par exemple, l'entité *article* de l'illustration 15 devient la table :

### **Règle 2**

- Une association binaire de type 1 : n disparaît, au profit d'une clé étrangère dans la table côté 0,1 ou 1,1 qui référence la clé primaire de l'autre table. Cette clé étrangère ne peut pas recevoir la valeur vide si la cardinalité est 1,1.

# CONVERSION MCD => MLD

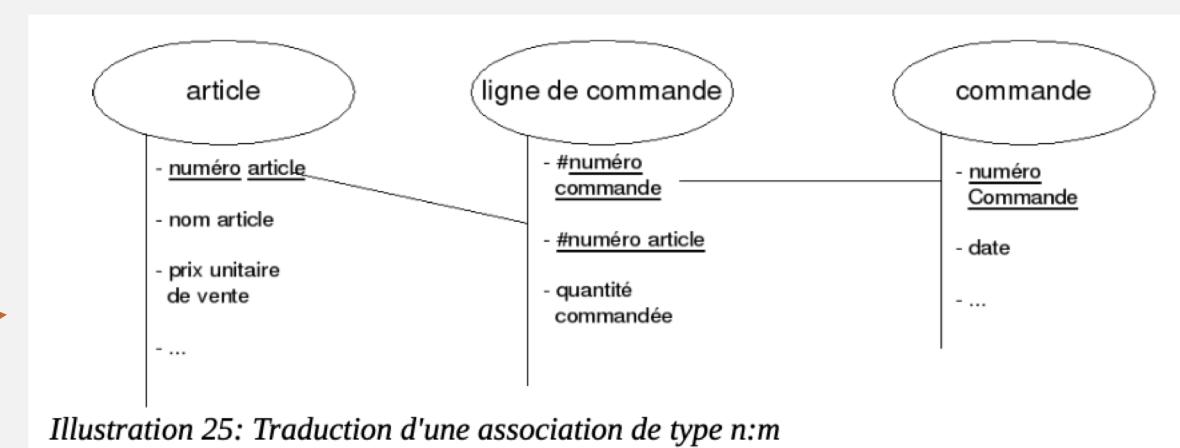
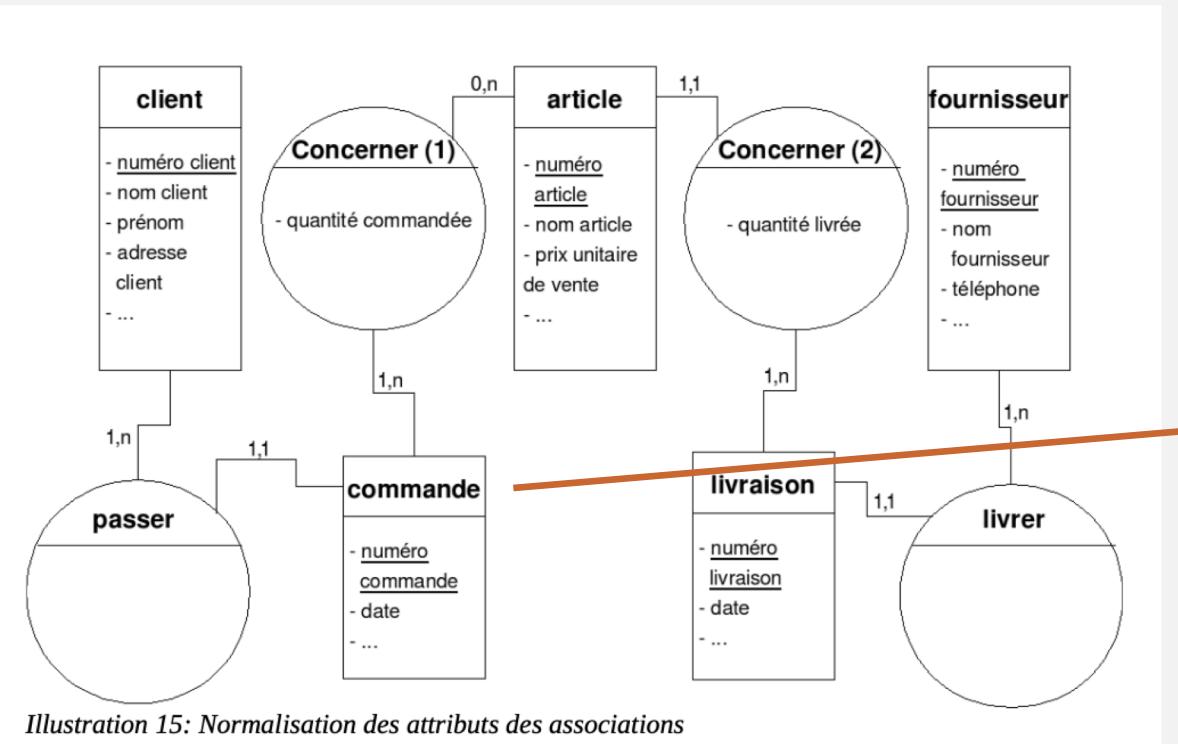


## **CONVERSION MCD => MLD**

### **Règle 3**

- Une association binaire de type n : m devient une table supplémentaire (parfois appelée table de jonction, table de jointure ou table d'association) dont la clé primaire est composée de deux clés étrangères (qui réfèrent aux deux clés primaires des deux tables en association). Les attributs de l'association deviennent des attributs de cette nouvelle table.

# CONVERSION MCD => MLD



# CONVERSION MCD => MLD

## Règle 4

- Une association binaire de type 1 : 1 disparaît, au profit d'une clé étrangère dans la table coté 1,1 qui référence la clé primaire de l'autre table. Cette clé étrangère ne peut pas recevoir la valeur vide. Si les deux cotés sont de cardinalité 0,1 alors la clé étrangère peut être placée indifféremment dans l'une des deux tables.

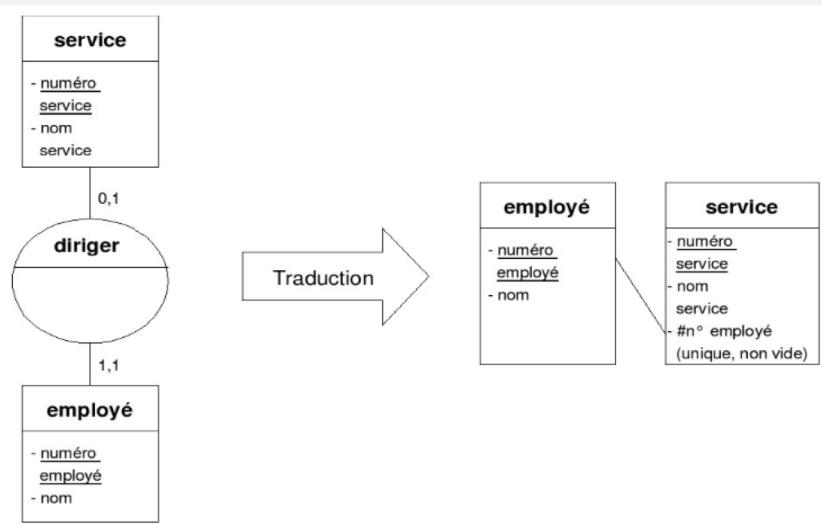


Illustration 26: Traduction d'une association de type 1:1

# CONVERSION MCD => MLD

## Règle 4

- Une association binaire de type 1 : 1 disparaît, au profit d'une clé étrangère dans la table coté 1,1 qui référence la clé primaire de l'autre table. Cette clé étrangère ne peut pas recevoir la valeur vide. Si les deux cotés sont de cardinalité 0,1 alors la clé étrangère peut être placée indifféremment dans l'une des deux tables.

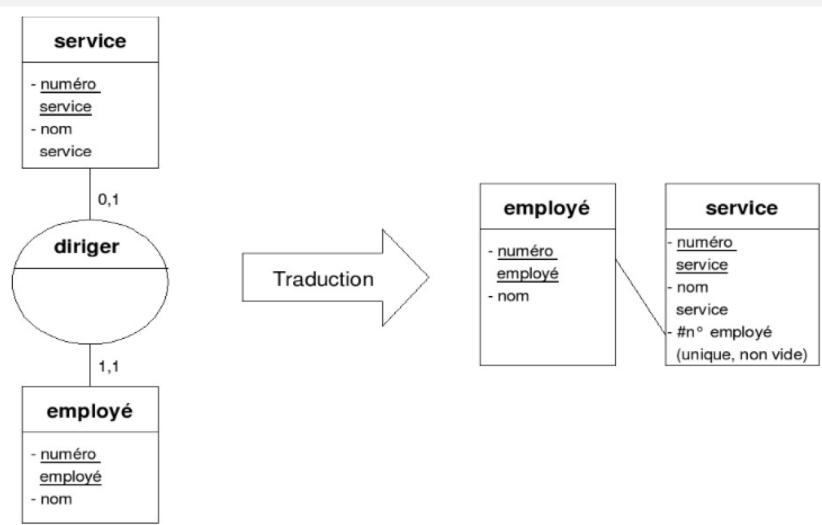


Illustration 26: Traduction d'une association de type 1:1

# CONVERSION MCD => MLD

## Règle 5

Pour une association réflexive, nous avons deux cas :

- **Premier cas** : pour une association de type 1:n , la clé primaire de l'entité se dédouble et devient une clé étrangère dans la nouvelle table. Exactement comme si l'entité se dédoublait et était reliée par une relation binaire 1 : n.

**employé**(*numéro employé, nom, fonction, #numéro supérieur*)

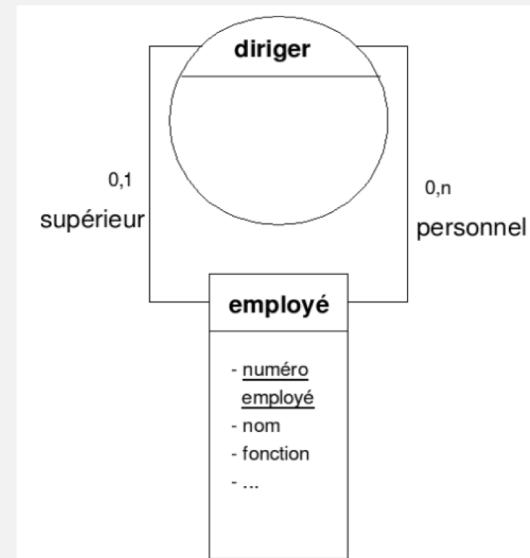
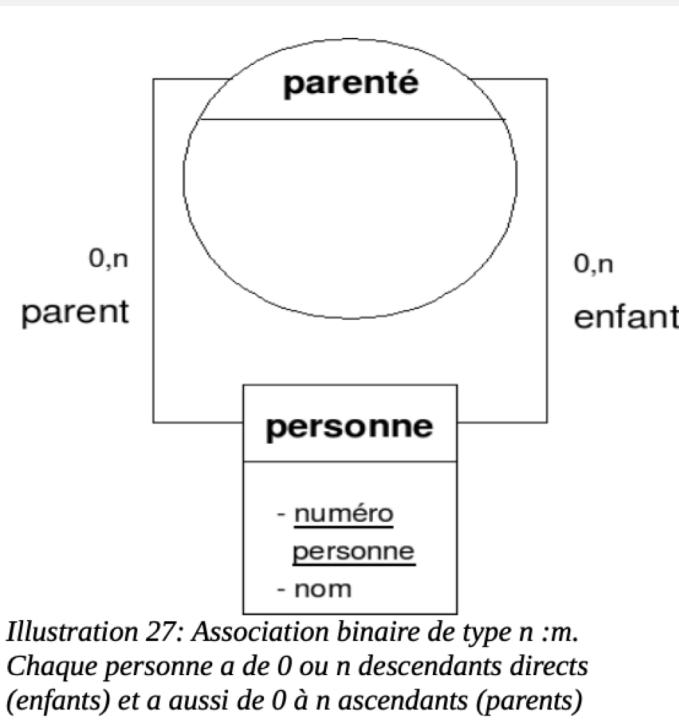


Illustration 6: Exemple d'association réflexive

## CONVERSION MCD => MLD

**Deuxième cas :** pour une association de type  $n : m$ , de même, tout se passe exactement comme si l'entité se dédoublait et était reliée par une relation binaire  $n : m$ . Il y a donc création d'une nouvelle table.



# EXEMPLE

## **Exemple : Un club de Parapente**

- Un club de parapente loue différents modèles de parapente à des pilotes.
- Ces pilotes effectuent des vols ; pour chacun de ces vols le club souhaite connaître le pilote, le modèle de parapente utilisé, le site de décollage, le site d'atterrissement, la date du vol, sa durée, la distance parcourue.
- Pour un parapente sont connues sa date de mise en service et sa couleur principale.  
Pour un pilote le club souhaite connaître son nom, prénom, date de naissance, niveau, poids, date de brevet et surtout son numéro de licence.
- La location d'un parapente par un pilote s'effectue à la journée ; le club souhaite enregistrer le nombre d'utilisations du parapente et la date d'utilisation par le pilote loueur.
- Un modèle de parapente est identifié par un nom ; on lui associe un niveau, une surface, un poids mini et un poids maxi.
- Un site d'atterrissement a un nom unique, une situation et une approche visuelle.  
Un site de décollage a un nom unique, un niveau requis du pilote, une orientation.