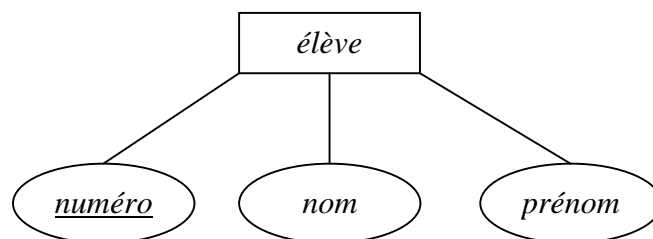




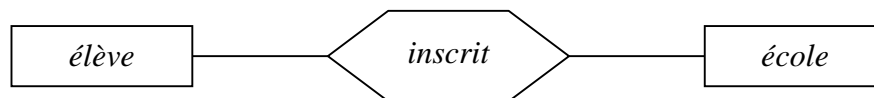
2. **L'attribut** est une donnée élémentaire, un champs, comme par exemple *nom*, *prénom*, *adresse*, *couleur*, *numéro INSEE*... L'attribut est rattachée à une entité. On le représente par un cercle ou un ovale.



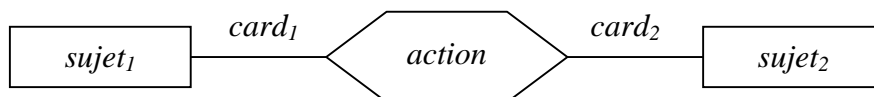
Parmi tous les attributs relatifs à une entité, un au moins doit être souligné, qui représente la **clé**.



3. **L'association** est un lien binaire entre entités, comme par exemple *possède*, *inscrit*, *habite*, *est édité*... Une association peut avoir des attributs, si cet attribut dépend à la fois des deux entités liées. On le représente par un losange.



Pour chaque lien, il convient de préciser **la cardinalité**. La cardinalité est un doublet d'entier du type  $(min,max)$  indiquant le nombre d'occurrence minimale et maximale qui participe à l'association. Les cardinalités minimales sont 0 ou 1 ; les cardinalités maximales sont comprises entre 1 et  $n$  (indéfini).



Une bonne règle pour déterminer les cardinalités :

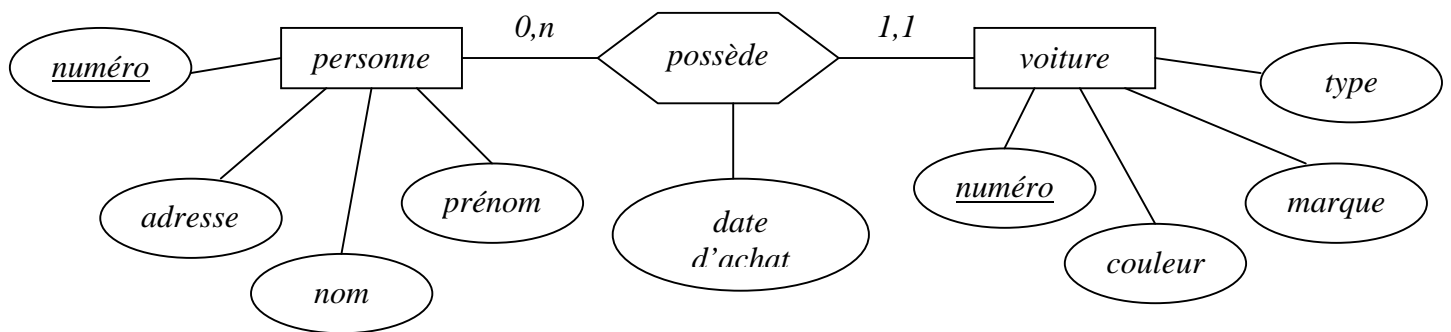
- 1 *sujet<sub>1</sub>* pour (actif) *card<sub>1</sub>* *sujet<sub>2</sub>*
- 1 *sujet<sub>2</sub>* pour (passif) *card<sub>2</sub>* *sujet<sub>1</sub>*

## Exemple

On désire créer une base de données gérant les voitures en circulation. Elle pourra permettre de donner les informations suivantes :

- à propos du propriétaire : n° INSEE, nom, prénom, adresse ;
- à propos du véhicule : numéro, couleur, date d'achat, marque et type.

On donne ci-dessous schéma entité – association pour cette base de données.



Remarquons qu'une personne peut avoir plusieurs voitures, mais qu'une voiture n'a toujours qu'un et un seul propriétaire.

## Modèle hiérarchique et réseau

(...)

## Modèle relationnel

### Définitions

- **Un domaine** est un ensemble de valeurs (entiers, couleurs...).
- Considérons  $n$  domaines  $D_i$ . **Une relation** est un sous-ensemble du produit cartésien  $D_1 \times \dots \times D_n = \{(X_1, \dots, X_n) / X_i \in D_i\}$ . La relation s'écrit sous la forme  $\langle \text{nom relation} \rangle$  ( $\langle \text{liste d'attributs} \rangle$ ).
- On représente une relation  $R$  par **une table**, comme avec l'exemple suivant :  $D_1 = \{0,1\}$  ;  $D_2 = \{\text{couleurs}\}$  ;  $R(\text{couleur}, \text{booléen}) = \{(\text{bleu}, 1), (\text{jaune}, 0)\}$ .

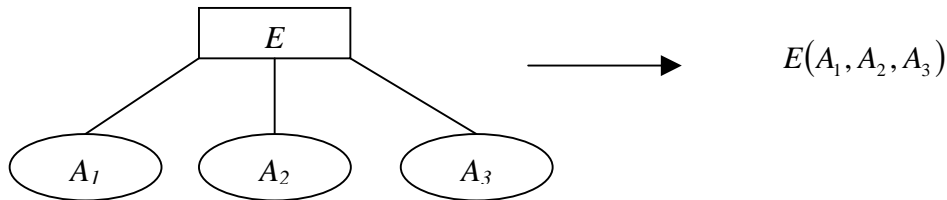
<i>couleur</i>	<i>booléen</i>
<i>bleu</i>	<i>1</i>
<i>jaune</i>	<i>0</i>

- **Un modèle relationnel** est défini par un ensemble de relations, du type :  $\begin{cases} R_1(A_1, A_2, \dots) \\ R_2(B_1, B_2, \dots) \\ \vdots \end{cases}$ .

**Traduction du modèle conceptuel en modèle relationnel (choix d'une SGBD relationnel)**

Les entités et les associations sont traduites par des relations.

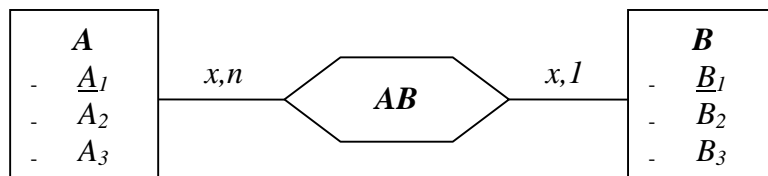
Traduction des entités



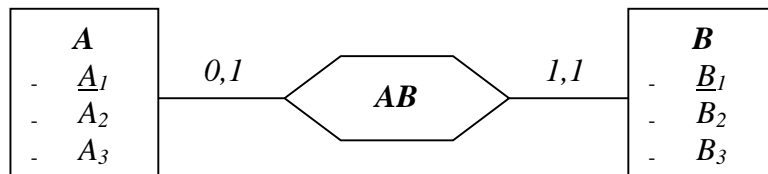
Traduction des associations (plusieurs cas possibles)

Pour passer du modèle conceptuel, on utilise une étape intermédiaire, le **modèle logique des données**. L'étape suivante consiste à transformer chaque table en une relation.

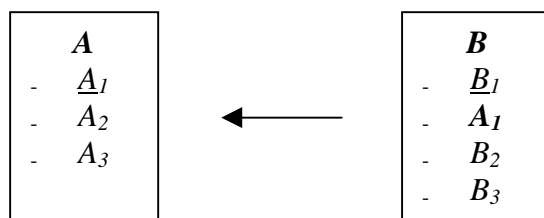
- 1<sup>er</sup> cas et 2<sup>ème</sup> cas



ou bien

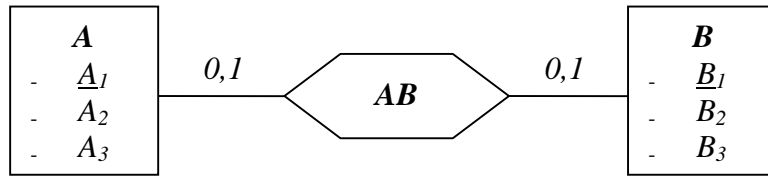


On donne le modèle logique des données associé. On ajoute la clé étrangère  $A_1$  dans  $E_2$ .

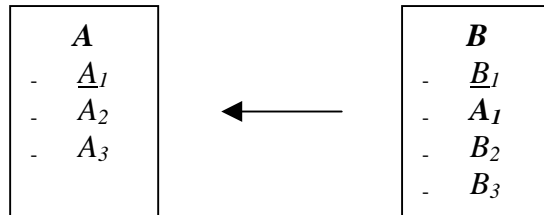


On déduit le modèle relationnel :  $\begin{cases} A(\underline{A_1}, A_2, A_3) \\ B(\underline{B_1}, A_1, B_2, B_3) \end{cases}$ .

▪ 3<sup>ème</sup> cas, cas symétrique

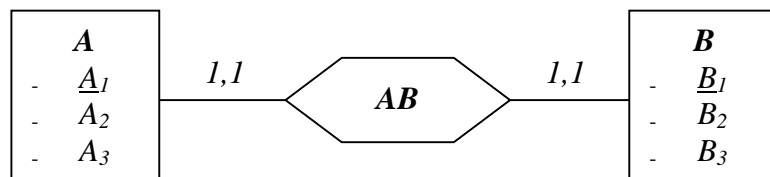


Choix arbitraire de l'une ou l'autre table pour y ajouter la clé étrangère.

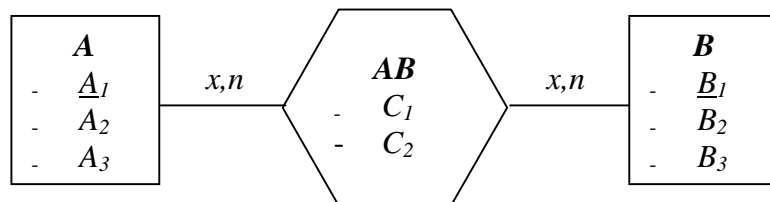


On déduit le modèle relationnel :  $\left\{ \begin{array}{l} A(\underline{A_1}, A_2, A_3) \\ B(\underline{B_1}, A_1, B_2, B_3) \end{array} \right.$  .

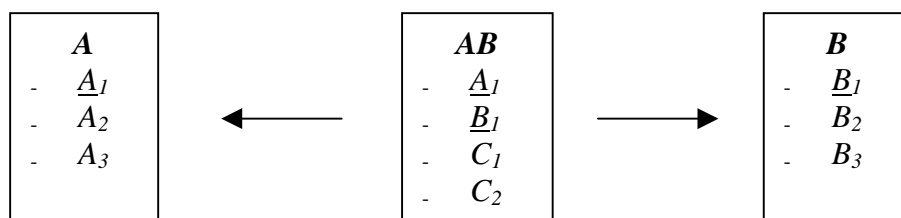
▪ 4<sup>ème</sup> cas, qui n'existe pas dans le modèle conceptuel (regrouper dans la même table)



▪ 5<sup>ème</sup> cas



Ce cas nécessite la création d'une table intermédiaire.



On déduit le modèle relationnel :  $\begin{cases} A(\underline{A}_1, A_2, A_3) \\ B(\underline{B}_1, B_2, B_3) \\ AB(\underline{A}_1, \underline{B}_1, C_1, C_2) \end{cases}$ .

## Algèbre relationnel

L'algèbre relationnel permet de répondre aux requêtes, car (*théorème*) toute requête peut être formulé à partir des quatre opérations suivantes.

- *l'union*

**R**

A	B	C
a	e	f
b	c	d
f	g	e

**S**

A	B	C
i	h	g
b	d	c

**R ∪ S**

A	B	C
a	e	f
b	c	d
f	g	e
i	h	g
b	d	c

- *la différence* :  $R - S = \{X \in R \text{ et } X \notin S\}$

**R**

A	B	C
a	e	f
a	b	c
d	e	g
f	i	j

**S**

A	B	C
a	b	c
d	e	g
a	j	k

**R-S**

A	B	C
a	e	f
f	i	j

- *la projection* :  $\prod_{\text{liste de colonnes}}(R)$  (*pas de redondance*)

**R**

A	B	C
---	---	---

a	b	c
a	b	d
a	c	e
a	b	e
a	c	f

$$\prod_{A,B}(R)$$

A	B
a	b
a	c

- **la sélection** :  $\sigma_{\text{formule}}(R) = \{X \in R / X \text{ vérifie la formule}\}$

**R**

A	B	C
a	b	c
a	b	d
a	c	e
a	b	e
a	c	f

$\sigma_{B='c' \text{ ou } C='d'}$

A	B	C
a	b	d
a	c	e
a	c	f

- **le produit cartésien** :  $R \times S = \{(X, Y) / X \in R, Y \in S\}$
- **l'intersection** :  $R \cap S = R - (R - S)$
- **la division** :  $R \div S$

Soit  $R(A_1, \dots, A_n)$  et  $S(A_{p+1}, \dots, A_n)$  avec  $p < n$ .

$$R \div S = \{X \in (A_1, \dots, A_p) / \forall Y \in S, (X, Y) \in R\}$$

**R**

B	C	D	E
a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

$$S$$

D	E
c	d
e	f

$$R \div S$$

B	C
a	b
e	d

Exemple : (fournisseur, produit)  $\div$  produit = fournisseurs qui vendent tous les produits.

- **la jointure** :  $R \underset{\text{formule}}{\times} S = \{(X, Y), X \in R, Y \in S / (X, Y) \text{ vérifie la formule}\}$

$$R$$

A	B	C
1	2	3
4	9	6
7	8	9

$$S$$

D	E
7	1
9	2

$$R \underset{B < D}{\times} S$$

A	B	C	D	E
1	2	3	7	1
1	2	3	9	2
7	8	9	9	2

- **La jointure naturelle** :  $R \bowtie S$  (cas particulier avec égalité sur les tuples communs)

$$R$$

A	B
1	2
3	2
1	5
1	6

$$S$$

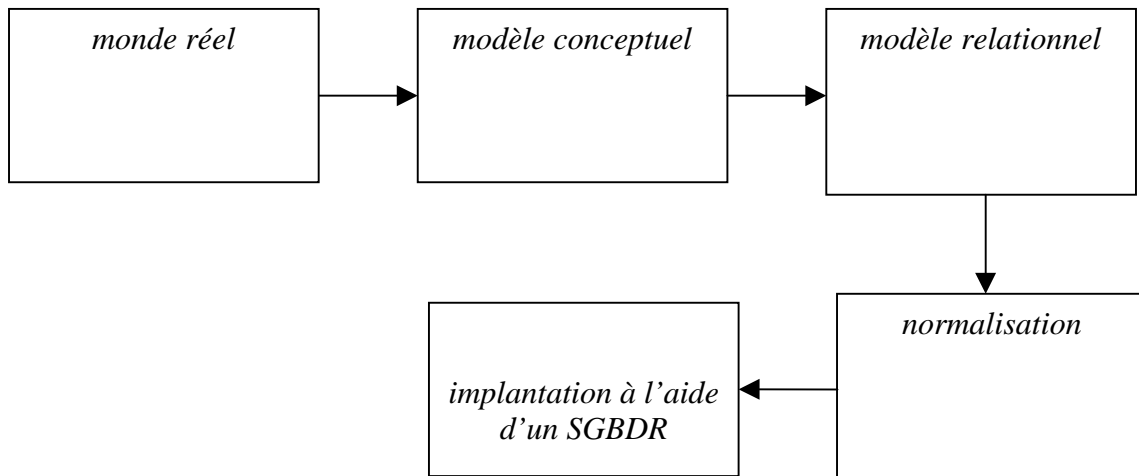
B	C
2	1
2	4
5	1

$$R \bowtie S$$

A	B	C
1	2	1
1	2	4
3	2	1
3	2	4
1	5	1

## Normalisation (modèle relationnel)

On distingue 5 formes normales, qui représentent des tests formels de validité et de cohérence de la base de données. Généralement, on s'arrête à la 3<sup>ème</sup> forme normale.



### Dépendance fonctionnelle

Soit une relation  $R(A, B, C, \dots)$ .  $A$  **détermine**  $B$  ou  $B$  **dépend fonctionnellement** de  $A$  si et seulement si  $a = a' \Rightarrow b = b'$ , c'est-à-dire que pour toute valeur de  $A$  correspond une seule valeur de  $B$ . On note  $A \rightarrow B$ .

$R$	$A$	$B$	$C$
	$a$	$y$	$u$
	$b$	$c$	$d$
	$a$	$c$	$d$
	$f$	$c$	$d$
	$h$	$u$	$v$

$A \rightarrow B$  est faux, mais  $B \rightarrow C$  est vrai.

### Propriétés

- **réflexivité** :  $\forall Y \subseteq X, X \rightarrow Y$
- **augmentation** :  $X \rightarrow Y \Rightarrow XZ \rightarrow YZ$
- **transitivité** :  $(X \rightarrow Y \text{ et } Y \rightarrow Z) \Rightarrow X \rightarrow Z$
- **union** :  $(X \rightarrow Y \text{ et } X \rightarrow Z) \Rightarrow X \rightarrow YZ$
- **pseudo-transitivité** :  $(X \rightarrow Y \text{ et } WY \rightarrow Z) \Rightarrow WX \rightarrow Z$
- **décomposition** :  $(X \rightarrow Y \text{ et } Z \subseteq Y) \Rightarrow X \rightarrow Z$

Soit  $F$  un ensemble de dépendances fonctionnelles. **La fermeture transitive**  $F^+$  est l'ensemble de toutes les dépendances de  $F$  augmenté de celle obtenues par les propriétés précédentes.

On note  $F \Leftrightarrow G$  si et seulement si  $F^+ = G^+$ .

Soit  $F$  un ensemble de dépendances fonctionnelles.  $G$  est **une couverture minimal de  $F$**  si  $G$  est minimal,  $F^+ = G^+$  et  $\forall G' \subset G, G'^+ \neq F^+$ .

Considérons la relation  $Fournisseur(Fnom, Fadresse)$ . Un fournisseur n'a qu'une adresse, par conséquent  $Fnom \rightarrow Fadresse$ .

Une **décomposition** est **sans perte** quand elle conserve les dépendances fonctionnelles. Soit  $R(A_1, A_2, \dots, A_n)$  avec  $A_1 \rightarrow A_2$ , la décomposition de  $R$  en  $R'(A_1, A_2, A_5)$  est sans perte. En revanche, la décomposition de  $R$  en  $R''(A_1, A_3, A_5)$  perd la dépendance fonctionnelle  $A_1 \rightarrow A_2$ .

## Clé

**Une clé** d'une relation est un attribut ou ensemble d'attributs le plus petit possible, qui détermine tous les autres.

Plus formellement, si l'on considère une relation  $R(A_1, A_2, \dots, A_n)$ , une clé  $X$  est un sous-ensemble de  $\{A_1, A_2, \dots, A_n\}$  qui vérifie  $X \rightarrow A_1, A_2, \dots, A_n$  et tel qu'il n'existe pas  $Y \subset X$  vérifiant  $Y \rightarrow A_1, A_2, \dots, A_n$ .

Une clé évidente est l'ensemble des attributs.

On parle de clé potentielle, quand il y a plusieurs clés possibles pour une même relation.

## Formes normales

### 1<sup>ère</sup> forme normale

$R$  est une 1<sup>ère</sup> forme normale si :

- elle possède une clé ;
- tous les attributs sont atomiques (pas d'ensembles, pas de listes).

### *Exemple de normalisation 1*

$R$	$A$	$B$	$C$
	$a$	$\{b, b'\}$	$c$

↓

$R$	$A$	$B$	$C$
	$a$	$b$	$c$
	$a$	$b'$	$c$

### 2<sup>ème</sup> forme normale

$R$  est une 2<sup>ème</sup> forme normale si :

- elle est déjà en 1<sup>ère</sup> forme normale ;

- tous les attributs (n'appartenant pas à la clé) dépendent **pleinement** de la clé  $E$ , c'est-à-dire que  $\forall A \notin E$ , il n'existe pas  $E' \subset E$  tel que  $E' \rightarrow A$ . En fait, tout attribut  $A$  n'appartenant pas à la clé ne doit pas dépendre fonctionnellement d'une partie de la clé.

### Exemple de normalisation 2

Considérons la relation *Fournisseur*(Nom, Adresse, Article, Prix). La clé est (Nom, Article).

On a les dépendances suivantes (entre des attributs appartenant à la clé et d'autres n'appartenant à la clé) :

- $Nom, Article \rightarrow Prix$
- $Nom \rightarrow Adresse$

La normalisation 2 consiste à séparer la relation *Fournisseur* en autant de sous-relations qu'il est nécessaire pour que chacune d'elle vérifie la 2<sup>ème</sup> forme normale, ce qui donne :

- $R_1(Nom, Article, Prix)$
- $R_2(Nom, adresse)$

### 3<sup>ème</sup> forme normale

$R$  est une 3<sup>ème</sup> forme normale si :

- elle est déjà en 2<sup>ème</sup> forme normale ;
- il n'y a pas de dépendances fonctionnelles entre attributs non clés.

### Exemple de normalisation 3

Considérons la relation *R*(NumPièce, CodeTVA, TauxTVA).

On a les dépendances suivantes entre attributs non clés :

- $CodeTVA \rightarrow TauxTVA$

La normalisation 3 consiste à séparer la relation *R* en autant de sous-relations qu'il est nécessaire pour que chacune d'elle respecte la 3<sup>ème</sup> forme normale, ce qui donne :

- $R_1(NumPièce, CodeTVA)$
- $R_2(CodeTVA, TauxTVA)$

### Regroupement

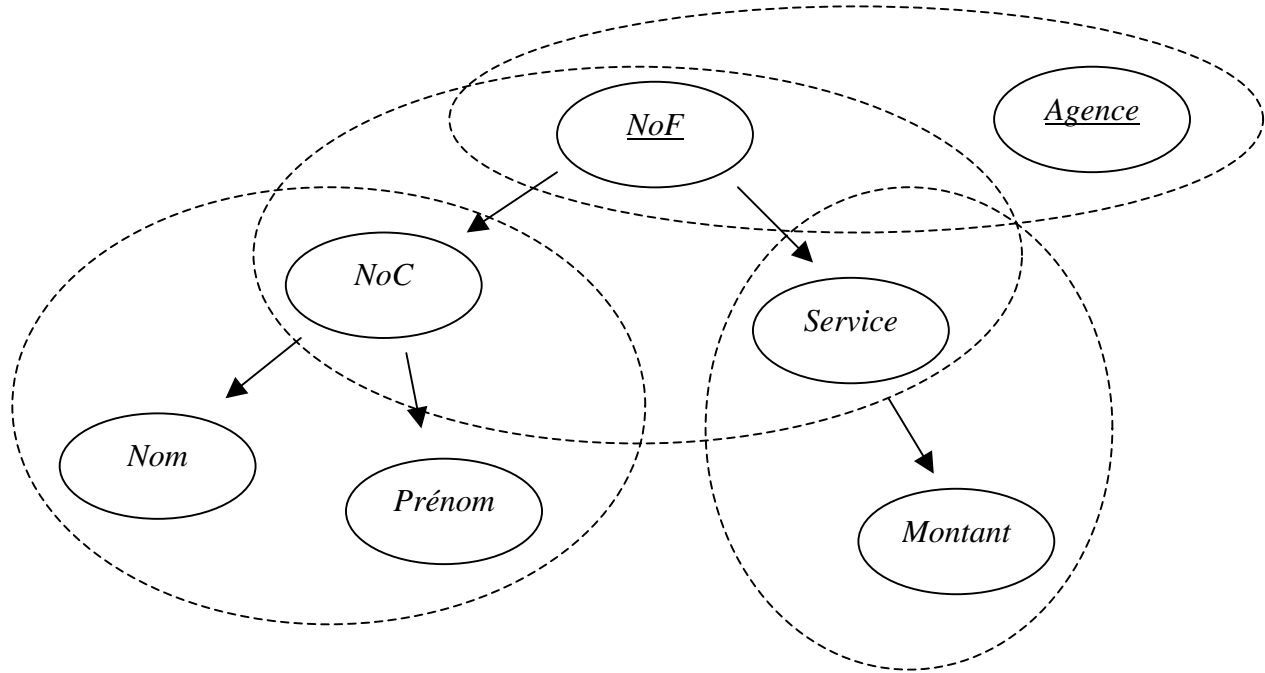
Certains regroupements peuvent être judicieux... La duplication d'une clé n'est pas grave, cela revient à la duplication d'une adresse mémoire.

### Exemple de normalisation

Considérons la relation *Factures*(NoC, Nom, Prénom, NoF, Service, Montant, Agence) avec les dépendances fonctionnelles suivantes :

- $NoC \rightarrow Nom$
- $NoC \rightarrow Prénom$
- $NoF \rightarrow Service$
- $NoF \rightarrow NoC$
- $Service \rightarrow Montant$

On utilise un graphe montrant les dépendances fonctionnelles.



De manière évidente, la clé est (*NoF*, *Agence*).

La normalisation en troisième forme normale consiste à construire les cercles pointillés sur le graphe ayant un rayon 1.

### 1<sup>ère</sup> forme normale

$R(\text{NoC}, \text{Nom}, \text{Prénom}, \text{NoF}, \text{Service}, \text{Montant}, \text{Agence})$

### 2<sup>ème</sup> forme normale (pleine dépendance des clés)

$R_1(\text{NoF}, \text{Agence})$

$R_2(\text{NoF}, \text{NoC}, \text{Nom}, \text{Prénom}, \text{Service}, \text{Montant})$

### 3<sup>ème</sup> forme normale (pas de dépendances entre les attributs non clés)

$R'_2(\text{NoF}, \text{NoC}, \text{Service})$

$R''_2(\text{NoC}, \text{Nom}, \text{Prénom})$

$R'''_2(\text{Service}, \text{Montant})$

## Oracle, SQL

(...)