Conception des bases de données : le modèle entités-associations

Chapitre 2 du cours
Base de Données et Langage SQL
de Laurent AUDIBERT

(En complément consulter l'ouvrage du même auteur paru en 2009 chez Ellipses dans la collection Info+ sous le titre Base de données - de la modélisation au SQL)

Avant-propos

Aujourd'hui, la disponibilité de systèmes de gestion de base de données fiables permet aux organisations de toutes tailles de gérer des données efficacement, de déployer des applications utilisant ces données et de les stocker. Les bases de données sont actuellement au cœur du système d'information des entreprises.

Les bases de données relationnelles constituent l'objet de ce cours. Ces bases sont conçues suivant le modèle relationnel, dont les fondations théoriques sont solides, et manipulées en utilisant l'algèbre relationnelle. Il s'agit, à ce jour, de la méthode la plus courante pour organiser et accéder à des ensembles de données.

……..

2.1 Introduction

2.1.1 Pourquoi une modélisation préalable ?

Il est difficile de modéliser un domaine sous une forme directement utilisable par un SGBD. Une ou plusieurs modélisations intermédiaires sont donc utiles, le modèle entités-associations constitue l'une des premières et des plus courantes. Ce modèle, présenté par Chen76, permet une description naturelle du monde réel à partir des concepts d'entité et d'association[1]. Basé sur la théorie des ensembles et des relations, ce modèle se veut universel et répond à l'objectif d'indépendance données-programmes. Ce modèle, utilisé pour la phase de conception, s'inscrit notamment dans le cadre d'une méthode plus générale et répandue : *Merise*.

2.1.2 Merise

MERISE (Méthode d'Étude et de Réalisation Informatique pour les Systèmes d'Entreprise) est certainement le langage de spécification le plus répandu dans la communauté de l'informatique des systèmes d'information, et plus particulièrement dans le domaine des bases de données. Une représentation Merise permet de valider des choix par rapport aux objectifs, de quantifier les solutions retenues, de mettre en œuvre des techniques d'optimisation et enfin de guider jusqu'à l'implémentation. Reconnu comme standard, Merise devient un outil de communication. En effet, Merise réussit le compromis difficile entre le souci d'une modélisation précise et formelle, et la capacité d'offrir un outil et un moyen de communication accessible aux non-informaticiens.

Un des concepts clés de la méthode Merise est la séparation des données et des traitements. Cette méthode est donc parfaitement adaptée à la modélisation des problèmes abordés d'un point de vue fonctionnel[2]. Les données représentent la *statique* du système d'information et les traitements sa *dynamique*. L'expression conceptuelle des données conduit à une modélisation des données en *entités* et en *associations*. Dans ce cours, nous écartons volontairement la modélisation des traitements puisque nous ne nous intéressons à la méthode Merise que dans la perspective de la modélisation de bases de données.

Merise propose une démarche, dite par niveaux, dans laquelle il s'agit de hiérarchiser les préoccupations de modélisation qui sont de trois ordres : la conception, l'organisation et la technique. En effet, pour aborder la modélisation d'un système, il convient de l'analyser en premier lieu de façon globale et de se concentrer sur sa fonction : c'est-à-dire de s'interroger sur ce qu'il fait avant de définir comment il le fait. Ces niveaux de modélisation sont organisés dans une double approche données/traitements. Les trois niveaux de représentation des données, puisque ce sont eux qui nous intéressent, sont détaillés ci-dessous.

Niveau conceptuel :

le *modèle conceptuel des données (MCD)* décrit les entités du monde réel, en terme d'objets, de propriétés et de relations, indépendamment de toute technique d'organisation et d'implantation des données. Ce modèle se concrétise par un *schéma entités-associations* représentant la structure du système d'information, du point de vue des données.

Niveau logique :

le *modèle logique des données (MLD)* précise le modèle conceptuel par des choix organisationnels. Il s'agit d'une transcription (également appelée dérivation) du MCD dans un formalisme adapté à une implémentation ultérieure, au niveau physique, sous forme de base de données relationnelle ou réseau, ou autres (cf. section 1.1.2). Les choix techniques d'implémentation (choix d'un SGBD) ne seront effectués qu'au niveau suivant.

Niveau physique :

le *modèle physique des données (MPD)* permet d'établir la manière concrète dont le système sera mis en place (SGBD retenu).

1
Dans la littérature, on utilise indifféremment le terme *relation* ou le terme *association*, on parle donc de modèle entités-relations (E-R) ou de modèle entités-associations (E-A). Nous préférons utiliser le terme *association* plutôt que le terme *relation* pour limiter la confusion avec les relations du modèle relationnel.

2
A contrario, Merise n'est pas adapté à la modélisation des problèmes abordés d'une manière orientée objet (dans ce cas, il faut, par exemple, utiliser UML).

2.2 Éléments constitutifs du modèle entités-associations

La représentation du modèle entités-associations s'appuie sur trois concepts de base :

- l'objet ou entité,
- l'association,
- la propriété.

L'objet est une entité ayant une existence propre. L'association est un lien ou relation entre objets sans existence propre. La propriété est la plus petite donnée d'information décrivant un objet ou une association.

2.2.1 Entité

Figure 2.1: Représentation graphique d'un exemple de type-entité.

Définition 1 ***-entité-*** *Une entité est un objet, une chose concrète ou abstraite qui peut être reconnue distinctement et qui est caractérisée par son unicité.*

Exemples d'entité : Jean Dupont, Pierre Bertrand, le livre que je tiens entre les mains, la Ferrari qui se trouve dans mon garage, etc.

Les entités ne sont généralement pas représentées graphiquement.

Définition 2 ***-type-entité-*** *Un type-entité désigne un ensemble d'entités qui possèdent une sémantique et des propriétés communes.*

Les personnes, les livres et les voitures sont des exemples de type-entité.Dans le cas d'une personne par exemple, les informations associées (les *propriétés*), comme le nom, le prénom, ne changent pas de nature.

Une entité est souvent nommée occurrence ou instance de son type-entité.

La figure 2.1 montre la représentation d'un exemple de type-entité (*Personne*) sans ses propriétés associées.

Les type-entité *Personne*, caractérisé par un nom et un prénom, et *Voiture*, caractérisé par un nom et une puissance fiscale, ne peuvent pas être regroupés car ils ne partagent leurs propriétés (le prénom est une chaîne de caractères et la puissance fiscale un nombre). Les type-entité *Personne*, caractérisé par un nom et un prénom, et *Livre*, caractérisé un titre et un auteur, possèdent tous les deux deux attributs du type *chaîne de caractères*. Pourtant, ces deux type-entités ne peuvent pas être regroupés car ils ne partagent pas une même sémantique : le nom d'une personne n'a rien à voir avec le titre d'un livre, le prénom d'une personne n'a rien à voir avec un auteur.

Par abus de langage, on utilise souvent le mot entité en lieu et place du mot type-entité, il faut cependant prendre garde à ne pas confondre les deux concepts.

2.2.2 Attribut ou propriété, valeur

Figure 2.2: Représentation graphique d'un exemple de type-entité comportant trois attributs

---

Définition 3 ***-attribut, propriété-*** *Un attribut (ou une propriété) est une caractéristique associée à un type-entité ou à un type-association.*
Exemples d'attribut : le nom d'une personne, le titre d'une livre, la puissance d'une voiture.
Définition 4 ***-valeur-*** *Au niveau du type-entité ou du type-association, chaque attribut possède un domaine qui définit l'ensemble des valeurs possibles qui peuvent être choisies pour lui (entier, chaîne de caractères, booléen, ...). Au niveau de l'entité, chaque attribut possède une* ***valeur*** *compatible avec son domaine.*
La figure 2.2 montre la représentation graphique d'un exemple de type-entité (*Personne*) avec trois attributs.
Règle 5 *Un attribut ne peut en aucun cas être partagé par plusieurs type-entités ou type-associations.*
Règle 6 *Un attribut est une donnée élémentaire, ce qui exclut des données calculées ou dérivées.*
Règle 7 *Un type-entité et ses attributs doivent être cohérents entre eux (*i.e. *ne traiter que d'un seul sujet).*
Par exemple, si le modèle doit comporter des informations relatives à des articles et à leur fournisseur, ces informations ne doivent pas coexister au sein d'un même type-entité. Il est préférable de mettre les informations relatives aux articles dans un type-entité *Article* et les informations relatives aux fournisseurs dans un type-entité *Fournisseur*. Ces deux type-entités seront probablement ensuite reliés par un type-association.

2.2.3 Identifiant ou clé

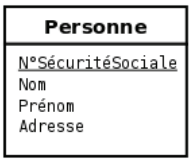

Figure 2.3: Représentation graphique d'un exemple de type-entité comportant quatre attributs dont un est un identifiant : deux personnes peuvent avoir le même nom, le même prénom et le même âge, mais pas le même numéro de sécurité sociale.

Définition 8 ***-identifiant, clé-*** *Un identifiant (ou clé) d'un type-entité ou d'un type-association est constitué par un ou plusieurs de ses attributs qui doivent avoir une valeur unique pour chaque entité ou association de ce type.*
Il est donc impossible que les attributs constituant l'identifiant d'un type-entité (respectivement type-association) prennent la même valeur pour deux entités (respectivement deux associations) distinctes.
Exemples d'identifiant : le numéro de sécurité sociale pour une personne, le numéro d'immatriculation pour une voiture, le code ISBN d'un livre pour un livre (mais pas pour un exemplaire).
Règle 9 *Chaque type-entité possède au moins un identifiant, éventuellement formé de plusieurs attributs.*
Ainsi, chaque type-entité possède au moins un attribut qui, s'il est seul, est donc forcément l'identifiant.
Dans la représentation graphique, les attributs qui constituent l'identifiant sont soulignés et placés en tête (cf. figure 2.3).

2.2.4 Association ou relation

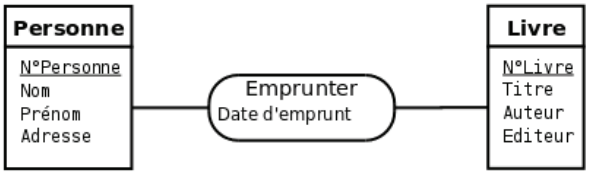

Figure 2.4: Représentation graphique d'un exemple de type-association liant 2 type-entités.

Définition 10 ***-association-*** *Une association (ou une relation) est un lien entre plusieurs entités.*

Exemples d'association : l'emprunt par l'étudiant Tanidute du 3 ème exemplaire du livre « Maîtrisez SQL ».
Les associations ne sont généralement pas représentées graphiquement.

Définition 11 ***-type-association-*** *Un type-association (ou un type-relation) désigne un ensemble de relations qui possèdent les mêmes caractéristiques. Le type-association décrit un lien entre plusieurs type-entités. Les associations de ce type-association lient des entités de ces type-entités.*

Comme les type-entités, les type-associations sont définis à l'aide d'attributs qui prennent leur valeur dans les associations.

Règle 12 *Un attribut peut être placé dans un type-association uniquement lorsqu'il dépend de toutes les entités liées par le type-association.*

Un type-association peut ne pas posséder d'attribut explicite et cela est relativement fréquent, mais on verra qu'il possède au moins des attributs implicites.

Exemples de type-association : l'emprunt d'un livre à la bibliothèque.

Une association est souvent nommée occurrence ou instance de son type-association.

La figure 2.4 montre la représentation graphique d'un exemple de type-association.

Par abus de langage, on utilise souvent le mot association en lieu et place du mot type-association, il faut cependant prendre garde à ne pas confondre les deux concepts.

Définition 13 ***-participant-*** *Les type-entités intervenant dans un type-association sont appelés les participants de ce type-association.*

Définition 14 ***-collection-*** *L'ensemble des participants d'un type-association est appelé la collection de ce type-association.*

Cette collection comporte au moins un type-entité (cf. section 2.3.2), mais elle peut en contenir plus, on parle alors de type-association *n-aire* (quand n=2 on parle de type-association binaire, quand n=3 de type-association ternaire, …).

Définition 15 ***-dimension ou arité d'un type-association-*** *La dimension, ou l'arité d'un type-association est le nombre de type-entités contenu dans la collection.*

Comme un type-entité, un type-association possède forcément un identifiant, qu'il soit explicite ou non.

Règle 16 *La concaténation des identifiants des type-entités liés à un type-association constitue un identifiant de ce type-association et cet identifiant n'est pas mentionné sur le modèle (il est implicite).*

Cette règle implique que deux instances d'un même type-association ne peuvent lier un même ensemble d'entités.

Souvent, un sous-ensemble de la concaténation des identifiants des type-entités liés suffit à identifier le type-association.

On admet également un identifiant plus naturel et explicite, à condition qu'il ne soit qu'un moyen d'exprimer plus simplement cette concaténation.

2.2.5 Cardinalité

Figure 2.5: Représentation graphique des cardinalités d'un type-association. Dans cet exemple pédagogique, on suppose qu'un livre ne peut posséder qu'un auteur.

Définition 17 ***-cardinalité-*** *La cardinalité d'une patte reliant un type-association et un type-entité précise le nombre de fois minimal et maximal d'interventions d'une entité du type-entité dans une association du type-association. La cardinalité minimale doit être inférieure ou égale à la cardinalité maximale.*

Exemple de cardinalité : une personne peut être l'auteur de 0 à n livre, mais un livre ne peut être écrit que par une personne (cf. figure 2.5).

Règle 18 *L'expression de la cardinalité est obligatoire pour chaque patte d'un type-association.*

Règle 19 *Une cardinalité minimal est toujours 0 ou 1 et une cardinalité maximale est toujours 1 ou n.*

Ainsi, si une cardinalité maximale est connue et vaut 2, 3 ou plus, alors nous considérons qu'elle est indéterminée et vaut n. En effet, 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 inconnue dès le départ. De la même manière, on ne modélise pas des cardinalités minimales qui valent plus de 1 car ces valeurs sont également susceptibles d'évoluer. Enfin, une cardinalité maximale de 0 n'a pas de sens car elle rendrait le

type-association inutile.
Les seuls cardinalités admises sont donc :
0,1 :
une occurrence du type-entité peut exister tout en étant impliquée dans aucune association et peut être impliquée dans au maximum une association.
0,n :
c'est la cardinalité la plus ouverte ; une occurrence du type-entité peut exister tout en étant impliquée dans aucune association et peut être impliquée, sans limitation, dans plusieurs associations.
1,1 :
une occurrence du type-entité ne peut exister que si elle est impliquée dans exactement (au moins et au plus) une association.
1,n :
une occurrence du type-entité ne peut exister que si elle est impliquée dans au moins une association.
Une cardinalité minimale de 1 doit se justifier par le fait que les entités du type-entité en questions ont besoin de l'association pour exister. Dans tous les autres cas, la cardinalité minimale vaut 0. Ceci dit, la discussion autour d'une cardinalité minimale de 0 ou de 1 n'est intéressante que lorsque la cardinalité maximale est 1. En effet, nous verrons que, lors de la traduction vers un schéma relationnel (cf. section 3.1.3), lorsque la cardinalité maximale est *n*, nous ne ferons pas la différence entre une cardinalité minimale de 0 ou de 1.
Remarques
La seule difficulté pour établir correctement les cardinalités est de se poser les question dans le bon sens. Pour augmenter le risque d'erreurs, il faut noter que, pour les habitués, ou les futurs habitués, du modèle UML, les cardinalités d'un type-association sont « à l'envers » (par référence à UML) pour les type-associations binaires et « à l'endroit » pour les n-aires avec $n>2$.
La notion de cardinalité n'est pas définie de la même manière dans le modèle Américain et dans le modèle Européen (Merise). Dans le premier n'existe que la notion de cardinalité maximale.
Avec un SGBD relationnel, nous pourrons contraindre des cardinalités à des valeurs comme 2, 3 ou plus en utilisant des déclencheurs (*trigger*).

2.3 Compléments sur les associations

2.3.1 Associations plurielles

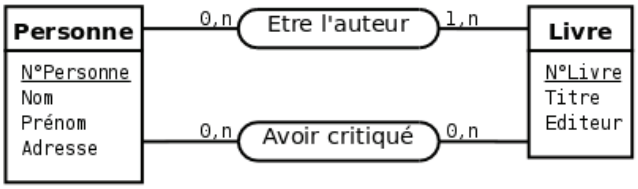

Figure 2.6: Exemple d'associations plurielles entre un type-entité *Personne* et un type-entité *Livre*. Sur ce schéma, un type-association permet de modéliser que des personnes écrivent des livres et un autre que des personnes critiquent (au sens de critique littéraire) des livres.

Deux mêmes entités peuvent être plusieurs fois en association (c'est le cas sur la figure 2.6).

2.3.2 Association réflexive

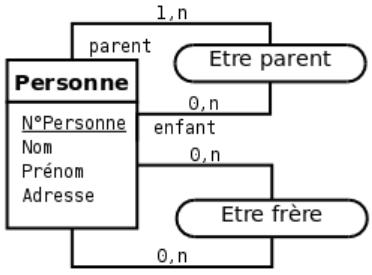

Figure 2.7: Exemple d'associations reflexives sur le type-entité *Personne*. Le premier type-association permet de modéliser la relation parent/enfant et le deuxième type-association la relation de fraternité.

Les type-associations réflexifs sont présents dans la plupart des modèles.

Définition 20 ***-Type-association réflexif-*** *Un type-association est qualifié de réflexif quand il matérialise une relation entre un type-entité et lui-même (cf. figure 2.7).*

Une occurrence de ce type-association (*i.e.* une association) associe généralement une occurrence du type-association (*i.e.* une entité) à une autre entité du même type. Cette relation peut être symétrique, c'est le cas du type-association *Etre frère* sur la figure 2.7, ou ne pas l'être, comme le type-association *Etre parent* sur cette même figure. Dans le cas où la relation n'est pas symétrique, on peut préciser les rôles sur les pattes du type-association comme pour la relation *Etre parent* de la figure 2.7. L'ambiguïté posée par la non-symétrie d'un type-association réflexif sera levée lors du passage au modèle relationnel (cf. section 3.1.3).

2.3.3 Association n-aire (*n*>2)

Dans la section 2.2.4 nous avons introduit la notion de type-association *n-aire*. Ce type-association met en relation *n* type-entités. Même s'il n'y a, en principe, pas de limite sur l'arité d'un type-association, dans la pratique on ne va rarement au-delà de trois. Les associations de degré supérieur à deux sont plus difficiles à manipuler et à interpréter, notamment au niveau des cardinalités.

Exemple d'association n-aire inappropriée

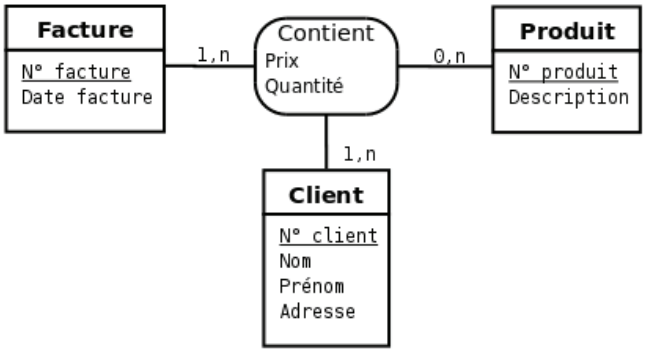

Figure 2.8: Exemple de type-association ternaire inapproprié.

Le type-association ternaire *Contient* associant les type-entités *Facture*, *Produit* et *Client* représenté sur la figure 2.8 est inapproprié puisqu'une facture donnée est toujours adressée au même client. En effet, cette modélisation implique pour les associations (instances du type-association) *Contient* une répétition du numéro de client pour chaque produit d'une même facture.

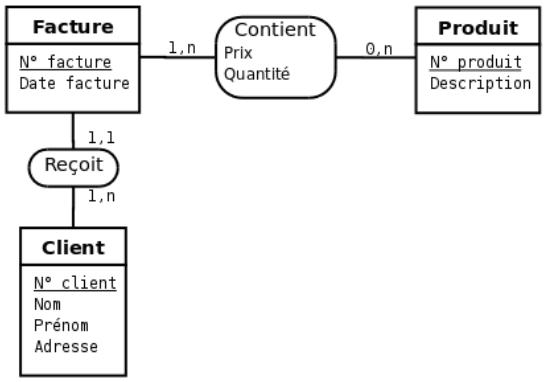

Figure 2.9: Type-association ternaire de la figure 2.8 corrigé en deux type-associations binaires.

La solution consiste à éclater le type-association ternaire *Contient* en deux type-associations binaires comme représenté sur la figure 2.9.

Décomposition d'une association n-aire

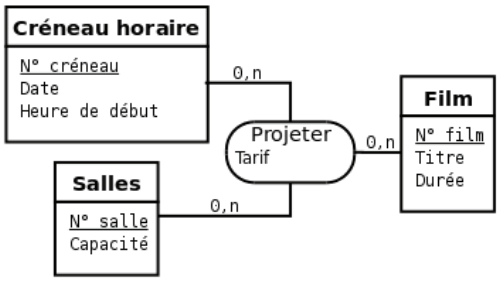

Figure 2.10: Exemple de type association ternaire entre des type-entités *Créneau horaire*, *Salle* et *Film*.

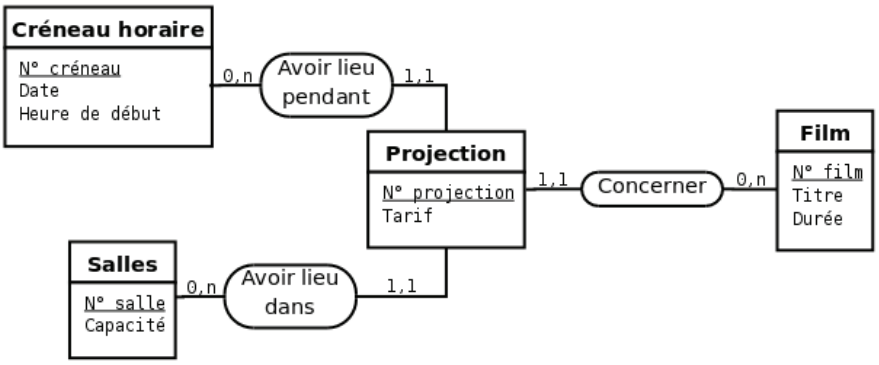

Figure 2.11: Transformation du type-association ternaire de la figure 2.10 en un type-entité et trois type-associations binaires.

La figure 2.10 nous montre un exemple de type-association ternaire entre les type-entités *Créneau horaire*, *Salle* et *Film*. Il est toujours possible de s'affranchir d'un type-association n-aire (n>2) en se ramenant à des type-associations binaires de la manière suivante :

- On remplace le type-association n-aire par un type-entité et on lui attribut un identifiant.
- On crée des type-associations binaire entre le nouveau type-entité et tous les type-entités de la collection de l'ancien type-association n-aire.
- La cardinalité de chacun des type-associations binaires créés est 1,1 du côté du type-entité créé (celui qui remplace le type-association n-aire), et 0,*n* ou 1,*n* du côté des type-entités de la collection de l'ancien type-association n-aire.

La figure 2.11 illustre le résultat de cette transformation sur le schéma de la figure 2.10.
L'avantage du schéma de la figure 2.11 est de rendre plus intelligible la lecture des cardinalités. Il ne faut surtout pas le voir comme un aboutissement mais comme une étape intermédiaire avant d'aboutir au schéma de la figure 2.10 (cf. règle 27). Ainsi, le mécanisme, que nous venons de détailler ci-dessus, de passage d'un type-association n-aire (n>2) à un type-entité et n type-associations binaires est tout à fait réversible à condition que :

- toutes les pattes des type-associations binaires autour du type-entité central ont une cardinalité maximale de 1 au centre et de *n* à l'extérieur ;
- les attributs du type-entité central satisfont la règle de bonne formation des attributs de type-association (cf. section 2.4.2).

Détection d'une erreur de modélisation par décomposition d'une association n-aire

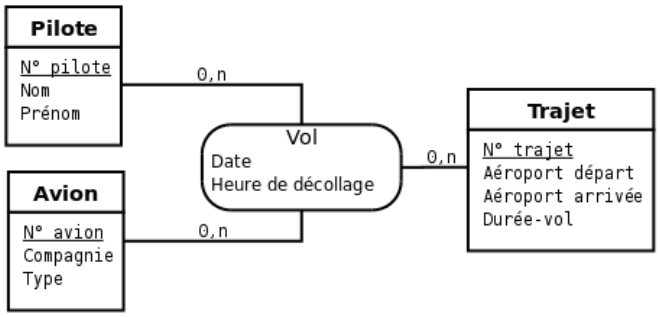

Figure 2.12: Modèle représentant un type-association ternaire *Vol* liant trois type-entités *Avion*, *Trajet* et *Pilote*.

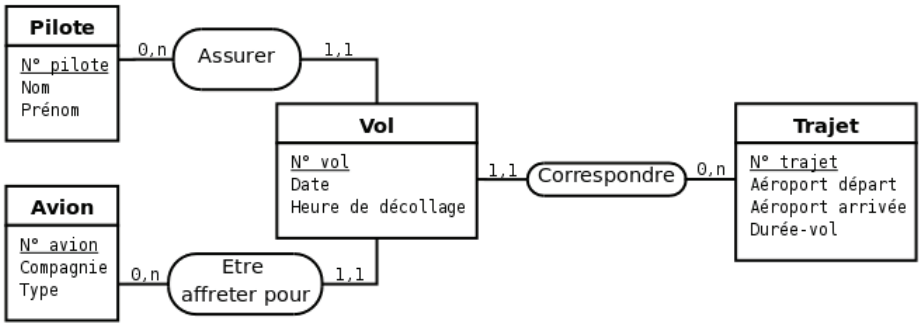

Figure 2.13: Transformation du type-association ternaire de la figure 2.12 en un type-entité et trois type-associations binaires.

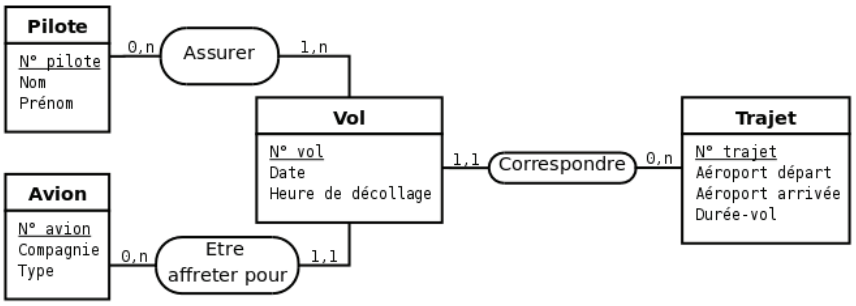

Figure 2.14: Modèle de la figure 2.13 corrigé au niveau des cardinalités.

Passer par cette étape intermédiaire ne comportant pas de type-association n-aire (n>2) peut, dans certains cas, éviter d'introduire un type-association n-aire inapproprié. Imaginons par exemple un type-association ternaire *Vol* liant trois type-entités *Avion*, *Trajet* et *Pilote* comme représenté sur la figure 2.12.
La transformation consistant à supprimer le type-association ternaire du modèle de la figure 2.12 produit le modèle de la figure 2.13. Ce modèle fait immédiatement apparaître une erreur de conception qui était jusque là difficile à diagnostiquer : généralement, à un vol donné sont affectés plusieur pilotes (par exemple le commandant de bord et un copilote) et non pas un seul.
Le modèle correct modélisant cette situation est celui de la figure 2.14 où le type-entité *Vol* ne peut être transformé en un type-association ternaire *Vol* comme sur la figure 2.12.

2.4 Règles de bonne formation d'un modèle E-A

La *bonne formation* d'un modèle entités-associations permet d'éviter une grande partie des sources d'incohérences et de redondance. Pour être bien formé, un modèle entités-associations doit respecter certaines règles et les type-entités et type-associations doivent être normalisées. Un bon principe de conception peut être formulé ainsi : **« une seule place pour chaque fait »**.
Bien que l'objectif des principes exposés dans cette section soit d'aider le concepteur à obtenir un diagramme entités-associations bien formé, ces principes ne doivent pas être interprété comme des lois. Qu'il s'agisse des règles de bonne formation ou des règles de normalisation, il peut exister, très occasionnellement, de bonnes raisons pour ne pas les appliquer.

2.4.1 Règles portant sur les noms

Règle 21 *Dans un modèle entités-associations, le nom d'un type-entité, d'un type-association ou d'un attribut doit être unique.*

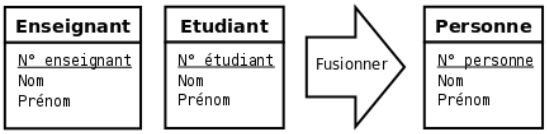

Figure 2.15: La présence des deux type-entités *Enseignant* et *Etudiant* est symptomatique d'une modélisation inachevée. A terme, ces deux type-entités doivent être fusionnés en un unique type-entité *Personne*. Référez vous à la règle 25 pour plus de précisions concernant cette erreur de modélisation.

Figure 2.16: Ici, les attributs *Adresse de facturation* sont redondants. Cette situation doit être évitée à tout prix car elle entraîne un gaspillage d'espace mémoire mais aussi et **surtout un grand risque d'incohérence**. En effet, que faire si, dans le cadre d'une occurrence du type-association *Correspondre*, la valeurs des deux attributs *Adresse de facturation* diffèrent ?

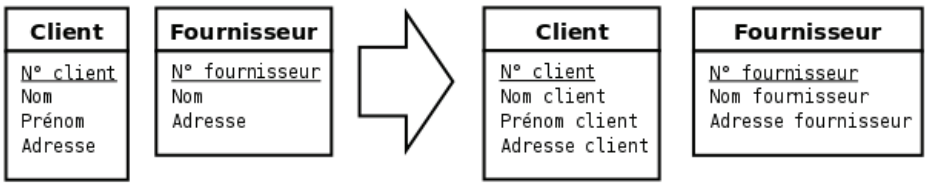

Figure 2.17: Dans cette situation, les deux attributs Adresse doivent simplement être renommés en Adresse client et Adresse fournisseur. Il en va de même pour les deux attributs Nom.

Lorsque des attributs portent le même nom, c'est parfois le signe d'une modélisation inachevée (figure 2.15) ou d'une redondance (figure 2.16). Sinon, il faut simplement ajouter au nom de l'attribut le nom du type-entité ou du type-association dans lequel il se trouve (figure 2.17). Il faut toutefois remarquer que le dernier cas décrit n'est pas rédhibitoire et que les SGDB Relationnel s'accommodent très bien de relations comportant des attributs de même nom. L'écriture des requêtes sera tout de même plus lisible si les attributs ont tous des noms différents.

2.4.2 Règles de normalisation des attributs

Règle 22 *Il faut remplacer un attribut multiple en un type-association et un type-entité supplémentaires.*

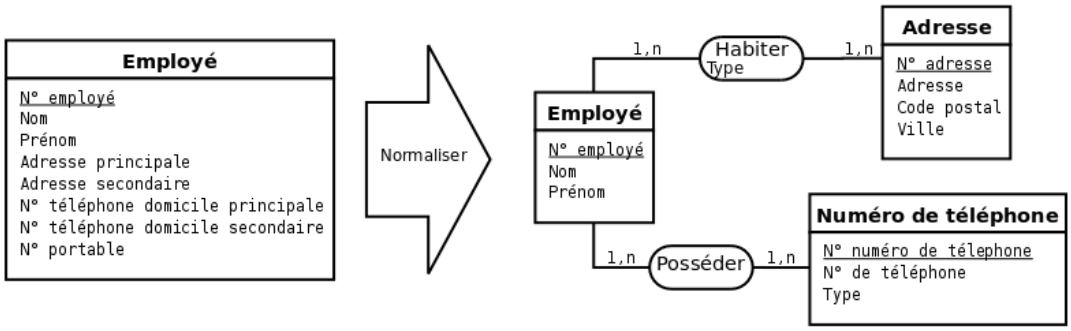

Figure 2.18: Remplacement des attributs multiples en un type-association et un type-entité et décomposition des attributs composites.

En effet, les attributs multiples posent régulièrement des problèmes d'évolutivité du modèle. Par exemple, sur le modèle de gauche de la figure 2.18, comment faire si un employé possède deux adresses secondaires ou plusieurs numéros de portable ?

Il est également intéressant de décomposer les attributs composites comme l'attribut *Adresse* par exemple. Il est en effet difficile d'écrire une requête portant sur la ville où habitent les employés si cette information est noyée dans un unique attribut *Adresse*.

Règle 23 *Il ne faut jamais ajouter un attribut dérivé d'autres attributs, que ces autres attributs se trouvent dans le même type-entité ou pas.*

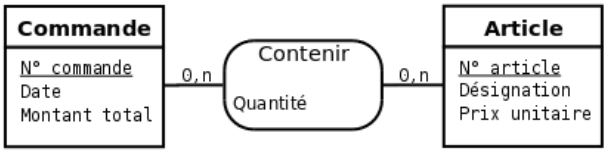

Figure 2.19: Il faut supprimer l'attribut *Montant total* du type-entité *Commande* car on peut le calculer à partir des attributs *Quantité* du type association *Contenir* et *Prix unitaire* du type-entité *Article*.

En effet, les attributs dérivés induisent un risque d'incohérence entre les valeurs des attributs de base et celles des attributs dérivés. La figure 2.19 illustre le cas d'un attribut *Montant total* dans un type-entité *Commande* qui peut être calculé à partir des attributs *Quantité* du type association *Contenir* et *Prix unitaire* du type-entité *Article*. Il faut donc supprimer l'attribut *Montant total* dans le type-entité *Commande*. D'autres attributs dérivés sont également à éviter comme l'âge, que l'on peut déduire de la date de naissance et de la date courante. Il faut cependant faire attention aux pièges : par exemple, le code postal ne détermine ni le numéro de département ni la Ville[3]
Comme nous l'avons déjà dit (cf. règle 12), les attributs d'un type-association doivent dépendre directement des identifiants de *tous* les type-entités de la collection du type-association.

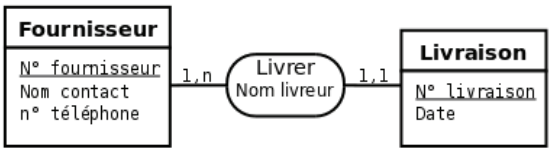

Figure 2.20: Comme la cardinalité maximale du type-association *Livrer* est 1 du côté du type-entité *Livraison*, l'attribut *Nom livreur* de *Livrer* doit être déplacé dans *Livraison*.

Par exemple, sur la figure 2.19, l'attribut *Quantité* du type-association *Contenir* dépend bien à la fois de l'identifiant *N° commande* et de *N° article* des type-entités de la collection de *Contenir*. Inversement, sur cette même figure, l'attribut *Prix-unitaire* ne dépend que de *N° article* du type-entité *Article*, il ne pourait donc pas être un attribut du type-association *Contenir*. Une conséquence immédiate de cette règle est qu'un type association dont la cardinalité maximale de l'une des pattes est 1 ne peut pas posséder d'attribut. Si elle en possédait, ce serait une erreur de modélisation et il faudrait les déplacer dans le type-entité connecté à la patte portant la cardinalité maximale de 1 (cf. figure 2.20).
Règle 24 *Un attribut correspondant à un type énuméré est généralement avantageusement remplacé par un type-entité.*
Par exemple, sur la figure 2.21, l'attribut *Type* caractérise le type d'une émission et peut prendre des valeurs comme : *actualité*, *culturelle*, *reportage*, *divertissement*, etc. Remplacer cet attribut par un type-entité permet, d'une part, d'augmenter la cohérence (en s'affranchissant, par exemple, des variations du genre *culturelle*, *culture*, *Culture*, …) et d'autre part, si les cardinalités le permettent, de pouvoir affecter plusieurs types à une même entité (ex : *actualité* et *culturelle*)

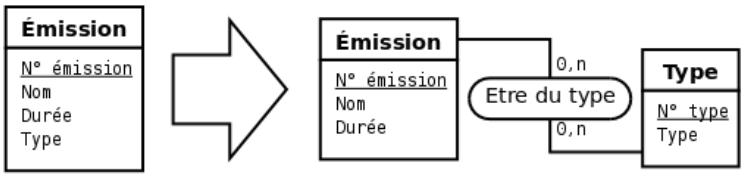

Figure 2.21: Un attribut correspondant à un type énuméré est généralement avantageusement remplacé par un type-entité..

2.4.3 Règles de fusion/suppression d'entités/associations

Règle 25 *Il faut factoriser les type-entités quand c'est possible.*

La spécialisation du type-entité obtenu peut se traduire par l'introduction d'un attribut supplémentaire dont l'ensemble des valeurs possibles est l'ensemble des noms des type-entités factorisés (figure 2.22).

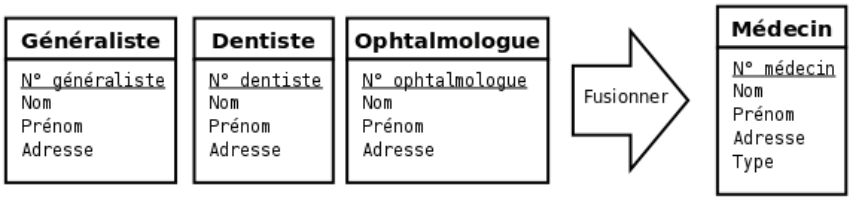

Figure 2.22: Il faut factoriser les type-entités quand c'est possible, éventuellement en introduisant un nouvel attribut.

Mais l'introduction d'un attribut supplémentaire n'est pas forcément nécessaire ou souhaitable. Par exemple, sur le modèle entités-associations final de la figure 2.23, on peut distinguer les entités qui correspondent à des écrivains ou des abonnés en fonction du type de l'association, *Ecrire* ou *Emprunter*, que l'entité en question entretient avec une entité du type *Livre*. Ne pas introduire d'attribut permet en outre de permettre à une personne d'être à la fois un *Abonné* et un *Écrivain*.

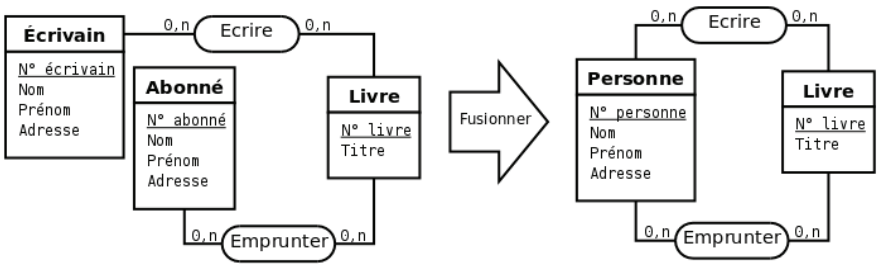

Figure 2.23: Il faut factoriser les type-entités quand c'est possible, mais l'introduction d'un attribut supplémentaire n'est pas toujours nécessaire. Remarque : ce diagramme est intentionnellement simplifié à outrance.

Règle 26 *Il faut factoriser les type-associations quand c'est possible.*
Cette règles est le pendant pour les type-associations de la règle 25 qui concerne les type-entités. La spécialisation du type-association obtenu peut se traduire par l'introduction d'un attribut supplémentaire dont l'ensemble des valeurs possibles est l'ensemble des noms des type-associations factorisés.

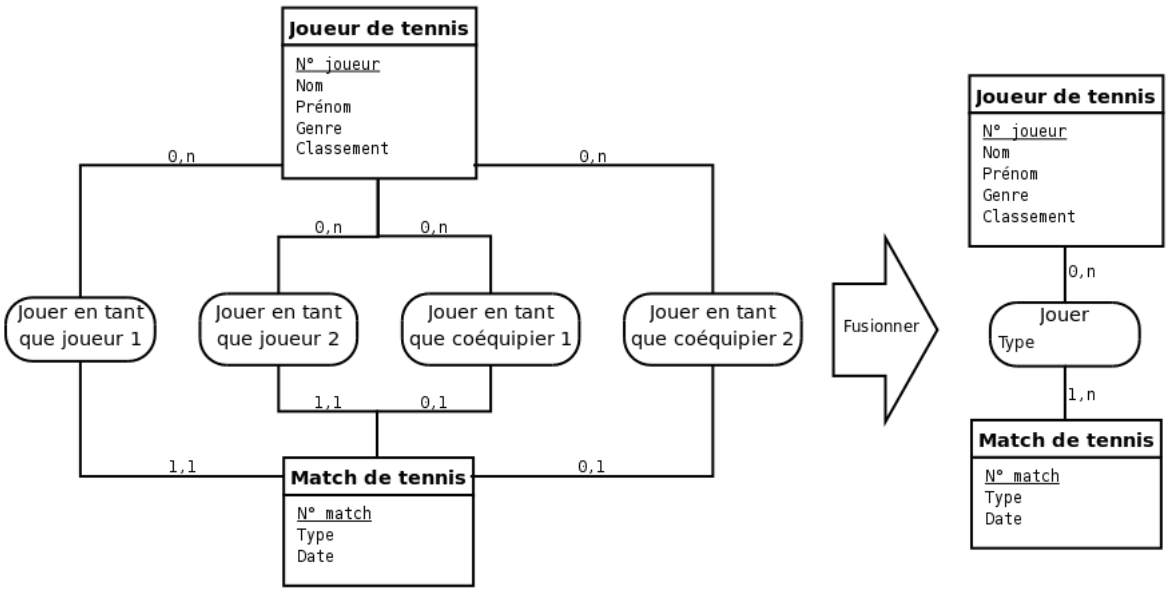

Figure 2.24: Un seul type-association suffit pour remplacer les quatre type-associations *Jouer en tant que ...*

La figure 2.24 montre un exemple de multiplication inutile de type-associations.

Règle 27 *Un type-entité remplaçable par un type-association doit être remplacé.*

Par exemple, le type-entité *Projection* de la figure 2.11 page ?? doit être remplacé par le type-association ternaire *Projeter* pour aboutir au schéma de la figure 2.10 page ??.

Règle 28 *Lorsque les cardinalités d'un type-association sont toutes 1,1 c'est que le type-association n'a pas lieu d'être.*

Il faut aussi se poser la question de l'intérêt du type-association quand les cardinalités maximale sont toutes de 1.

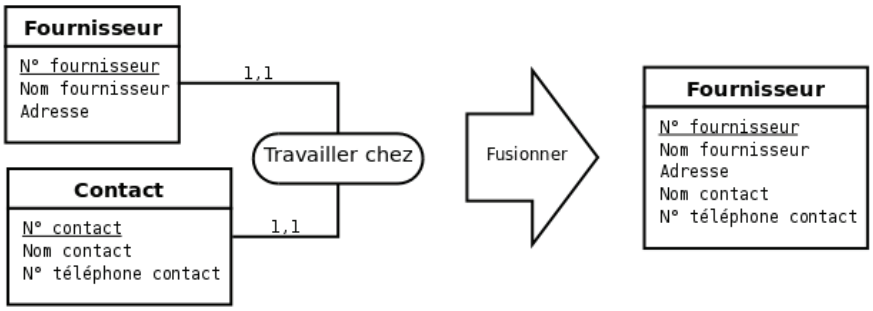

Figure 2.25: Lorsque les cardinalités d'un type-association sont toutes 1,1 c'est qu'il s'agit d'un type-association fantôme.

Lorsque les cardinalités d'un type-association sont toutes 1,1, le type-association doit généralement être supprimé et les type-entités correspondant fusionnés comme l'illustre la figure 2.25. Néanmoins, même si toutes ses cardinalités maximale sont de 1, il est parfois préférable de ne pas supprimer le type-association, comme dans l'exemple de la figure 2.26.

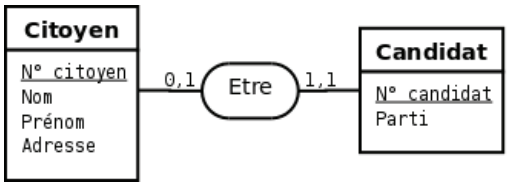

Figure 2.26: Même si toutes les cardinalités maximale sont de 1, il vaut mieux conserver le type-association *Etre*.

Règle 29 *Il faut veiller à éviter les type-associations redondants. En effet, s'il existe deux chemins pour se rendre d'un type-entité à un autre, alors ces deux chemins doivent avoir deux significations ou deux durées de vie distinctes. Dans le cas contraire, il faut supprimer le chemin le plus court puisqu'il est déductible des autres chemins.*

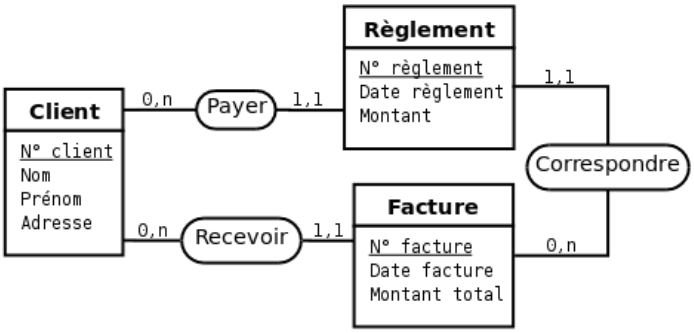

Figure 2.27: Si un client ne peut pas régler la facture d'un autre client, alors le type-association *Payer* est inutile.

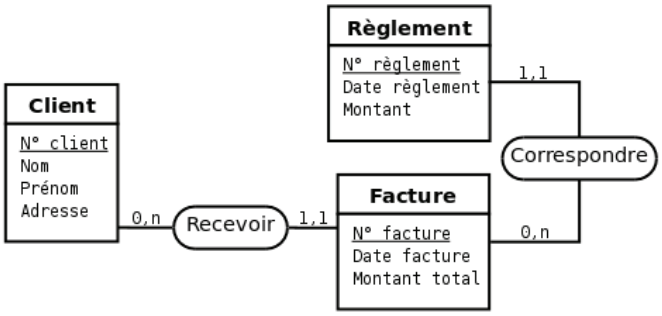

Figure 2.28: Solution au problème de la redondance du type-association de la figure 2.27.

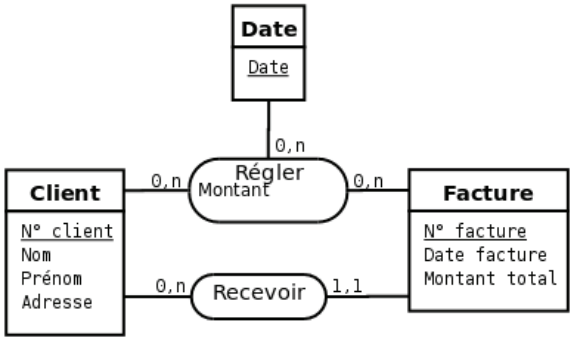

Figure 2.29: Dans le modèle de la figure 2.27, si un client peut régler la facture d'un autre client, il faut remplacer le type-entité *Règlement* par un type-association *Régler*.

Par exemple, dans le modèle représenté sur la figure 2.27, si un client ne peut pas régler la facture d'un autre client, alors le type-association *Payer* est redondant et doit purement et simplement être supprimé du modèle (cf. figure 2.28). On pourra toujours retrouver le client qui a effectué un règlement en passant par la facture correspondante.
Par contre, si un client peut régler la facture d'un autre client, alors c'est la règle 27 qu'il faut appliquer : on remplace le type-entité *Règlement* par un type-association *Régler* (cf. figure 2.29).

2.4.4 Normalisation des type-entités et type-associations

Introduction

Les formes normales sont différent stades de qualité qui permettent d'éviter la redondance, source d'anomalies. La normalisation peut être aussi bien effectuée sur un modèle entités-associations, où elle s'applique sur les type-entités et type-associations, que sur un modèle relationnel.
Il existe 5 formes normales principales et deux extensions. Plus le niveau de normalisation est élevé, plus le modèle est exempte de redondances. Un type-entité ou un type-association en forme normale de niveau n est automatiquement en forme normale de niveau $n-1$. Une modélisation rigoureuse permet généralement d'aboutir directement à des type-entités et type-associations en forme normale de Boyce-Codd.
Nous avons décidé de présenter deux fois cette théorie de la normalisation :

- Une première fois, dans le cadre du modèle entités-associations (la présente section 2.4.4), en privilégiant une approche plus intuitive qui n'introduit pas explicitement la notion de dépendance fonctionnelle (et encore moins les notions de dépendance multivaluée et de jointure). Nous nous arrêterons, dans cette section, à la forme normale de Boyce-Codd.
- Puis une seconde fois, dans le cadre de modèle relationnel (section 3.2), en privilégiant une approche plus formelle s'appuyant sur la définition des dépendances fonctionnelle, multivaluée et de jointure. Nous irons alors jusqu'à la cinquième forme normale.

Première forme normale (1FN)

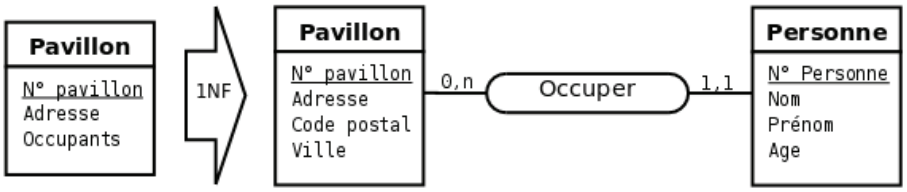

Figure 2.30: Exemple de normalisation en première forme normale.

Définition 30 ***-Première forme normale (1FN)-*** *Un type-entité ou un type-association est en première forme normale si tous ses attributs sont élémentaires, c'est-à-dire non décomposables.*

Un attribut composite doit être décomposés en attributs élémentaires (comme l'attribut *Adresse* sur la figure 2.30) ou faire l'objet d'une entité supplémentaire (comme l'attribut *Occupants* sur la figure 2.30.

L'*élémentarité* d'un attribut est toutefois fonction des choix de gestion. Par exemple, la propriété *Adresse* peut être considérée comme élémentaire si la gestion de ces adresses est globale. Par contre, s'il faut pouvoir considérer les codes postaux, les noms de rues, …, il convient d'éclater la propriété *Adresse* en *Adresse* (au sens numéro d'appartement, numéro et nom de rue, …), *Code postal* et *Ville*. En cas de doute, il est préférable (car plus général) d'éclater une propriété que d'effectuer un regroupement.

Deuxième forme normale (2FN)

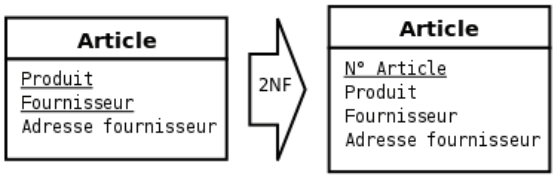

Figure 2.31: Exemple de normalisation en deuxième forme normale. On suppose qu'un même fournisseur peut fournir plusieurs produits et qu'un même produit peut être fourni par différents fournisseurs.

Définition 31 ***-Deuxième forme normale (2FN)-*** *Un type-entité ou un type-association est en deuxième forme normale si, et seulement si, il est en première forme normale et si tout attribut n'appartenant pas à la clé dépend de la totalité de cette clé.*

Autrement dit, les attributs doivent dépendre de l'ensemble des attributs participant à la clé. Ainsi, si la clé est réduite à un seul attribut, ou si elle contient tous les attributs, le type-entité ou le type-association est, par définition, forcément en deuxième forme normale.

La figure 2.31 montre un type-entité *Article* décrivant des produits provenant de différents fournisseurs. On suppose qu'un même fournisseur peut fournir plusieurs produits et qu'un même produit peut être fourni par différents fournisseurs. Dans ce cas, les attributs *Produit* ou *Fournisseur* ne peuvent constituer un identifiant du type-entité *Article*. Par contre, le couple *Produit/Fournisseur* constitue bien un identifiant du type-entité *Article*. Cependant, l'attribut *Adresse fournisseur* ne dépend maintenant que d'une partie de la clé (*Fournisseur*). Opter pour une nouvelle clé arbitraire réduite à un seul attribut *N° article* permet d'obtenir un type-entité *Article* en deuxième forme normale. On va voir dans ce qui suit que cette solution n'a fait que déplacer le problème.

Troisième forme normale (3FN)

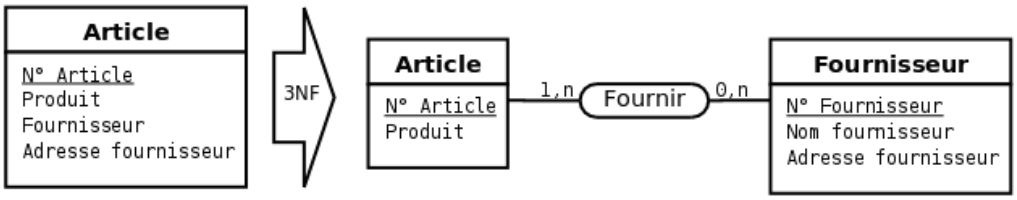

Figure 2.32: Exemple de normalisation en troisième forme normale. Dans cet exemple, l'attribut *Adresse fournisseur* dépend de l'attribut *Fournisseur*.

Définition 32 ***-Troisième forme normale (3FN)-*** *Un type-entité ou un type-association est en troisième forme normale si, et seulement si, il est en deuxième forme normale et si tous ses attributs dépendent*

directement de sa clé et pas d'autres attributs.
Cette normalisation peut amener à désimbriquer des type-entités cachées comme le montre la figure 2.32.
Un type-entité ou un type-association en deuxième forme normale avec au plus un attribut qui n'appartient pas à la clé est, par définition, forcément en troisième forme normale.

Forme normale de Boyce-Codd (BCNF)

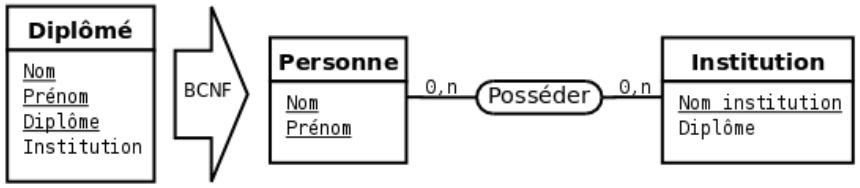

Figure 2.33: Exemple de normalisation en forme normale de Boyce-Codd.

Définition 33 ***-Forme normale de Boyce-Codd (BCNF)-*** *Un type-entité ou un type-association est en forme normale de Boyce-Codd si, et seulement si, il est en troisième forme normale et si aucun attribut faisant partie de la clé dépend d'un attribut ne faisant pas partie de la clé.*

Intéressons-nous, par exemple (cf. figure 2.33), à un type-entité *Diplômé* modélisant des personnes (*Nom* et *Prénom*) possédant un diplôme (*Diplôme*) d'une institution (*Institution*). On suppose qu'il n'y a pas d'homonyme, qu'une même personne ne possède pas deux fois le même diplôme mais qu'elle peut posséder plusieurs diplômes différents. Une institution ne délivre qu'un type de diplôme, mais un même diplôme peut être délivré par plusieurs institutions (par exemple, plusieurs écoles d'ingénieurs délivrent des diplômes d'ingénieur). Une clé possible pour le type-entité *Diplômé* est donc *Nom, Prénom, Diplôme*. Le type-entité obtenu est en troisième forme normale, mais une redondance subsiste car l'attribut *Institution* détermine l'attribut *Diplôme*. Le type-entité *Diplômé* n'est donc pas en forme normale de Boyce-Codd.
Un modèle en forme normale de Boyce-Codd est considéré comme étant de qualité suffisante pour une implantation.

Autres formes normales

Il existe d'autres formes normales. La quatrième et la cinquième forme normale sont présentées dans la section 3.2 dans le cadre du modèle relationnel.

3
Le code postal en France identifie le bureau distributeur qui achemine le courrier dans une commune. En conséquence et d'après cette définition, il n'existe pas de relation entre le code postal et le code du département de la commune. Par exemple, la commune *La Feuillade*, dont le code postal est 19600, est située dans le département de la *Dordogne* (24). Dans cette non correspondance entre code postal et département, il y a toute la Corse ! Il n'y a pas non plus de correspondance biunivoque entre le code postal et une ville. Une commune peut avoir plusieurs codes postaux, un code postal peut recouvrir plusieurs communes.

2.5 Élaboration d'un modèle entités-associations

2.5.1 Étapes de conceptions d'un modèle entités-associations

Pour concevoir un modèle entités-associations, vous devrez certainement passer par une succession d'étapes. Nous les décrivons ci-dessous dans l'ordre chronologique. Sachez cependant que la conception d'un modèle entités-associations est un travail non linéaire. Vous devrez régulièrement revenir à une étape précédente et vous n'avez pas besoin d'en avoir terminé avec une étape pour commencer l'étape suivante.

Recueil des besoins –

C'est une étape primordiale. Inventoriez l'ensemble des données à partir des documents de l'entreprise, d'un éventuel cahier des charges et plus généralement de tous les supports de l'information. N'hésitez pas à poser des questions.

Tri de l'information –

Faites le tri dans les données recueillies. Il faut faire attention, à ce niveau, aux problèmes de synonymie/polysémie. En effet, les attributs ne doivent pas être redondants. Par exemple, si dans le langage de l'entreprise on peut parler indifféremment de *référence d'article* ou de *n° de produit* pour désigner la même chose, cette caractéristique ne devra se concrétiser que par un unique attribut dans le modèle.

Inversement, on peut parler d'adresse pour désigner l'adresse du fournisseur et l'adresse du client, le contexte permettant de lever l'ambiguïté. Par contre, dans le modèle, il faudra veiller à bien distinguer ces deux caractéristiques par deux attributs distincts.
Un autre exemple est celui d'une entreprise de production fabricant des produits à destination d'une autre société du même groupe. Il se peut que dans ce cas, le prix de production (*i.e.* le coût de revient industriel) soit le même que prix de vente (aucune marge n'est réalisée). Même dans ce cas où les deux caractéristiques sont identiques pour chaque entité (prix de production égale prix de vente), il faut impérativement les scinder en deux attributs au niveau du type-entité *Produit*. Sinon, cette égalité factuelle deviendrait une contrainte imposée par le modèle, obligeant alors l'entreprise de production à revoir son système le jour où elle décidera de réaliser une marge (prix de production inférieure au prix de vente).

Identification des type-entités –
Le repérage d'attributs pouvant servir d'identifiant permet souvent de repérer un type-entité. Les attributs de ce type-entité sont alors les attributs qui dépendent des attributs pouvant servir d'identifiant.
Attention, un même concept du monde réel peut être représenté dans certains cas comme un attribut et dans d'autres cas comme un type-entité, selon qu'il a ou non une existence propre. Par exemple, la marque d'une automobile peut être vue comme un attribut du type-entité *Véhicule* de la base de données d'une préfecture mais aussi comme un type-entité *Constructeur automobile* dans la base de données du Ministère de l'Industrie.
Lorsqu'on ne parvient pas à trouver d'identifiant pour un type-entité, il faut se demander s'il ne s'agit pas en fait d'un type-association. Si ce n'est pas le cas, un identifiant arbitraire numérique entier peut faire l'affaire.

Identification des type-associations –
Identifiez les type-associations reliant les type-entités du modèle. Le cas échéant, leur affecter les attributs correspondant.
Il est parfois difficile de faire un choix entre un type-entité et un type-association. Par exemple, un mariage peut être considéré comme un type-association entre deux personnes ou comme un type-entité pour lequel on veut conserver un numéro, une date, un lieu, …, et que l'on souhaite manipuler en tant que tel.
Étudiez également les cardinalités des type-associations retenus. Lorsque toutes les pattes d'un type-association portent la cardinalité 1,1, il faut se demander si ce type-association et les type-entités liés ne décrivent pas en fait un seul type-entité (cf. règle 2.25).
Vérification du modèle –
Vérifiez que le modèle respecte bien les règles que nous avons énoncés et les définitions concernant la normalisation des type-entités et des type-associations. Le cas échéant, opérez les modifications nécessaires pour que le modèle soit bien formé.
Remarque :
pour faciliter la lecture du schéma, il est assez courant de ne pas y faire figurer les attributs ou de ne conserver que ceux qui font partie des identifiants. Les attributs cachés doivent alors absolument être spécifiés dans un document à part.

2.5.2 Conseils divers
Concernant le choix des noms
Pour les type-entités, choisissez un nom commun décrivant le type-entité (ex : Étudiant, Enseignant, Matière). Certain préfèrent mettre le nom au pluriel (ex : Étudiants, Enseignants, Matières). Restez cependant cohérents, soit tous les noms de type-entité sont au pluriel, soit ils sont tous au singulier.
Pour les type-association, choisissez un verbe à l'infinitif, éventuellement à la forme passive ou accompagné d'un adverbe (ex : Enseigner, Avoir lieu dans).
Pour les attributs, utilisez un nom commun au singulier éventuellement accompagné du nom du type-entité ou du type-association dans lequel il se trouve (ex : nom de client, numéro d'article).
Concernant le choix des identifiants des type-entités
Évitez les identifiants composés de plusieurs attributs (comme, par exemple, un identifiant formé par les attributs *nom* et *prénom* d'un type-association *Personne*) car :
- ils dégradent les performances du SGBD,
- mais surtout l'unicité supposée par une telle démarche finit généralement, tôt ou tard, par être

démentie !
Évitez les identifiants susceptibles de changer au cours du temps (comme la plaque d'immatriculation d'un véhicule).
Évitez les identifiants du type chaîne de caractère.
En fait, il est souvent préférable de choisir un identifiant arbitraire de type entier pour les type-entités. Cet identifiant deviendra une clé primaire dans le schéma relationnel et le SGBD l'incrémentera automatiquement lors de la création de nouvelles instances. L'inconvénient de cette pratique est qu'il devient possible de se retrouver avec deux instances du type-entités représentant le même objet mais avec deux numéros différents. Malgré cette inconvénient, cette politique de l'identifiant reste largement avantageuse dans la pratique et permet, en outre, de s'affranchir (en la satisfaisant automatiquement) de la deuxième forme normale (cf. section 2.4.4).
Bien distinguer les concepts de données et de traitements