

SIO

Introduction aux bases de données

Table des matières

1. Schéma conceptuel.....	4
Le modèle hiérarchique.....	4
Le modèle réseaux sémantiques.....	4
Le modèle entité/association.....	5
Le modèle objet.....	5
2. Modèle conceptuel de données (MCD).....	5
a) Schéma entités-associations.....	5
Entités et associations.....	5
Attributs et identifiants.....	6
Cardinalités.....	7
Associations plurielles.....	8
Association réflexive.....	9
Associations non binaires.....	10
b) Règles de normalisation.....	12
Les bonnes pratiques dans un schéma entités-associations.....	12
Les formes normales.....	17
c) Dépendances fonctionnelles.....	19
Définitions et propriétés.....	19
Graphe de couverture minimale.....	20
Traduction vers un schéma entités-associations.....	21
Gestion des dates et du caractère historique.....	22
Dépendances plurielles et réflexives.....	24
Associations sans attributs.....	25
d) Méthodologie de base.....	26
3. Modèle logique de données (MLD).....	27
a) Systèmes logiques.....	27
b) Modèle logique relationnel.....	28
Tables, lignes et colonnes.....	28
Clés primaires et clés étrangères.....	28
Schéma relationnel.....	29
c) Traduction d'un MCD en un MLDR.....	29
4. Modèle physique de données (MPD).....	34
a) Distinction entre MLD et MPD.....	34
b) Optimisations.....	34
5. Rétro-conception.....	35
a) Traduction inverse.....	35
b) Cas particuliers.....	36

<u>6.Compléments</u>	37
<u>a)Agrégation</u>	37
<u>Association de type 1:n</u>	37
<u>Association de type n:m</u>	39
<u>Tables de codification ou tables de référence</u>	41
<u>b)Identifiant relatif ou lien identifiant</u>	42
<u>Résolution d'un problème sur le schéma relationnel</u>	42
<u>Modèle conceptuel correspondant</u>	43
<u>Discussion autour de la numérotation des exemplaires</u>	45
<u>c)Héritage</u>	46
<u>Sous-entité</u>	46
<u>Utilisation de l'héritage pour séparer les informations complémentaires</u>	47
<u>Spécialisation des associations</u>	49
<u>7.Conclusion</u>	50
<u>Annexe</u>	51
<u>8.Index des illustrations</u>	51

1. Schéma conceptuel

De manière générale, un schéma conceptuel est une représentation d'un ensemble de concepts reliés sémantiquement entre eux. Les concepts sont connectés par des lignes fléchées auxquelles sont accolés des mots.

La relation entre les concepts s'appuie sur des termes exprimant cette relation: « appartient à », « passe par », « mène à », « prévient que », « favorise »...

Le schéma conceptuel a pour buts de représenter le modèle mental d'une situation, que cette représentation soit personnelle, celle d'un groupe ou d'une organisation ;

Il est la représentation graphique qui sert à décrire le fonctionnement d'une base de données et représente ainsi les objets principaux contenus dans cette dernière, le modèle réseaux sémantiques, leurs caractéristiques et les relations qui s'établissent entre ces différents objets.

Cette représentation est normée suivant une modélisation bien définie.

Plusieurs types de schémas conceptuels existent, correspondants aux différents types de base de données que l'on peut rencontrer :

- le modèle hiérarchique ;
- le modèle réseaux sémantiques ;
- le modèle entité / association ;
- le modèle objet ;
- ...

Le modèle hiérarchique

L'information est organisée de manière arborescente (selon une hiérarchie), accessible uniquement à partir de la racine de cette arborescence.

Il permet de facilement ajouter des éléments et d'en modifier la structure.

Le défaut principal de cette représentation provient du fait que le point d'accès à l'information est bien unique (c'est la racine de l'arbre hiérarchique). Cela entraîne que toute recherche de donnée demande le parcours de toute, ou au moins partie, de la hiérarchie, en passant par la racine.

L'exemple le plus courant est celui de l'arborescence utilisé pour les systèmes de fichiers (bien qu'il ne s'agit pas de [SGBD](#) à proprement parler, comme nous le verront plus loin), où l'on navigue depuis la racine du système vers des sous répertoires et des fichiers contenant les données.

Le modèle réseaux sémantiques

Ce modèle décrit le fonctionnement d'une base de données réseau. Ce type de base de données fonctionne sur le principe du regroupement des différents éléments de la base de données par leur sens. Toutes les informations peuvent y être associées les unes aux autres et servir de point d'accès. Cela en fait un modèle complexe dont l'extraction de données est difficile.

L'exemple le plus concret d'un tel modèle est le [web sémantique](#).

Le modèle entité/association

Le modèle entité-association est un type de schéma conceptuel qui est notamment utilisé par les bases de données relationnelles.

Les points d'entrées y sont indépendant de la structure de la base de données, l'utilisateur saisit simplement une requête, sans avoir à naviguer dans la base, et la machine se charge de l'exécuter pour atteindre les données escomptés.

Le principal inconvénient est la rigidité de la structure du modèle définie et les difficultés que cela entraîne pour faire évoluer un schéma existant lorsqu'un besoin nouveau apparaît ou qu'une erreur de conception est découverte.

Cela inclus notamment tous les systèmes se basant sur le langage [SQL](#).

Le modèle objet

Dans le cadre de ce modèle, comme dans la [programmation orientée objet](#), les données sont décrites comme des classes et représentées sous forme d'objet possédant des attributs et des méthodes.

Chaque objet peut servir de point d'entrée pour d'autres objets, permettant une navigation à la fois souple et aisée dans la base.

L'inconvénient principal est l'importante quantité de ressources (mémoire et temps de calcul) que nécessite cette approche.

2. Modèle conceptuel de données (MCD)

Le MCD permet de modéliser la problématique à traiter d'un point de vue conceptuel et donc indépendamment du logiciel utilisé.

a) Schéma entités-associations

Entités et associations

Une entité est une population d'individus homogènes. Par exemple, les produits ou les articles vendus par une entreprise peuvent être regroupés dans une même entité articles, car d'un article à l'autre, les informations ne changent pas de nature (à chaque fois, il s'agit de la désignation, du prix unitaire, etc.).



Illustration 1: Entités

Par contre, les articles et les clients ne peuvent pas être regroupés : leurs informations ne sont pas homogènes (un article ne possède pas d'adresse et un client ne possède pas de prix unitaire).

Il faut donc leur réserver deux entités distinctes : l'entité articles et l'entité clients.

Une association est une liaison qui a une signification précise entre plusieurs entités. Dans notre exemple, l'association *commander* est une liaison évidente entre les entités articles et clients, tandis que l'association *livrer* établit le lien sémantique entre les entités articles et fournisseurs.

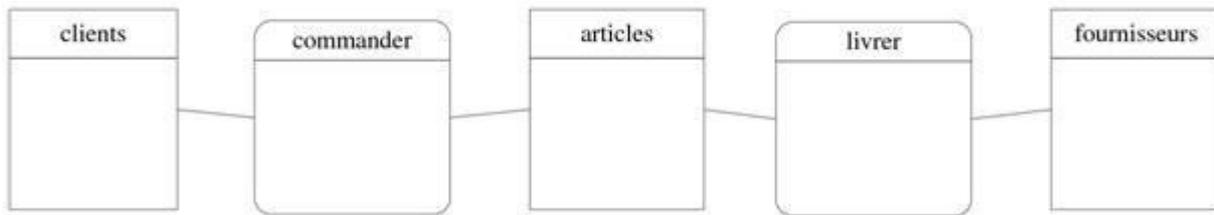


Illustration 2: Associations

Dans ce schéma, les entités *clients* et *fournisseurs* ne sont pas liées directement, mais indirectement, via l'entité *articles*, ce qui est assez naturel.

Attributs et identifiants

Un attribut est une propriété d'une entité ou d'une association.

Le prix unitaire est un attribut de l'entité *articles*, le nom de famille est un attribut de l'entité *clients*, la quantité commandée est un attribut de l'association *commander* et la date de livraison est un attribut de l'association *livrer*.



Illustration 3: Attributs

Une entité et ses attributs ne doivent traiter que d'un seul sujet afin d'assurer une certaine cohérence au modèle. Il est donc préférable de ne pas mettre les informations relatives aux fournisseurs dans l'entité des *articles* mais plutôt dans une entité *fournisseurs* séparées.

Chaque individu d'une entité doit être identifiable de manière unique, c'est pourquoi toutes les entités doivent posséder un attribut **sans doublon** (c'est-à-dire ne prenant pas deux fois la même valeur). Il s'agit de l'identifiant **que l'on souligne sur le schéma**, par convention.

Le numéro de client constitue un identifiant classique pour l'entité clients.

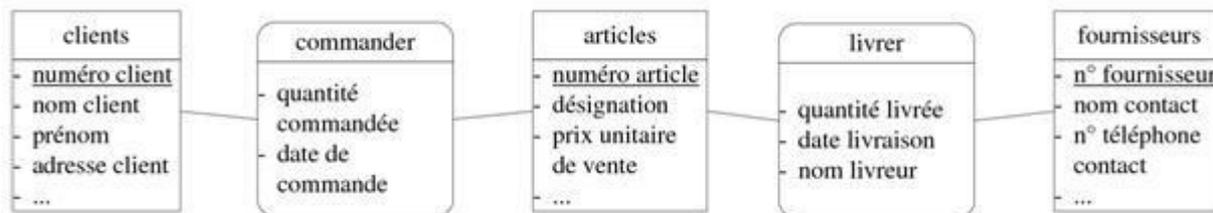


Illustration 4: Identifiants

Remarques :

- une entité possède au moins un attribut (son identifiant) ;
- une association peut être dépourvue d'attribut.

Cardinalités

La cardinalité d'un lien entre une entité et une association précise le minimum et le maximum de fois qu'un individu de l'entité peut être concerné par l'association.

Exemple :

un client a au moins commandé **un** article et peut commander **n** articles (**n** étant indéterminé), tandis qu'un article peut avoir été commandé entre **0** et **n** fois (même si ce n'est pas le même **n** que précédemment).



Illustration 5: Cardinalités

Une cardinalité minimale de **1** doit se justifier par le fait que les individus de l'entité en question ont besoin de l'association pour exister.

En effet, un client n'existe pas avant d'avoir commandé quoique ce soit, donc la cardinalité minimale de l'entité clients dans l'association **commander** est **1**.

Dans tous les autres cas, la cardinalité minimale vaut **0**, car c'est le cas de la liste préétablie d'articles.

La discussion autour d'une cardinalité minimale 0 ou 1 n'est vraiment intéressante que lorsque la cardinalité maximale est 1. Nous verrons en effet lors de la traduction vers un schéma relationnel, que lorsque la cardinalité maximale est **n**, nous ne pouvons pas faire la différence entre une cardinalité minimale de **0** et une cardinalité minimale de **1**.

Nota Bene :

Un article peut être commandé par plusieurs clients car tous les crayons rouges (par exemple) ne sont pas numérotés individuellement, mais portent un numéro d'article **collectif**.

En toute rigueur, notre entité articles aurait du s'appeler *types d'article*. Ainsi, un crayon rouge peut être commandé par plusieurs clients, ce n'est simplement pas le même crayon à chaque fois.

Il s'agit d'un choix de modélisation, le lecteur peut très légitimement faire le choix inverse qui consiste à numéroté individuellement chaque crayon rouge.

La seule difficulté pour établir correctement les cardinalités est de se poser les questions dans le bon sens.

Si on prend l'exemple de l'association commander :

- côté clients, la question est « un client peut commander combien d'articles ? » et la réponse est « entre 1 et plusieurs » ;
- côté articles, la question est « un article peut être commandé par combien de client ? » et cette fois-ci, la réponse est « entre 0 et plusieurs ».

Associations plurielles

Deux mêmes entités peuvent être plusieurs fois en association.

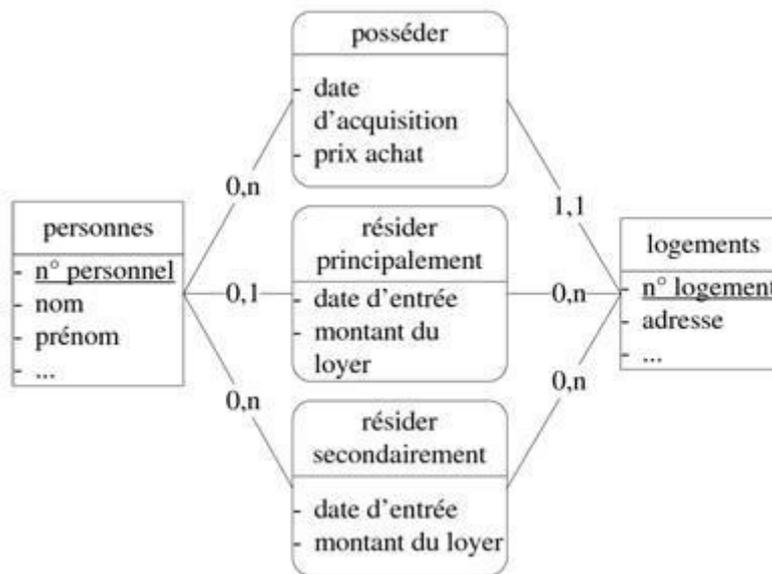


Illustration 6: Associations plurielles

Dans cet exemple issu d'une agence immobilière, une personne peut être propriétaire, résider principalement ou résider secondairement dans un logement géré par l'agence. Les logements qui ne sont pas gérés par l'agence ne figurent pas dans l'entité des logements, ce qui explique certaines cardinalités 0 du schéma. Nous supposons également qu'un logement n'est détenu que par une seule personne et que ce propriétaire figure obligatoirement dans l'entité des personnes.

Association réflexive

Il est permis à une association d'être branchée plusieurs fois à la même entité, comme par exemple l'association binaire réflexive.

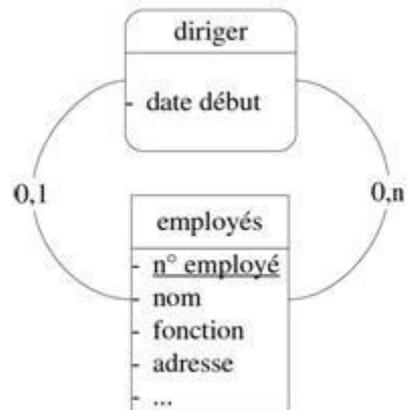


Illustration 7: Association réflexive

Dans cet exemple, tout employé est dirigé par un autre employé (sauf le directeur général) et un employé peut diriger plusieurs autres employés, ce qui explique les cardinalités sur le schéma.

Associations non binaires

Lorsqu'autour d'une entité, toutes les associations ont pour cardinalités maximales 1 au centre et n à l'extérieur, cette entité est candidate pour être remplacée par une association branchée à toutes les entités voisines avec des cardinalités identiques $0,n$.

La deuxième condition qu'il faut impérativement satisfaire est la règle de normalisation des attributs des associations. Cette règle conduit parfois à l'apparition d'associations qui établissent un lien sémantique entre 3 entités ou plus.

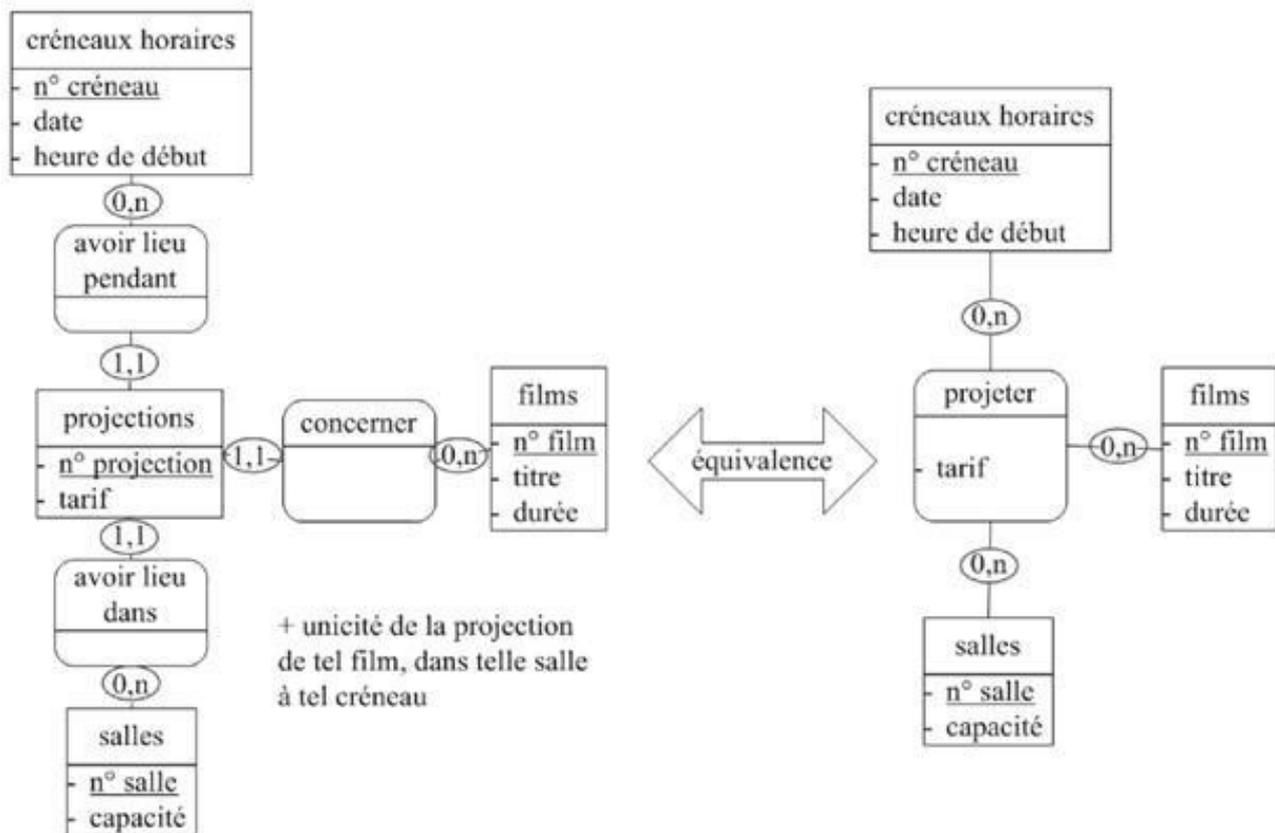


Illustration 8: Entité remplaçable par une association ternaire

Sur l'exemple précédent, l'entité projections est uniquement entourée d'associations dont les cardinalités maximales sont 1 côté projections et n de l'autre côté.

De plus, la donnée d'un créneau, d'un film et d'une salle suffit à déterminer une projection unique. On peut donc la remplacer par une association projeter branchée aux trois entités salles, créneaux horaires et films. On parle alors d'association ternaire.

La difficulté de concevoir une association ternaire (ou plus) directement est d'établir les bonnes cardinalités. Il est donc conseillé d'en passer par un schéma entités-associations dans lequel on ne trouve que des associations binaires, puis de repérer les entités remplaçables par des associations, comme sur la illustration 8 à gauche.

Cette règle de conduite permet d'éviter d'introduire une association ternaire abusive, par exemple entre les avions, les pilotes et les vols (illustration 9), car le concepteur peut s'apercevoir que l'une des cardinalités maximales ne convient pas.

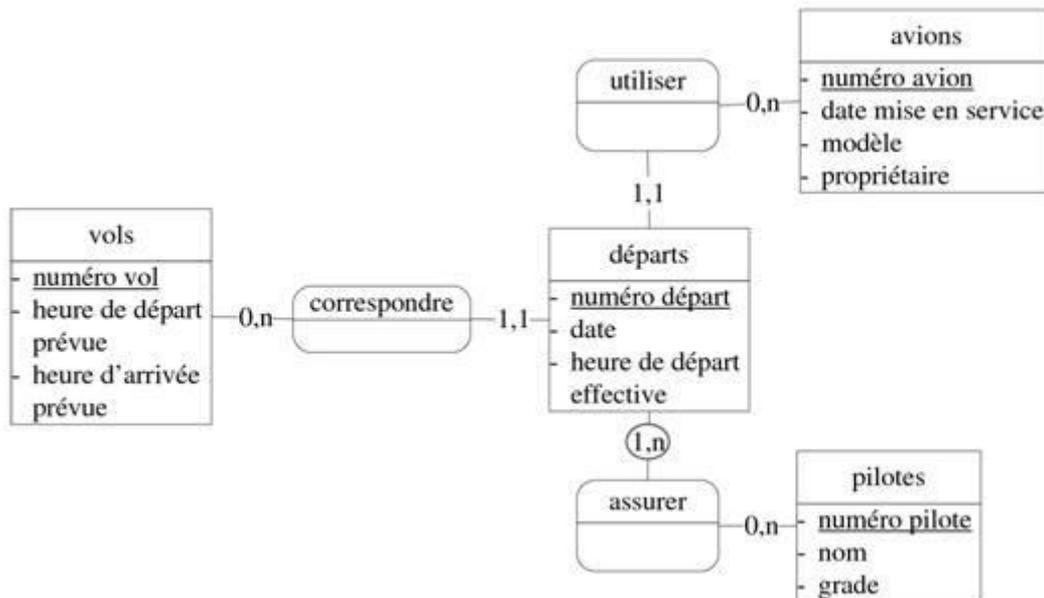


Illustration 9: Contre-exemple : l'entité départs n'est pas remplaçable par une association ternaire

Par ailleurs, une association peut être branchée à plus de trois entités, comme sur la illustration 10. Là encore, le conseil pour être sûr de la légitimité de cette association quaternaire, est de vérifier les cardinalités sur un schéma intermédiaire faisant apparaître à la place, une entité occupations et quatre associations binaires.

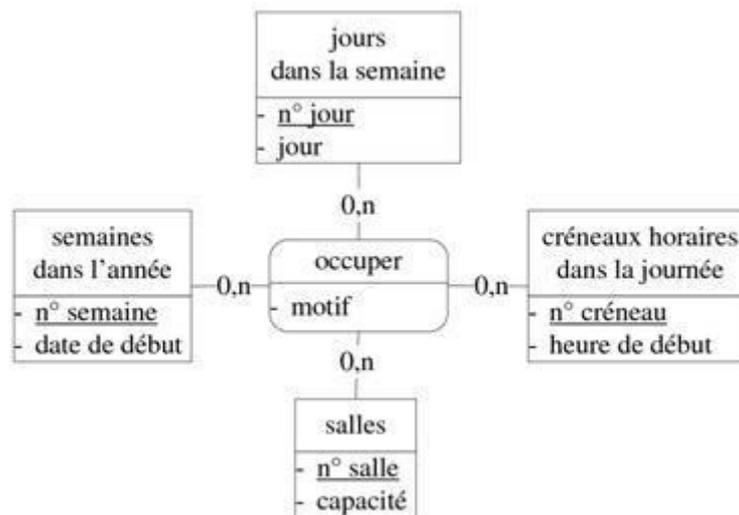


Illustration 10: Exemple d'entité quaternaire

b) Règles de normalisation

Un bon schéma entités-associations doit répondre à 9 règles de normalisation, que le concepteur doit connaître par cœur.

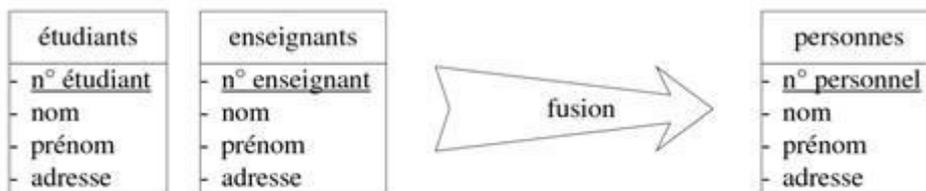
Les bonnes pratiques dans un schéma entités-associations

1. **Normalisation des entités** (importante) : toutes les entités qui sont remplaçables par une association doivent être remplacées (comme sur la illustration 8).
2. **Normalisation des noms** : le nom d'une entité, d'une association ou d'un attribut doit être unique.

Conseils :

- pour les entités, utiliser un nom commun au pluriel (par exemple : clients) ;
- pour les associations, utiliser un verbe à l'infinitif (par exemple : effectuer, concerner) éventuellement à la forme passive (être commandé) et accompagné d'un adverbe (avoir lieu dans, pendant, à) ;
- pour les attributs, utiliser un nom commun singulier (par exemple : nom, numéro, libellé, description), éventuellement accompagné du nom de l'entité ou de l'association dans laquelle il se trouve (par exemple : nom de client, numéro d'article).

Remarque : lorsqu'il reste plusieurs fois le même nom, c'est parfois symptomatique d'une modélisation qui n'est pas terminée, comme sur la illustration 11(a), ou le signe d'une redondance comme sur la illustration 11(b).



(a) Deux entités homogènes peuvent être fusionnées

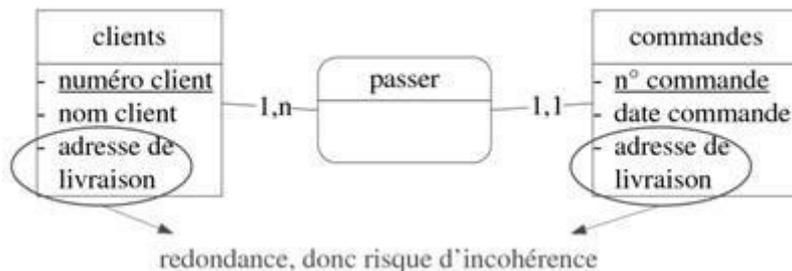


Illustration 11: Contre-exemples de la normalisation des noms

(a) Deux entités homogènes peuvent être fusionnées

(b) Si deux attributs contiennent les mêmes informations, alors la redondance induit non seulement un gaspillage d'espace mais également un risque d'incohérence (les adresses peuvent ne pas correspondre !)

3. **Normalisation des identifiants** : chaque entité doit posséder un identifiant.

Conseils :

- éviter les identifiants composés de plusieurs attributs comme, par exemple, un identifiant formé par les attributs nom et prénom, car d'une part c'est mauvais pour les performances et d'autre part, l'unicité supposée par une telle démarche finit tôt ou tard par être démentie ;
- préférer un identifiant court pour rendre la recherche la plus rapide possible et éviter notamment les chaînes de caractères comme un numéro de plaque d'immatriculation, un numéro de sécurité sociale ou un code postal ;
- éviter également les identifiants susceptibles de changer au cours du temps comme les plaques d'immatriculation ou les numéros de sécurité sociale provisoires.

Conclusion : l'identifiant sur un schéma entités-associations (et donc la future clé primaire dans le schéma relationnel) doit être un entier, de préférence incrémenté automatiquement.

4. **Normalisation des attributs** (importante) : remplacer les attributs en plusieurs exemplaires en une association supplémentaire de cardinalités maximales n et ne 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. Par exemple, sur la illustration 12 à gauche, comment faire si un employé a deux adresses secondaires ? D'autre part, les attributs calculables induisent un risque d'incohérence entre les valeurs des attributs de base et celles des attributs calculés, comme sur la illustration 13.

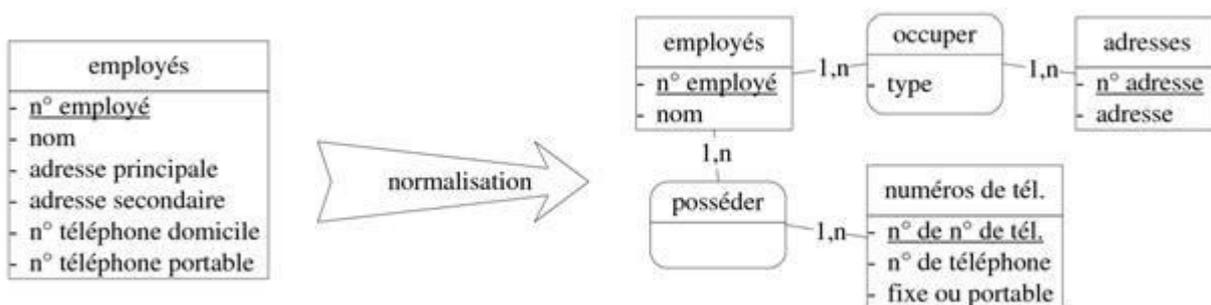


Illustration 12: Contre-exemples de la normalisation des attributs (a)

(a) Attributs en plusieurs exemplaires remplacés par une association supplémentaire

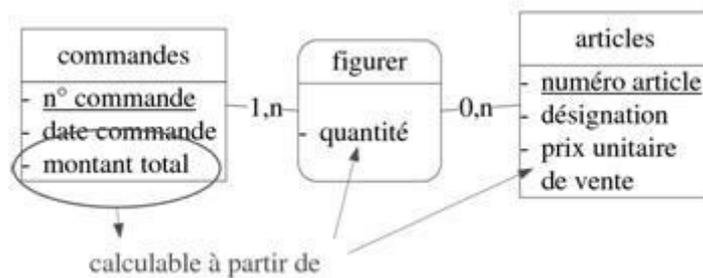


Illustration 13: Contre-exemples de la normalisation des attributs (b)

(b) Attribut calculable qu'il faut retirer du schéma

D'autres d'attributs calculables classiques sont à éviter, comme l'âge (qui est calculable à partir de la date de naissance) ou encore le département (calculable à partir d'une sous-chaîne du code postal).

5. **Normalisation des attributs des associations** (importante) : les attributs d'une association doivent dépendre directement des identifiants de toutes les entités en association.

Par exemple, sur l'illustration 14 la quantité commandée dépend à la fois du numéro de client et du numéro d'article, par contre la date de commande non. Il faut donc faire une entité commandes à part, idem pour les livraisons.

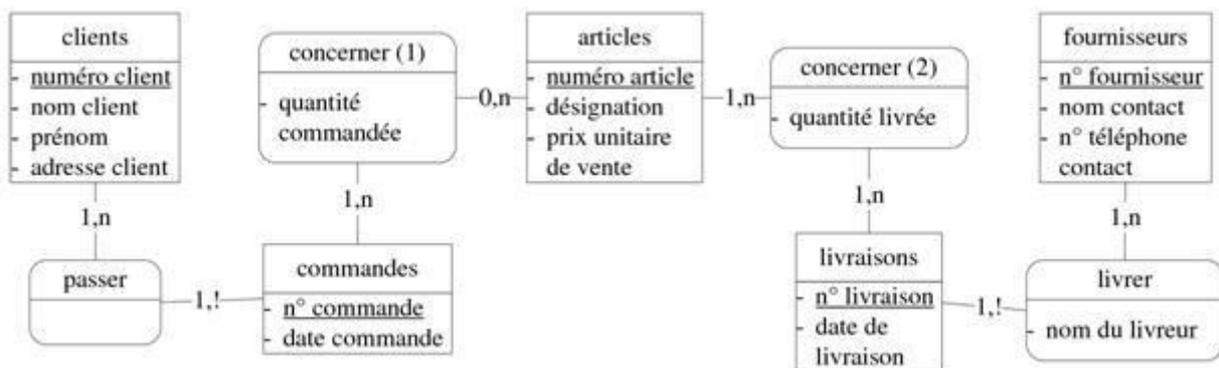


Illustration 14: Normalisation des attributs des associations

L'inconvénient de cette règle de normalisation est qu'elle est difficile à appliquer pour les associations qui ne possèdent pas d'attribut. Pour vérifier malgré tout qu'une association sans attribut est bien normalisée, on peut donner temporairement à cette association un attribut imaginaire (mais pertinent) qui permet de vérifier la règle.

Par exemple, entre les entités livres et auteurs de la illustration 16, l'association écrire ne possède pas d'attribut. Imaginons que nous ajoutons un attribut pourcentage qui contient le pourcentage du livre écrit par chaque auteur (du même livre). Comme cet attribut pourcentage dépend à la fois du numéro de livre et du numéro d'auteur, l'association écrire est bien normalisée.

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 (illustration 15).

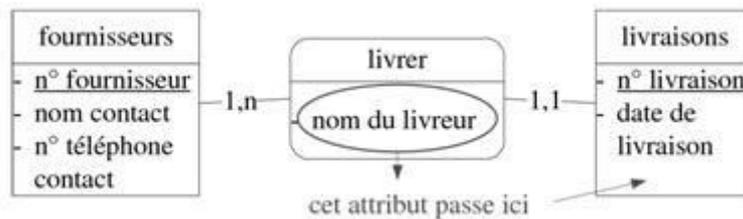


Illustration 15: Cardinalité 1,1 et attributs d'une association

6. **Normalisation des associations** (importante) : il faut éliminer les associations fantômes (illustration 16), redondantes ou en plusieurs exemplaires (illustration 17).

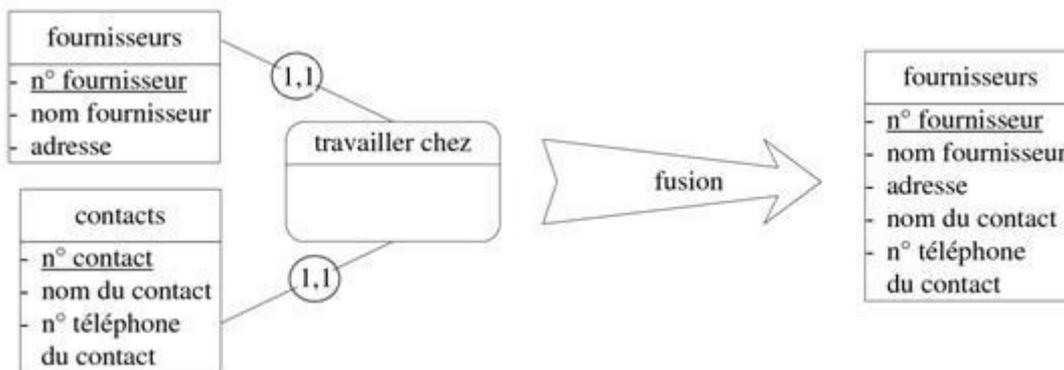


Illustration 16: les cardinalités sont toutes 1,1 donc c'est une association fantôme

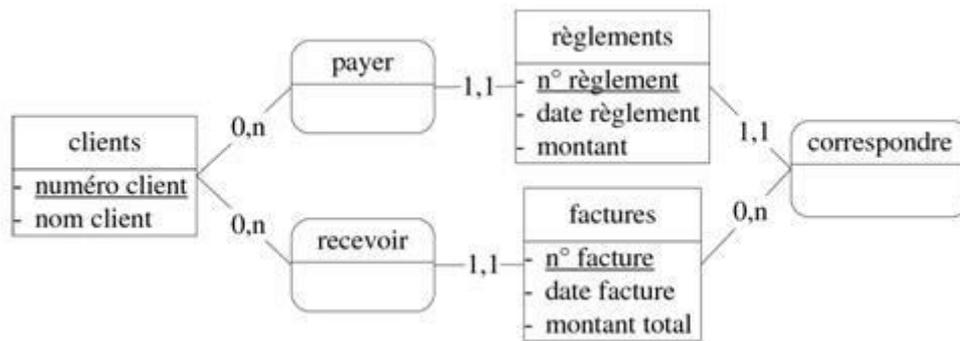


Illustration 17: si un client ne peut pas régler la facture d'un autre client, alors l'association payer est inutile et doit être supprimée

En ce qui concerne les associations redondantes, cela signifie que s'il existe deux chemins pour se rendre d'une entité à une autre, alors ils doivent avoir deux significations ou deux durées de vie différentes. Sinon, il faut supprimer le chemin le plus court, car il est déductible à partir de l'autre chemin. Dans notre exemple de la illustration 17, si on supprime l'association payer, on peut retrouver le client qui a payé le règlement en passant par la facture qui correspond.

Remarque : une autre solution pour le problème de la illustration 17 consiste à retirer l'entité règlements et d'ajouter une association régler avec les mêmes attributs (sauf l'identifiant) entre les entités clients et factures.

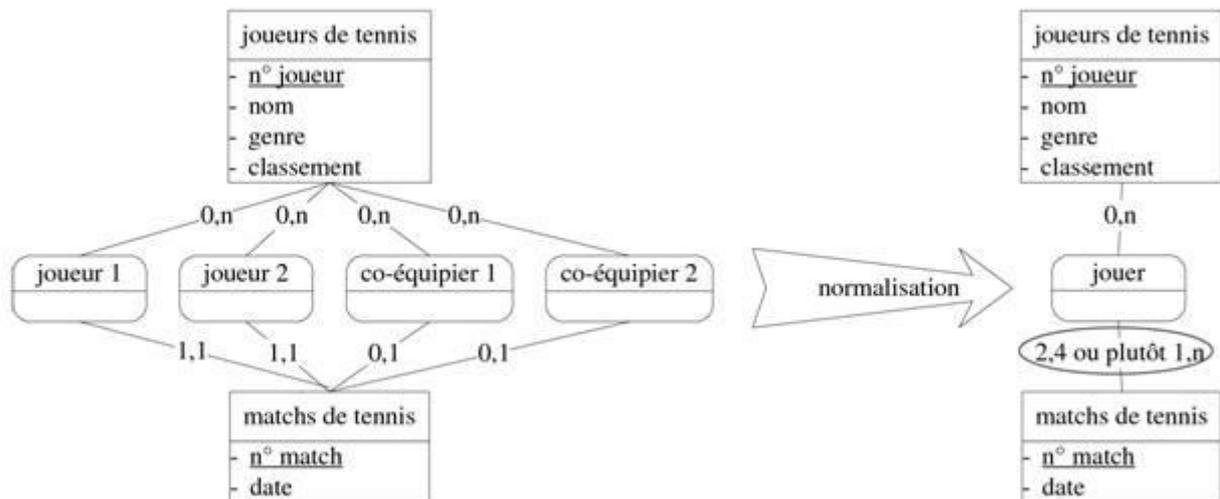


Illustration 18: une association suffit pour remplacer les 4 associations participer en tant que ...

- 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 (comme sur la illustration 18 à droite, ou pour un nombre limité d'emprunts dans une bibliothèque), alors nous considérons quand même qu'elle est indéterminée 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.

Les formes normales

À ces 6 règles de normalisation, il convient d'ajouter les 3 premières formes normales traditionnellement énoncées pour les schémas relationnels, mais qui trouvent tout aussi bien leur place en ce qui concerne les schémas entités-associations.

8. **Première forme normale** : à un instant donné dans une entité, pour un individu, un attribut ne peut prendre qu'une valeur et non pas, un ensemble ou une liste de valeurs.

Si un attribut prend plusieurs valeurs, alors ces valeurs doivent faire l'objet d'une entité supplémentaire, en association avec la première (illustration 19).

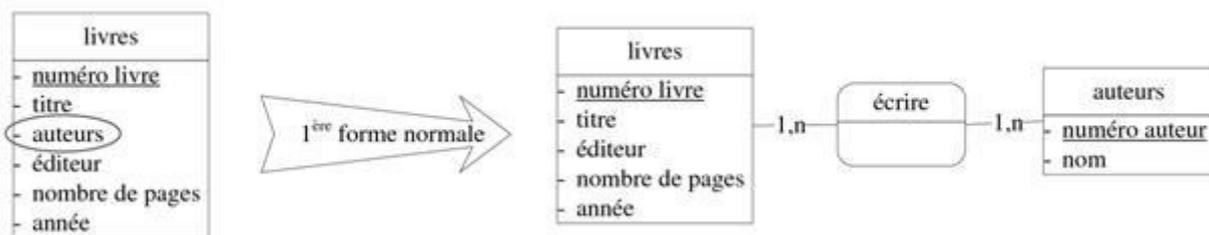


Illustration 19: Première forme normale : il peut y avoir plusieurs auteurs pour un livre donné

9. **Deuxième forme normale** : l'identifiant peut être composé de plusieurs attributs mais les autres attributs de l'entité doivent dépendre de l'identifiant en entier (et non pas une partie de cet identifiant).

Cette deuxième forme normales peut être oubliée si on suit le conseil de n'utiliser que des identifiants non composés et de type entier. En vérité, elle a été vidée de sa substance par la règle de [normalisation des attributs des associations](#).

10. **Troisième forme normale de Boyce-Codd** (importante) : tous les attributs d'une entité doivent dépendre directement de son identifiant et d'aucun autre attribut. Si ce n'est pas le cas, il faut placer l'attribut pathologique dans une entité séparée, mais en association avec la première.

numéro avion	constructeur	modèle	capacité	propriétaire
1	Airbus	A380	180	Air France
2	Boeing	B747	314	British Airways
3	Airbus	A380	180	KLM

Illustration 20: Il y a redondance dans les colonnes constructeur et capacité

Par exemple, l'entité avions (illustration 21 à gauche) dont les valeurs sont données dans l'illustration 20, n'est pas en troisième forme normale de Boyce-Codd, car la capacité et le constructeur d'un avion ne dépendent pas du numéro d'avion mais de son modèle.

La solution améliorée est donnée dans l'illustration 21 à droite.

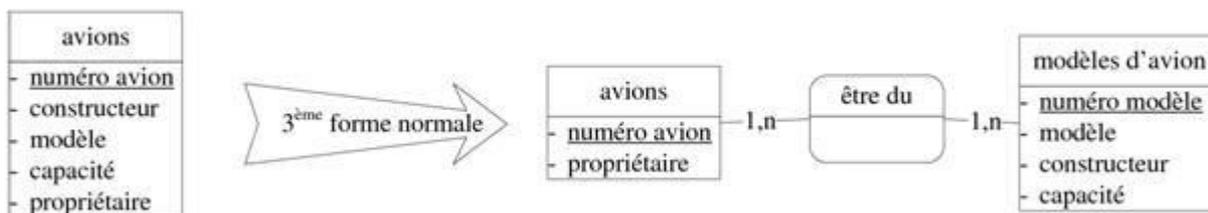


Illustration 21: Application de la troisième forme normale de Boyce-Codd

En toute rigueur, la colonne constructeur ne doit pas être maintenue dans l'entité modèles d'avion, mais doit être placée dans une entité séparée constructeurs (en association avec modèles d'avion), afin d'éviter la redondance du nom d'un constructeur pour tous ses modèles.

c) *Dépendances fonctionnelles*

Pour établir efficacement un modèle entités-associations bien normalisé, on peut étudier au préalable les dépendances fonctionnelles entre les attributs puis, les organiser en graphe de couverture minimale. Cette technique est traditionnellement employée pour normaliser des schémas relationnels, mais elle s'applique très bien en amont, au niveau des modèles conceptuels.

Définitions et propriétés

Un attribut Y dépend fonctionnellement d'un attribut X si et seulement si une valeur de X induit une unique valeur de Y. On note une dépendance fonctionnelle par une flèche simple : $X \rightarrow Y$.

Par exemple, si X est le numéro de client et Y le nom de client, alors on a bien $X \rightarrow Y$.

Par contre, on a pas $Y \rightarrow X$, car plusieurs clients de numéros différents peuvent porter le même nom.

Transitivité : si $X \rightarrow Y$ et $Y \rightarrow Z$ alors $X \rightarrow Z$.

Par exemple, on a numéro de commande \rightarrow numéro de client \rightarrow nom de client, donc on a aussi numéro de commande \rightarrow nom de client.

Mais la dépendance fonctionnelle numéro de commande \rightarrow nom de client est dite transitive, car il faut passer par le numéro de client pour l'obtenir.

Au contraire, la dépendance fonctionnelle numéro de client \rightarrow nom de client est directe. Seules les dépendances fonctionnelles directes nous intéressent. D'autres exemples sont donnés dans l'illustration 22.

dépendance fonctionnelle	directe ?
numéro de livraison \rightarrow date de livraison	oui
numéro de livraison \rightarrow numéro du fournisseur	oui
numéro du fournisseur \rightarrow nom du fournisseur	oui
numéro de livraison \rightarrow nom du fournisseur	non

Illustration 22: Exemples de dépendances fonctionnelles

Un attribut Y peut avoir une dépendance fonctionnelle qui repose sur la conjonction de plusieurs attributs, auquel cas la dépendance est dite non élémentaire. Les dépendances fonctionnelles non élémentaires sont notées par une flèche unique mais comportant plusieurs points d'entrée (regroupés autour d'un cercle).

Par exemple, la quantité commandée (d'un article dans une commande) dépend de deux attributs : le numéro de commande et le numéro d'article (illustration 23). Notons que cette dépendance numéro de commande + numéro d'article → quantité est à la fois non élémentaire et directe.

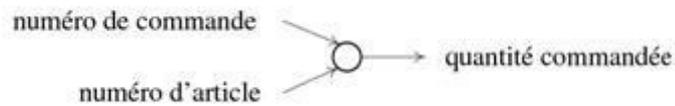


Illustration 23: *Dépendance fonctionnelle non élémentaire, mais directe*

Graphe de couverture minimale

En représentant tous les attributs et toutes les dépendances fonctionnelles directes entre eux, nous obtenons un réseau appelé graphe de couverture minimale. Dans notre exemple sur les clients, les commandes et les articles, ce graphe est donné sur l'illustration 24.

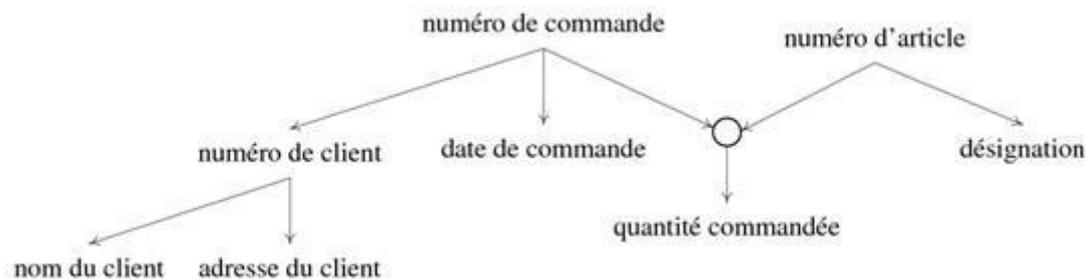


Illustration 24: *Graphe de couverture minimale*

La technique de traduction en un schéma entités-associations qui suit, suppose qu'aucun attribut n'a été oublié sur le graphe de couverture minimal et notamment, aucun identifiant.

D'ailleurs toutes les dépendances fonctionnelles du graphe doivent partir d'un identifiant. Si ce n'est pas le cas, c'est qu'un identifiant a été omis.

Traduction vers un schéma entités-associations

À partir du graphe de couverture minimale (illustration 24), le schéma entités-associations normalisé correspondant apparaît naturellement (l'illustration 25), en suivant quelques étapes simples.

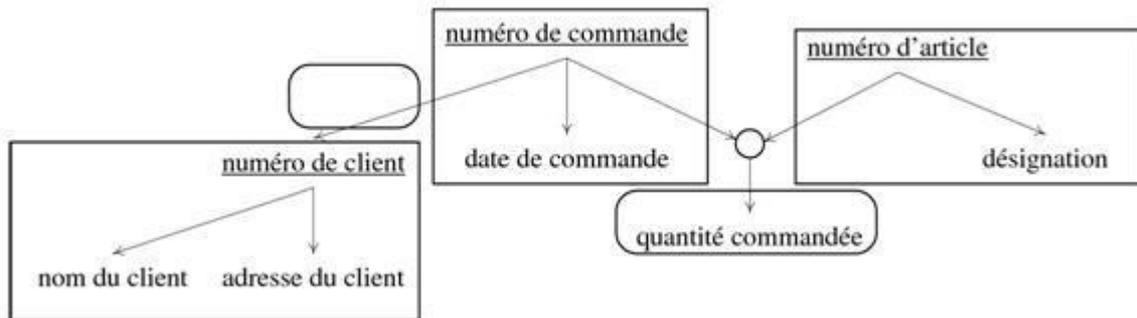


Illustration 25: Identification des entités et des associations sur un graphe de couverture minimale

Étape 1 : il faut repérer et souligner les identifiants.

Étape 2 : tous les attributs non identifiant qui dépendent directement d'un identifiant et d'un seul, forment une entité (avec l'identifiant, bien sûr).

Étape 3 : les dépendances élémentaires entre les identifiants forment des associations binaires dont les cardinalités maximales sont 1 au départ de la dépendance fonctionnelle et n à l'arrivée.

Étape 4 : sauf si entre deux identifiants se trouvent deux dépendances élémentaires réflexives, auquel cas l'association binaire a deux cardinalités maximales valant 1.

Étape 5 : les attributs (non identifiants) qui dépendent de plusieurs identifiants sont les attributs d'une association supplémentaire dont les cardinalités maximales sont toutes n.

La traduction du graphe de couverture minimale de l'illustration 25 en un schéma entités-associations normalisé est donnée sur l'illustration 26 .

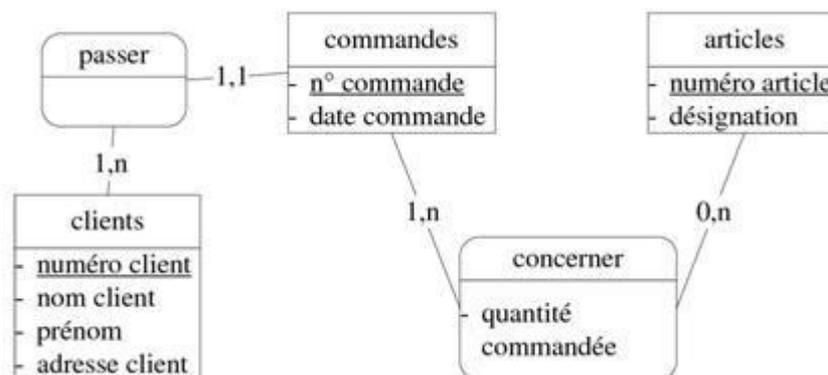


Illustration 26: Schéma entités-associations normalisé obtenu à partir du graphe de couverture minimale

Dans ce genre de traduction, il faut donner un nom aux entités et aux associations, car ce n'est pas le cas sur le graphe de couverture minimale et il reste les cardinalités minimales à établir.

Remarquons également qu'en réalité, il faut déjà connaître les entités en présence pour établir correctement le graphe de couverture minimale, ne serait-ce que pour y faire figurer leurs identifiants. Donc finalement, cette technique n'est une aide pour établir les associations entre les entités et pour normaliser les entités et leurs associations (jusqu'en troisième forme normale de Boyce-Codd).

Gestion des dates et du caractère historique

Dans une bibliothèque, on peut vouloir stocker les emprunts en cours (illustration 27) et/ou les emprunts historiques (illustration 27).

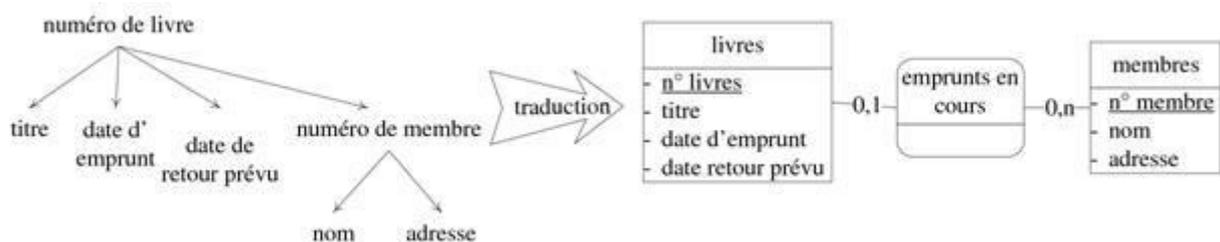


Illustration 27: *Sans historisation des emprunts, pas de problème*

Pour les emprunts en cours, la date de retour prévu est un attribut de l'entité livres car un livre ne peut faire l'objet que d'un seul emprunt en cours. Dans ce cas, l'établissement du graphe de couverture minimal ne pose aucun problème.

Par contre, un livre peut faire l'objet de plusieurs emprunts historiques et dans ces conditions, la date d'emprunt est déterminante pour connaître la date de retour prévue (illustration 27 en haut à gauche). Or une date n'est pas un identifiant et une dépendance fonctionnelle ne peut partir que d'un ou plusieurs identifiant(s). C'est le signe qu'il manque un identifiant : le numéro d'emprunt (l'illustration 27 en haut à droite).

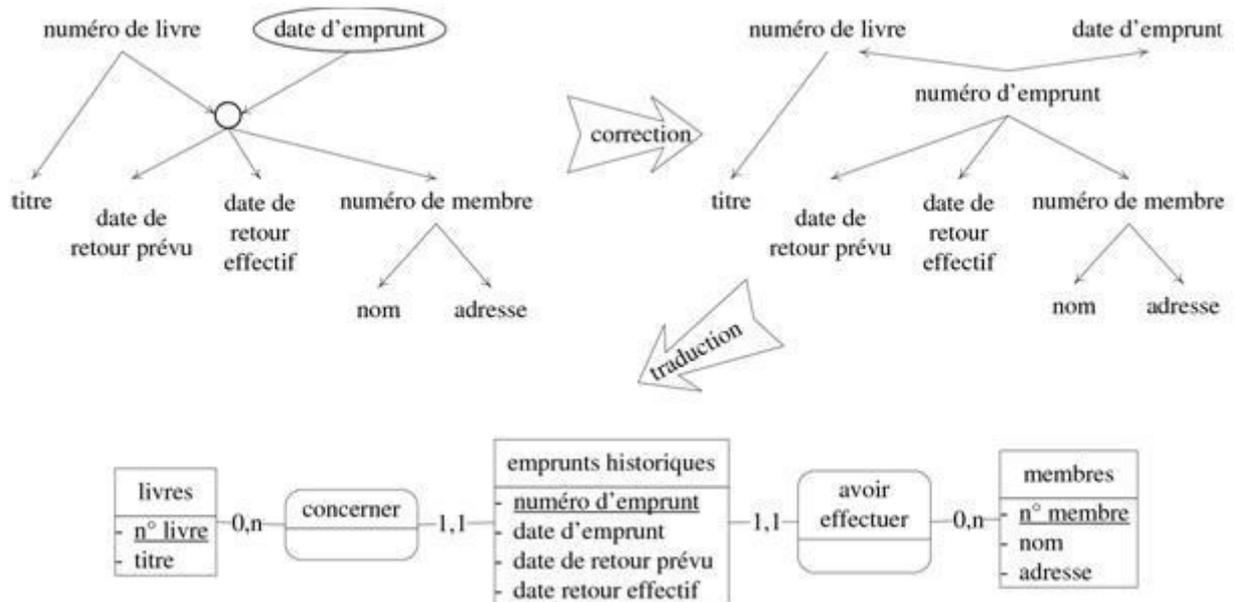


Illustration 28: *Même pour une entité historisée, il faut éviter que la date entre dans l'identifiant*

Notons que l'entité emprunts historiques supplémentaire qui apparaît après traduction (illustration 28 en bas) ne peut pas être transformée en une association comme on pourrait le croire au simple examen des cardinalités qui l'entourent.

En effet, les attributs de l'association qui en résulterait ne vérifieraient pas la normalisation des attributs des associations. Notamment, la date de retour effectif ne dépend pas du numéro de livre et du numéro de membre, mais du numéro de livre et de la date d'emprunt.

La normalisation des entités ne s'applique donc pas aux entités qui ont un caractère historique. À moins que les dates ne soient regroupées dans une entité séparée, ce qui n'est pas conseillé tant qu'aucune information liée aux dates (comme le caractère férié, par exemple) n'est nécessaire.

Dépendances plurielles et réflexives

Une ou plusieurs dépendances fonctionnelles peuvent partir ou arriver plusieurs fois du même attribut. Pour clarifier la signification de chaque dépendance fonctionnelle, on peut ajouter un commentaire sur la flèche (figure 26). Ce commentaire sert ensuite à donner un nom aux associations correspondantes.

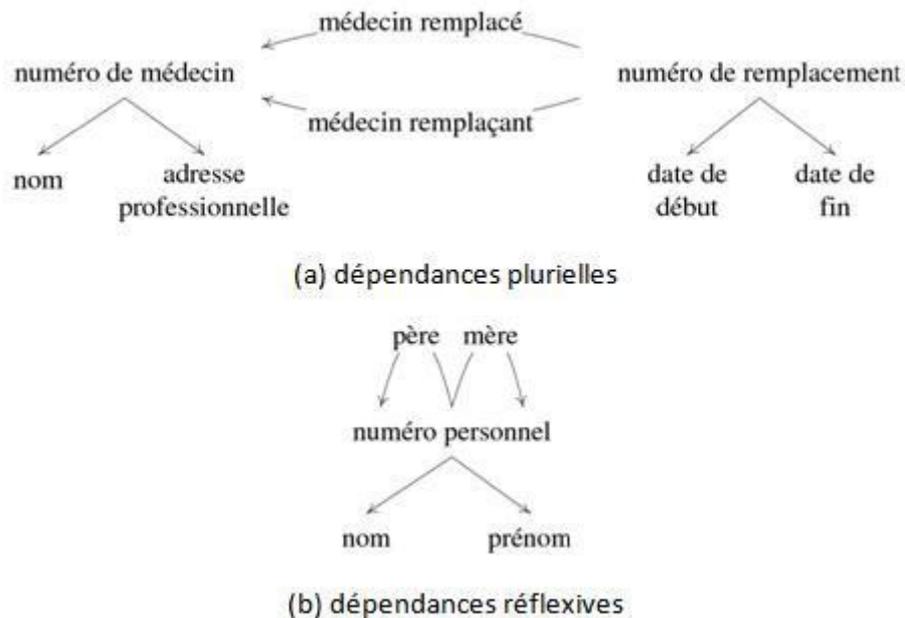


Illustration 29: Dépendances fonctionnelles commentées

Les dépendances fonctionnelles plurielles entre les médecins et les remplacements (illustration 29(a)) deviennent, après traduction, des associations plurielles entre les entités médecins et remplacements. Notons que l'entité remplacements ainsi générée, a aussi un caractère historique.

Les fonctionnelles réflexives ($X \leftrightarrow X$), quoique toujours vraies, ne présentent aucun intérêt, à moins qu'elles aient une signification particulière. Un exemple de dépendance réflexive licite sur un graphe de couverture minimale est la dépendance fonctionnelle personne ? personne, lorsqu'elle signifie « diriger », « être en couple avec » ou « être le père ou la mère de » (illustration 29(b)).

Dans le même ordre d'idée, il est inutile de faire figurer sur le graphe de couverture minimal des dépendances fonctionnelles non élémentaires vraies, mais idiotes, comme par exemple numéro de commande + numéro d'article ? numéro de commande.

Associations sans attributs

La lacune majeure de cette méthode reste tout de même le fait que les associations dont toutes les cardinalités maximales sont n mais qui sont sans attribut ne figurent pas sur le graphe de couverture minimale. Il faut alors, soit leur inventer temporairement un attribut (comme pour la normalisation des attributs des associations), soit introduire une notation spéciale (par exemple, une dépendance non élémentaire qui ne débouche sur aucun attribut).

Pour illustrer ce défaut, prenons l'exemple des films et des acteurs (figure 27).



Illustration 30: *dépendance non élémentaire et sans enfant sur un graphe de couverture minimal*

Il n'y a pas d'attribut qui dépende à la fois du numéro de film et du numéro d'acteur (à moins d'imaginer le temps d'apparition à l'écran). Et pourtant, les deux entités films et acteurs sont en association. Grâce à la dépendance non élémentaire et sans enfant, on peut rendre compte de cette situation sur le graphe de couverture minimale et faire ainsi apparaître l'association sur le schéma entités-associations qui en est traduit.

d) Méthodologie de base

Face à une situation bien définie (soit à travers un énoncé précis, soit à travers une collection de formulaires ou d'états que le nouveau système d'information est censé remplacer), nous pouvons procéder sans établir le graphe de couverture minimale :

- identifier les entités en présence ;
- lister leurs attributs ;
- ajouter les identifiants (numéro arbitraire et auto-incrémenté) ;
- établir les associations binaires entre les entités ;
- lister leurs attributs ;
- calculer les cardinalités ;
- vérifier les règles de normalisation et en particulier, la normalisation des entités (c'est à ce stade qu'apparaissent les associations non binaires), des associations et de leurs attributs ainsi que la troisième forme normale de Boyce-Codd ;
- effectuer les corrections nécessaires.

Mais, il est parfois plus intuitif d'en passer par l'étude des dépendances fonctionnelles directes :

- identifier les entités en présence et leur donner un identifiant (numéro arbitraire et auto-incrémenté) ;
- ajouter l'ensemble des attributs et leur dépendances fonctionnelles directes avec les identifiants (en commençant par les dépendances élémentaires) ;
- traduire le graphe de couverture minimale obtenu en un schéma entités-associations ;
- ajuster les cardinalités minimales ;
- à ce stade, la majorité des règles de normalisation devraient être vérifiées, il reste tout de même la normalisation des noms, la présence d'attributs en plusieurs exemplaires et d'associations redondantes ou en plusieurs exemplaires, à corriger.

Il faut garder également à l'esprit que le modèle doit être exhaustif (c'est-à-dire contenir toutes les informations nécessaires) et éviter toute redondance qui, on ne le dira jamais assez, constitue une perte d'espace, une démultiplication du travail de maintenance et un risque d'incohérence.

Il faut par ailleurs veiller à éliminer les synonymes (plusieurs signifiants pour un signifié, exemple : nom, patronyme, appellation) et les polysèmes (plusieurs signifiés pour un signifiant, exemples : qualité, statut).

Il va de soi que cette méthodologie ne doit pas être suivie pas-à-pas une bonne fois pour toute. Au contraire, il faut itérer plusieurs fois les étapes successives, pour espérer converger vers une modélisation pertinente de la situation.

3. Modèle logique de données (MLD)

Maintenant que le MCD est établi, on peut le traduire en différents systèmes logiques et notamment les bases de données relationnelles qui proposent une vision plus concrète pour modéliser la situation.

a) Systèmes logiques

Avant l'apparition des systèmes de gestion de base de données (SGBD ou DBMS pour Data Base Management System), les données étaient stockées dans des fichiers binaires et gérées par des programmes exécutables (développés en Basic, Cobol ou Dbase, par exemple).

Sont alors apparus les SGBD hiérarchiques dans lesquels les données sont organisées en arbre (IMS-DL1 d'IBM, par exemple), puis les SGBD réseaux dans lesquels les données sont organisées selon un graphe plus général (IDS2 de Bull, par exemple). Ces deux types de SGBD sont dit navigacionnels car on peut retrouver l'information à condition d'en connaître le chemin d'accès.

Aujourd'hui, ils sont largement remplacés par les SGBD relationnels (SGBDR) avec lesquels l'information peut être obtenue par une requête formulée dans un langage quasiment naturel (la langage SQL pour Structured Query Langage). Parmi les SGBDR les plus répandus nous trouvons Oracle, SQL Server et DB2. Nous nous contentons ici d'exposer le modèle logique de données relationnel (MLDR).

Plus récemment, sont apparus le modèle logique orienté objet et même des SGBD orientés objets. Pourtant, les SGBD relationnels restent extrêmement majoritaires, tandis que l'approche orienté objet est parfaitement adaptée au développement d'applications clientes dynamiques et liées aux données du système d'information.

b) *Modèle logique relationnel*

Tables, lignes et colonnes

Lorsque des données ont la même structure (comme par exemple, les renseignements relatifs aux clients), on peut les organiser en table dans laquelle les colonnes décrivent les champs en commun et les lignes contiennent les valeurs de ces champs pour chaque enregistrement (illustration 31).

numéro client	nom	prénom	adresse
1	Dupont	Michel	127, rue...
2	Durand	Jean	314, boulevard...
3	Dubois	Claire	51, avenue...
4	Dupuis	Marie	2, impasse...
...

Illustration 31: *Contenu de la table clients, avec en première ligne les intitulés des colonnes*

Clés primaires et clés étrangères

Les lignes d'une table doivent être uniques, cela signifie qu'une colonne (au moins) doit servir à les identifier. Il s'agit de la clé primaire de la table.

L'absence de valeur dans une clé primaire ne doit pas être autorisée. Autrement dit, la valeur vide (NULL) est interdite dans une colonne qui sert de clé primaire, ce qui n'est pas forcément le cas des autres colonnes, dont certaines peuvent ne pas être renseignées à toutes les lignes.

De plus, la valeur de la clé primaire d'une ligne ne doit pas changer au cours du temps.

Il se peut qu'une colonne Colonne1 d'une table ne doive contenir que des valeurs prises par la colonne Colonne2 d'une autre table (par exemple, le numéro du client sur une commande doit correspondre à un vrai numéro de client). 2 doit être sans doublons (bien souvent il s'agit d'une clé primaire). On dit alors que 1 est clé étrangère et qu'elle réfère 2.

Par convention, on souligne les clés primaires et on fait précéder les clés étrangères d'un dièse # dans la description des colonnes d'une table :

clients(numéro client, nom client, prénom, adresse client)

commandes(numéro commande, date de commande, #numéro client (non vide))

Remarques :

- une même table peut avoir plusieurs clés étrangères mais une seule clé primaire (éventuellement composées de plusieurs colonnes) ;
- une colonne clé étrangère peut aussi être primaire (dans la même table) ;
- une clé étrangère peut être composée (c'est le cas si la clé primaire référencée est composée) ;
- implicitement, chaque colonne qui compose une clé primaire ne peut pas recevoir la valeur vide (NULL interdit) ;
- par contre, si une colonne clé étrangère ne doit pas recevoir la valeur vide, alors il faut le préciser dans la description des colonnes.

Les SGBDR vérifient au coup par coup que chaque clé étrangère ne prend pas de valeurs en dehors de celles déjà prises par la ou les colonne(s) qu'elle référence. Ce mécanisme qui agit lors de l'insertion, de la suppression ou de la mise à jour de lignes dans les tables, garantit ce que l'on appelle l'intégrité référentielle des données.

Schéma relationnel

On peut représenter les tables d'une base de données relationnelle par un schéma relationnel dans lequel les tables sont appelées relations et les liens entre les clés étrangères et leur clé primaire est symbolisé par un connecteur (illustration 32).

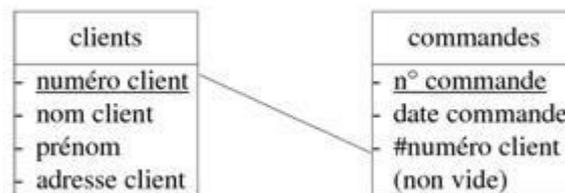


Illustration 32: Schéma relationnel simple entre deux tables

c) Traduction d'un MCD en un MLDR

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

Remarques :

On dit qu'une association binaire (entre deux entités ou réflexive) est de type :

- 1 : 1 (un à un) si aucune des deux cardinalités maximales n'est n ;
- 1 : n (un à plusieurs) si une des deux cardinalités maximales est n ;
- n : m (plusieurs à plusieurs) si les deux cardinalités maximales sont n.

Un schéma relationnel ne peut faire la différence entre 0,n et 1,n. Par contre, il peut la faire entre 0,1 et 1,1 (règles 2 et 4).

Règle 1 : toute entité devient une table dans laquelle les attributs deviennent les colonnes. L'identifiant de l'entité constitue alors la clé primaire de la table.

Par exemple, l'entité articles de l'illustration 13 devient la table :

articles(numéro article, désignation, prix unitaire de vente)

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.

Par exemple, l'association livrer de la l'illustration 14 est traduite par :

fournisseurs(n° fournisseur, nom contact, n° téléphone contact)

livraisons(n° livraison, date de livraison, nom livreur, #n° fournisseur (non vide))

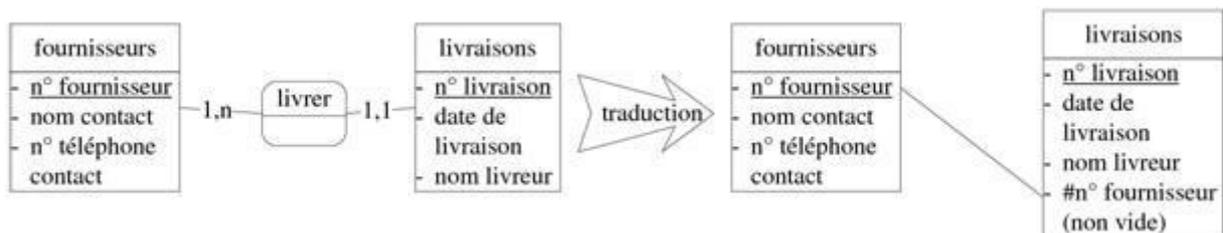


Illustration 33: Traduction d'une association de type 1 : n

Il ne devrait pas y avoir d'attribut dans une association de type 1 : n, mais s'il en reste, alors ils glissent vers la table côté 1.

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 référencent les deux clés primaires des deux tables en association). Les attributs de l'association deviennent des colonnes de cette nouvelle table.

Par exemple, l'association concerner (1) de l'illustration 34 est traduite par la table supplémentaire lignes de commande :

lignes de commande(#n° commande, #n° article, quantité commandée)

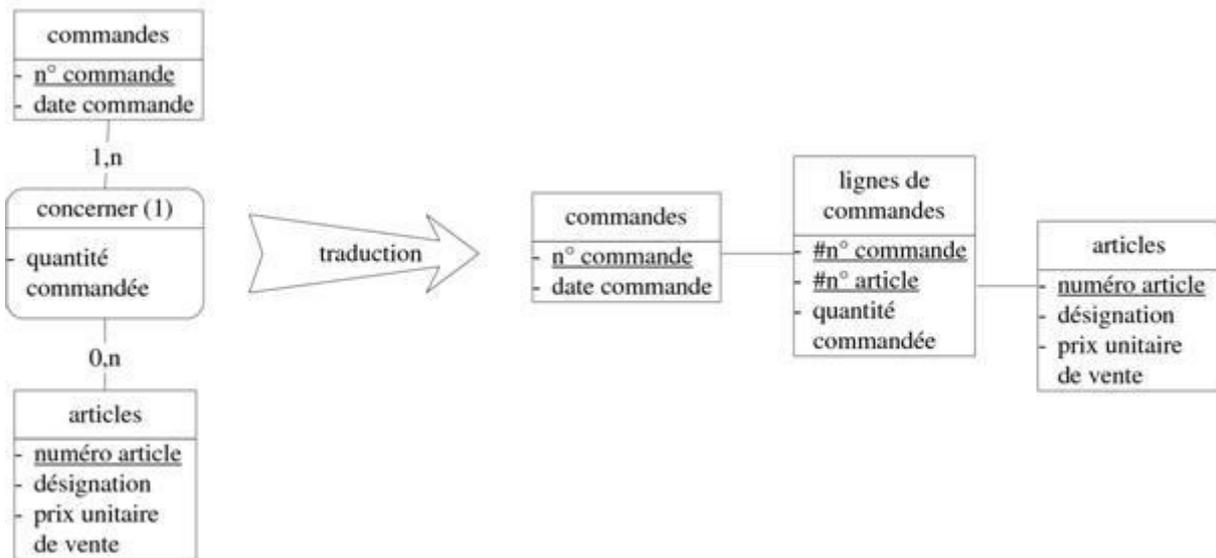


Illustration 34: Traduction d'une association de type $n : m$

Règle 4 : une association binaire de type 1 : 1 est traduite comme une association binaire de type 1 : n sauf que la clé étrangère se voit imposer une contrainte d'unicité en plus d'une éventuelle contrainte de non vacuité (cette contrainte d'unicité impose à la colonne correspondante de ne prendre que des valeurs distinctes).

Si les associations fantômes ont été éliminées, il devrait y avoir au moins un côté de cardinalité 0,1. C'est alors dans la table du côté opposé que doit aller la clé étrangère. Si les deux côtés sont de cardinalité 0,1 alors la clé étrangère peut être placée indifféremment dans l'une des deux tables.

Par exemple, l'association diriger de l'illustration 35 est traduite par :

services(n° service, nom service, #numéro employé (non vide, unique))

employés(numéro employé, nom)

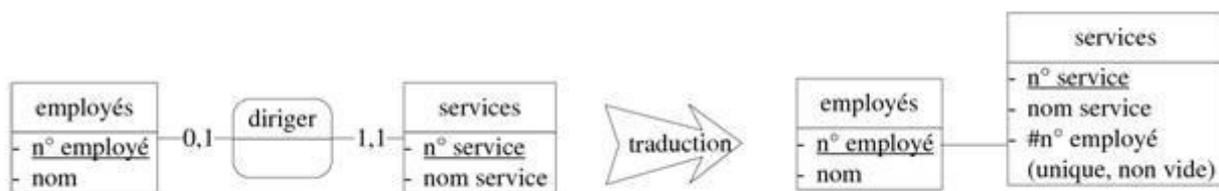


Illustration 35: Traduction d'une association de type 1 : 1

En réalité, la règle 4 proposée ici considère qu'une association binaire de type 1 : 1 correspond à une association binaire de type 1 : n particulière. Une alternative consiste à voir une association binaire de type 1 : 1 comme une association binaire de type n : m particulière. Il suffit pour cela d'ajouter une contrainte d'unicité sur chacune des clés étrangères de la table de jonction supplémentaire :

services(n° service, nom service)

directions(#n° service (unique), #numéro employé (unique))

employés(numéro employé, nom)

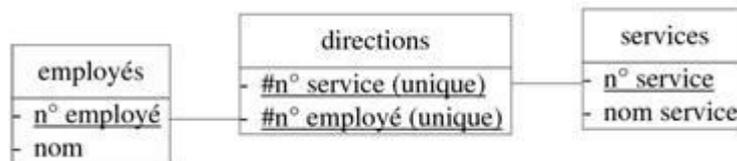


Illustration 36: Traduction alternative d'une association de type 1 : 1

Mais rien ne garantit, dans cette traduction alternative, qu'un service possède un dirigeant, alors que c'est obligatoire. La première traduction est donc préférable.

Remarque : d'autres techniques sont parfois proposées pour cette règle 4 (fusionner les tables, utiliser une clé primaire identique, utiliser deux clés étrangères réflexives) mais elles ne sont pas exploitables dans le cas général.

Règle 5 : une association non binaire est traduite par une table supplémentaire dont la clé primaire est composée d'autant de clés étrangères que d'entités en association. Les attributs de l'association deviennent des colonnes de cette nouvelle table.

Par exemple, l'association projeter de la figure 8 devient la table :

projections(#n° film, #n° salle, #n° créneau, tarif)

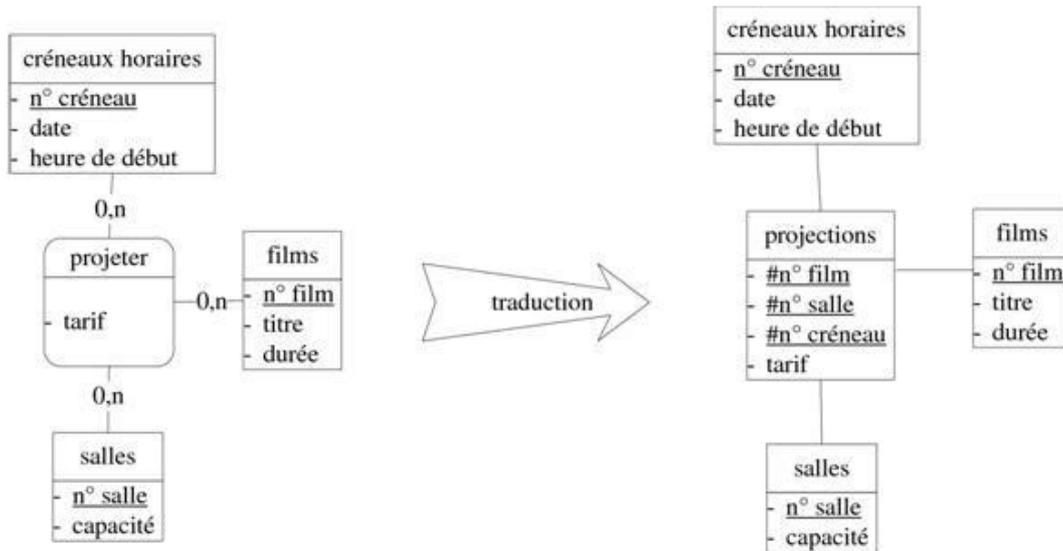


Illustration 37: Traduction d'une association ternaire

4. Modèle physique de données (MPD)

Un modèle physique de données est l'implémentation particulière du modèle logique de données par un logiciel.

a) Distinction entre MLD et MPD

La traduction d'un MLD conduit à un MPD qui précise notamment le stockage de chaque donnée à travers son type et sa taille (en octets ou en bits). Cette traduction est également l'occasion d'un certain nombre de libertés prises par rapport aux règles de normalisation afin d'optimiser les performances du système d'information.

La traduction d'un MLD relationnel en un modèle physique est la création (par des requêtes SQL de type CREATE TABLE et ADD CONSTRAINT) d'une base de données hébergée par un SGBD relationnel particulier. Il peut s'agir d'une base Oracle, d'une base SQL Server, d'une base Access ou d'une base DB2, par exemple. Le fait que tous les SGBDR reposent sur le même modèle logique (le schéma relationnel) permet à la fois la communication entre des bases hétérogènes et la conversion d'une base de données d'une SGBDR à l'autre.

b) Optimisations

L'optimisation des performances en temps de calcul se fait toujours au détriment de l'espace mémoire consommé. Dans le pire des cas, réduire les temps de réponse consiste à dé-normaliser volontairement le système d'information, avec tous les risques d'incohérence et les problèmes de gestion que cela comporte.

Pour les bases de données relationnelles, l'optimisation qui vise à accélérer les requêtes peut passer par :

- l'ajout d'index aux tables (au minimum sur les colonnes clés primaires et clés étrangères) ; ces index consomment de l'espace mémoire supplémentaire, mais la base de données reste normalisée ;
- l'ajout de colonnes calculées ou de certaines redondances pour éviter des jointures coûteuses (auquel cas la base est dé-normalisée) ; il faut alors veiller à ce que la cohérence entre les colonnes soit respectée, soit par l'utilisation de déclencheurs, soit dans les applications clientes du système d'information ;
- la suppression des contraintes d'unicité, de non vacuité ou encore de clé étrangère (auquel cas, l'intégrité des données doit être assurée par le code client du système d'information).

Par exemple, la table commandes de la figure 31 peut être supprimée et la date de commande est alors ajoutée à la table lignes de commandes.

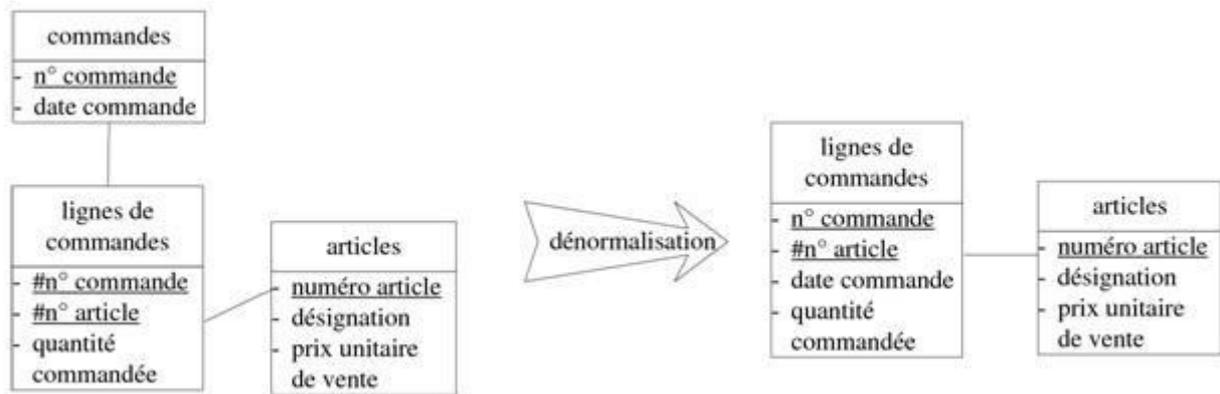


Illustration 38: Sacrifice de la troisième forme normale

On renonce donc à la troisième forme normale (figure 35) puisque la date de commande est répétée autant de fois qu'il y a de lignes dans la commande, mais on évite ainsi une jointure coûteuse en temps de calcul lors des requêtes SQL..

Le conseil le plus précieux, en matière d'optimisation, est de ne jamais optimiser *a priori*, mais toujours *a posteriori*, c'est-à-dire en réponse à une lenteur que le SGBDR n'est pas capable de résoudre tout seul. Il faut alors mesurer le gain de toute optimisation manuelle en effectuant des tests (chronométrages avant/après) sur un volume de données significatif et de préférence en exploitation.

5. Rétro-conception

Dans la majorité des cas, le travail du concepteur de bases de données consiste non pas à créer une base de données *ex nihilo*, mais plutôt à corriger ou étendre une base existante. Dans ce cas, la matière de travail initiale est un modèle physique et la méthode de rétro-conception ou reverse engineering consiste à traduire ce MPD en un modèle conceptuel, modifier le MCD obtenu puis modifier le modèle physique en conséquence.

a) Traduction inverse

Dans le cadre des bases de données relationnelles, il faut convertir le modèle physique en un schéma relationnel normalisé (en détricotant les optimisations éventuelles et en renommant les colonnes des tables pour assurer l'unicité et le caractère explicite (non codé) des noms), puis appliquer les règles de traduction de la section « Traduction d'un MCD en un MLDR » dans le sens inverse.

Étape 1 : chaque table dont la clé primaire ne contient pas de clé étrangère devient une entité dont l'identifiant est la clé primaire de la table et dont les attributs sont les colonnes de la table qui ne sont pas clé étrangère.

Étape 3 : chaque table dont la clé primaire est composée exclusivement de clés étrangères qui réfèrent plusieurs clés primaires, devient une association autour de laquelle toutes les cardinalités maximales valent n, c'est-à-dire soit une association binaire de type n : m soit une association ternaire ou plus (les autres colonnes non clés étrangères de la table deviennent des attributs de l'association).

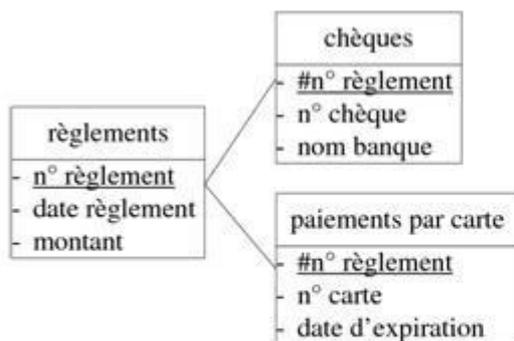
Étape 5 : les colonnes clés étrangères restantes deviennent des associations binaires de type 1 : n s'il n'y a pas de contrainte d'unicité ou de type 1 : 1 s'il y a une contrainte d'unicité (il faut trouver un nom à cette association).

Étape 6 : la cardinalité minimale vaut 1 pour les clés étrangères qui font partie d'une clé primaire ou qui possèdent une contrainte (non vide), sinon elle vaut 0.

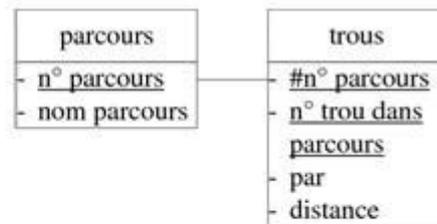
b) Cas particuliers

Malheureusement, ces quatre étapes ne suffisent pas pour traduire tous les schémas relationnels possibles. Notamment, les tables de la figure 36 nécessitent l'insertion d'étapes supplémentaires.

1. clé sur une colonne, mais à la fois primaire et étrangère
2. clé primaire composée partiellement de clé étrangère



(a) clé sur une colonne, mais à la fois primaire et étrangère



(b) clé primaire composée partiellement de clé étrangère

Illustration 39: Tables particulières en rétro-conception

Étape 2 : chaque table dont la clé primaire est composée exclusivement de clés étrangères qui référencent une seule clé primaire, devient une sous-entité ou une sous-association (les autres colonnes non clés étrangères de la table deviennent des attributs de cette sous-entité).

Étape 4 : chaque table dont la clé primaire est composée partiellement de clés étrangères provient soit d'une optimisation qu'il faut défaire (comme sur la figure 35) soit d'un identifiant relatif d'une entité comme dans la section « Identifiant relatif ou lien identifiant » (auquel cas les autres colonnes non clés étrangères de la table deviennent des attributs de cette entité).

6. Compléments

Aucune situation complète, ou presque, ne peut être parfaitement modélisée si le concepteur se contente des fonctionnalités déjà abordées. Il est nécessaire d'introduire de nouvelles notations sur le schéma entités-associations. Les trois extensions majeures présentées dans cette section font partie de la version 2 de Merise et permettent de traiter davantage de situations réelles et souvent de manière plus simple.

Dans cette section, nous reprenons la démarche qui consiste à étudier les dépendances fonctionnelles directes sur le graphe de couverture minimale, puis à traduire ce graphe en schéma entités-associations, pour obtenir finalement un schéma relationnel. Les notions abordées ici ne permettent plus au schéma relationnel d'être écrit textuellement sans ambiguïté. Afin de lever toute ambiguïté pour savoir quelle clé primaire est référencée par telle clé étrangère, il est impératif de représenter le schéma relationnel de manière graphique, ce que nous nous contentons de faire.

a) Agrégation

Une association n'est pas forcément établie exclusivement entre des entités.

Association de type 1:n

Considérons l'exemple de la figure 40 issu du monde des courses hippiques. La dépendance fonctionnelle n° cheval + n° course ? n° jockey est la première dépendance fonctionnelle non élémentaire vers un identifiant que nous rencontrons. Ce type de dépendance fonctionnelle nous incite à créer une association binaire de type 1 : n entre l'entité jockeys et l'association binaire de type n : m qu'il y a entre les entités chevaux et courses. D'un point de vue sémantique, la logique est respectée puisque un jockey ne monte pas un cheval, mais un cheval-qui-participe-à-une-course.

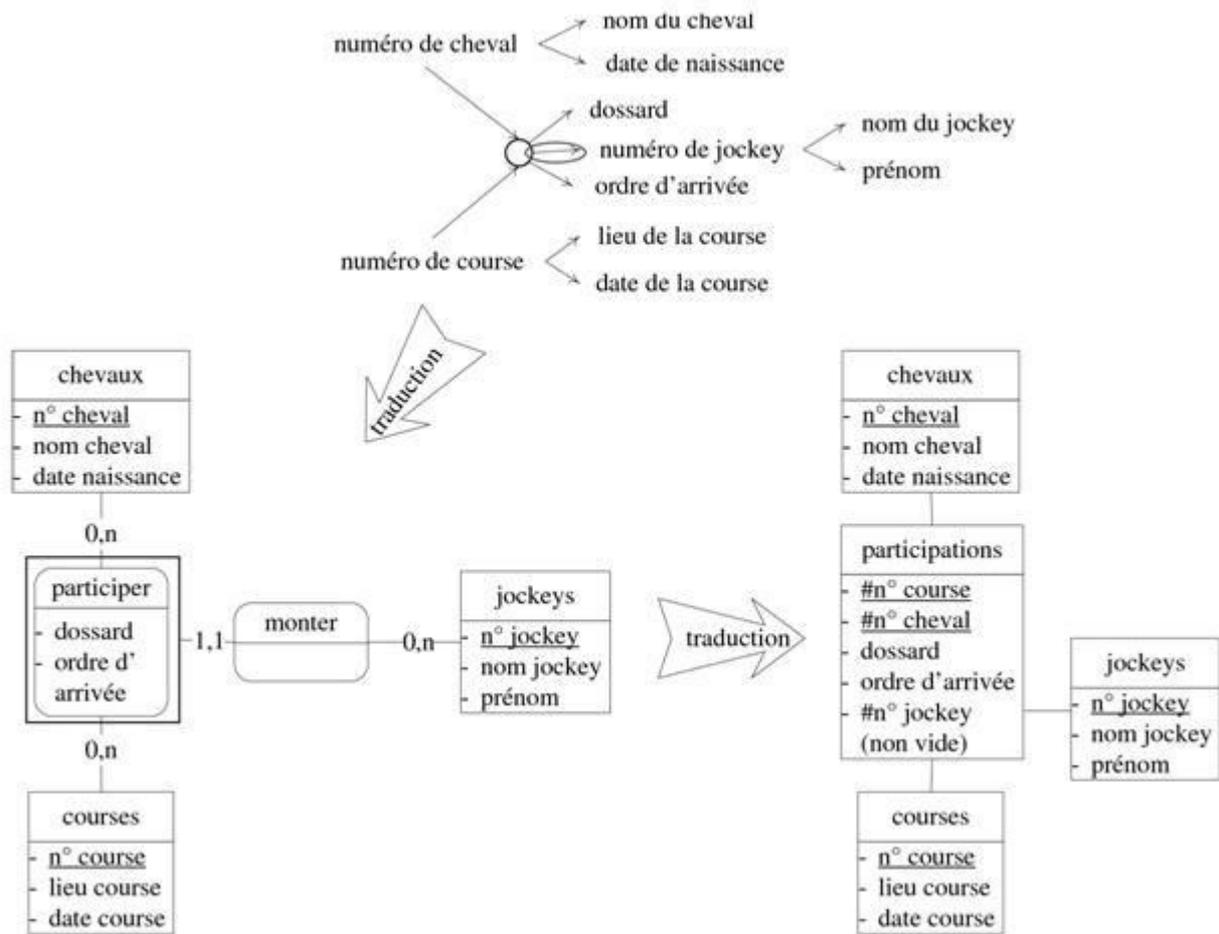


Illustration 40: Association binaire de type 1 : n (*monter*), liée à une association binaire de type n : m (*participer*)

Pour tenir compte de ce nouveau cas de dépendance fonctionnelle, il convient d'ajouter une sixième étape à la technique de traduction d'un graphe de couverture minimal en un schéma entités-associations, telle qu'elle est commencée section « Traduction vers un schéma entités-associations » :

Étape 6 : lorsqu'un identifiant dépend de plusieurs autres identifiants, son entité est en association de type 1 : n avec l'association qui lie les autres identifiants.

Certains auteurs considèrent que l'agrégation des entités chevaux, courses et de l'association participer constitue une nouvelle entité participations qui englobe ces trois éléments graphiques. Dans ce cas, l'association monter fait le lien entre les deux entités (participations et jockeys). Le résultat final sur le schéma relationnel est le même. Malheureusement, cette notation n'est pas très pratique car le schéma entités-associations devient vite illisible lorsqu'une entité participe à plusieurs agrégations.

Nous préférons donc autoriser, dans ce document, qu'une association puisse être liée à une association binaire de type n : m ou à une association ternaire (ou plus). Cependant pour ne pas confondre les liens entre associations et entités avec les liens entre associations, nous encadrons soigneusement les associations qui interviennent dans une agrégation, comme sur la figure 40 en bas à gauche.

Une association ne peut pas être liée à une association binaire de type 1:n ou 1:1. Dans ce cas, l'association doit être directement liée à l'entité qui se trouve du côté où la cardinalité maximale est 1.

Sur le schéma relationnel final (figure 40 en bas à droite), la table de jonction participations reçoit une clé étrangère supplémentaire, mais qui contrairement aux autres, ne participe pas à la clé primaire.

Association de type n:m

À présent, ajoutons les parieurs à l'exemple de la figure 40. Étant donné que nous avons la dépendance fonctionnelle n° cheval + n° course + n° parieur ? montant de la mise (figure 41 en haut), nous pourrions avoir une association ternaire entre les entités chevaux, courses et parieurs. Mais dans ce cas, un parieur peut miser sur un cheval dans une course, alors que ce cheval ne participe pas à cette course.

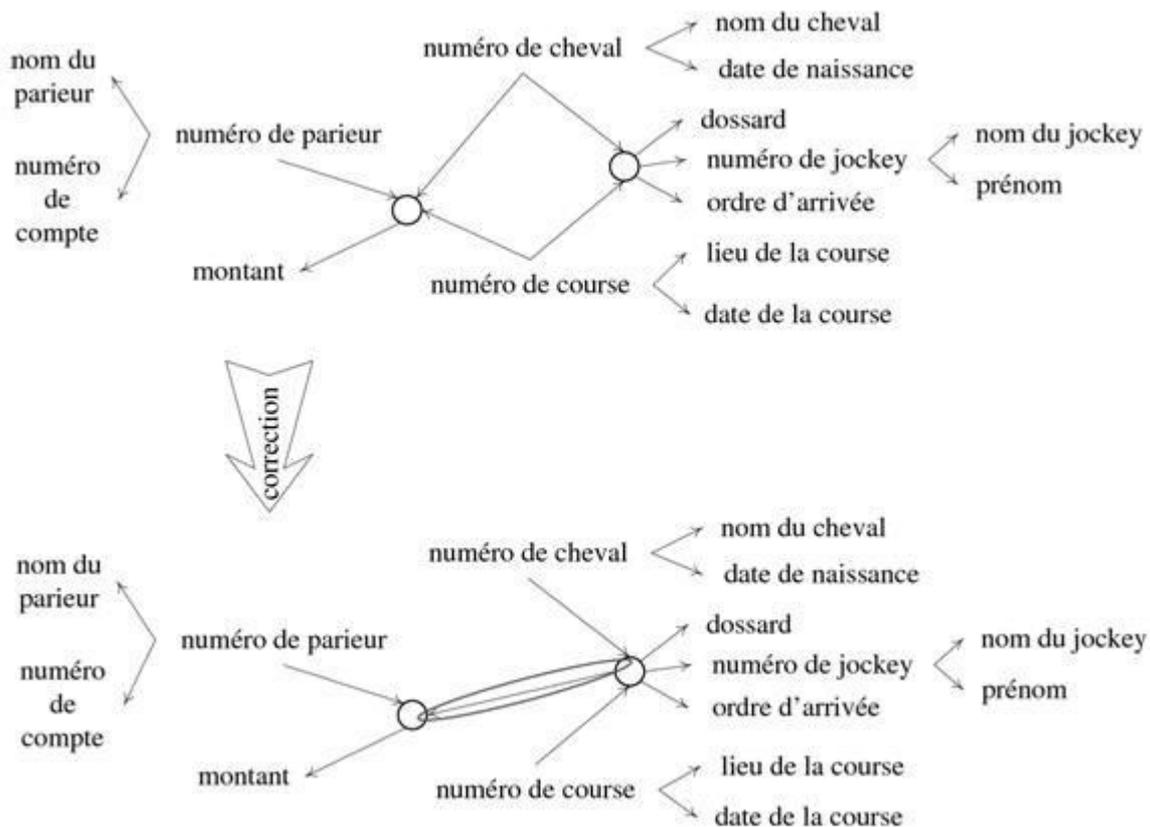


Illustration 41: Association ternaire remplacée par deux associations binaires

Pour pallier à cette lacune, on pourrait faire appel à des déclencheurs programmés dans la base de données finale. Les déclencheurs sont des procédures SQL qui, dans notre exemple, permettraient à chaque insertion ou mise à jour de lignes dans la table des paris, d'assurer qu'un pari ne puisse pas concerner un cheval dans une course à laquelle il ne participe pas. Cependant, il existe une solution plus simple qui repose uniquement sur l'intégrité référentielle.

En réalité (figure 41 en bas), la vraie dépendance fonctionnelle directe est n° cheval + n° course + n° parieur ? montant, ce qui garantit qu'un parieur ne peut miser que sur un cheval-qui-participe-à-une-course.

Le fait qu'une association ternaire (ou plus) disparaissent au profit d'une ou plusieurs agrégations est fréquent lorsque l'on modélise une situation complète. À tel point qu'on peut partir du principe qu'un schéma entités-associations sans agrégation est généralement faux.

Dans notre exemple, la traduction de la nouvelle dépendance fonctionnelle en une association de type n:m (figure 42 en haut) se fait en appliquant, comme d'habitude, l'étape 4 de la section « Traduction vers un schéma entités-associations ».

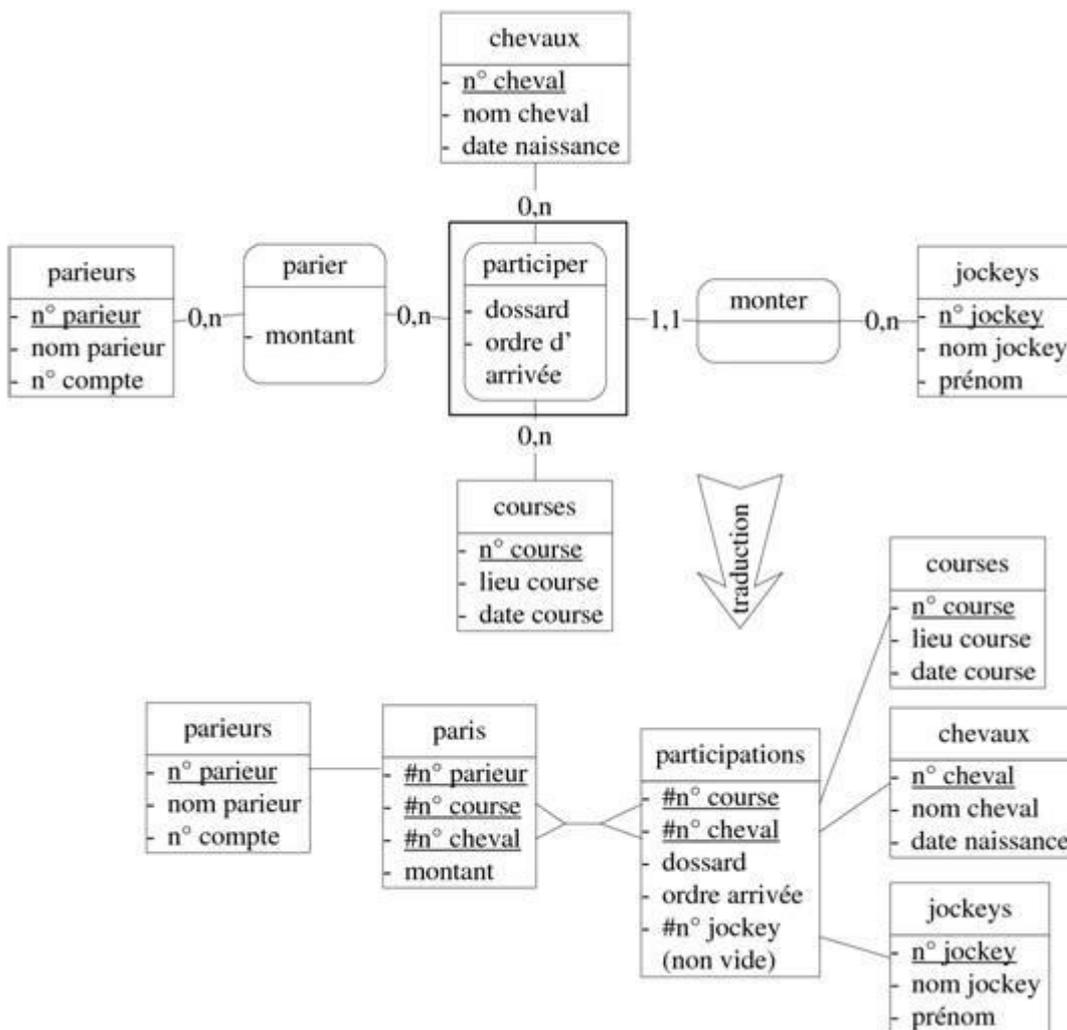


Illustration 42: Association binaire de type n:m (parier), liée à une autre association binaire de type n:m

Sur le schéma relationnel obtenu (figure 42 en bas), la traduction de l'association binaire de type n:m liée à une autre association binaire de type n:m fait apparaître dans la table paris une clé étrangère composite qui référence la clé primaire composite de la table participations.

Rappelons qu'il est déconseillé d'utiliser des identifiants composites. Mais la clé primaire composite de la table participations est légitime puisqu'elle est issue d'une association binaire de type n:m. En conséquence de quoi la clé étrangère composite de la table paris est également légitime puisqu'elle est aussi issue d'une association binaire de type n:m.

On peut ainsi imaginer avoir sur un schéma relationnel des clés primaires ou étrangères composées d'un nombre arbitraire de colonnes, sans pour autant qu'il n'y ait un seul identifiant composite sur le schéma entités-associations correspondant.

Tables de codification ou tables de référence

Certains attributs ne peuvent prendre qu'un jeu volontairement limité de valeurs. C'est le cas sur la figure 43 à gauche, pour les attributs enseignant et matière. Cela évite sur cet exemple qu'une même matière ne soit décrite de deux manières différentes et qu'un même nom d'enseignant ne soit orthographié deux fois.

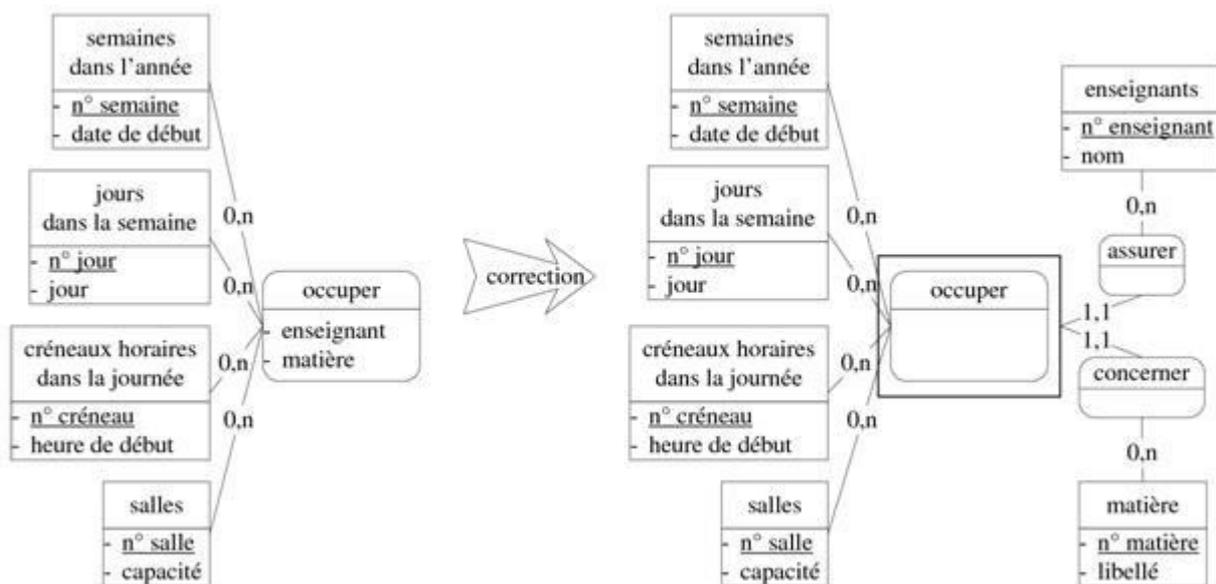


Illustration 43: Agrégation et entités de codification

Il est recommandé de regrouper ces valeurs au sein d'une entité dite de codification (qui donnera ensuite une table de codification). Si l'attribut concerné appartient à une entité, alors cette entité est en association binaire de type 1:n avec cette entité de codification. Par contre, si l'attribut fait partie d'une association, il faut recourir à l'agrégation afin de mettre en association l'entité de codification avec l'association de cet attribut (figure 43 à droite).

Ainsi, l'agrégation évite notamment aux entités de codification de transformer une association binaire en une association ternaire (ou plus).

b) Identifiant relatif ou lien identifiant

Même en utilisant des agrégations, il reste des situations où tout le potentiel de l'intégrité référentielle n'est pas exploité.

Résolution d'un problème sur le schéma relationnel

Prenons par exemple le schéma relationnel en haut de la figure 44, tiré d'une base de données pour un centre de golf. Dans la table trous, la clé primaire n° trou est en incrément automatique, tandis que la colonne n° trou dans parcours est un nombre (généralement compris entre 1 et 18) qui correspond à la numérotation des trous dans le parcours.

Le problème de ce schéma relationnel est qu'en l'état, il peut y avoir un score dans la table scores pour un trou qui n'appartient pas au parcours sur lequel la partie se joue (le lecteur est invité à bien observer la figure pour s'en apercevoir).

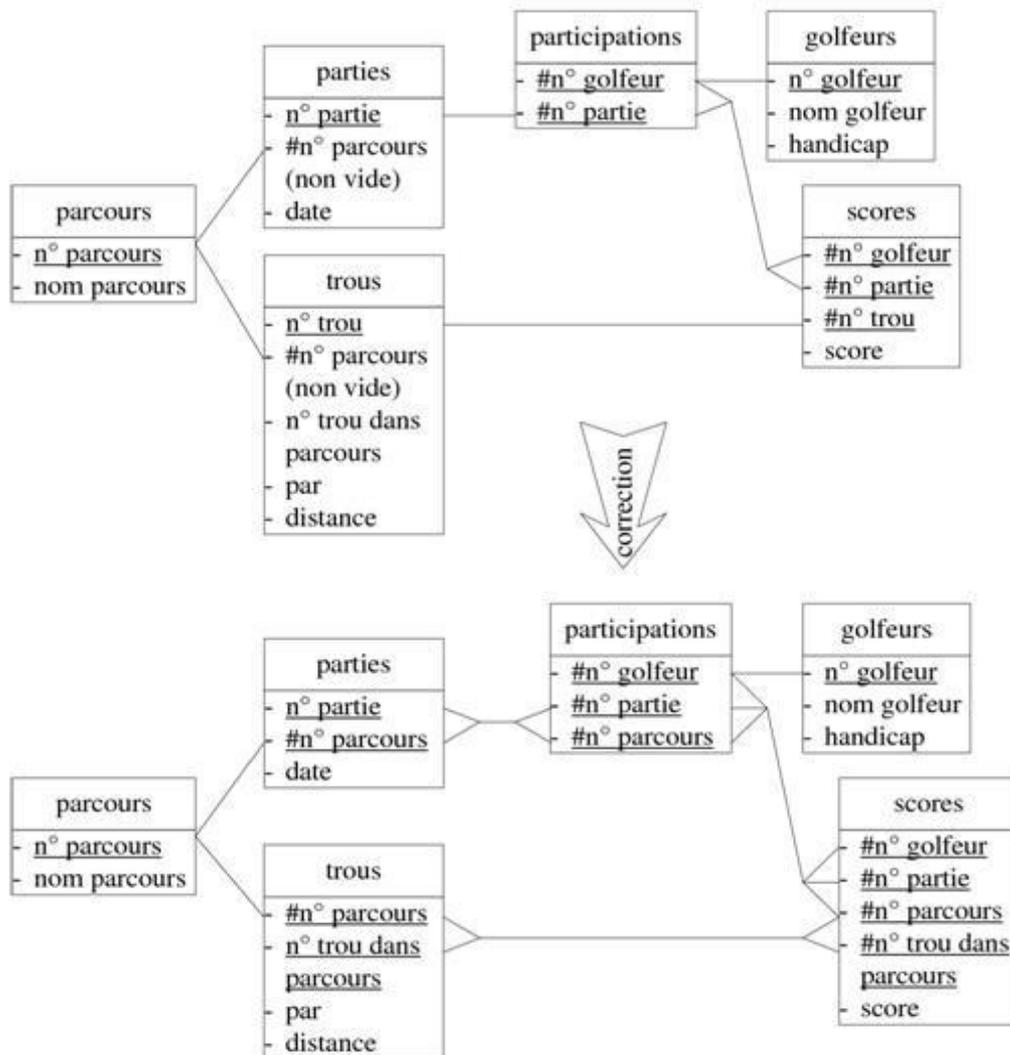


Illustration 44: Utilisation de clés primaires partiellement étrangères

Pour régler ce problème, on peut à nouveau se reposer sur l'emploi de déclencheurs. Mais là-encore, il existe une solution ne faisant appel qu'à l'intégrité référentielle.

Cette solution consiste à faire entrer le numéro de parcours dans la numérotation des trous (remplaçant ainsi le n° trou) ainsi que dans la numérotation des parties (en conservant cette fois-ci le n° partie en incrément automatique). Les tables trous et parties possèdent alors une clé primaire composite et partiellement étrangère (figure 44 en bas).

Les clés étrangères des tables participations et scores qui référencent ces nouvelles clés primaires sont alors complétées par une nouvelle colonne (le numéro de parcours). Dans la table des scores, comme cette colonne n° parcours n'est introduite qu'une fois, il n'est plus possible pour un joueur d'avoir un score sur un trou qui n'appartient pas au parcours sur lequel se joue la partie.

Modèle conceptuel correspondant

En rétro-conception, pour tenir compte du fait que le numéro de parcours fera partie de la clé primaire de la table trous sur le schéma entités-associations, il suffit de mettre entre parenthèses la cardinalité 1,1 de l'association entre les entités trous et parcours (figure 42). L'identifiant de l'entité côté 1,1 devient alors relatif à celui de l'autre entité en association.

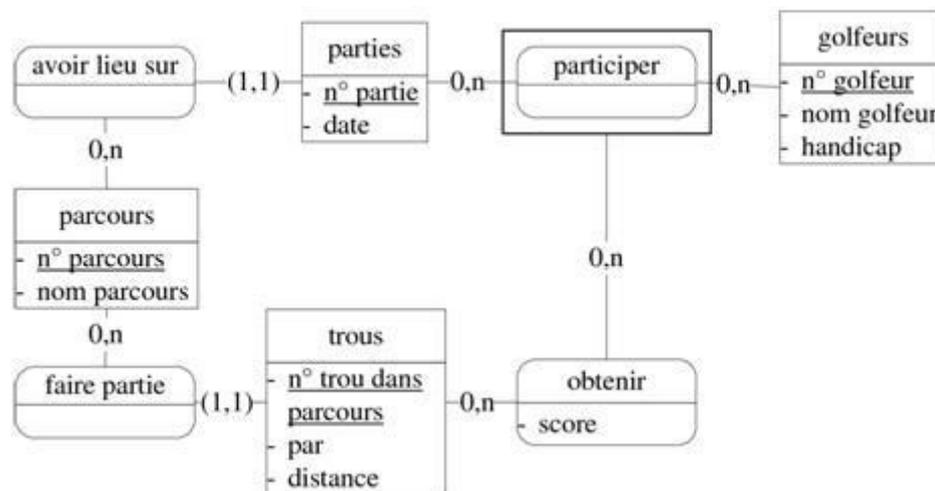


Illustration 45: Représentation des identifiants relatifs

De même, sur le graphe de couverture minimal, nous introduisons une nouvelle notation (flèche en pointillés) pour représenter le caractère relatif des identifiants (figure 46).

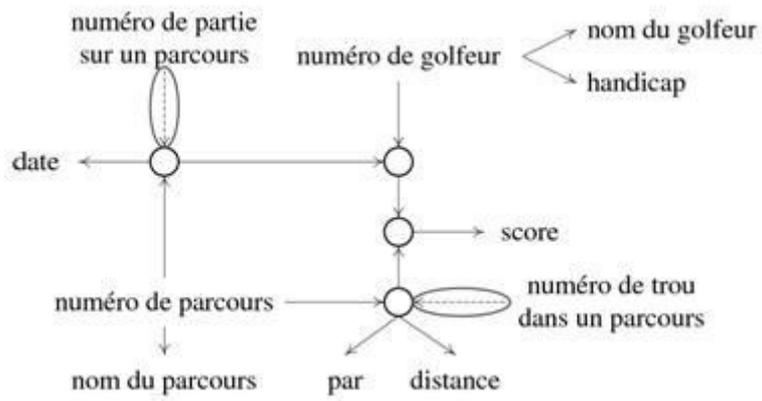


Illustration 46: *Représentation des identifiants relatifs sur le graphe de couverture minimale*

Si les flèches étaient pleines, les numéros de trou dans un parcours et de partie sur un parcours figureraient dans des entités séparées et réduites à leur identifiant (à éviter).

Discussion autour de la numérotation des exemplaires

Dans un magasin de location de vidéos, le gérant peut vouloir numérotter séparément les exemplaires de chaque vidéo (figure 47 colonne de gauche), alors que le concepteur de la base de données aurait tendance à vouloir numérotter globalement l'ensemble des exemplaires (colonne de droite).

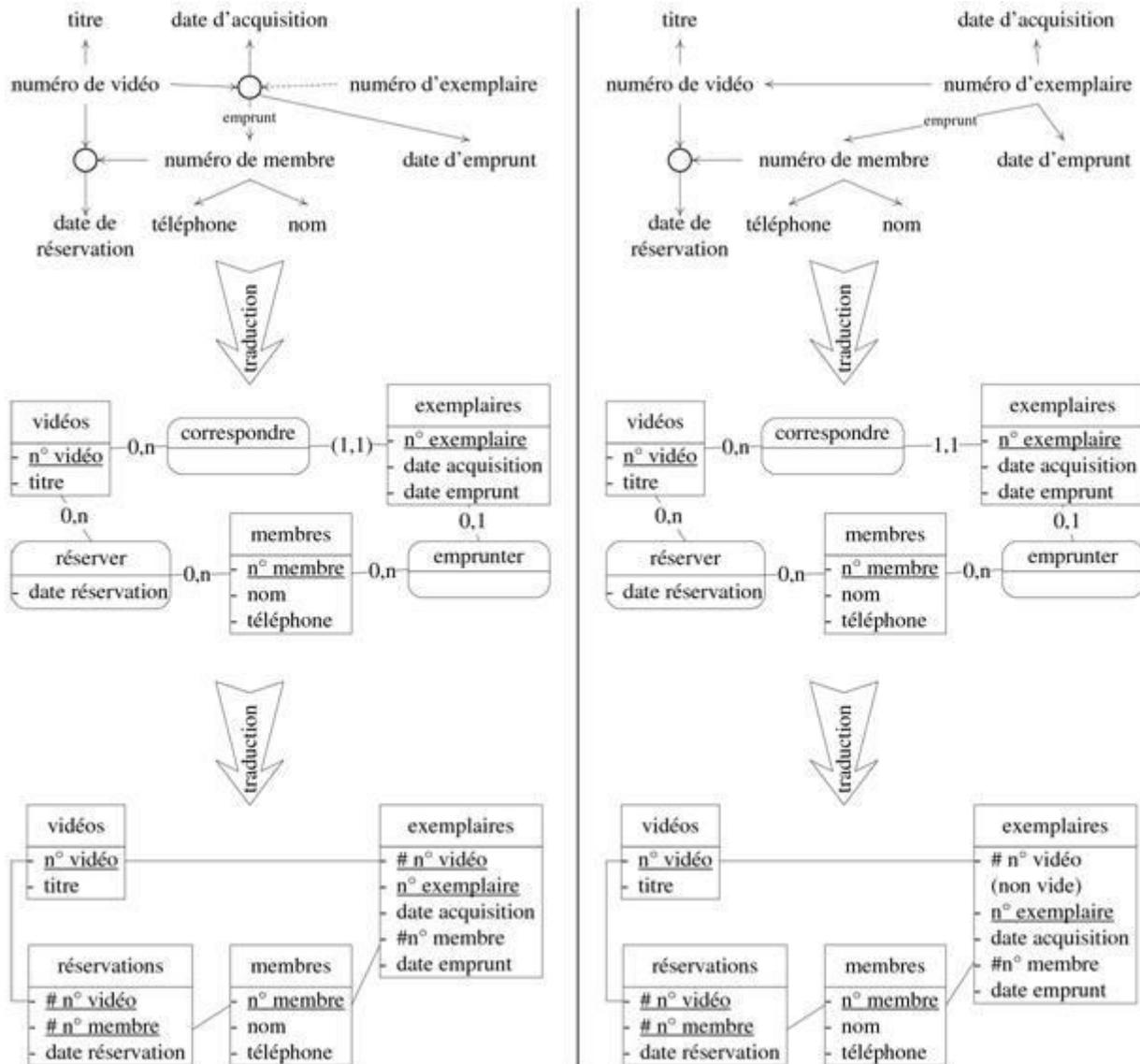


Illustration 47: Numérotations alternatives des exemplaires

La seule différence entre les deux solutions est l'entrée ou non de la colonne numéro vidéo dans la clé primaire de la table exemplaire. L'inconvénient majeur de la solution avec identifiant relatif (colonne de gauche), est le traitement de l'incrément automatique du numéro d'exemplaire, car il faut un compteur pour chaque vidéo et non pas un compteur pour l'ensemble des vidéos, contrairement à la solution de la colonne de droite. Le concepteur devrait donc retenir sa solution pour la base de données et proposer une colonne supplémentaire avec la numérotation du gérant, afin de lui faire plaisir.

c) Héritage

Enfin, il est parfois utile de factoriser les attributs communs à plusieurs entités au sein d'une entité mère.

Sous-entité

Considérons l'exemple suivant : les factures d'une entreprise font l'objet d'un règlement par chèque ou par carte.

Cette entreprise souhaite connaître pour chaque règlement la date, le montant et :

- le numéro et le nom de la banque des chèques ;
- ou le numéro et la date d'expiration des paiements par carte.

On a donc une entité générique règlements et deux entités spécialisées chèques et paiements par carte. Ces deux sous-entités de l'entité règlements ont des attributs propres mais pas d'identifiant propre. Au niveau logique objet, on retrouve la notion d'héritage.

Conformément aux notations objets, sur le schéma entités-associations, on représente le lien qui unit une sous-entité à son entité générique par une flèche creuse (figure 48 au centre). Ce lien remplace une association être de type 1 : 1 (un chèque « est un » règlement et un paiement par carte « est un » règlement).

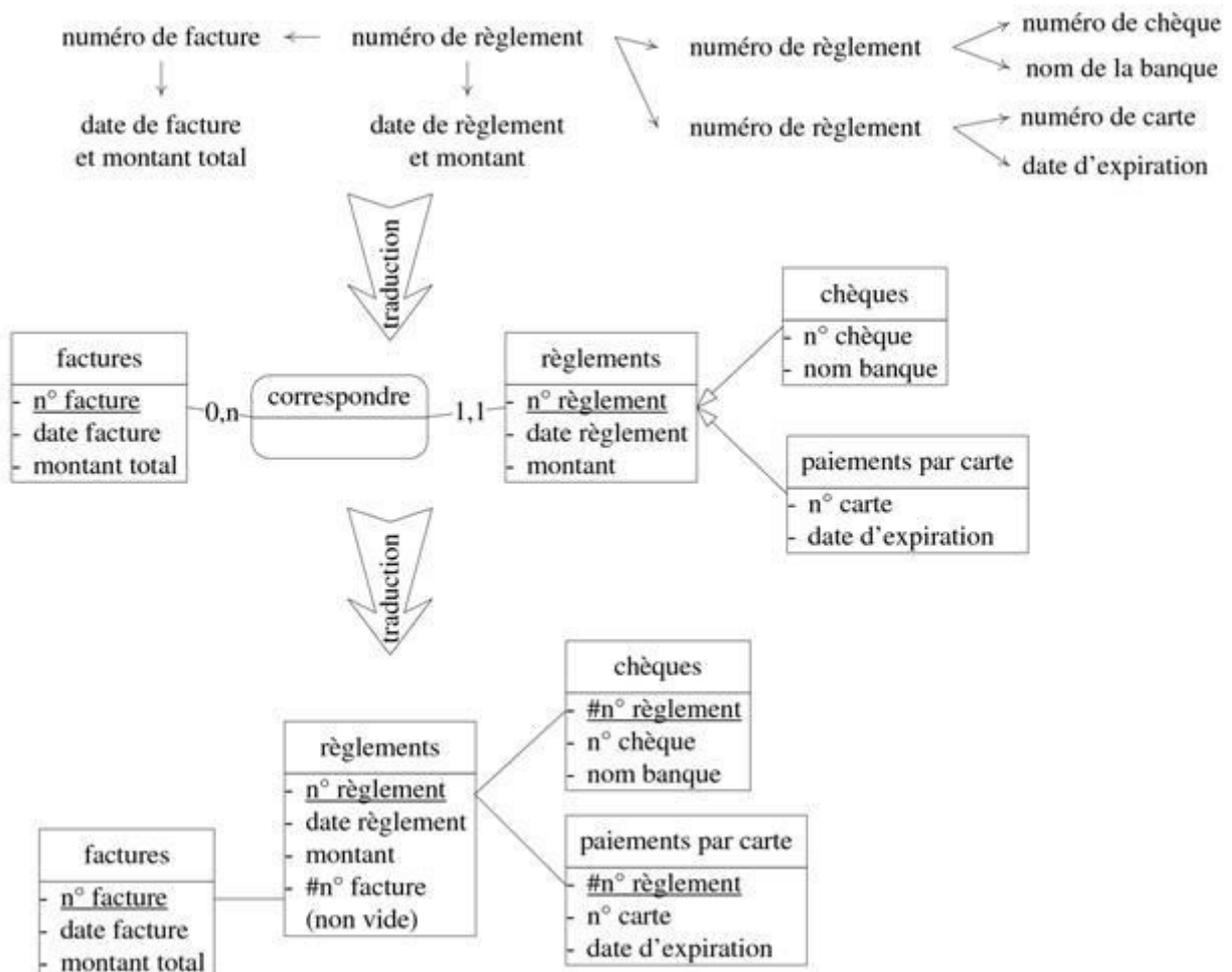


Illustration 48: Représentation des sous-entités

Toutefois, il ne faut pas voir d'héritage à chaque fois que l'on peut dire « est un », car il faut en plus que l'entité mère ne possède que les attributs communs de ses entités filles. Par exemple, un cercle « est mathématiquement un » ovale. Mais l'entité cercles (avec les attributs centre et rayon) n'est pas une sous-entité de l'entité ovales car celle-ci possède davantage d'attributs (centre, rayon principal, rayon secondaire et rotation).

La traduction des sous-entités au niveau logique relationnel fait intervenir une clé primaire identique à celle de l'entité mère, mais dans les sous-entités la clé primaire est aussi étrangère (figure 45 en bas).

Sur le graphe de couverture minimale (figure 47 en haut), l'identifiant dont dépendent les attributs communs est volontairement dupliqué autant de fois que nécessaire pour les attributs spécialisés. Nous pouvons alors remarquer que les attributs qui dépendent d'un même identifiant peuvent être regroupés avec des « et » logiques tandis que dès qu'il est nécessaire de faire appel à un « ou » logique, c'est le signe d'une spécialisation.

Sur la figure 47, il est tentant de traduire directement le graphe de couverture minimale en le schéma relationnel, car il en est beaucoup plus proche que le schéma entités-associations. C'est une technique licite, à condition de traduire correctement les associations de type 1:1 (étape 4).

Utilisation de l'héritage pour séparer les informations complémentaires

L'héritage peut être utilisé même lorsqu'il n'y a qu'une entité spécialisée. C'est utile pour stocker dans une table séparée des informations complémentaires.

Considérons la table clients dans laquelle nous stockons déjà le numéro, le nom et le code postal. Nous souhaitons désormais stocker également le numéro de téléphone, l'adresse courrier et l'adresse électronique. La première idée consiste à ajouter trois colonnes supplémentaires dans la table clients. Mais pour les clients qui ont déjà été saisis dans la table, ces trois colonnes seront vides.

Pour gagner de la place, ces trois colonnes peuvent constituer une nouvelle table annuaire clients dont la clé primaire référence celle de la table clients (figure 49). Formellement, annuaire clients est issu d'une sous-entité de l'entité clients.

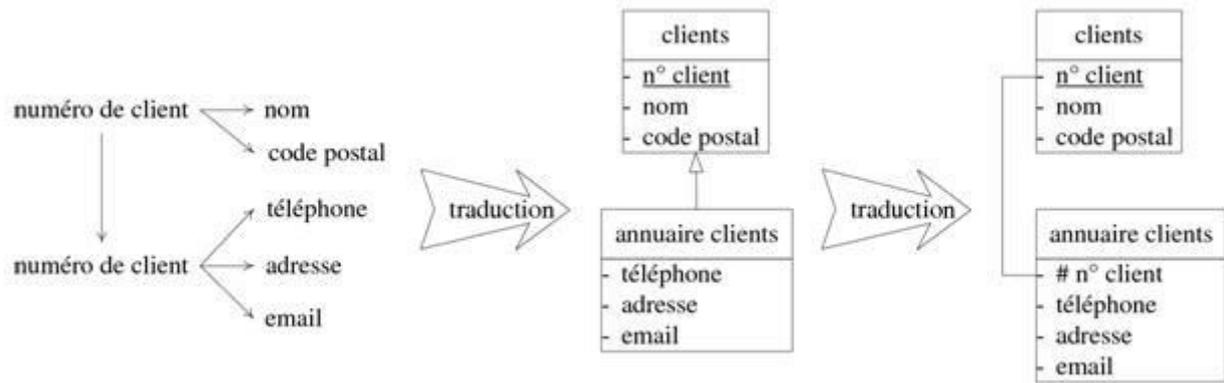


Illustration 49: Séparation des informations complémentaires par héritage

La configuration d'héritage sur le schéma relationnel (figure 49 à droite) constituera une occasion d'écrire des requêtes SQL avec des jointures externes (c'est-à-dire facultatives).

Spécialisation des associations

La notion d'héritage est valable également pour les associations. Nous pouvons donc faire appel à des sous-associations avec des attributs spécifiques et des associations génériques qui contiennent les attributs communs. Mais sans aller jusqu'à l'introduction de sous-associations, dès qu'un schéma entités-associations fait appel à des sous-entités, il est fréquent que les associations concernées par ces sous-entités soient elles-mêmes spécialisées.

Considérons une entreprise artisanale qui vend non seulement des articles produits en série à prix unitaire fixe, mais aussi des articles fait sur mesure et dont le prix unitaire est calculé à partir de la durée de confection et d'un taux horaire. Dans ce cas, non seulement l'entité articles est spécialisée en articles en série et articles sur mesure, mais en plus, l'association concerner entre les entités commandes et article est spécialisée selon qu'il s'agit d'un article en série ou sur mesure (figure 50 au centre).

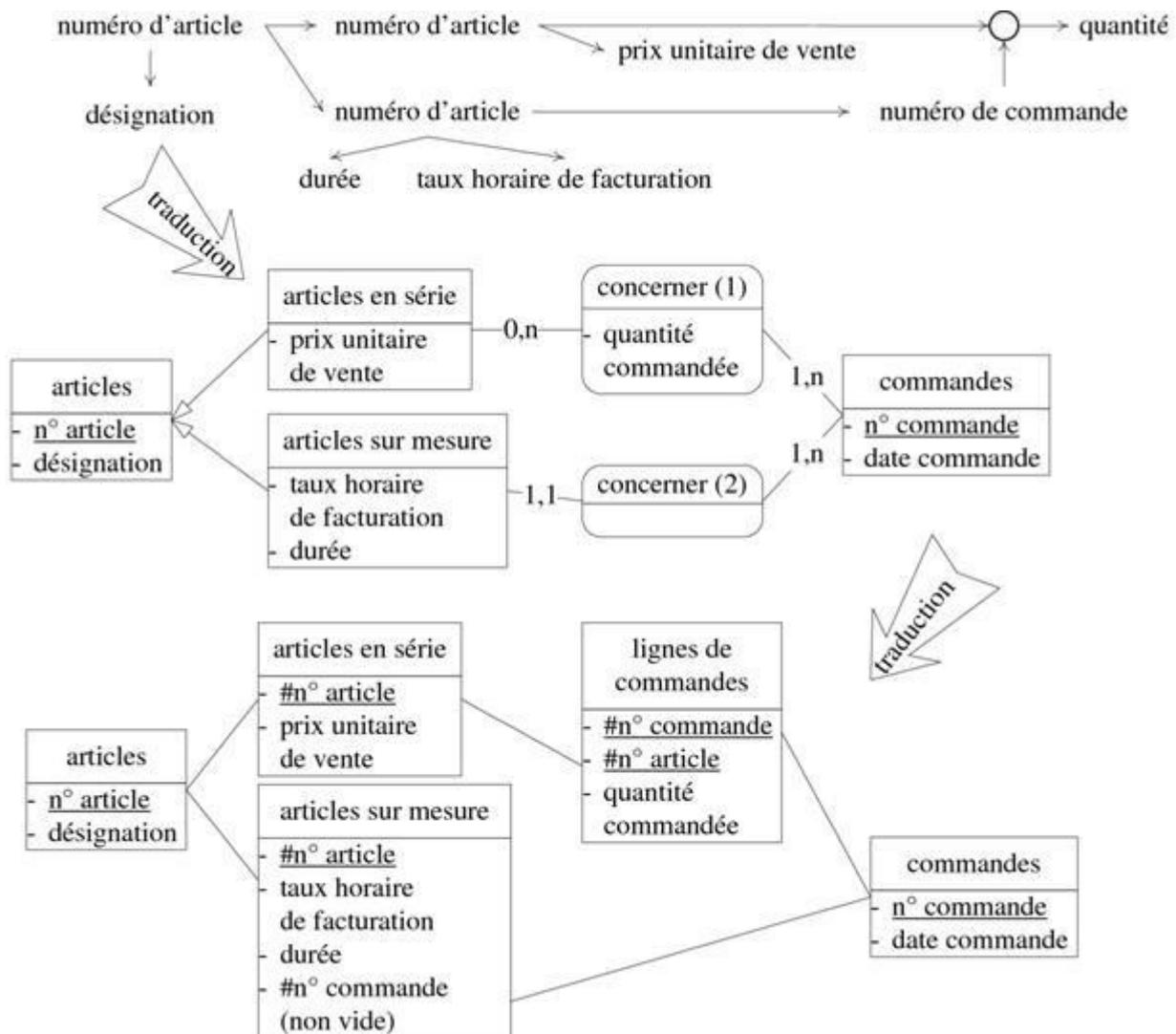


Illustration 50: Spécialisation des associations en présence de sous-entités

Le fait d'avoir volontairement dédoubler l'identifiant commun des entités en héritage, permet d'utiliser chaque identifiant dupliqué dans l'association qui le concerne.

Sur le schéma relationnel (figure 50 en bas), les associations spécialisées sont traduites de manière classique. À charge ensuite pour le développeur du formulaire de facturation, d'effectuer la réunion des articles commandés.

7. Conclusion

Avec la pratique, vient un moment où le concepteur peut se passer du modèle entités-associations et produire directement des schémas relationnels corrects. Pourtant, continuer de travailler à un niveau conceptuel plutôt qu'à un niveau logique reste une tactique payante pour lui, dans la mesure où les données pourtant stockées sous une forme relationnelle, doivent de nos jours être accédées par des applications orientées objet. Le modèle conceptuel permet de faire le lien entre d'une part la représentation objet des données et d'autre le stockage relationnel des mêmes données.

Par exemple, on peut très bien imaginer qu'un schéma entités-associations soit d'un côté traduit en un schéma relationnel puis implémenté dans une base de données Oracle ; tandis qu'en parallèle, il est traduit en un diagramme de classe (modèle logique objet), lui-même implémenté dans un ensemble de classes Java. Ces classes Java permettent ensuite aux développeurs de construire des applications clientes orientées objet et qui attaquent de manière transparente les données de la base Oracle. Il s'agit d'une solution de passage entre la modélisation orientée objet (pertinente pour développer des interfaces graphiques) et la modélisation relationnelle (pertinente pour gérer les données).

Par ailleurs, la méthodologie Merise est certes typiquement française, mais en Grande-Bretagne, la méthodologie standard s'appelle SSADM (Structured Systems Analysis and Design Method) et repose sur les mêmes principes. Les nord-américains quant à eux utilisent ce qu'on appelle des diagrammes de flux, dont les principes sont repris par la version 2 de Merise.

Aujourd'hui, ce sont les modélisations objets et leur unification UML (Unified Modeling Language, autrement dit langage unifié de modélisation) qui se placent à la pointe de l'état de l'art. Malheureusement, UML n'est qu'un ensemble de notations (d'ailleurs moins intuitives que celles des schémas entités-associations). La connaissance de ce langage ne permet donc pas au concepteur de faire l'économie d'une méthodologie de conception. Voilà pourquoi il n'est pas anachronique de ré-éditer en 2005 un document sur des méthodes qui auront bientôt 30 ans ;-)

Annexe

8. Index des illustrations

Index des illustrations

Illustration 1:Entités.....	5
Illustration 2:Associations.....	6
Illustration 3:Attributs.....	6
Illustration 4:Identifiants.....	7
Illustration 5:Cardinalités.....	7
Illustration 6:Associations plurielles.....	8
Illustration 7:Association réflexive.....	9
Illustration 8:Entité remplaçable par une association ternaire.....	10
Illustration 9:Contre-exemple : l'entité départs n'est pas remplaçable par une association ternaire.....	11
Illustration 10:Exemple d'entité quaternaire.....	11
Illustration 11:Contre-exemples de la normalisation des noms.....	12
Illustration 12:Contre-exemples de la normalisation des attributs (a).....	13
Illustration 13:Contre-exemples de la normalisation des attributs (b).....	14
Illustration 14:Normalisation des attributs des associations.....	14
Illustration 15:Cardinalité 1,1 et attributs d'une association.....	15
Illustration 16:les cardinalités sont toutes 1,1 donc c'est une association fantôme.....	15
Illustration 17:si un client ne peut pas régler la facture d'un autre client, alors l'association payer est inutile et doit être supprimée.....	16
Illustration 18:une association suffit pour remplacer les 4 associations participer en tant que	16
Illustration 19:Première forme normale : il peut y avoir plusieurs auteurs pour un livre donné.....	17
Illustration 20:Il y a redondance dans les colonnes constructeur et capacité.....	18
Illustration 21:Application de la troisième forme normale de Boyce-Codd.....	18
Illustration 22:Exemples de dépendances fonctionnelles.....	19
Illustration 23:Dépendance fonctionnelle non élémentaire, mais directe.....	20
Illustration 24:Graphe de couverture minimale.....	20
Illustration 25:Identification des entités et des associations sur un graphe de couverture minimale.....	21
Illustration 26:Schéma entités-associations normalisé obtenu à partir du graphe de couverture minimale....	21
Illustration 27:Sans historisation des emprunts, pas de problème.....	22
Illustration 28:Même pour une entité historisée, il faut éviter que la date entre dans l'identifiant.....	23
Illustration 29:Dépendances fonctionnelles commentées.....	24
Illustration 30:dépendance non élémentaire et sans enfant sur un graphe de couverture minimal.....	25
Illustration 31:Contenu de la table clients, avec en première ligne les intitulés des colonnes.....	28
Illustration 32:Schéma relationnel simple entre deux tables.....	29
Illustration 33:Traduction d'une association de type 1 : n.....	30

Illustration 34:Traduction d'une association de type n : m.....	31
Illustration 35:Traduction d'une association de type 1 : 1.....	31
Illustration 36:Traduction alternative d'une association de type 1 : 1.....	32
Illustration 37:Traduction d'une association ternaire.....	33
Illustration 38:Sacrifice de la troisième forme normale.....	35
Illustration 39:Tables particulières en rétro-conception.....	36
Illustration 40:Association binaire de type 1 : n (monter), liée à une association binaire de type n : m (participer).....	38
Illustration 41:Association ternaire remplacée par deux associations binaires.....	39
Illustration 42:Association binaire de type n:m (parier), liée à une autre association binaire de type n:m.....	40
Illustration 43:Agrégation et entités de codification.....	41
Illustration 44:Utilisation de clés primaires partiellement étrangères.....	42
Illustration 45:Représentation des identifiants relatifs.....	43
Illustration 46:Représentation des identifiants relatifs sur le graphe de couverture minimale.....	44
Illustration 47:Numérotations alternatives des exemplaires.....	45
Illustration 48:Représentation des sous-entités.....	46
Illustration 49:Séparation des informations complémentaires par héritage.....	48
Illustration 50:Spécialisation des associations en présence de sous-entités.....	49