Database Concepts and Design

Chapitre 5 : MERISE

Table des matières

1 Modèle conceptuel de données (MCD)

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

1.1 Schéma entités-associations

1.1.1 Rappels

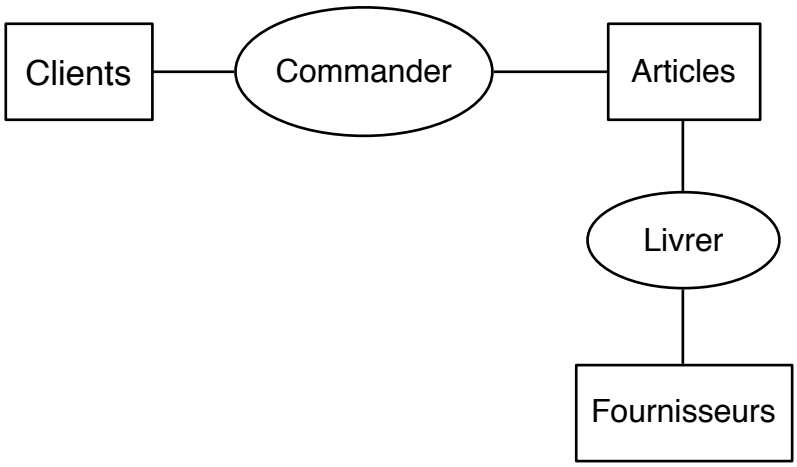
Une entité est une population d'individus homogènes. Par exemple, les produits ou les articles vendus par une entreprise peuvent être regroupés dans une même entité, car d'un article à l'autre, les informations ne changent pas de nature (à chaque fois, il s'agit de la désignation, du prix unitaire, etc.).

Le schéma ci-dessous représente les entités :

Remarque : ce schéma sera complété au fur et à mesure de l'avancée de ce chapitre. Il comporte volontairement des erreurs de conception qui seront identifiées et corrigées ultérieurement.

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.

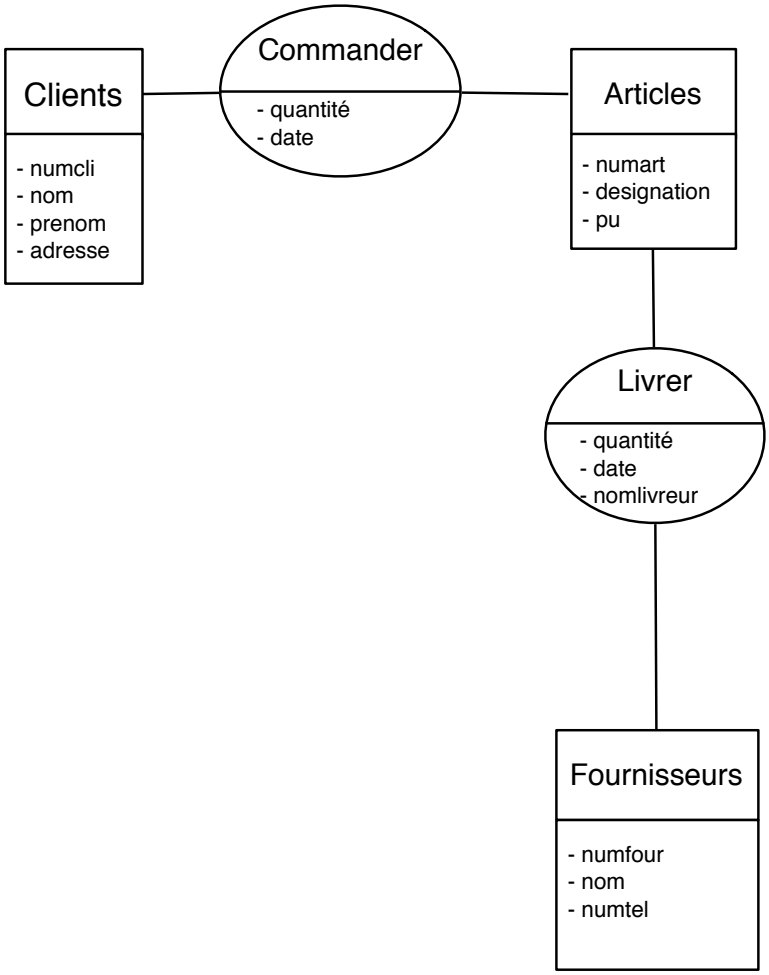


Remarquons que dans ce schéma, les entités clients et fournisseurs ne sont pas liées directement, mais indirectement, via l'entité articles, ce qui est assez naturel.

1.1.2 Attributs et identifiants

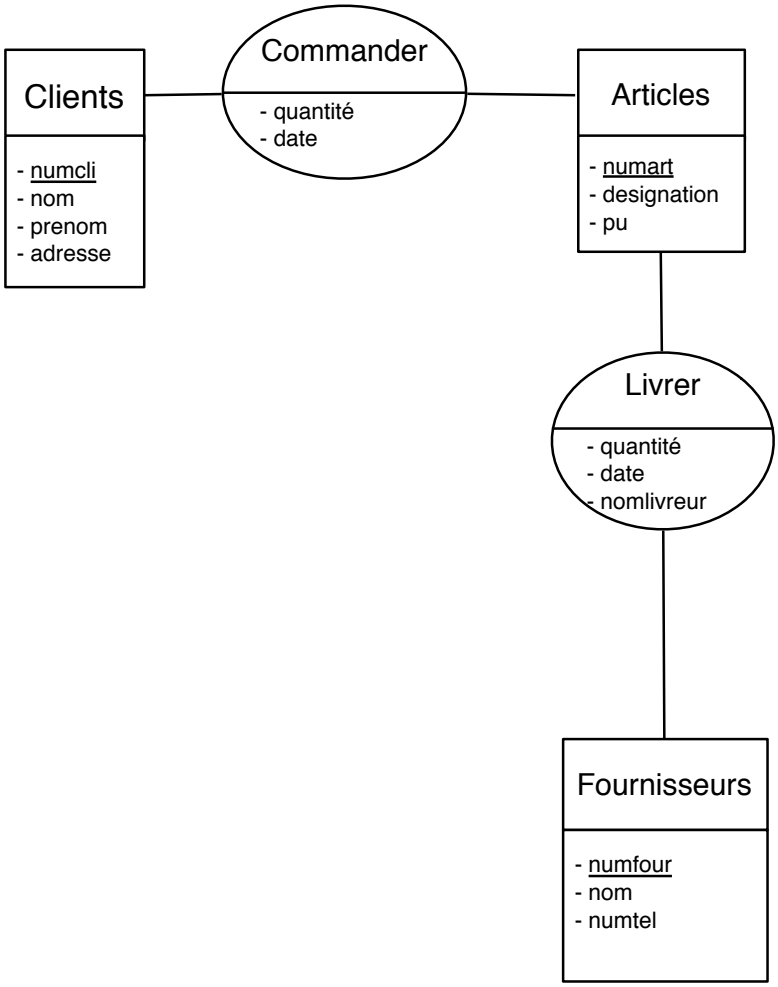
Un attribut est une propriété d'une entité ou d'une association.

Toujours dans notre exemple, le prix unitaire est un attribut de l'entité 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.



Une entité et ses attributs ne doivent traiter que d'un seul sujet afin d'assurer une certaine cohérence au modèle. Dans notre exemple, il est donc préférable de ne pas mettre les informations relatives aux fournisseurs dans l'entité des articles mais plutôt dans une entité fournisseurs séparée (et liée à l'entité articles via l'association livrer).

Ensuite, 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 (ce qui peut être vérifié par les dépendances fonctionnelles vues dans un autre chapitre de ce cours).

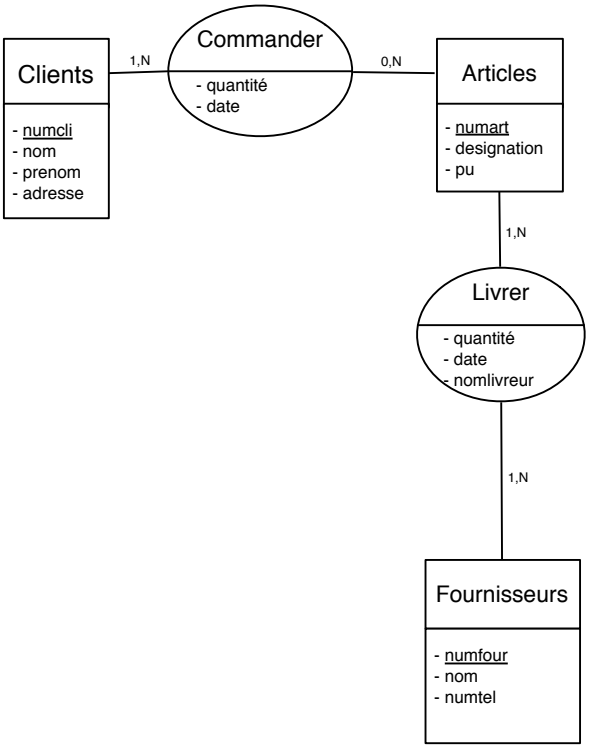


Remarque : une entité possède au moins un attribut (son identifiant). Au contraire, une association peut être dépourvue d'attribut.

1.1.3 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). On obtient alors le schéma entités-associations complet :



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 (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 (c'est le cas pour une liste pré-établie d'articles par exemple).

Ceci dit, 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.

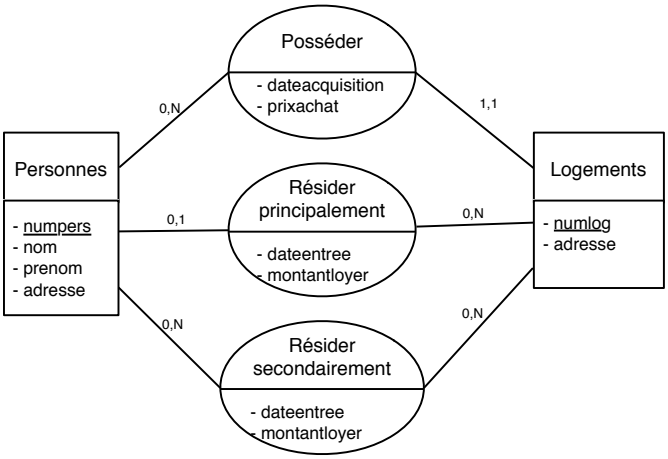
Notons que sur notre exemple, un article peut être commandé par plusieurs clients. Cela provient du fait que tous les crayons rouges ne sont pas numérotés individuellement, mais portent un numéro d'article collectif. En toute rigueur, notre entité articles aurait du s'appeler types d'article. Ainsi, 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, d'autres concepteurs peuvent très légitimement faire le choix inverse qui consiste à numéroter individuellement chaque crayon rouge.

La seule difficulté pour établir correctement les cardinalités est de se poser les questions dans le bon sens. Autour de l'association commander, par exemple (la syntaxe des questions est volontairement maladroite car elle guide le sens de lecture du MCD) :

- 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 clients ? » et cette fois-ci, la réponse est « entre 0 et plusieurs ».

1.1.4 Associations plurielles

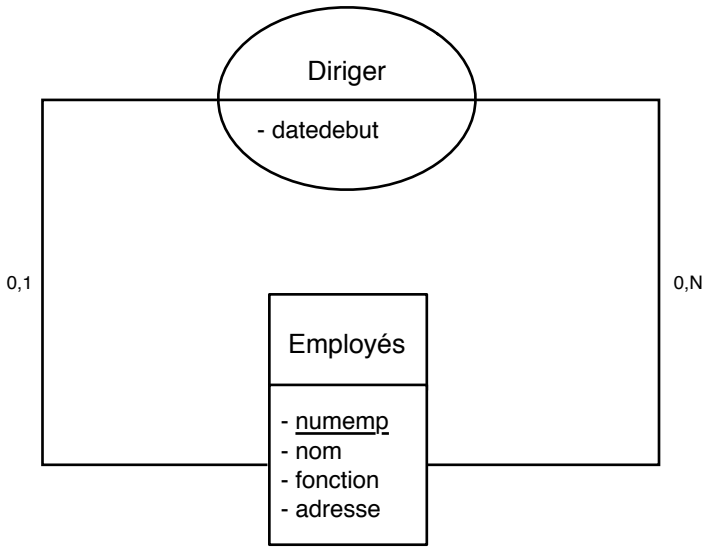
Deux mêmes entités peuvent être plusieurs fois en association :



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.

1.1.5 Association réflexive

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



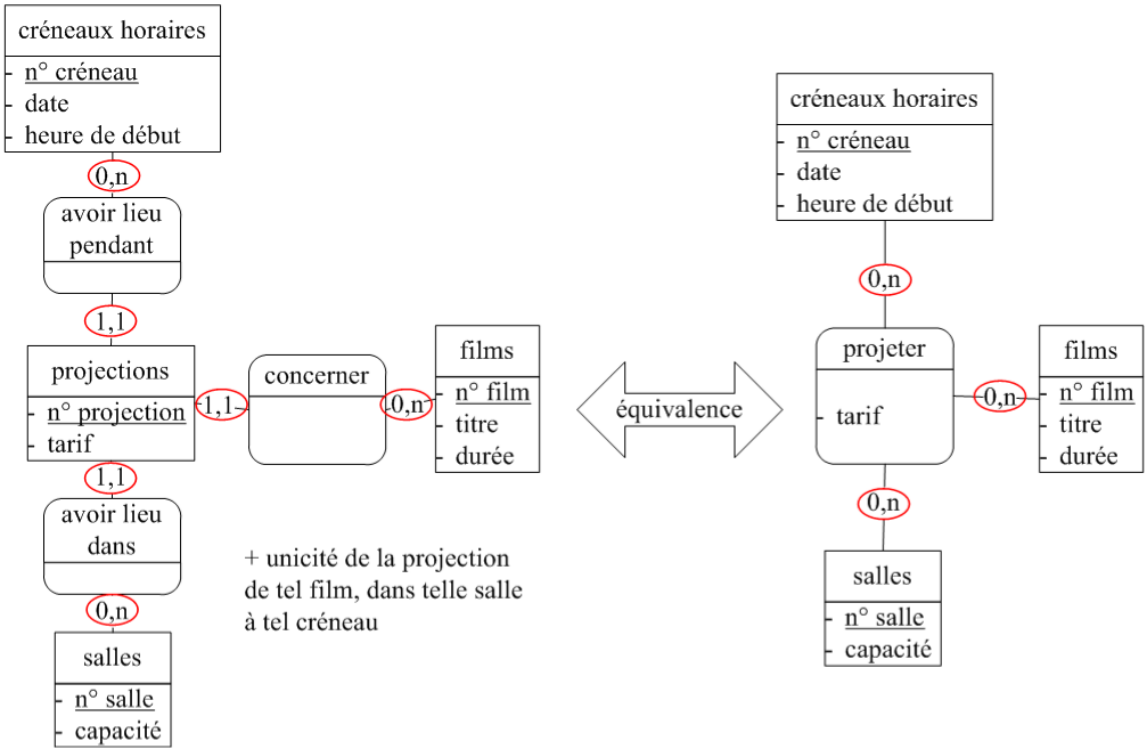
Dans cet exemple, tout employé est dirigé par un autre employé (sauf le directeur général) et un employé peut diriger plusieurs autres employés, ce qui explique les cardinalités sur le schéma.

1.1.6 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.

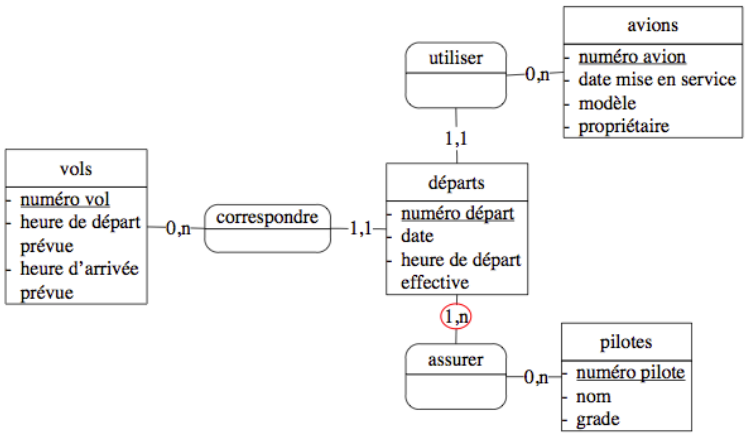
La deuxième condition qu'il faut impérativement satisfaire est la règle de normalisation des attributs des associations. Cette règle conduit parfois à l'apparition d'associations qui établissent un lien sémantique entre 3 entités ou plus.

Sur l'exemple de la figure suivante issu d'un cinéma, 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 (grâce à la date dans l'entité créneaux). On peut donc la remplacer par une association projeter branchée aux trois entités salles, créneaux horaires et films. On parle alors d'association ternaire.



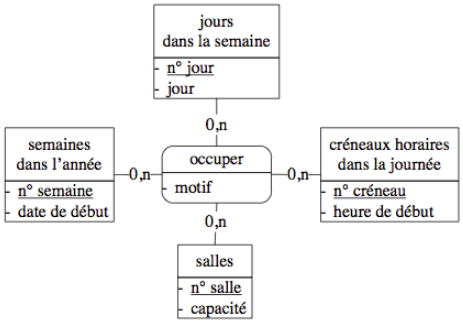
La difficulté de concevoir une association ternaire (ou plus) directement est d'établir les bonnes cardinalités. Il est donc conseillé d'en passer par un schéma entités-associations dans lequel on ne trouve que des associations binaires, puis de repérer les entités remplaçables par des associations, comme sur la figure précédente 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 (figure suivante), car le concepteur peut s'apercevoir que l'une des cardinalités maximales ne convient pas.



Dans ce 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 figure suivante. Là encore, le conseil pour être sûr de la légitimité de cette association 4-aire, est de vérifier les cardinalités sur un schéma intermédiaire faisant apparaître à la place, une entité occupations et quatre associations binaires.



1.2 Règles de normalisation

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

1.2.1 Les bonnes manières dans un schéma entités-associations

Normalisation des entités (importante) : toutes les entités qui sont remplaçables par une association doivent être remplacées.

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 ou le signe d'une redondance.

Normalisation des identifiants : chaque entité doit posséder un identifiant.

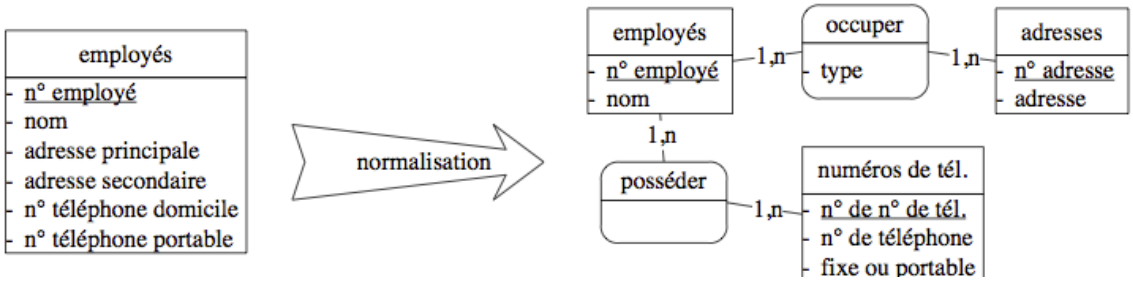
Conseils :

- 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'autres 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 (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).

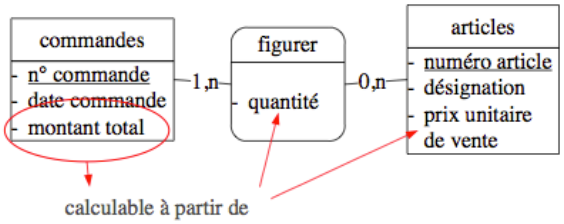
Conclusion : l'identifiant sur un schéma entités-associations (et donc la future clé primaire dans le schéma relationnel) doit être un entier, de préférence incrémenté automatiquement (lorsque le SGBD le permet).

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 et d'autre part, les attributs calculables induisent un risque d'incohérence entre les valeurs des attributs de base et celles des attributs calculés.

Exemple d'attribut en plusieurs exemplaires :



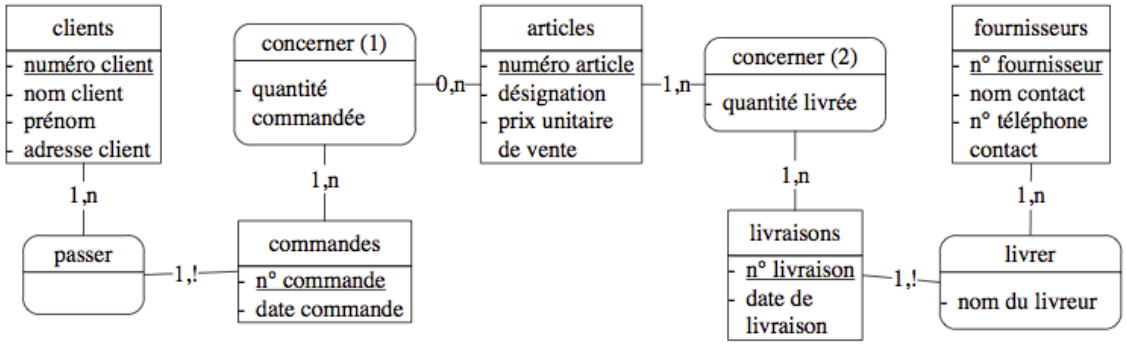
Exemple d'attribut calculable à retirer du schéma :



D'autres d'attributs calculables classiques sont à éviter, comme l'âge (qui est calculable à partir de la date de naissance).

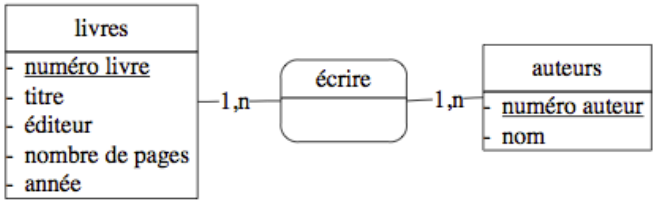
Normalisation des attributs des associations (importante): les attributs d'une association doivent dépendre directement des identifiants de toutes les entités en association.

Par exemple, sur la figure suivante 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 :

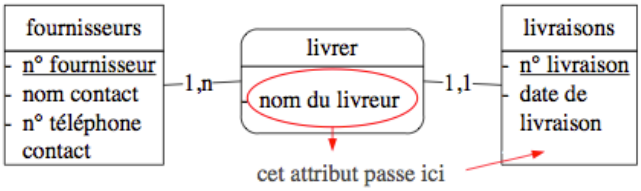


L'inconvénient de cette règle de normalisation est qu'elle est difficile à appliquer pour les associations qui ne possèdent pas d'attribut. Pour vérifier malgré tout qu'une association sans attribut est bien normalisée, on peut donner temporairement à cette association un attribut imaginaire (mais pertinent)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

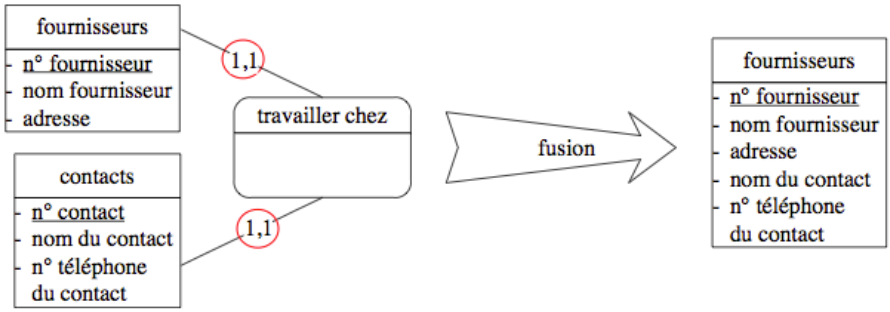
Par exemple, entre les entités livres et auteurs, l'association écrire ne possède pas d'attribut. Imaginons que nous ajoutions un attribut pourcentage qui contient le pourcentage du livre écrit par chaque auteur (du même livre). Comme cet attribut pourcentage dépend à la fois du numéro de livre et du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



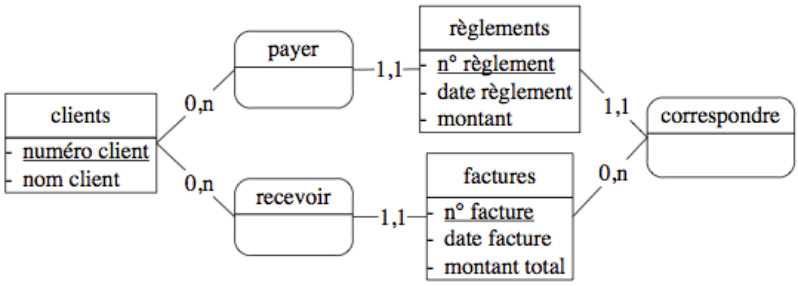
Autre conséquence de la normalisation des attributs des associations : une entité avec une cardinalité de 1,1 ou 0,1 aspire les attributs de l'association :



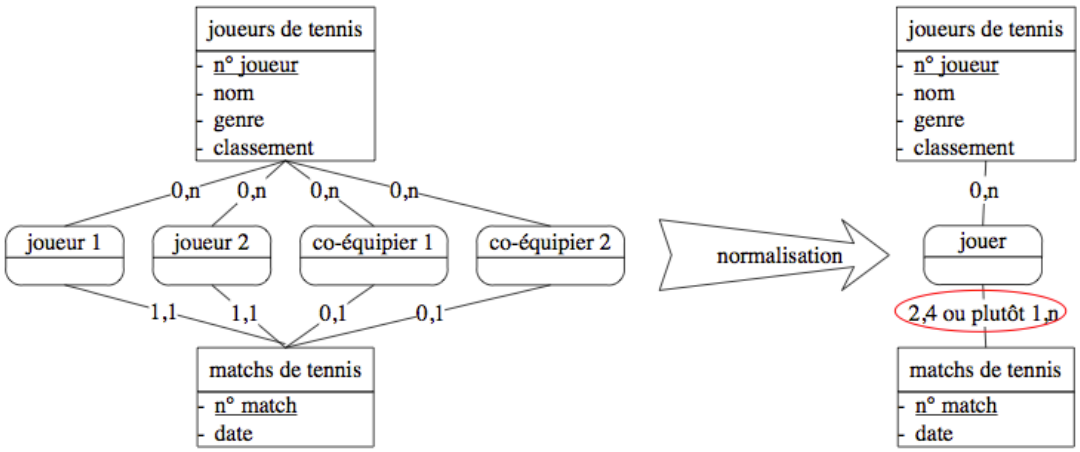
Normalisation des associations (importante) : il faut éliminer les associations fantômes, redondantes ou en plusieurs exemplaires.



(a) les cardinalités sont toutes 1,1 donc c'est une association fantôme



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



(c) une association suffit pour remplacer les 4 associations `participer en tant que ...`

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 figure précédente (b), si on supprime l'association payer, on peut retrouver le client qui a payé le règlement en passant par la facture qui correspond.

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 figure précédente (c) 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.

Cela signifie également qu'on ne modélise pas les cardinalités minimales qui valent plus de 1 car ce genre de valeur est aussi amenée à évoluer. Par ailleurs, avec une cardinalité maximale de 0, l'association n'aurait aucune signification.

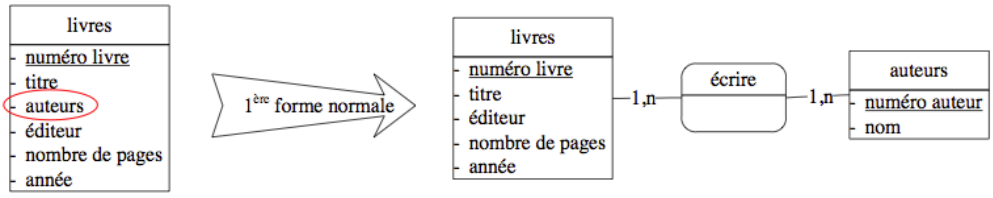
Dans un SGBD relationnel, nous pourrions assurer les cardinalités valant 2, 3 ou plus, via l'utilisation de déclencheurs (triggers). Mais cette notion n'est pas abordée dans ce cours.

1.2.2 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.

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 :



Deuxième forme normale : en plus d'être en première 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).

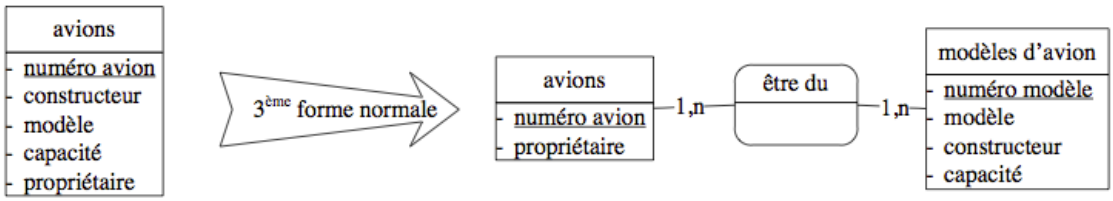
Cette deuxième forme normale peut être oubliée si on suit le conseil de n'utiliser que des identifiants non composés et de type entier. En vérité, elle a été vidée de sa substance par la règle de normalisation des attributs des associations.

Considérons malgré tout le contre-exemple suivant : dans une entité clients dont l'identifiant est composé des attributs nom et prénom, la date de fête d'un client ne dépend pas de son identifiant en entier mais seulement de prénom. Elle ne doit pas figurer dans l'entité clients, il faut donc faire une entité calendrier à part, en association avec l'entité clients.

Troisième forme normale (importante) : en plus d'être en deuxième forme normale, 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.

Exemple :



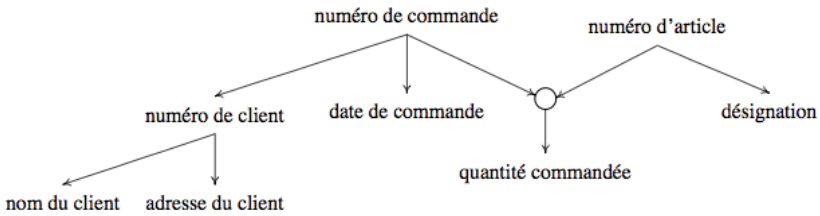
1.2.3 Dépendances fonctionnelles

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. Ce point n'est pas approfondi davantage dans ce chapitre car il fait l'objet d'un chapitre séparé.

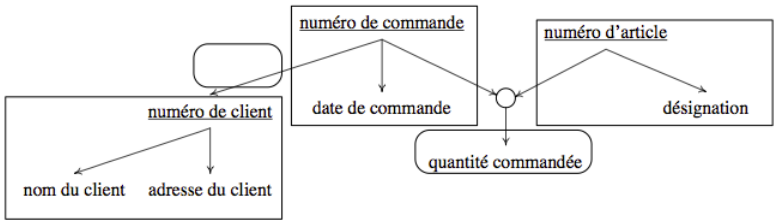
1.2.4 Traduction vers un schéma entités-associations

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.

Graphe de couverture minimale :



À partir du graphe de couverture minimale, le schéma entités-associations normalisé correspondant apparaît naturellement, en suivant quelques étapes simples.



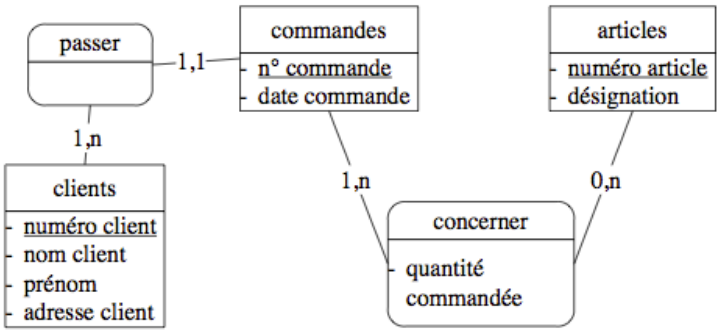
Etape 1 : il faut repérer et souligner les identifiants.

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

Etape3 : ensuite, 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.

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

Etape 5: enfin, 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 en un schéma entités-associations normalisé est illustrée dans la figure suivante :



1.3 Méthodologie de base

Face à une situation bien précise, 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 ;
- effectuer les corrections nécessaires.

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, comme souvent en modélisation il faut itérer plusieurs fois les étapes successives, pour espérer converger vers une modélisation pertinente de la situation.

2 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.

2.1 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.

Numéro client	Nom	Prénom	Adresse
1	DUPONT	Michel	127 rue …
2	DURANT	Gérard	62 boulevard…
3	BIDOCHON	Robert	69 avenue…
4	MARTIN	Claire	24 rue…
…	…	…	…

2.2 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 devrait pas, en principe, changer au cours du temps.

Par ailleurs, 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). La Colonne2 doit être sans doublons (bien souvent il s'agit d'une clé primaire). On dit alors que la Colonne1 est clé étrangère et qu'elle référence la Colonne2.

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 :

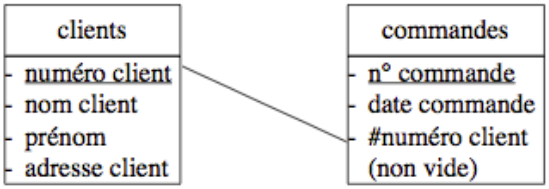
- une même table peut avoir plusieurs clés étrangères mais une seule clé primaire (éventuellement composée de plusieurs colonnes) ;
- une colonne clé étrangère peut aussi être primaire (dans la même table) ;

- 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 valeur 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.

2.3 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.



Certains éditeurs inscrivent sur le connecteur un symbole 1 côté clé primaire et un symbole ∞ côté clé étrangère (à condition que celle-ci ne soit pas déjà clé primaire). Il faut prendre garde avec cette convention, car le symbole ∞ se trouve du côté opposé à la cardinalité maximale n correspondante.

2.4 Traduction d'un MCD en un MLDR

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

Notations : 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.

En fait, 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).

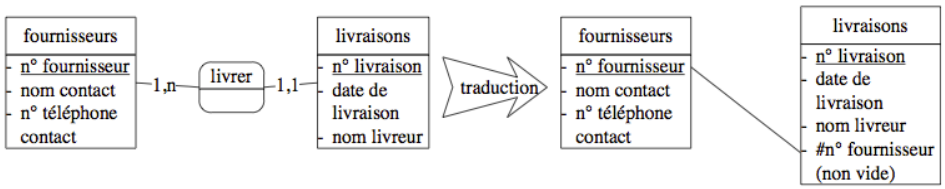
Règle 1 : toute entité devient une table dans laquelle les attributs deviennent les colonnes. L'identifiant de l'entité constitue alors la clé primaire de la table.

Par exemple, l'entité articles précédemment présentée 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.

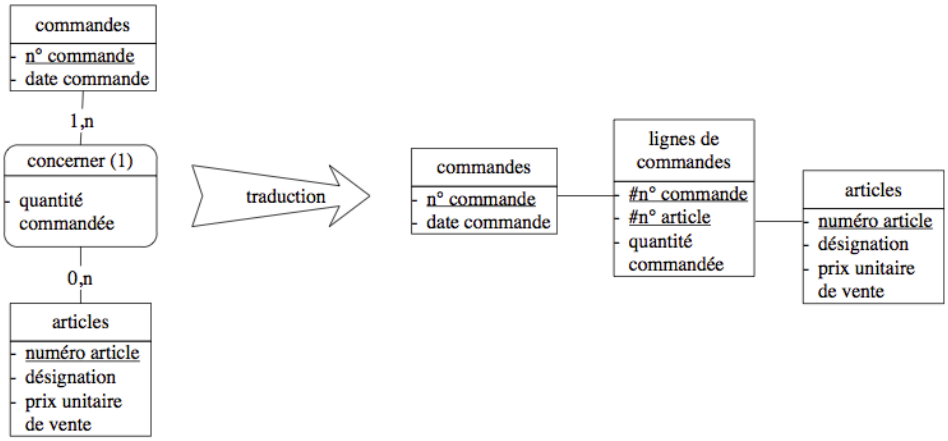
Par exemple :



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.

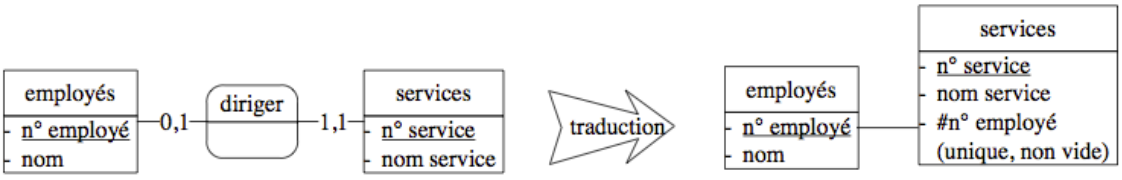
Par exemple :



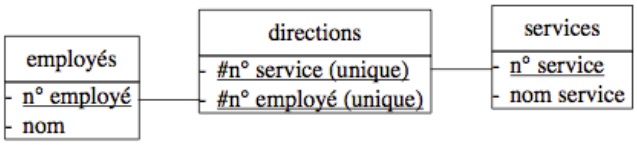
Règle 4 : une association binaire de type 1 : 1 est traduite comme une association binaire de type 1 : n sauf que la clé étrangère se voit imposer une contrainte d'unicité en plus d'une éventuelle contrainte de non vacuité (cette contrainte d'unicité impose à la colonne correspondante de ne prendre que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

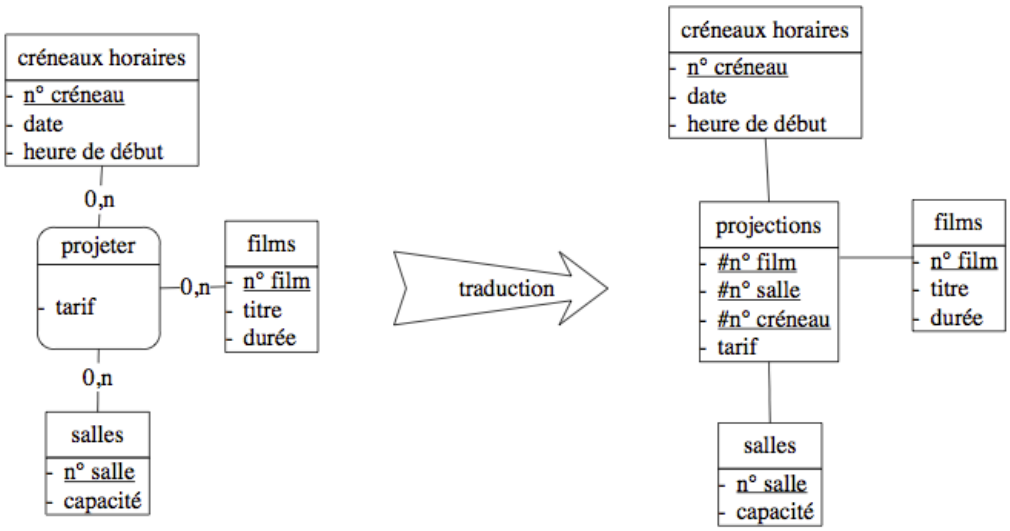


En réalité, la règle 4 proposée ici considère qu'une association binaire de type 1: 1 correspond à une association binaire de type 1 : n particulière. Une alternative consiste à voir une association binaire de type 1: 1 comme une association binaire de type n: m particulière. Il suffit pour cela d'ajouter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



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 :



3 Modèle physique de données (MPD)

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

3.1 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.

3.2 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 dénormaliser 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 dénormalisée) ; 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). Cette solution est fortement déconseillée puisqu'elle nuit de fait à l'intégrité référentielle.