[IMAC S2] Programmation Back-end

TD 03 : Bases de données - Merise

Support - Partie I

Pascale Ho - 2018 - Ingénieur IMAC (Mise à jour le 12/03/2018)

Introduction

Jusqu'à présent on a appris à manipuler le langage PHP (via notre bibliothèque de films), à créer des classes et à instancier des objets. Mais jusqu'à présent la durée de vie de tous nos éléments créés ne dépassent pas la durée de l'exécution du script. Un des moyens de rendre nos informations persistantes c'est de les sauvegarder dans une base de données. On va pouvoir retrouver ces objets d'une visite à l'autre du site.

Pour cela on d'abord apprendre à créer des bases de données et à les manipuler via phpMyAdmin. **Il va y avoir beaucoup de "théorie" à voir niveau Merise, donc je vous conseille de retenir en priorité les notions abordées dans le TL;DR**.

TL;DR

MCD

- À partir des informations et des règles de gestion qu'on a recueillies, on va construire un Modèle Conceptuel de Données.
- Une entité est unique et est décrite par un ensemble de **propriétés ou attributs (ou caractéristiques)**. Elle doit posséder un attribut particulier qu'on appelle **identifiant ou clé primaire**.
- Une **association** définit un lien entre une ou plusieurs entités.Une association peut parfois posséder des propriétés, dans ce cas il a juste besoin de tous les identifiants des entités qui sont reliées pour accéder à ces propriétés.
- Entre chaque patte d'une association se trouvent les **cardinalités** : ce sont des informations précisant le nombre de fois minimal et maximal d'interventions d'une entité dans une association. Il ne peut y avoir que quatre principaux cas de figure possibles pour les cardinalités : **(0,1), (0,N), (1,1) et (1,N)**.
- Par raccourci de langage on va parler très souvent que des cardinalités maximales. Lorsqu'on l'une des deux cardinalités maximales de l'association vaut 1 et l'autre N, on dira qu'on a une **association 1-N**. Si nous avons deux cardinalités à N, on dira

qu'il s'agit d'une **association N-M**. (très souvent les propriétés supplémentaires se retrouvent dans des associations N-M)

MLD

- Le Modèle Logique de Donnés servira de "plan" de notre base de donnés. Il est composé uniquement de ce que l'on appelle des **relations ou tables**, issues à la fois des entités du MCD mais aussi d'associations, dans certains cas.
- Règles de conversion :
    - En règle générale, **toute entité du MCD devient une relation** dont la **clé primaire *(Primary Key, PK)*** est l'identifiant de l'entité.
    - Pour les **associations N-M, on crée une nouvelle relation** dont leur clé primaires sont les clés primaires des relations issues des entités de l'association.
    - Pour les **associations 1-N, si l'une des deux cardinalités vaut (1,1)** : on rajoute une **clé étrangère (Foreign Key, FK)** dans la relation qui correspond à l'entité se situant du côté de cette cardinalité (1,1). Cette clé fera donc référence à la clé de la relation correspondant à l'entité reliée par l'association dans le MCD.

Création de bases de données avec Merise

Rappel sur Merise (CM 02)

Merise est une **méthode d'analyse et de conception** basée sur la séparation des données (structure des informations) et des traitements (réaction aux événements). On va s'intéresser uniquement de la partie sur les données. Il existe plusieurs niveaux d'abstractions :

- niveau **conceptuel** : exprime les choix de gestion, les objectifs de l'organisation. Du point de vue des données : signification, structure et liens.
- niveau **logique** : exprime la forme que doit prendre l'outil informatique pour être adapté à l'utilisateur. Il décrit le schéma de la bdd.
- niveau **physique** : traduction des choix techniques. Définit l'implantation physique des données.

Chaque niveau possède un modèle :

- niveau conceptuel : **Modèle Conceptuel des Données (MCD)**. Décrit des données et les relations (entité, relation/association, propriétés/attributs)
- niveau logique : Modèle Logique des Données (MLD)
- niveau physique : Modèle Physique des Données (MPD)

Ici on va juste apprendre à concevoir un MCD (et vite fait un MLD), qui nous permettra de décrire une ensemble de données afin de les intégrer ensuite dans une bdd.

Modèle Conceptuel de Données (MCD)

Un MCD est une représentation graphique et structurée des informations. Il se base sur deux notions principales : **entités et associations**, d'où son autre nom : **schéma Entité/Association**.

Rappel CM 02 : Étapes de construction du MCD (Facultatif)

Avant de créer un MCD et une base de données, il faut recueillir les besoins des utilisateurs de votre site. Pour vous aider voici un processus d'élaboration du MCD qui se fait en 3-4 étapes. **Cette partie est facultative**, mais cela peut vous aider à construire plus facilement votre modèle.

Mise en place des règles de gestion

Le principe consiste à établir à partir des besoins des règles et des données à conserver. Il arrive que ce soit le client qui vous fournit les règles, mais des fois ce n'est pas le cas. En tant que développeu.r.se, vous devez chercher les informations ou plus de détails en interrogeant les clients (ou par vous même).

On va reprendre l'exemple du cours précédant : Gaëtan veut créer une base de données pour l'emprunt de matériel. Il faut que vous arriviez à établir cet exemple de règles :

- pour chaque matériel, on doit connaître le nom de l'objet, un résumé et un type
- chaque matériel est identifié par une référence composée de lettres et de chiffres
- chaque matériel a un état : neuf, très bon, bon, correct, dégradé
- un utilisateur est identifié par un numéro et on doit retenir son nom, prénom, adresse email, téléphone, son statut (élève, professeur).
- un utilisateur peut faire zéro, un ou plusieurs emprunts qui concernent chacun un et un seule exemplaire. Pour chaque emprunt on connaît la date et le délai accordé (en nombre de jours).

Élaboration du dictionnaire des données

Alors c'est une **étape intermédiaire**, pas très indispensable sauf si on travaille dans une bdd d'un volume conséquent. C'est un document sous forme de tableau qui regroupe toutes les données qu'on va conserver dans notre bdd (et qui seront dans notre MCD).

D'après l'exemple on obtiendrait ce dictionnaire :

Libellé	Description	Type	Taille	Remarque

ref_materiel	référence du matériel	AN	15	sert d'identifiant
nom_materiel	nom du matériel	AN	50	
resume_materiel	description du matériel	AN	1000	
id_type	identifiant du type de matériel	N		
nom_type	nom du type de matériel	AN	30	
etat_materiel	état du matériel	AN	10	valeurs : neuf, très bon, bon, correct, dégradé
id_utilisateur	identifiant de l'utilisateur	N		
nom_utilisateur	nom de l'utilisateur	A	30	
prenom_utilisateur	prenom de l'utilisateur	A	30	
email_utilisateur	adresse e-mail de l'utilisateur	AN	100	
rue_utilisateur	rue où habite l'utilisateur	AN	100	
ville_utilisateur	ville où habite l'utilisateur	A	50	
code_postal_utilisateur	code postal de la ville	AN	5	
telephone_utilisateur	numéro de téléphone de l'utilisateur	AN	15	
statut_utilisateur	type d'utilisateur	AN	5	valeurs : élève/prof
id_emprunt	identifiant de l'emprunt			
date_emprunt	date de l'emprunt	Date		format AAAA-MM-JJ
delais_emprunt	délai autorisé lors de l'emprunt du matériel	N	2	s'exprime en nombre de jours (maximum 15)

Pour chaque donnée, on indique son libellé, sa description, son type de donnée (Alphabétique, Numérique, AlphaNumérique, Date, Booléen), sa taille (nombre de caractères maximal) et des remarques supplémentaires.

Après **ce qu'il faut retenir** c'est qu'à partir des règles de gestion il faut être capable d'établir des données **élémentaires**, c'est-à-dire qu'elles **ne sont pas issues de calculs** d'autre données existantes et elles **ne doivent être composées** (dans notre exemple l'adresse de l'utilisateur est obtenue à partir d'une rue, d'une ville et d'un code postal : ces trois dernières informations seront figurées dans le MCD).

Recherche des Dépendances Fonctionnelles

#Théorème de mathématiques :

> Soit deux propriétés/données P1 et P2. On dit que P1 et P2 sont reliées par une dépendance fonctionnelle (DF) si et seulement si une valeur de P1 permet de connaître une et seule occurrence de P2.

On représente cette dépendance ainsi : **P1 → P2**. Ainsi avec une valeur de P1 on peut retrouver P2.

Autres exemples :

P1 → P2, P3 à partir de P1 on peut retrouver P2 et P3

P1, P2 → P3 à partir de P1 et P2 on peut obtenir P3

P1, P2 → P3, P4, P5 à partir de P1 et P2 on peut obtenir P3, P4 et P5

Une DF doit être :

- **élémentaire** : si on a P1 → P3, alors P1, P2 → P3 n'est pas élémentaire
- **directe :** si P1 → P2 et P2 → P3, alors P1 → P3 n'est pas directe (on l'obtient par transitivité)

Toute DF qui n'est ni élémentaire ni directe n'a pas besoin d'être indiquée.

Le principe des DFs permet entre autres de distinguer les futures entités du MCD et leurs identifiants.

Retour sur l'exemple. On obtient alors les DFs suivantes :

id_type → nom_type

ref_materiel → nom_materiel, resume_materiel, etat_materiel

id_utilisateur → nom_utilisateur, prenom_utilisateur, email_utilisateur, rue_utilisateur, ville_utilisateur, code_postal_utilisateur, telephone_utilisateur, statut_utilisateur

id_emprunt → date_emprunt, delais_emprunt, id_utilisateur, ref_materiel

Élaboration du MCD

On va voir dans le prochain paragraphe comment on fait un MCD !

Entité

On parle **d'entité** tout ce qui peut être un objet, quelque chose de concret ou d'abstrait qui possède des mêmes caractéristiques (des personnes, des véhicules, des salles, des villes, des factures, de la nourriture, …). *(En vérité on devrait parler d'entité-type, les entités étant en fait des instances d'entité-types. Mais par simplicité on parle d'entité.)*

Une entité est unique et est décrite par un ensemble de **propriétés ou attributs (ou caractéristiques)**. Elle doit posséder un attribut particulier qu'on appelle **identifiant ou clé primaire** : celle-ci doit posséder des occurrences uniques qui sert à retrouver toutes les autres attributs à partir de celui-ci.

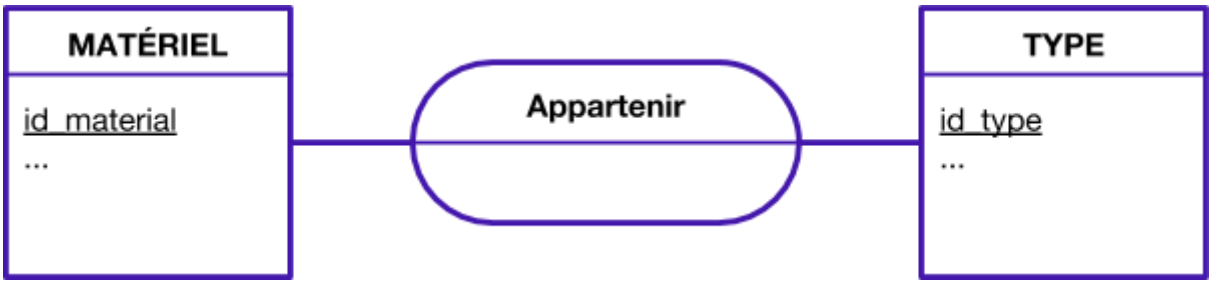
Schématiquement, une entité prend cette forme ----------→

Le nom de l'entité est inscrit dans la partie supérieure, dans la partie inférieure se trouvent les propriétés, dont l'identifiant qui est souligné.

Si on reprend notre bibliothèque et qu'on a suivi les méthodes précédentes, on schématise notre entité Utilisateur comme ceci :

Association

Une **association** définit un lien entre une ou plusieurs entités. Schématiquement, le nom d'une association est généralement un **verbe** définissant le lien entre les deux entités qui sont reliées par cette dernière. Une association peut parfois posséder des propriétés, dans ce cas il a juste besoin de tous les identifiants des entités qui sont reliées pour accéder à ces propriétés.



Cardinalités

Il manque une information essentielle pour la suite : à combiens d'éléments de l'autre entité chaque élément peut-il être associé ? C'est pour cela qu'existent les **cardinalités** : ce sont des informations précisant le nombre de fois minimal et maximal d'interventions d'une entité dans une association. On écrit une cardinalité comme ceci : **minimum, maximum**. Notre exemple précédent va avoir les cardinalités suivantes :

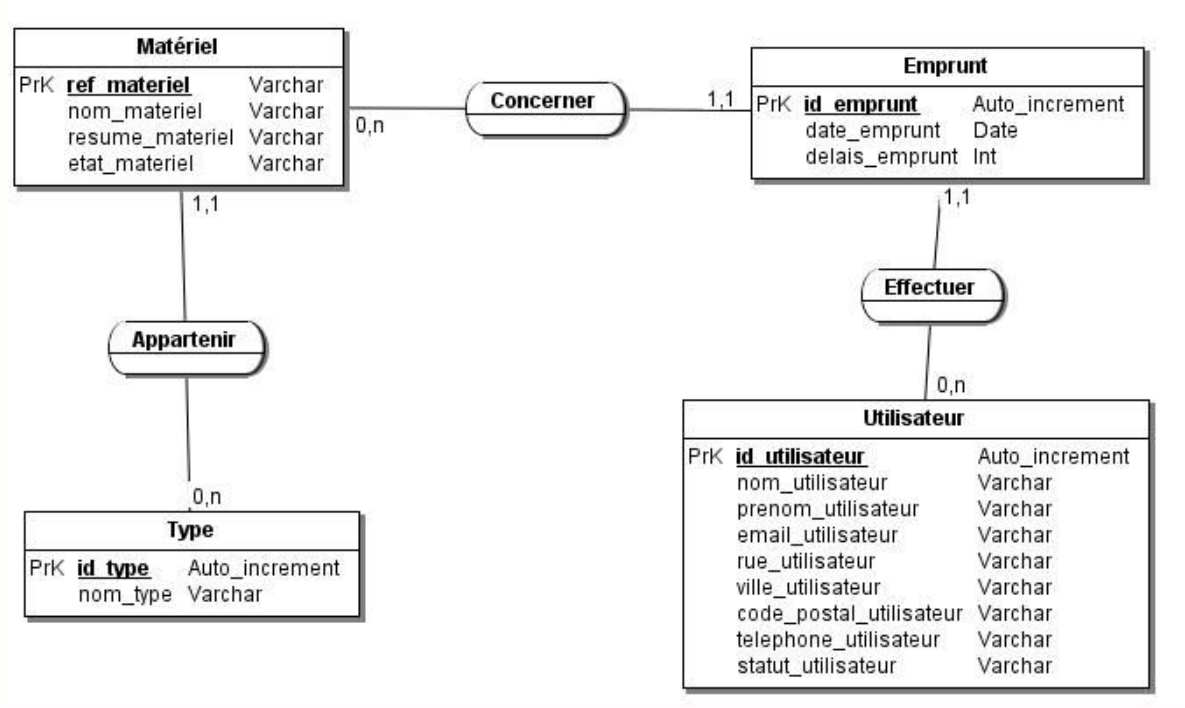
Comment lit-on cette association ?

- Un matériel appartient dans **un et un seul** type.
- Pour un type de matériel appartiennent **aucun, un ou plusieurs** matériels.

Il ne peut y avoir que quatre principaux cas de figure possibles pour les cardinalités : **(0,1), (0,N), (1,1) et (1,N)**, N signifiant plusieurs.

Il faut savoir que par raccourci de langage on va parler très souvent que des cardinalités maximales, car elles sont plus décisives sur l'architecture de notre base de données. Lorsqu'on l'une des deux cardinalités maximales vaut 1 et l'autre N, on dira qu'on a une **association 1-N**. Si nous avons deux cardinalités à N, on dira qu'il s'agit d'une **association N-M**. (très souvent les propriétés supplémentaires se retrouvent dans des associations N-M)

Un association 1-1 est un cas limite. Cela signifie que nous avons créé deux entités qui en réalité n'en forment qu'une seule, puisque chaque élément de l'une correspond à un élément de l'autre, et à un seul. Ce n'est pas vraiment une faute, mais on a toujours intérêt à se demander pourquoi avoir créé deux entités plutôt qu'une seule. Voici ce que pourrait donner notre MCD de la bibliothèque :



Il existe plusieurs softs pour créer des MCD, personnellement j'utilisais **JMerise** qui a l'avantage de générer du code SQL pour créer notre base de données ! Mais je vais quand même vous expliquer comment passer du MCD à la création de tables en SQL.

Dimension d'une association

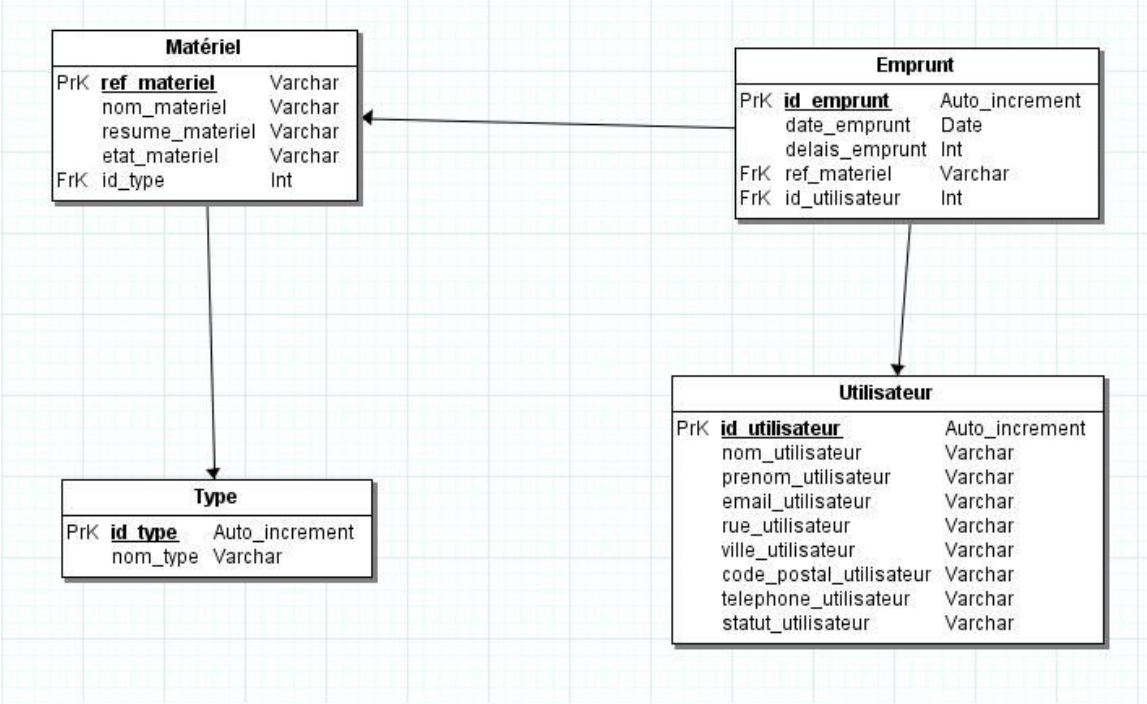
Il s'agit du nombre d'entités qui sont reliées par l'association. (Généralement les associations sont de dimension 2 mais il arrive qu'il peut y avoir plus de 2 entités pour une association.)

Du MCD aux tables SQL via MLD

Le **Modèle Logique de Données**, est contrairement au MCD beaucoup moins abstrait et plus proche de la base de données telle qu'elle existera sur les machines.

Le MLD est composé uniquement de ce que l'on appelle des **relations ou tables**, issues à la fois des entités du MCD mais aussi d'associations, dans certains cas.

Le MLD est normalement représenté de manière textuelle. Mais il existe une représentation graphique équivalente comme ci-dessous.



Règles de conversion

Conversion d'une entité

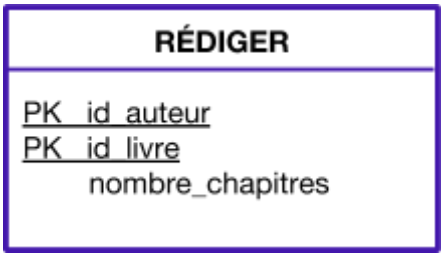
En règle générale, toute entité du MCD devient une relation dont la **clé primaire *(Primary Key, PK)*** est l'identifiant de l'entité. Ensuite selon les cardinalités des associations, les choses vont se passer différemment.

Cas des associations N-M

C'est-à-dire lorsque les cardinalités maximales de l'association sont (0,n) ou (1,n). Dans le MLD, cette association devient alors une **nouvelle table,** elle même en relation avec les deux tables produites par les deux entités du MCD. On parle alors de table de jonction. Cette table va posséder comme clé primaire toutes les clés primaires des autres entités reliées à elle.

Exemple :

L'association "Rédiger" peut être traduite en table :



Cas des association 1-N

Deux possibilités :

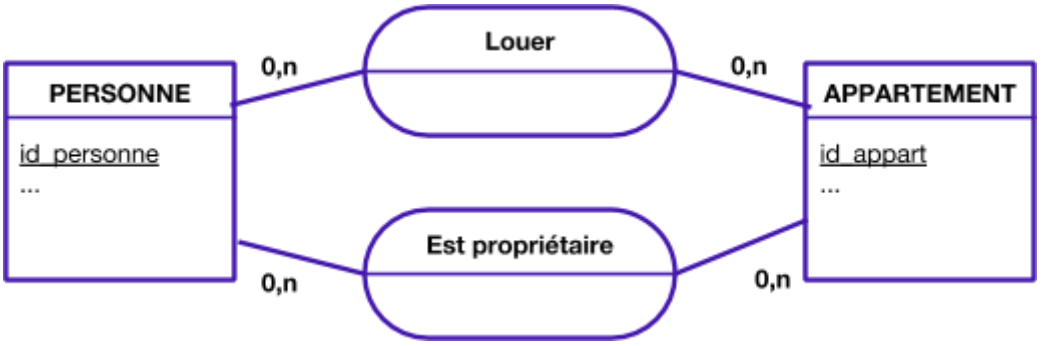
- Si l'une des deux cardinalités vaut (1,1) : on rajoute une **clé étrangère (Foreign Key, FK)** dans la relation qui correspond à l'entité se situant du côté de cette cardinalité (1,1). Cette clé fera donc référence à la clé de la relation correspondant à l'entité reliée par l'association dans le MCD. (c.f. la table Matériel de notre exemple de bibliothèque)
- Si l'une des deux cardinalités vaut (0,1) : Soit on est plutôt "à la cool" et on se dit qu'on va faire au plus simple, il suffit de créer une clé étrangère et lorsqu'il n'y a pas d'occurrences, on mettra une valeur null à cette clé. Soit on est psycho-rigide et on se dit que ce n'est pas une bonne chose d'avoir des valeurs Null dans une table, et vaut mieux être secure et créer une table de jointure, comme pour le cas d'une association N-M (dans ce cas il faut faire un contrôle pour être sûr.e que chaque relation n'apparaît pas plus d'une fois dans la table de jonction…). À vous de voir.

Vous voilà avec votre "plan" de construction de votre base de données !! Si vous avez utilisé un éditeur de MCD comme JMerise il existe une fonctionnalité qui permet de générer automatiquement un MLD *(et du code SQL pour la construction de votre base de données).* Mais on va s'y attaquer et commencer à manipuler dans la partie suivante.

Cas particuliers d'associations

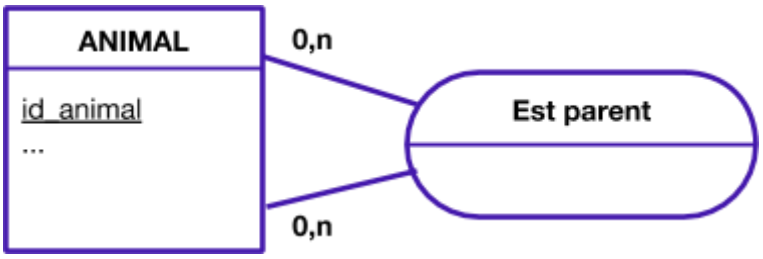
Relations multiples

Il est possible que deux entités soient reliées à la fois par plus d'une relation.



Réflexivité

Une association peut concerner des éléments d'une même entité.



Résumé

Esquisse d'une méthode générale :

1. Identifier l'ensemble des entités. Décomposer celles-ci autant que nécessaire, en se demandant si, pour chacune d'elles, il existe des propriétés dont les valeurs sont redondantes. Si oui, pour ~100% de ces cas, ces propriétés doivent être "externalisées" vers une nouvelle entité.

2. Éliminer les valeurs multiples et les synonymes.

3. Quelles sont les associations ? Pour chaque association entre A et B, se demande "un élément de A entre-t-il plusieurs fois en relation avec le même élément de B ?" Si c'est non, c'est une association simple. Si c'est oui, on recherche les informations qui différencient ces relations. Si ces informations ne sont pas redondantes, elles sont des attributs de l'association.

4. Vérifier que toute association multiple, se justifie par le fait que chacune des entités exprime des significations différentes.