Bases de Données 1
#2 - Compléments pour conception

Matthieu Nicolas
Polytech S5 - II
Slides réalisées à partir de celles de Claude Godart et Malika Smaïl

Plan

- Notions simples
- Réflexions sur classes, attributs et associations
- Notions avancées
- Patrons de conception

Notions simples

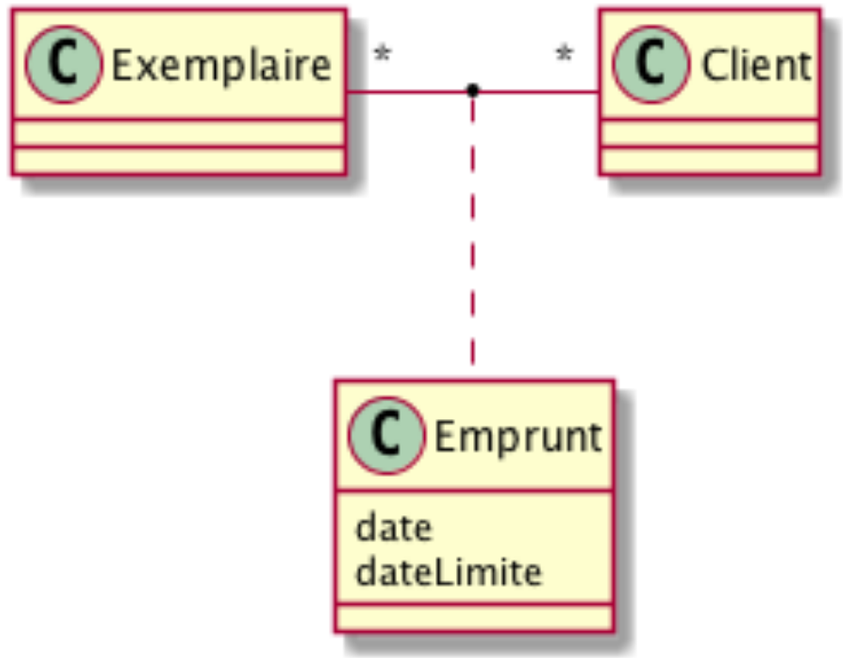
Base de Données 1
#2 - Compléments pour conception

Classe d’association - 1

- Notion aperçue lors de l’exercice Bibliothèque (TD1, ex1)

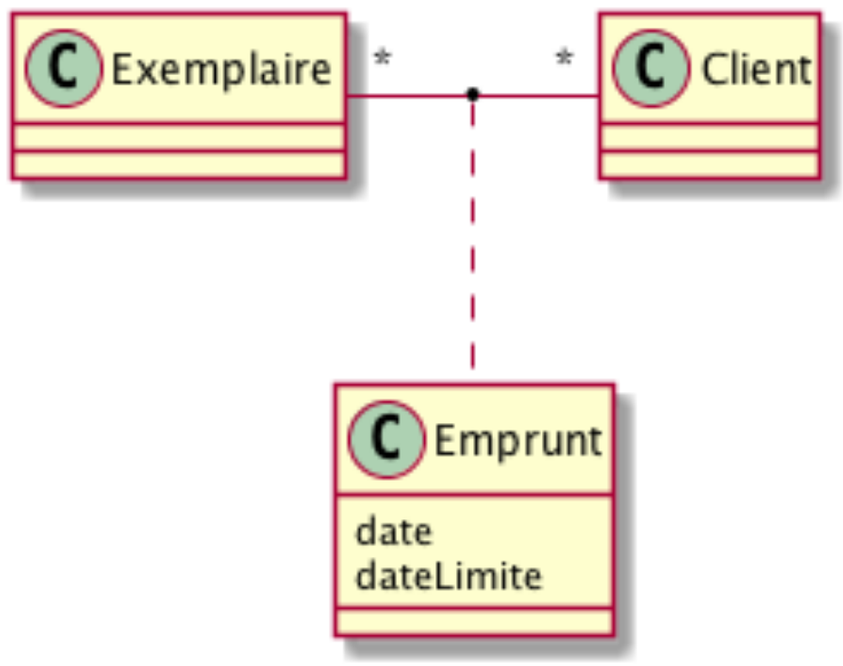
- *Un client peut emprunter des exemplaires d’ouvrages*

- *La date de l’emprunt* et la *date limite de retour* **dépendent du couple** (Exemplaire, Client), pas d’un Exemplaire seul ou d’un Client seul



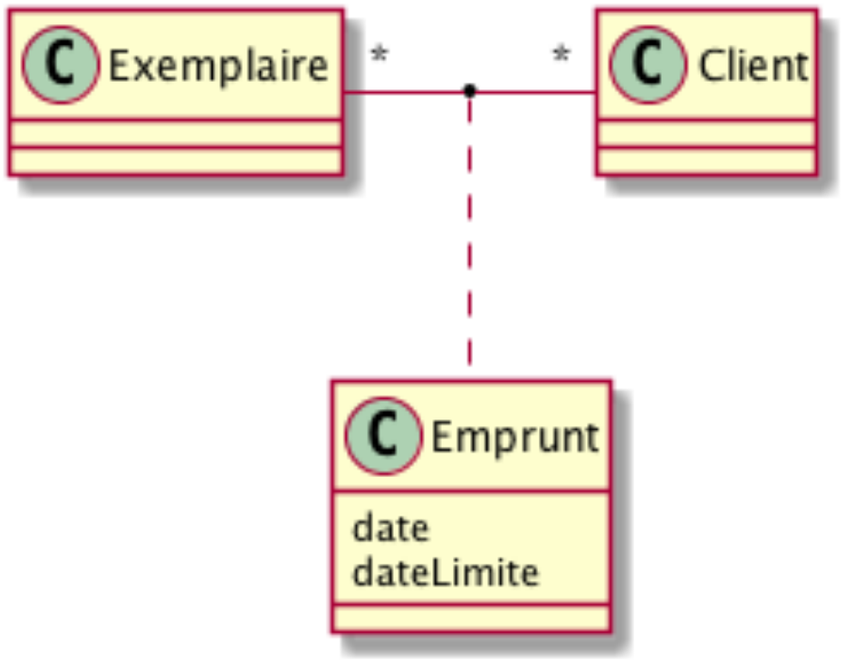
Classe d’association - 2

- L’**association** possède ses propres **attributs**
- On parle alors de **classe d’association**

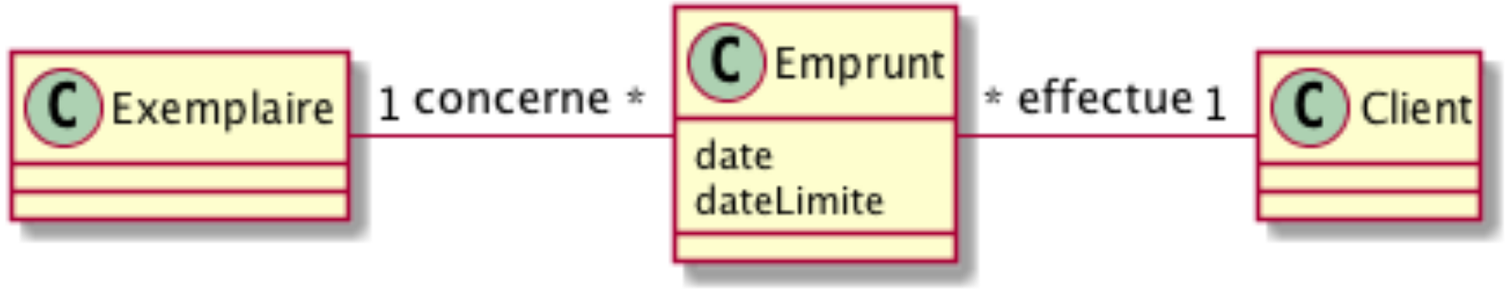


Classe d’association - 3

- Pour la représenter, on **créé** une **classe correspondante**



est équivalent à



Contraintes d’intégrité - 1

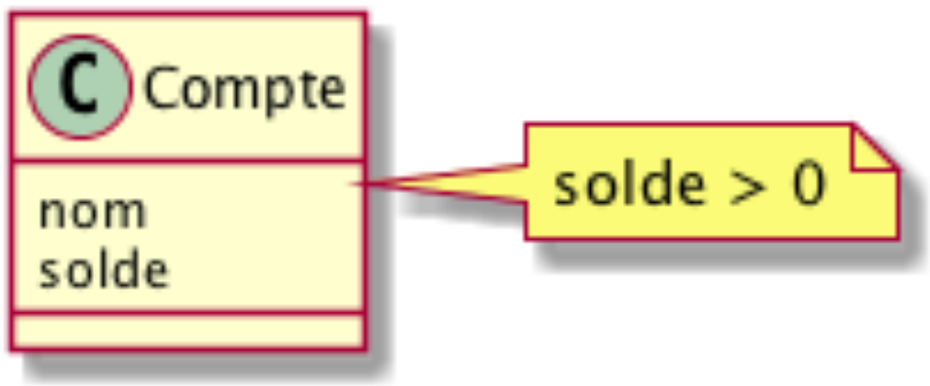
- Rappel : une CI est une **propriété de l’application à modéliser**

- Malheureusement, pas supportées nativement par *UML*

- Nous utiliserons des **notes** (*commentaires*) pour les représenter

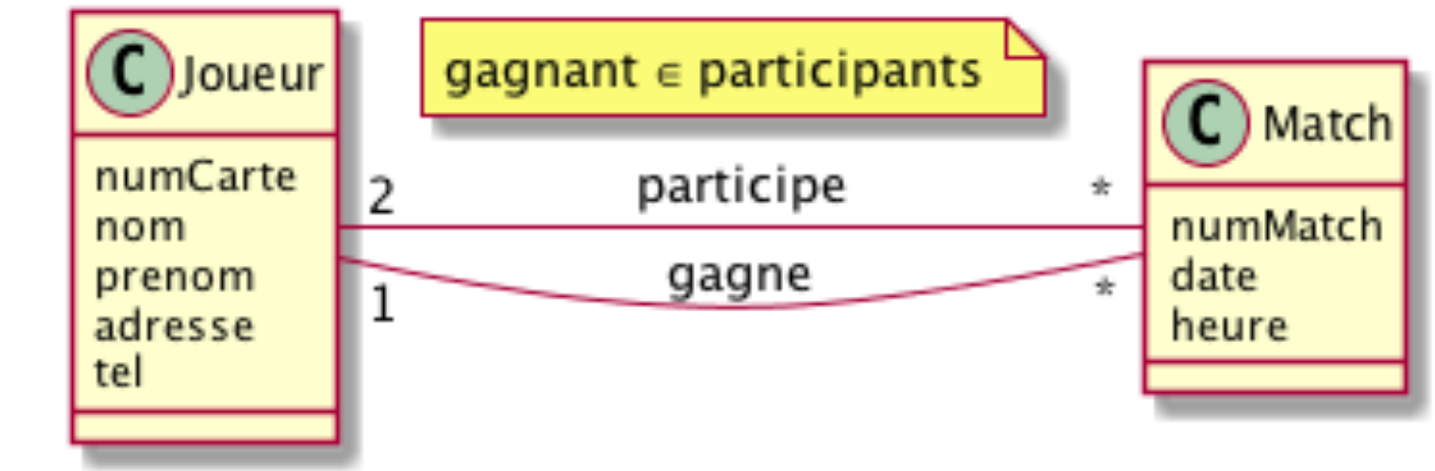
Contraintes d’intégrité - 2

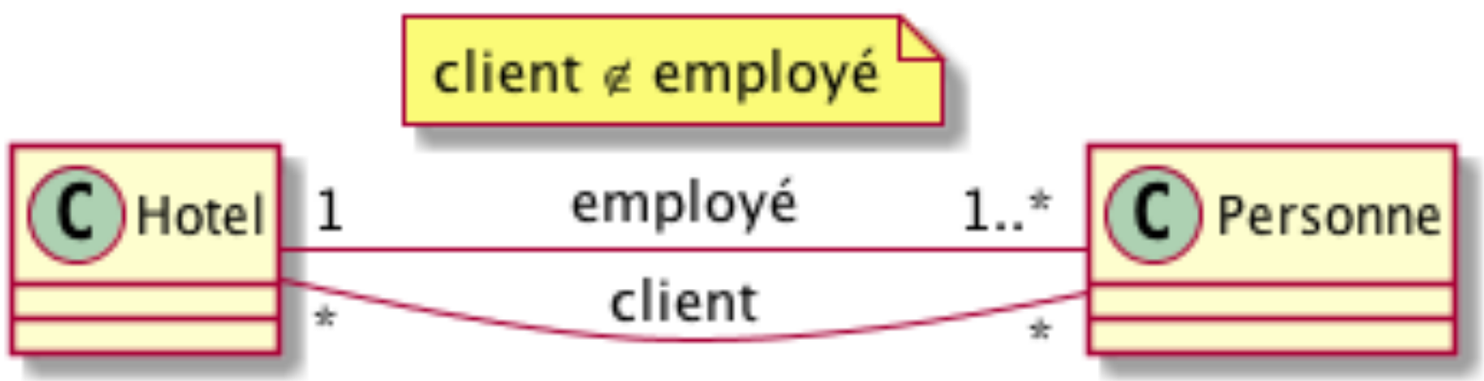
- Peut porter sur la **valeur** d’un attribut



Contraintes d’intégrité - 3

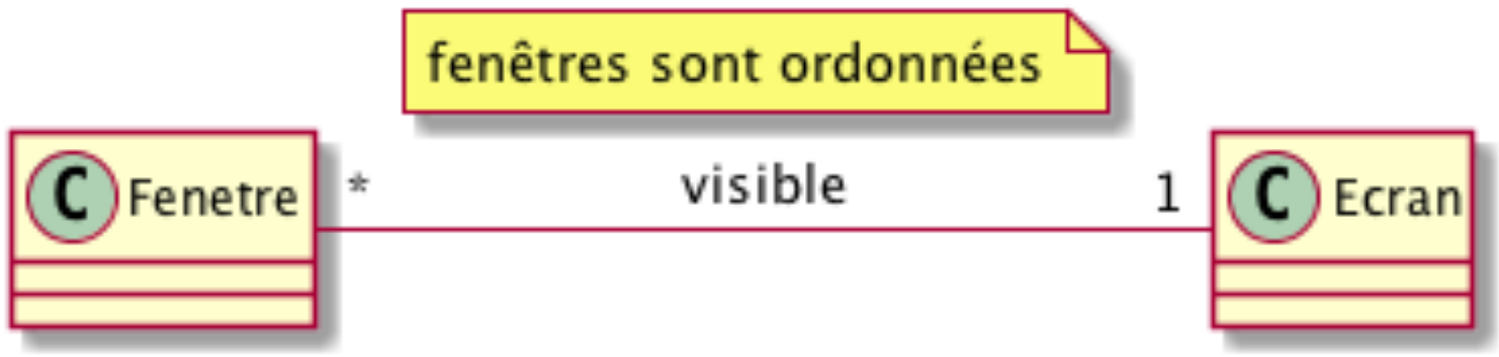
- Peut aussi avoir des contraintes **ensemblistes**





Contraintes d’intégrité - 4

- Peut aussi avoir des contraintes d’**ordonnancement**



Réflexions sur classes, attributs et associations

Base de Données 1
#2 - Compléments pour conception

Attribut ou classe ? - 1

- Un attribut est de type **atomique** (*une seule valeur*)

- Une classe est **composée** (*plusieurs valeurs, plusieurs attributs*)

Attribut ou classe ? - 2

- Considérons **l'adresse** d'un client : attribut ou classe ?
  - L'adresse est considérée comme une chaîne de caractères : **attribut**
  - L'adresse distingue un numéro, un nom de rue, un code postal, une ville : c'est une **classe**

Attribut ou association ?

- Conceptuellement, si la propriété doit représenter un **objet** ou un **ensemble d'objets**, alors une **association**
- Mais dans une solution de *mise en oeuvre avec un langage de programmation objet*, il n'y a **plus** de notions **d'association**…
  - Les **associations** sont mises en oeuvre comme des **attributs**

Classe ou association ?

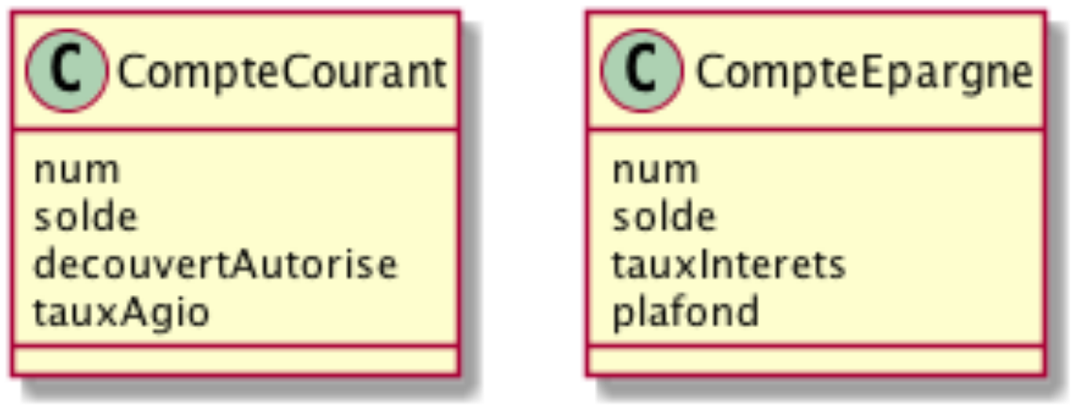
- En général, un **choix** conceptuel
- Encore plus lorsqu'une association a des attributs (**classe d'association**)
- **Spoiler** : dans une solution de mise en oeuvre avec une BD relationnelle, **classes** et **associations** sont représentées par le **même concept**
  - On parle de **relations** (ou tables)

Notions avancées

Base de Données 1
#2 - Compléments pour conception

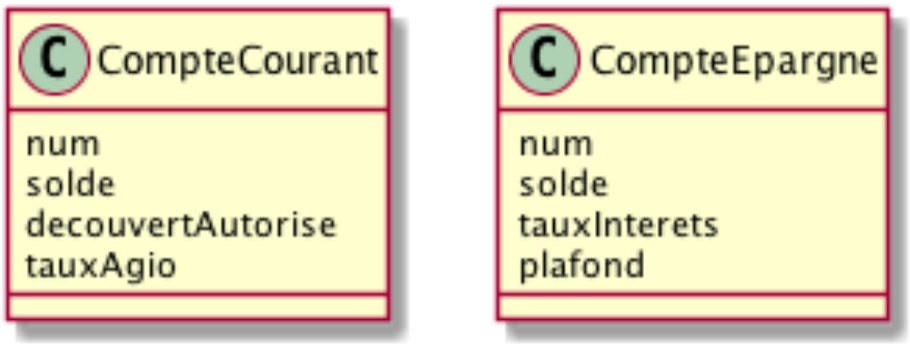
Hiérarchie de classes - 1

- Dans certains cas, peut avoir des classes avec des **propriétés communes**

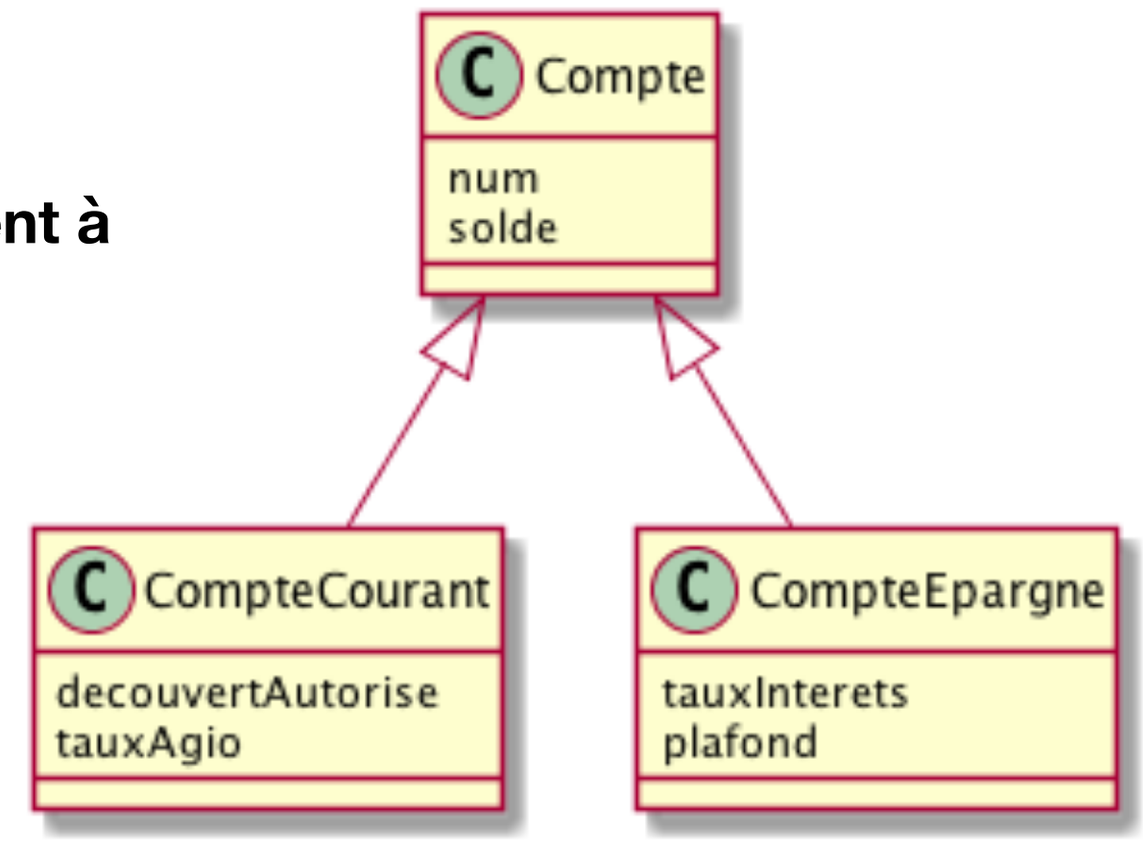


Hiérarchie de classes - 2

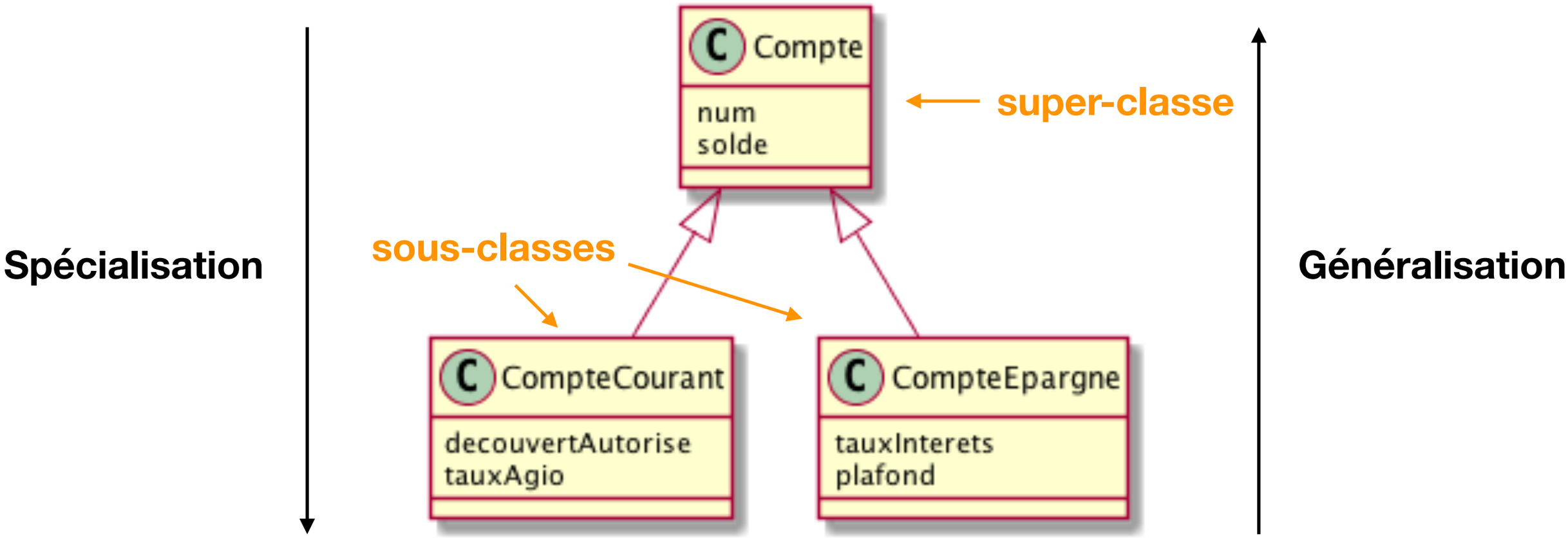
- Peut utiliser la relation d'**héritage** pour souligner ces caractéristiques communes



équivalent à



Hiérarchie de classes - 3



- **Spécialisation** : raffinement d'une classe en plusieurs sous-classes en distinguant les propriétés propres à chacune

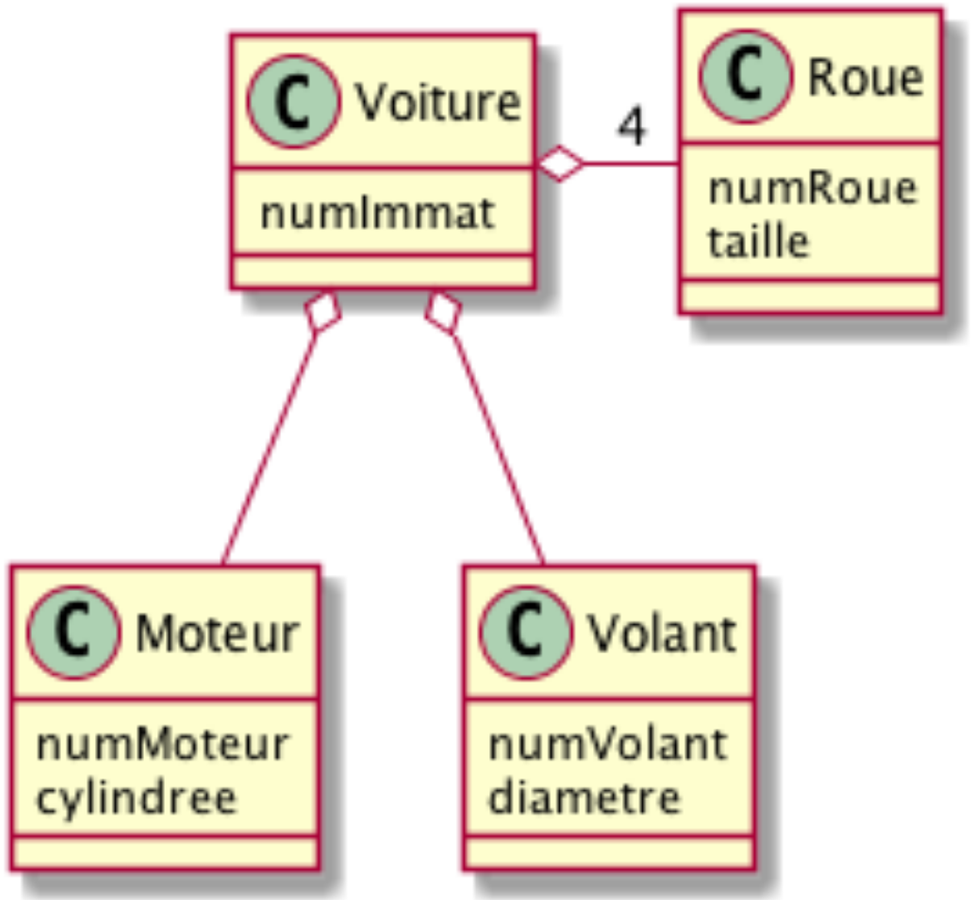
- **Généralisation** : regroupement des propriétés communes à plusieurs classes dans une super-classe

Agrégation et Composition

- **Types spéciaux d'associations** qui décrivent comment des composants sont combinées pour créer un tout : le composé
- Donnent aussi une information sur le **lien entre le cycle de vie** du composé et celui de ses composants

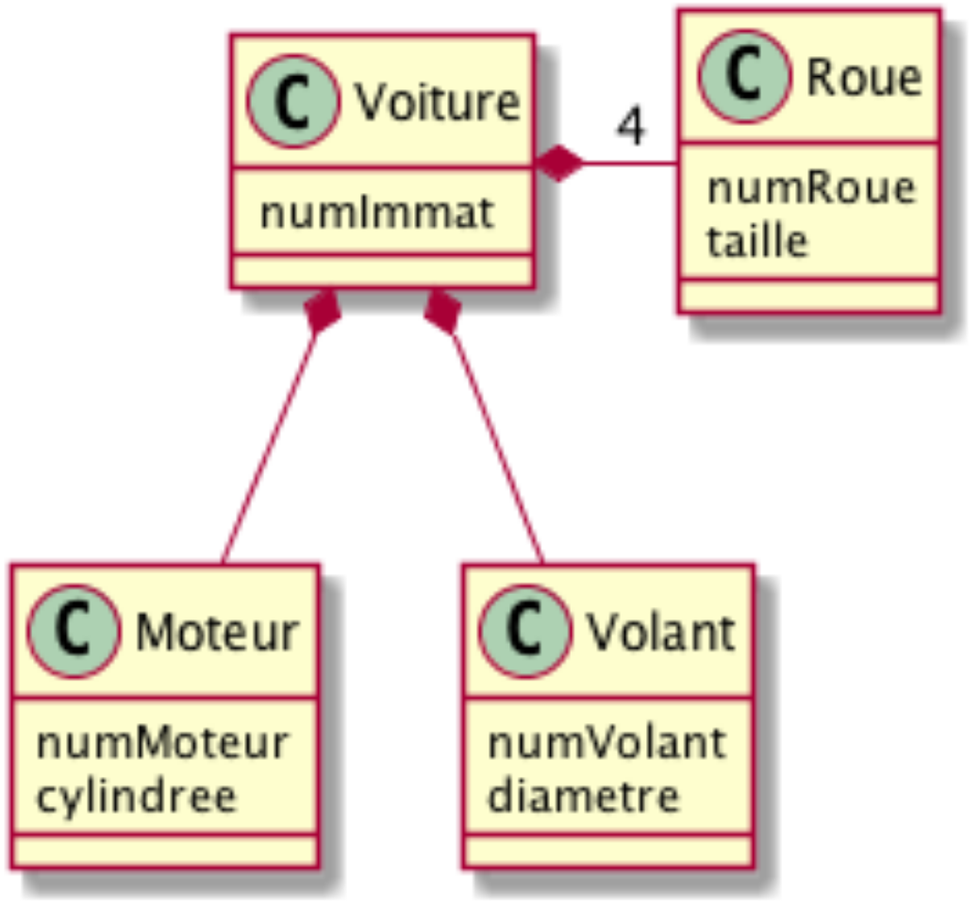
Agrégation

- La vie du composant **n'est pas liée** à celle du composé
- En d'autres termes : si on **détruit** l'objet **composé**, ses **composants survivent** individuellement



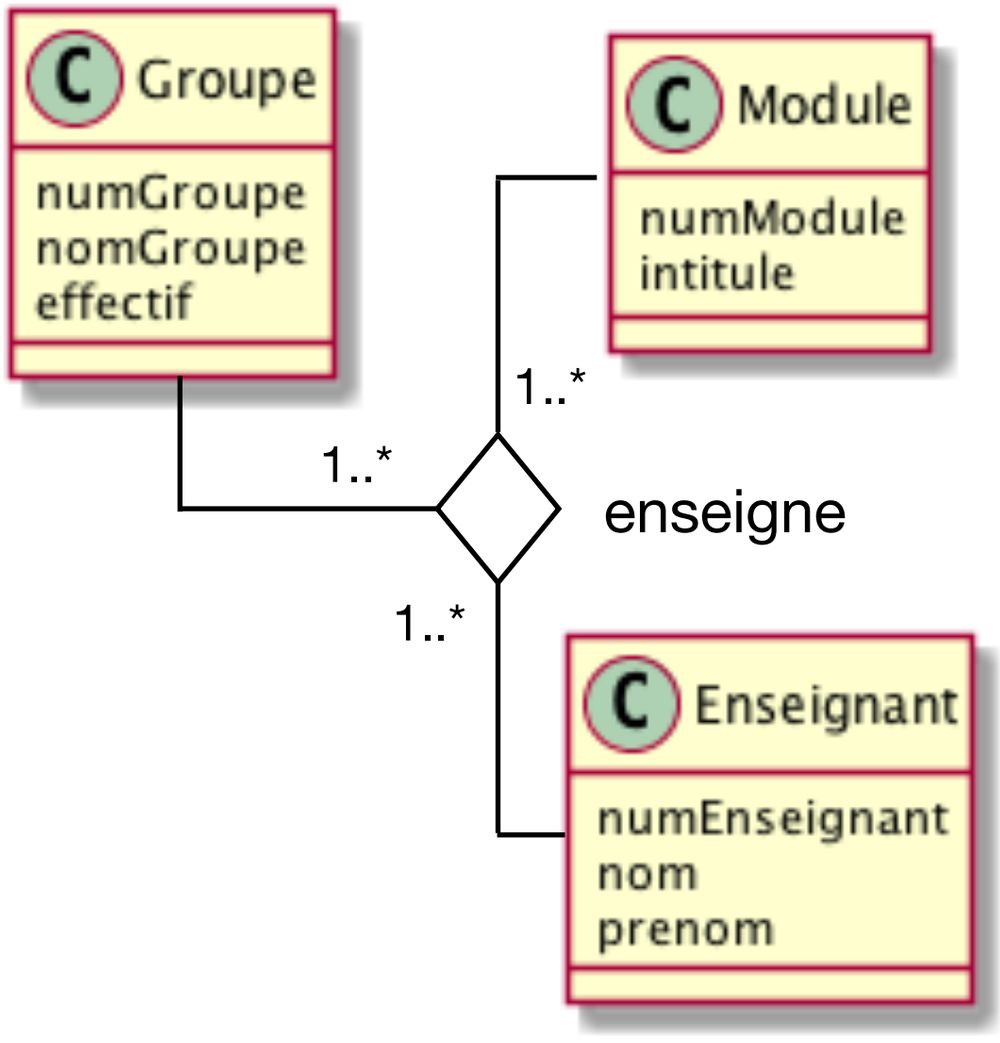
Composition

- La vie du composant **est liée** à celle du composé
- En d'autres termes : si on **détruit** l'objet **composé**, on **détruit ses composants**



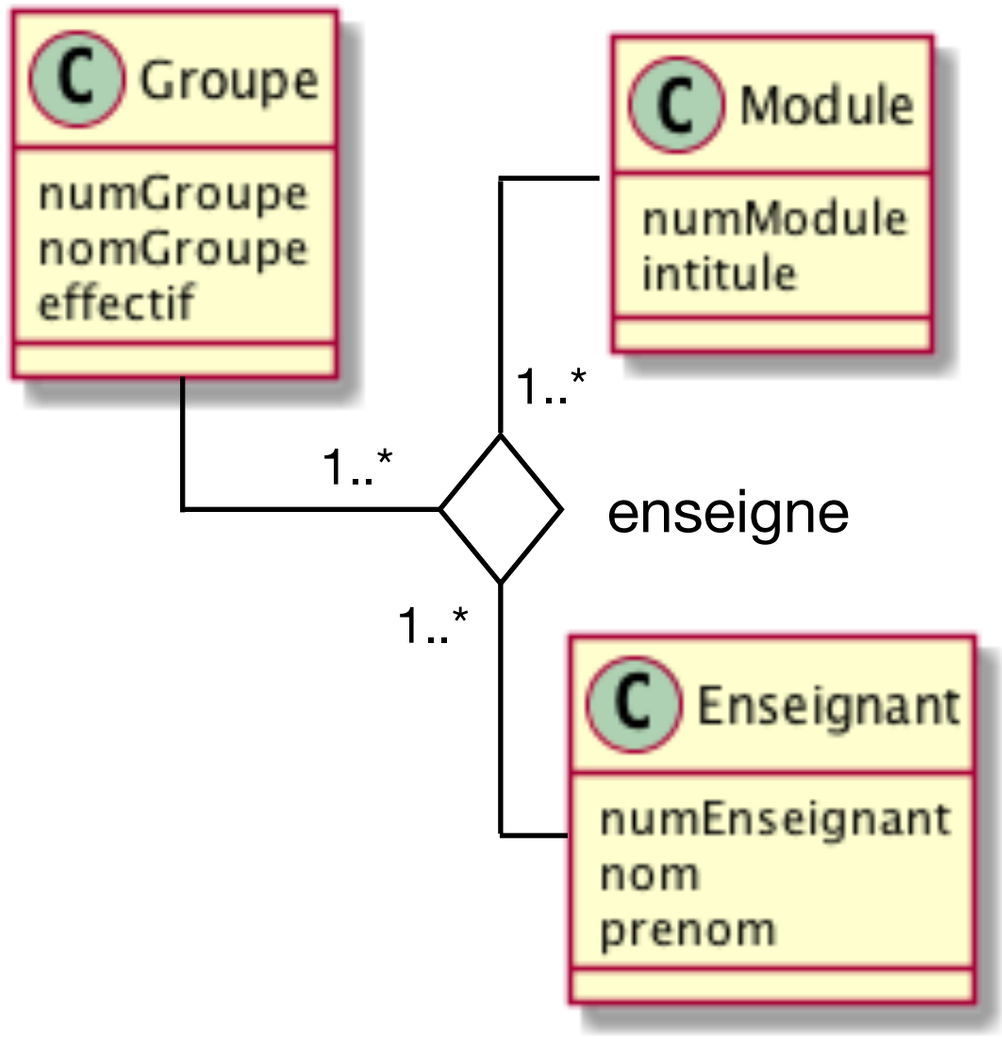
Associations n-aires - 1

- Peut rencontrer des associations reliant plus que 2 classes



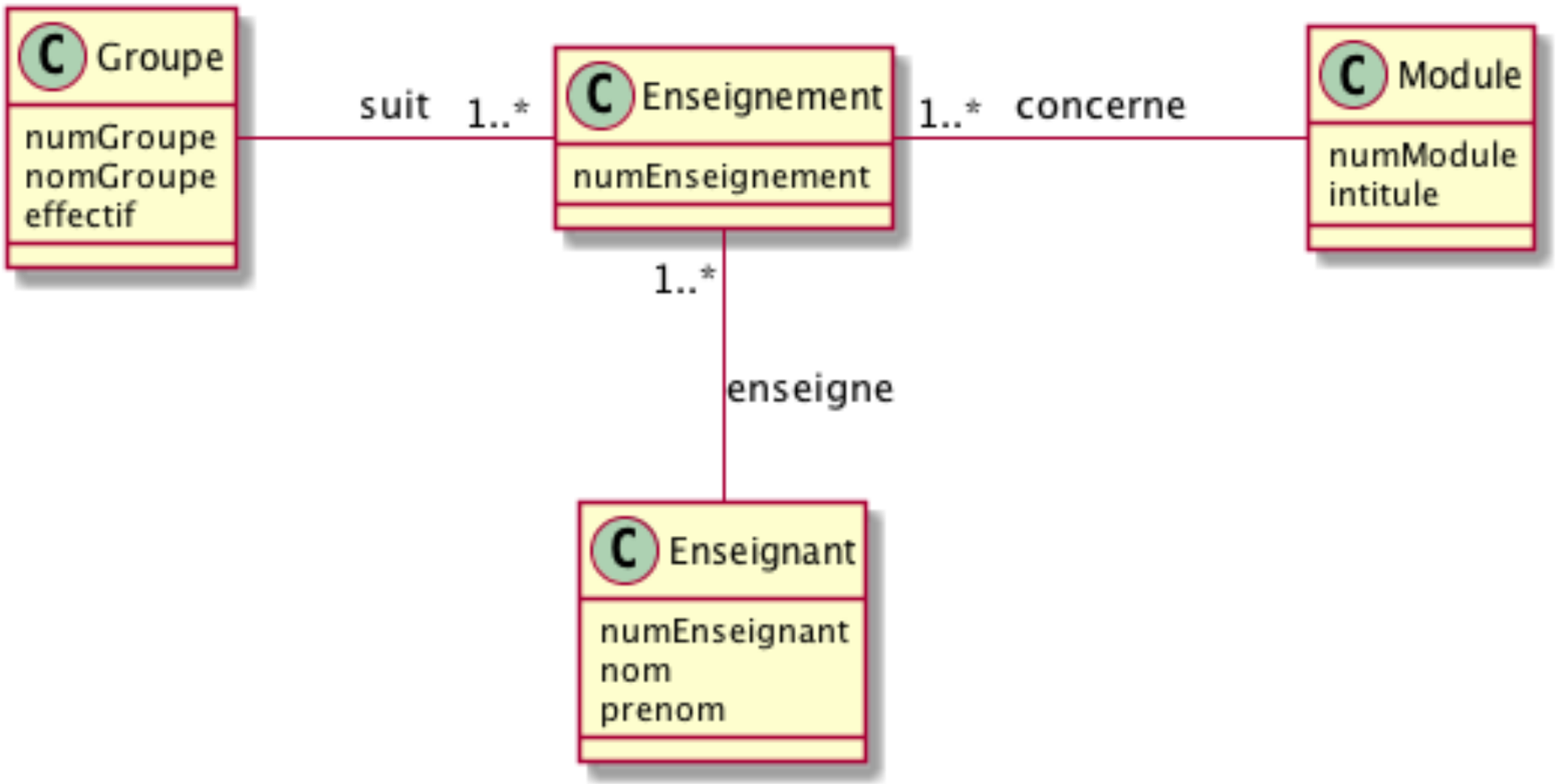
Associations n-aires - 2

- L’association *enseigne* dépend bien de ces 3 classes
- Un **enseignant** peut enseigner plusieurs **modules** à des **groupes** différents
- Un **groupe** peut suivre plusieurs **modules** par des **intervenants** différents
- Un **module** peut être enseigné à plusieurs **groupes** par différents **intervenants**



Associations n-aires - 3

- L'association *enseigne* correspond à la classe **enseignement**



Associations n-aires - 4

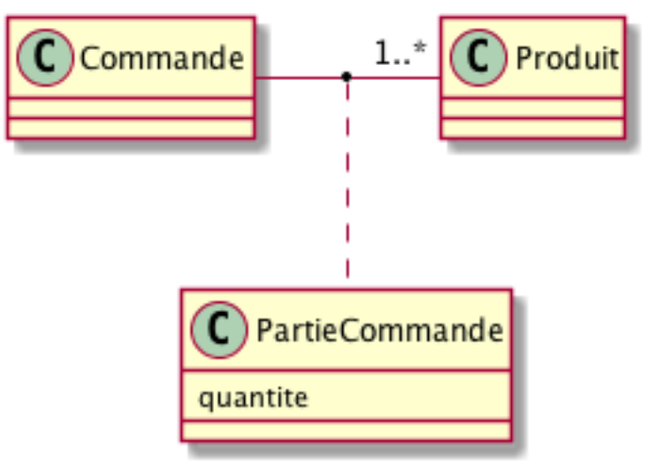
- En général, on ne va pas au-delà de 3 classes
- Une association reliant **4+ entités** indique peut-être une **erreur de conception**

Patrons de conception

Base de Données 1
#2 - Compléments pour conception

Exemple d’une commande

- Reprenons l’exemple de la commande de plusieurs produits (TD1, ex3)



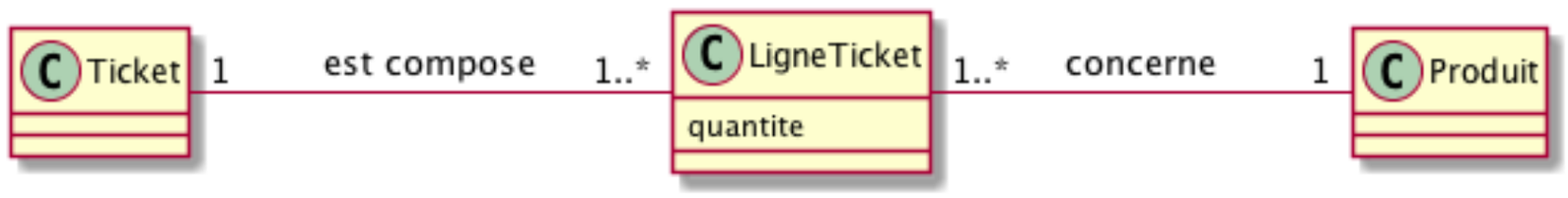
est équivalent à

Pourquoi introduire la notion de partie de commande ?

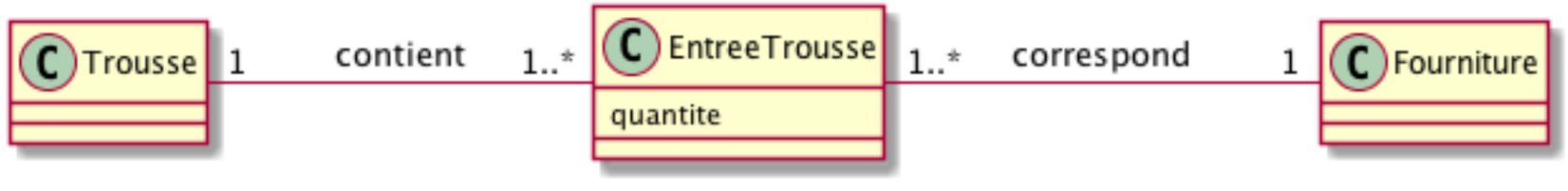
- Nous permet de gérer l'attribut *quantité*
  - *quantité* n'est pas un attribut de **Commande**, sinon cela voudrait dire que tous les **Produits** d'une **Commande** seraient commandés en même quantité
  - *quantité* n'est pas un attribut de **Produit**, sinon cela voudrait dire que dans toutes les **Commandes**, un **Produit** donné serait commandé en même quantité
  - *quantité* dépend de l'association d'une **Commande** et d'un **Produit**
- Correspond bien au modèle de commande en général

Une première notion de patron

- En fait, ce modèle de commande est une abstraction qui peut se spécialiser dans plusieurs cas particuliers
- Exemple d’un ticket de caisse



- Exemple de la composition d’une trousse



Notion de patron de conception

- En anglais : **design pattern**
- Provient de la **conception orientée objet**
- S’agit de la **meilleure solution connue à un problème** de conception récurrent
    - Peut s’agir d’un problème de **création** des objets, d’**architecture** de l’application ou de **comportement** des objets
- Décrit une structure de classes, associations, rôles et responsabilités

Liste de patrons

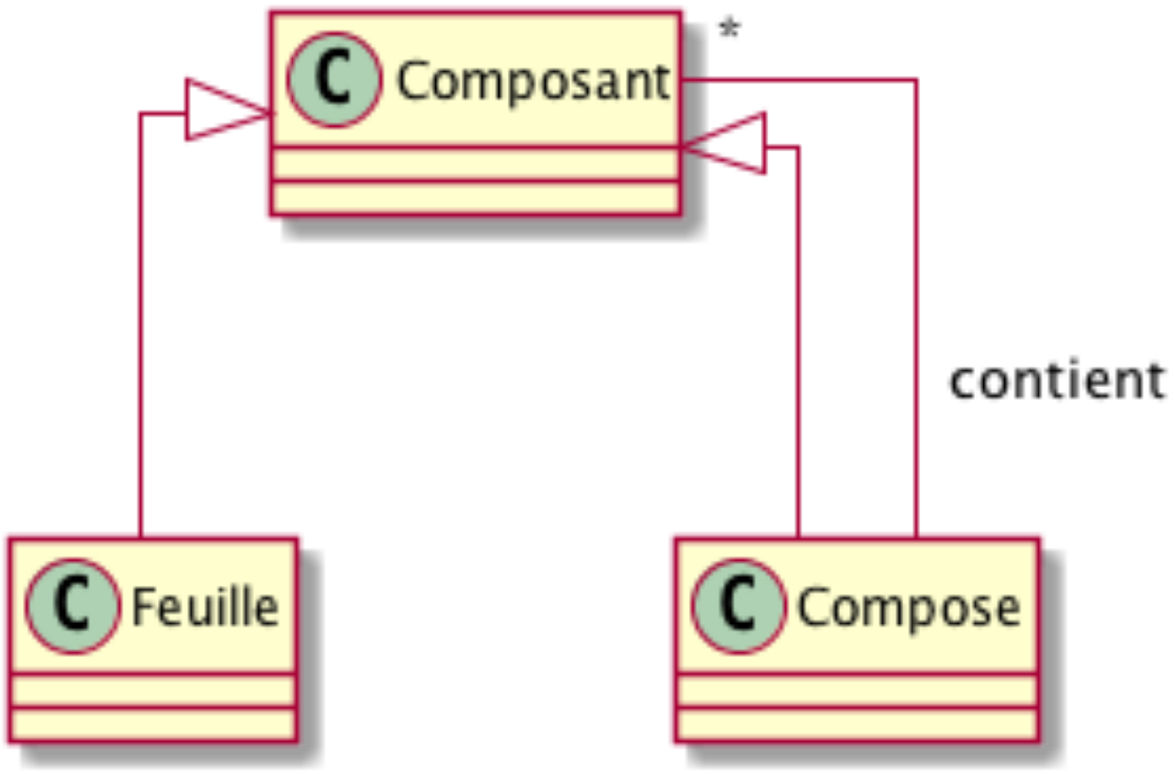
- Existe des dizaines de patrons
- Pour citer les plus connus
  - **Fabrique** : permet d'instancier des objets d'une classe abstraite sans connaître leur classe concrète
  - **Adaptateur** : permet de convertir l'interface d'une classe en l'interface d'une autre classe pour les interconnecter
  - **Modèle-vue-contrôleur** : architecture visant à séparer et découpler les données (*modèle*) de leur représentation (*vue*) et des moyens d'actions de l'utilisateur (*contrôleur*)

Patrons de conception en BD

- Moins documentés/référencés qu'en conception objet
- À nous de nous faire notre propre portfolios de patrons
- Les patrons issus de la conception objet **décrivant une structure de données** peuvent nous être utiles

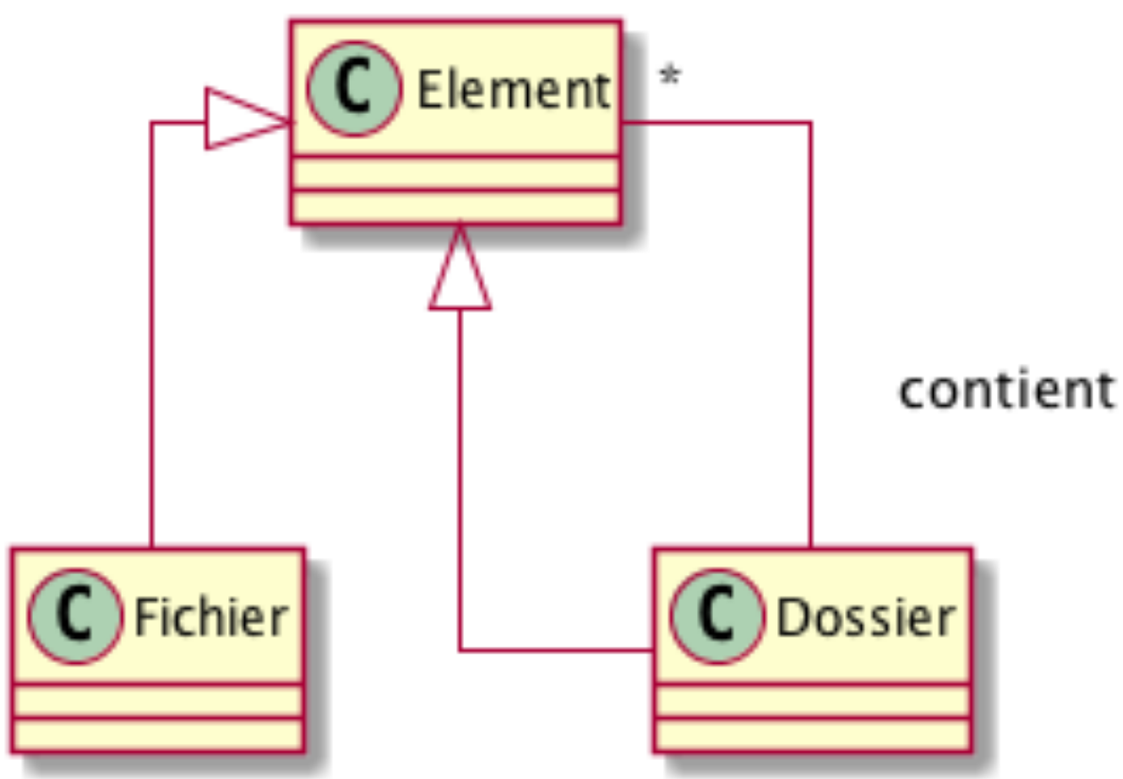
Exemple de Composite - 1

- Permet de décrire une arborescence d'objets et de manipuler de manière uniforme un élément unique, une branche ou l'ensemble de l'arbre



Exemple de Composite - 2

- Peut être réutilisé pour représenter un système de fichiers



Résumé

- Les **associations possédant des attributs** sont représentées sous la forme d'**entités**
- On gère de la même manière les **associations n-aires**
- Des **patrons de conception** permettent de résoudre les problèmes de conception courants
    - Encore faut-il les trouver, les connaître et les reconnaître…