Diagramme de classes et d'objets

Dr Idrissa Sarr
Idrissa.sarr@ucad.edu.sn

Objectifs du cours

- Identifier les concepts du système et les modéliser en tant que classes
- Identifier les associations pertinentes entre les concepts
- Définir les multiplicités de chaque extrémité d'association
- Utiliser les classes d'association, contraintes et qualificatifs
- Structurer le système en *packages*

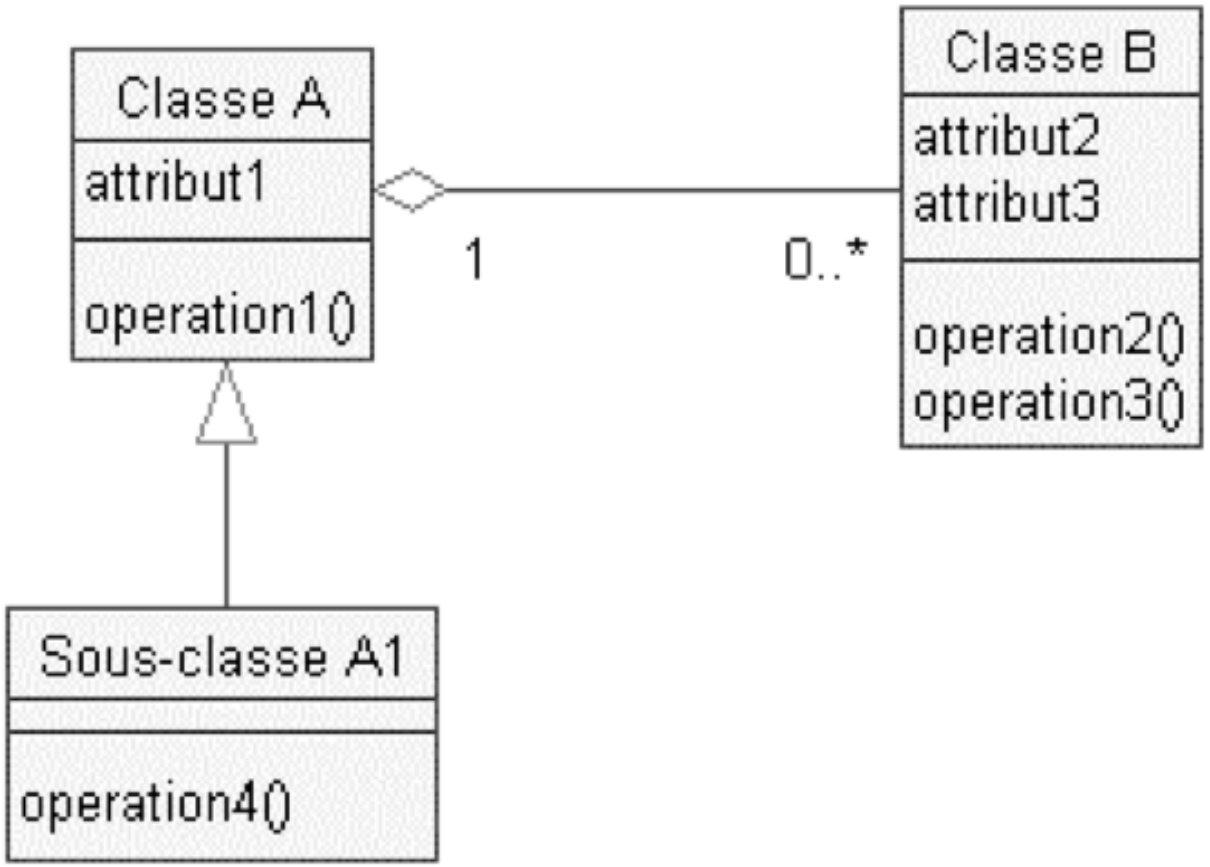
Diagramme de classes

- diagramme le plus important dans toutes les méthodes orientées objet.
  - celui que les outils de génération automatique de code utilisent en priorité
  - contient la plus grande gamme de notations et de variantes

- **En analyse**
  - décrit la structure des entités manipulées par les utilisateurs

- **En conception**
  - représente la structure d'un code orienté objet ou, à un niveau de détail plus important, les modules du langage de développement

Comment les représenter?



Classe et Objet

- **Classe**
  - représente la description abstraite d'un ensemble d'objets possédant les mêmes caractéristiques
  - Exemple : Voiture et Personne

- **Objet**
  - une entité aux frontières bien définies, possédant une identité et encapsulant un état et un comportement
  - Un objet est une instance (ou occurrence) d'une classe
  - Exemple : Ford est une instance d'une classe personne, « Ndoye Seulement » est une instance de la classe Personne

Objets et classes

Objet : une **entité concrète** avec une identité bien définie qui encapsule un **état** et un **comportement**.
L'état est représenté par des valeurs d'attribut et des associations, le comportement par des méthodes.

Un objet est une instance d'une classe.

Classe : une description d'un **ensemble d'objets** qui partagent les mêmes attributs, opérations, méthodes, relations et contraintes.

Une classe peut posséder des attributs ou des méthodes « de classe ».

MaVoiture : Voiture
marque = Renault Modèle = Nevada Immatriculation = 648ADX38 AnnéeModele = 1992 Kilométrage = 285 000

Voiture
marque : chaîne Modèle : chaîne Immatriculation : chaîne (8) AnnéeModele : date Age_moyen : entier
Rouler ( ) Kilometrage_annuel_moyen ( )

Identifier les objets/classes

Attribut et Opération

- **Un *attribut*** est une propriété commune à tous les objets d'une classe.
  - représente un type d'information contenu dans une classe
  - vitesse courante, cylindrée, numéro d'immatriculation, etc. sont des attributs de la classe Voiture.

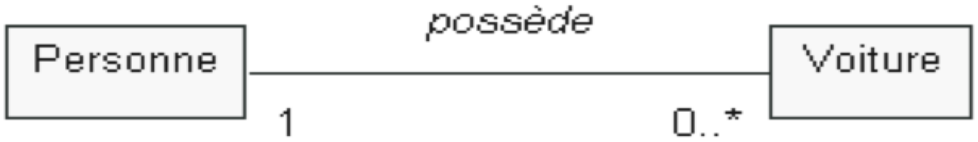
- **Une *opération***
  - représente un élément de comportement (un service) contenu dans une classe.

Associations entre classes

- Une association représente une relation sémantique entre les objets d'une classe.

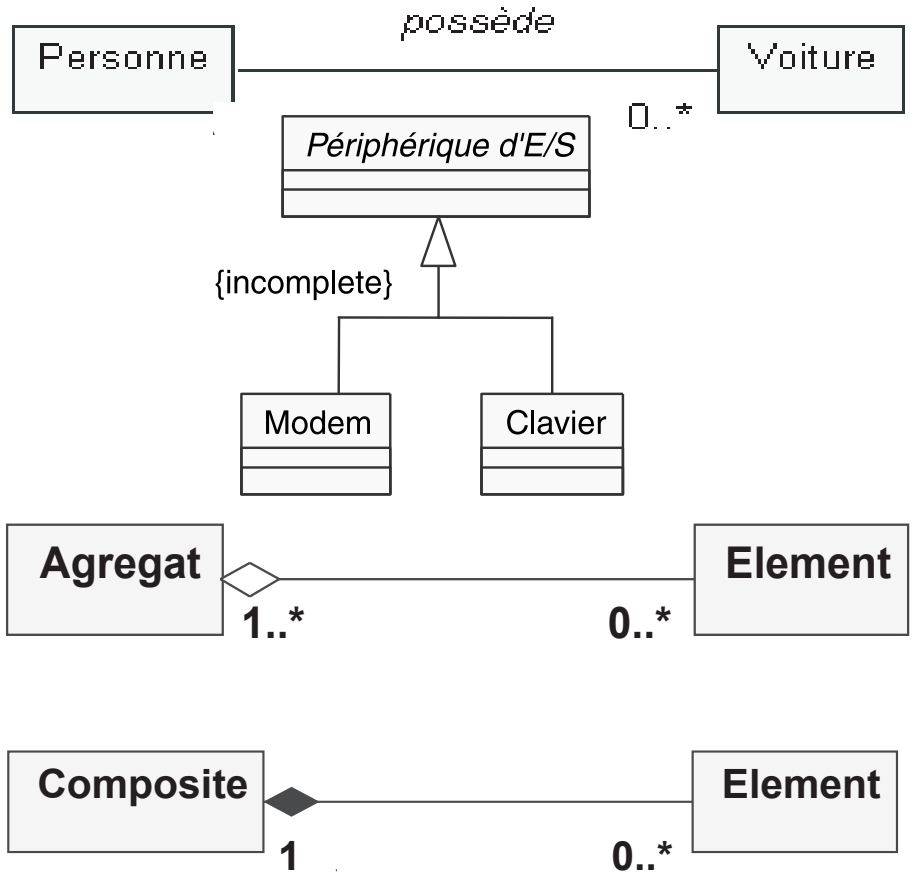
- Exemple :
  - une **personne** peut **posséder** des **voitures**.
  - La relation ***possède*** est une association entre les classes Personne et Voiture.

- Une association est représentée par un trait plein entre les classes associées.
  - Elle est complétée par un nom
  - Avec une précision du sens de lecture en cas d'ambigüité
  - Chaque extrémité de la relation indique le rôle de la classe dans l'association et précise le nombre d'objets de la classes qui y interviennent (multiplicité)



Types d'associations

- **L'association simple:** lien entre instances de classes.
- **La généralisation/spécialisation:** factorisation des propriétés communes à plusieurs classes.
- **L'agrégation:** lien de type ensemble/élément
- **La composition:** cas particulier de l'agrégation avec un couplage fort.

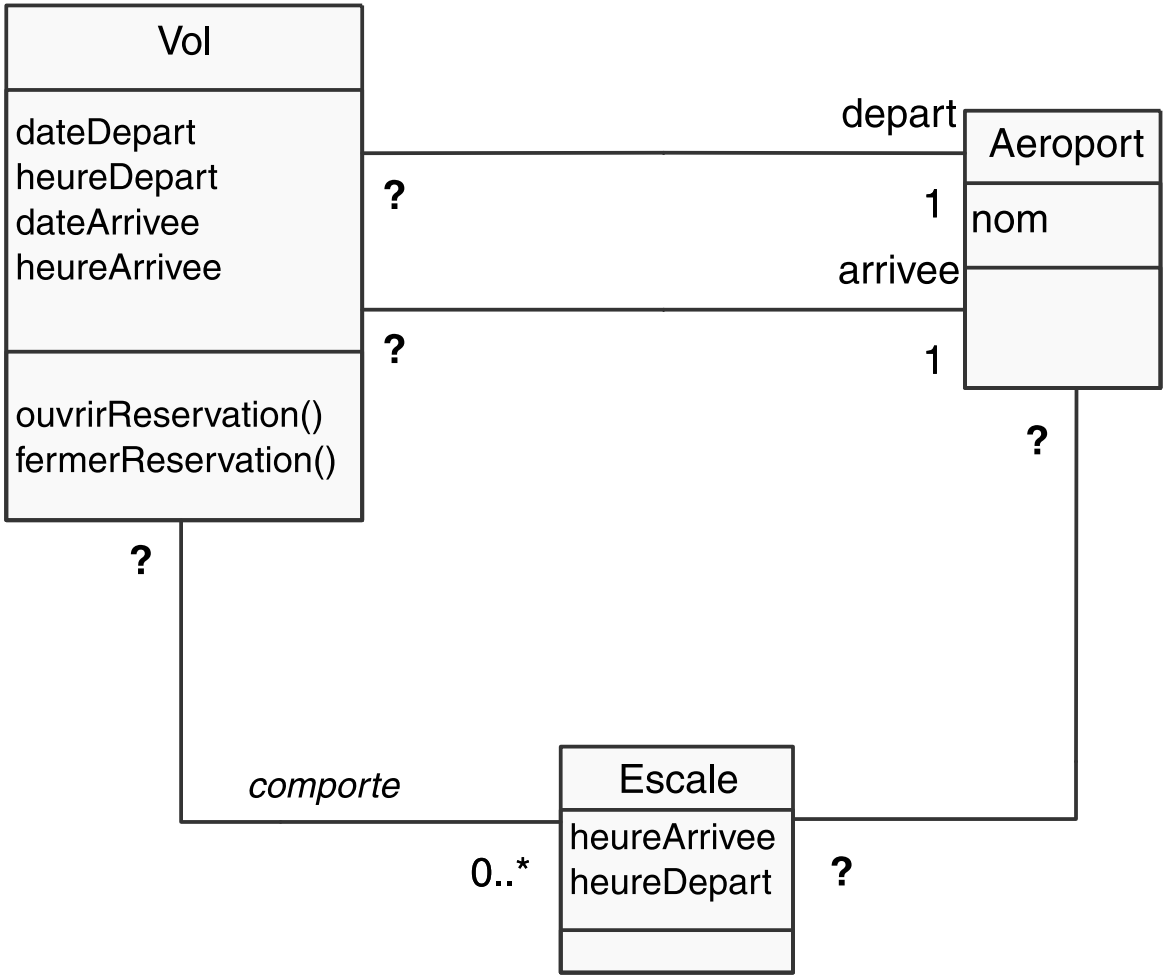


Relations entre classes

- **Multiplicité :**
  - un ensemble de valeurs indiquant le nombre possible d'instances de la classe destination du rôle qui peuvent être reliées à une instance de la classe origine du rôle.
  - peut être associée à une terminaison **d'association simple, d'agrégation** ou **de composition.**

1	Un et un seul
0..1	zéro ou un
n	n (entier naturel)
m..n	De m à n (entier naturel)
*	plusieurs
0..*	De zéro à plusieurs
1..*	De un à plusieurs

Essayez de trouver les multiplicités



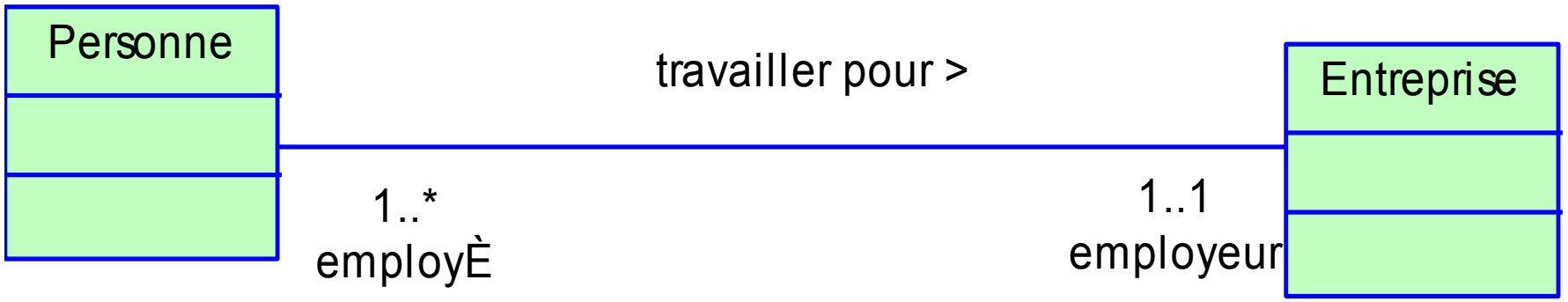
Cas d'études

- Cette étude de cas concerne un système simplifié de réservation de vols pour une agence de voyages.
- Les interviews des experts métier auxquelles on a procédé ont permis de résumer leur connaissance du domaine sous la forme des phrases suivantes :
- Des compagnies aériennes proposent différents vols.
- Un vol est ouvert à la réservation et refermé sur ordre de la compagnie.
- Un client peut réserver un ou plusieurs vols, pour des passagers différents.
- Une réservation concerne un seul vol et un seul passager.
- Une réservation peut être annulée ou confirmée.
- Un vol a un aéroport de départ et un aéroport d'arrivée.
- Un vol a un jour et une heure de départ, et un jour et une heure d'arrivée.
- Un vol peut comporter des escales dans des aéroports.
- Une escale a une heure d'arrivée et une heure de départ.
- Chaque aéroport dessert une ou plusieurs villes.

Documentation d'une association (1)

- Une association binaire est matérialisée par un trait plein entre les classes associées.
- Elle peut être ornée d'un **nom**, avec éventuellement une précision du sens de lecture > ou < .

- Le **rôle** tenu par une classe vis-à-vis d'une association peut être précisé (indispensable pour les associations réflexives).

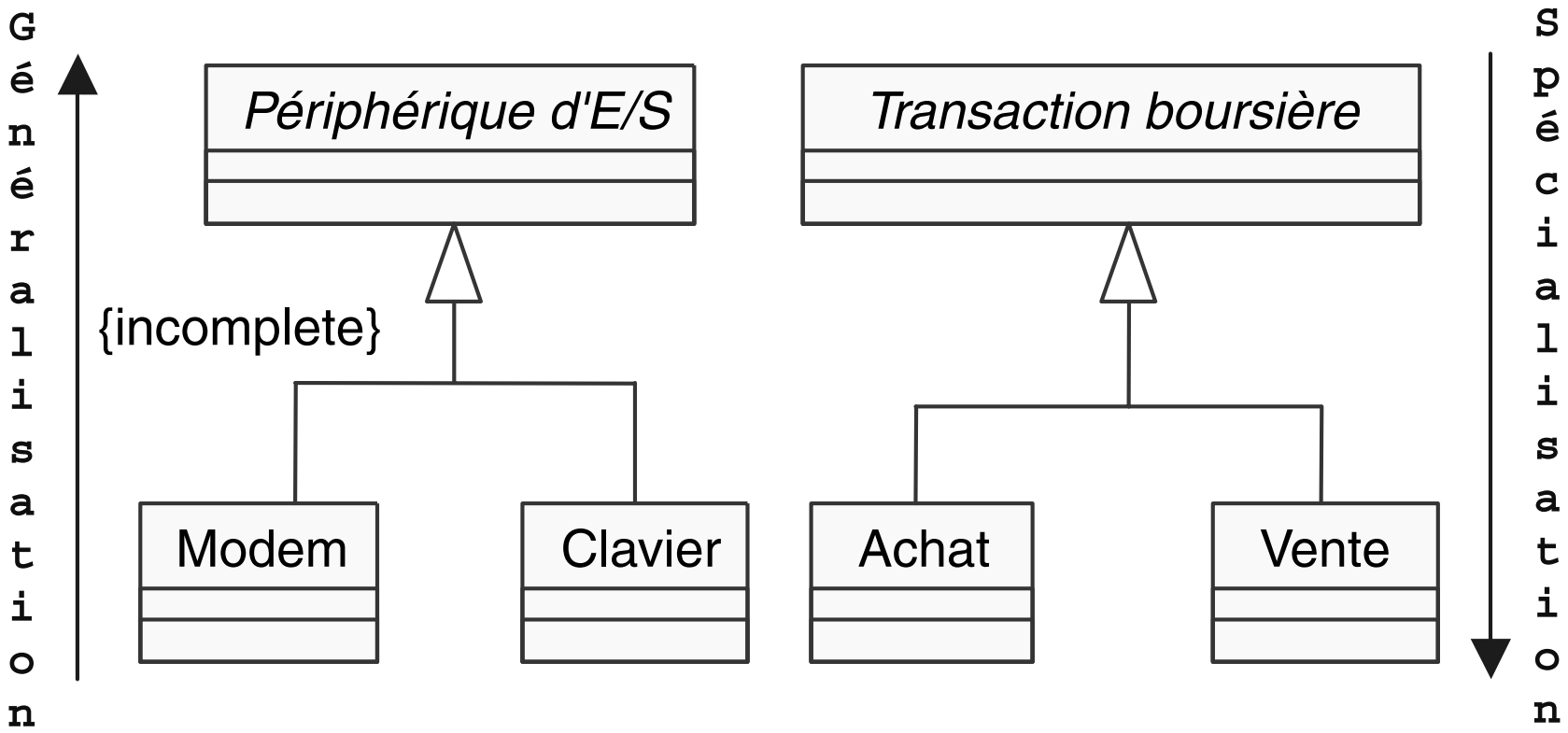


Généralisation/Spécialisation

- La **généralisation** est la relation qui existe entre une classe générale (super classe) et plusieurs autres classes (sous-classes) plus spécifiques.

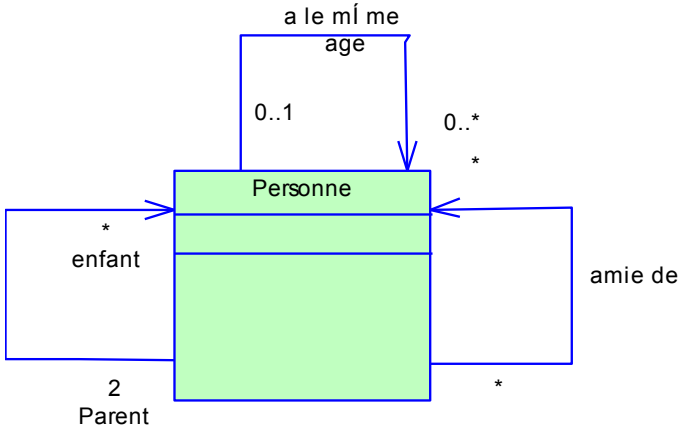
- Une sous-classe possède toutes les propriétés de la super classe.

Exemple Généralisation/ Spécialisation



Association réflexive

- Une association est dite réflexive quand les 2 extrémités pointent vers le même classeur.
- Elle a pour principale fonction de structurer les objets d'une même classe.
  - Association **asymétrique**: permet de hiérarchiser les objets. Il est conseillé d'ajouter les rôles des extrémités dans le cas des associations asymétriques.
  - Association **symétrique** et **transitive**: permet de regrouper les objets d'une même classe en classes d'équivalence.
  - Association **symétrique** et **non transitive**: partitionne les objets de la classe.

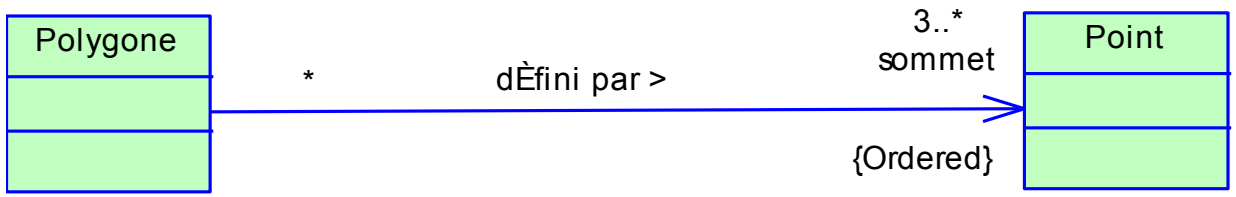


Contraintes

- Les contraintes permettent de rendre compte des détails que n'expriment pas les éléments du langage UML afin de restreindre la portée du modèle.
- Elles sont exprimées par OCL ou:
  - Une expression mathématique
  - Un langage de programmation
  - Le langage naturel
- Les contraintes les plus courantes expriment:
  - Les règles de l'héritage
    - {complete}, {incomplete}, {overlaps}, {distinct}, …
  - Restriction de la portée d'une association
    - {subset}, {XOR}, …
  - Mode d'évolution des objets
    - {frozen}, {addonly}, …
  - Organisation des objets
    - {odered}, {frozen}, …

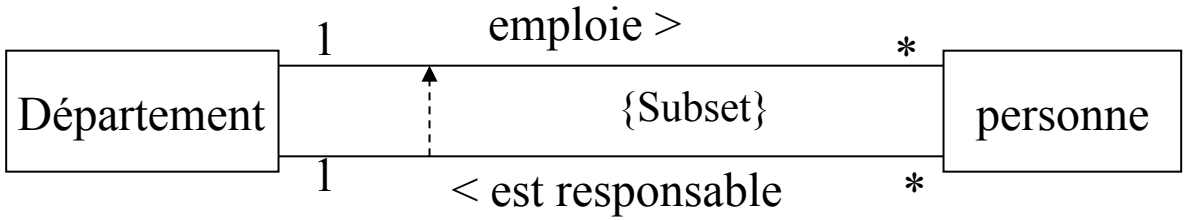
Association avec contraintes

- **Ordonnée**
  - C'est une contrainte qui spécifie que les objets sont ordonnés (selon un attribut: la clé, le nom, la date, etc.)
  - La manière dont les objets sont ordonnés peut être spécifiée à travers une note.



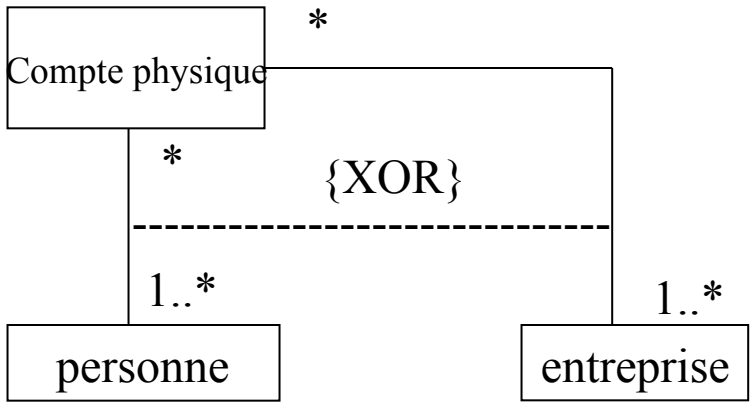
Association avec contraintes

- **Sous-ensemble**
  - C'est une contrainte qui indique qu'une collection est incluse dans une autre collection.
  - La contrainte est placée à proximité d'une relation de dépendance entre deux associations.
  - La flèche de la relation de dépendance indique le sens de la contrainte.



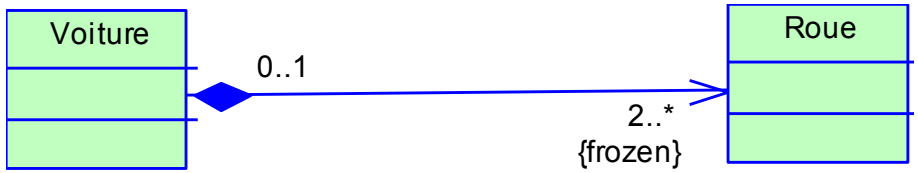
Association avec contraintes

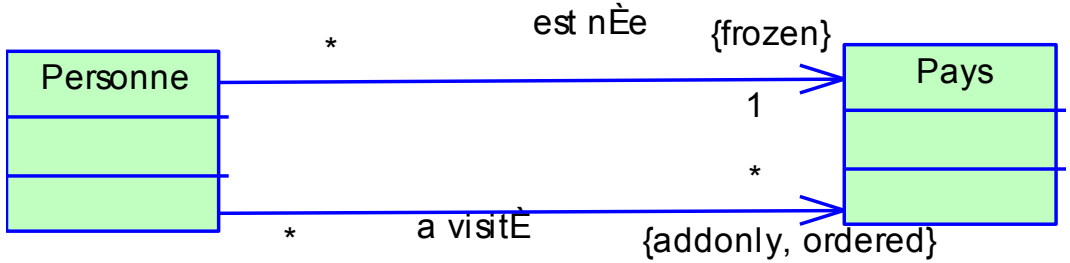
- **Ou exclusif (XOR)**
  - précise que, pour un objet donné, une seule association parmi un groupe d'associations est valide
  - Exemple :Un contrat d'assurance concerne une entreprise ou une personne mais pas les deux en même temps



Association avec contraintes

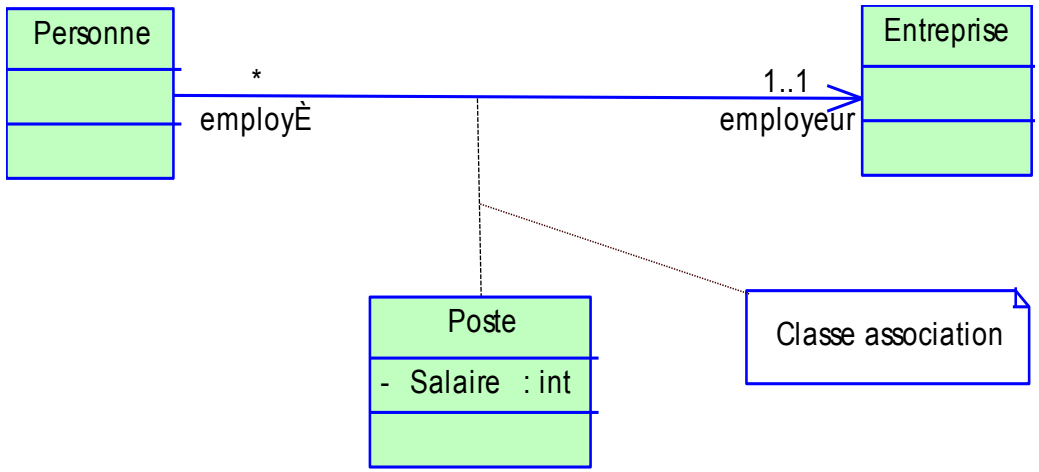
- Frozen et addonly
  - Frozen: spécifie que le nombre d'objets ne varient pas
  - Addonly: indique que le nombre d'objets ne peut que croître.





Classe-association

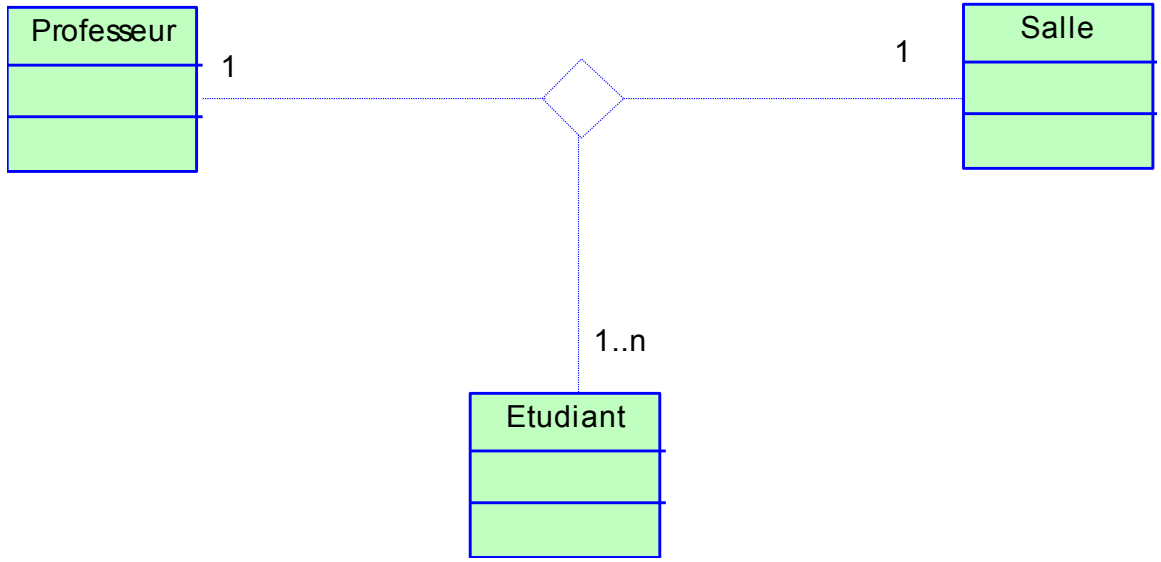
- Une association peut avoir ses propres propriétés.
- Ses propriétés ne sont disponibles dans aucune des classes qu'elle lie.
- possède à la fois les caractéristiques d'une association et celles d'une classe.
- se connecte à deux ou plusieurs classes et peut posséder des attributs et des opérations.
- Exemple: Les personnes qui travaillent dans une entreprise occupent un poste. Le salaire d'un employé est lié au poste.



Association n-aire

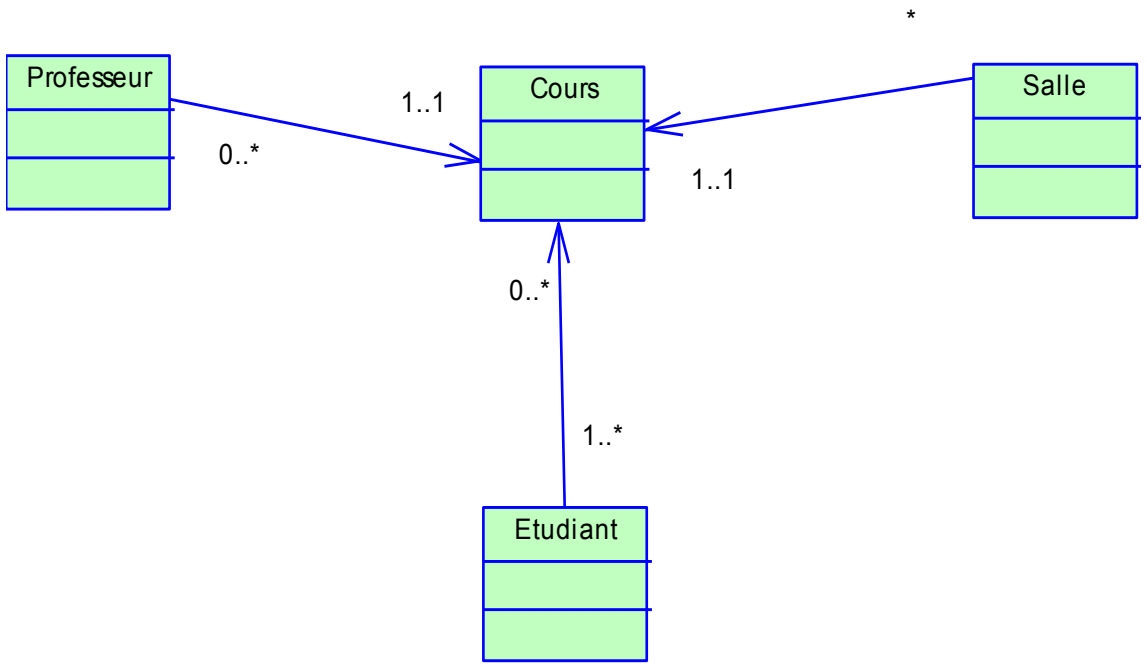
- Une association peut relier plus de deux classes; elle est dite dans ce cas n-aire
  - On représente une association n-aire par un grand losange avec un chemin partant vers chaque classe participante.
  - Le nom de l'association, le cas échéant, apparaît à proximité du losange.

Exemple d'association ternaire



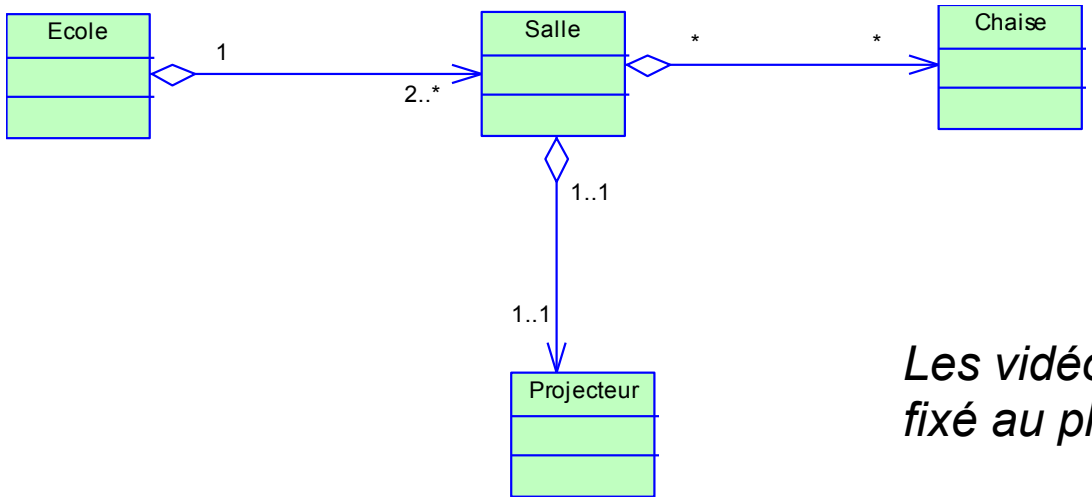
Classe d'association

- NB : les associations ternaires sont difficiles à déchiffrer et peuvent induire en erreur. Mieux vaut utiliser des associations binaires combinés avec des contraintes du langage OCL.



Agrégation

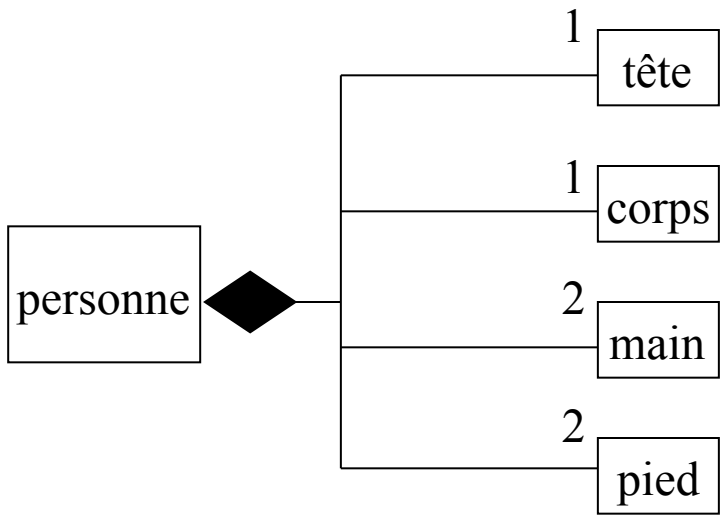
- C'est une **association asymétrique** dans laquelle **l'une** des **extrémités** joue un rôle **prédominant** par rapport à **l'autre.**
- C'est une relation « composé-composant ».
- Une agrégation peut notamment (mais pas nécessairement) exprimer :
  - qu'une classe (un "élément") fait partie d'une autre ("l'agrégat"),
  - qu'un changement d'état d'une classe, entraîne un changement d'état d'une autre,
  - qu'une action sur une classe, entraîne une action sur une autre.



Les vidéo-projecteur sont fixé au plafond

Composition

- C'est un cas particulier de l'agrégation avec une contrainte de durée de vie entre la classe composite et la ou les classes composées.
- La destruction de l'agrégat implique automatiquement la destruction de tous ses composants.
- Elle implique une contrainte sur la valeur de la multiplicité du côté de la classe composite qui doit être 0 ou 1.

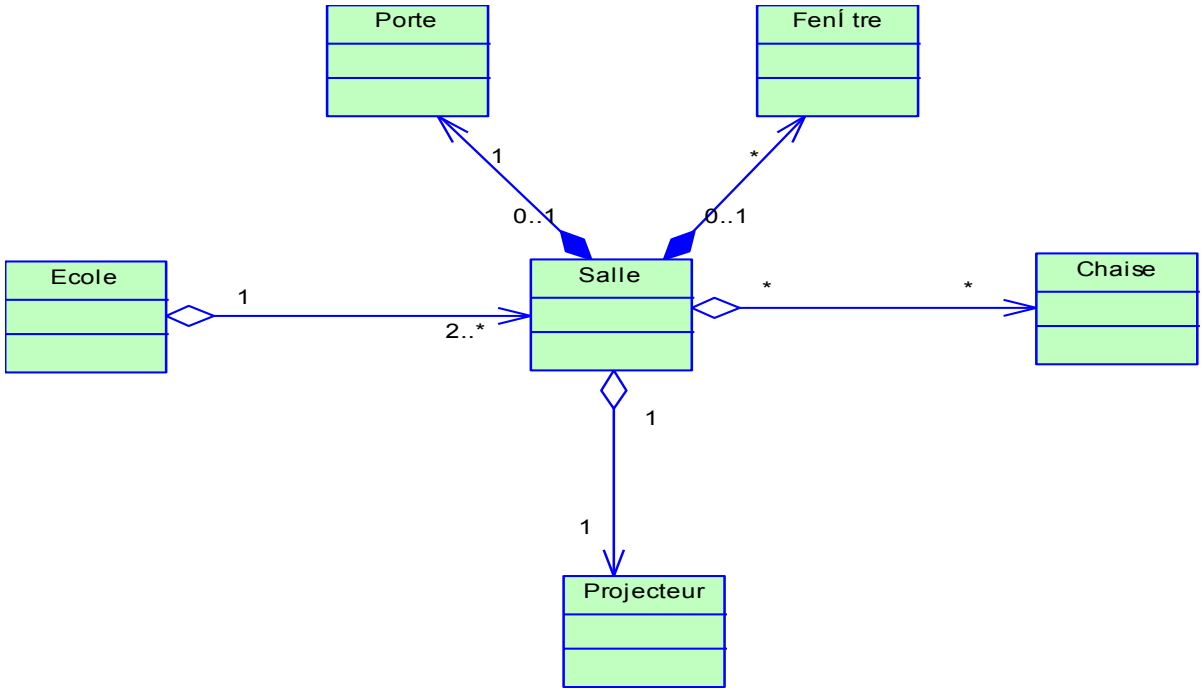


Règles obligatoires pour la composition

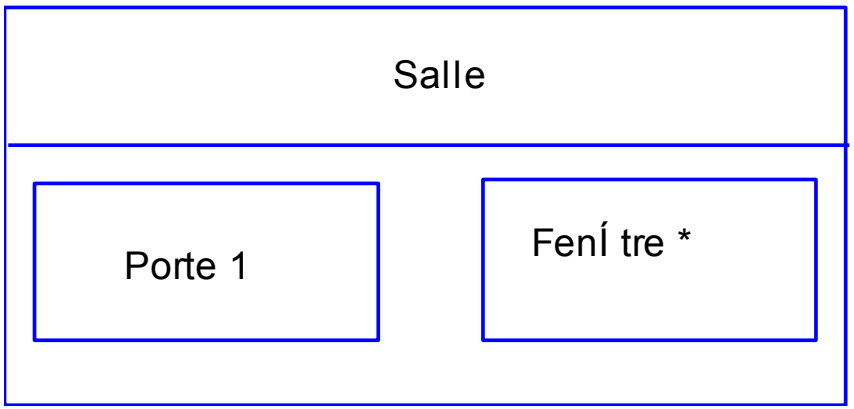
• La suppression du composite entraîne la suppression des composants
• Les attributs du composite sont utilisés dans les composants
• Un composant ne peut pas être en relation avec d'autres classes externes au composite.

Composition (suite)

On peut compléter l'exemple sur l'agrégation en indiquant qu'une salle est composée d'une porte et de plusieurs fenêtres

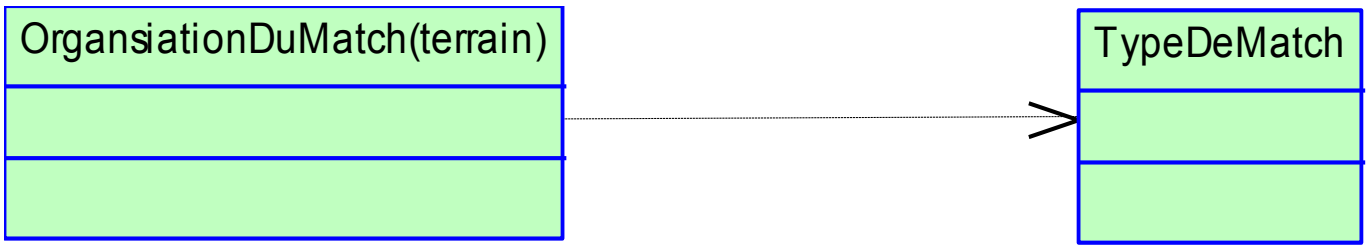


NB: La composition étant structurelle, on peut alors adopter la représentation ci-contre



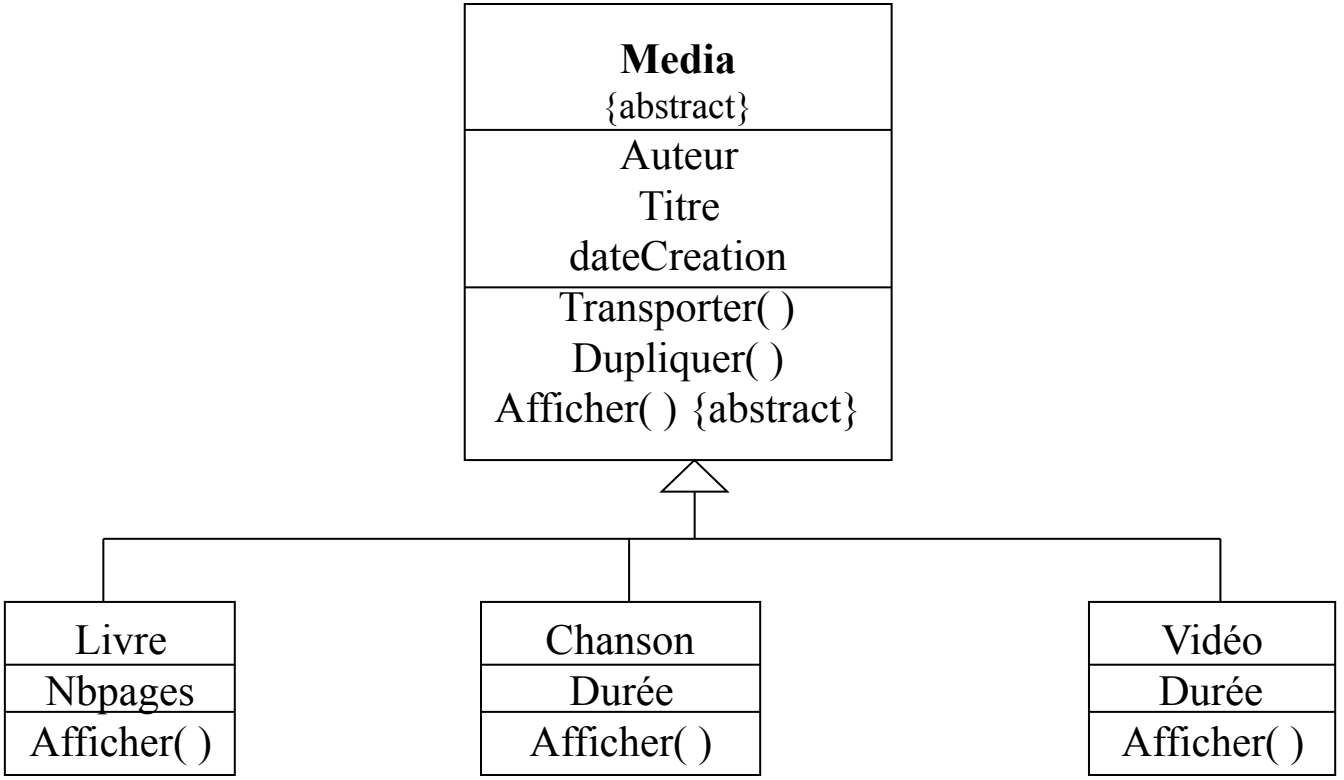
Relation de dépendance

- Relation unidirectionnelle exprimant une dépendance sémantique entre éléments du modèle.
  - Indique que la modification de la cible implique le changement de la source.
  - Représentée par un trait discontinue + souvent un stéréotype pour mieux expliciter le lien sémantique.
    - Stéréotypes:
      - «friend»: on accorde une visibilité spéciale à la classe source dans la classe cible
      - «dérive»: source calculée à partir de la cible
      - « appel », « send », « copie », « create »



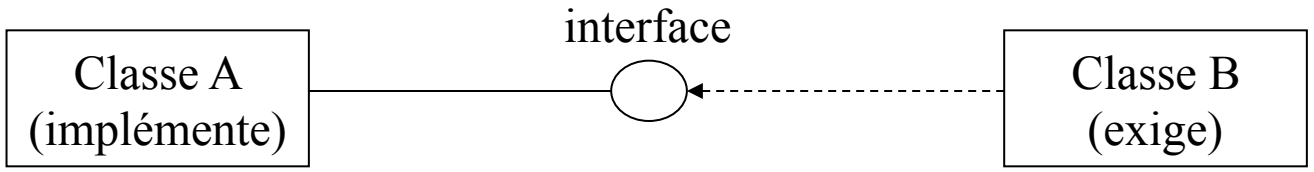
Classe abstraite

- Une classe est abstraite si elle dispose d'un attribut ou d'une méthode abstraite et dans ce cas, elle n'est pas instanciable.



Interface

- Elle décrit le comportement visible d'une classe.
- Elle n'est pas une classe réelle mais une liste de services accessibles par les autres classes.
- Le comportement visible d'une interface est décrit par des **opérations abstraites** dont la **visibilité est publique**
- Une interface est représentée par un petit cercle ayant un nom
- Une classe qui utilise l'interface (implémentée par une autre classe) est connectée via une relation de dépendance vers le cercle représentatif de cette interface.



Interface

- Pour montrer les opérations dans une interface, on la spécifie comme une classe avec le stéréotype «interface».
- Une interface peut être réalisée par plusieurs classes et une même classe peut réaliser plusieurs interfaces.
- Exemple:

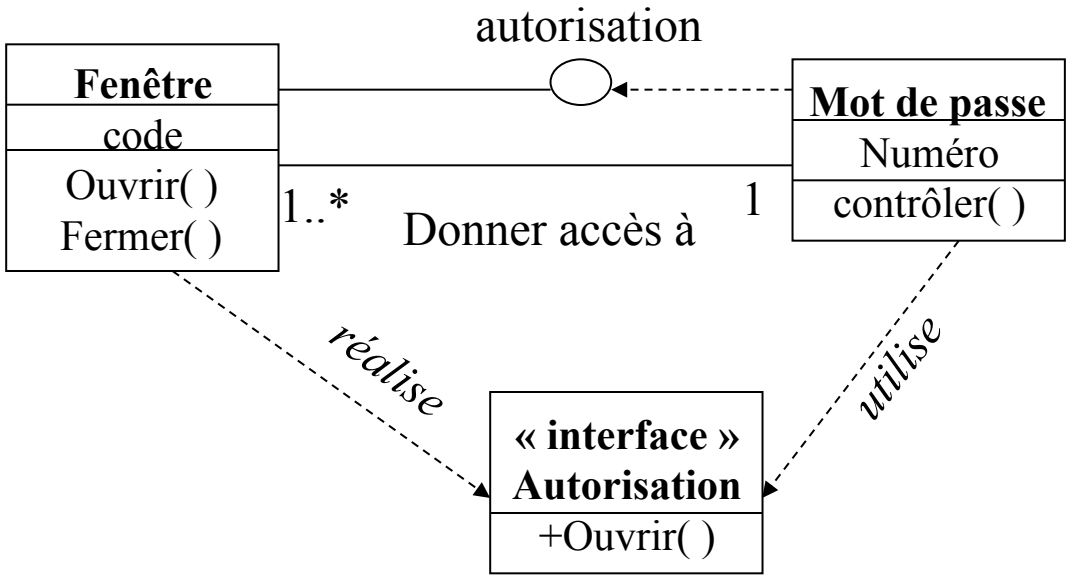
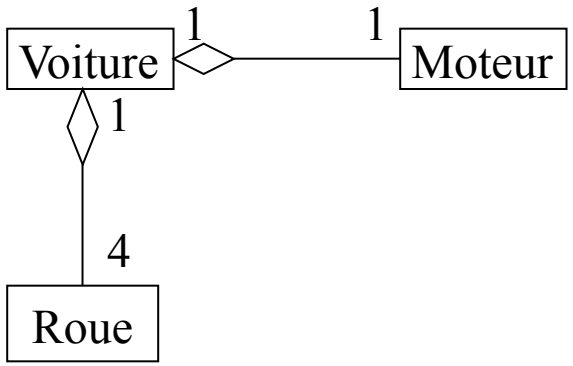


Diagramme d'objets

- Un DOB est une instance d'un DCL et illustre l'état d'un système à un instant t donnée dans le temps.
- Les DOB sont utilisées pour montrer un contexte (avant ou après une interaction entre objets par exemple).
- Un DOB est composé :
  - d'objets (instances de classes),
  - de liens (instances d'associations).
- Représentation d'un objet
  - Un message vers le groupe d'objet atteint l'ensemble des objets du groupe

Diagramme d'objets

- Dans un DOB, le compartiment des opérations n'est pas utile, mais les attributs doivent recevoir des valeurs.
- Dans un DOB, les relations du diagramme de classes deviennent des liens.
- Naturellement, on ne représente pas les multiplicités des extrémités des liens (elles valent toujours 1).



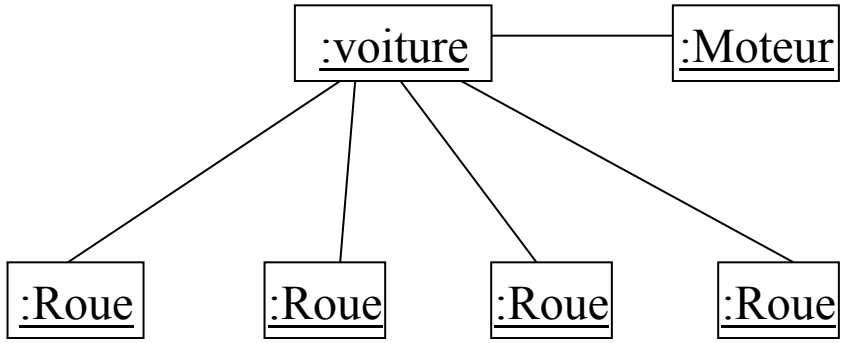
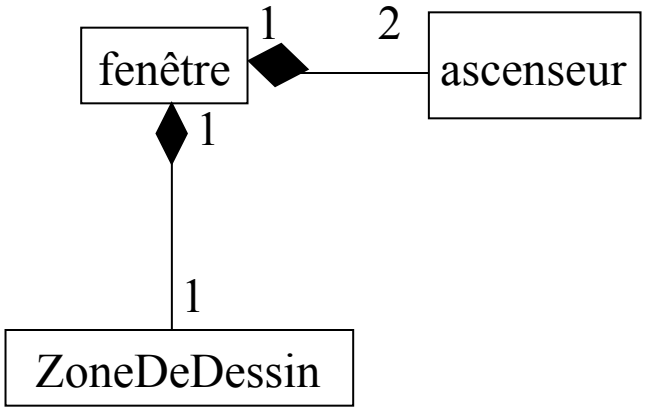
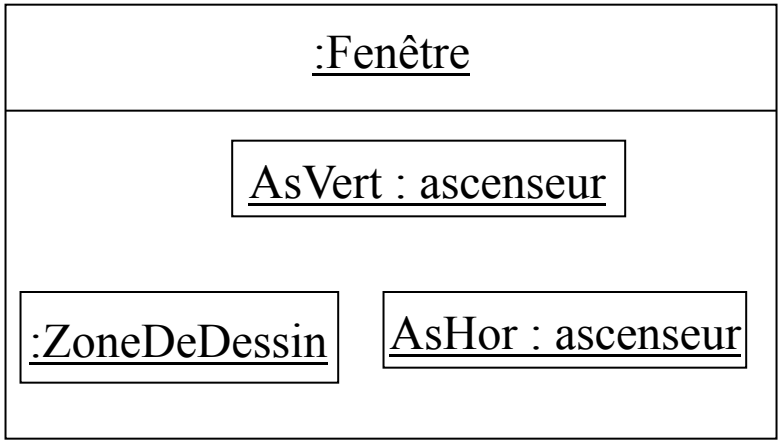


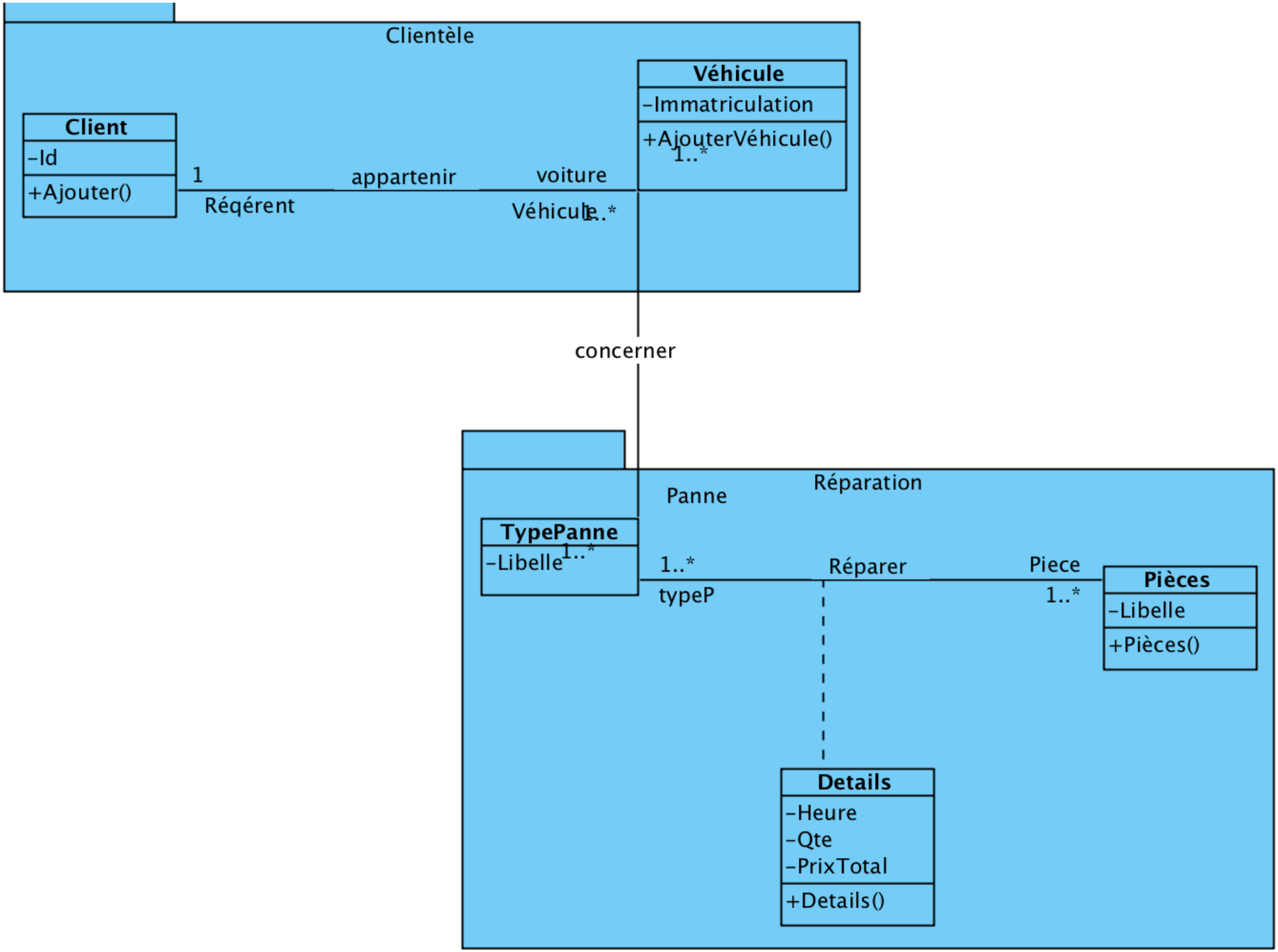
Diagramme d'objets

- Il est possible de représenter les objets composés de sous objets au moyen d'un objet composite.
- L'objet composite se présente comme un objet habituel avec la différence que les attributs sont remplacés par les sous objets.
- Exemple





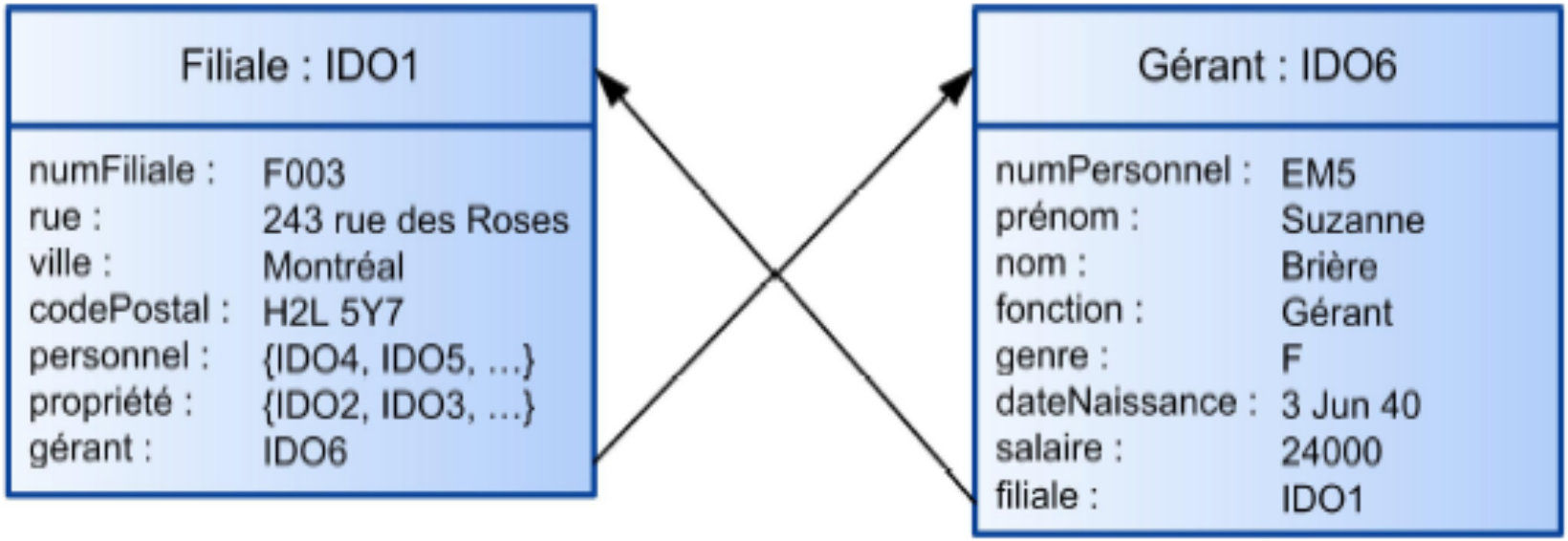
Packages


Clientèle
Véhicule
-Immatriculation
+AjouterVéhicule()
1..*
Client
-Id
+Ajouter()
1
appartenir
voiture
Réqérent
Véhicule
1..*
concerner
Réparation
Panne
TypePanne
-Libelle
1..*
1..*
Réparer
Piece
typeP
1..*
Pièces
-Libelle
+Pièces()
Details
-Heure
-Qte
-PrixTotal
+Details()

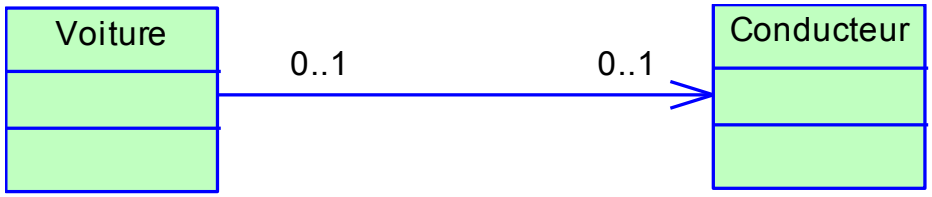
Modèle Statique

- Passage d ’un diagramme de classe en un code

Association 1-1 : exemple 1



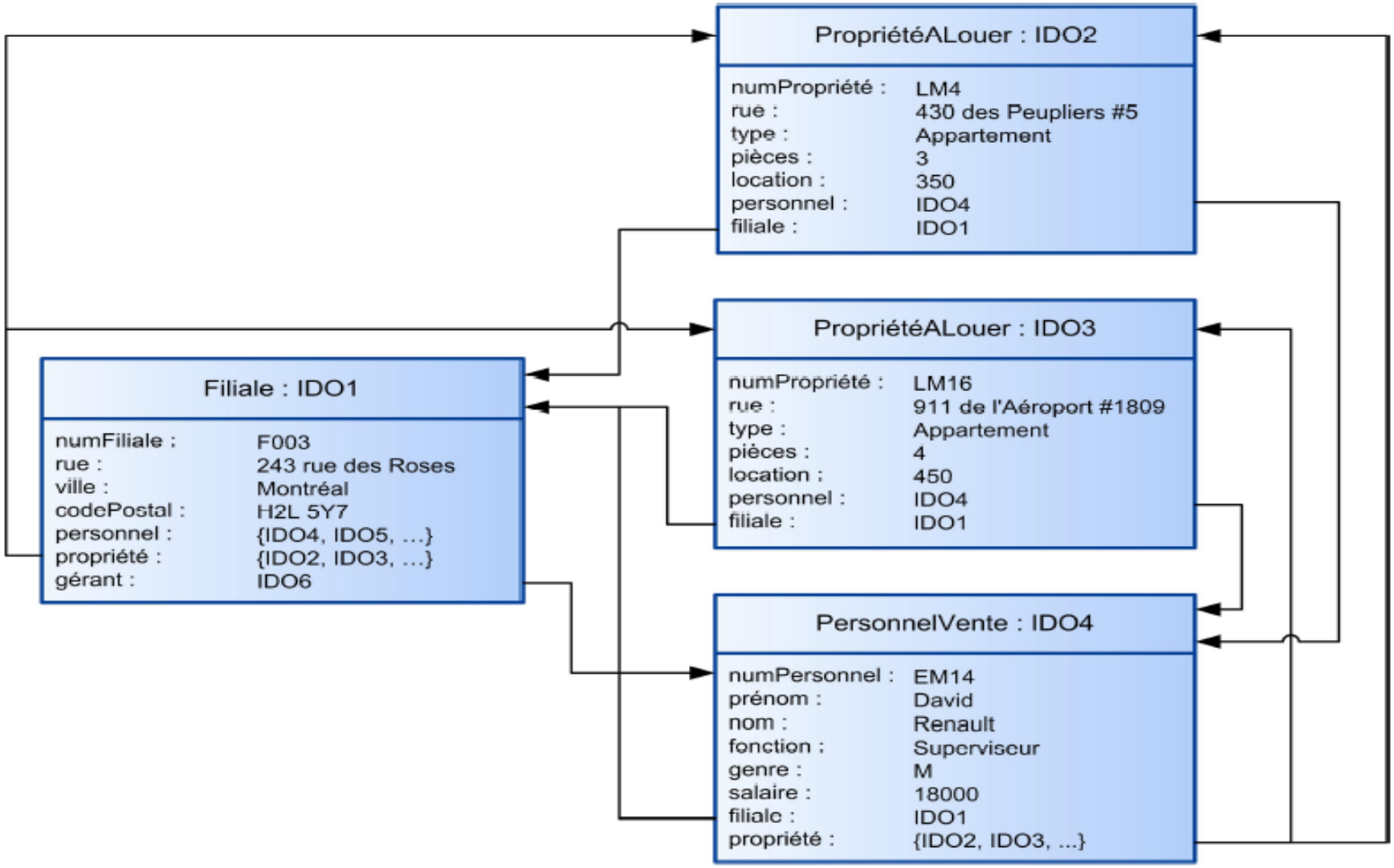
Association 1-1 : exemple 2



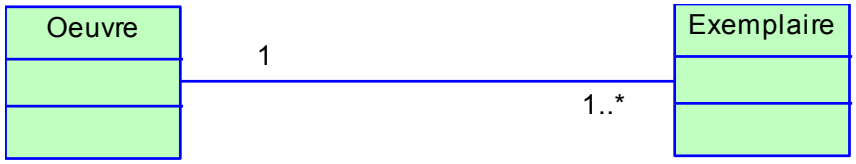

```
Package bagnole

public class Voiture {

Conducteur conducteur ;

public void setConducteur( Conducteur
chauffeur ) {

This.conducteur = chauffeur;

Conducteur.voiture=this }

}
```

```
public class Conducteur { ...
 Voiture voiture
 public void setVoiture (Voiture voiture) {
        this.voiture = voiture;
        voiture.conducteur = this;

}
Public static void main (String [ ] argv) {
        Voiture voiture =  new Voiture ();
        Conducteur conducteur = new Conducteur ();
        conducteur.addVoiture (voiture);
}
}
```

Association 1 vers plusieurs (exemple 1)



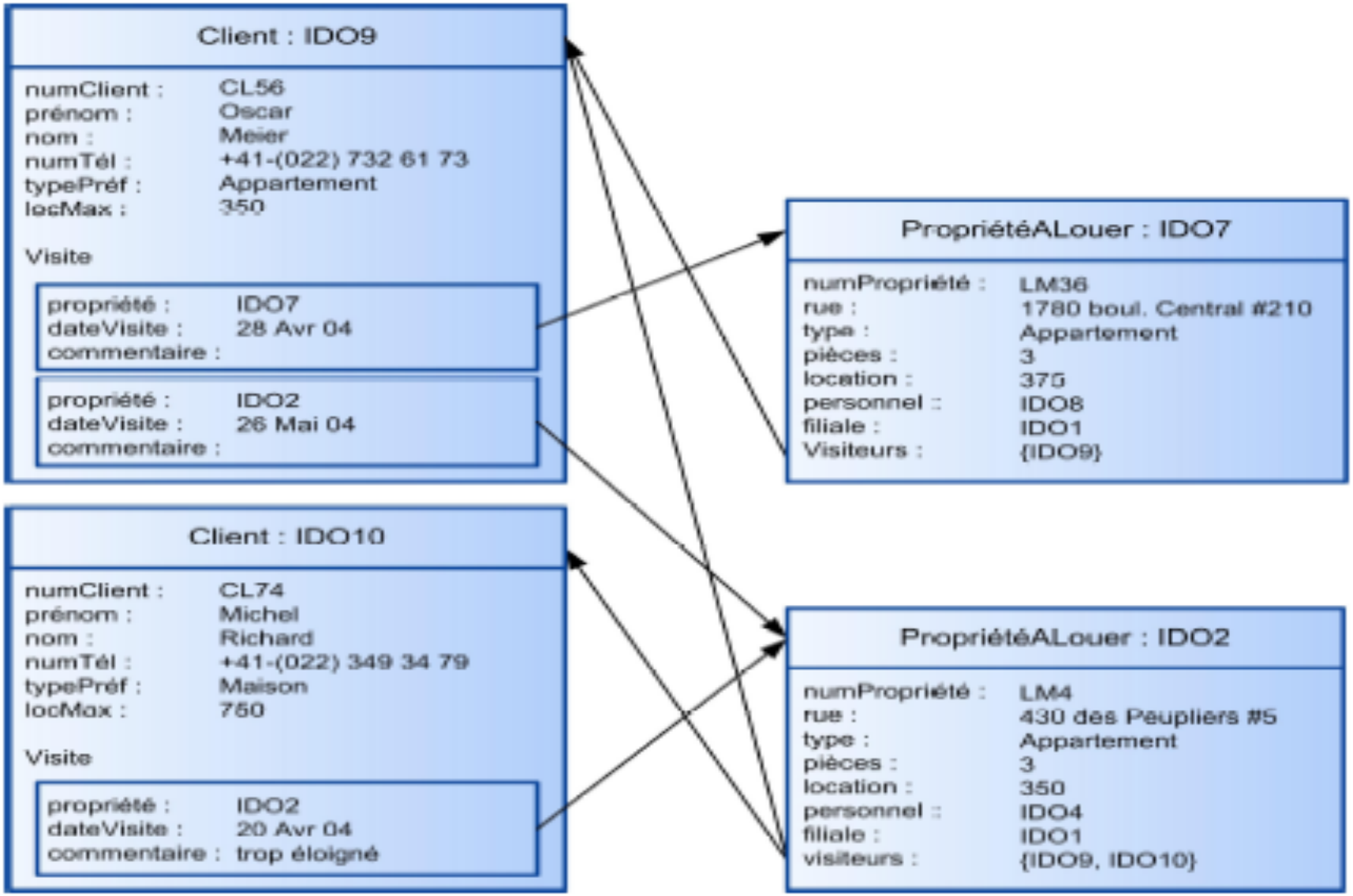
Association 1 vers plusieurs (exemple 2)



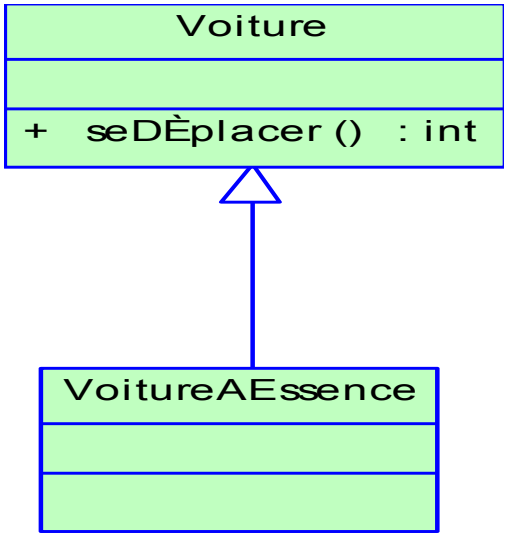

```
Import java.util.Vector
class Œuvre {
Vector exemplaires = new Vector();

public void addExemplaire ( Exemplaire exemplaire )
{
        exemplaires.addElement (exemplaire) ;}

public static void main (string [ ] argv ) {
        Œuvre Kirikou = new Oeuvre();
        Exemplaire exemplaire = new Exemplaire (1);
        Kirikou.addExemplaire (exemplaire);
        monEntreprise.removeClient (client);
}
}
```

```
Class Exemplaire {
int numéro
Œuvre œuvre ;
public Exemplaire (int numéro) {
this.numéro = numéro; }
public void setOeuvre (Oeuvre
œuvre) {
Œuvre.exemplaires.addElement
(this);
this.oeuvre = œuvre;
}
}
```

Associations plusieurs - plusieurs



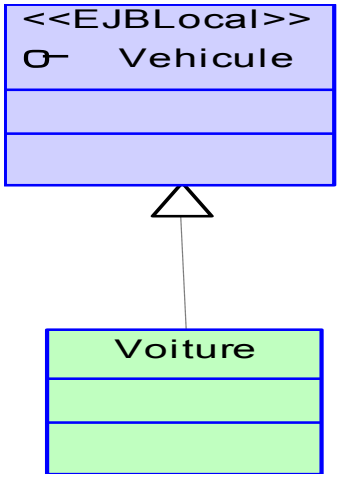
Implémentation de l'Héritage




```
Public class Voiture {
                    public void seDéplacer () {
                            // ….
                            }
                }
```

```
public class VoitureAEssence extends Voiture { }
```

Implémentation d'une interface




```
Public interface  Vehicule {
                    public void seDéplacer ();
                  }

                        public class Voiture  implements Vehicule {
                          public void seDéplacer () {
                                                    // ....
                                                  }
                                                }
```

Exercices

- Faire le schéma de génération pour
  - Agrégation
  - Composition
  - Héritage

Exercices Pratiques

- Modélisez les cas suivants:
  - Une personne est caractérisée par son nom, son prénom, son sexe et son âge. Les responsabilités de la classe sont entre autres le calcul de l'âge, le calcul du revenu et le paiement des charges. Les attributs de la classe sont privées; le nom, le prénom ainsi que l'âge de la personne font partie de l'interface de la classe *Personne*.
  - Les étudiants peuvent être comparés par rapport à l'attribut moyenne générale et les livres sont comparables par rapport à leur prix de vente. Pour des raisons d'homogénéité des interfaces présentées par les classes, tous les objets comparables utilisent la même opération *compareTo(instance)*.

Exercices Pratiques (suite)

- Le chat et le chien sont des animaux. Les animaux possèdent tous un nom. Proposez une modélisation de cette situation en faisant apparaître que d'autres animaux que les chats et les chiens existent et qu'un animal ne peut pas être à la fois un chat et un chien.

- Les animaux, en fonction de leur catégorie (chat, chien …), ont un cri spécifique. Si la catégorie de l'animal n'est pas connue, le cri ne peut être réalisé. Compléter la modélisation précédente pour inclure cette propriété.

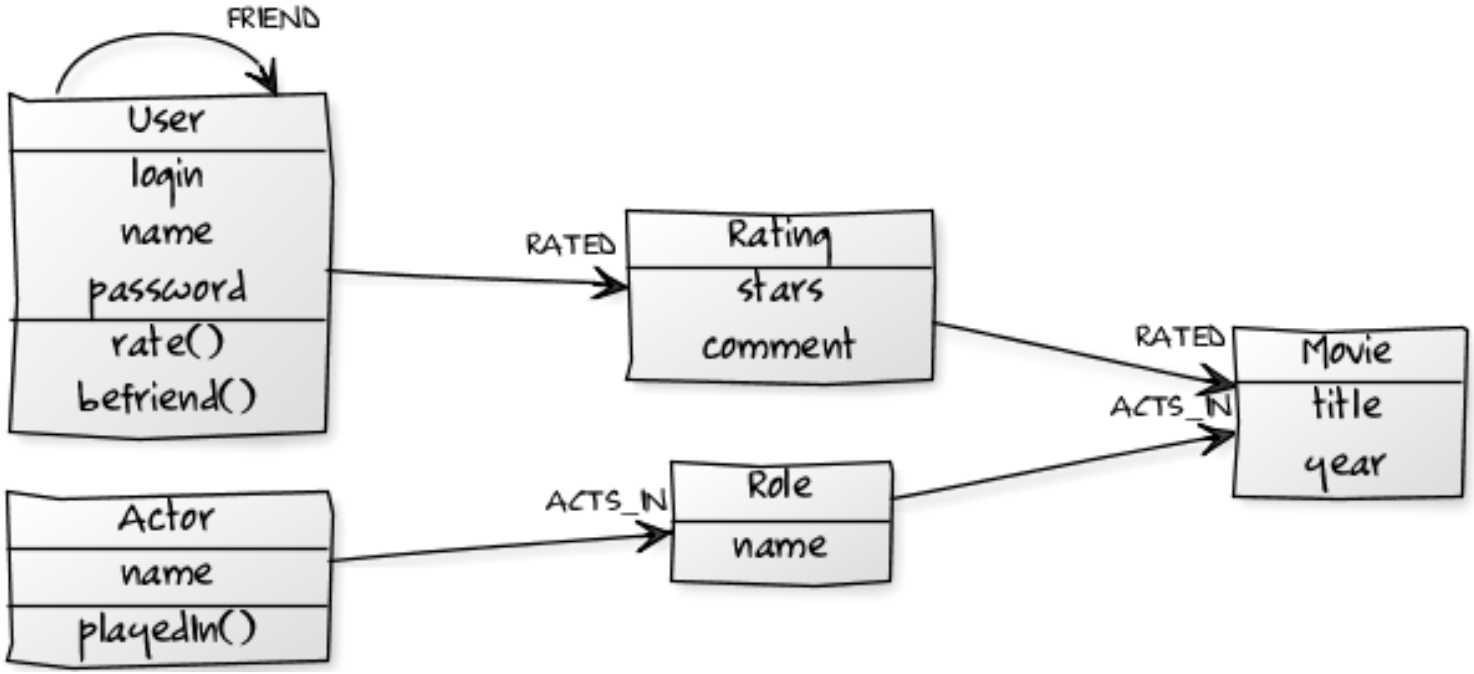
- Un étudiant et un enseignant sont des personnes particulières. Un doctorant est un étudiant qui assure des enseignements.

BD et Modèles

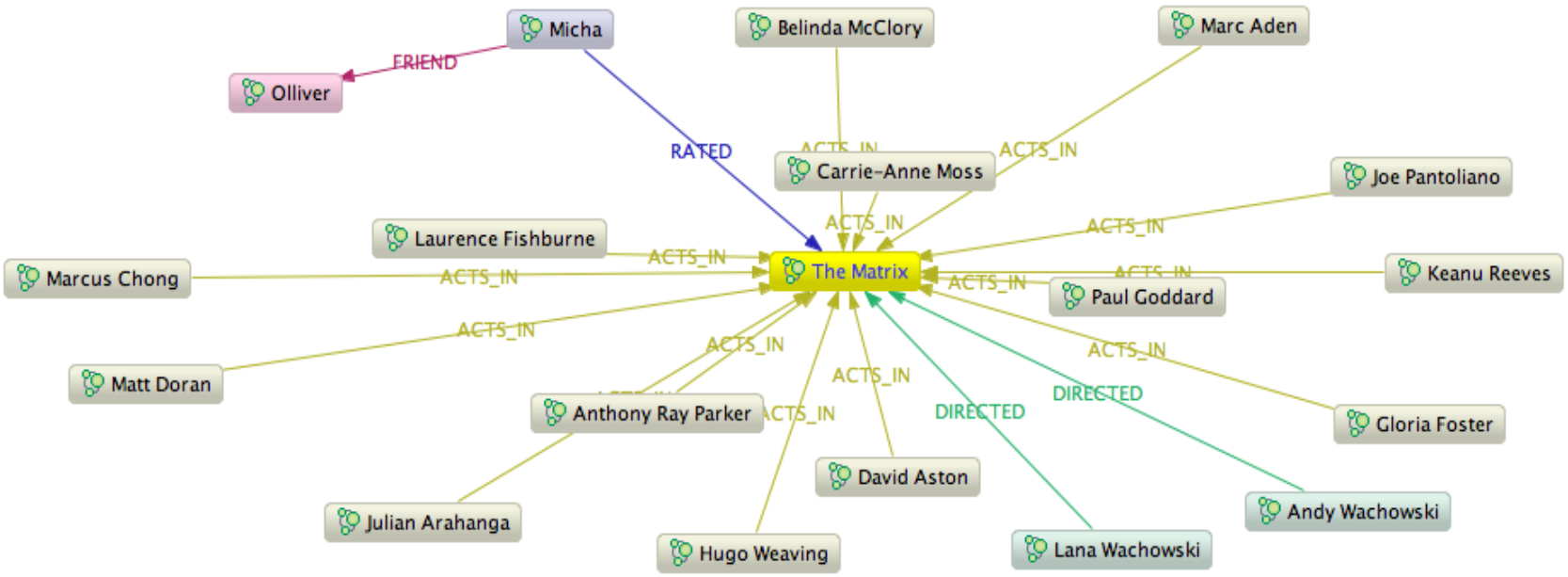
- Base de données (BD)
  - Collection d'occurrences d'un diagramme de classe stockée en mémoire secondaire
  - gérée par un système de gestion de bases de données (SGBD)

- Type d'organisation (modèle de données)
  - Relationnel
  - Graphe
  - Clé-valeur
  - Colonne
  - Document
- Le type d'organisation visé a un impact sur la modélisation conceptuelle

Modèle de données



Modèle de données : Exemple Graphe



Modèle de données : Exemple Relations

login	Name	Friend
Micha	Micha	Olivier
Olivier	Olivier	Null

Id	star	Comments	Name
iii	1	Boring	Micha
oooo	2	Exciting	Micha

title	Year	rating
Matrix	2000	iii
Un Café sans	2016	Micha

Concepts de base

- Codd (1970)
- Relation
  - *Schéma* (intention): liste d'attributs et contraintes d'intégrité
  - *Table* (extension)
- Schéma d'une BD
  - Ensemble de schémas de relations
  - Contraintes d'intégrité entre relations
- Contraintes d'intégrité sémantique
  - Clés
  - Dépendances fonctionnelles, ...

Concepts de base

- Relation
  - Schéma (intention) : liste d'attributs et de contraintes d'intégrité

    EDITEUR(<u>code_ed</u>, Nom_ed, Lieu)

    Lieu ∈ {Paris, Montréal, Boston, New York}
  - Table, extension, ou état

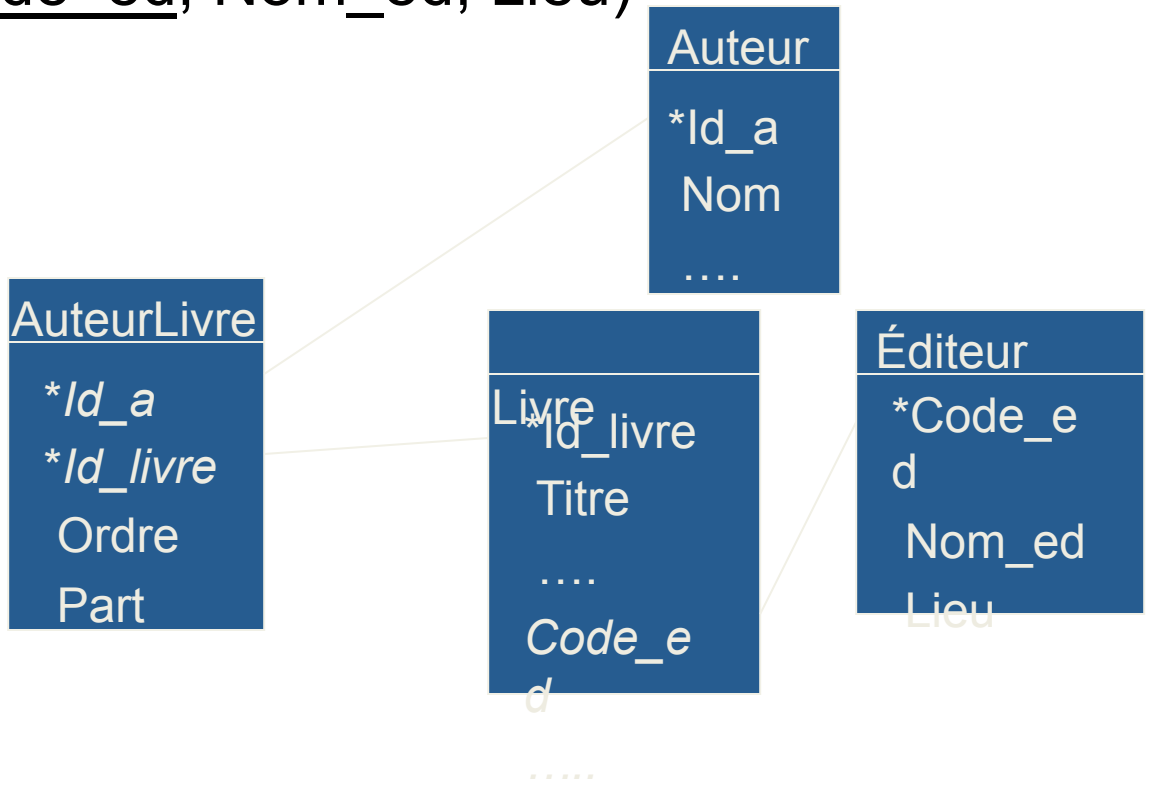
Concepts de base

- Tuple
  - Enregistrement ou ligne (“row”, “record”)
  - Instance (occurrence) particulière d’entité ou d’association (ex. un éditeur)
- Attribut
  - Colonne ou champ (“column”, “field”)
  - propriété d’une entité ou d’une association
  - Ex: Lieu est un attribut pour la relation *editeur*
- Domaine
  - d’un attribut : ensemble de valeurs possibles
  - Ex: Lieu $\in$ au domaine {Montréal, Paris, New York, Grand Bassam}

Concepts de base

- Ex. Schéma relationnel de la base *BD_Auteur*

  Auteur (Id_a, Nom, Téléphone, Ville, Province, Contrat)
  Livre (Id_livre, Titre, Catégorie, *Code_ed*, Prix, Droits, Date)
  AuteurLivre (*Id_a, Id_livre*, Ordre, Part)
  Éditeur (Code_ed, Nom_ed, Lieu)



Concepts de base

- Ex. Contenu de BD_Auteur

Auteur

id_a	Nom	Téléphone	Ville	Province	Contrat
10	Alain Dubuc	819 734-8889	Hull	Qc	0
20	Pierre Lebrun	514 655-1750	Montreal	Qc	0
30	Amina Missaoui	418 678-3444	Quebec	Qc	1
40	Brian Adams	416 998-2345	Toronto	Ont	0
50	Luc Nadeau	506 877-4466	Moncton	Nb	1
60	Laks Agrawal	416 888-9834	Toronto	Ont	1
70	Michel Montignac	331 678-5555	Paris		0
80	C-J Date	202 812-5675	Washington	Dc	1
90	Jacques Dupont	331 876-0685	Paris		1

Éditeur

Code_ed	Nom_ed	Lieu
11	Gaetan Morin	Montreal
22	Addison Wesley	New York
33	Flammarion	Paris
44	Eyrolles	Paris
55	MS Press	Boston

Concepts de base

Livre

id_livre	Titre	Catégorie	Code_ed	Prix	Droits	Date
1000	Menus quebecois	cuisine	11	20.99	10	Jun 17 1987
2000	SQL Server 6.5	informatique	55	65.00	15	Apr 21 1995
3000	BD	informatique	44	57.99	16	Feb 4 1998
4000	Management	gestion	44	40.99	15	Oct 30 1990
5000	A guide to SQL	informatique	22	43.99	20	Feb 24 1991
6000	Je maigris	dietetique	33	8.99	15	Jan 15 1999

AuteurLivre

Id_a	Id_livre	Ordre	Part
10	1000	1	5
10	3000	3	17
20	1000	2	5
30	4000	2	5
40	2000	1	15
50	4000	1	5
60	4000	3	5
70	6000	1	15
80	3000	1	8
80	5000	1	13
90	3000	2	7
90	2000	2	10

Contraintes d'intégrité sémantique

- Types de contraintes
  - Clé primaire et clé étrangère
  - Contraintes de domaine : les valeurs d'un attribut appartiennent à un domaine spécifique

    Ex. Ville $\in$ {Paris, Boston, ...}

    Ex. part $\leq$ 20
  - Règles d'implication

    Ex. Si l'employé est analyste, alors son salaire dépasse 50000$
  - Dépendances fonctionnelles
  - Etc.

Contraintes d'intégrité sémantique

- Clés d'une relation
  - Clé primaire : attribut ou groupe d'attributs identifiant d'une manière unique les occurrences. La clé est toujours renseignée ("NOT NULL").

    Ex. *Id_livre* est la clé primaire de la relation *livre*
  - Clé étrangère d'une relation R : fait référence à la clé primaire d'une relation R'. R et R' ne sont pas nécessairement des relations distinctes.
  - Ex. *Id_livre* est une clé étrangère de la relation *auteurLivre* car elle fait référence à *Id_livre*, clé de la table *livre*

Conversion de diagramme de classes

Règle 0 & 1: attribut et classe

Classe

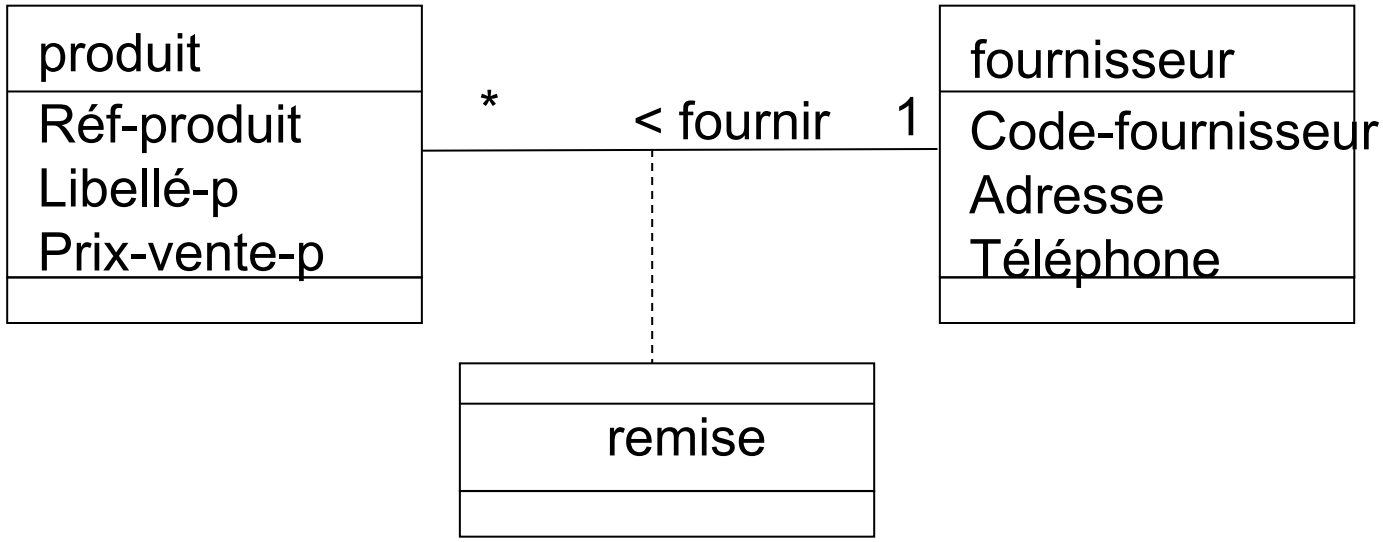
Relation / Table

Produit (Réf-produit, Libellé-p, Prix-vente-p)

Fournisseur (Code-fournisseur, Adresse, Téléphone)

Règle 2 : relation de multiplicité (1)

Classe



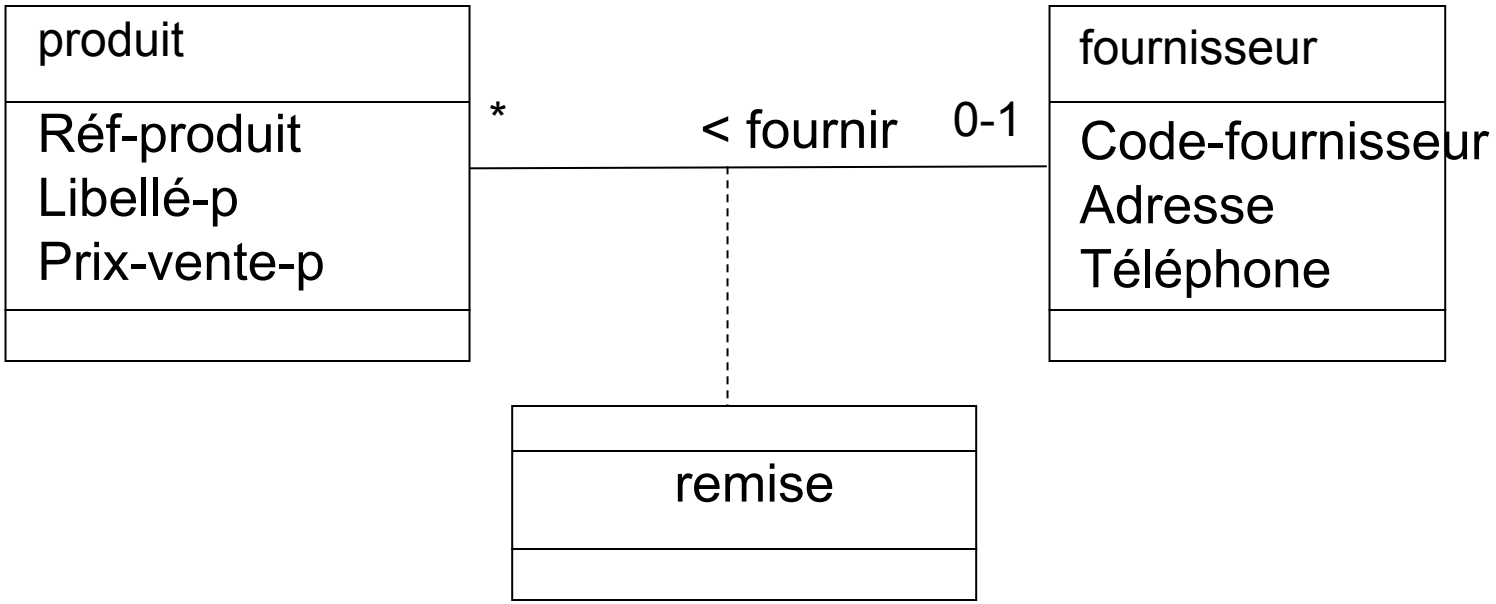
Relation / Table

Produit (Réf-produit, Libellé-p, Prix-vente-p, Code-fournisseur, remise)

Fournisseur (Code-fournisseur, Adresse, Téléphone)

Règle 3 : relation de multiplicité (0-1)

Classe



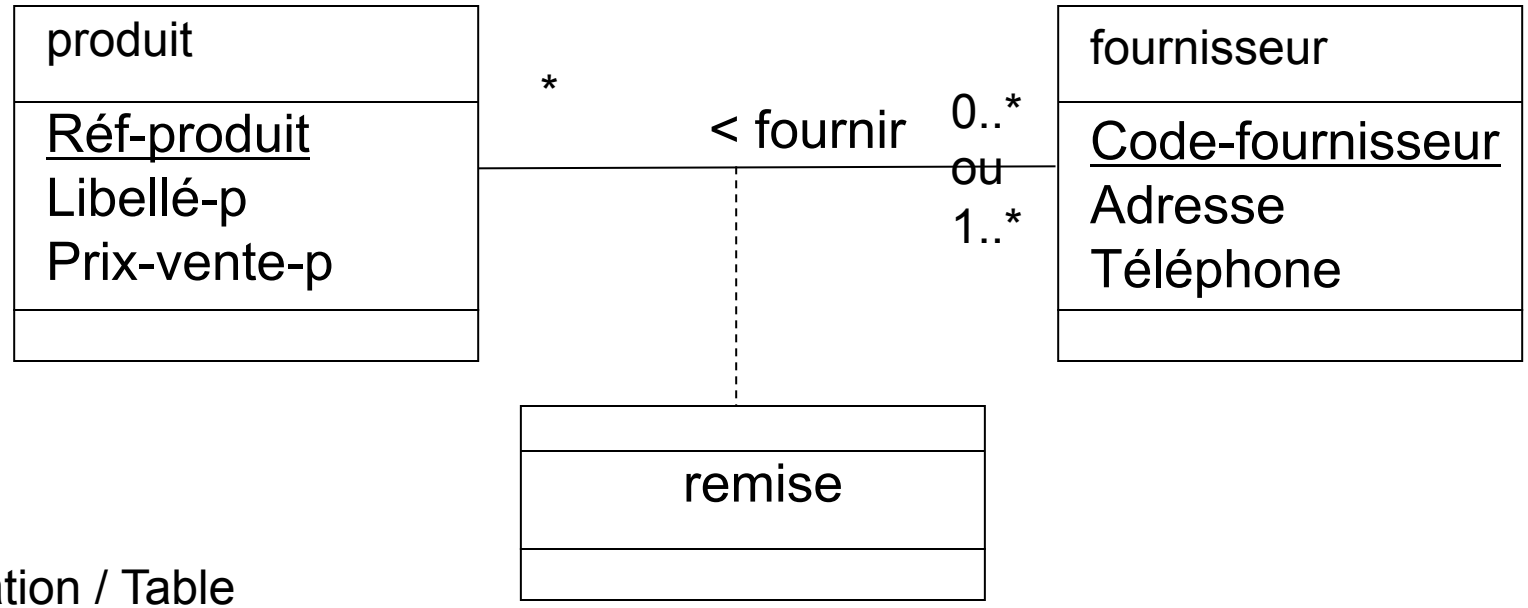
Relation / Table

Produit (Réf-produit, Libellé-p, Prix-vente-p, remise, Code-fournisseu

Fournisseur (Code-fournisseur, Adresse, Téléphone)

Règle 4 : relation de multiplicité (0..*) (1..*)

Classe



Relation / Table

Produit (Réf-produit, Libellé-p, Prix-vente-p)

Fournisseur (Code-fournisseur, Adresse, Téléphone)

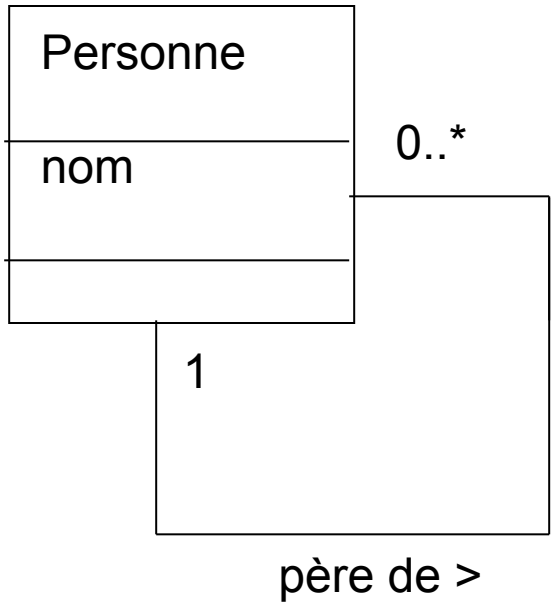
Fournir (Réf-produit, Code-fournisseur, remise)

Règle 5 : relation réflexive orientée

Relation / Table

Père (nom-fils, nom-père)

Classe

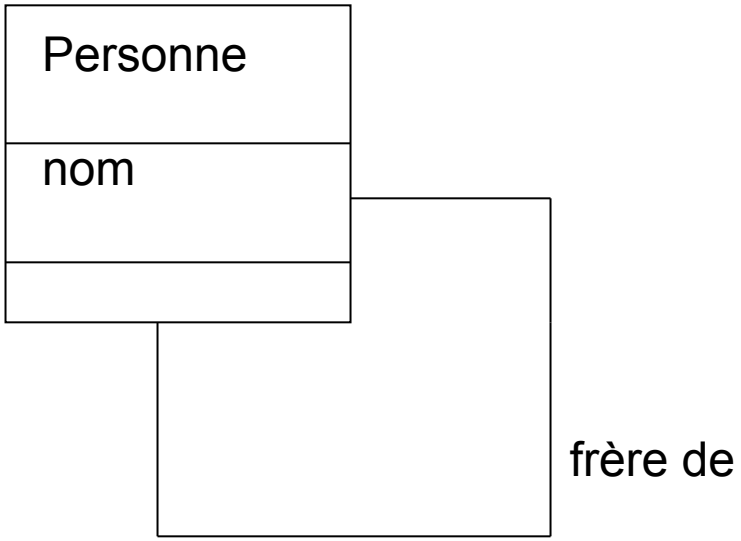


Règle 6 relation réflexive symétrique

Relation / Table

Personne (Nom)
Frère (nom, nom)

Classe



Attention, la relation étant transitive, des traitements devront être associés au modèle.

FIN

Diagramme de classes (DCL) et d'objets (DOB)

- Les classes et les objets modélisent les entités matérielles ou immatérielles qui existent dans le système étudié.

- Le DCL fournit une représentation abstraite des objets du système qui doivent interagir pour réaliser les fonctions du système.

- Le DCL montre la structure statique du système
  - Pas de prise en compte du facteur temporel

- Le DOB est un exemple du DCL qui donne une photographie du système à un instant *t*.