Bases de données

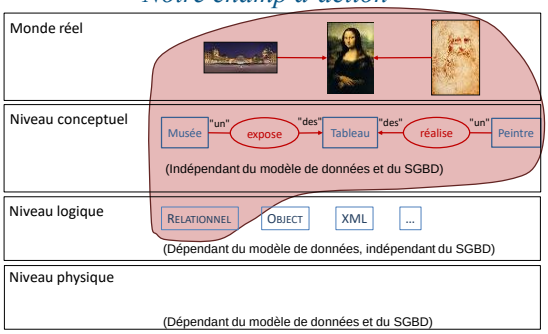
Modélisation

Xavier Tannier
xavier.tannier@sorbonne-universite.fr

Modélisation

- **Pourquoi** modéliser ?
  - Pour mieux comprendre les données
  - Pour mieux séparer les données
  - Pour mieux assurer la cohérence de l'ensemble
  - Pour faciliter la visualisation du système
- **Comment** modéliser ?
  - Isoler les concepts fondamentaux et les sous-concepts
  - Déterminer les relations entre les concepts

Notre champ d'action



Modélisation

- Une base de données peut être modélisée par
  - Une collection d'entités

  - Des associations (relations) entre ces entités

Entité

- Une entité est un objet qui existe et que l'on peut différencier des autres objets
  - Les entités ont des attributs
  - Un ensemble d'entités regroupe des entités qui ont le même type et les mêmes attributs

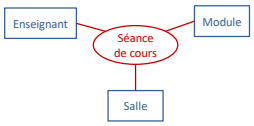
Musée

ID	Nom	Ville
1354	Le Louvre	Paris
78954	Centre Pompidou	Paris
9731	Centre Pompidou	Metz
4231	Musée Soulages	Rodez
35	MoMA	New York
…	…	…

Association

- Une association relie deux entités ou plus

Association de degré 2



Association de degré 3

Association

- Ensemble d'associations

Musée

ID	Nom	Ville
1354	Le Louvre	Paris
78954	Centre Pompidou	Paris
9731	Centre Pompidou	Metz
4231	Musée Soulages	Rodez
35	MoMA	New York
…	…	…

EXPOSE

Oeuvre

ID	Œuvre
62456	La Joconde
26853	La Tristesse du Roi
9743	Brous de noix
154	La Nuit étoilée
8432	Les Noces de Cana
…	…

Association

- Une association peut également avoir des attributs

Musée

ID	Nom
1354	Le Louvre
78954	Centre Pompidou
9731	Centre Pompidou
4231	Musée Soulages
35	MoMA
…	…

EXPOSE

depuis : 1802
depuis : 1989
depuis : 1798
depuis : 2014
depuis : 1941

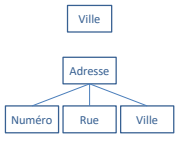
Oeuvre

ID	Œuvre
62456	La Joconde
26853	La Tristesse du Roi
9743	Brou de noix
154	La Nuit étoilée
8432	Les Noces de Cana
…	…

Attribut, domaine

- Les attributs sont des propriétés descriptives que possède un ensemble d'entités ou d'associations
- Le **domaine** est l'ensemble des valeurs que peut prendre un attribut
- Un attribut peut être :

Simple ou composite



Mono- ou multi-valué

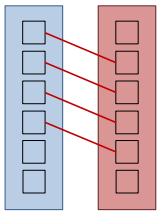
Dérivé

Cardinalité

- Le nombre d'entités à laquelle une autre entité peut être liée par une association (binaire) donnée

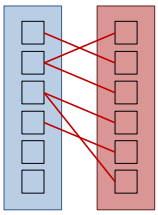
- Une à une (*one to one*)
- Une à plusieurs (*one to many*)
- Plusieurs à une (*many to one*)
- Plusieurs à plusieurs (*many to many*)

Cardinalité



Une à une (*one to one*)

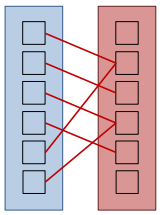
- Un·e étudiant·e Sorbonne Université ↔ un compte informatique
- Un·e citoyen·ne français·e ↔ 0 ou 1 mariage/PACS (à la fois)



Une à plusieurs (*one to many*)

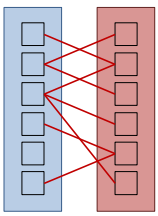
- Un musée contient des œuvres
- Une personne a 0, 1 ou plusieurs téléphones portables

Cardinalité



Plusieurs à une (*many to one*)

- Une personne a un seul médecin référent, un médecin a *n* patient·e·s
- Une personne a une seule adresse principale



Plusieurs à plusieurs (*many to many*)

- Enseignant·e·s / étudiant·e·s
- Artistes / musées

Clés

Clé d'une entité

- Une clé (ou clé candidate) :
  - Est un sous-ensemble d'attributs d'un type d'entité
  - Suffit à identifier une entité de façon unique
  - Ne peut pas être réduite sans perdre la propriété précédente
- Clé primaire :
  - Une des clés

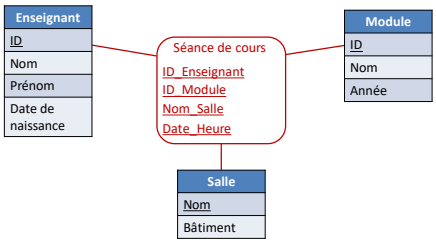
Clé d'une entité

ID	Série	Saison	Épisode	1ère diffusion	Titre
342	Game of Thrones	1	1	17/04/2011	Winter is coming
343	Game of Thrones	1	2	24/04/2011	The Kingsroad
456	Game of Thrones	2	4	22/04/2012	Garden of Bones
673	Homeland	2	3	14/10/2012	State of Independence
843	Homeland	3	2	06/10/2013	Uh... Oh... Ah...
...	...	...		...	

- *{Série, Épisode}* n'est pas une clé (pas unique)
- *{Série, Saison, Épisode, 1ère diffusion}* n'est pas une clé (unique mais pas irréductible)
- *{ID, Série, Saison, Épisode}* n'est pas une clé (unique mais pas irréductible)
- *{ID}* et *{Série, Saison, Épisode}* sont des clés
- La clé primaire reste à définir

Clé d'une association

- La combinaison des clés primaires de toutes les entités associées + d'autres attributs éventuels



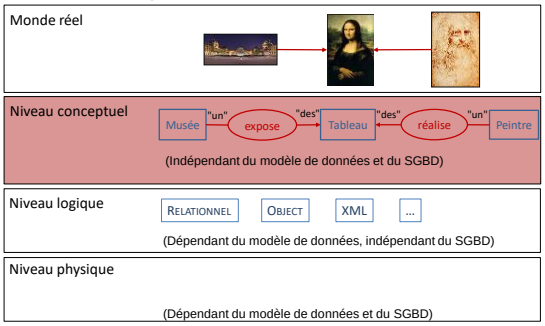
SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS DEPUIS 1257

POLYTECH SORBONNE

Diagramme de classe UML (simplifié)

Xavier Tannier
xavier.tannier@sorbonne-universite.fr

Diagramme de classes UML

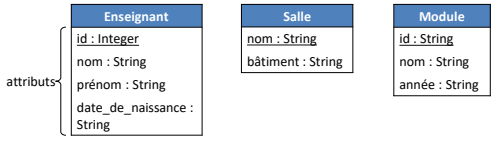


Modèles

- **Modèle Entité-Association**
  - Modèle de Chen (1976)
  - Spécifique à la modélisation de bases de données
  - Très utilisé, surtout en France (Merise)
  - Très proche du diagramme de classes UML
- **Diagramme de classes UML**
  - Plus répandu internationalement
  - S'inscrit dans le cadre générique de la modélisation UML (diagramme de séquences, états-transitions, etc.)
  - De nombreux outils de modélisation graphique
  - C'est de lui qu'on va parler (mais les équivalences sont en général évidentes)

Classe

- Classe = entité

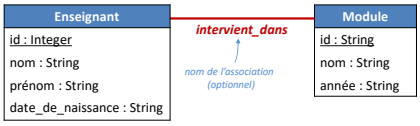


Les types de base :
- *Integer*
- *String*
- *Real*
- *Boolean*

(on en reparle)

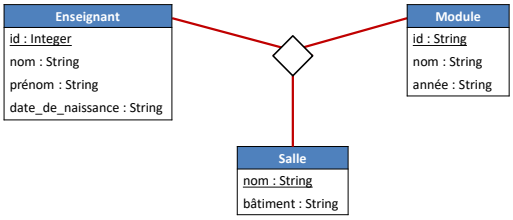
Association (relation)

- Association binaire

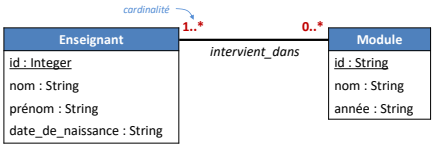


Association (relation)

- Association n-aire



Cardinalité



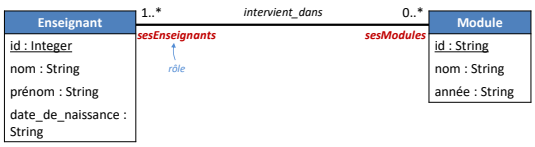
- 0..* (ou N) = zéro, un ou plusieurs
- 1..* (ou N) = un ou plusieurs
- 1..2 = un ou deux
- 1 = un et un seul

Et donc :

- 0..* → 0..* = many to many
- 1 → 0..* = one to many
- 0..* → 1 = many to one
- 1 → 1 = one to one

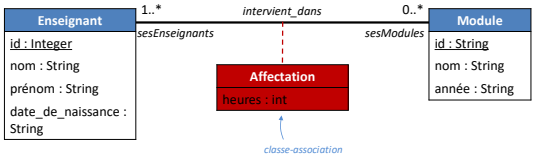
Rôle

- Rôle : nomme l'extrémité d'une association, permet d'accéder aux objets liés par l'association à un objet donné



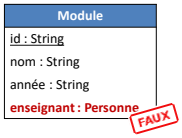
Association avec attributs

- « Classe-association »
  (pour ajouter des attributs à une association)



Attribut et association

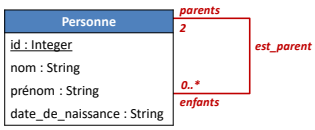
- ⚠ Pas d'attribut ayant pour type une entité du modèle



- On remplacera toujours cet attribut par une association

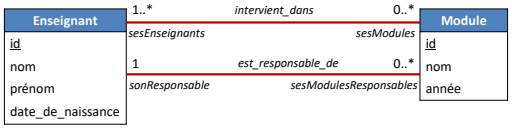
Association réflexive

- Association entre une classe et elle-même

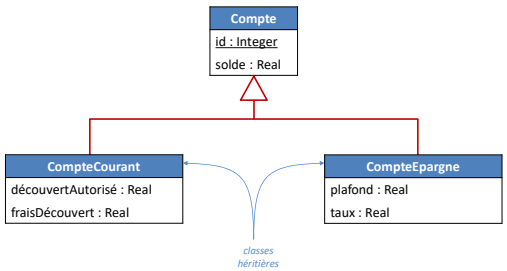


Association multiple

- Plusieurs associations entre deux classes



Hiérarchisation



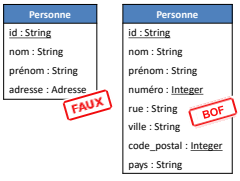
Conception :
Bonnes pratiques

Trouver les classes et les associations

- On part du texte de **spécifications**
- Les **classes** sont :
  - Des objets physiques, des personnes, lieux, organisation, des entités abstraites
  - Des **noms** qui apparaissent dans le texte
  - *compte, film, client, enseignant, série, épisode, réservation, cours…*
- Les **associations** sont :
  - Des relations entre les classes
  - Souvent exprimés par des **verbes** dans le texte
  - *distribue, enseigne, achète, voit, réserve, effectue, donne…*
  - Associations spéciales *: est un*, *est composé de*, *contient*, *a un*, etc.
    → Penser à l'héritage, à la composition

Les attributs

- Un attribut ne doit pas avoir comme type une classe du modèle
- Un attribut multiple doit souvent être transformé en classe

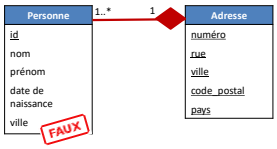


(ou on utilise un type de données complexe – dataType)

En général

- Il est souvent préférable d'avoir plusieurs relations binaires plutôt qu'une relation n-aire
  (on peut toujours transformer une relation n-aire en plusieurs relations binaires)

- Faire la chasse aux informations redondantes



En général

- Voir aussi : normalisation de bases de données (plus tard)

Les types de base

- UML n'est pas un langage de programmation et la question des types de base n'est pas déterminante
- Types de base « officiels » :
  - Integer et UnlimitedNatural
  - String
  - Real
  - Boolean
- Par exemple, pas de Date
- Pour représenter une date, plusieurs possibilités :
  1. On utilise un type complexe (dataType)
  2. On dit String et on voit plus tard pour l'implémentation
  3. On ferme les yeux et on dit Date

Exemple : les produits alimentaires

- Une marque vend des produits alimentaires
- Un produit alimentaire a un nom, un code, un type de packaging, des valeurs nutritionnelles,

- Un produit alimentaire est vendu dans un ou plusieurs pays
- Un produit alimentaire contient des ingrédients (dans une certaine quantité) et inscrit des ingrédients sur la liste des ingrédients. Ainsi, un ingrédient peut être présent dans le produit mais pas indiqué, indiqué mais pas présent, ou présent et indiqué.
- Un produit contient également des additifs. Ces additifs sont des colorants, conservateurs, édulcorants, émulsifiants, exhausteurs… Certains sont nocifs et d'autres non.

UML : Fin

- ⚠ Nous n'avons vu ici qu'une petite partie de la conception en diagrammes de classe (la partie utile pour la conception d'une base de données)
- ⚠ Nous n'avons vu ici qu'une infime partie d'UML

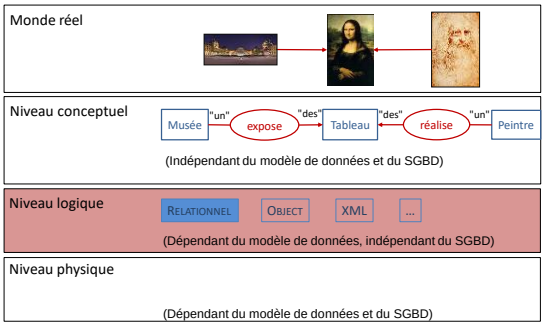
SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS DEPUIS 1257

POLYTECH SORBONNE

Modèle relationnel

Xavier Tannier
xavier.tannier@sorbonne-universite.fr

Modèle relationnel



Modèle relationnel

- Une relation :

Colonnes (ou attributs)

ID	Œuvre	Lieu	Artiste
62456	La Joconde	Le Louvre	Léonard de Vinci
26853	La Tristesse du Roi	Centre Pompidou	Matisse
13796	Nymphéas bleus	Musée d'Orsay	Monet
3579	La Célestine	Musée Picasso	Picasso
…	…	…	…

Lignes (ou tuples, ou n-uplets, ou enregistrements…)

Modèle relationnel

- Les valeurs des attributs doivent être atomiques (indivisibles)

ID	Œuvre	Lieu
62456	La Joconde	Le Louvre, Paris
87433	Christina's world	MoMA, New York
34578	A Young Lady's Adventure	Tate Modern, London
3579	La Célestine	Musée Picasso, Paris
…	…	…

Point vocabulaire : schéma et instance

Schéma

– Un **musée** **expose** des **tableaux** qui sont **réalisés par** des **peintres**

(schéma non contractuel)

Instance

– Le **Louvre** **expose** la **Joconde** qui est **réalisée par** **Leonard de Vinci**

Point vocabulaire : domaine et valeur NULL

Domaine

L'ensemble des valeurs que peut prendre un attribut

NULL

Une valeur spéciale présente dans tous les domaines
Ajoute du piment à toutes les opérations algébriques

ID	Œuvre	Artiste
62456	La Joconde	Léonard de Vinci
62045	La Venus d'Arles	NULL
13796	Nymphéas bleus	Monet
…	…	…

Point vocabulaire : cardinalité et arité

Cardinalité Nombre de lignes

Arité Nombre de colonnes

ID	Œuvre	Lieu	Artiste
62456	La Joconde	Le Louvre	Léonard de Vinci
26853	La Tristesse du Roi	Centre Pompidou	Matisse
13796	Nymphéas bleus	Musée d'Orsay	Monet
3579	La Célestine	Musée Picasso	Picasso
…	…	…	…

Remarque

- L'ordre des lignes dans la base est arbitraire.
  Si vous voulez un ordre, il faudra le demander

ID	Oeuvre	Lieu	Artiste
62456	La Joconde	Le Louvre	Léonard de Vinci
26853	La Tristesse du Roi	Centre Pompidou	Matisse
13796	Nymphéas bleus	Musée d'Orsay	Monet
3579	La Célestine	Musée Picasso	Picasso
…	…	…	…

Formellement…

Si $A_1, A_2, A_3, \dots, A_n$ sont des **attributs** :

- $R = (A_1, A_2, A_3, \dots, A_n)$ est un **schéma de relation**
  par exemple, Œuvre = (id, nom, artiste, année)

- Pour $i = 1..n$, si l'ensemble de valeurs D_i est le **domaine** de l'attribut A_i, alors une **relation** *r* est un sous-ensemble de
  $$D_1 \times D_2 \times \cdots \times D_n$$
- Autrement dit, une relation est un ensemble de tuples $(a_1, a_2, \dots, a_n)$ où chaque $a_i \in D_i$.
  - Les valeurs d'une relation sont spécifiées par une table.
  - Un élément *t* de *r* est un tuple, représenté par une ligne dans une table.

Clé étrangère

- Clé étrangère
  - Une clé d'une entité référencée dans une autre relation
  - Permet de s'assurer de l'intégrité référentielle entre deux tables

Cours

Cours	Date et heure	Bâtiment	Salle
Prog. Web	…	640	C101
BD	…	620	A101
Algo	…	640	C101
…	…	…	…

Salle

Bâtiment	Salle	Type
640	C101	TP
620	A101	Cours
640	E213	TP
…	…	

Du diagramme des classes au modèle relationnel

Principe

- Le diagramme des classes a deux types de structures (entité/association)
- Le modèle relationnel n'a qu'un type de structure (la relation, ou table)
- On veut passer de l'un à l'autre :
  - Sans perdre ni ajouter d'information
  - En conservant la cohérence sémantique de l'ensemble

Classe

- Classe → Relation
  - Même nom
  - Mêmes attributs
  - Même clé

Enseignant
id : Integer
nom : String
prenom : String
date_de_naissance : String

→ Enseignant(id, nom, prenom, date_de_naissance)

Classe

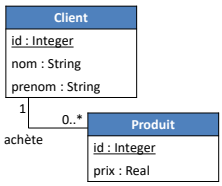
- On peut éventuellement ajouter un identifiant unique pour simplifier la gestion de la future base

Episode
serie : String
saison : Integer
episode : Integer
titre : String

→ Episode(id, serie, saison, episode, titre)

*Association 1:**

- Deux classes liées par une association 1:0..* ou 1:1..* →
  - La relation '*' prend un attribut supplémentaire
  - La clé de la relation '1' devient clé étrangère dans la relation '*'



→

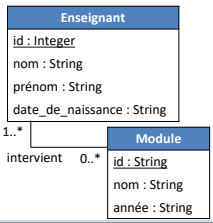
Client(id, nom, prenom)

Produit(id, **client_id**, prix)

(Pour chaque produit, un seul client l'achète)

*Association *:**

- Deux classes liées par une association *x..*:y..** →
  - Nouvelle relation du même nom que l'association
  - La clé primaire de la nouvelle relation est la combinaison des clés des deux relations correspondant aux classes (également des clés étrangères)



→

Enseignant(id, nom, prénom)

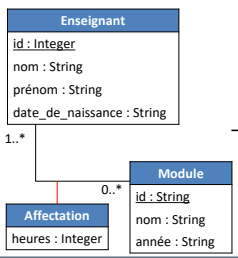
Module(id, nom, année)

Affectation(ens_id, mod_id)

(Un enseignant intervient dans plusieurs modules, un module peut avoir plusieurs enseignants)

*Association *:**

- Deux classes liées par une association *x..*:y..** →
  - Une « classe-association » compose les attributs de la nouvelle relation



→

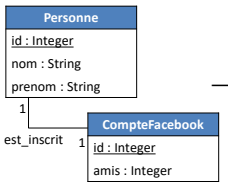
Enseignant(id, nom, prénom)

Module(id, nom, année)

Affectation(ens_id, mod_id, **heures**)

Association 1:1

- Deux classes liées par une association 1:1 →
  1. Fusion des deux classes dans une relation
     - Renommage des attributs si nécessaire
     - Choix d'une clé sur les deux

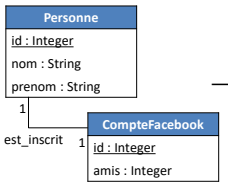


→ Personne(personne_id, personne_nom, personne_prenom, fb_id, fb_amis)

Association 1:1

- Deux classes liées par une association 1:1 →

ou 2. Deux relations
  - Liées par une clé étrangère
  - Mais : « autorise » à ne pas respecter les contraintes de cardinalité

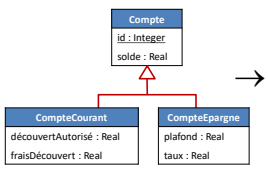


→ Personne(id, nom, prenom)

CompteFacebook(id, amis, **pers_id**)

Héritage

- Toutes les classes sont transformées en relation
- La clé primaire de la classe mère devient clé étrangère de chaque classe fille



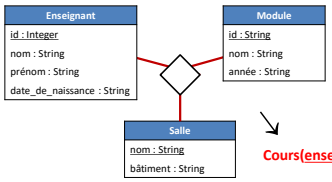
→ Compte(id, solde)

CompteCourant(**compte_id**, découvert, frais)

CompteEpargne(**compte_id**, plafond, taux)

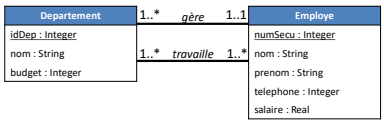
Association n-aire

- Association n-aire : si le cas n'a pas déjà été traité en UML : on crée une nouvelle relation dont la clé primaire est composé de l'ensemble des clés primaires des classes de l'association
- Chacune de ces clés devient également clé étrangère

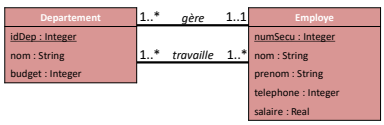


Cours(enseignant_id, module_id, salle_nom)

Exemple

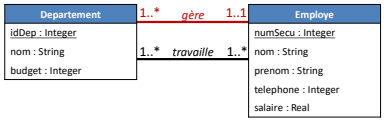


Exemple



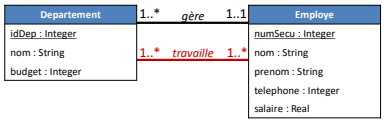
Departement(idDep, nom, budget)
Employe(numSecu, nom, prenom, telephone, salaire)

Exemple



Departement(idDep, nom, budget, **gestionnaire**)
Employe(numSecu, nom, prenom, telephone, salaire)

Exemple



Departement(idDep, nom, budget, gestionnaire)
Employe(numSecu, nom, prenom, telephone, salaire)
Affiliation(numSecuEmploye, idDep)