ensem
filière ISA

inpl
nancy

Bases de Données

Olivier Christmann

christma@loria.fr

bureau 407 rouge

Organisation du module

Volume horaire

- 7 x 2h de cours
- 3 x 2h de TD
- 3 x 3h de TP

⇒ **29 heures** (+ 6h de langage C ✓)

Évaluation

D'après le syllabus : contrôle continu → 2 comptes rendus de TP et 1 contrôle

Agenda

- **Cours** : lundi **30 mars**, **20 avril**, **27 avril** *16h15-18h15*
- **TD n°1** : lundi **4 mai** *16h15-18h15*
- **Cours** : lundi **11 mai** *16h15-18h15*
- **TD n°2** : lundi **18 mai** *16h15-18h15*
- **Cours** : lundi **25 mai** *16h15-18h15* et mercredi **3 juin** *14h00-16h00*
- **TD n°3** : mercredi **10 juin** *14h00-16h00*
- **Cours** : lundi **15 juin** *14h00-16h00*

TP n°1 : 18-20-25 mai (matin)

TP n°2 : 28 mai et 3-4 juin (matin)

TP n°3 : 8-10-11 juin (matin)

Programme

Programme résumé

- modèle entité/association, modèle relationnel, algèbre relationnelle
- langage SQL, SGBD mySQL et JDBC

Programme détaillé

1. Introduction générale : données, bases de données et SGBD
2. Le modèle Entité/Association : entités, attributs et identifiants – associations binaires – avantages et inconvénients du modèle E/A
3. Le modèle relationnel : définition d'un schéma relationnel – passage d'un schéma E/A à un schéma relationnel – le langage de définition de données SQL
4. L'algèbre relationnelle : les opérateurs de l'algèbre relationnelle – expression de requêtes avec l'algèbre
5. Le langage SQL : requêtes simples, requêtes sur plusieurs tables, requêtes imbriquées – agrégation – mise à jour.
6. Schémas relationnels : schémas – contraintes et assertions – vues – triggers.
7. Programmation avec SQL : Interfaçage avec le langage C et Java (JDBC, ODBC).

Premières notions

- Besoin de persistance : notion de fichier et Système de Gestion de Fichiers (SGF)
- Indépendance des niveaux logique et physique

  → SGF assure le lien entre désignation symbolique et désignation interne
- Bases de données : regrouper les données dans un système d'information
- Les données sont une représentation partielle et simplifiée du monde réel

Artiste

Titre de l'album

Titre des morceaux

Durée totale

…

~~Fabricant du boîtier~~

~~Composition chimique du support~~

~~Imprimeur du livret~~

~~…~~

- La base de données est également une partielle et simplifiée du monde réel

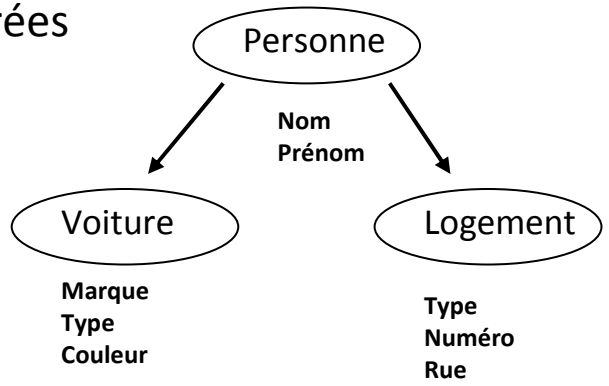
  → relations entre les données
- Cette représentation est obtenue par un processus de **modélisation**

Modèle hiérarchique

Les modèles de données correspondent à la manière de **structurer** l'information dans une base de données

Modèle hiérarchique

- Manipulation de structures arborescentes (nombre quelconque d'éléments en largeur et en profondeur)
- Les nœuds représentent les données et les arêtes représentent les liens entre ces données
- Très forte dépendance entre la description des données et la manière dont elle sont enregistrées



- Conséquence : un "enregistrement" n'a qu'un seul possesseur

  → ce modèle ne peut traduire la réalité de la structuration des données

Modèles réseau et relationnel

Le modèle réseau

- Extension du modèle précédent
- Il permet des liaisons transversales, utilise une structure de graphe
- Les BDD réseau reposent aussi sur des fichiers, reliés entre eux par des pointeurs
- Inconvénients :
  - lourdeur de la gestion des pointeurs
  - hétérogénéité lors de la mise à jour, de l'insertion ou de la suppression de données
  - nombreuses spécifications possibles pour les requêtes

Le modèle relationnel

- Introduit en 1970 par E. F. Codd
- But : créer un langage d'interrogations des BDD plus proche du langage naturel
- Représentation tabulaire des relations
- Formalisation des opérateurs de manipulation regroupés dans une algèbre, l'algèbre relationnelle

Le modèle relationnel

Cote	Titre	Auteur	Editeur
1	Création de bases de données	N. Larrousse	Pearson
2	Automatique : Cours et exercices corrigés	Y. Granjon	Dunod
3	TD d' informatique	Y. Granjon	Dunod

Ouvrage (Cote, Titre, Auteur, Editeur)

- 9 opérateurs : union, intersection, différence, produit cartésien, projecion, sélection, jointure division et agrégation
- Les structures complexes se mettent en œuvre via des jointures
- SQL (*Structured Query Language*) a été mis au point sur ces différents principes
- SQL est devenu un standard de fait
- La généralisation des systèmes relationnels est effective depuis la fin des années 80
- Les principaux systèmes sont : Oracle, Sybase, DB2 (IBM), SQL Serveur (Microsoft)

Modèle orienté objet

- Avec les langages orientés objet (C++, Java, …), le concept objet a été appliqué aux Bases de données
- Avantages :
  - permet de représenter directement un objet réel, quand les structures d'éléments complexes sont réparties entre plusieurs tables dans le modèle relationnel
  - mieux adapté pour modéliser des volumes de texte importants ou des données multimédia (son, images, vidéos)
  - manipulation plus simple lorsqu'on utilise un langage orienté objet
- Fonctionnalités BDD et OO :
  - sécurité, confidentialité, gestion de l'espace disque
  - héritage, gestion des versions, classes
- Mais faible diffusion $\rightarrow$ robustesse et popularité du modèle relationnel et problèmes de performance

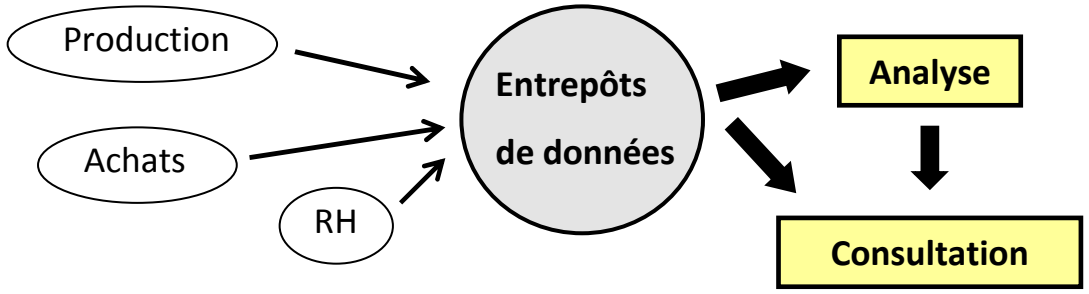
Bases de données réparties

- Les BDD réparties sont apparues avec le fort déploiement des réseaux et l'augmentation des débits de transfert
- Les données sont réparties sur plusieurs sites géographiques : cela facilite la décentralisation des organisations
- L'ensemble est manipulé comme s'il s'agissait d'une seule base de données

  → Gestion transparente aux utilisateurs
- Objectifs :
  - transparence de la localisation : schéma global, schéma de fragmentation (association fragments/sites) et schéma de localisation (interconnexion entre sites)
  - décomposition automatique des requêtes globales
  - recomposition des réponses aux sous-requêtes
  - performances : réplication
  - cohérence des données réparties
- Mais techniques de sécurisation coûteuses

Entrepôts de données

- En anglais *datawharehouse*
- Les entrepôts de données sont des bases de données récapitulatives, à partir de différentes sources de l'entreprise (comptabilité, ventes, achats, …)
- Elles permettent un accès homogène à l'ensemble des données : masquage de l'hétérogénéité des sources
- **But :** faciliter l'analyse décisionnelle en obtenant des indicateurs de gestion par individus ou par métiers



- Modélisation multidimensionnelle : différentes dimensions + aspect temporel (si données "historisées") et différents niveaux d'agrégation
- Mais : mise en œuvre lourde et coûteuse

XML

- e**X**tended **M**arkup **L**anguage : langage de description
- Les moteurs de recherche peuvent extraire le contenu de pages web, mais ne peuvent différencier la nature des informations
- Comme pour un document HTML, un document XML a une structure arborescente

```
- <catalogue>
  - <produit>
      <nom>mirabelle</nom>
      <prix>3,00</prix>
      <poids>0,456</poids>
    </produit>
  - <produit>
      <nom>pomme de terre</nom>
      <prix>2,25</prix>
      <poids>2,5</poids>
    </produit>
</catalogue>
```

→ La présentation des données repose sur une feuille de style

- Pour les BDD : modèles dits "semi-structurés"
- Format d'échange entre SGBD et d'un SGBD vers d'autres logiciels
- Il faut donc que les SGBD supportent des données "semi-structurées"

Généralités (1)

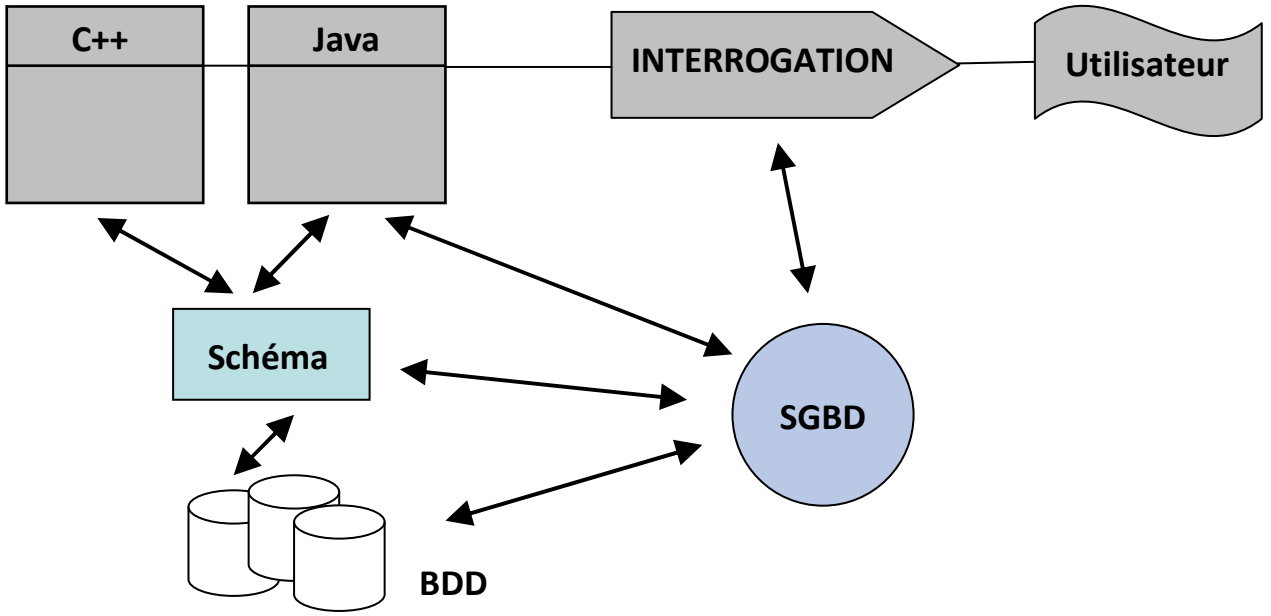
- En anglais *DataBase Management Systems* (**DBMS**)
- Ensemble coordonné de logiciels permettant de décrire, mémoriser, manipuler, interroger les ensembles de données constituant la base
- Modèle en **3 couches** :
  - niveau **physique** : structures de stockage de l'information (dépendance du SGBD employé)
  - niveau **conceptuel** : description des éléments constitutifs de la base
  - niveau **externe** : description des vues des utilisateurs → permet la gestion des droits des utilisateurs
- Les SGBD doivent respecter des principes fondamentaux :
  - centraliser l'information : suppression de la redondance, unicité de la saisie
  - assurer l'indépendance données/traitements
  - intégrité des données : fiabilité et cohérence (liste de valeur ou intervalle, format, cohérence avec d'autres informations)
  - sécurité : reprise après panne, gestion des transactions

Aparté : les transactions

- Transaction = traitement atomique
- Propriétés **ACID** :
  - **A**tomicité : une transaction est exécutée entièrement ou abandonnée
  - **C**ohérence : la transaction se fait d'un état cohérent à un autre état cohérent
  - **I**solement : des transactions simultanées ne doivent pas interférer
  - **D**urabilité : la transaction a des effets permanents, même en cas de panne
- Exemple d'un virement d'un compte bancaire à un autre :
  - Si une opération échoue, l'autre requête ne sera pas exécutée → atomicité
  - Lecture du compte pendant la mise à jour du solde → isolement
  - Modification du solde avec un montant aberrant → cohérence
  - Une fois que le solde est validé, il est persistant → durabilité
- Opérations : commit / rollback
- Tous les systèmes de gestion de bases de données relationnelles (SGBD/R) implémentent nativement ces propriétés

Généralités (2)

- Les SGBD doivent également respecter ces principes :
  - partage des données : transactions + accès concurrent (verrous)
  - confidentialité : identification, authentification, niveau d'accès
- Fonctionnalités supplémentaires :
  - réplication des données : copie automatisée de sauvegarde
  - haute disponibilité des données : duplication pour minimiser la charge des serveurs



Généralités (3)

- Description des données

  → Langage de Description de Données (LDD ou *DDL* en anglais)

  - domaine des données : nombre, chaîne de caractères, date, booléen, …
  - regroupement des données (lien conceptuel) en entités
  - définition des liens entre entités différentes
  - contraintes

  SQL est majoritairement utilisé

- Manipulation des données

  → Langage de Manipulation de Données (LMD ou *DML* en anglais)

  - gestion et utilisation des données : lecture, écriture, effacement, tri, réorganisation
  - modification des données : validation à l'issue d'une transaction

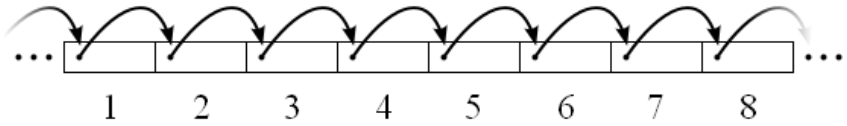
  SQL est majoritairement utilisé

⇒ Une fois les données décrites (LDD), le SGBD doit rendre possible l'**ajout** et la **modification** des données, ainsi que l'**accès** aux données (LMD)

Généralités (4)

Méthodes d'accès aux données

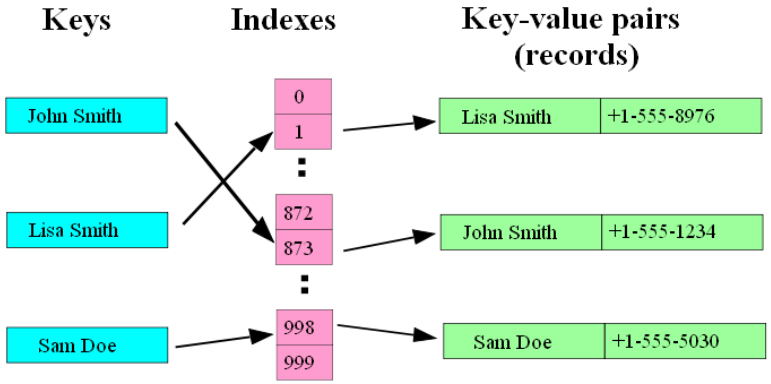
- Accès **séquentiel**
  - l'accès à un enregistrement *n* impose de parcours les *n-1* précédents
  - structure de données typique : liste chaînée



- Tables **descriptives**
  - une alternative au parcours séquentiel est la construction de tables descriptives
  - une table permet l'**accès direct** à un enregistrement par une clé associée à l'adresse de l'enregistrement
  - une autre table contient l'ordre relatif des éléments suivant les valeurs d'un champ

Généralités (5)

- Exemple d'index par hachage
  - association clé-élément par fonction de hachage
  - l'accès à un élément se fait par la transformation de la clé en valeur de hachage : l'élément est à l'index correspondant au hachage



- **Aparté** : MD5
  - fonction de hachage cryptographique (empreinte) sur 128 bits
  - pour vérifier l'intégrité d'un fichier téléchargé, pour crypter un mot de passe
  - peut être « cassé » par des algorithmes de « force brute », par dictionnaire

Voici la signature obtenue sur une phrase :

```
MD5("Wikipedia, l'encyclopedie libre et gratuite") = d6aa97d33d459ea3670056e737c99a3d
```

En modifiant un caractère, la signature change radicalement :

```
MD5("Wikipedia, l'encyclopedie libre et gratuitE") = 5da8aa7126701c9840f99f8e9fa54976
```

Généralités (5)

Quelques SGBD

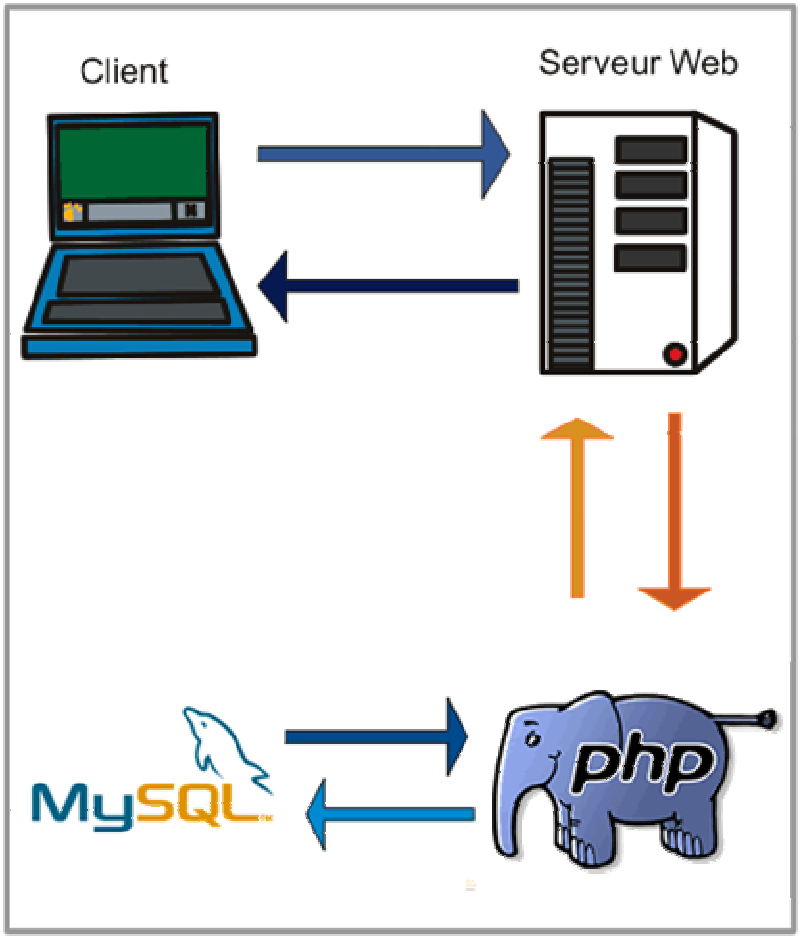
- SGBD à base de logiciels libres
  - **mySQL**
  - Postgre SQL
  - *OpenOffice.org Base*
  - SQLite
- SGBD à base de logiciels propriétaires
  - Sybase (Sybase)
  - *Access (Microsoft)*
  - *FileMaker Pro (FileMaker)*
  - 4D (4D SA)
  - **DB2** (IBM)
  - **Oracle** (Oracle Corporation)
  - **SQL Server** (Microsoft)

mySQL

Présentation

- Système de gestion de bases de données (SGBD, RDBMS en anglais)
- Utilisé dans le monde entier, par le grand public et les professionnels, comme *Oracle* ou *Microsoft SQL Server*
- *MySQL AB* a été acheté en 2008 par *Sun Microsystems*
- serveur de bases de données relationnelles *SQL*
- logiciel libre
- fonctionne sur de nombreux systèmes d'exploitation différents : *FreeBSD*, *Linux*, *Mac OS X*, *Solaris*, *Windows*, …
- Les bases de données sont accessibles via : *C*, *C++*, *Java*, *Perl*, *PHP*, *Python*, *Ruby*, *Tcl*, …
- une *API* spécifique est disponible pour chacun d'entre eux

Exemple d'application (1)



LAMP – WAMP - MAMP

1 - Le navigateur envoie l'adresse que vous avez tapée

2 - Le serveur web cherche dans son arborescence si le fichier existe, et si celui-ci porte une extension reconnue comme une application PHP. Si c'est le cas, le serveur web transmet ce fichier à PHP.

3 - PHP parse le fichier (analyse du code entre les balises *<?php* et *?>*. Si ce code contient des requêtes vers une base de données MySQL, PHP envoie la requête SQL.

4 - La base de données renvoie les informations au script qui peut les exploiter (pour les afficher par exemple).

5 - PHP continue de parser la page, puis retourne le fichier dépourvu de code PHP au serveur web.

6 - Le serveur web renvoie donc un fichier ne contenant plus de PHP, mais seulement du HTML au navigateur, qui l'interprète et l'affiche.

Exemple d'application (2)

En quelques mots

- LAMP = **L**inux, **A**pache, **M**ySQL, **P**HP (**W**AMP = **W**indows…)
- ces différents systèmes n'ont pas été conçus à l'origine pour fonctionner de concert
- leur combinaison est devenue populaire en raison de leur faible coût et de leur intégration par défaut dans les distributions Linux
- tous les composants peuvent être situés :
  - sur une même machine
  - sur 2 machines distinctes : Apache + script et MySQL
  - sur plusieurs machines : répartition de charge
- PHP peut être remplacé par un autre langage de script : *Perl* ou *Python*
- WAMP, architecture souvent utilisée pour faire du développement web sur une machine, la mise en production se faisant sous Linux
- Solutions WAMP courantes :

  WampServer, *XAMPP*, *VertrigoServ*, ***EasyPHP***, *Mov'AMP*,…

Exemple d'application (3)

- Il est donc possible, grâce au PHP, de se connecter à et d'interagir avec une base de données
- Des fonctions spécifiques PHP permettent la connexion à une BdD MySQL.
- Pour utiliser une base de donnée écrite en MySQL avec PHP, il faut :
  - se connecter au serveur MySQL
    - création d'une ressource
    - authentification
    - sélection de la base de données
  - puis utiliser des tables MySQL (via PHP)
    - lecture
    - écriture
    - modification
- La communication avec les BdD se fait au moyen de requêtes SQL

Les Métiers

Consultant/analyste

- Analyse des activités et des flux d'information de l'ensemble à modéliser

Concepteur de la base

- Traduction du modèle précédent en modèle logique exploitable par le SGBD
- Préparation des tables, vues et schémas d'accès

Administrateur

- Responsabilité du fonctionnement général
- Gère les droits d'accès au système

Utilisateurs standard et programmeurs

- Utilisateurs du système d'information
- Accès par l'intermédiaire des vues définies par le concepteur

Introduction

Expression des besoins et formalisme

- A l'issue de la caractérisation des besoins, on dispose d'une description textuelle de la réalité à modéliser : objets, données et liens entre objets
  $\rightarrow$ exemple d'un hôtel
- représenter l'ensemble des données et des liens que les relient

$\Rightarrow$ nécessité d'un formalisme pour constituer ce schéma

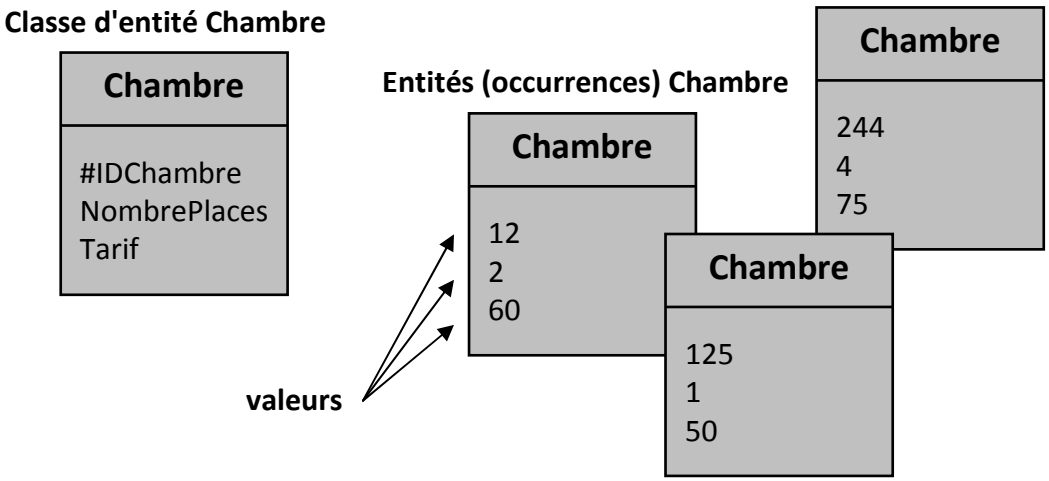
Modèle entité association

- En anglais *entity-relationship*
- Présenté par P. Chen en 1976
- Modèle relationnel $\rightarrow$ modèle sémantique
- Il est également possible d'utiliser UML pour la représentation du schéma
- Composants de base :
  - entité
  - attribut
  - association

Entité, association et attribut (1)

Classe d'entité, attribut et identifiant

- Définition : *personne, objet, lieu, statut, événement qui a une existence dans le monde réel. C'est un objet concret ou abstrait*
- Les ***classes d'entités*** sont composées de champs de données appelés ***attributs***
- Un attribut doit être choisi comme ***identifiant*** pour identifier de manière unique une occurrence de l'entité
- Une classe d'entité est entièrement définie par son nom, son identifiant et ses attributs
- Toute classe d'entité possède au moins un attribut : l'identifiant



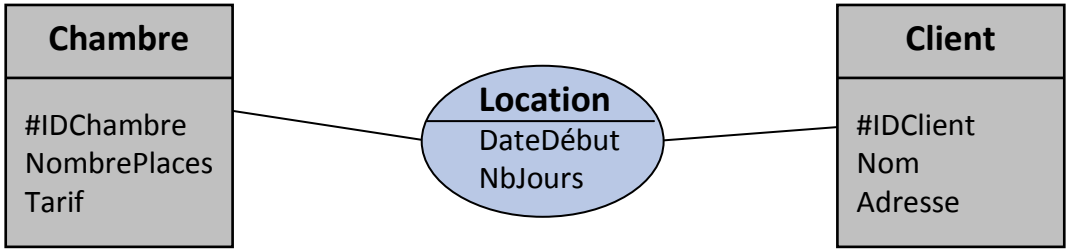
Entité, association et attribut (2)

Association

- Définition : *lien entre les entités. Une association n'a d'existence qu'au travers des entités qu'elle relie*
- Les associations peuvent être composées ***d'attributs***
- Par conséquent, une association n'a pas nécessairement un identifiant
- L'association représente un verbe matérialisant une relation entre entités
- Exemple : Pierre **étudie** la langue anglaise

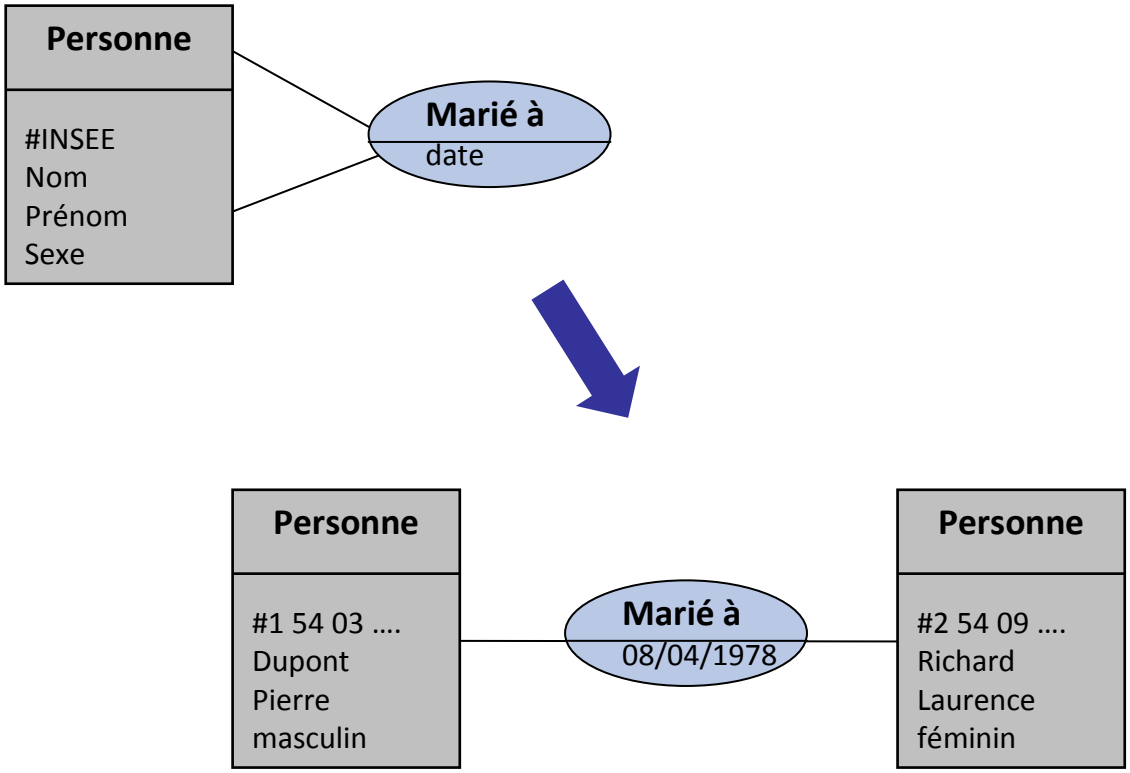
  → association "*étudie*" entre l'entité "*Pierre*" et l'entité "*langue anglaise*"

  De façon générique, association "*étudie*" entre "*étudiant*" et "**matière**"



Réflexivité

- Il est possible de relier par une association une entité à elle-même
- L'association est dite **réflexive**
- Les associations réflexives ne sont pas symétriques !

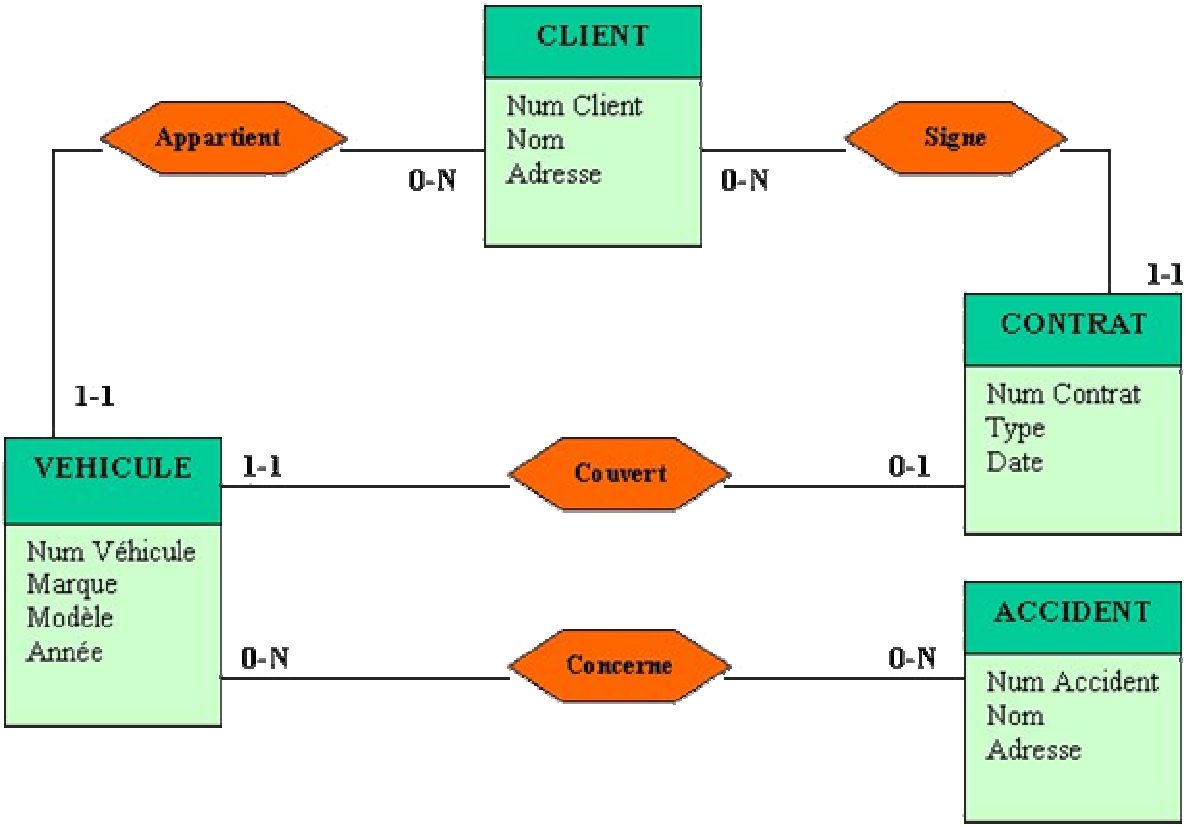


Cardinalité (1)

- Les cardinalités décrivent les caractéristiques de l'association entre les entités
- Le type de liaison est désigné par un couple de valeurs **M-N**
- Ces nombres modélisent le nombre d'occurrences minimales et maximales des entités impliquées dans l'association
- Une association binaire sera donc caractérisée entièrement par 4 nombres
- Les cardinalités peuvent prendre les valeurs suivantes :
  - **1,1** : à une entité de la classe A est associée une entité de la classe B
  - **1,N** : à une entité de la classe A sont associées plusieurs entités de la classe B
  - **0,1** : aucune ou une instance
  - **0,N** : de zéro à plusieurs

	Minimum	Maximum
0	l'entité peut ne pas participer	
1	l'entité participe obligatoirement	l'entité peut participer au plus 1 fois
N		l'entité peut participer plusieurs fois

Cardinalité (2)



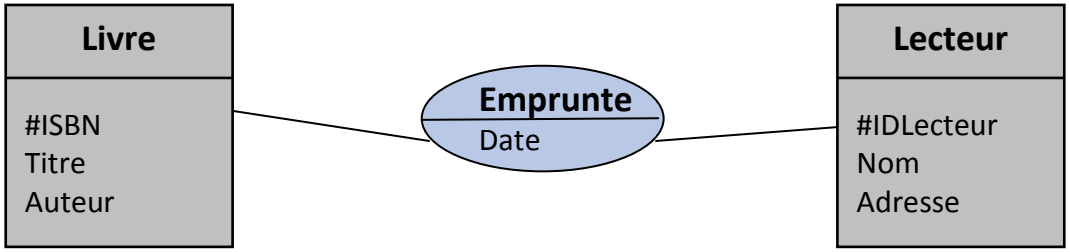
Evolution du modèle (1)

Qualité des attributs

- Ne stocker que les attributs strictement nécessaires
- Ne pas retenir d'attributs qui peuvent être déduits d'autres attributs
  - CA calculé à partir du prix de vente et de la quantité
  - Référence construite à partir du nom de produit et d'un fournisseur (lien sémantique)
- Nom d'attribut explicite

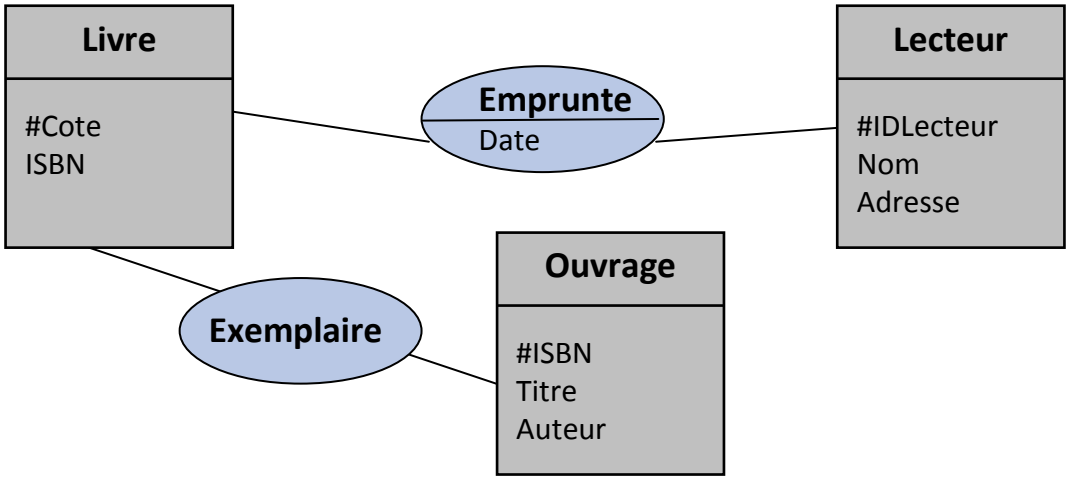
Réorganisation des entités

→ Exemple : "un lecteur emprunte un livre"

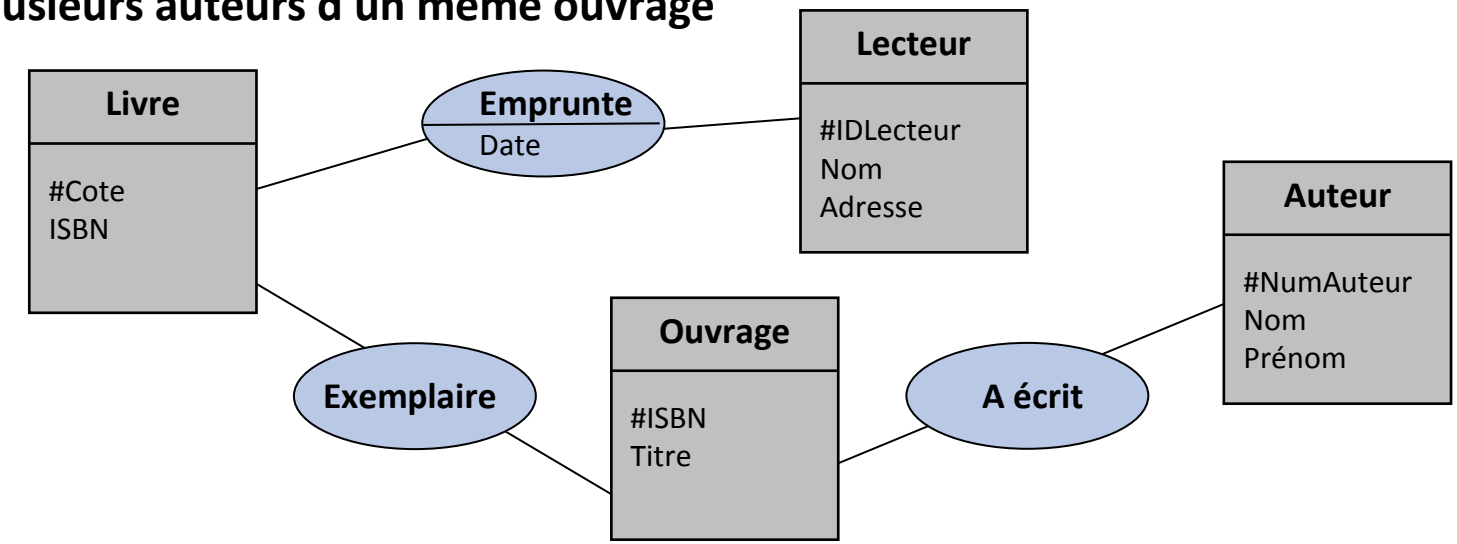


Evolution du modèle (2)

Si "plusieurs exemplaires du même ouvrage"

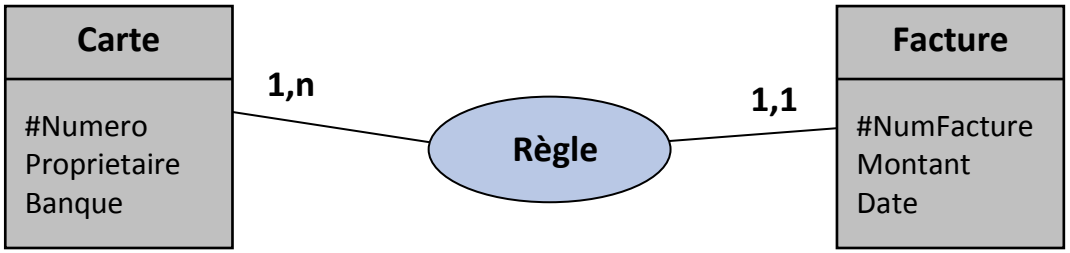


Si "plusieurs auteurs d'un même ouvrage"

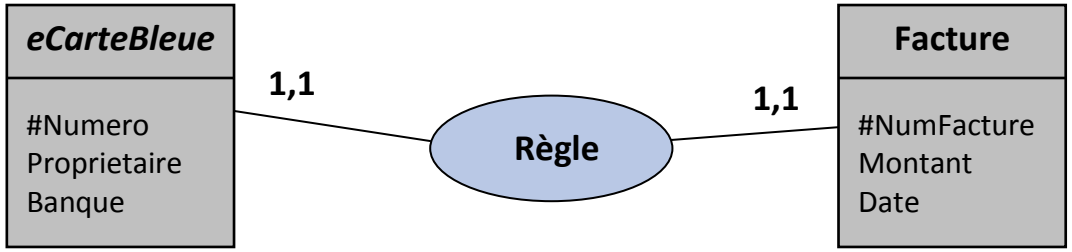


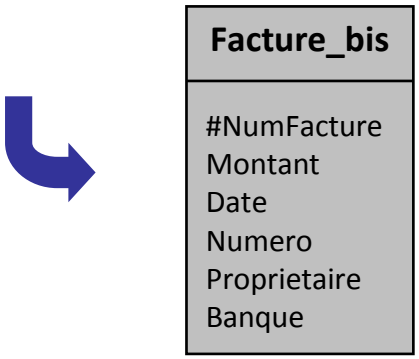
Elimination d'association

⇒ **Fusion d'entité**



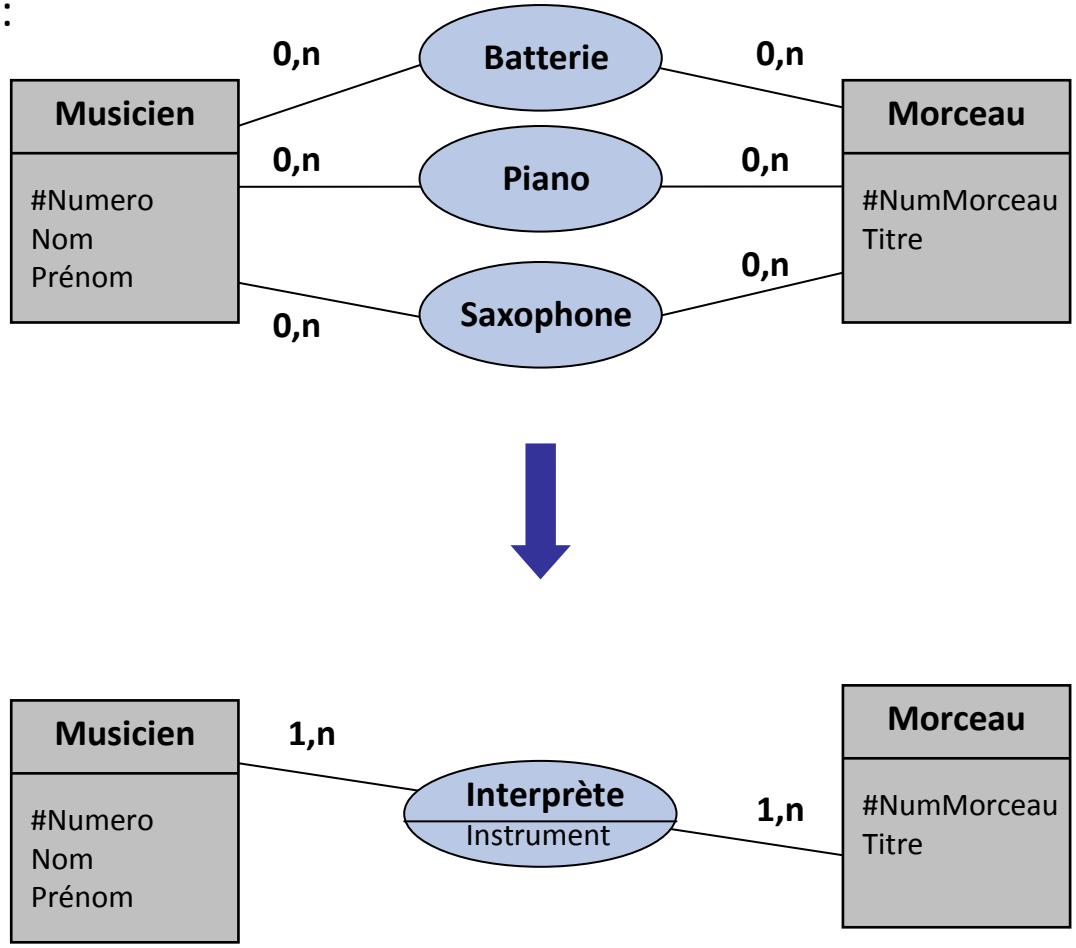
Mais, si on utilise une *eCarteBleue*, le numéro est unique à chaque achat :





Fusion d'associations

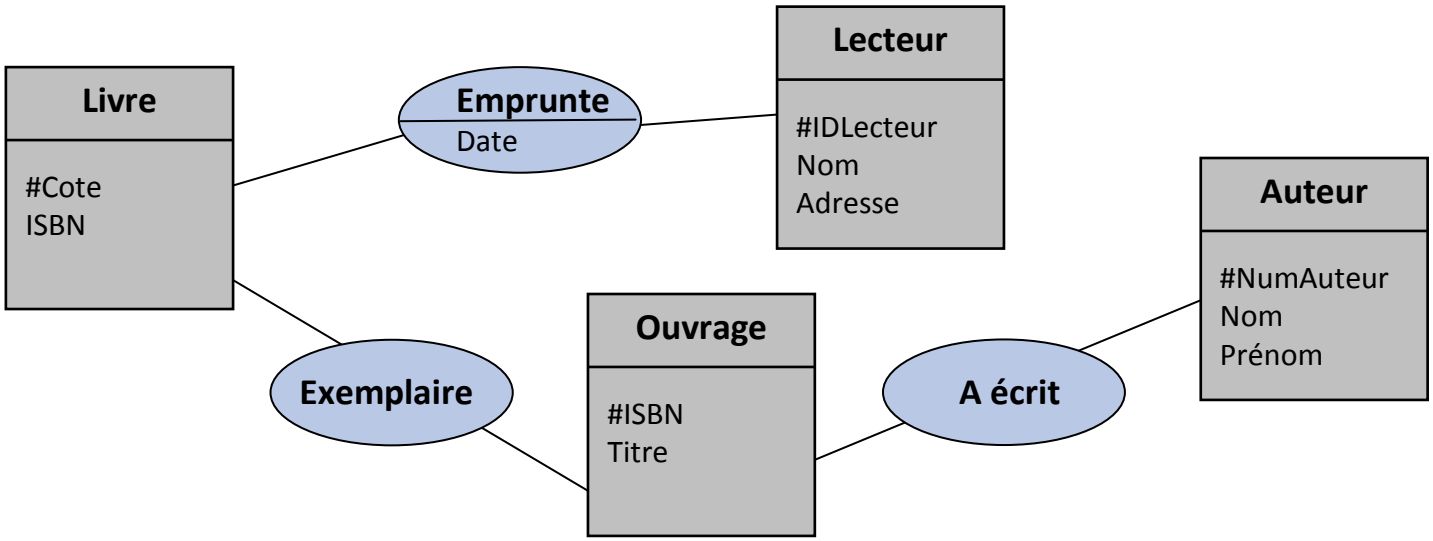
Il est également possible de fusionner plusieurs associations ayant le même rôle sémantique :



Règles de construction d'un schéma

- Déterminer la liste des classes d'entités
- Pour chaque entité :
  - établir la liste des attributs
  - parmi, les attributs, choisir un identifiant
- Déterminer les relations entre les classes d'entités
- Pour chaque relation :
  - dresser la liste des attributs éventuels
  - vérifier la dimension
  - définir les cardinalités
- Vérifier le schéma obtenu
- Valider avec les utilisateurs

Exercice 1 : cardinalités (1)



- Association "**est un exemplaire**"
  - du côté de la classe d'entité **ouvrage**
    - → peut-on avoir la fiche ouvrage sans disposer d'un livre ? **1,n**
    - → un ouvrage peut-il correspondre à plusieurs livres ?
  - du côté de la classe d'entité **livre**
    - → une livre peut-il exister sans fiche ouvrage associée ? **1,1**
    - → une livre peut-il correspondre à plusieurs fiches ouvrage ?

Exercice 1 : cardinalités (2)

- Association "**emprunte**"
  - du côté de la classe d'entité **livre**
    - → un livre peut-il n'avoir jamais été emprunté ?
    - → un livre peut-il avoir été emprunté plusieurs fois ?

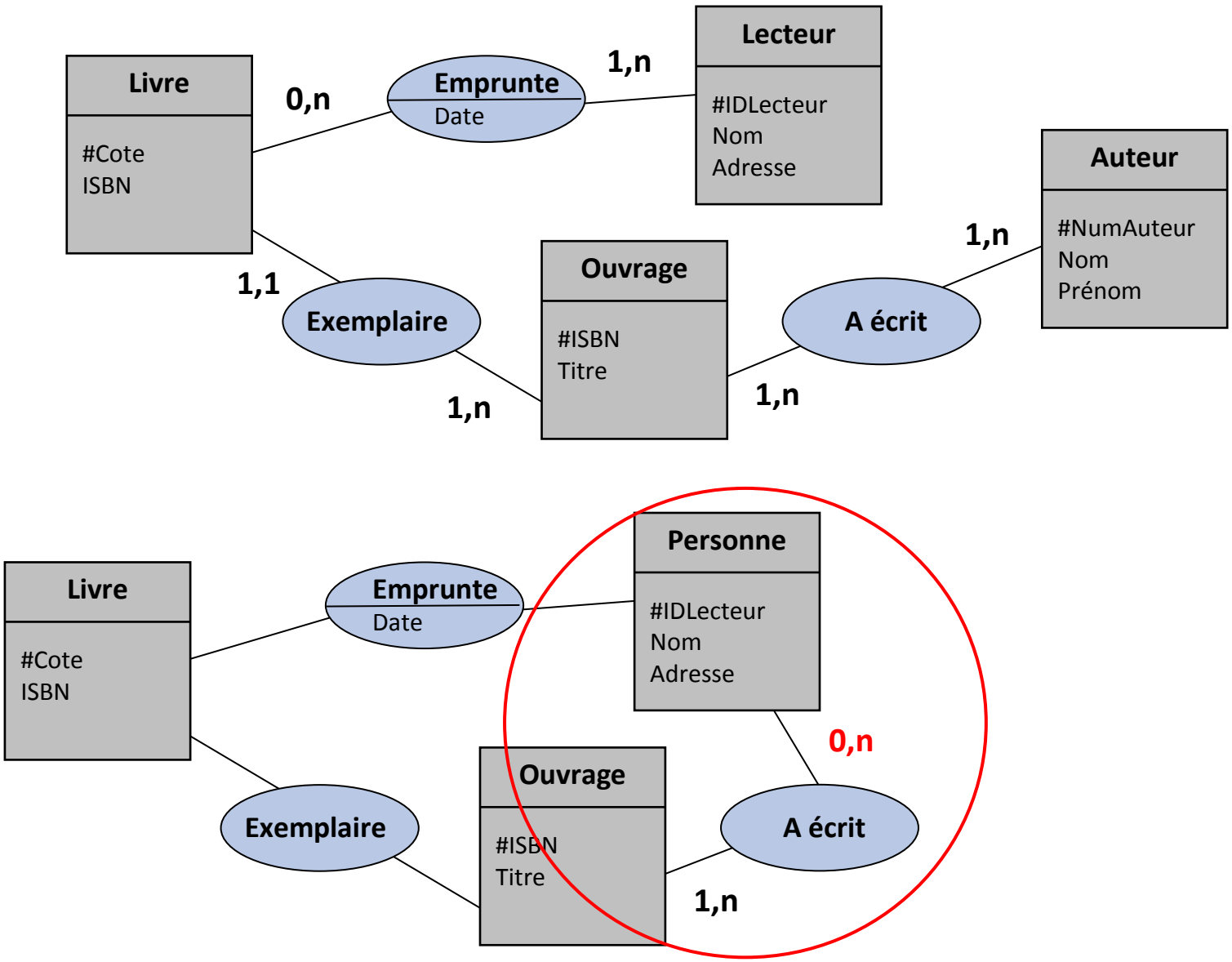
    0,n
  - du côté de la classe d'entité **lecteur**
    - → un lecteur peut-il ne jamais avoir emprunté de livre ?
    - → un lecteur peut-il emprunter plusieurs livres ?

    1,n
- Association "**a écrit**"
  - du côté de la classe d'entité **ouvrage**
    - → un ouvrage peut-il ne pas avoir d'auteur ?
    - → un ouvrage peut-il avoir plusieurs auteurs ?

    1,n
  - du côté de la classe d'entité **auteur**
    - → un auteur peut-il ne pas avoir écrit de livre ?
    - → un auteur peut-il avoir écrit plusieurs livres ?

    1,n

Exercice 1 : cardinalités (3)



Exercice 2 : pays, fleuve, espace maritime (1)

Enoncé général

On souhaite créer une base de données destinée à la gestion des pays, des fleuves et des espaces maritimes

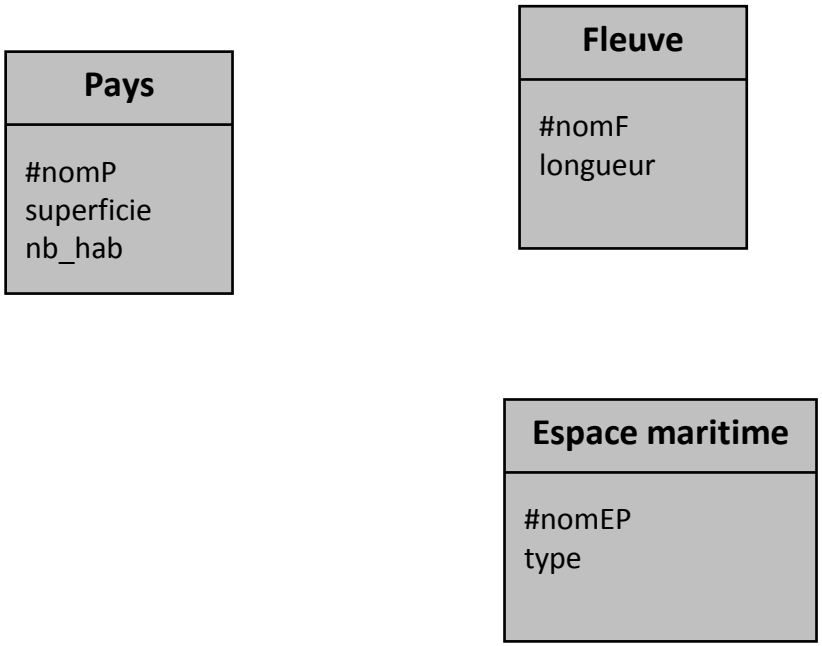
Précisions

- Un pays est connu par un nom, une superficie, un nombre d'habitants, la liste des pays ayant une frontière commune et la liste des fleuves qui le traversent
- Un fleuve est connu par son nom, sa longueur, l'espace maritime dans lequel il se jette, le nom du pays dans lequel il prend sa source, la liste des pays qu'il traverse, la distance parcourue dans chacun des pays
- Un espace maritime est connu par un nom, un type (mer ou océan), la liste des pays qu'il côtoie et la liste des fleuves qui s'y jettent

→ Donnez le schéma conceptuel de la BDD à l'aide d'un diagramme entité-association

Exercice 2 : pays, fleuve, espace maritime (2)

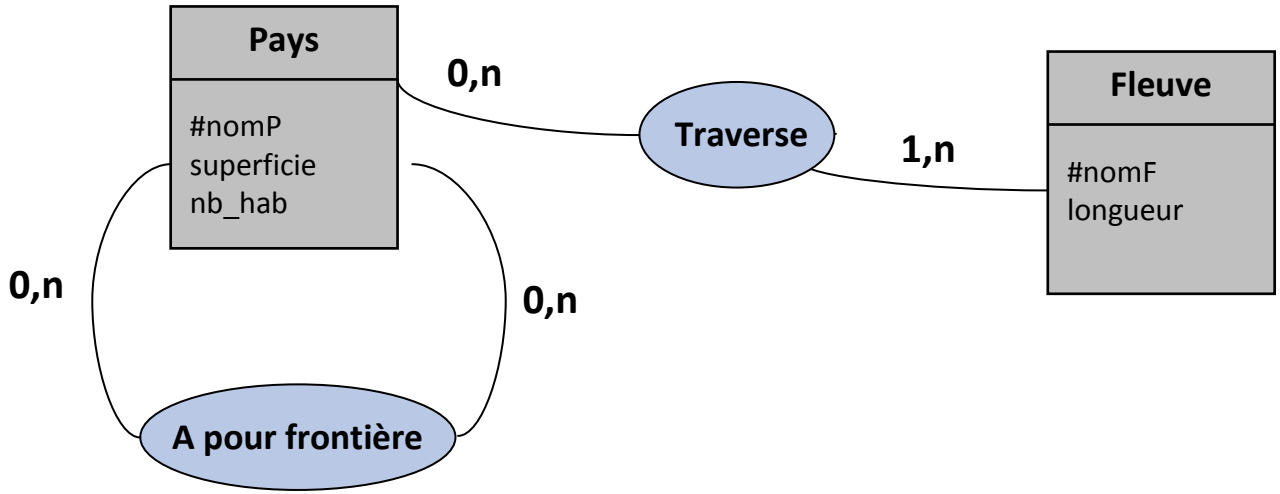
Identification des entités



Exercice 2 : pays, fleuve, espace maritime (3)

Identification des associations : pays

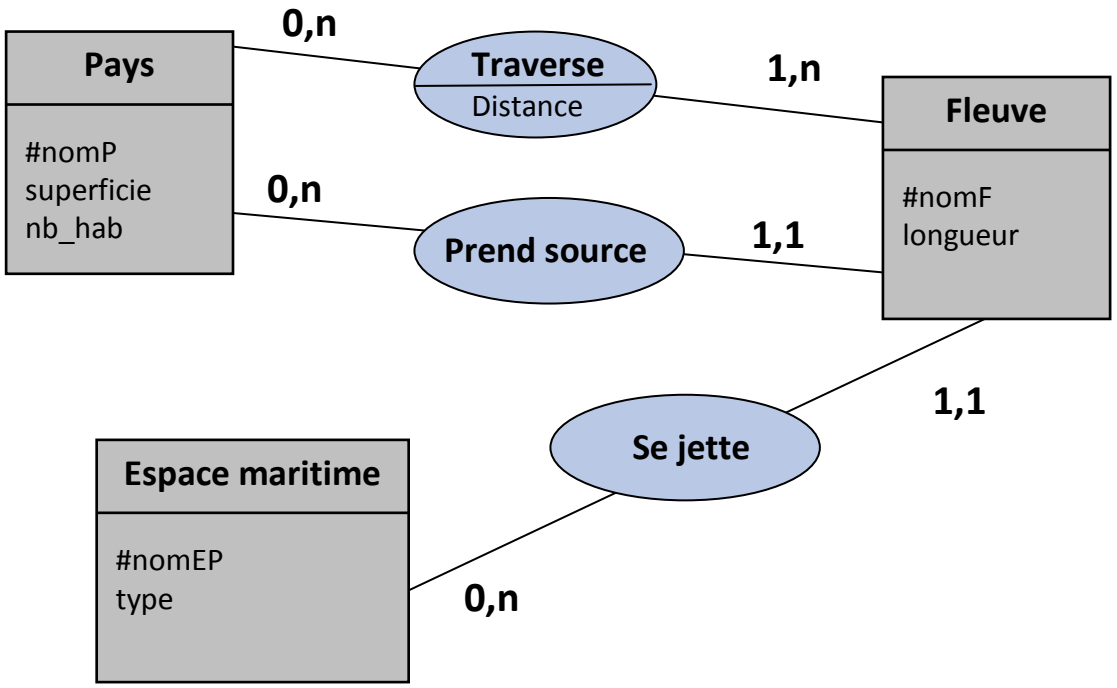
- Un pays est connu par la liste des pays ayant une frontière commune
- Un pays est connu par la liste des fleuves qui le traversent



Exercice 2 : pays, fleuve, espace maritime (4)

Identification des associations : fleuve

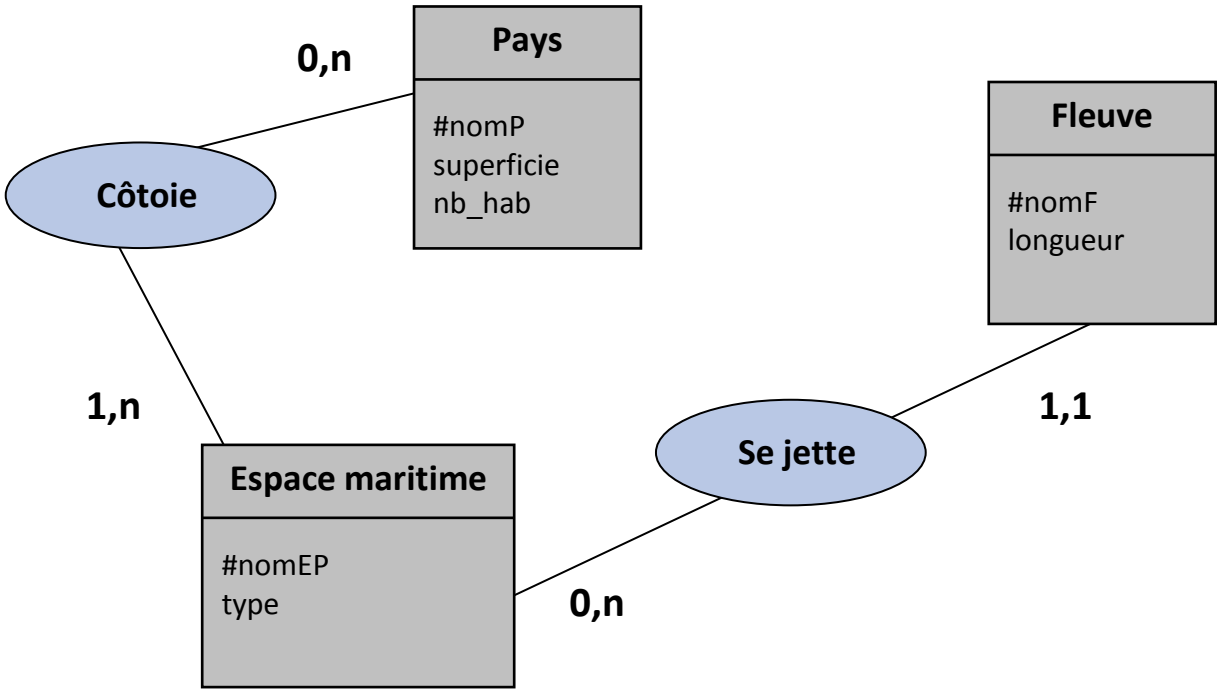
- Espace maritime dans lequel il se jette
- Nom du pays dans lequel il prend sa source
- Liste des pays qu'il traverse
- Distance parcourue dans chacun des pays



Exercice 2 : pays, fleuve, espace maritime (5)

Identification des associations : espace maritime

- Liste des pays qu'il côtoie
- La liste des fleuves qui s'y jettent



Exercice 2 : pays, fleuve, espace maritime (6)

Schéma final