

BDay-MI

Bases de données avancées

Cours de Cristina Sirangelo

IRIF, Université Paris Diderot

Assuré en 2021-2022 par Amélie Gheerbrant

amelie@irif.fr

Infos pratiques

- **Cours** : Amélie Gheerbrant <amelie@irif.fr>
jeudi 14h - 16h
Amphi 5C Halle aux Farines
- **TD/TP**
aux horaires prévus dans l'EDT
 - ▶ Alexandra Rogova <rogova@irif.fr>
 - ▶ Sylvain Schmitz <sylvain.schmitz@irif.fr>
 - ▶ Wieslaw Zielonka <zielonka@irif.fr>

Infos pratiques - suite

- Contrôle des connaissances :
 - ▶ Un projet en binôme
 - pré-soutenances dans le créneau du TP, soutenances en mai
 - ▶ Examen terminal
- Mode de calcul de la note finale: CC intégral
 - ▶ session 1 :
15% modélisation projet + 35% implémentation projet + 50% contrôle terminal
 - ▶ session 2 :
100 % contrôle terminal de session 2
- Page du cours sur moodle : **S'INSCRIRE !**
 - ▶ <https://moodle.u-paris.fr/course/view.php?id=10715>
 - ▶ sujets des TP/projet, soumission des projets, **annonces**
 - ▶ **!!MOYEN PRINCIPAL POUR VOUS CONTACTER !!**

Pré-requis

- Connaissances requises
 - ▶ modèle relationnel des données
 - ▶ modélisation E/R
 - ▶ algèbre relationnelle
 - ▶ SQL
 - DDL (création et modification de tables)
 - syntaxe de base des requêtes
 - requêtes imbriquées
 - requêtes de mise à jour
 - agrégats
 - vues
- Une expérience minimale avec un SGBD (Oracle ou PostgreSQL, ou MySQL, ou autre)
 - ▶ TP de ce cours sur PostgreSQL

Plan du cours

- Introduction
- Conception / développement de systèmes d'information
 - ▶ Modélisation - rappel en TD
 - conceptuelle (modèle E/R)
 - logique (modèle relationnel)
 - ▶ Théorie de la normalisation
 - ▶ SQL avancé et extensions de SQL
 - Triggers et fonctions stockées, Assertions
- Implémentation de systèmes de gestion de bases de données :
 - ▶ Organisation physique des données et indexation
 - ▶ Evaluation et optimisation de requêtes
 - ▶ Gestion des transactions, concurrence et pannes
- ...

Textes conseillés pour ce cours et bibliographie

- *Database Systems Concepts*
par Silberschatz, Korth and Sudarshan, 6eme edition, McGraw-Hill.
- *Database Systems: the Complete Book*
par H. Garcia-Molina, J.Ullman and J.Widom, Prentice Hall.
- *Database Management Systems*
par Raghu Ramakrishnan and Johannes Gehrke. McGraw-Hill.
- Pour aller plus loin en théorie des bases de données :
Foundations of Databases
par S.Abiteboul, R.Hull and V.Vianu, Addison-Wesley, 1995.

Introduction

Sources (quelques slides empruntés et réadaptés) :

- cours pour les classes prépa - B. Nguyen, U. d'Orléans
- cours *database systems principles* - V. Vianu, UCSD

De la difficulté d'interroger les grandes masses de données ...

- Soit un ensemble de données représentant des élèves, les modules qu'ils suivent, et leurs notes.
- On souhaite interroger ces données pour retrouver les notes d'un élève, calculer des moyennes, etc.
 - Il faut modéliser ces données (existe-t-il une méthode générique simple applicable ?)
 - Il faut définir pour chaque opération d'interrogation, un *programme* qui réalise cette opération. On pourra définir des *sous-programmes* pour des tâches à réaliser fréquemment.
- On souhaite rendre les données pérennes :
 - ▶ Il faut les sauvegarder sur un média durable
 - ▶ Il faut gérer les pannes à tout moment
- On souhaite modifier les données
- On souhaite sécuriser l'accès aux données

Exemple : des élèves et leurs notes

- Définir une structure élève complexe qu'on va mettre dans un tableau. En soi c'est déjà compliqué.

```
struct eleve {  
    nom : string;  
    notes : tableau[X] de int;  
           // ou quelque chose de plus compliqué  
}
```

- Calculer la moyenne des notes d'un élève = écrire une fonction

```
float moyenne (eleve e) {  
    float somme = 0;  
    for(int i=0; i<e.notes.length; i++)  
        somme += e.notes[i];  
    return somme/e.notes.length;  
}
```

Exemple : des élèves et leurs notes - suite

- Stocker les données = définir un format de fichier et les procédures permettant de lire ou écrire des données.
- Modifier les données = écrire un programme
- Sécuriser les donnée = écrire (plusieurs) programmes
- Etc.

DEJA SUR CET EXEMPLE CE N'EST PAS SIMPLE !!

Les problèmes des systèmes à base de fichiers

- Format de fichiers non standard (chacun crée le sien)
- Redondance de données (incohérences possibles ou difficiles à gérer)
- Écriture d'un programme spécifique pour chaque interrogation : coûts de développement élevés, coûts de maintenance élevés
- Gestion des pannes à programmer soi même (l'OS ne suffit pas)
- Partage de données difficile
- Sécurisation des données difficile

BREF, TOUT EST À FAIRE SOI-MEME !

La réponse « SGBD » = un « package » qui garantit :

- **Indépendance physique**

- ▶ les applications interagissent avec un modèle abstrait des données (modèle logique) indépendant de son implémentation physique dans des structures de stockage
- ▶ Possibilité de modifier les structures de stockage sans modifier les applications
- ▶ Ecriture des applications par des non-spécialistes des fichiers

- **Indépendance logique**

- ▶ Vues multiples (virtuelles) des données
- ▶ Possibilité d'ignorer une partie de données (les données d'autres applications)
- ▶ Possibilité de protéger (rendre confidentielles) certaines données

- **Manipulation aisée**

- ▶ Par le biais d'un langage déclaratif (SQL) équivalent à la logique du 1^{er} ordre

- ...

La réponse « SGBD » = un « package » comprenant :

- Exécution et optimisation
 - ▶ Les requêtes sont traduites en un langage procédural (algèbre relationnelle) qui peut être optimisé *automatiquement*. (des années de recherche en BD...)
- Intégrité logique (contraintes d'intégrité)
 - ▶ Contrôle sur les données élémentaires et les relations (assertions)
 - ▶ Détection de mises à jour erronées (triggers)
- Intégrité physique (tolérance aux pannes)
 - ▶ gestion des transactions
- Partage de données (gestion de la concurrence)
 - ▶ tout le monde peut agir en même temps, chacun comme s'il était seul
- Confidentialité
- Standardisation
- ...

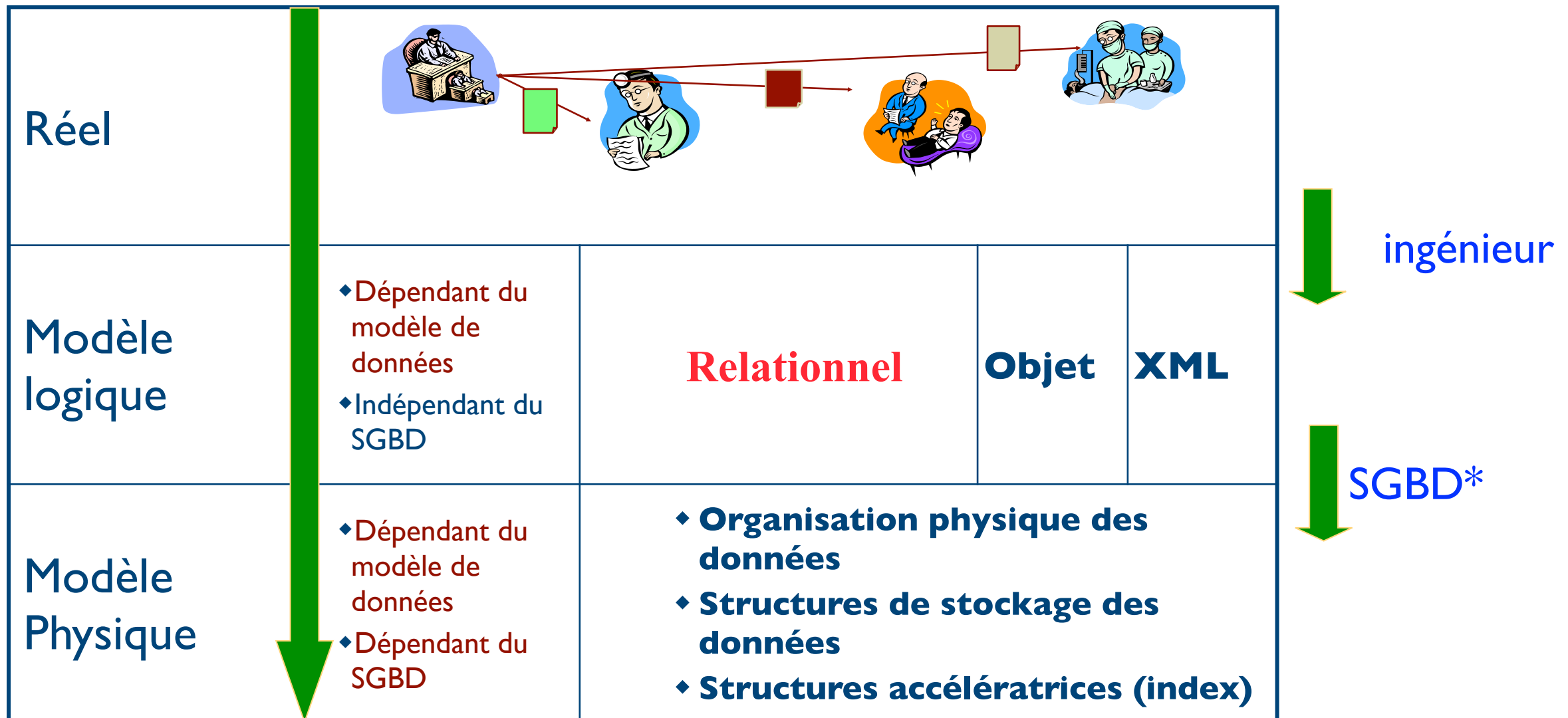
SGBD en résumé

Un **système générique**, qui fournit une couche pour le stockage, organisation physique, et accès (interrogation / manipulation) de données représentées dans un **modèle logique**

- ▶ SGBD **relationnel/ objets/ XML/ graphe/...** :
SGBD fondé sur un modèle logique relationnel/ objets/ XML/ graphe/...

Représentation des données

- La conception d'un modèle logique des données d'intérêt à partir du réel est à la charge de l'ingénieur (phase de **modélisation**)
- Le lien entre le modèle logique et le modèle physique de représentation des données est à la charge du SGBD (séparation des niveaux)



Rappel : modèle relationnel des données

- Une base de données consiste en plusieurs **tables (relations)**
- Chaque table a un nom
- Dans une table, chaque colonne a un nom
- les noms des colonnes d'une table sont appelés **attributs**
- Chaque attribut a un **domaine** associé (i.e. l'ensemble des valeurs possibles pour cet attribut)
- Les données dans chaque table sont un ensemble de **lignes (tuples)**
 - **une ligne** fournit à chaque attribut une valeur de son domaine

Rappel : modèle relationnel des données

Nom de la relation

Attributs

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25

Tuples

Rappel : modèle relationnel des données

- Schéma de relation
 - le nom de la relation
 - l'ensemble des attributs
 - le domaine de chaque attribut
 - des contraintes d'intégrité qui doivent être respectées

Dans l'exemple, le schéma de la relation est :

STUDENT	(Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)
	Strings Strings 10-digits Strings 10-digits integers reals

Contrainte : deux étudiants différents ne peuvent pas avoir le même SSN (clef)

- Instance de relation. Le contenu actuel de la relation : un ensemble de lignes (tuples) sur les attributs, dont les valeurs sont prises dans les domaines des attributs, et qui respectent les contraintes d'intégrité

Rappel : modèle relationnel des données

- **Schéma de bd relationnelle** : un ensemble de schémas de relations + contraintes

STUDENT (Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)
Strings Strings 10-digits Strings 10-digits integers reals

COURSE (Id, Title)
integers Strings

EXAM (Student-ssn, Course-Id)
Strings integers

Contraintes :

- deux étudiants différents ne peuvent pas avoir le même SSN (clef)
- deux cours différents ne peuvent pas avoir le même Id (clef)
- Les valeurs de Student-SSN dans EXAM apparaissent aussi dans la relation STUDENT (clef étrangère)
- Les valeurs de Course-Id dans EXAM apparaissent aussi dans la relation COURSE (clef étrangère)

Rappel : modèle relationnel des données

- **Instance de base de données relationnelle** : un ensemble composé d'une instance de chaque relation du schéma, qui satisfait toutes les contraintes

STUDENT

Name	SSN	Phone	Adress	OfficePhone	Age	GPA
Bayer	347294	333	Albyn Pl.	367	18	3.24
Ashly	5784673	466	Queen St.	390	20	3.53
Davidson	4357387	589	Princes St.	678	25	3.25

COURSE

Course-Id	Title
12	CS
34	DB

EXAM

Student-SSN	Course-Id
347294	12
5784673	12
4357387	34
347294	34

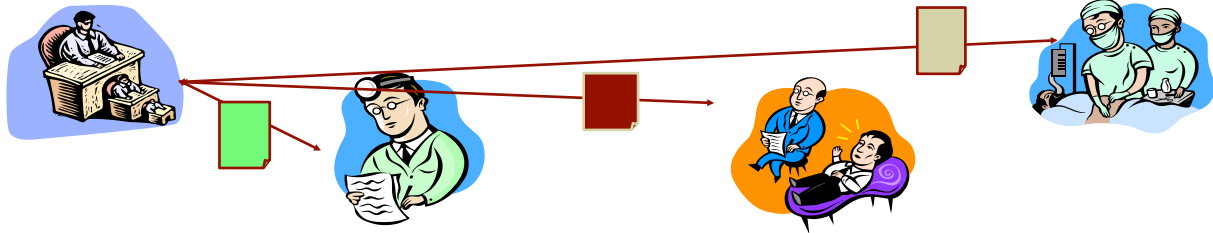
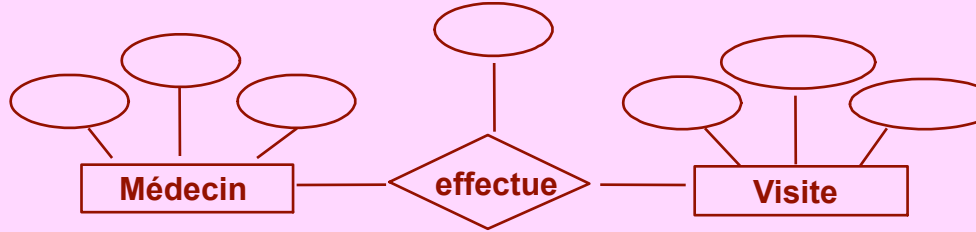
Modélisation : conception du modèle logique

- Comment choisir le modèle relationnel le plus approprié pour représenter les données d'intérêt ?
 - ▶ Approche “*brute force*” :
 - identifier les attributs qu'on veut représenter
 - (*décomposition*) les répartir dans plusieurs relations pour garantir des “bonnes propriétés” du schéma (voir plus loin pour “bonnes propriétés” - formes normales)

Modélisation : conception du modèle logique

- Comment choisir le modèle relationnel le plus approprié pour représenter les données d'intérêt ?
 - ▶ Approche “haut-niveau” (dite conceptuelle)
 - (modélisation conceptuelle) utiliser un modèle de données plus abstrait que le modèle relationnel
 - E/R, UML, ODL... : notions abstraites d'entités, relations, classes, etc. pour représenter les données d'intérêt
 - traduire le résultat de la modélisation conceptuelle dans le modèle relationnel (traduction entre modèles, souvent automatisable)
 - Si on a bien fait la modélisation conceptuelle, le schéma relationnel produit a souvent des “bonnes propriétés”, mais cela n'est pas garanti
 - Si ce n'est pas le cas, raffiner le modèle relationnel obtenu (ultérieure décomposition) ou bien revenir sur la modélisation conceptuelle

Modélisation à plusieurs niveaux

Réel				
Modèle conceptuel	<ul style="list-style-type: none">♦ Indépendant du modèle de données♦ Indépendant du SGBD			
Modèle logique	<ul style="list-style-type: none">♦ Dépendant du modèle de données♦ Indépendant du SGBD	Relationnel	Objet	XML
Modèle Physique	<ul style="list-style-type: none">♦ Dépendant du modèle de données♦ Dépendant du SGBD	<ul style="list-style-type: none">♦ Organisation physique des données♦ Structures de stockage des données		