

Informatique et Sciences du Numérique, 2016-2017

Bases de Données

Gilles Subrenat¹

¹Laboratoire XLIM-SIC
Gilles.Subrenat@univ-poitiers.fr

Journées de formation 2016-2017, 8 février 2017

Plan général : A - Bases de Données

- 1 Introduction
- 2 Conception
- 3 Exploitation
- 4 Optimisation
- 5 Environnement
- 6 BD réparties
- 7 Sécurité (web)

Plan → A - Bases de Données

1 Introduction

- Présentation
- Exemples et caractéristiques

2 Conception

3 Exploitation

4 Optimisation

5 Environnement

6 BD réparties

7 Sécurité (web)

- Manipulations sur une base de données
 - Stocker des données
 - Retrouver des données
 - Modifier des données
 - Effacer des données

⇒ CRUD (Create, Read, Update, Delete)
- Contraintes
 - cohérence (données correctes)
 - optimisation (rapidité)
 - universalité (langage haut niveau)
 - robustesse (gestion des dysfonctionnements)
 - sécurité (piratage)

• Exemples

- mes DVD
- les favoris de mon navigateur
- mon blog sur internet
- les formations, les emplois du temps, ...
- les comptes d'une banque
- les cartes d'un calculateur d'itinéraires
- les horaires des trains
- les comptes d'un réseau social
- l'archivage des pages d'un moteur de recherche
- big data : ensemble des heures de départ/arrivée des trains sur un an
- ...

• Caractéristiques

- confidentialité
- volume
- temps de traitement

Plan → A - Bases de Données

1 Introduction

2 Conception

- Schéma relationnel : entités - associations
- Tables
- (In)Cohérences

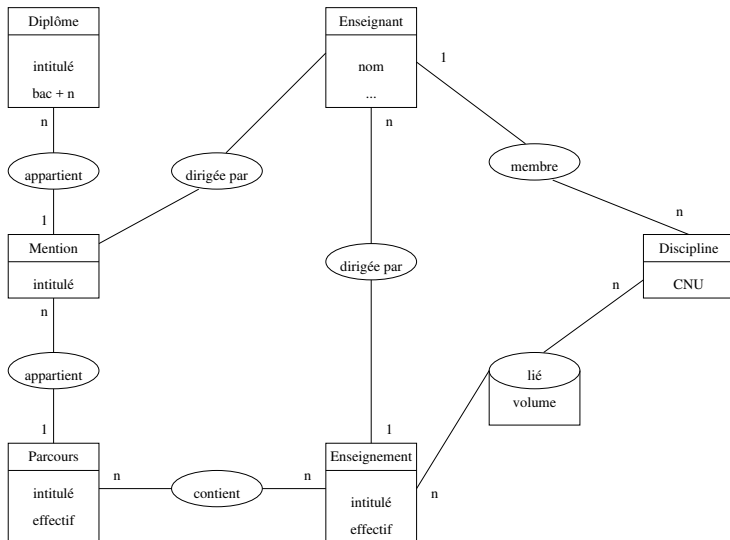
3 Exploitation

4 Optimisation

5 Environnement

6 BD réparties

- Identifier
 - les différents **regroupements de données**
 - les **relations** entre eux
 - les **cardinalités**
- But
 - **spécifier**, lier les informations
 - discuter avec un **client** non informaticien
- Exemple : formations universitaires
 - diplômes, mentions, parcours
 - unités d'enseignements
 - enseignants
 - disciplines
 - ...
→ cf. schéma



- Définition

- Enregistrements (lignes) et propriétés (colonnes)
- clé primaire
- types des données

→ exemple : mentions

- Dépendances entres tables

- clés étrangères

→ exemple : mentions/diplômes

- Cardinalité n-n

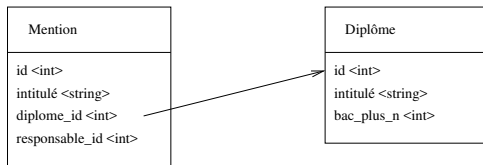
- table supplémentaire

→ exemple : enseignements/disciplines

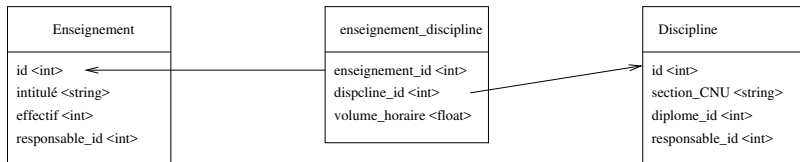
- Schéma physique

→ cf. document externe

- Clé étrangère



- Relation n-n



- Redondance

- cas 1 : dupliquer une donnée (ex : éviter une recherche sur une table distante)
- cas 2 : stocker un calcul (ex : solde d'un compte)
- intérêt : gain de temps en lecture
- inconvénient : perte de temps de mise à jour, risque d'incohérence

- Clés étrangères incohérentes (erreur 404 !)

- clé étrangère pointe une donnée disparue
- solution : gestion par le moteur (ex InnoDB)
- intérêt : aucun risque d'incohérence
- inconvénient : moteur moins performant

Plan → A - Bases de Données

1 Introduction

2 Conception

3 Exploitation

- Langage SQL
- Procédures stockées

4 Optimisation

5 Environnement

6 BD réparties

7 Sécurité (web)

- **Requête** en langage “universel”
 - théoriquement indépendante du SGBD (Oracle, MySQL, ...)
- Construction d’une requête :
 - **SELECT ... FROM** : choix des colonnes et des tables
 - jointure (WHERE ou **JOIN**) : définir les liens entre les tables (clés étrangères)
 - filtres (**WHERE**) : supprimer les cas non désirés
 - tri (ORDER BY) : pour préciser l’ordre d’affichage
 - ...
- Exemple

```
1  SELECT
2      personnes.nom, personnes.age, villes.nom
3  FROM
4      personnes, villes
5  WHERE
6      personnes.ville_id = villes.id -- jointure
7      AND villes.population > 1000   -- filtres
8      AND personnes.age > 30
9  ORDER BY
10     villes.nom,
11     personnes.nom
```

- Définition : **code SQL stocké** dans base de données
- Exécuté sur **événement**
 - sur la mise à jour d'un enregistrement
 - sur un effacement
 - ...
- Exemple
 - **contraintes** (ex : un âge est positif)
 - **redondance** : un changement dans une table est automatiquement répercuté sur la donnée redondante
 - **champs automatiques** (date de dernière modification, nombre de modifications, ...)

Plan → A - Bases de Données

1 Introduction

2 Conception

3 Exploitation

4 **Optimisation**

- Index
- Structure interne

5 Environnement

6 BD réparties

7 Sécurité (web)

- **Tri** sur une **colonne** (nom, âge, ...)
- Avantages
 - **recherche** (select) **rapide** : complexité logarithmique
- Inconvénients
 - **perte de place**
 - **modifications** de la table **ralenties**
- Exemple sur la table "personnes"
 - un index sur le nom
 - un index sur l'âge

id	nom	âge		rang nom	rang âge	index nom ordre id	index âge ordre id
1	Martin	98		2	4	1 5	1 8
5	Dupont	12		1	2	2 1	2 5
6	Quatorze	35		4	3	3 8	3 6
8	Parker	3		3	1	4 6	4 1
...	...	...				...	...

- Utilisation de **structures de données** (et d'algorithmes) **complexes**
 - tables de hashage
 - arbres : binaires, n-aires
 - B-arbre
 - combinaisons !
- c'est un métier

Plan → A - Bases de Données

1 Introduction

2 Conception

3 Exploitation

4 Optimisation

5 Environnement

- Muti-utilisateurs
- Undo (roll back)
- Tolérance aux pannes

6 BD réparties

- Accès concurrents
- Conflit d'une seule requête
 - géré par le SGBD
 - transparent pour le programmeur
- Ensemble de requêtes inter-dépendantes
 - notion de verrouillage de tables
 - géré par l'utilisateur qui verrouille les tables
 - géré par le SGBD qui bloque les accès à ces tables pour les autres utilisateurs
- Exemple : mise à jour d'un solde bancaire
 - lecture du solde existant dans la table
 - calcul du nouveau solde
 - modification de la table avec le nouveau solde

- **Architecture logicielle complexe**

- installation d'un **serveur**
- petit système d'exploitation
- lourdeur de fonctionnement
- exemples : Oracle, MySQL/MariaDB, Postgres, ...

- **Et si **mono-utilisateur** ?**

- architecture simplifiée
- gain de performance
- exemple : SQLITE
- exemple de clients : Firefox, gestion des contacts d'un mobile, ...

- Pouvoir **annuler une opération** ... ou plusieurs !
- Solution : garder une sauvegarde du ou des états antérieurs de la base de données
- Coût
 - éventuellement prohibitif (exemple : vider une table)
 - restrictions : undo à un niveau, quelques opérations non réversibles
- Autre solution
 - à la demande
 - début commandes réversibles
 - quelques requêtes
 - éventuellement décision d'annuler
 - quelques requêtes
 - fin commandes réversibles
- Solution connexes
 - dumps réguliers de la base (i.e. gestion des sauvegardes)

- Que faire si le **serveur plante au milieu d'une requête** ?
 - une seule requête, exemple : mise à jour d'un enregistrement
 - ensemble de requêtes, exemple : mise à jour du solde bancaire
- Solution : système de **journalisation**
 - les différentes étapes d'avancement sont consignées dans un journal
 - les modifications sont effectuées dans une table clone
 - le résultat final est recopié dans la table "officielle"
 - requêtes multiples liées : cf. undo
 - (note : les partitions d'un disque peuvent être journalisées)
- Reprise après panne
 - rejouer le scénario journalisé
- Défaillance d'un disque
 - redondance de données (RAID)
 - sauvegardes incrémentale, totale.

Plan → A - Bases de Données

- 1 Introduction
- 2 Conception
- 3 Exploitation
- 4 Optimisation
- 5 Environnement
- 6 BD réparties**
- 7 Sécurité (web)

- Principe
 - les **tables** sont réparties sur **plusieurs machines**
 - Pourquoi ?
 - trop grande **masse de données**
 - volonté de **traitement parallèle**
 - exemple : référencements de Google
 - Inconvénients
 - SGBD très complexe
 - comment gérer la cohérence des données, les pannes, ... ?
 - **Stratégie de répartition**
 - 1) une table par machine ?
 - 2) chaque table est fragmentée sur plusieurs machines ?
 - Choix
 - solution 2
 - intérêt : interrogation d'une table traitée en parallèle
 - difficulté : agglomérer toutes les réponses partielles
- stratégie d'Amazon pour répartir les livres, disques durs multi-plateaux

Plan → A - Bases de Données

1 Introduction

2 Conception

3 Exploitation

4 Optimisation

5 Environnement

6 BD réparties

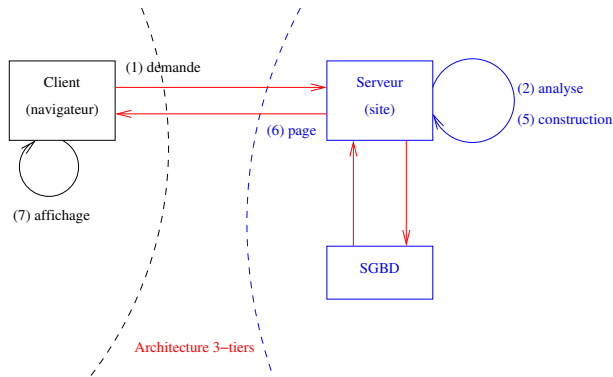
7 Sécurité (web)

- Modèle clients-serveurs

- Intrusions

- Architecture n-tiers

- les services sont répartis sur plusieurs machines



- Serveur de la base attaquable isolément

- **Empêcher les intrusions**

- fermer à clé, ne pas noter le mot de passe sur un post-it, ...
- pare-feu
- mise à jour des logiciels/systèmes
- injection SQL (ou autre : javascript, ...)
- protocole https
- limiter l'accès au serveur à des machines identifiées

- **Limiter les dégâts**

- mots de passe stockés en crypté
- ne pas stocker les numéros de CB
- ... ou alors sur un second serveur protégé (notion de double pare-feu)