Bases de données

Luc Segoufin
INRIA

luc.segoufin@inria.fr

http ://www-rocq.inria.fr/~segoufin

INTRODUCTION

DÉBUT DE L'INFORMATIQUE :

systèmes pour calculer (e.g. équations différentielles, inversion de matrices ...)

TENDANCE ACTUELLE :

gestion de grosses quantités d'information (e.g. banque, bibliothèque, minitel, secrétariat, google)

PROBLÈME : comment stocker et manipuler ces données ?

DONNÉES $\longrightarrow$ BASE DE DONNÉES

LOGICIEL $\longrightarrow$ SGBD

INTERMÉDIAIRE ENTRE UN ÊTRE HUMAIN ET LES DONNÉES

SGBD

CARACTÉRISTIQUES :

- persistance des données (années)
- taille de ces données (Giga/Tera octets)
- accès diversifiés (utilisateurs, programmes)
- accès simultanés (100 $\longrightarrow$ 10.000 utilisateurs en même temps)

FONCTIONNALITÉS :

- gestion de la concurrence
- gestion des transactions
- reprise sur panne
- maintien de la cohérence de la base
- confidentialité

EXEMPLE D'APPLICATIONS :

- Gestion (paye, stock,...)
- Transactionnel (banque, centrale d'achat, SNCF, Air-France, Amazon...)
- Documentation (bibliothèque,...)
- Annuaire (pages jaunes, blanches...)

But du Cours :

Conception d'un SGBD : les problèmes, les solutions et leur justification

Introduction au modèle relationnel et à SQL

Construction d'un SGBD : comment ça marche

But du TD :

Utilisation d'un SGBD : mise en place, test des fonctionalités, interrogation via SQL etc...

Projet :

Écrire une application qui tourne au dessus d'un SGBD.

Aujourd'hui :

Introduction aux SGBD

Comprendre les principes de bases

Présentation du modèle relationnel

SUPPORTS PHYSIQUES :

- NATURE : disque dur, CDROM, DVD, bandes ...
- ORGANISATION : index, arbre-B, hachage...
- GÉOGRAPHIE : distribuées dans le monde entier via internet

UTILISATION :

- UTILISATEURS : administrateurs, programmeurs, secrétaires, etc...
- SYSTÈMES : via des interfaces (menus etc...)

Diversité ⇒ complexité

Doit marcher pour un non expert

L'utilisateur

- Sait ce qu'il veut,
- ne sait pas forcément ce qu'il faut faire pour l'obtenir,
- ne veut pas savoir comment ça marche,
- veut en faire un minimum pour faire tourner son appli.

Comparez

"Donne moi la liste des noms des X suivant le cours de BD"

Avec

```
Dans le directory /COURS/
ouvre le fichier BD et parcours le ligne à ligne
sachant que chaque ligne est de la forme
    ``identifiant, note''
récupère l'identifiant de l'élève
pour chacun des identifiants faire:
  dans le directory /ELEVE/
  ouvre le fichier PROMO-04
  scan ce fichier jusqu'au bon identifiant
  extrait le nom et prénom

etc...
```

IDÉES PRINCIPALES :

Séparer la définition conceptuelle des données de leur organisation réelle sur le support physique.

Langage de requête déclaratif au dessus du niveau conceptuel

Compilateur pour transformer la requête en un plan d'évaluation sur le support physique

ARCHITECTURE ANSI-SPARC (75) : 3 niveaux

APPROCHE BD

Couche externe	Couche conceptuelle	Couche interne
Environnement graphique	Structure	Fichiers
Ce que voit l'utilisateur	Compilation	Index/table de H
	Optimisation	Optimisation
	Contraintes	Concurrence
		Reprise sur panne
		Mise à jour
Peu de CPU	**Pas de I/O du CPU**	**Tout le I/O + CPU**

Exemple

Couche conceptuelle

Structure

Élèves(Nom,Prénom,Promo,Téléphone,Adresse)

Cours(Intitulé,Salle,Horaire,Professeur)

Suit(Élève,Cours,Note)

Contraintes

#Élève/cours ≤ 30

$0 \leq$ note ≤ 20

Schéma conceptuel = structure + contraintes

Exemple

Couche externe

Vue 1 (ADMIN)	Vue 2 (ÉLÈVES)
Étudiants	Cours
Nom	Intitulé
Prénom	Niveau
Promo	
Téléphone	
Liste des cours suivis	

Exemple

Couche interne

/ELEVE/PROMO-02.txt

Classé par ordre de l'identifiant

Chacune des lignes est de la forme :

identifiant#Nom#Prénom#Téléphone#Adresse

/ELEVE/PROMO-02.ind est un index sur le nom de famille

/ELEVE/PROMO-03.txt

Classé par ordre de l'identifiant

Chacune des lignes est de la forme :

identifiant#Nom#Prénom#Téléphone#Adresse

/ELEVE/PROMO-03.ind est un index sur le nom de famille

/ENSEIGNEMENT/COURS.txt

Classé par ordre de l'identifiant

Chacune des lignes est de la forme :

identifiant#Intitulé#Salle#Horaires#Prof

/ENSEIGNEMENT/SUIT.txt

Classé par ordre de l'identifiant

Chacune des lignes est de la forme :

identifiant-élève#identifiant-cours#note

Interne vs. Conceptuel

EXEMPLE : Un seul fichier étudiant

(identifiant, nom, prénom, adresse, téléphone,promo)

Je veux faire 2 types d'accès : par le nom et par l'identifiant

APPROCHE CLASSIQUE

Je choisis une implantation :

1. un H-code sur les noms
2. l'ensemble des enregistrements trié par identifiant

J'écris mes programmes d'application.

Mauvaises performances : je modifie l'implantation physique :
tri sur le nom, H-code sur l'identifiant

$\Longrightarrow$ IL FAUT RÉÉCRIRE LES APPLICATIONS ! ! !

Interne vs. Conceptuel

APPROCHE BD :

1. Je (utilisateur) décide de ma structure conceptuelle
2. Je choisit une structure physique
3. J'écris les programmes d'application en utilisant uniquement la structure conceptuelle
4. Je compile les programmes en exécutables
5. Je change de structure physique, il suffit de recompiler !

Les performances dépendront de la qualité de la compilation (importance de l'optimisation).

C'EST L'INDÉPENDANCE DES DONNÉES

Interne vs. Conceptuel

L'INDÉPENDANCE DES DONNÉES

INDÉPENDANCE PHYSIQUE :

On peut modifier l'implantation physique sans modifier le niveau conceptuel.

INDÉPENDANCE LOGIQUE :

On peut modifier le niveau conceptuel sans toucher à l'implantation.

On aura donc une description physique de la base (type de fichiers, index, chemins d'accès etc...) et une description conceptuelle pour l'utilisateur.

Externe vs. Conceptuel

Vue du Prof : des cours + pour chaque cours :

– Horaires et salles

– Liste des élèves (nom, prénom, note)

Vue de l'administration (suivi des élèves) : des élèves + pour chaque élève :

– (Nom, prénom,promo,téléphone)

– Liste des cours suivis + note

Vue des Administrateurs d'emploi du temps : des salles ...

Externe vs. Conceptuel

VUES DE LA BASE DE DONNÉES

1. Aucun ne voit tout
2. Aucun ne la voit de la même façon
3. Éventuellement aucun ne donne le même nom aux mêmes choses.

INTÉGRATION DE CES VUES

1. Chaque usager a sa vision du monde
2. L'ensemble de ces vues forme une description unique : le **SCHÉMA CONCEPTUEL**
3. Chacune de ces vues est définie par une fonction SCHÉMA EXTERNE $\longrightarrow$ SCHÉMA CONCEPTUEL

En résumé

Trois couches : Externe, conceptuelle, interne

> L'utilisateur ne voit que la vue de la base qui lui a été attribuée. Celle-ci est définie de manière abstraite par un schéma externe. C'est la couche Externe.
>
> Le contenu de la base est défini de manière abstraite par le schéma conceptuel. C'est la couche conceptuelle .
>
> Le niveau physique contient tous les détails sur les fichiers : leur localisation, leur organisation, la présence d'index ou non, etc.

Chacune des couches peut être modifiée indépendamment des autres.

concrètement : Le modèle relationnel

MODÈLE DE BASES DE DONNÉES :

LDD (langage de définition de données) +

LMD (langage de manipulation de données)

MODÈLE RELATIONNEL (Ted Codd 1970) :

Les données sont structurées en tables

LANGAGES : Algèbre, Calcul, SQL

BASES : Calcul des prédicats du 1er ordre

Langage de définition de données

TABLE : *RELATION*, e.g., *Film*

COLONNES : *ATTRIBUT*, e.g., *Titre*

LIGNES : *TUPLE* (ou n-uplet ou enregistrement)

titre	année	genre
Alien	1979	Science-Fiction
Vertigo	1958	Suspense
Volte-face	1997	Thriller
Pulp-Fiction	1995	Policier

Vocabulaire

DOMAINE (de valeur) : int, string, etc...

ATTRIBUTS : colonne d'une table, associé à un domaine.

SCHÉMA de relation : liste de paires attribut, domaine.

Film(titre :string, année :int, genre :string)

SCHÉMA de base de donnée : liste de schémas de relation.

BD = { Film, Cinéma, Acteur }

TUPLE d'une relation (ou n-uplet ou enregistrement) : une ligne bien typée d'une table.

un tuple de Film $\in$ string$\times$int$\times$string

INSTANCE d'un schéma de relation : un ensemble de tuples bien typés pour cette relation.

INSTANCE d'un schéma : Associe à chaque nom de relation du schéma une instance.

Vue ensembliste

– En théorie

Pas deux lignes identiques

Tous les attributs d'une ligne sont connus

L'ordre des lignes est à priori sans importance

– En pratique (vous verrez en TD)

C'est un peu plus compliqué...

Interrogation des données

En utilisant la couche conceptuelle l'utilisateur spécifie ce qu'il veut : langage de requête déclaratif

Le système doit compiler la requête pour obtenir un exécutable.

PERFORMANCES = QUALITÉ DE LA COMPILATION

$\Rightarrow$ il faut un compilateur efficace

Déclaratif $\rightarrow$ exécutable

Constat : on ne sait pas tout compiler efficacement

Il faut limiter les possibilités de l'utilisateur : SQL

PS : SQL est déjà assez compliqué, il pose déjà de nombreux problèmes NP-complets

Concrètement : Modèle Relationnel
Langage de manipulation de données

- 1989 SQL
- 1992 SQL2, révision majeure de la syntaxe
- 2000 SQL3, rajout d’une dose d’orienté objet

SQL : principe

SQL (Structured Query Language) est le langage de requêtes standard pour les SGBD relationnels

Expression d'une requête par un bloc *SELECT FROM WHERE*

SELECT	<liste des attributs a projeter>
FROM	<liste des relations arguments>
WHERE	<conditions sur un ou plusieurs attributs>

```
SELECT *
FROM Cours
```

En résumé

Trois couches : Externe, conceptuelle, interne

L'utilisateur ne voit que la vue de la base qui lui a été attribuée.

Il spécifie ce qu'il veut via un langage de requête déclaratif .

Une fonction défini le passage schéma externe $\longrightarrow$ schéma conceptuel

Une compilation effectue le passage conceptuel $\longrightarrow$ interne

Chacune des couches peut être modifiée indépendamment des autres.

Les performances dépendent de la qualité de la compilation $\Rightarrow$ restriction des possibilité de l'utilisateur : SQL.

Parcours type d’une requête

Niveau externe

– Formulation dans SQL.

– Traduction dans le niveau conceptuel.

Niveau conceptuel

– Traduction dans une algèbre de plans d’évaluation.

– Pré-optimisation à l’aide de règles et d’heuristiques

Niveau interne

– Optimisation en fonction des index disponibles, statistiques sur la base etc...

– Compilation du plan obtenu

– Évaluation

Qui fait quoi ?

Le constructeur du SGBD

Conçoit l'optimiseur et le compilateur,

Implémente divers index et algorithmes utilisant ces index

L'administrateur de la base

discute avec les différents utilisateurs

conception des différentes vues

intégration de ces vues en un schéma conceptuel

conception du schéma physique

gère l'évolution de la base (nouveaux besoins, utilisateurs)

Outils à sa disposition : langages de définition de schémas (physique, logique, externe)

Historique

GRANDES ÉTAPES

60-70	HIÉRARCHIQUE ET RESEAU
80	SUCCÈS DU RELATIONNEL
	PART CROISSANTE DE MARCHÉ
	EXPLOSION SUR MICRO
90	NOUVELLES APPLIS + Orienté Objet
	RELATIONNELS ÉTENDUS
2000	IMPACT DU WWW

Impact du Web

Protocole d'échange standard

Facilité pour exporter des données

Répartition des données

HTML : manque de structure + mélange données et affichage

XML : plus de structure + feuilles de présentation
→ mieux adapté pour les BD

ARCHITECTURES À LA MODE

Entrepôts de données

Médiateurs

Plan du cours

Le modèle relationnel : bases + SQL

Le modèle relationnel : fondements (algèbre, FO)

Complexités sous-jacentes : de Pspace-complet à PTime

Améliorer les performances :

Niveau physique : index, hachage etc..

Niveau conceptuel : règles de réécriture, optimisation, heuristiques

Gestion des transactions : two phases locking

XML et le WEB

Plan du TD

TD1 : Premiers pas

TD2 : Requêtes

TD3 : contraintes, confidentialité, vues, dictionnaire de données et tables système, redirection des entrées/sorties

TD4 : PL/SQL (triggers, procédures stockées)

TD5 : Accès à une base depuis un programme Java

TD6 : PHP+Oracle

TD7 : Optimisation

TD8 : reprise sur panne, concurrence

TD9 : Révisions

Non traité dans ce cours

- Conception de schéma
- Contraintes d'intégrité
- Reprise sur panne
- BD distribuées
- Orienté objet
- Mises à jour
- Gestion des vues

Biblio

La base : Ullman J.D., *Principles of Database and Knowledge-Base Systems*, 2 volumes, Computer Science Press

La théorie : Abiteboul S, R. Hull et V. Vianu, *Foundations of Databases*, Addison-Wesley

En français : Gardarin G., P. Valduriez, *Les SGBD relationnels*, Eyrolles

Gardarin G., *Bases de Données, objet et relationnel*, Eyrolles

les transparents du cours : http ://www-rocq.inria.fr/~segoufin