LIF4 - Initiation aux Bases de données

E.Coquery

emmanuel.coquery@liris.cnrs.fr

http://liris.cnrs.fr/~ecoquery
→ Enseignement → LIF4

Informations diverses

Cours de réseaux : Olivier Glück

Démarrage des TD / TP
- TD : jeudi 8/10 (Grp B,C,D) ou vendredi 9/10 (Grp A)
- TP : mardi 20/10 (Grp A,D) ou jeudi 22/10 (Grp B,C)

TPs :
- Réseaux
- SQL
- Projet PHP/MySQL
  - Cette année : gestionnaire de suivi de bugs
  - Nécessite une clé USB d'au moins 256 Mo

Plan du cours de bases de données

- Introduction
- Modèle relationnel
- Algèbre relationnelle
- Calcul relationnel
- SQL
- Schémas entités-associations
- PHP
- Optimisation algébrique
- (Requêtes hiérarchiques)

Plan

Données

Un ensemble de données c'est :
- Des objets :
  - un nom, par exemple Emmanuel
  - un cours, par exemple Initiation aux bases de données
  - une date, par exemple 16/02/2007
  - ...
- Mais aussi des des liens ou relations entre ces objets :
  - Emmanuel enseigne le cours "Initiation aux bases de données" le 16/02/2007

Une base de données est une application qui permet de stocker, d'interroger et de mettre à jour un ensemble de données.

Fichiers

On peut utiliser des fichiers pour stocker un ensemble de données :
- Collection d'applications où chaque application définit et gère ses fichiers.
- Un fichier est une suite d'enregistrements contenant des données logiquement liées.
  - Il est possible d'utiliser des bibliothèques dans les différents langages pour simplifier la lecture et l'écriture dans ces fichiers :
    - fichiers d'enregistrement en Pascal
    - "sérialisation" en Java
    - ...
- Nécessite une intégration étroite entre le programme et les fichiers.
  - La manipulation des fichiers est directement intégrée dans le programme.

Exemple

Données sur les étudiants dans une université :

- L'adresse d'un étudiant est utilisée pour ses inscriptions, à la bibliothèque, ...
- Chaque application doit gérer un ensemble de fichiers de données et les maintenir à jour.
- Les formats des fichiers peuvent varier.
- Les mises à jour sont redondantes, donc sources d'erreurs et d'incohérences
  - ex : mise à jour de l'adresse : au service des inscriptions, à la bibliothèque, ...

Inconvénients des fichiers

- Lourdeur d'accès aux données :
  - En pratique, il est nécessaire d'écrire un (gros) morceau de programme pour accéder à une donnée.
  - L'efficacité en termes d'accès aux données (par ex, utilisation d'un index) peut apporter une complexité de programmation supplémentaire.
- Fichiers séparés : Redondance dans la définition et le stockage des données.
- Manque de sécurité : Si tout programmeur peut accéder aux fichiers, il est impossible de garantir la sécurité et l'intégrité des données.
- Pas de contrôle de concurrence : si plusieurs utilisateurs accèdent aux fichiers simultanément, des problèmes de corruption de données peuvent se produire (lecture et écriture simultanées ou deux écritures simultanées).

Bases de données

Objectif : pallier aux insuffisances de la gestion de données directe via des fichiers.

Une base de données c'est un ensemble de données :

- enregistré (sur un support adressable),
- dont la structure ne dépend pas de l'application mais des données à stocker
- cohérent
- de redondance minimale
- accessible de manière concurrente par plusieurs utilisateurs

Qui fait quoi

Le concepteur gère :

- la structuration
- non redondance
- mise en commun et la distribution éventuelle des données

Le Système de Gestion de Bases de Données (SGBD, DBMS en anglais) gère :

- le stockage
- la disponibilité des données
- l'accès
- la concurrence

Le SGBD

SGBD : Ensemble d'outils logiciels permettant la création et l'utilisation de bases de données.
Fonctions d'un SGBD :

- Définition d'une base de donnée :
  - spécification des type des données
  - structuration des données
  - contraintes d'intégrité (de cohérence) sur les données stockées
- interrogations des données
- mise à jour des données
- garantie de l'intégrité des données
- gestion de la concurrence
- gestion de la confidentialité de données
- sécurité

Schéma d'une base de donnée

- Description centrale de la base à travers un Langage de Description de Données (LDD) :
  - organisation des données
  - type des données
  - contraintes d'intégrité
- Unique, commun aux différentes applications
  $\Rightarrow$ ce n'est pas l'application qui guide la structuration mais les données à représenter.

Manipulation des données

- Outils et mécanismes permettant de faire communiquer la base de données et les applications qui en font usage.
- Recherche, création, modification et suppression de données.
- Langage de Manipulation de Données (LMD) :
  - On spécifie ce que l'on veut faire ou obtenir plutôt que comment le faire ou l'obtenir.
- Indépendance données - programmes

Interaction avec le SGBD

- Interpréteur de commandes
- Interface graphique
- Dans une langage de programmation
  - C, C++, Java, Python, PHP, ...
  - via des bibliothèques adéquates pour envoyer les requêtes (écrite en LMD) vers le SGBD.
- Via des environnements dits de quatrième génération, avec aide à la formulation de requête, formulaires, ...

Intégrité des données

- Contraintes d'intégrité, spécifiées dans le schéma de la base
  - préservées par le SGBD
  - possibilités de programmation pour les contraintes complexes
- Sécurité de fonctionnement et reprise
- Stockage :
  - Journalisation des actions (Log)
- Concurrence
  - Verrouillage
- Validation/Annulation (Transactions)

Sécurité en confidentialité

- Mise en commun des données
- Logins et mots de passe
- Privilèges et droit d'accès
- Vues

Architecture typique d'un SGBD

Organisé en 3 couches :

- Couche externe :
  - dialogue avec les utilisateurs
  - vues associées à chaque groupe d'utilisateurs
- Couche interne :
  - stockage de données sur des supports physiques,
  - gestion des structures de mémorisation (fichiers, gestion des index)
- Couche logique :
  - contrôle global et structure globale des données

Plan

Modèle de donnée

Un modèle de donnée définit un mode de représentation de l'information :

- Un mode de représentation des données. (LDD)
- Un mode de représentation des contraintes sur ces données. (LDD)
- Un ensemble d'opérations pour manipuler les données. (LMD)

Il est indépendant de la représentation physique des données, ce qui simplifie :

- l'administration ;
- l'optimisation ;
- l'utilisation.

Le modèle relationnel

Modèle ensembliste :

- Les objets sont simples, atomiques :
  - entier, flottants, chaînes de caractères, dates, ...
- Pas d'objets complexes :
  - Pas de listes, pas de tableaux, pas de structures, ...
- Par contre, on s'autorise les opérations ensemblistes usuelles :
  - Union, intersection, différence
  - Produit cartésien
- On utilise les relations pour représenter et manipuler les données :
  - Vision ensembliste :
    une relation portant sur n ensembles $E_1, \ldots, E_n$ est un sous-ensemble du produit cartésien $E_1 \times \cdots \times E_n$.

Avantages du modèle relationnel

Un modèle fondamentalement simple :

- plus facilement compréhensible ;
- plus facilement optimisable ;

mais assez expressif :

- possibilité de représenter des objets plus complexes

utilisé en pratique depuis les années 80, implémenté dans de nombreux SGBD :

- Oracle, MySQL, PostgresQL, DB2, SQL Server, ...

Schéma relationnel

Un schéma relationnel est composé :

- d'un ensemble d'attributs :
  - décrit les données atomiques que l'on veut manipuler
  - ex : `titre`, `année`, `genre`
- d'un ensemble de relations ou tables sur ces attributs :
  - représente les liens entre les données atomiques
  - permet de représenter des objets complexes
  - ex : `Film(titre, année, genre)`
    Rmq : on dit que `titre`, `année` et `genre` sont les attributs de la relation `Film`
  - Arité d'une relation : nombre d'attributs de cette relation
  - une table est une relation d'un schéma

Schéma relationnel - suite

Dans un schéma relationnel on trouve également des contraintes :

- Le type des attributs :
  - `titre : string, année : integer, genre : string`

  Domaine de valeurs : Ensemble d'instances d'un type élémentaire.
  - Exemple : les entiers, les réels, chaîne de caractères, ...
- Des contraintes plus complexes comme :
  - "Dans la relation `Film`, il ne peut y avoir qu'une seule `année` et un seul `genre` correspondant à un titre donné"

Représentation d'un schéma relationnel

On peut représenter un schéma relationnel par :

- un ensemble de schémas de relations décrivant le contenu relation avec les éléments (domaines, attributs, noms de relations) :
  - nom de relations + nom d'attributs + types d'attributs
  - Ex :
    `Film(titre : string, année : integer, genre : string)`

  L'ensemble des attributs qui apparaissent dans les schémas de relations donne l'ensemble des attributs du schéma relationnel
- les contraintes complexes :
  - "Dans la relation `Film`, il ne peut y avoir qu'une seule `année` et un seul `genre` correspondant à un titre donné"

C'est cette représentation qui est plus couramment utilisée.

Conception de schéma

- La création d'un schéma est simple une fois toutes le relations déterminées.
- Le choix des relations est difficile :
  - il détermine les caractéristiques de qualité de la base : performances, exactitude, exhaustivité, disponibilité des informations.
- Il existe des méthodologies de conception de logiciels aidant au choix des relations à utiliser dans un schéma :
  - Schéma entités-associations
  - Merise
  - UML

Instances

Une instance d'une base de données est un ensemble d'instances de relations (une par relation du schéma de la base)
Une instance d'une relation $R(A_1, \ldots, A_n)$ est un sous-ensemble fini du produit cartésien des domaines de ses attributs :

- Si D_1 est le domaine (du type) de A_1, ..., D_n est le domaine (du type) de A_n
- toute instance de R est incluse dans $D_1 \times \cdots \times D_n$
- vision ensembliste des relations

Conséquences :

- l'ordre est éléments n'est pas important
- absence de doublons
- toutes les valeurs des attributs dans l'instance sont connues

En pratique, les choses sont différentes

Représentation des données par des instances

Ce sont des instances des relations du schéma qui servent à représenter les données :

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

L'instance est un ensemble de *n*-uplets (tuples en anglais) :
{(Alien, 1979, Science-fiction), (Vertigo, 1958, Suspense), (Volte-face, 1997, Thriller), (Pulp fiction, 1995, Policier)}

Ce sont ces instances de relation qui sont stockées.

Manipulation des données

L'interrogation de données se fait via la manipulation des relations :

- Opérations :
  - Entrée : une ou plusieurs relations (qui peuvent être ou non des tables stockées)
  - Sortie : une relation
- Types d'opérations :
  - Sélection de n-uplets intéressants
  - Opérations ensemblistes classiques : union, intersection, différence, produit cartésien

Mise à jour : ajout et/ou suppression de n-uplets dans les tables.
Ces n-uplets peuvent être récupérés via une interrogation des données

Plan

Langages de manipulation de données pour le modèle relationnel

- Deux catégories de langages de manipulation de données :
  - algébrique (algèbre relationnelle)
  - prédicats (calculs relationnels)
- Puissance d'expression équivalente
- Servent de base à d'autres langages plus conviviaux pour les utilisateurs (SQL)

Intérêt de l'algèbre relationnelle

- Proposée par E. Codd, 1969
- Identification des opérateurs fondamentaux pour l'utilisation d'une Base de Données Relationnelle.
- Définition des principales fonctions à optimiser dans un SGBD Relationnel.
- A donné naissance à des LMD pour les utilisateurs (ISBL par IBM en habillant l'algèbre d'une syntaxe plus agréable)

Algèbre relationnelle

- Un ensemble d'opérations pour la manipulation des relations considérées comme des ensembles de n-uplets :
- Création d'une nouvelle relation temporaire à partir 2 de relations (a une durée de vie limitée, détruite à la fin du programme ou de la transaction qui l'a créée).
- La relation a mêmes caractéristiques qu'une relation de la base et peut être manipulée de nouveau par les opérateurs de l'algèbre.

Opérateurs

Formellement l'algèbre comprend :

- 5 opérateurs de base : sélection, projection, union, différence et produit.
- 1 opérateur syntaxique, le renommage, qui ne fait que modifier le schéma et pas les n-uplets.
- D'autres opérateurs proposés équivalents à la composition d'opérateurs de base, les opérateurs déduits :
  - des raccourcis d'écriture n'apportant aucune fonctionnalité nouvelle, mais pratiques
  - intersection, jointure naturelle, thêta jointure et division.

On peut les regrouper en deux catégories :

- opérateurs ensemblistes : union, intersection, différence, produit
- opérateurs spécifiques BDDR : sélection, projection, jointures, division, renommage

Union ∪

- $R \cup S$ crée une relation comprenant tous les n-uplets existants dans l'une ou l'autre des relations R et S.
- Les 2 relations doivent avoir le même nombre d'attributs, et les mêmes types (*i.e.* même domaine)
- Éliminations des doublons.

Exemple ∪

Etudiants

Prénom	Nom
Susan	Yao
Ramesh	Shah
Barbara	Jones
Amy	Ford
Jimmy	Wang

Profs

PrénomP	NomP
John	Smith
Ricardo	Brown
Susan	Yao
Francis	Johnson
Ramesh	Shah

Etudiants ∪ Profs

Prénom	Nom
Susan	Yao
Ramesh	Shah
Barbara	Jones
Amy	Ford
Jimmy	Wang
John	Smith
Ricardo	Brown
Francis	Johnson

Intersection ∩

- R ∩ S crée une nouvelle relation de même schéma et de population égale à l'ensemble des n-uplets de R tels qu'il existe n-uplet de même valeur dans S.
- Les 2 relations doivent avoir le même nombre d'attributs, et les mêmes types (*i.e.* même domaine)

Exemple ∩

Etudiants

Prénom	Nom
Susan	Yao
Ramesh	Shah
Barbara	Jones
Amy	Ford
Jimmy	Wang

Profs

PrénomP	NomP
John	Smith
Ricardo	Brown
Susan	Yao
Francis	Johnson
Ramesh	Shah

Etudiants ∩ Profs

Prénom	Nom
Susan	Yao
Ramesh	Shah

Différence − ou \

- R \ S crée une relation de même schéma et de population égale à l'ensemble des n-uplets de R moins ceux de S, c'est-à-dire les n-uplets qui se trouvent dans R mais pas dans S.
- Les 2 relations doivent avoir le même nombre d'attributs, et les mêmes types (*i.e.* même domaine)

Exemple − ou \

Etudiants

Prénom	Nom
Susan	Yao
Ramesh	Shah
Barbara	Jones
Amy	Ford
Jimmy	Wang

Profs

PrénomP	NomP
John	Smith
Ricardo	Brown
Susan	Yao
Francis	Johnson
Ramesh	Shah

Etudiants − Profs

Prénom	Nom
Barbara	Jones
Amy	Ford
Jimmy	Wang

Produit cartésien ×

- R × S crée une nouvelle relation où chaque n-uplet de R est associé à chaque n-uplet de S
- Le nombre de lignes est $|R| \times |S|$ (où $|R|$) est le nombre de lignes dans la relation R).
- Associe chaque ligne de R à chaque ligne de S.
- Intérêt lié à la jointure

Exemple ×

Etudiants

Prénom	Nom
Susan	Yao
Ramesh	Shah

Profs

PrénomP	NomP
John	Smith
Ricardo	Brown
Susan	Yao

Etudiants × Profs

Prénom	Nom	PrénomP	NomP
Susan	Yao	John	Smith
Susan	Yao	Ricardo	Brown
Susan	Yao	Susan	Yao
Ramesh	Shah	John	Smith
Ramesh	Shah	Ricardo	Brown
Ramesh	Shah	Susan	Yao

Renommage ρ

- $\rho_{A_1/A'_1,\ldots,A_k/A'_k}(R)$
- Changement du nom d'un ou plusieurs attributs d'une relation R : A_1 devient A'_1, ..., A_k devient A'_k
- Utile en cas de problème d'homonymie ou avant certaines opérations ensemblistes.

Exemple ρ

Employe

Prénom	Nom	NoDept
John	Smith	5
Ricardo	Brown	3
Susan	Yao	5
Daniel	Johnson	2
Francis	Johnson	2
Ramesh	Shah	4
Ramesh	Shah	2

$\rho_{Prenom/First,Nom/Last}(\text{Employe})$

First	Last	NoDept
John	Smith	5
Ricardo	Brown	3
Susan	Yao	5
Daniel	Johnson	2
Francis	Johnson	2
Ramesh	Shah	4
Ramesh	Shah	2

Sélection σ

- $\sigma_C(R)$ sélection les n-uplets de R en utilisant la condition C
- Conditions : combinaisons de comparaisons $(=, <, >, \leq, \geq)$
  - entre deux attributs
  - ou entre un attribut et une constante
- Exemple : $\sigma_{NoDept=5}(Employe)$

Exemple σ

Employe

Prénom	Nom	NoDept
John	Smith	5
Ricardo	Brown	3
Susan	Yao	5
Daniel	Johnson	2
Francis	Johnson	2
Ramesh	Shah	4
Ramesh	Shah	2

$\sigma_{NoDept=5}(\text{Employe})$

Prénom	Nom	NoDept
John	Smith	5
Susan	Yao	5

Projection π

- $\pi_{A_1,\ldots,A_k}(R)$ ne garde que les attributs $A_1, \ldots, A_k$ de la relation R.
- On ne supprime pas de ligne mais des colonnes.
- Élimination des doublons (il ne peut y avoir deux fois le même élément dans un ensemble).

Exemple π

Employe

Prénom	Nom	NoDept
John	Smith	5
Ricardo	Brown	3
Susan	Yao	5
Daniel	Johnson	2
Francis	Johnson	2
Ramesh	Shah	4
Ramesh	Shah	2

$\pi_{Nom,NoDept}(\text{Employe})$

Nom	NoDept
Smith	5
Brown	3
Yao	5
Johnson	2
Shah	4
Shah	2

Jointure $\bowtie$

Jointure naturelle : $R \bowtie S$

- R et S ont les attributs $A_1, \ldots, A_k$ en commun
- On obtient l'ensemble des n-uplets constitués à partir de n-uplets n_1 de R et de n-uplets n_2 de R ayant les mêmes valeur pour les attributs $A_1, \ldots, A_k$.
- Les n-uplets obtenus sont construits comme suit :
  - On ajoute à n_1 la valeur des attributs de n_2 qui ne sont pas dans R

θ-jointure : $R \bowtie_C S$

- La condition d'égalité entre des attributs communs est remplacée par la condition de jointure C
- Utile lorsqu'il n'y a pas d'attributs en commun entre R et S

Exemple ⋈

Employe

Prénom	Nom	NoDept
John	Smith	5
Ricardo	Brown	3
Susan	Yao	5
Francis	Johnson	2
Ramesh	Shah	4

Emplacement

NoDept	Building
1	centre
3	sud
4	est
5	ouest

Employe ⋈ Emplacement

Prénom	Nom	NoDept	Building
John	Smith	5	ouest
Ricardo	Brown	3	sud
Susan	Yao	5	ouest
Ramesh	Shah	4	est