Principes d'utilisation des systèmes de gestion de bases de données

Introduction

quoi ? qui ? quand ? pourquoi ? comment ? etc. (1/2)

« Principes d'utilisation des SGBD »

- SGBD : qu'est-ce que c'est ?

- utilisation :
  - qui peut l'utiliser ?
  - dans quelles situations est-ce utile?
  - pourquoi ? (quel est le problème ?) pourquoi faire ?
  - comment l'utiliser ?
  - sous quels modes ?
    - en mode interactif : depuis clavier
    - en mode programme : depuis programme

quoi ? qui ? quand ? pourquoi ? comment ? etc. (2/2)

- principes :
  - approche raisonnées (pas liste de commandes)
  - indépendant d'un SGBD particulier

introduction

- questions...
- les problèmes de BD : quand, pour quoi faire ?
- vocabulaire : quoi
- but du cours (1/2) : pourquoi, comment
- qui
- les problèmes du mode programme
- but du cours (2/2)
- vue d'ensemble et architecture
- structure du cours
- compétences à acquérir
- utilité du cours
- construction et théorie
- compléments éventuels

les problèmes de base de données

- contexte : vraie vie :
  - Programmeurs écrivent applications
  - Rencontrent nombreux problèmes (algorithmique, réseau, graphique, etc.)
  - Il faut les gérer
  - on va s'intéresser à un ensemble particulier de problèmes

- exemple : application = situation à informatiser : réservations de places SNCF
  - Clients effectuent réservations (client, date, train, etc.)
  - Employés gèrent trains (départ, arrivée, horaire, etc.)

persistance

- situation :
  - Client effectue sa réservation sur le web (minitel, etc.)
  - Il se déconnecte
- il faut :
  - Sa réservation ne doit pas disparaître (stockée)
  - Son numéro de carte bleue doit disparaître après paiement (non stockée)

grandes quantités

- Situation :
  - Quantité d'informations supérieure à mémoire vive
  - Accès disque lent
- Il faut :
  - Temps de réponse raisonnable
- (résoudre : optimisation)

reprise sur panne

- Situation :
  - Réservation et n° CB tapés par client
  - Paiement effectué
  - Panne de courant avant que réservation stockée
- Il faut :
  - Paiement et réservation ne doivent pas être dissociés

concurrence

- Situation :
  - Il reste une place dans le Paris-Saint Tropez de 17h05
  - 2 clients se connectent simultanément
- Il faut :
  - On accepte que l'un passe en premier et l'autre n'ait rien
  - Mais pas que les deux paient pour le même siège

confidentialité

- Situation :
  - Un client veut modifier
    - L'horaire d'un train
    - Les réservation d'autres clients
- Il faut :
  - Qu'il ne puisse pas
  - Un employé doit pouvoir

contraintes d'intégrité

- Situation :
  - un employé distrait (ou programme bugué) introduit un kilométrage négatif
- Il faut :
  - Date ou kilométrage ne doivent pas être négatifs
  - Pour un train donné : nombre de réservations + nombre de places libres = nombre total de places

répartition

- Situation :
  - Pour une réservation Paris-Venise votre application doit dialoguer avec l'application homologue italienne
- Il faut :
  - Tous les problèmes doivent être gérés en coopération entre les deux applications

indépendance des niveaux

- Situation :
  - une fois votre application écrite, vous décidez de modifier l'organisation des données sur le disque (ex : cause lenteur)
- Il faut :
  - Vous ne voulez pas réécrire toute votre application à chaque fois
  - (« modularité » entre le niveau du disque et le niveau de l'application ; analogue esprit objet)

modèle, conception, mise à jour, interrogation

- Situation :
  - l'application doit manipuler ses informations (clients, trains, etc.)

- Il faut :
  - Les représenter : modèle de données (= langage de description)

    (ex : en Java, concepts : objets, champs, classes, etc.)
  - Les organiser : conception (ex : Java : bonnes classes)
  - Les créer, modifier, détruire : mise à jour
  - Les interroger : interrogation
  - De manière simple (maintenance, efficacité, Java trop compliqué)

Vocabulaire (1/3)

- Information = donnée
  - Ex : Il y a un train Paris-Saint Tropez
  - Ex : Jules a une place dans le Paris-Saint Tropez
- Informations = base de données
- Ces 12 problèmes sont appelés les « problèmes de base de données »

Paris-Saint Tropez

- Domaine des bases de données = étude des outils théoriques et pratiques pour résoudre ces problèmes (recherche, R & D, technologie, méthodologie, etc.)

P a r i s - S a i n t T r o p e z

Vocabulaire (2/3)

- Définition : SGBD = logiciel pour gérer les problèmes de base de données
- Pour chaque problème, le SGBD fournit un ensemble d'outils (conceptuels et/ou pratiques) permettant au programmeur de gérer ce problème (peut être difficile)

Vocabulaire (3/3)

- On utilise le SGBD au moyen d'un ensemble d'ordres dits « de base de données » : le langage SQL
- Une application = des ordres de base de données
  - tapés directement au clavier : mode interactif
    - ex : SNCF : un expert BD tape en permanence tout ce qu'il faut pour gérer les problèmes de base de données de l'appli
  - lancés depuis un programme : mode programme
    - ex : SNCF : ces ordres sont insérés dans un programme autonome, Java par exemple

But du cours (1/2)

Lors écriture application généraliste, savoir utiliser le SGBD : outils (SQL) pour gérer les 12 problèmes BD rencontrés

qui ? (1/2)

- 3 catégories d'acteurs dans une application (BD)
  - programmeur dans une SSII (vous) : écrit l'application de A à Z
  - utilisateur non informaticien (votre grand'mère) : utilise l'application, ne comprend rien à l'informatique
  - administrateur de la base : DBA (ingénieur DSI) :
    - installe et administre les logiciels utilisés par le programmeur pour écrire l'application : SGBD (ex : Oracle), langage (ex : Java), etc.
    - création comptes utilisateurs, sauvegardes, etc.

qui (2/2)

- seul le programeur gère les problèmes BD et du mode programme
  - DBA n'a pas vraiment besoin de savoir gérer les problèmes BD
  - secrétaire, PDG : non : trop complexe pour non informaticien

introduction

- questions...
- les problèmes de BD : quand, pour quoi faire ?
- vocabulaire : quoi
- but du cours (1/2) : pourquoi, comment
- qui
- les problèmes du mode programme
- but du cours (2/2)
- vue d'ensemble et architecture
- structure du cours
- compétences à acquérir
- utilité du cours
- construction et théorie
- compléments éventuels

les problèmes du mode programme

- quoi ? qui ? quand ? pourquoi ? comment
- exemple : Java
  - notre application SNCF écrite en Java
  - il faut lancer les ordres BD depuis ce programme Java
- l'insertion d'ordres BD dans un programme pose les problèmes suivants (exemples)
- ces 9 problèmes sont appelés « les problèmes du mode programme »

But du cours (2/2)

Lors écriture application généraliste, savoir utiliser le mode programme : outils pour gérer les 9 problèmes rencontrés lors accès au serveur BD

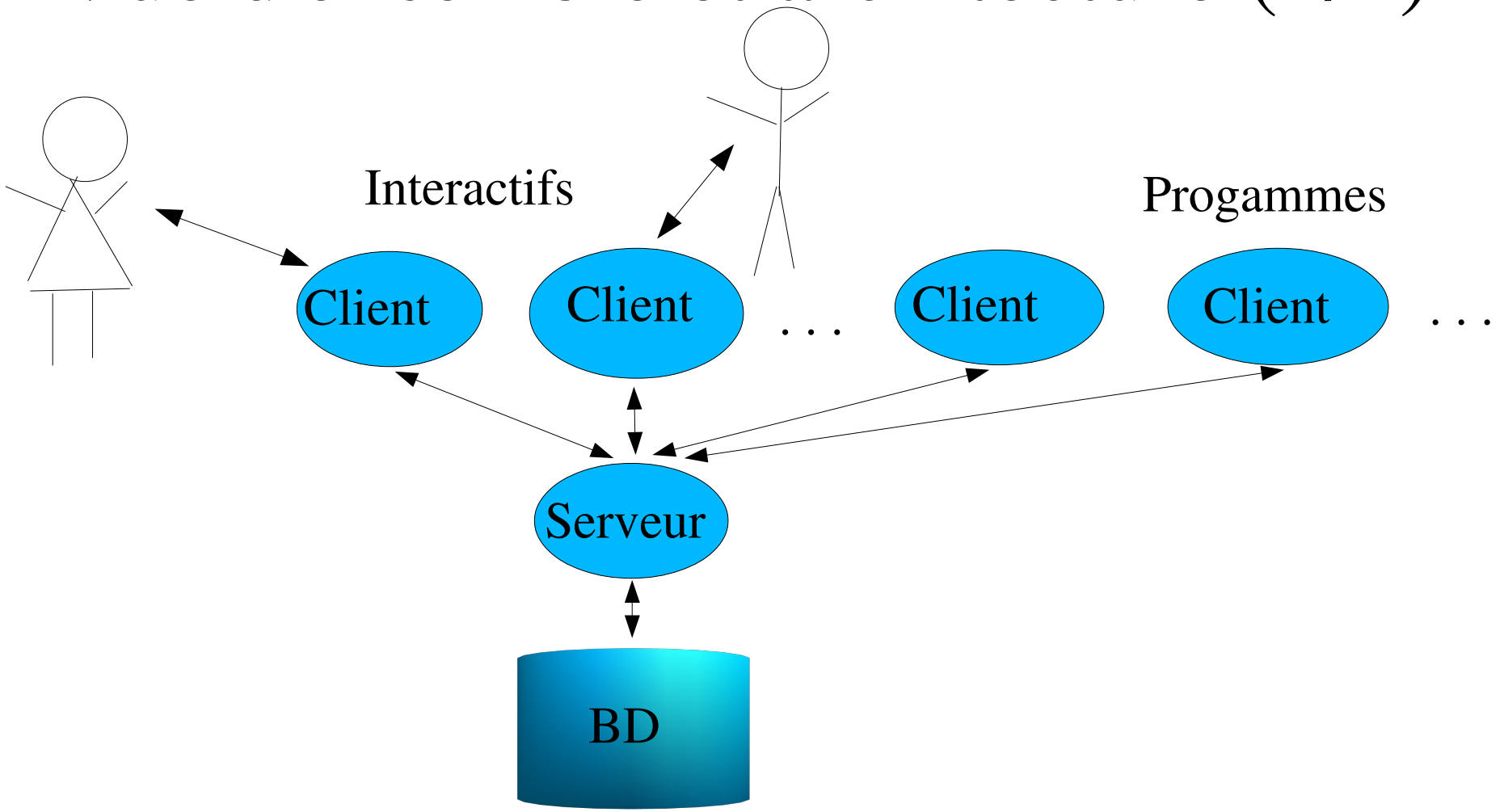
introduction

- questions...
- les problèmes de BD : quand, pour quoi faire ?
- vocabulaire : quoi
- but du cours (1/2) : pourquoi, comment
- qui
- les problèmes du mode programme
- but du cours (2/2)
- vue d'ensemble et architecture
- structure du cours
- compétences à acquérir
- utilité du cours
- construction et théorie
- compléments éventuels

Vue d'ensemble et architecture

- La bonne architecture pour gérer les problèmes BD (transparent suivant) :
  - La base de données : données
  - Le SGBD (programmes en train de s'exécuter) :
    - Le serveur
      - Unique
      - Tourne en permanence
    - Les clients (interactifs ou applications)
      - En nombre quelconque
      - Se (dé)connectent à volonté
      - Non coordonnés (subtilités lors cours transactions)

Vue d'ensemble et architecture (1/2)



Vue d'ensemble et architecture (2/2)

- Vous en train de taper devant votre clavier : mode interactif
  - Gérer la partie base de données de votre application en écrivant les ordres de base de données nécessaires
  - Session = les ordres SQL entre connexion et déconnexion
- Programme en train de s'exécuter : mode programme
  - Mêmes gestion et ordres de base de données qu'en interactif
  - Lancés depuis un programme (Java, web, etc.)

structure du cours (1/2)

les problèmes de base de données :

- on va les considérer un par un
- pour chacun, on va :
  - Comprendre où est le problème
  - Comprendre et utiliser les outils que fournit le SGBD :
    - Les concepts pour approcher ce problème
    - Les ordres de base de données pour programmer une solution concrète au problème concret rencontré par une application

compétences à acquérir

- concernant les problèmes de base de données :
  - comprendre les mécanismes au coeur de ces problèmes : pourquoi
  - Pour une application donnée :
    - Considérer chacun des 12 problèmes
    - Détecter toutes ses occurrences dans l'application : quand (ex : accès concurrents)
    - Programmer la meilleure solution possible pour chacune des occurrences : comment
  - Répartition : hors programme
- concernant les problèmes du mode programme : même chose

pourquoi il ne suffit pas de lire la doc

- brillants collègues (non BD) qui l'ont fait tombent dans tous les pièges
- comparable à Java :
  - paraît facile quand L3 terminé (en gros...)
  - impossible apprendre correctement seul

construction d'un SGBD et théorie (1/4)

- pour chacun des 12 problèmes BD :
  - le comprendre
  - définir un ensemble d'outils permettant à chaque programmeur de gérer chaque occurrence de ce problème dans chaque application

  : étude théorique, système et implantation

construction d'un SGBD et théorie (2/4)

- ex : modèle de données
  - le constructeur doit fournir un « bon » modèle
  - avec ce modèle le programmeur doit pouvoir représenter ses données de manière « efficace »
  - ex :
    - hiérarchique : existe encore (ex : KLM)
    - relationnel : environ 80% du marché mondial : Oracle
    - objet : disparus (O2 : français), il en reste un
    - XML : en expansion

construction d'un SGBD et théorie (3/4)

- ex : interrogation
  - constructeur doit fournir « bon » langage (ensemble d'ordres)
  - tel que programmeur doit pouvoir formuler ses interrogations « efficacement »
  - ex : SQL, algèbre, calcul, datalog, fixpoint
  - théorie :
    - qu'est-ce qu'une « interrogation » (« requête ») ?
    - quel est l'objet formel correspondant ?
    - quelles propriétés : monotonie, inclusion, complexité, expressivité ?
    - rem : programmeur confronté à théorie : thm (admis) : il existe des interrogations qu'aucun ordre SQL ne peut exprimer

construction d'un SGBD et théorie (4/4)

- ex : concurrence :

  constructeur doit fournir bon système de verrous

Autres SGBD (célèbres)

- Coût Oracle entreprise : environ 100 000 euros
- SQLServer (Microsoft), DB2 (IBM), Ingres, etc.
- Libres : MySQL5, Postgres
- (Access, Microsoft)
- etc.

bibliographie

- SQL pour Oracle, Christian Soutou, Eyrolles, 4ème édition, 2010 (670 pages), 32 euros
- Maîtriser MySQL 5, Yves Darmaillac et Philippe Rigaux, O'Reilly, 2005 (500 pages) [BD]
- Pratique de MySQL et PHP, Philippe Rigaux, O'Reilly, 3ème édition, 2005 (600 pages) [Web et BD]
- Comprendre XSLT, Bernd Amann et Philippe Rigaux, O'Reilly, 2002 (500 pages) [XML]
- (voir page Rigaux : premiers chapitres libres)

But du cours

Lors écriture application généraliste, savoir utiliser :

- Le SGBD : outils (SQL) pour gérer les 12 problèmes BD rencontrés
- Le mode programme : outils pour gérer les 9 problèmes rencontrés lors accès au serveur BD

Les deux parties du cours

- création et gestion de la base
  - les problèmes BD
    - liés à la construction de la base :

      modèle, conception, indépendance des niveaux, contraintes, confidentialité, mise à jour, persistance
    - liés à la dynamique de la base :

      reprise sur panne, contrôle de concurrence
    - liés à l'interrogation de la base :

      interrogation, grandes quantités (optimisation)
  - les traitements bas niveau en mode programme : PL/SQL

Rappels L2/MAN BD

- Modèle
- Mises à jour
- Persistance
- Interrogation

introduction

- questions...
- les problèmes de BD : quand, pour quoi faire ?
- vocabulaire : quoi
- but du cours (1/2) : pourquoi, comment
- qui
- les problèmes du mode programme
- but du cours (2/2)
- vue d'ensemble et architecture
- structure du cours
- compétences à acquérir
- utilité du cours
- construction et théorie
- compléments éventuels

Ce qu'il faut se rappeler

- Pourquoi utiliser un SGBD ? Pour ne pas avoir à reprogrammer tous les outils qu'il fournit pour gérer les problèmes BD
- Toute application manipulant des données persistantes doit utiliser un SGBD