PERSISTANCE

JÉRÉMY PERROUAULT

INTRODUCTION

INTRODUCTION

PROBLÉMATIQUE

- Vie d'une application et d'une machine
  - Lancement, exécution, arrêt, faillite
  - Les données en mémoire vive sont volatiles
- Altération de la donnée
  - Génère des incohérences

SOLUTIONS

- Techniquement, plusieurs solutions sont possibles
  - Stockage de données sérialisées (flux de données à plat) dans un fichier
  - Stockage des données dans une structure bi-dimensionnelle (tables)
  - Stockage des données dans une structure de documents
  - Stockage des données dans une structure hiérarchique (graphes)

- Sauvegarde des données
  - Dans un fichier, sur le disque
  - Dans une base de données relationnelle (ou pas relationnelle)
  - Dans une base de données graphe
  - Dans une base documentaire
  - etc.

SOLUTIONS

- Le système de stockage choisi, il faut implémenter la **logique de persistance**
  - Règles de transformation de la mémoire vers le support
  - Création des modèles / structures nécessaires

- C'est une problématique complexe
  - Nécessite la mise en œuvre de mécanismes spécifiques

SOLUTIONS

- Couche d'accès aux données (ou de persistance)
  - Le code d'accès aux données est séparé du code métier
    - Garantie un couplage faible entre le métier et la données
    - Le code métier est simplifié – il ne s'occupe pas d'accéder à la données
  - Ajoute un niveau d'abstraction entre la source et l'utilisation des données
  - Masque les traitements complexes d'accès aux données (requêtes, écriture, lecture, …)

- Manipulation de
  - CRUD (Create – Read – Update – Delete)
  - Entités / Documents
  - DAO (Data Access Object) / Repositories
  - ORM (Object-Relational Mapping)

Attention ! ORM est persistance, mais persistance n'est pas ORM

PRINCIPE

- Structuration des données et de la logique
- Intégrité référentielle, cohérence
- Exploitation et maintenance

PRINCIPE

- Intégration d'un moteur de persistance : persistance transparente
  - Principe d'orthogonalité
    - Tout objet doit pouvoir être persisté, peut importe son type
    - Tout objet doit pouvoir être déclaré comme étant « temporaire » ou « transitoire » (**Transient**) – non-persistant
  - Principe de transparence
    - Du point de vue métier, pas de différence dans la manipulation d'objets persistés et non persistés
  - Principe de transparence transitive (*persistence by reachability*)
    - Tout objet non-persistant référencé au travers d'un objet persisté **doit** être persisté automatiquement
      - Tout le graphe doit être persisté
    - Tout objet référencé par un objet supprimé **doit** être supprimé, s'ils ne sont pas référencés par d'autres
  - Principe d'héritage
    - Si les instances d'une classe sont persistés, tout objet des sous-classes doit pouvoir être persisté

STRUCTURES

DIFFÉRENTS SYSTÈMES

FICHIER

- Généralement une sérialisation d'un objet ou d'un graphe d'objets est réalisé
  - JSON ou XML par exemple

SQL

- Une base de données relationnelle
  - Assurent l'atomicité des transactions et la cohérence des données (propriétés *ACID*)

- Exemples de systèmes :
  - MySQL / MariaDB
  - PostgreSQL
  - SQLite
  - MS SQL
  - Oracle

NOSQL – ORIENTÉ DOCUMENT

- Plus adaptée aux grands volumes de données hétérogènes
- Utilise généralement des modèles de données plus simples et moins structurés
- Certaines peuvent assurer l'atomicité des transactions

- Exemples de systèmes :
  - MongoDB
  - CouchDB

NOSQL – ORIENTÉ GRAPHE

- Modélisation, stockage de manipulation de données complexes liées par des relations variables
  - Compte Facebook lié aux « amis »
  - Alexa, Siri, etc.

- Exemples de systèmes :
  - Neo4j
  - Knowledge Graph

NOSQL – ORIENTÉ COLONNES

- Très pratique pour des requêtes qui traitent seulement un sous-ensemble
- Stocke les informatiques avec une clé unique par rangée

ID	Nom	Prenom
01	DUPONT	Albert
02	DOE	John

  - DUPONT:01;Albert:01
  - DOE:02;John:02

- Exemples de systèmes :
  - BigTable
  - HBase
  - MS SQL (depuis 2012), si utilisation d'index columnstore ou de tables verticales

NOSQL – ORIENTÉ CLÉ / VALEURS

- Une base de données « clé / valeur », la plus simple
  - L'information est stockée dans une collection de paires clé = valeur
    - Chaque clé est unique
    - La valeur est souvent représentée par une chaine de caractères
    - Pas de schéma

- Exemples de systèmes :
  - Fichiers properties
  - Riak
  - Redis