Projet DANT NTW

11/02/2019

Vue d'ensemble

Des équipes de 2 à 3 étudiants vont développer un moteur d'index distribué à l'image d'Elasticsearch. Un moteur d'index permet d'accélérer la récupération de données d'une base de données. La donnée est stocké en mémoire.

Objectifs

1. **Ingérer de la donnée :** Charger de la donnée le plus rapidement possible.
2. **Retrouver les lignes :** Effectuer des requêtes permettant de retrouver rapidement de la donnée fine. Les fonctions SQL similaires sont le SELECT … WHERE … GROUP BY
3. **Distribuer le dataset :** Tournant sur 3 noeuds, développer le meilleur moyen de répartir la donnée

Technologies

Back-end : JAX-RS ou vert.x, full Java

Infrastructure : Git, Maven, GitHub

Caractéristiques

En utilisant JAX-RS ou vert.x, les étudiants créeront une API REST au format JSON pour :

- Créer des tables : À l'image des tables SQL, elles possèdent des colonnes nommées ayant un type

- Ajouter des index : Un index est défini sur 1 … n colonnes de la table
- Charger de la donnée : Au format CSV, 1 fichier à la fois qui sera parsé
- Récupérer les lignes d'index : En utilisant l'index **approprié**, le moteur pourra renvoyer les lignes recherchées

Grandes étapes

1. Développer l'API REST

   Définir une API REST standard permettant d'effectuer les 4 opérations ci-dessus.

2. Ajouter le support des tables / index

   La structure interne restera simple (HashMap, …) mais permettra d'obtenir des premiers résultats probants sur une faible volumétrie.

3. Charger de la données

   Réaliser le parsing du CSV et l'ingestion des lignes dans les index d'une table

4. Distribuer le serveur

   Être capable de faire communiquer les noeuds. Lorsqu'une action s'effectue sur un noeud, elle est répercutée sur les autres. Chaque noeud connaît les tables, index … LEs notions de tolérance aux pannes et réplication ne sont pas l'objectif du projet.

5. Rendre le programme performant

   - Être capable d'avoir le plus de lignes en mémoire disponible
   - Loader / Récupérer de la donnée le plus vite possible
   - Découvrir les techniques d'indexation (B-Tree, hash, …)

Fonctionnalités supplémentaires

Plus les étudiants ajouteront des features à forte valeur ajoutée, plus leur note augmentera. Quelques exemples :

- Support des aggrégats (SUM, AVG, MIN, MAX, COUNT) comme en SQL
- Mise à jour / suppression de lignes dans les index
- Support du HAVING ORDER BY LIMIT
- Administration des tables / index (suppression, ajout de colonnes …)
- Ajout de fonctions comme CONTAINS, YEAR … dans les requêtes de sélection

Où trouver de la vraie donnée ?

La soutenance se fera sur le dataset de NYC CAB. Il s'agit de donnée publique sur les trajets ds taxis de New-York. Elle est accessible ici : https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page et nous nous focaliserons sur les fichiers nommés 'Yellow' donc ni Green, ni FHV). Utiliser la donnée de **2009 à 2015**.

Notation et rendu

- Chaque équipe fournira les accès à son GitHub au professeur : olivier.pitton@gmail.com
- 4 points sur la soutenance avec un PowerPoint
- 4 points sur la qualité du code (**tests unitaires** !)
- 12 points sur les fonctionnalités.

Ressources

ElasticSearch : https://www.elastic.co/fr/products/elasticsearch

Github DANT : https://github.com/olivier-pitton/dant

Java Collections Performance : http://blog.xebia.fr/2011/02/17/java-collection-performance/