Écosystème Haadop – TP Map/Reduce
Map/Reduce sur MongoDB

proposé par Mouhamadou Lamine Ba
mouhamadoulamine.ba@uadb.edu.sn

1 Objectif du TP

L'objectif de ce TP est de mettre en œuvre des traitements Map/Reduce sur MongoDB. Le TP est à réaliser en binôme, de préférence sous Linux, et sera rendu au plus tard le 30 Mars 2019 à minuit.

Vous devrez rendre une archive zip nommée mongodb_mapreduce_binôme1_binôme2.zip où binôme1 et binôme2 sont les noms complets des membres du binôme.

2 Environnement de travail

2.1 Références

1. https://docs.mongodb.com/manual/introduction/
2. https://docs.mongodb.com/manual/
3. http://docs.mongodb.org/manual/reference/command/mapReduce/
4. http://stackoverflow.com/questions/6333164/how-do-i-use-map-reduce-in-mongodb
5. http://www.w3schools.com/js/default.asp

2.2 Serveur Mongo

Vous utiliserez un serveur MongoDB que vous devrez installer et configurer sur votre machine de travail.

3 Collections

Vous utiliserez le client en ligne de commande mongo depuis Linux. Après authentification à votre serveur MongoDB, vous devrez créer les collections suivantes dans la base de données masid_mongodb :

1. RESTAURANTS : une collection d'évaluations de restaurants aux États-Unis ;
2. ZIPS : une collection de données sur les code postaux aux États-Unis ;
3. STATES : les états des États-Unis et leur population d'après la collection zips ;

Pour travailler sur votre propre serveur, vous pouvez récupérer et importer ces jeux de données avec les commandes suivantes :

— *mongoimport –db glsi_mongodb –collection zips –drop –file zips.json*
— *mongoimport –db glsi_mongodb –collection restaurants –drop –file restaurants.json*

4 Exercice 1 : premiers jobs Map/reduce

Le fichier de réponses pour cet exercice est nommé *MASID_TP_MapReduce_Ex1.js*.

1. Calculer le nombre total d'enregistrements de la collection zips en remplissant les fonctions ex01q1map et ex01q1red. Le résultat du job Map/Reduce sera stocké dans la variable ex01q1.
2. Modifier la fonction ex01q1red pour utiliser la fonction *values.reduce()* [1] à la place de la boucle et l'enregistrer dans la variable ex01q2red.
3. Modifier la fonction ex01q1red pour utiliser la fonction *values.forEach* [2] à la place de la boucle et l'enregistrer dans la variable ex01q3red.

5 Pipeline Map/Reduce sur MongoDB

Exercice 2 : requêtes en Map/Reduce

Le fichier de réponses pour cet exercice est nommé *MASID_TP_MapReduce-Ex2.js*. Vous donnerez les définitions des variables demandées correspondant aux jobs Map/Reduce des questions suivantes.

1. https://developer.mozilla.org/JavaScript/Reference/Global_Objects/Array/Reduce
2. https://developer.mozilla.org/JavaScript/Reference/Global_Objects/Array/forEach

1. Donner les codes postaux et leur population quand elle est supérieure à 100.000 en filtrant dans le *map*.
2. Même question mais en filtrant avec le paramètre query de la fonction *mapReduce()* [3]. Laquelle des deux solutions faut-il privilégier ?
3. Donner la population totale de chaque état, puis en javascript avec *sort* et *slice*, filtrer ce résultat en javascript pour ne garder que les 3 états les plus peuplés. Vérifiez que votre résultat est cohérent avec la réalité.
4. Donner la liste des villes des US, attention à ne pas mélanger des villes homonymes mais dans des états différents. Pour tester votre requête, utiliser le paramètre *limit* de *mapReduce()*.

Exercice 3 : requêtes avancées en Map/reduce

Le fichier de réponses pour cet exercice est nommé MASID_TP_MapReduce-Ex3.js. Vous donnerez les définitions des variables demandées correspondant aux jobs Map/Reduce des questions suivantes.

1. Pour chaque état, donne la liste de ses villes sans supprimer les doublons. Aide : il faut impérativement que le type des valeurs émises par le *map* (retourné par emit) soit le même que le type d'entrée des valeurs du *reduce* (paramètre values de la fonction) et le type de sortie du *reduce* (retourné par *return*) soient les mêmes [4] [5].
2. Même question, mais en supprimant les doublons. Aide : utiliser un objet *json* comme un tableau associatif pour garantir l'unicité.
3. Pour chaque état, donner à la fois le code postal le plus peuplé et le moins peuplé. Aide : utiliser un objet *json* comme valeur du *emit*.
4. Donner l'état le plus peuplé des US un job Map/Reduce sur la collection *states*.
5. Même question avec un seul job directement depuis la collection zips. Aide : tout le calcul se passe désormais dans le *reduce* et dans *finalize*.

Fin du TP !

3. http ://docs.mongodb.org/manual/reference/operator/query/
4. `http://isurues.wordpress.com/2013/05/28/what-is-re-reduce-in-mongodb-map-reduce/`
5. `http://isurues.wordpress.com/2013/05/28/what-is-re-reduce-in-mongodb-map-reduce/`