Suite à ma recherche sur comment on peut écrire un code map-reduce en java on a compris que :

* On doit écrire en plus du code de mapper et reducer la classe de driver.
* La classe mapper hérite de **Mapper<KEYIN, VALUEIN, KEYOUT, VALUEOUT>** et implémente la méthode **map(LongWritable key, Text value, Context context)** .

Exemple de montant des ventes par magasin :

L’input de mapper est la paire (LongWritable,Text).

L’output de mapper est la paire (Text,FloatWritable) :la clé est le magasin de type texte et la valeur est le montant de type float.

Donc la déclaration de mapper est la suivante :

**public class MapperExemple1 extends Mapper<LongWritable, Text, Text, FloatWritable>**

* La classe reducer hérite de **Reducer<Text, FloatWritable, Text, FloatWritable>** et implémente la méthode **reduce(Text key, Iterable<FloatWritable> values, Context context)**.
* La classe driver est responsable de l’exécution de MapReduce. L’objet Job permet de configurer Mapper, Reducer, InputFormat, OutputFormat etc.
* La différence entre le code écrit en python et java est qu’en java et durant le shuffle hadoop transforme l’output de map en (clé, toutes les valeurs qui correspondent à cette clé).

Exemple :

(« clé », [545525,668666])

Par contre en python on aura :

(« clé » ,545525)

(« clé », 668666)

Premier exemple : Le montant de ventes par magasin

* On a écrit le mapper de telle sorte que la clé soit le magasin et le montant de vente sera la valeur. Donc on extrait à l’aide de split sur les champs magasin et prix puis on les met dans l’output.
* Dans le reducer on parcourt les valeurs et on fait la somme puisque toutes les valeurs qui correspondent à la même clé sont rangées ensemble.
* Dans le driver on met la configuration nécessaire.
* On teste notre code en local c’est-à-dire le fichier purchases.txt sera sur notre disque puis on va le tester sur hdfs.
* On exporte le projet dans un fichier jar.
* On tape la commande **hadoop jar EX1.jar myinput myoutput**.
* Pour voir le résultat on tape **hadoop fs –get myinput** puis cd myinput et enfin **cat part-r-00000**.

Deuxième exemple : Le montant de ventes par produit

On va changer dans le code de mapper de manière à avoir le produit comme clé.

Troisième exemple : Le montant de ventes le plus élevé pour chaque magasin

On va garder le mapper du premier exemple et on va changer le reducer.

On doit parcourir les valeurs et trouver leur maximum.

Quatrième exemple : Le nombre de ventes montant de ventes de tous magasins confondus

Dans cet exemple le mapper va avoir la même clé pour tous les couples et la valeur est la somme des montants de vente.

Le reducer va faire la somme de toutes les sommes partielles fournies par le map et va calculer le nombre de vente en utilisant un compteur.