


BIG DATA
2016-2017

TRAVAUX PRATIQUES N°4

MAP/REDUCE 2

NAJI ZAOU
YAHYA BACHIR



Le but de ce Travail était de réaliser un histogramme des villes données dans le fichier "worldcitiespop.txt", cet histogramme contiendra les catégories des villes, le nombre de villes dans chaque catégorie, la population moyenne des villes de chaque catégorie, et le minimum et maximum population des villes. Finalement, il fallait permettre à l'utilisateur de spécifier la largeur des catégories.

Exercice 1:

Dans l'exercice 1 l'objectif était de réaliser un histogramme simple qui ne contient que les catégories 10, 100, 1000, ... et le nombre de villes contenues dans chaque catégorie.

Dans le Mapper:

Pour cela nous avons utilisé le Logarithme dans la base 10 en lui donnant comme argument la population de la ville pour voir à quelle catégorie elle appartient:

```
(int)Math.pow(10,(int)Math.log10(Double.parseDouble(Data[4])));
```

Si la ville contient une population entre [100,1000] cette ligne de code renvoie 1000, et de même pour les autres catégories.

On met le retour de cette ligne de code comme key, et la population de la ville comme value, dans le couple (key,value) que le Mapper va envoyer au Reducer.

Dans le Reducer:

Le Reducer après avoir reçu les données du Mapper regroupe les valeurs selon leurs clés: (key,List<value>).

Pour chaque clé / catégorie, on parcourt la liste des valeurs, et pour chaque itération dans cette liste on incrémente notre variable "total" par 1. A la fin, la variable total contiendra le nombre de ville de la catégorie, et on passe à la deuxième catégorie, et ainsi de suite.

Finalement, on écrit dans le fichier de sortie le couple (key,value) avec key = catégorie et value = le nombre de ville dans cette catégorie.

Exercice 2:

Dans ce second exercice l'objectif était de faire un résumé qui contient la moyenne, le maximum, minimum de la population des villes de chaque catégorie.

Dans Le Mapper:

On ne changera pas notre code dans le mapper.

Dans le Reducer:

Dans le reducer on crée de nouvelles variables pour pouvoir stocker nos données, une pour la moyenne, une pour la somme de la population dans cette catégorie, une pour la population maximum et une autre pour le minimum.

Quand on parcourt la liste des valeurs regroupées par leurs clés, on fait maintenant :

- On incrémente total par 1.

- On incrémente la variable Somme par le nombre de population

```
Sum += iteration.get();
```

- On teste si la valeur de cette itération est supérieure au dernier Maximum stocké, si oui on la lui affecte.

-On test si la valeur de cette itération est inférieure au dernier Minimum stocké, si oui on la lui affecte.

A la Fin, on calcul la moyenne de population des villes : $avg = \text{Sum} / \text{total}$; vu que Sum et total contiennent respectivement la somme de population dans cette catégorie et le total de ville dans la catégorie.

Finalement, on affiche la clé / catégorie et une chaîne de caractères qui contient toutes les valeurs calculées pour chaque catégorie.

Exercice 3:

Le but de cet exercice est de donner à l'utilisateur la possibilité de bien fragmenter les catégories, au lieu des catégories initial 10,100,1000,10000 ... on pourra avoir des catégories du genre : 10,20,30,40.....,100,200,300,....,10000,..

Pour cela, on attend que l'utilisateur rentre un argument de plus à côté du fichier d'entrée et le fichier de sortie, cet argument servira à préciser le nombre de fragmentations désiré par l'utilisateur:

- A l'appel de la commande : `$yarn jar MonJar.jar args[0] args[1] args[2]`
- Puis dans le main on prend l'argument de l'utilisateur et on le met dans une variable de configuration "FragmentRate":

```
Configuration conf = new Configuration();  
conf.set("FragmentRate", args[0]);
```

Une fois notre paramètre récupéré, on peut maintenant l'utiliser dans notre mapper ou reducer, grâce à :

- `context.getConfiguration()` : qui nous retourne la configuration utilisée.
- `Configuration.get("MaVariable")` : qui nous retourne le contenu de MaVariable.

Dans le Mapper:

Pour faire la fragmentation on a suivi les étapes:

- Après calcul de l'algorithme de la population dans la base 10, et récupération du paramètre saisi par l'utilisateur. On calcul notre interval, qui sera la taille de nos nouvelles catégories.
- On calcul combien d'intervalle la population de notre ville couvre.
- Finalement on définit la clé, envoyé au reducer, comme étant le produit des deux.

Dans le Reducer:

On garde les mêmes fonctionnalités du Reducer de l'exercice 2

N.B : On avait compris qu'il fallait donner à l'utilisateur la possibilité de préciser la base du logarithme qu'on va utiliser.