# Projet TPA – Hadoop Map Reduce

Groupe 3

Membre de l'équipe:

- Nour DRIDI
- Anthony MALVESIN
- Jules VESNAT
- Quentin BERTRAMO

Code source: https://github.com/QuentinIUTNice/M2_MIAGE_TPA/tree/main/Hadoop

## Table des matières

## Introduction

Pour cette partie Hadoop, 2 parties distinctes sont à distinguer, la manipulation/refactoring du fichier CO2.csv et l'ajout des moyennes calculées par marque dans Catalogue.csv.

Pour la première partie, un programme Hadoop map/reduce en java à été mis en place utilisant de toute évidence le fichier CO2.csv ainsi qu'une liste de Marques (*makes.txt*) qui nous permettra de séparer la marque du modèle.

Le programme permettant d'intégrer ces moyennes dans le fichier Catalogue.csv à été réalisé avec un script R.

## Adaptation du fichier CO2.csv

Plusieurs manipulations ont été effectuées sur le fichier CO2.csv

Tout d'abord, nous avons séparé la première colonne qui auparavant contenait la marque et le modèle et qui maintenant sont respectivement dans leurs propres colonnes. Pour se faire, le programme Hadoop prend en 2ème argument un fichier qui liste toutes les marques de voiture à travers le monde : *makes.txt* (le 1er argument étant le fichier CO2.csv). Ce fichier a été récupéré sur internet à cette adresse.

La première manipulation JAVA a été de mettre en majuscule toutes les marques de manière a s'aligner sur les marques du fichier CO2.csv

Le fichier CO2.csv comportait plusieurs erreurs de syntaxe :

Des virgules et des doubles quotes en trop :

```
1 usage
public CarWritable(String string) { //AUDI E-TRON SPORTBACK 55 (408ch) quattro,-6NBSP000€NBS
    string = string.replaceAll( regex: ", ", replacement: " "); //fixed: virgule en trop
    string = string.replaceAll( regex: "\"", replacement: ""); //fixed: double quote en trop
```

Des valeurs manquantes pour le bonus/malus remplacées par 0 :

```
public int get_bonus_malus_serialized(String bon_mal) { //"-6 000€ 1"
    String str = bon_mal.split( regex: "€")[0]; //+/-....
    try {
        str = str.split( regex: "\\+")[1].trim(); //Positive value
    } catch (Exception e) { }
    str = str.replaceAll( regex: "[^0-9-]", replacement: ""); //"-6000" //Accept only number and minus symbol

    if (str.equals("-") || str.equals("")) return 0; //Empty values
    return Integer.parseInt(str);
}
```

Des valeurs manquantes pour le rejet CO2 remplacées par 0 :

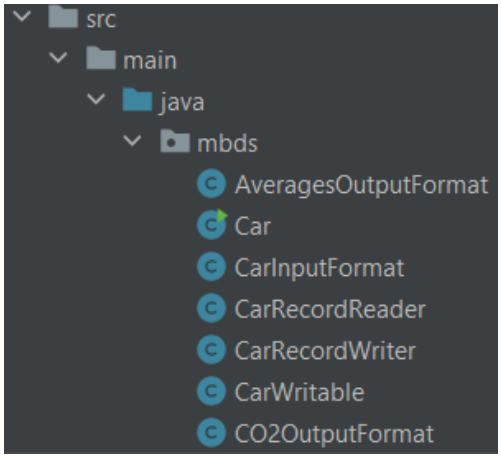
```
rejet = 0;
try {
    rejet = Integer.parseInt(elems[2]);
} catch (Exception e) { }
```

## Programme et exécution Hadoop

Le programme va avoir en sorti 2 fichiers :

- average_per_brand.csv -> Toutes les marques avec leurs moyennes
- co2_reformatted.csv -> Fichier CO2 corrigé

Voici l'architecture du programme map/reduce :




Dans le driver nous utilisons les MultipleOutputs objects afin de séparer les 2 fichiers en sortie

```
FileInputFormat.addInputPath(job, new Path(co2File));
FileOutputFormat.setOutputPath(job,new Path( pathString: "/car-map-reduce-output"));
MultipleOutputs.addNamedOutput(job, namedOutput: "averages", AveragesOutputFormat.class , LongWritable.class, Text.class);
MultipleOutputs.addNamedOutput(job, namedOutput: "co2", CO2OutputFormat.class, LongWritable.class, Text.class);
```

Pour récupérer les marques du fichier makes.txt nous utilisons les outils du filesystem puis les intégrons dans la configuration map reduce pour pouvoir les réutiliser dans le map.

```
FileSystem fs = new Path(makes_txt).getFileSystem(conf);
Path hdfsreadpath = new Path(makes_txt); // p-e le folder path ?
FSDataInputStream inputStream = fs.open(hdfsreadpath);
String makesList = IOUtils.toString(inputStream, encoding: "UTF-8");

conf.set("makesList", makesList); //Store makes.txt list in configuration
```

Pour séparer la marque et le modèle, pour chaque row (Voiture) dans le map, nous comparons la marque du fichier CO2.csv et les marques du fichier makes.txt

```
public void setRightProperties(List<String> makesList) {
    for (String brand : makesList) { //For each make of makes.txt file
        brand = brand.trim(); //Remove white space before and after
        if (marqueModele.contains(brand + " ") && marqueModele.length() > 0) {
            this.brand = brand;
            this.modele = marqueModele.replaceFirst(brand, replacement: "").trim();
        }
    }
}
```

Nous avons opté pour une solution fiable et bien plus lisible que de transférer des chaines de caractères entre le map/shuffle/reduce : Un objet voiture

```java
@Override
public void readFields(DataInput in) throws IOException {
    String line = in.readUTF();
    System.out.println("READING LINE " + line); //DS,DS3 CROSSBACK E-Tense (136ch),-6000,0,251
    String[] elems = line.split( regex: ",");
    brand = elems[0]; //MERCEDES AMG...
    System.out.println("MARQUE: " + brand); //DS
    modele = elems[1]; //MERCEDES AMG...
    System.out.println("MODELE: " + modele);
    bonus_malus = Integer.parseInt(elems[2]); //-6000
    System.out.println("BONUS: " + bonus_malus);
    rejet = Integer.parseInt(elems[3]);
    System.out.println("REJET: " + rejet);
    cout_energie = Integer.parseInt(elems[4]);
    System.out.println("COUT: " + cout_energie);
}
```

Une fois l'objet Voiture complété avec les bonnes valeurs, nous utilisons comme clé la marque de voiture et en valeur l'objet voiture :

```java
context.write(new Text(car.brand), car);
```

Dans le reduce, nous obtenons tous les modèles par marques. Nous calculons donc la moyenne et en profitons pour write tous les objets Car proprement dans un autre fichier en sortie.

```java
public void reduce(Text key, Iterable<CarWritable> values, Context context) throws IOException, InterruptedException {
    // ex. key = "MERCEDES"
    // ex. values = Object CarWritable
    if (key.equals(new Text( string: "999"))) {
        mos.write( namedOutput: "co2", new Text( string: "Marque"), columns_co2, baseOutputPath: "co2_reformated");
        mos.write( namedOutput: "averages", new Text( string: "Marque"), columns_averages, baseOutputPath: "averages_per_brand");
    } else {
        System.out.println("Key: \"" + key + "\"");
        int count = 0;
        int total_cout_energie = 0;
        int total_rejet = 0;
        int total_bonus_malus = 0;

        Iterator<CarWritable> i = values.iterator();
        while (i.hasNext()) {
            CarWritable car = i.next();
            total_cout_energie += car.cout_energie;
            total_rejet += car.rejet;
            total_bonus_malus += car.bonus_malus;
            System.out.println("Car: " + car.get_serialized());
            count++;
            mos.write( namedOutput: "co2", key, new Text(car.get_attributes()), baseOutputPath: "co2_reformated"); //Reformatted file
        }
        int moyenne_cout_energie = total_cout_energie / count;
        int moyenne_rejet = total_rejet / count;
        int moyenne_bonus_malus = total_bonus_malus / count;
        mos.write( namedOutput: "averages", key, new Text( string: String.valueOf(moyenne_cout_energie) + "," + String.valueOf(moyenne_
    }
}
```

## Programme R

Dans le but d'intégrer les moyennes par marque dans le fichier Catalogue, nous avons décidé d'écrire un programme en R, langage qui correspond tout à a fait à la manipulation et au traitement de données.

Dans un premier temps, le script prend en considération la première ligne comme le nom des colonnes pour éviter de faire des calculs inutiles :

df -> Dataframe average_per_brand.csv

```
header.true <- function(df) {
  names(df) <- as.character(unlist(df[1,]))
  df[-1,]
}
averages_per_brand <- header.true(averages_per_brand)
```

Nous créons les nouvelles colonnes à ajouter dans le fichier dataframe Catalogue :

```
empty_cols <- c("Moyenne cout energie", "Moyenne Rejet CO2", "Moyenne Bonus Malus")
Catalogue[ , empty_cols] <- NA
```

Pour finir nous parcourons toutes les lignes du fichier Catalogue et pour chaque marque nous ajoutons la moyenne associée venant du fichier average_per_brand.csv puis nous exportons sous le nom Catalogue_with_averages.csv :

```
for(i in 1:nrow(Catalogue)) {
  #Get row
  row <- Catalogue[i,]

  #Add value from averages_per_brand file to the good column id field
  row[10] <- averages_per_brand$"Moyenne cout energie"[averages_per_brand$"Marque"==toupper(row[1])]
  row[11] <- averages_per_brand$"Moyenne rejet CO2"[averages_per_brand$"Marque"==toupper(row[1])]
  row[12] <- averages_per_brand$"Moyenne Bonus Malus"[averages_per_brand$"Marque"==toupper(row[1])]

  #Replace modified row into Catalogue dataframe
  Catalogue[i,] <- row
}

write.csv(Catalogue,"D:\\Users\\Anthony\\Desktop\\M2_miage\\S1\\tpa\\Catalogue_with_averages.csv", row.names = FALSE, quote=FALSE)
```