Université Abderrahmane Mira de Bejaia
Faculté des Sciences Exactes
Département d'informatique
Année universitaire : 2022/2023

2ème Année Master IA (S3) Module : **Big Data**

TP3 : Map Reduce

Objectifs :

Le but de ce TP est d'écrire des fonctions map reduce pour une base de données Mangodb.

Consignes :

Le langage utilisé est le JavaScript. Il faut donc des notions dans celui-ci pour pouvoir programmer cet algorithme dans MongoDB. Vous pourrez trouver des ressources sur le net.

Au-delà de la syntaxe de JS, il faut noter que les deux fonctions map() et reduce() à passer en paramètre doivent respecter les contraintes suivantes :

map() : la fonction ne prend aucun paramètre, on accède à l'objet analysé via l'opérateur *this*, la fonction peut émettre des couples via la fonction *emit(key, value)* autant de fois que nécessaire dans la fonction

reduce() : la fonction prendre deux paramètres key et values (tableau des valeurs de la clé) elle peut être appeler plusieurs fois pour la même clé, elle doit donc renvoyer une valeur de même type que celles dans le tableau.

On doit passer un troisième paramètre, un littéral JSON, qui représente les options de la fonction. La principale option (out) est la collection dans laquelle le résultat sera placé. Si l'on veut voir le résultat, sans le stocker, il est possible d'indiquer out: { inline: 1 }.

Exemple 1 :

Dans l'exemple qui suit, nous allons chercher à calculer le nombre d'hommes et de femmes. Tout d'abord, on définit la fonction map(), qui émettra un couple (sexe, 1), pour chaque sportif.

```
var map1 = function() {
   emit(this.Sexe, 1)
};
```

Ensuite, nous définissons la fonction reduce(), qui fera la somme des valeurs (qui seront tous des 1 donc).

```
var red1 = function(key, values) {
   return Array.sum(values);
};
```

Ensuite, nous faisons appel à la fonction mapReduce() sur notre collection.

```
db.Sportifs.mapReduce(
   map1,
   red1,
   { out: { inline: 1 } }
)
```

On remarque qu'on a m et M. On peut résoudre ce problème en faisant une transformation (en majuscule ou en minuscule) dans la fonction map().

```
var map1 = function() {
   emit(this.Sexe.toUpperCase(), 1)
}
```

Au final, on peut aussi tout déclarer dans la fonction mapReduce(), en utilisant le principe des fonctions anonymes.

```
db.Sportifs.mapReduce(
   function() {
      emit(this.Sexe.toUpperCase(), 1)
   },
   function(key, values) {
      return Array.sum(values);
   },
   { out: { inline: 1 } }
)
```

Exemple 2 :

Si on cherche à calculer le nombre de sportifs jouant pour chaque sport, il faut émettre des couples (sport, 1) pour chaque sport joué, pour chaque sportif. Mais il faut d'une part prendre en compte qu'il existe des sportifs dans la base, qui ne joue aucun sport. Et d'autre part que pour certains sportifs, il n'y a qu'un sport, et que dans ce cas, nous avons juste une chaîne et non un tableau.

```
db.Sportifs.mapReduce(
   function() {
      if (this.Sports) { // le sportif joue
         if (typeof this.Sports.Jouer != "string") {
            for (sp of this.Sports.Jouer) {
               emit(sp, 1)
            }
         } else {
            emit(this.Sports.Jouer, 1)
         }
      }
   },
   function(key, values) {
      return Array.sum(values);
   },
   { out: { inline: 1 } }
)
```

Exercices : Répondez en utilisant le paradigme **Map-Reduce**

1. Calculer le nombre de gymnases pour chaque ville
2. Calculer le nombre de séances pour chaque jour de la semaine
3. De même pour chaque sport
4. Calculer la superficie moyenne des gymnases, pour chaque ville, pour cela, vous devez calculer la somme des superficies ET la nombre de gymnase (à émettre dans un même objet et à réduire en tenant compte que ce double aspect)
5. Calculer pour chaque sport, le nombre de séance pour chaque jour de la semaine
   - Il faudra émettre, pour chaque sport, une valeur complexe (littéral JSON pour le jour)
   - Il faudra réfléchir à l'étape de réduction