

TP4 – MapReduce avec MongoDB

Introduction au MapReduce sur MongoDB



NICOLAS Ethan

G5SI2
2025–2026

Table des matières

1	Introduction	2
2	MapReduce : exercices	2
2.1	1. Compter le nombre total de films	3
2.2	2. Compter le nombre de films par genre	3
2.3	3. Compter les films par réalisateur	3
2.4	4. Compter les acteurs uniques	3
2.5	5. Films par année de sortie	4
2.6	6. Note moyenne par film (à partir de grades)	4
2.7	7. Note moyenne par genre	4
2.8	8. Note moyenne par réalisateur	4
2.9	9. Film avec la note maximale	5
2.10	10. Nombre de notes > 70	5
2.11	11. Acteurs par genre (sans doublons)	5
2.12	12. Acteurs dans le plus grand nombre de films	6
2.13	13. Classement des films par grade majoritaire	6
2.14	14. Note moyenne par année	6
2.15	15. Réalisateurs avec moyenne > 80	7

1 Introduction

Ce TP vise à explorer MapReduce dans MongoDB en utilisant la collection de films du TP1. MongoDB permet d'exécuter des traitements distribués via trois éléments : une fonction `map`, une fonction `reduce`, et optionnellement une fonction `finalize`. Chaque question correspond à une transformation MapReduce appliquée sur les documents.

2 MapReduce : exercices

Dans tous les exemples, la commande générale exécutée est :

```
db.films.mapReduce(mapFunction, reduceFunction, {  
  out: "<nom_sortie>"  
})
```

2.1 1. Compter le nombre total de films

Map

```
function() { emit("total_films", 1); }
```

Reduce

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

2.2 2. Compter le nombre de films par genre

Map

```
function() {  
  if (this.genre) {  
    this.genre.forEach(g => emit(g, 1));  
  }  
}
```

Reduce

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

2.3 3. Compter les films par réalisateur

Map

```
function() {  
  if (this.director) emit(this.director, 1);  
}
```

Reduce

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

2.4 4. Compter les acteurs uniques

Map

```
function() {  
  if (this.actors) this.actors.forEach(a => emit(a, 1));  
}
```

Reduce

```
function(key, values) { return 1; }
```

2.5 5. Films par année de sortie

Map

```
function() { emit(this.year, 1); }
```

Reduce

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

2.6 6. Note moyenne par film (à partir de grades)

Map

```
function() {  
  if (this.grades) {  
    const mean = this.grades.reduce((a,b)=>a+b.score,0) / this.grades.length;  
    emit(this.title, mean);  
  }  
}
```

Reduce

```
function(k, v) { return Array.sum(v)/v.length; }
```

2.7 7. Note moyenne par genre

Map

```
function() {  
  if (this.grades && this.genre) {  
    const avg = this.grades.reduce((a,b)=>a+b.score,0)/this.grades.length;  
    this.genre.forEach(g => emit(g, avg));  
  }  
}
```

Reduce

```
function(k,v) { return Array.sum(v)/v.length; }
```

2.8 8. Note moyenne par réalisateur

Map

```
function() {  
  if (this.director && this.grades) {  
    const m = this.grades.reduce((a,b)=>a+b.score,0)/this.grades.length;  
    emit(this.director, m);  
  }  
}
```

```
}
```

Reduce

```
function(k,v) { return Array.sum(v)/v.length; }
```

2.9 9. Film avec la note maximale

Map

```
function() {  
  if (this.grades) {  
    const max = Math.max(...this.grades.map(g => g.score));  
    emit(this.title, max);  
  }  
}
```

Reduce

```
function(k, v) { return Math.max(...v); }
```

2.10 10. Nombre de notes > 70

Map

```
function() {  
  if (this.grades) {  
    const c = this.grades.filter(g => g.score > 70).length;  
    emit("notes_sup_70", c);  
  }  
}
```

Reduce

```
function(k,v) { return Array.sum(v); }
```

2.11 11. Acteurs par genre (sans doublons)

Map

```
function() {  
  if (this.genre && this.actors) {  
    this.genre.forEach(g =>  
      this.actors.forEach(a => emit(g, a))  
    );  
  }  
}
```

Reduce

```
function(k,v) { return Array.from(new Set(v)); }
```

2.12 12. Acteurs dans le plus grand nombre de films

Map

```
function(){ if (this.actors) this.actors.forEach(a => emit(a, 1)); }
```

Reduce

```
function(k,v){ return Array.sum(v); }
```

2.13 13. Classement des films par grade majoritaire

Map

```
function(){  
  if (this.grades) {  
    const freq = {};  
    this.grades.forEach(g => freq[g.grade] = (freq[g.grade]||0)+1);  
    const major = Object.keys(freq).reduce((a,b)=>freq[a]>freq[b]?a:b);  
    emit(major, this.title);  
  }  
}
```

Reduce

```
function(k,v){ return v; }
```

2.14 14. Note moyenne par année

Map

```
function(){  
  if (this.grades && this.year) {  
    const avg = this.grades.reduce((a,b)=>a+b.score,0)/this.grades.length;  
    emit(this.year, avg);  
  }  
}
```

Reduce

```
function(k,v){ return Array.sum(v)/v.length; }
```

2.15 15. Réalisateurs avec moyenne > 80

Map

```
function(){  
  if (this.director && this.grades) {  
    const m = this.grades.reduce((a,b)=>a+b.score,0)/this.grades.length;  
    emit(this.director, {sum:m, count:1});  
  }  
}
```

Reduce

```
function(k,v){  
  let s=0, c=0;  
  v.forEach(x => { s+=x.sum; c+=x.count; });  
  return {sum:s, count:c};  
}
```

Finalize

```
function(k,v){  
  const avg = v.sum / v.count;  
  return avg > 80 ? avg : null;  
}
```