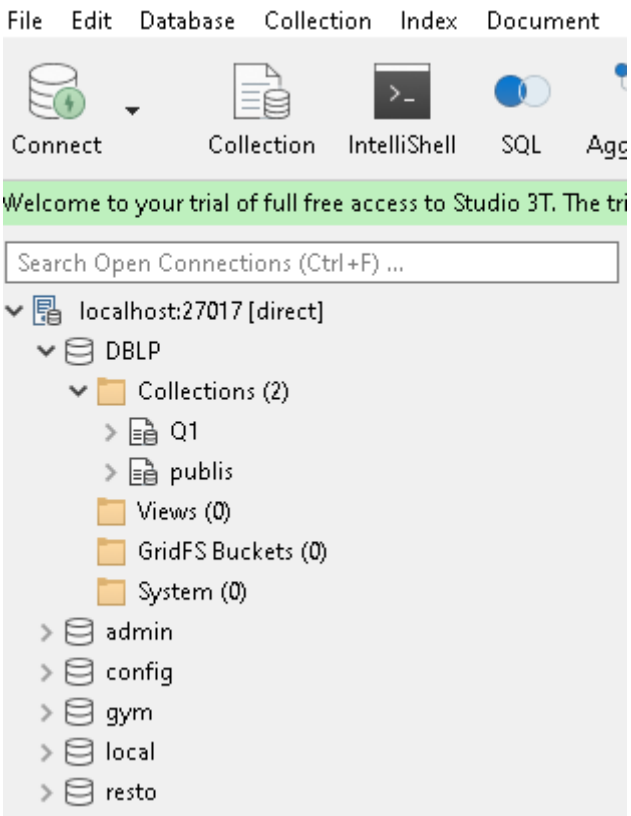
TP Map/reduce avec MongoDB (mode Stand alone)

L'objectif est ici de construire des requêtes map/reduce sur une instance serveur MongoDB en mode standalone.

Ce TP utilise les données de la base dblp.json , à importer dans votre MongoDB , sauf si cela est déjà fait.

Pour élaborer vos script vous pourrez le faire directement avec le shell Mongo ou bien utiliser RoboMongo – qui est un très bon client. Il constitue un bon compromis entre interface graphique et exécution de commandes MongoDB :

https://robomongo.org/



Import de la base DLP depuis RoboMongo

Pour les recherches suivantes, donnez la requête MapReduce sur la base en changeant la requête Map et/ou Reduce

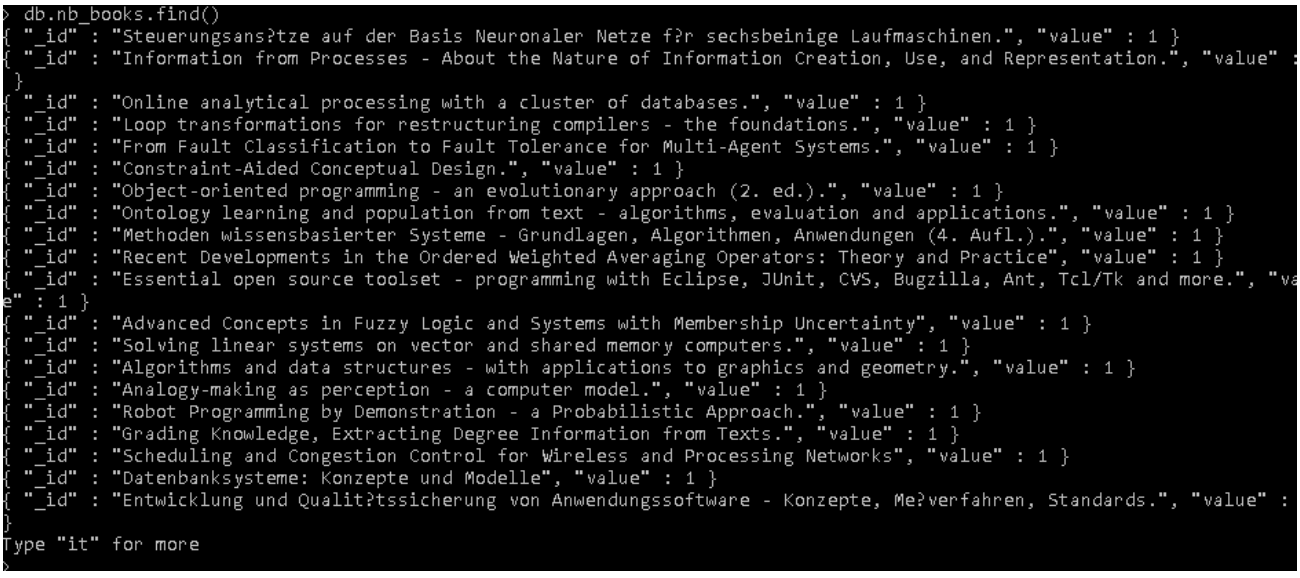
1. **Pour chaque document de type livre, émettre le document avec pour clé "title" et donner le nombre d'instances documents de ce livre ; compter le nombre de document en entrée de la fonction map, et ceux en sortie de la fonction reduce : qu'en concluez-vous ?**

```
db.dblp.mapReduce(
        function(){
                emit(this.title, 1);
                },

        function(key, values){
                return Array.sum(values)},{query:{type:"Book"},out:"nb_books »}
                )
```

Soit 11 403 documents

Dans le mongo shell, en appliquant la fonction « find », on obtient le résultat suivant à titre d'exemple.



J'ai réalisé ainsi les deux manipulations (avec Robot et avec Mongo Shell) pour le reste des requêtes.

2. **Pour chacun de ses livres, donner le nombre de ses auteurs**

```
db.dblp.mapReduce(
        function() {
                emit( this.title , this.authors.length);
                },
        function(key, values) {
                return Array.sum(values) }, { query : { "type": "Book"}, out : "
        nb_of_authors_by_book " } ) { "result" : " nb_of_authors_by_book ",
        "ok" : 1 }
                )
```

3. Pour chaque document ayant "booktitle" (chapitre) publié par Springer, donner le nombre de ses chapitres.

```
db.dblp.mapReduce(
        function(){
                emit(this.title, this.booktitle);
                } ,
        function(key, values){
                return values} ,{query:{publisher:"Springer"},out:"nb_springer"})
```

4. Pour chaque éditeur "Springer", donner le nombre de publication par année

```
db.dblp.mapReduce(
        function(){
                emit(this.year, 1);
        } ,
        function(key, values){
                return Array.sum(values)}
        ,{query:{publisher:"Springer"},out:"q4"})
```

5. Pour chaque clé "publisher & année" (pour ceux qui ont un publisher), donner le nombre de publications

```
var mapFunction = function() {
   var key = { pub: this.publisher, year:this.year }
   var value = 1; emit(key, value);
   };
```

```
db.dblp.mapReduce(mapFunction ,function(key, values){
return Array.sum(values)} ,{out:"q5"}

)
```

6. **Pour l'auteur "Toru Ishida", donner le nombre de publication par année**

```
var mapFunction = function() {
        var key = this.year
        var value = 1;
        emit(key, value);
};

db.dblp.mapReduce(mapFunction ,function(key, values){
    return Array.sum(values)} ,{
    query:{authors:"Toru Ishida"
    },out:"q6"})
```

7. **Pour l'auteur "Toru Ishida", donner le nombre moyen de pages pour ses articles (type Article)**

```
var mapFunction = function() {
        var key = this.type
        var value = this.pages.end - this.pages.start;
        emit(key, value);
};
db.dblp.mapReduce(mapFunction ,function(key, values){
    return Array.avg(values)
    } ,{query:{authors:"Toru Ishida",type:"Article"},out:"q7"})
```

```
db.q7.find()
{ "_id" : "Article", "value" : 11.285714285714286 }
```

8. **Pour chaque auteur, donner le titre de ses publications**

```
var mapFunction = function() {

for (var idx = 0; idx < this.authors.length; idx++) {
```

```
        var key = this.authors[idx];
        var value = this.title;
        emit(key, value);
        }
};
db.dblp.mapReduce(mapFunction ,function(key, values){return values}
,{out:"q8"})
```

9. Pour chaque auteur, donner le nombre de publications associé à chaque année

```
var mapFunction = function() {
for (var idx = 0; idx < this.authors.length; idx++) {
        var key = this.authors[idx];
        var value = {year:this.year,title:this.title, count:1};
        emit(key, value); } };
dn
var reduceFunction = function(key, values) {
reducedVal = {year:0, count:0} ;

for (var idx = 0; idx < values.length; idx++) {
reducedVal.count += values[idx].count; reducedVal.year = values[idx].year; }
return reducedVal; };

db.dblp.mapReduce(mapFunction ,reduceFunction ,{out:"q9"})
```

```
> db.q9.find()
{ "_id" : "Francesca Palumbo", "value" : { "year" : 2007, "count" : 1 } }
{ "_id" : "Andrew Burn", "value" : { "year" : 2010, "count" : 1 } }
{ "_id" : "Jos? A. Rodrigues Nt.", "value" : { "year" : 2009, "count" : 1 } }
{ "_id" : "Florian Berger", "value" : { "year" : 2008, "count" : 3 } }
{ "_id" : "Alexandra Potapova", "value" : { "year" : 2010, "count" : 1 } }
{ "_id" : "Lorenzo Palma", "value" : { "year" : 2014, "count" : 2 } }
```

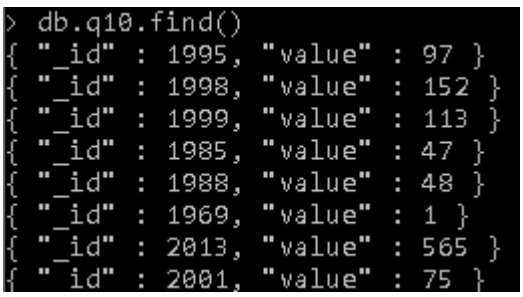
10. Pour l'éditeur "Springer", donner le nombre d'auteurs par année

```
var mapFunction = function() {
for (var idx = 0; idx < this.authors.length; idx++) {
        var key = this.year;
        var value = {author:this.authors[idx], count: 1};
```

```
        emit(key, value); } };

var reduceFunction = function(key, values) {
reducedVal = 0 ;
for (var idx = 0; idx < values.length; idx++) {
        reducedVal += values[idx].count; }
return reducedVal; };

 db.dblp.mapReduce(mapFunction ,reduceFunction
,{query:{publisher:"Springer"},out:"q10"})
```



11. Compter les publications de plus de 3 auteurs

```
db.dblp.mapReduce(function(){
emit(this.title, this.authors.length);} ,function(key, values){
return values[0];} ,{out:"q11"})

db.q11.find({value:{$gte:3}})
```

12. Donner pour chaque publieur, donner le nombre moyen de pages par publication

```
var mapFunction = function() {
var key = this.publisher;
var value = this.pages.end - this.pages.start;
emit(key, value); }

db.dblp.mapReduce(mapFunction ,function(key, values){ return
Array.avg(values)} ,{query:{pages:{$exists:true}},out:"q12"})
```

```
> db.q12.find()
{ "_id" : "Pearson", "value" : NaN }
{ "_id" : "Dryden Press", "value" : NaN }
{ "_id" : "Idea Group Publishing", "value" : NaN }
{ "_id" : "GOCMAR", "value" : NaN }
{ "_id" : "Yourdon Press", "value" : NaN }
{ "_id" : "Verlag f?r Geschichte der Naturwissenschaften und der Technik", "value" : 502 }
{ "_id" : "Institut Fiziko-Techniceskoj Informatiki", "value" : NaN }
{ "_id" : "Omnigena Verlag", "value" : NaN }
{ "_id" : "Deutscher Wissenschaftsverlag", "value" : 291 }
{ "_id" : "Deutscher Universit?tsverlag", "value" : NaN }
{ "_id" : "Rotbuch Verlag", "value" : 255.5 }
```

13. Pour chaque auteur, donner le minimum et le maximum des années, ainsi que le nombre de publication totale

```
var mapFunction = function() {
 for (var idx = 0; idx < this.authors.length; idx++) {
var key = this.authors[idx];
var value = {year:this.year,title:this.title, count:1};
emit(key, value); } };


var reduceFunction = function(key, values) {
reducedVal = {min_year:9999, max_year:0 ,count:0};
for (var idx = 0; idx < values.length; idx++) {
reducedVal.count = values[idx].count;
 if (values[idx].year < reducedVal.min_year) { reducedVal.min_year = values[idx].year; }
if (values[idx].year > reducedVal.max_year) { reducedVal.max_year = values[idx].year; }
}
return reducedVal; };


db.dblp.mapReduce(mapFunction ,reduceFunction ,{out:"q13"}
```