Exercice 1 :

```
 ./mongoimport.exe -d movies-db -c movies --file C:/data/movies.json
$ ./mongoimport.exe -d movies-db -c users --file C:/data/user.json
```

Exercices 2 :

```
db.getCollection('users').count()

db.getCollection('movies').count();

db.getCollection('users').find({age : {$gt :  18 , $lte : 30}  })

db.getCollection('users').find({$or : [{occupation : "artist" }, {occupation : "scientist" }}]})

db.users.find({gender : 'F', occupation : 'writer'}).sort({age : -1}).limit(10);

db.users.distinct("occupation");
```

Exercice 3 :

Question 1 :

```
db.users.find(
{
  name : 'Zineb',
    gender :  'F',
    age : 30,
    occupation : 'programmer'
}
)
```

Question 2 :

```
db.users.deleteMany({ name: 'Zineb'})

db.users.deleteOne({ name: 'Zineb'})


db.users.updateMany({occupation : 'programmer'},

{$set : {occupation : 'developper'}})
```

```
var myfuction =  function(doc) {
      var str = doc.title.split("(");
      var myyear  = str[1].substring(0,4);
      var newTitle = str[0];

      db.movies.updateMany(
         {title : doc.title},
         {'$set' : {'year' : myyear}}
      );

      db.movies.updateMany(
         {_id : doc._id},
         {'$set' : {'title' : newTitle}}
      );

};
db.movies.find().forEach(myfuction);
```

```
var myFunction = function(doc) {
   var genresArray  = doc.genres.split("|");
   //print(genresArray);
   doc.genres = genresArray;
   db.movies.save(doc);
}
db.movies.find().forEach(myFunction);
```

Modifier la collection users en remplaçant pour chaque utilisateur le champ timestamp par un nouveau champ date, de type Date.

```
var modificationDate = function(doc){
    doc.movies.forEach(function(movie){
        movie.timestamp = new Date(movie.timestamp);
    })
    db.movies.save(doc);
}
db.users.find().forEach(modificationDate);
```

```
db.movies.aggregate(
[
{$match : {$and :[
   { year : {$gt :  "1899"}},
   { year : {$lt :  "2000"}}
   ] }},
{$group : {_id : "$year"  , nombreFilms : {$sum : 1} }}
]
)
```

```
db.movies.aggregate(
[
{$match : {$and :[
   { year : {$gt :  "1899"}},
   { year : {$lt :  "2000"}}
   ] }},
{$group : {_id : "$year"  , nombreFilms : {$sum : 1} }},
{ $sort: { nombreFilms : -1}}
]
)
```

```
db.users.aggregate([
  {$match :{ 'movies.movieid': 296,'movies.rating' : { $gt : 0}} },
  {
    $project: {
      movies: {
        $filter: {
          input: "$movies",
          as: "item",
          cond: { $eq: [ "$$item.movieid", 296 ] }
        }
      },
      _id: 0
    }
  },
  { $unwind: "$movies" },
  {$group : {_id :'$movies.movieid', moyenne : {$sum : '$movies.rating'} }}
])
```

```
db.users.aggregate([
  { $unwind: "$movies" },
  {$group : {_id :'$_id', name : {$first : '$name'} ,name : {$first : '$name'},notemax : {$max :
'$movies.rating'},
  notemin : {$min : '$movies.rating'}, moyenne : {$avg : '$movies.rating'} }},
  {$sort : {'moyenne' : 1}}
]
```

```
var functionmapB = function() {
  if(this.movies && this._id)
  for (var idx = 0; idx < this.movies.length; idx++) {
    emit(this._id, this.movies[idx].rating);
  }
}
var functionReduceB = function(key, values) {
   var maxNote = 0;
   for (var idx = 0; idx < values.length; idx++) {
      if(values[idx] > maxNote){
         maxNote = values[idx];
      }
    }
   return maxNote;
}
db.users.mapReduce(functionmapB, functionReduceB,
{
   out : "resultat"}


   );
```

```
var functionmapB = function() {
  if(this.movies && this._id){
     for (var idx = 0; idx < this.movies.length; idx++) {
        var obj = {
              note : this.movies[idx].rating,
              film : this.movies[idx].movieid
           };
       emit(new Date(this.movies[idx].timestamp), obj);
     }
   }
```

```
}
var functionReduceB = function(key, values) {
   var maxNote = 0;
   var obj = {note : 0, film : 0};
   for (var idx = 0; idx < values.length; idx++) {
      if(values[idx].note > maxNote){
         obj.note = values[idx].note;
         obj.film = values[idx].film;
      }
    }
   return obj;
}
db.users.mapReduce(functionmapB, functionReduceB,
{

   out : "resultat_max"}

   ).explain();
```