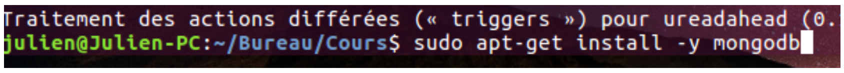
Compte rendu du TP MongoDB dans l'instance Openstack

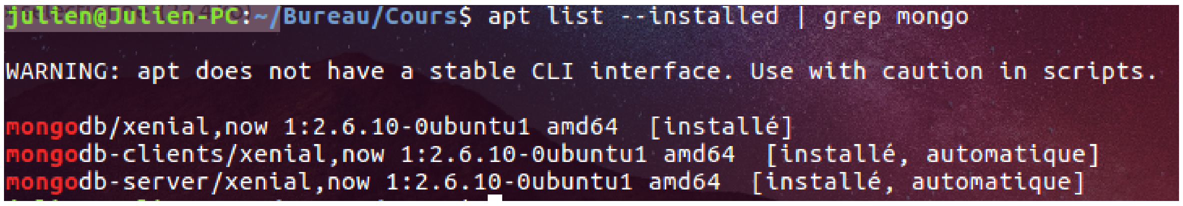
Julien GIRAUD, Thomas LIEGHIO
G2S4
UNIVERSITÉ LYON 1

1.1



```
Traitement des actions différées (« triggers ») pour ureadahead (0.
julien@Julien-PC:~/Bureau/Cours$ sudo apt-get install -y mongodb
```

1.2

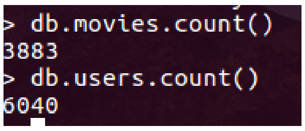


```
julien@Julien-PC:~/Bureau/Cours$ apt list --installed | grep mongo

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

mongodb/xenial,now 1:2.6.10-0ubuntu1 amd64  [installé]
mongodb-clients/xenial,now 1:2.6.10-0ubuntu1 amd64  [installé, automatique]
mongodb-server/xenial,now 1:2.6.10-0ubuntu1 amd64  [installé, automatique]
```

3.4
6040 pour users.json et 3883 pour movies.json

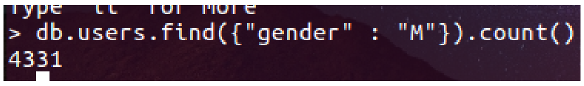


```
> db.movies.count()
3883
> db.users.count()
6040
```

4.1
Cette commande parcourt les les « users » et s'arrête une fois passé le premier film. Elle nous affiche ce qu'elle a parcouru au format json.

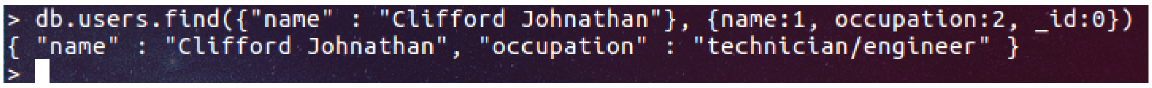
6.1.1
Voir la 3.4

6.1.2



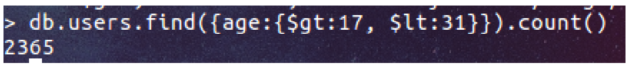
```
> db.users.find({"gender" : "M"}).count()
4331
```

6.1.3



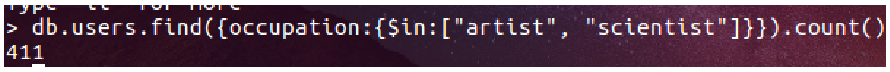
```
> db.users.find({"name" : "Clifford Johnathan"}, {name:1, occupation:2, _id:0})
{ "name" : "Clifford Johnathan", "occupation" : "technician/engineer" }
>
```

6.1.4



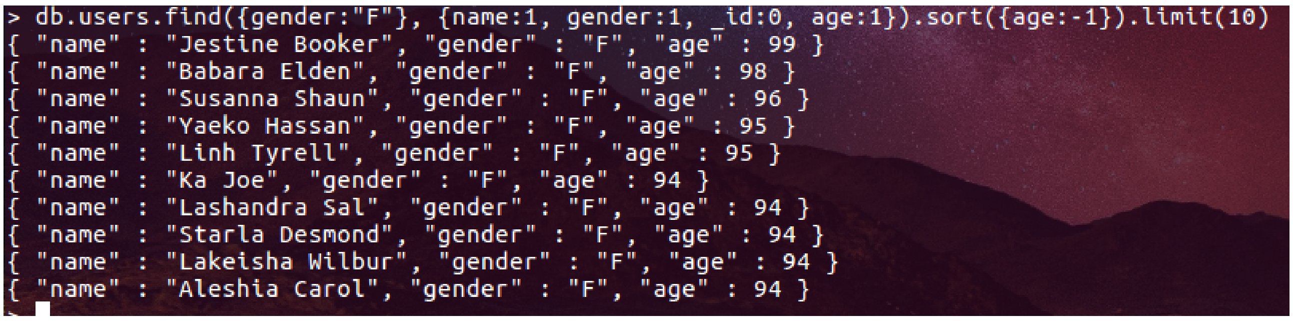
```
> db.users.find({age:{$gt:17, $lt:31}}).count()
2365
```

6.1.5



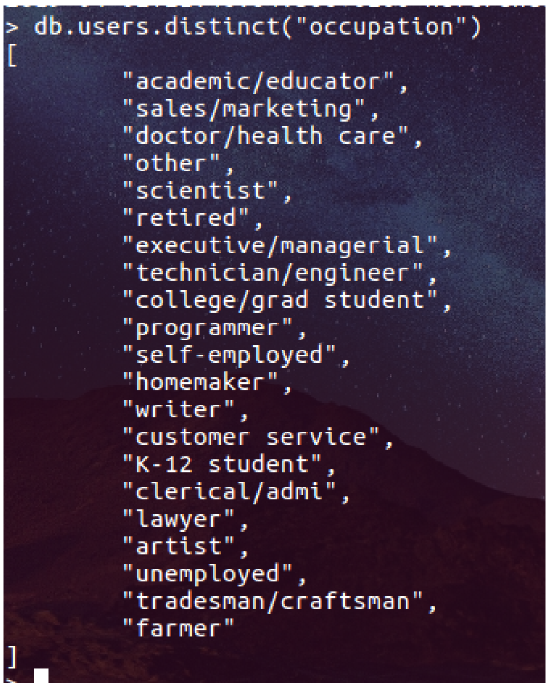
```
> db.users.find({occupation:{$in:["artist", "scientist"]}}).count()
411
```

6.1.6



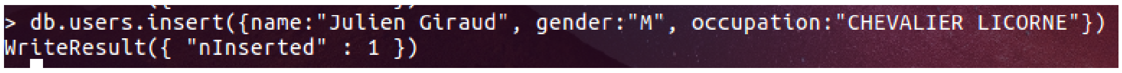
```
> db.users.find({gender:"F"}, {name:1, gender:1, _id:0, age:1}).sort({age:-1}).limit(10)
{ "name" : "Jestine Booker", "gender" : "F", "age" : 99 }
{ "name" : "Babara Elden", "gender" : "F", "age" : 98 }
{ "name" : "Susanna Shaun", "gender" : "F", "age" : 96 }
{ "name" : "Yaeko Hassan", "gender" : "F", "age" : 95 }
{ "name" : "Linh Tyrell", "gender" : "F", "age" : 95 }
{ "name" : "Ka Joe", "gender" : "F", "age" : 94 }
{ "name" : "Lashandra Sal", "gender" : "F", "age" : 94 }
{ "name" : "Starla Desmond", "gender" : "F", "age" : 94 }
{ "name" : "Lakeisha Wilbur", "gender" : "F", "age" : 94 }
{ "name" : "Aleshia Carol", "gender" : "F", "age" : 94 }
```

6.1.7



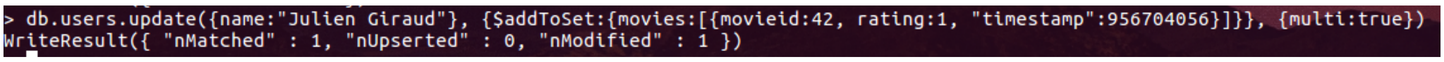
```
> db.users.distinct("occupation")
[
        "academic/educator",
        "sales/marketing",
        "doctor/health care",
        "other",
        "scientist",
        "retired",
        "executive/managerial",
        "technician/engineer",
        "college/grad student",
        "programmer",
        "self-employed",
        "homemaker",
        "writer",
        "customer service",
        "K-12 student",
        "clerical/admi",
        "lawyer",
        "artist",
        "unemployed",
        "tradesman/craftsman",
        "farmer"
]
```

6.2.1



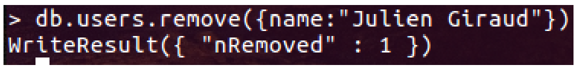
```
> db.users.insert({name:"Julien Giraud", gender:"M", occupation:"CHEVALIER LICORNE"})
WriteResult({ "nInserted" : 1 })
```

6.2.2



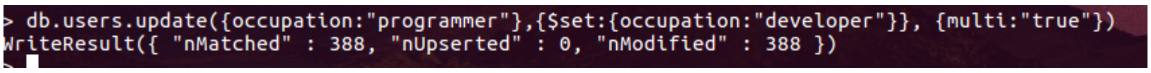
```
> db.users.update({name:"Julien Giraud"}, {$addToSet:{movies:[{movieid:42, rating:1, "timestamp":956704056}]}}, {multi:true})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

6.2.3



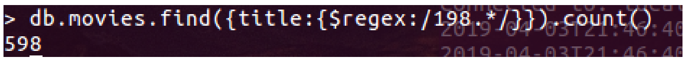
```
> db.users.remove({name:"Julien Giraud"})
WriteResult({ "nRemoved" : 1 })
```

6.2.4



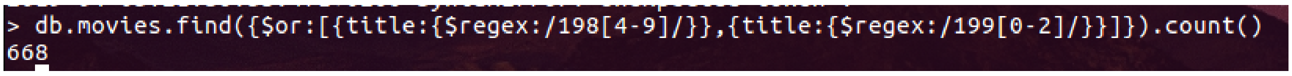
```
> db.users.update({occupation:"programmer"},{$set:{occupation:"developer"}}, {multi:"true"})
WriteResult({ "nMatched" : 388, "nUpserted" : 0, "nModified" : 388 })
```

7.1



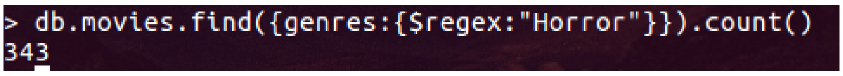
```
> db.movies.find({title:{$regex:/198.*/}}).count()
598
```

7.2



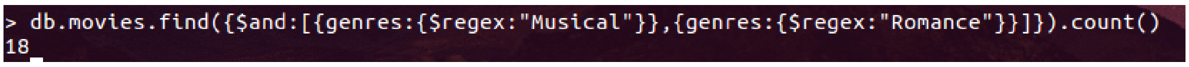
```
> db.movies.find({$or:[{title:{$regex:/198[4-9]/}},{title:{$regex:/199[0-2]/}}]}).count()
668
```

7.3



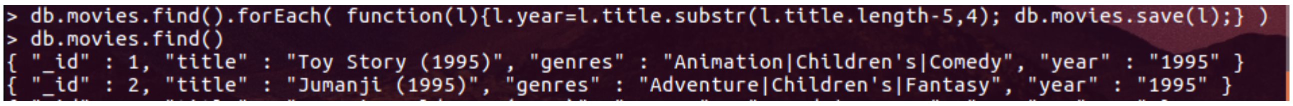
```
> db.movies.find({genres:{$regex:"Horror"}}).count()
343
```

7.4



```
> db.movies.find({$and:[{genres:{$regex:"Musical"}},{genres:{$regex:"Romance"}}]}).count()
18
```

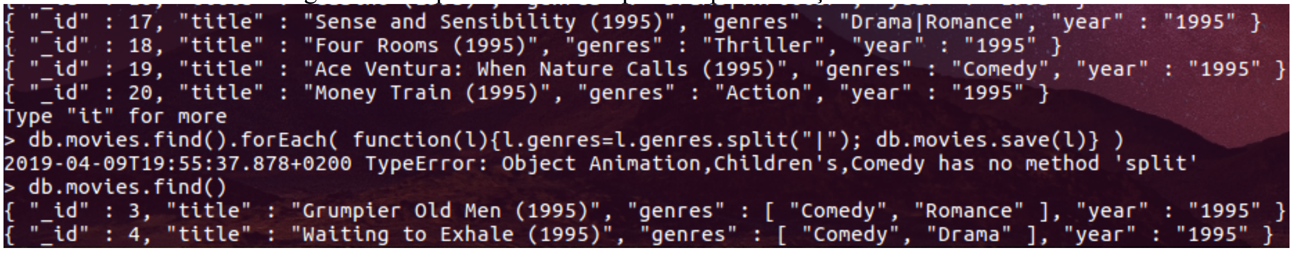
8.1



```
> db.movies.find().forEach( function(l){l.year=l.title.substr(l.title.length-5,4); db.movies.save(l);} )
> db.movies.find()
{ "_id" : 1, "title" : "Toy Story (1995)", "genres" : "Animation|Children's|Comedy", "year" : "1995" }
{ "_id" : 2, "title" : "Jumanji (1995)", "genres" : "Adventure|Children's|Fantasy", "year" : "1995" }
```

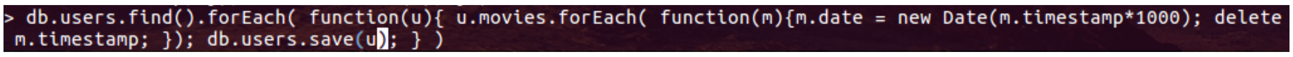
8.2

Nous avons eu un message d'erreur que nous n'avons pas compris mais ça a marché.



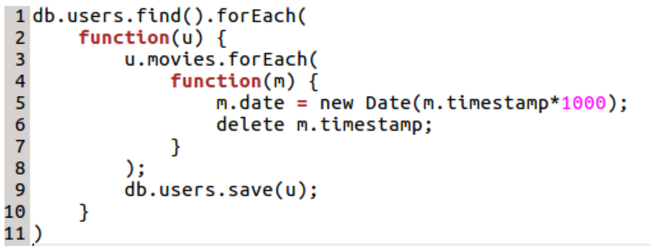
```
{ "_id" : 17, "title" : "Sense and Sensibility (1995)", "genres" : "Drama|Romance", "year" : "1995" }
{ "_id" : 18, "title" : "Four Rooms (1995)", "genres" : "Thriller", "year" : "1995" }
{ "_id" : 19, "title" : "Ace Ventura: When Nature Calls (1995)", "genres" : "Comedy", "year" : "1995" }
{ "_id" : 20, "title" : "Money Train (1995)", "genres" : "Action", "year" : "1995" }
Type "it" for more
> db.movies.find().forEach( function(l){l.genres=l.genres.split("|"); db.movies.save(l)} )
2019-04-09T19:55:37.878+0200 TypeError: Object Animation,Children's,Comedy has no method 'split'
> db.movies.find()
{ "_id" : 3, "title" : "Grumpier Old Men (1995)", "genres" : [ "Comedy", "Romance" ], "year" : "1995" }
{ "_id" : 4, "title" : "Waiting to Exhale (1995)", "genres" : [ "Comedy", "Drama" ], "year" : "1995" }
```

8.3



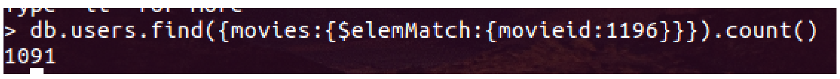
```
> db.users.find().forEach( function(u){ u.movies.forEach( function(m){m.date = new Date(m.timestamp*1000); delete
 m.timestamp; }); db.users.save(u); } )
```

Version propre :



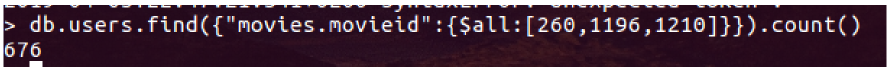
```
db.users.find().forEach(
    function(u) {
        u.movies.forEach(
            function(m) {
                m.date = new Date(m.timestamp*1000);
                delete m.timestamp;
            }
        );
        db.users.save(u);
    }
)
```

9.1.1



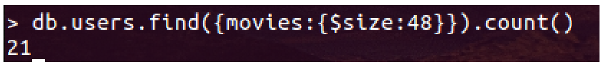
```
> db.users.find({movies:{$elemMatch:{movieid:1196}}}).count()
1091
```

9.1.2



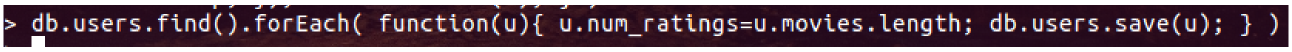
```
> db.users.find({"movies.movieid":{$all:[260,1196,1210]}}).count()
676
```

9.1.3



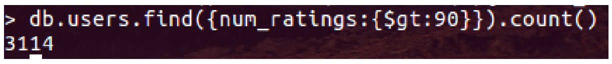
```
> db.users.find({movies:{$size:48}}).count()
21
```

9.1.4



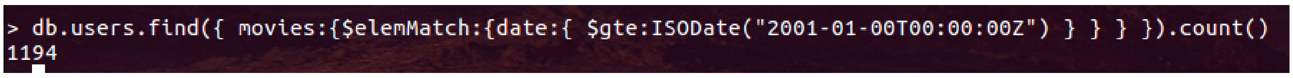
```
> db.users.find().forEach( function(u){ u.num_ratings=u.movies.length; db.users.save(u); } )
```

9.1.5



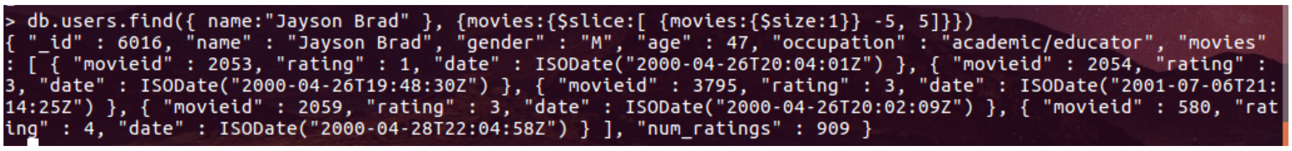
```
> db.users.find({num_ratings:{$gt:90}}).count()
3114
```

9.1.6



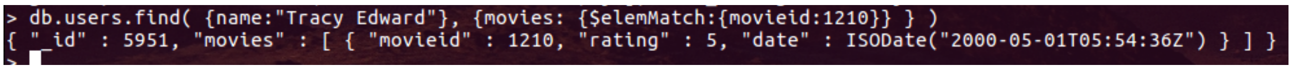
```
> db.users.find({ movies:{$elemMatch:{date:{ $gte:ISODate("2001-01-00T00:00:00Z") } } } }).count()
1194
```

9.1.7



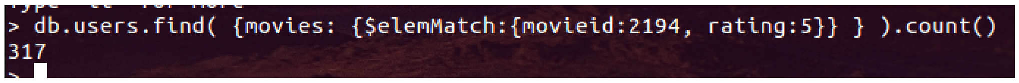
```
> db.users.find({ name:"Jayson Brad" }, {movies:{$slice:[ {movies:{$size:1}} -5, 5]}})
{ "_id" : 6016, "name" : "Jayson Brad", "gender" : "M", "age" : 47, "occupation" : "academic/educator", "movies"
: [ { "movieid" : 2053, "rating" : 1, "date" : ISODate("2000-04-26T20:04:01Z") }, { "movieid" : 2054, "rating" :
3, "date" : ISODate("2000-04-26T19:48:30Z") }, { "movieid" : 3795, "rating" : 3, "date" : ISODate("2001-07-06T21:
14:25Z") }, { "movieid" : 2059, "rating" : 3, "date" : ISODate("2000-04-26T20:02:09Z") }, { "movieid" : 580, "rat
ing" : 4, "date" : ISODate("2000-04-28T22:04:58Z") } ], "num_ratings" : 909 }
```

9.1.8



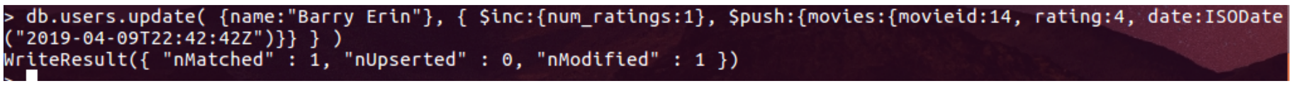
```
> db.users.find( {name:"Tracy Edward"}, {movies: {$elemMatch:{movieid:1210}} } )
{ "_id" : 5951, "movies" : [ { "movieid" : 1210, "rating" : 5, "date" : ISODate("2000-05-01T05:54:36Z") } ] }
```

9.1.9



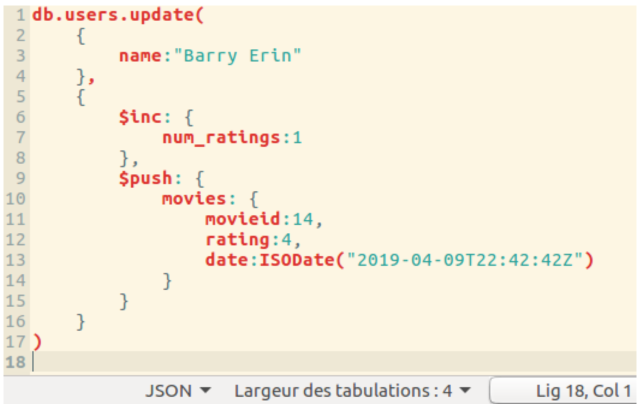
```
> db.users.find( {movies: {$elemMatch:{movieid:2194, rating:5}} } ).count()
317
```

9.2.1



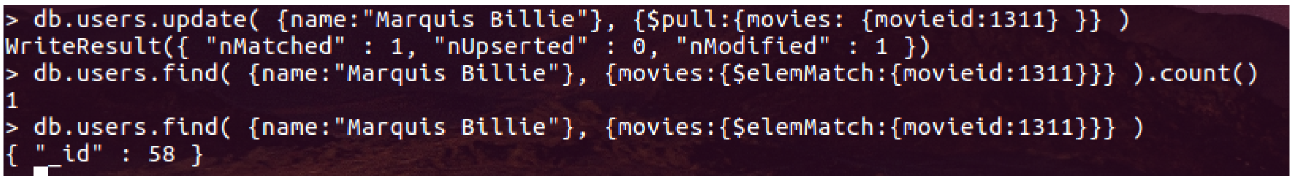
```
> db.users.update( {name:"Barry Erin"}, { $inc:{num_ratings:1}, $push:{movies:{movieid:14, rating:4, date:ISODate
("2019-04-09T22:42:42Z")}} } )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Version propre :



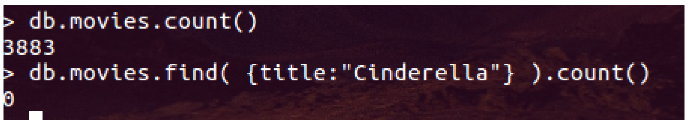
```
db.users.update(
    {
        name:"Barry Erin"
    },
    {
        $inc: {
            num_ratings:1
        },
        $push: {
            movies: {
                movieid:14,
                rating:4,
                date:ISODate("2019-04-09T22:42:42Z")
            }
        }
    }
)
```

9.2.2
Nous n'avons pas vérifié si le film était présent mais si c'était le cas, il n'y est plus.



```
> db.users.update( {name:"Marquis Billie"}, {$pull:{movies: {movieid:1311} }} )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.users.find( {name:"Marquis Billie"}, {movies:{$elemMatch:{movieid:1311}}} ).count()
1
> db.users.find( {name:"Marquis Billie"}, {movies:{$elemMatch:{movieid:1311}}} )
{ "_id" : 58 }
```

9.2.3
En soit nous aurions bien voulu mettre à jour les champs du film Cinderella mais vu qu'il n'est pas présent dans la base de données il n'y a pas de champs à mettre à jour.



```
> db.movies.count()
3883
> db.movies.find( {title:"Cinderella"} ).count()
0
```