TP sur MongoDB

Inspired from labs by P.Rigaux et al.

Solutions

Solutions for Section 1

```
docker run hello-world
docker images
docker ps -a
docker pull mongo:latest
docker run  -d --name mongoserv mongo mongod
docker exec -it mongoserv mongo

(we do not need to run docker pull, actually, and mongod is actually the default program in the mongo i
```

```
use bdtest
show dbs
db.movies.insert({'a': 120})
show dbs
show collections
db.movies.find()
exit
```

```
docker cp enron.json mongoserv:enron.json
docker exec  mongoserv mongoimport --db bdenron --collection emails --file enron.json
```

Solutions for Section 2

I do not write instructions such as `exit` or `docker rm -f`

```
docker exec -it mongoserv mongo
```

```
use bdenron
db.emails.count()
db.emails.find({'sender': 'no.address@enron.com'})

var myCursor = db.emails.find({'sender': 'no.address@enron.com'}).sort({'to': 1})
var documentArray = myCursor.toArray();
print(tojson(documentArray[9]))

db.emails.find({'sender': 'no.address@enron.com'}).sort({'to': 1}).skip(9).limit(1).pretty()
db.emails.distinct('folder')
db.emails.find({},{"sender": 1, "to":1}).sort({"sender":1}).pretty()

db.emails.find({date: {$gte: "2001-04-01", $lt: "2002-01-01"}})

db.emails.find({"replyto": {$not: {$type: 'null'}}})
db.emails.find({"replyto": {$ne : null}})

db.emails.aggregate(
    [
      { $match: {"date": {$gte: "2001-04-01", $lt: "2002-01-01"}}},
      { $group: { "_id": "$sender", "nb_mails": { $sum: 1}  }  },
      { $sort: {'nb_mails': -1}}
    ])

var mapf1 = function () {emit(this.sender, 1);}
var redf1 = function (sendId, nb) {return Array.sum(nb);}
db.emails.mapReduce(
mapf1,
redf1,
{out: "nbmails"},
{query: {"date": {$gte: "2001-04-01", $lt: "2002-01-01"}}}
)
```

```
// ou :
db.emails.mapReduce(
function () {emit(this.sender, 1);},
function (sendId, nb) {return Array.sum(nb);},
{out: "nbmails",
query: {"date": {$gte: "2001-04-01", $lt: "2002-01-01"}}
}
)
// db.nbmails.find().sort({"value":-1})

// retourne les docs dont dernière ligne contient "president Bush" ou "President Bush".
db.emails.find( { text: {$regex: '^(?si).*P(?-si)resident Bush.*$'} } ).pretty()
```

Solutions for Section 3

```
docker rm -f mongoserv1
docker rm -f mongoserv2
docker rm -f mongoserv3
docker run --name mongoserv1 --net host mongo mongod --replSet repl --port 27018
docker run --name mongoserv2 --net host mongo mongod --replSet repl --port 27019
docker run -d  --name mongoserv3 --net host mongo mongod --replSet repl --port 27020
docker attach mongoserv3

docker exec -it mongoserv1 mongo --port 27018
```

```
rs.initiate()
rs.add ("a-140395.ups.u-psud.fr:27019")
rs.add ("a-140395.ups.u-psud.fr:27020")

rs.status()['member']
rs.status().members.forEach(function(z) {printjson(z._id+": "+z.stateStr+" - "+z.name)})
```

```
docker cp enron.json mongoserv1:enron.json
docker exec  mongoserv1 \
    mongoimport --port 27018 --db bdenron --collection emails --file enron.json
```

```
rs.slaveOk()
db.shutdownServer()
```

Solutions for Section 4

```
docker run -d --name mconfig --net host mongo \
    mongod --configsvr --replSet serveurconfig --port 27018
docker exec -it  mconfig mongo --port 27018
```

```
rs.initiate()
```

```
docker run -d --name mrouteur --net host mongo mongos --port 27017 \
  --configdb serveurconfig/$(hostname):27018

docker run -d --name mshard1 --net host mongo mongod --shardsvr --replSet rset1 --port 27030
docker run -d --name mshard2 --net host mongo mongod --shardsvr --replSet rset2 --port 27031
docker exec -it  mshard1 mongo --port 27030
```

```
rs.initiate()
```

```
docker exec -it  mshard2 mongo --port 27031
```

```
rs.initiate()
```

```
docker exec -it  mrouteur mongo --port 27017
```

```
sh.addShard("rset1/a-140395.ups.u-psud.fr:27030")
sh.addShard("rset2/a-140395.ups.u-psud.fr:27031")
db.adminCommand( { enableSharding: "bdpart" } )
sh.shardCollection("bdpart.dblp", {"ident": "hashed"} )
```

```
sed -i.back 's/"_id"/"ident"/g' dblp.json
docker cp dblp.json mrouteur:dblp.json
docker exec mrouteur \
    mongoimport --port 27017 --db bdpart --collection dblp --file dblp.json --jsonArray
```

```
use bdpart
db.dblp.find({'type': 'Article'}).sort({authors:1}).explain()
```