Héma LAMBERT

TP MongoDb

PREMIÈRE PARTIE

1. Vérifiez qu'aucun processus mongo tourne actuellement sur votre machine.

```
$ ps -aux | grep mongo
```

Si c'est le cas, arretez le**.**

```
$ kill -9 [pid du processus]
```

Ensuite lancez une instance mongod avec le dbpath par défaut. Connectez vous sur le shell mongo et affichez le port utilisé et les infos du host depuis le shell.

```
$ mkdir /data/db
$ mongod
$ mongo
MongoDB shell version: 2.4.9
connecting to: test
```

2. Arretez le processus depuis le shell.

```
> use admin
> db.shutdownServer()
```

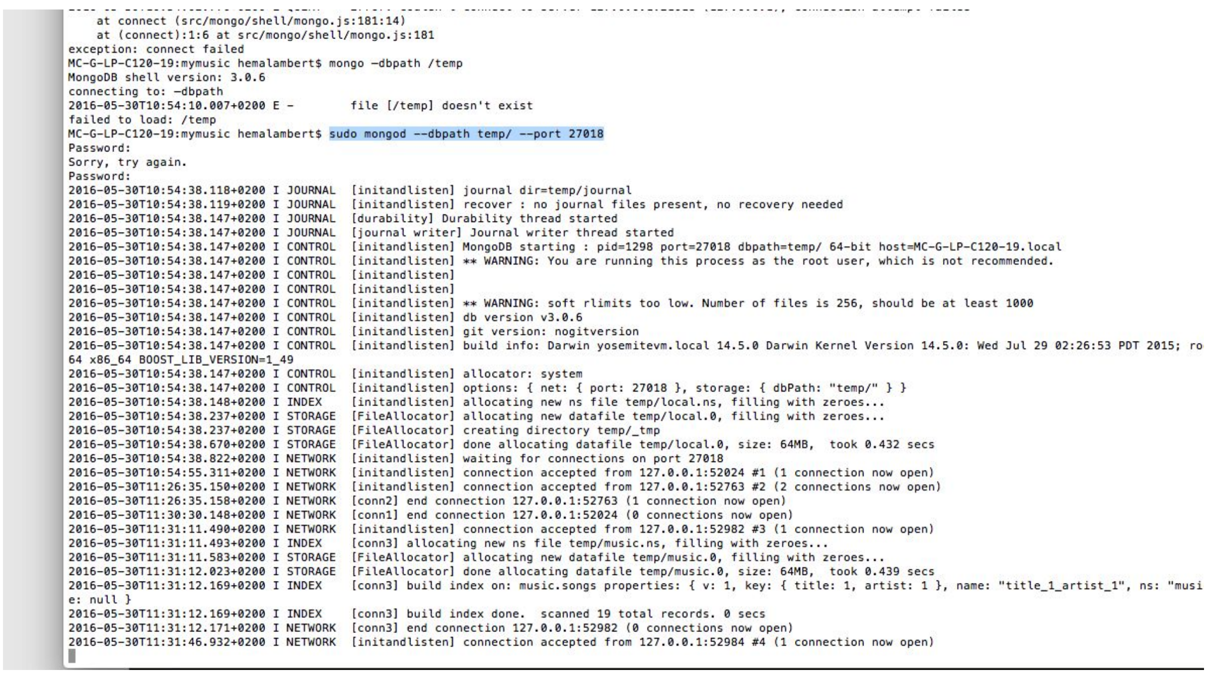
3. Lancez à nouveau une instance de mongod mais cette fois, modifiez le dbpath et le fichier de sortie de logs. Connectez vous sur le shell et affichez les infos utilisées pour la configuration du processus. Vérifiez aussi que les logs sont bien écrit dans le fichier avec un tail f ou un cat

```
$ mkdir /temp/data
$ sudo mongod --dbpath /temp --port 27018
$ mongo --port 27018
```

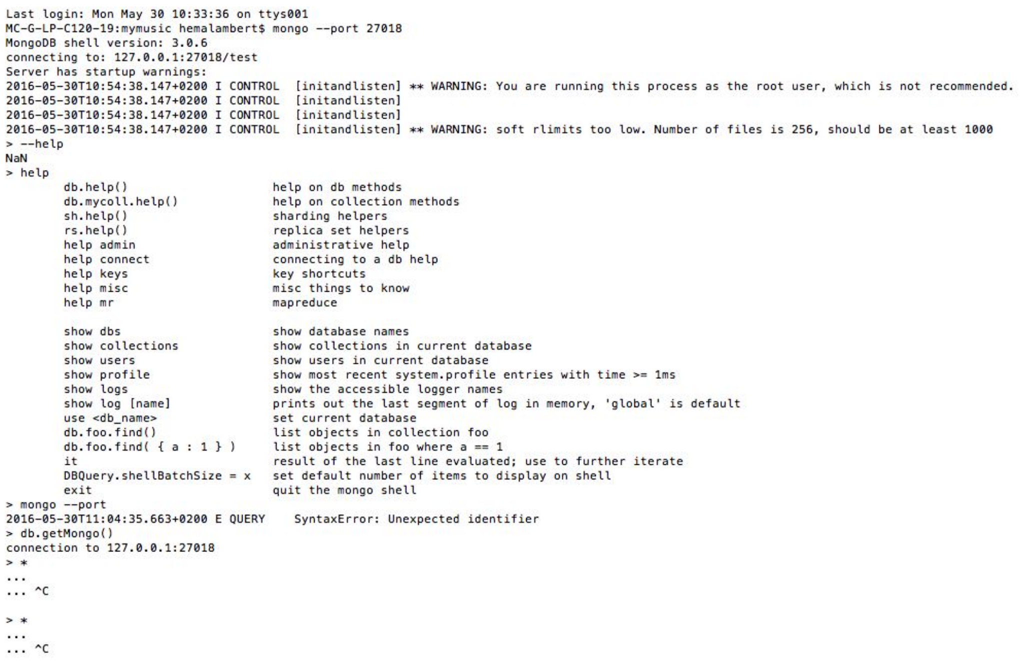
myPort et hostname() permettent de connaitre le port utilisé et le nom du host.

```
> myPort()
27018
> hostname()
MC-G-LP-C120-19.local
```

Héma LAMBERT



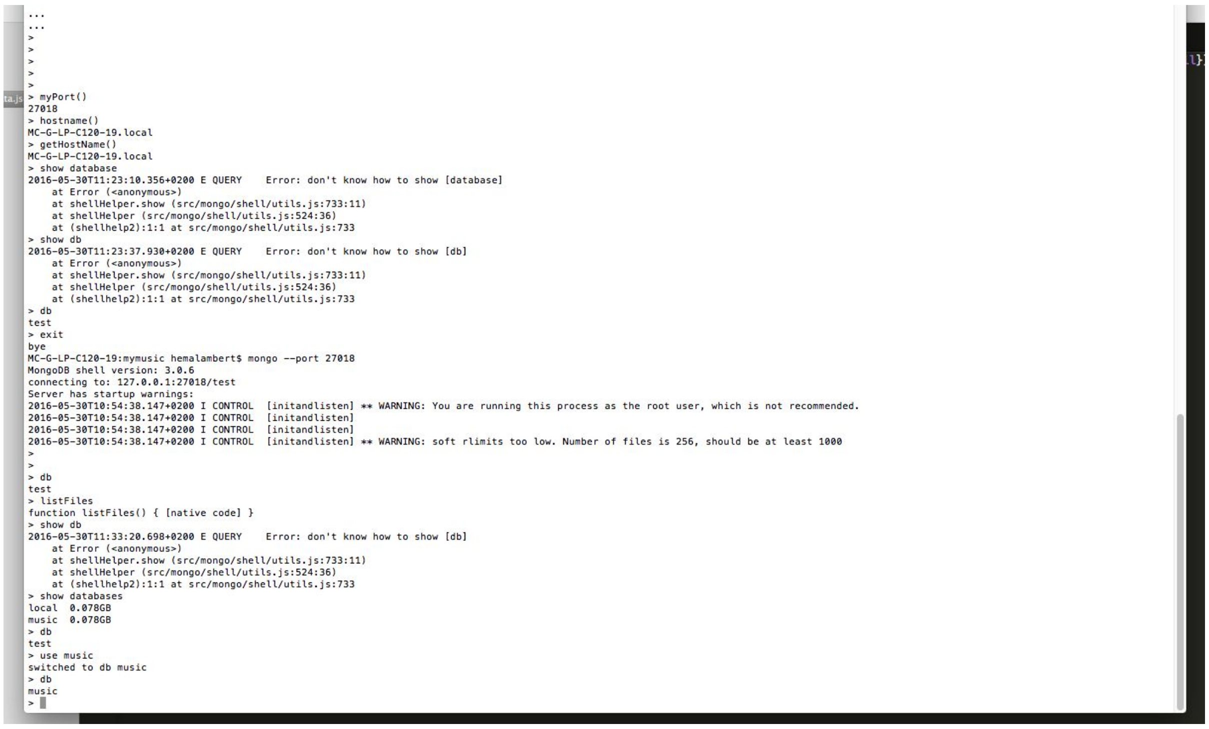
```
    at connect (src/mongo/shell/mongo.js:181:14)
    at (connect):1:6 at src/mongo/shell/mongo.js:181
exception: connect failed
MC-G-LP-C120-19:mymusic hemalambert$ mongo —dbpath /temp
MongoDB shell version: 3.0.6
connecting to: —dbpath
2016-05-30T10:54:10.007+0200 E -        file [/temp] doesn't exist
failed to load: /temp
MC-G-LP-C120-19:mymusic hemalambert$ sudo mongod --dbpath temp/ --port 27018
Password:
Sorry, try again.
Password:
2016-05-30T10:54:38.118+0200 I JOURNAL  [initandlisten] journal dir=temp/journal
2016-05-30T10:54:38.119+0200 I JOURNAL  [initandlisten] recover : no journal files present, no recovery needed
2016-05-30T10:54:38.147+0200 I JOURNAL  [durability] Durability thread started
2016-05-30T10:54:38.147+0200 I JOURNAL  [journal writer] Journal writer thread started
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] MongoDB starting : pid=1298 port=27018 dbpath=temp/ 64-bit host=MC-G-LP-C120-19.local
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be at least 1000
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] db version v3.0.6
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] git version: nogitversion
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] build info: Darwin yosemitevm.local 14.5.0 Darwin Kernel Version 14.5.0: Wed Jul 29 02:26:53 PDT 2015; ro
64 x86_64 BOOST_LIB_VERSION=1_49
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] allocator: system
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] options: { net: { port: 27018 }, storage: { dbPath: "temp/" } }
2016-05-30T10:54:38.148+0200 I INDEX    [initandlisten] allocating new ns file temp/local.ns, filling with zeroes...
2016-05-30T10:54:38.237+0200 I STORAGE  [FileAllocator] allocating new datafile temp/local.0, filling with zeroes...
2016-05-30T10:54:38.237+0200 I STORAGE  [FileAllocator] creating directory temp/_tmp
2016-05-30T10:54:38.670+0200 I STORAGE  [FileAllocator] done allocating datafile temp/local.0, size: 64MB,  took 0.432 secs
2016-05-30T10:54:38.822+0200 I NETWORK  [initandlisten] waiting for connections on port 27018
2016-05-30T10:54:55.311+0200 I NETWORK  [initandlisten] connection accepted from 127.0.0.1:52024 #1 (1 connection now open)
2016-05-30T11:26:35.150+0200 I NETWORK  [initandlisten] connection accepted from 127.0.0.1:52763 #2 (2 connections now open)
2016-05-30T11:26:35.158+0200 I NETWORK  [conn2] end connection 127.0.0.1:52763 (1 connection now open)
2016-05-30T11:30:30.148+0200 I NETWORK  [conn1] end connection 127.0.0.1:52024 (0 connections now open)
2016-05-30T11:31:11.490+0200 I NETWORK  [initandlisten] connection accepted from 127.0.0.1:52982 #3 (1 connection now open)
2016-05-30T11:31:11.493+0200 I INDEX    [conn3] allocating new ns file temp/music.ns, filling with zeroes...
2016-05-30T11:31:11.583+0200 I STORAGE  [FileAllocator] allocating new datafile temp/music.0, filling with zeroes...
2016-05-30T11:31:12.023+0200 I STORAGE  [FileAllocator] done allocating datafile temp/music.0, size: 64MB,  took 0.439 secs
2016-05-30T11:31:12.169+0200 I INDEX    [conn3] build index on: music.songs properties: { v: 1, key: { title: 1, artist: 1 }, name: "title_1_artist_1", ns: "musi
e: null }
2016-05-30T11:31:12.169+0200 I INDEX    [conn3] build index done.  scanned 19 total records. 0 secs
2016-05-30T11:31:12.171+0200 I NETWORK  [conn3] end connection 127.0.0.1:52982 (0 connections now open)
2016-05-30T11:31:46.932+0200 I NETWORK  [initandlisten] connection accepted from 127.0.0.1:52984 #4 (1 connection now open)
```



```
Last login: Mon May 30 10:33:36 on ttys001
MC-G-LP-C120-19:mymusic hemalambert$ mongo --port 27018
MongoDB shell version: 3.0.6
connecting to: 127.0.0.1:27018/test
Server has startup warnings:
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be at least 1000
> --help
NaN
> help
        db.help()                    help on db methods
        db.mycoll.help()             help on collection methods
        sh.help()                    sharding helpers
        rs.help()                    replica set helpers
        help admin                   administrative help
        help connect                 connecting to a db help
        help keys                    key shortcuts
        help misc                    misc things to know
        help mr                      mapreduce

        show dbs                     show database names
        show collections             show collections in current database
        show users                   show users in current database
        show profile                 show most recent system.profile entries with time >= 1ms
        show logs                    show the accessible logger names
        show log [name]              prints out the last segment of log in memory, 'global' is default
        use <db_name>                set current database
        db.foo.find()                list objects in collection foo
        db.foo.find( { a : 1 } )     list objects in foo where a == 1
        it                           result of the last line evaluated; use to further iterate
        DBQuery.shellBatchSize = x   set default number of items to display on shell
        exit                         quit the mongo shell
> mongo --port
2016-05-30T11:04:35.663+0200 E QUERY    SyntaxError: Unexpected identifier
> db.getMongo()
connection to 127.0.0.1:27018
> *
...
... ^C

> *
...
... ^C
```



```
...
...
>
>
>
>
>
> myPort()
27018
> hostname()
MC-G-LP-C120-19.local
> getHostName()
MC-G-LP-C120-19.local
> show database
2016-05-30T11:23:10.356+0200 E QUERY    Error: don't know how to show [database]
    at Error (<anonymous>)
    at shellHelper.show (src/mongo/shell/utils.js:733:11)
    at shellHelper (src/mongo/shell/utils.js:524:36)
    at (shellhelp2):1:1 at src/mongo/shell/utils.js:733
> show db
2016-05-30T11:23:37.930+0200 E QUERY    Error: don't know how to show [db]
    at Error (<anonymous>)
    at shellHelper.show (src/mongo/shell/utils.js:733:11)
    at shellHelper (src/mongo/shell/utils.js:524:36)
    at (shellhelp2):1:1 at src/mongo/shell/utils.js:733
> db
test
> exit
bye
MC-G-LP-C120-19:mymusic hemalambert$ mongo --port 27018
MongoDB shell version: 3.0.6
connecting to: 127.0.0.1:27018/test
Server has startup warnings:
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten]
2016-05-30T10:54:38.147+0200 I CONTROL  [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be at least 1000
>
>
> db
test
> listFiles
function listFiles() { [native code] }
> show db
2016-05-30T11:33:20.698+0200 E QUERY    Error: don't know how to show [db]
    at Error (<anonymous>)
    at shellHelper.show (src/mongo/shell/utils.js:733:11)
    at shellHelper (src/mongo/shell/utils.js:524:36)
    at (shellhelp2):1:1 at src/mongo/shell/utils.js:733
> show databases
local  0.078GB
music  0.078GB
> db
test
> use music
switched to db music
> db
music
> 
```

4. Faites l’import des données contenues dans le fichier zip donnée par l’enseignant afin de construire une base de données appelé “music”.

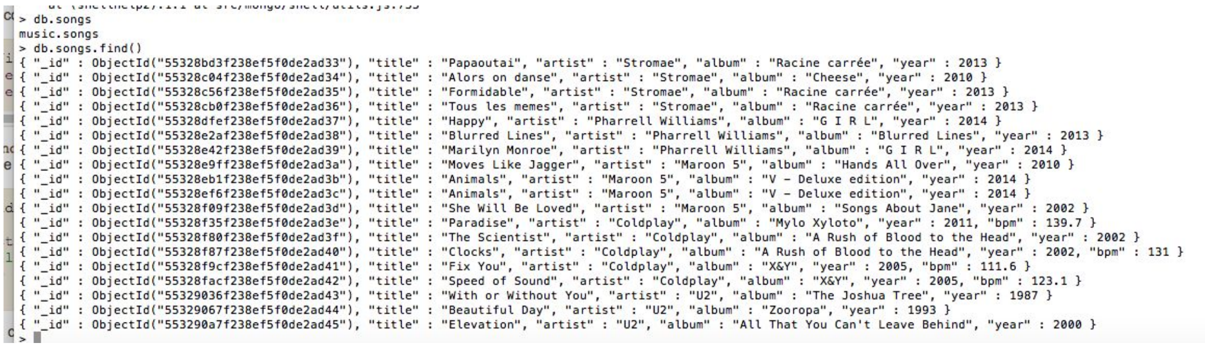
```
$ mongorestore --host=127.0.0.1 --post=27018 --db=music --drop mymusic
$ mongo --port 27018
> db
test
> show databases
local     0.078GB
music   0.078GB
> use music
switched to db music
> db
music
```

On utilise mongorestore pour importer et construire la base de données music
“db” permet de connaître la base de donnée actuellement utilisée
“show databases” permet de lister toutes les bases de données disponible
“use” permet de choisir la base de donnée à utiliser

DEUXIÈME PARTIE

1. Affichez les documents de la collection songs.

> db.songs.find()



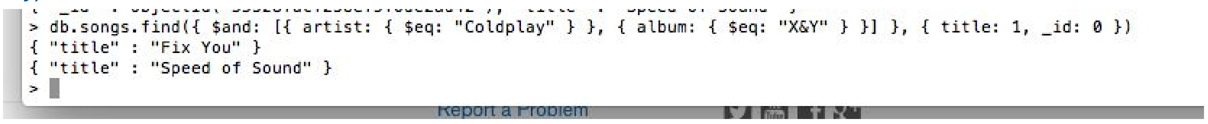
```
> db.songs
music.songs
> db.songs.find()
{ "_id" : ObjectId("55328bd3f238ef5f0de2ad33"), "title" : "Papaoutai", "artist" : "Stromae", "album" : "Racine carrée", "year" : 2013 }
{ "_id" : ObjectId("55328c04f238ef5f0de2ad34"), "title" : "Alors on danse", "artist" : "Stromae", "album" : "Cheese", "year" : 2010 }
{ "_id" : ObjectId("55328c56f238ef5f0de2ad35"), "title" : "Formidable", "artist" : "Stromae", "album" : "Racine carrée", "year" : 2013 }
{ "_id" : ObjectId("55328cb0f238ef5f0de2ad36"), "title" : "Tous les memes", "artist" : "Stromae", "album" : "Racine carrée", "year" : 2013 }
{ "_id" : ObjectId("55328dfef238ef5f0de2ad37"), "title" : "Happy", "artist" : "Pharrell Williams", "album" : "G I R L", "year" : 2014 }
{ "_id" : ObjectId("55328e2af238ef5f0de2ad38"), "title" : "Blurred Lines", "artist" : "Pharrell Williams", "album" : "Blurred Lines", "year" : 2013 }
{ "_id" : ObjectId("55328e42f238ef5f0de2ad39"), "title" : "Marilyn Monroe", "artist" : "Pharrell Williams", "album" : "G I R L", "year" : 2014 }
{ "_id" : ObjectId("55328e9ff238ef5f0de2ad3a"), "title" : "Moves Like Jagger", "artist" : "Maroon 5", "album" : "Hands All Over", "year" : 2010 }
{ "_id" : ObjectId("55328eb1f238ef5f0de2ad3b"), "title" : "Animals", "artist" : "Maroon 5", "album" : "V - Deluxe edition", "year" : 2014 }
{ "_id" : ObjectId("55328ef6f238ef5f0de2ad3c"), "title" : "Animals", "artist" : "Maroon 5", "album" : "V - Deluxe edition", "year" : 2014 }
{ "_id" : ObjectId("55328f09f238ef5f0de2ad3d"), "title" : "She Will Be Loved", "artist" : "Maroon 5", "album" : "Songs About Jane", "year" : 2002 }
{ "_id" : ObjectId("55328f35f238ef5f0de2ad3e"), "title" : "Paradise", "artist" : "Coldplay", "album" : "Mylo Xyloto", "year" : 2011, "bpm" : 139.7 }
{ "_id" : ObjectId("55328f80f238ef5f0de2ad3f"), "title" : "The Scientist", "artist" : "Coldplay", "album" : "A Rush of Blood to the Head", "year" : 2002 }
{ "_id" : ObjectId("55328f87f238ef5f0de2ad40"), "title" : "Clocks", "artist" : "Coldplay", "album" : "A Rush of Blood to the Head", "year" : 2002, "bpm" : 131 }
{ "_id" : ObjectId("55328f9cf238ef5f0de2ad41"), "title" : "Fix You", "artist" : "Coldplay", "album" : "X&Y", "year" : 2005, "bpm" : 111.6 }
{ "_id" : ObjectId("55328facf238ef5f0de2ad42"), "title" : "Speed of Sound", "artist" : "Coldplay", "album" : "X&Y", "year" : 2005, "bpm" : 123.1 }
{ "_id" : ObjectId("55329036f238ef5f0de2ad43"), "title" : "With or Without You", "artist" : "U2", "album" : "The Joshua Tree", "year" : 1987 }
{ "_id" : ObjectId("55329067f238ef5f0de2ad44"), "title" : "Beautiful Day", "artist" : "U2", "album" : "Zooropa", "year" : 1993 }
{ "_id" : ObjectId("553290a7f238ef5f0de2ad45"), "title" : "Elevation", "artist" : "U2", "album" : "All That You Can't Leave Behind", "year" : 2000 }
>
```

2. Comptez le nombre de documents existants dans la collection songs.

> db.songs.count()
19

3. Affichez exclusivement les titres des chansons du Coldplay de l'album X&Y.

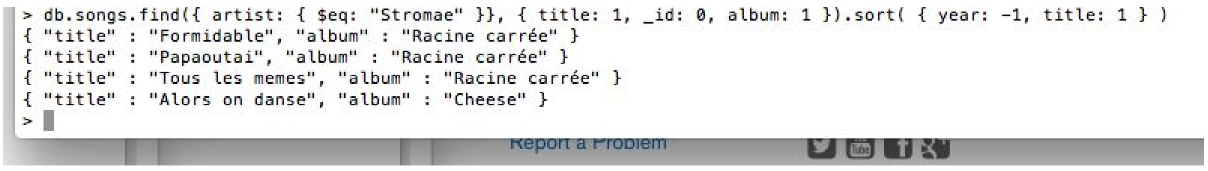
> db.songs.find({ $and: [{ artist: { $eq: "Coldplay" } }, { album: { $eq: "X&Y" } }] }, { title: 1, _id: 0 })



```
> db.songs.find({ $and: [{ artist: { $eq: "Coldplay" } }, { album: { $eq: "X&Y" } }] }, { title: 1, _id: 0 })
{ "title" : "Fix You" }
{ "title" : "Speed of Sound" }
>
```

4. Affichez le titre et album des chansons de Stromae, ordonnés par année de la plus récente à la plus ancienne, et triés par ordre alphabétique par titre.

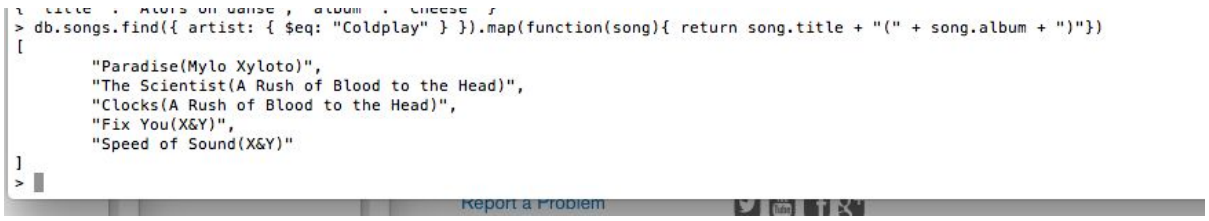
> db.songs.find({ artist: { $eq: "Stromae" }}, { title: 1, _id: 0, album: 1 }).sort({ year: -1, title: 1 })



```
> db.songs.find({ artist: { $eq: "Stromae" }}, { title: 1, _id: 0, album: 1 }).sort( { year: -1, title: 1 } )
{ "title" : "Formidable", "album" : "Racine carrée" }
{ "title" : "Papaoutai", "album" : "Racine carrée" }
{ "title" : "Tous les memes", "album" : "Racine carrée" }
{ "title" : "Alors on danse", "album" : "Cheese" }
>
```

5. Affichez les chansons du group Coldplay dans un tableau, où les éléments sont des strings ayant comme format TITRE (ALBUM).

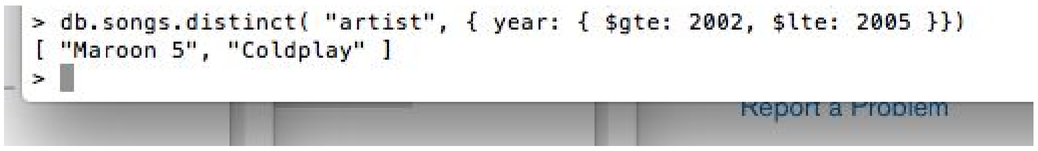
> db.songs.find({ artist: { $eq: "Coldplay" } }).map(function(song){ return song.title + "[" + song.album + "]" })



```
{ "title" : "Alors on danse", "album" : "Cheese" }
> db.songs.find({ artist: { $eq: "Coldplay" } }).map(function(song){ return song.title + "(" + song.album + ")"})
[
        "Paradise(Mylo Xyloto)",
        "The Scientist(A Rush of Blood to the Head)",
        "Clocks(A Rush of Blood to the Head)",
        "Fix You(X&Y)",
        "Speed of Sound(X&Y)"
]
>
```

6. Affichez, une seule fois, le noms des artistes ayant produit des chansons entre 2002 et 2005.

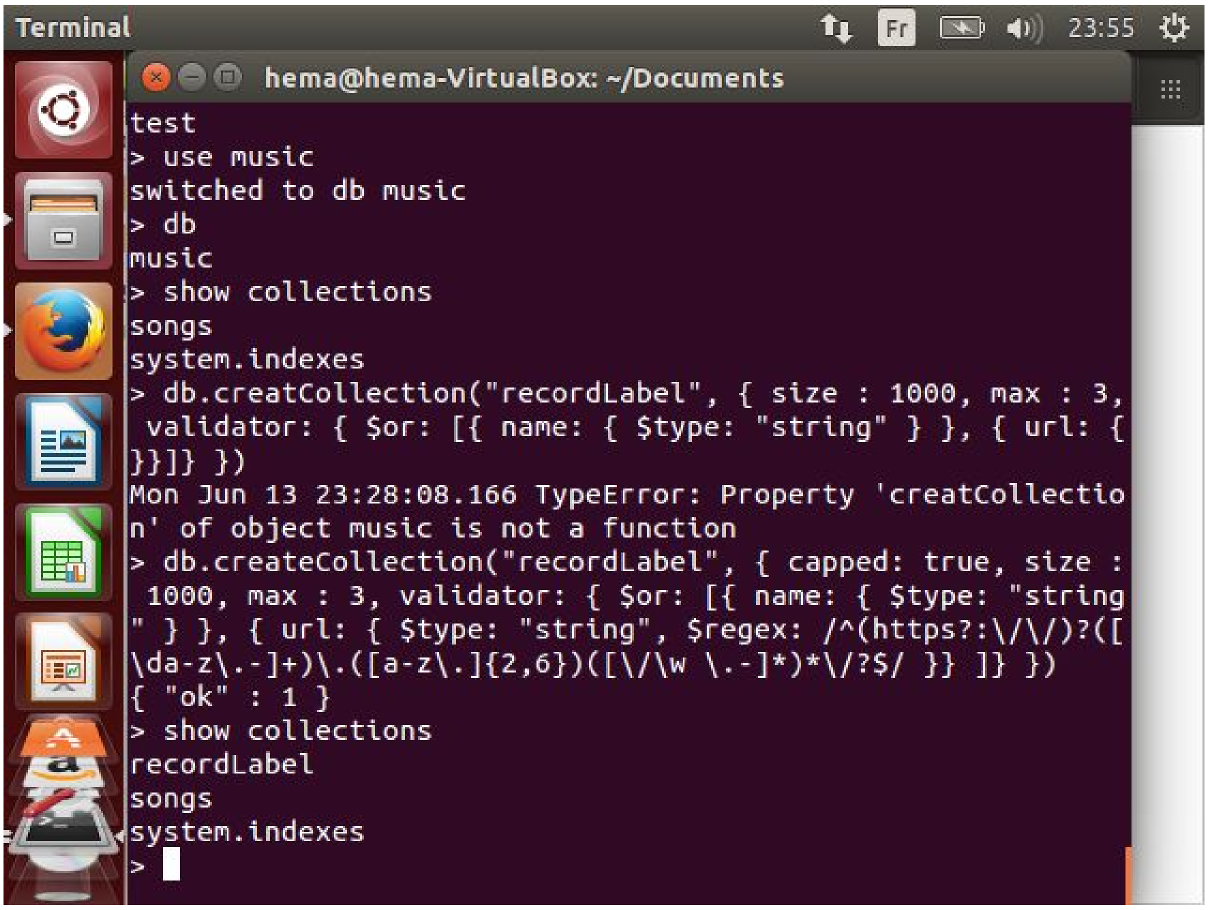
> db.songs.distinct(“artist”, { year: { $get: 2002, $lte: 2005 } })



```
> db.songs.distinct( "artist", { year: { $gte: 2002, $lte: 2005 }})
[ "Maroon 5", "Coldplay" ]
>
```

7. Créez une collection recordLabel, qui puisse stocker maximum 3 documents ou 1 KB et dont la structure doit être :
nom: string
url: string
La validation doit être stricte. Cherchez les regex nécessaires pour les attributes.

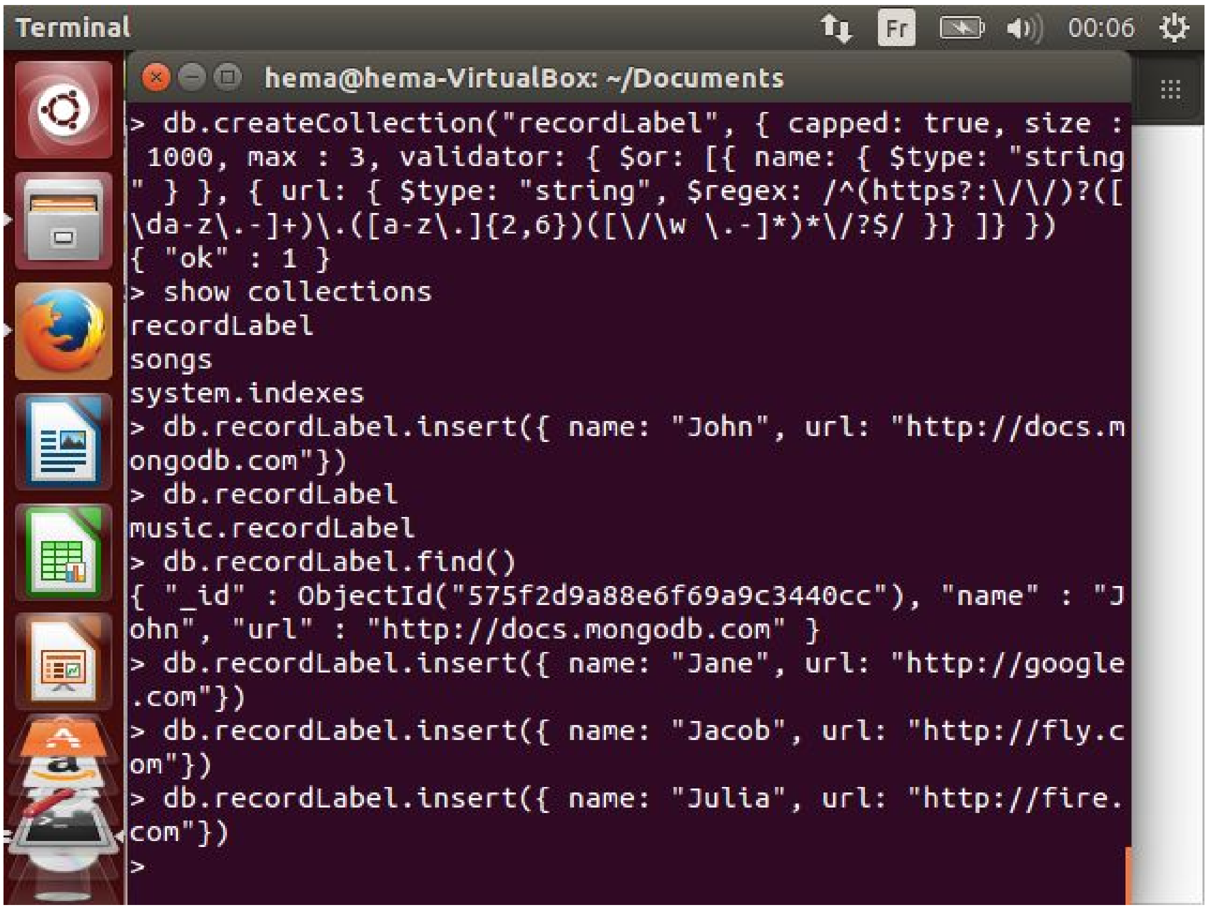
```
> db.createCollection(“recodLabel”, { capped: true, size:1000, max: 3, validator: { $or: [
        {name: { $type: "string" } },
        {url: { $type: "string", $regex: /^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/}}
  ]} })
```



```
test
> use music
switched to db music
> db
music
> show collections
songs
system.indexes
> db.creatCollection("recordLabel", { size : 1000, max : 3,
 validator: { $or: [{ name: { $type: "string" } }, { url: {
}}]} })
Mon Jun 13 23:28:08.166 TypeError: Property 'creatCollectio
n' of object music is not a function
> db.createCollection("recordLabel", { capped: true, size :
 1000, max : 3, validator: { $or: [{ name: { $type: "string
" } }, { url: { $type: "string", $regex: /^(https?:\/\/)?([
\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/ }} ]} })
{ "ok" : 1 }
> show collections
recordLabel
songs
system.indexes
> 
```

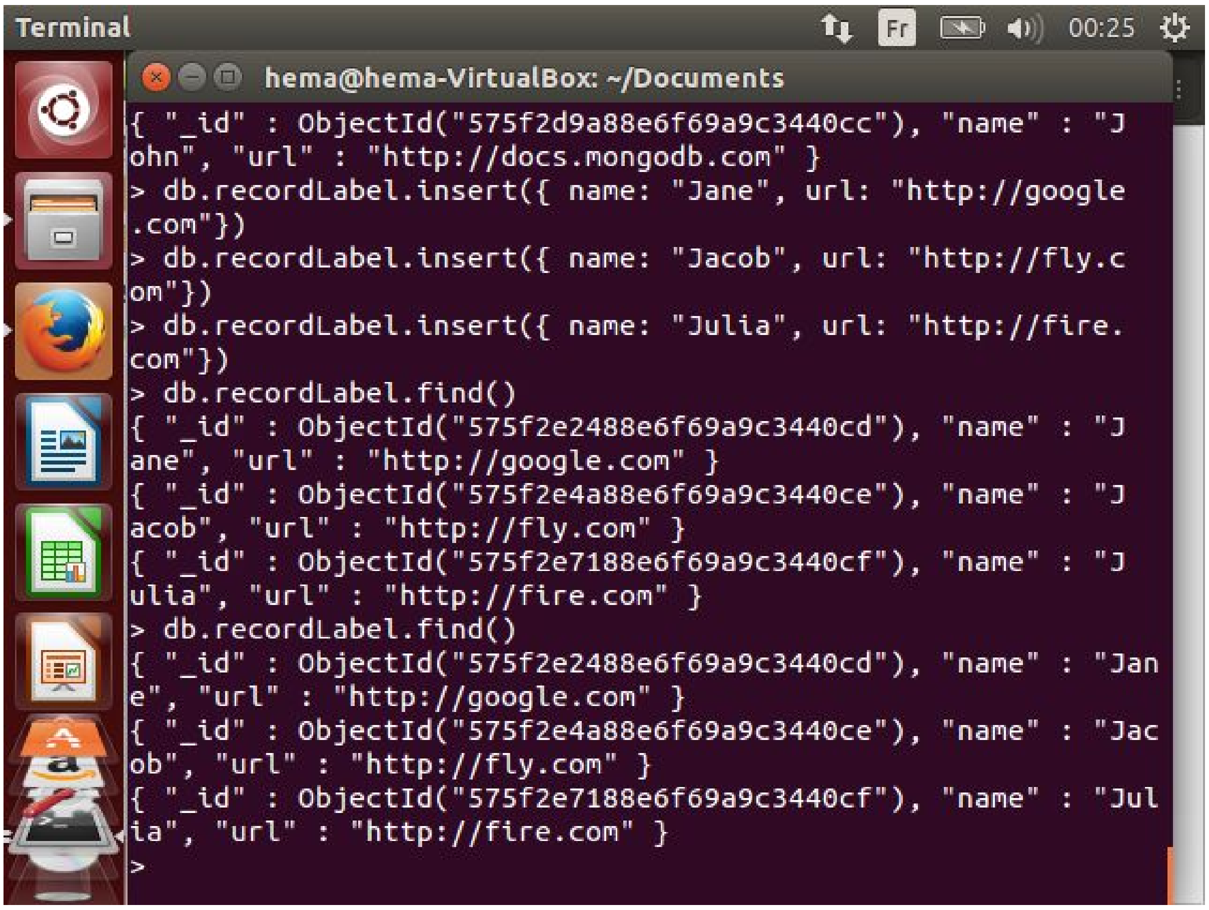
8. Insérez les 3 registres dans la collection. Qu’estce qui se passe lorsque vous essayez insérer un 4ème ?

```
> db.recordLabel.insert({ name: “Jane”, url: “http://google.com” })
```



```
> db.createCollection("recordLabel", { capped: true, size :
 1000, max : 3, validator: { $or: [{ name: { $type: "string
" } }, { url: { $type: "string", $regex: /^(https?:\/\/)?([
\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/ }} ]} })
{ "ok" : 1 }
> show collections
recordLabel
songs
system.indexes
> db.recordLabel.insert({ name: "John", url: "http://docs.m
ongodb.com"})
> db.recordLabel
music.recordLabel
> db.recordLabel.find()
{ "_id" : ObjectId("575f2d9a88e6f69a9c3440cc"), "name" : "J
ohn", "url" : "http://docs.mongodb.com" }
> db.recordLabel.insert({ name: "Jane", url: "http://google
.com"})
> db.recordLabel.insert({ name: "Jacob", url: "http://fly.c
om"})
> db.recordLabel.insert({ name: "Julia", url: "http://fire.
com"})
>
```

Lorsque l’on tente de rajouter un 4e registre, rien ne se passe aucune erreur n’est relevée. Le registre est ajouté mais la taille de la collection reste la même en supprimant le 1er registre.



```
{ "_id" : ObjectId("575f2d9a88e6f69a9c3440cc"), "name" : "J
ohn", "url" : "http://docs.mongodb.com" }
> db.recordLabel.insert({ name: "Jane", url: "http://google
.com"})
> db.recordLabel.insert({ name: "Jacob", url: "http://fly.c
om"})
> db.recordLabel.insert({ name: "Julia", url: "http://fire.
com"})
> db.recordLabel.find()
{ "_id" : ObjectId("575f2e2488e6f69a9c3440cd"), "name" : "J
ane", "url" : "http://google.com" }
{ "_id" : ObjectId("575f2e4a88e6f69a9c3440ce"), "name" : "J
acob", "url" : "http://fly.com" }
{ "_id" : ObjectId("575f2e7188e6f69a9c3440cf"), "name" : "J
ulia", "url" : "http://fire.com" }
> db.recordLabel.find()
{ "_id" : ObjectId("575f2e2488e6f69a9c3440cd"), "name" : "Jan
e", "url" : "http://google.com" }
{ "_id" : ObjectId("575f2e4a88e6f69a9c3440ce"), "name" : "Jac
ob", "url" : "http://fly.com" }
{ "_id" : ObjectId("575f2e7188e6f69a9c3440cf"), "name" : "Jul
ia", "url" : "http://fire.com" }
>
```

9. Modifiez le validator sur la collection afin d'ajouter le pays en utilisant le code (I SO 31661 alpha2)

```
> db.runCommand( { "collMod": "recordLabel" , "validator": { $and: [
   { country: { $type: "string", $regex:
/(AF|AX|AL|DZ|AS|AD|AO|AI|AQ|AG|AR|AM|AW|AU|AT|AZ|BS|BH|BD|BB|BY|BE|BZ|BJ|BM|BT|BO|BA|BW|BV|BR|IO|BN|BG|BF|BI|KH|CM|CA|CV|KY|CF|TD|CL|CN|CX|CC|CO|KM|CG|CD|CK|CR|CI|HR|CU|CY|CZ|DK|DJ|DM|DO|EC|EG|SV|GQ|ER|EE|ET|FK|FO|FJ|FI|FR|GF|PF|TF|GA|GM|GE|DE|GH|GI|GR|GL|GD|GP|GU|GT|GG|GN|GW|GY|HT|HM|VA|HN|HK|HU|IS|IN|ID|IR|IQ|IE|IM|IL|IT|JM|JP|JE|JO|KZ|KE|KI|KR|KW|KG|LA|LV|LB|LS|LR|LY|LI|LT|LU|MO|MK|MG|MW|MY|MV|ML|MT|MH|MQ|MR|MU|YT|MX|FM|MD|MC|MN|ME|MS|MA|MZ|MM|NA|NR|NP|NL|AN|NC|NZ|NI|NE|NG|NU|NF|MP|NO|OM|PK|PW|PS|PA|PG|PY|PE|PH|PN|PL|PT|PR|QA|RE|RO|RU|RW|BL|SH|KN|LC|MF|PM|VC|WS|SM|ST|SA|SN|RS|SC|SL|SG|SK|SI|SB|SO|ZA|GS|ES|LK|SD|SR|SJ|SZ|SE|CH|SY|TW|TJ|TZ|TH|TL|TG|TK|TO|TT|TN|TR|TM|TC|TV|UG|UA|AE|GB|US|UM|UY|UZ|VU|VE|VN|VG|VI|WF|EH|YE|ZM|ZW)/ } }
  ]
} } );
```

10. Pour allez plus loin:
a. Qu'estce que le TTL ?
TTL (Time To Live) attribue une durée de vie à un document, il sera ainsi supprimé après le TTL écoulé.

b. Quelles sont les modifications à faire sur une collection pour rajouter du TTL?
Pour rajouter du TTL sur une collection, il faut utiliser la méthode "db.collection.createIndex()" avec l'option "expireAfterSeconds" sur un champs dont la valeur est soit une date soit un tableau de dates.

Exemple : db.eventlog.createIndex({ "lastModifiedDate": 1 }, { expireAfterSeconds: 3600 })

c. Si vous devez faire cette manipulation sur la collection recordLabel, il faudrait faire quoi exactement ?
Si la collection est en capped il faudrait changer cette valeur en false, car on ne peut pas appliquer des TTL sur des collections capped. Mongo ne peut pas supprimer de documents d'une collection capped.

d. Créez une nouvelle collection recordLabel2, avec le même validator, mais avec une TTL sur les documents de 10 secondes.

```
db.createCollection("recordLabel2", { size:1000, max: 3, validator: { $and: [
    { country: { $type: "string", $regex:
/(AF|AX|AL|DZ|AS|AD|AO|AI|AQ|AG|AR|AM|AW|AU|AT|AZ|BS|BH|BD|BB|BY|BE|BZ|BJ|BM|
BT|BO|BA|BW|BV|BR|IO|BN|BG|BF|BI|KH|CM|CA|CV|KY|CF|TD|CL|CN|CX|CC|CO|KM|CG|
CD|CK|CR|CI|HR|CU|CY|CZ|DK|DJ|DM|DO|EC|EG|SV|GQ|ER|EE|ET|FK|FO|FJ|FI|FR|GF|P
F|TF|GA|GM|GE|DE|GH|GI|GR|GL|GD|GP|GU|GT|GG|GN|GW|GY|HT|HM|VA|HN|HK|HU|IS
|IN|ID|IR|IQ|IE|IM|IL|IT|JM|JP|JE|JO|KZ|KE|KI|KR|KW|KG|LA|LV|LB|LS|LR|LY|LI|LT|LU|MO|
MK|MG|MW|MY|MV|ML|MT|MH|MQ|MR|MU|YT|MX|FM|MD|MC|MN|ME|MS|MA|MZ|MM|NA|
NR|NP|NL|AN|NC|NZ|NI|NE|NG|NU|NF|MP|NO|OM|PK|PW|PS|PA|PG|PY|PE|PH|PN|PL|PT
|PR|QA|RE|RO|RU|RW|BL|SH|KN|LC|MF|PM|VC|WS|SM|ST|SA|SN|RS|SC|SL|SG|SK|SI|S
B|SO|ZA|GS|ES|LK|SD|SR|SJ|SZ|SE|CH|SY|TW|TJ|TZ|TH|TL|TG|TK|TO|TT|TN|TR|TM|TC|
TV|UG|UA|AE|GB|US|UM|UY|UZ|VU|VE|VN|VG|VI|WF|EH|YE|ZM|ZW)/ } },
    { name: { $type: "string" } },
    { url: { $type: "string", $regex: /^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/ } }
  ]}
});
db.recordLabel2.createIndex( { expireAfterSeconds: 10 } );
```

QUATRIÈME PARTIE (Schema Design)

1. Considérez les concepts de CV et Personne. Comment peut on représenter ces deux concepts avec un Embedded Design ? Comment le faire avec le Separated Collection Design ? Donnez des exemples d’utilisation de chaque possible schema.

Embedded Design

Option 1 - Collection Personne

```
{
        “firstname” : “Nathan”,
        “lastname” : “Dupont”,
        “age” :        “22”,
        “adress” :    “1 Allée Jean Jaurès”,
        “city” :       “Gennevilliers”,
        “country” :  “France”,
        “cv” : {
                “title” : “Front-end developper”,
                “studies” : [
                        “Licence Pro Gennevilliers”?
                        “DUT MMI de Marne la Vallée”,
                        “DUT Info de Marne la Vallée”
                ],
                “experiences” : [
                        “Developper Front chez MyAgency”
                ],
                “skills” : [
                        “HTML”,
                        “CSS”,
                        “JS”,
                ]

}
```

Option 2 - Collection CV

```
{
        “title” : “Front-end developper”,
        “studies” : [
                “Licence Pro Gennevilliers”?
                “DUT MMI de Marne la Vallée”,
                “DUT Info de Marne la Vallée”
        ],
        “experiences” : [
                “Developper Front chez MyAgency”
        ],
        “skills” : [
                “HTML”,
                “CSS”,
                “JS”,
        ],
        “personne” : {
                “firstname” : “Nathan”,
                “lastname” : “Dupont”,
                “age” :          “22”,
                “adress” :     “1 Allée Jean Jaurès”,
                “city” :          “Gennevilliers”,
                “country” :   “France”
        }
}
```

Separated Collection Design

Personnes

```
{
        “id” :    1,
        “firstname” : “Nathan”,
        “lastname” : “Dupont”,
        “age” :          “22”,
        “adress” :     “1 Allée Jean Jaurès”,
        “city” :          “Gennevilliers”,
        “country” :   “France”,
        “cv_id” :          2
},
{
        “id” :    2,
        “firstname” : “Julie”,
        “lastname” : “Martin”,
        “age” :          “27”,
        “adress” :     “1 Rue Jaurès”,
        “city” :          “Lognes”,
        “country” :    “France”,
        “cv_id” :          1
}
```

CV

```
{
	“id” :	1,
	“title” : “Front-end developper”,
	“studies” : [
		“Licence Pro Gennevilliers”?
		“DUT MMI de Marne la Vallée”,
		“DUT Info de Marne la Vallée”
	],
	“experiences” : [
		“Developper Front chez MyAgency”
	],
	“skills” : [
		“HTML”,
		“CSS”,
		“JS”,
	],

},
{
	“id” :	2,
	“title” : “Backend developper”,
	“studies” : [
		“Licence Pro Gennevilliers”?
		“DUT Info de Marne la Vallée”
	],
	“experiences” : [
	],
	“skills” : [
		“PHP”,
	],

}
```

PARTIE FINALE

1. Exportez la collections des chansons.

```
$ mongodump -d music -c songs --out [myRepository]
```

2. Exportez la collection des utilisateurs de la base des données n'ayant aucune chanson dans la liste des favorites.

```
$ mongodump -d music -c users -q '{"favoriteSongs": {"$size": 0}}' --out [myRepository]
```

3. Créez une nouvelle base de données appelé 'nofavorites' contenant les utilisateurs exportés.

```
$ mongorestore --db no-favorites noFav/music
```

4. Recherche: Quelles autres commandes permettent sur mongodb de faire export et import ? Quelles sont les différences avec mongodump et mongorestore ?

Autre que mongodump et mongorestore, il existe la commande mongoexport pour l'export et mongoimport pour l'import.
Mongodump permet d'exporter tout d'une base de données avec typage contrairement à mongoexport qui exporte que les collections en perdant le typage des données.

Mongorestore permet d'importer tous les documents/données avec typage contrairement à mongoimport qui permet d'importer que les collections de certains types de fichiers (CSV, JSON, TSV).