JSON Stores Lab (MongoDB)

CHEN Hang

System: MaxOS

MongoDB: setting up

- **Install MongoDB**

```
brew install mongodb
```

- **Create a directory for the server and launch it. Write down the command for launching it; on which port it runs**

```
sudo mkdir -p /data/db
sudo /usr/local/Cellar/mongodb/4.0.2/bin/mongod
```

It runs on the port : **27017**

```
2018-10-06T11:13:00.758+0200 I NETWORK  [initandlisten] waiting for
connections
on port 27017
2018-10-06T11:13:09.205+0200 I NETWORK  [listener] connection accepted
from
127.0.0.1:59125 #1 (1 connection now open)
```

- **Import the document moviepeople-10.json into the server**

```
mongoimport --db tp --collection moviepeople10 --file
/Users/hchen/Desktop/DK/Architectures/lab2/labJSON/moviepeople-10.json
```

- **Launch a client (mongo shell), retrieve all the documents by asking a query in the client**

Launch a client from shell

```
/usr/local/Cellar/mongodb/4.0.2/bin/mongo
```

Retrieve all the documents

```
> show dbs
> use tp
> show collections
> db.moviepeople10.find()
```

MongoDB: import, query

- **Import the documents moviepeople-3000.json and cities.json into the server**

  Import moviepeople-3000.json

  ```
  mongoimport --db tp --collection moviepeople3000 --file
  /Users/hchen/Desktop/DK/Architectures/lab2/labJSON/moviepeople-
  3000.json
  ```

  Answer:

  Maybe there is some errors in the file, so we can only import 2736 documents.

  ```
  2018-10-10T23:07:05.080+0200    connected to: localhost
  2018-10-10T23:07:05.392+0200    Failed: error processing document
  #2742:
  invalid character '(' after array element
  2018-10-10T23:07:05.392+0200    imported 2736 documents
  ```

  Import cities.json

  ```
  mongoimport --db tp --collection cities --file
  /Users/hchen/Desktop/DK/Architectures/lab2/labJSON/cities.json
  ```

  Answer:

  ```
  2018-10-10T23:08:01.671+0200    connected to: localhost
  2018-10-10T23:08:02.675+0200    imported 99838 documents
  ```

- **In the mongo shell client, write queries for finding:**

1. **The person named Anabela Teixeira**

   "pretty()" is for making the format of output more beautiful.

```
> db.moviepeople3000.find({"person-name" : "Teixeira,
Anabela”}).pretty()
```

   Answer:


```
{
    "_id" : ObjectId("5bbe69f92c72e90c69e4b8f5"),
    "person-name" : "Teixeira, Anabela",
    "info" : {
        "trivia" : [
            "Her favorite actor is 'Marcello Mastroianni' (qv) and
her favorite
             actress is 'Juliette Binoche' (qv).",
            "Partner of musician Frederico Pereira.",
            "Has two brothers, one biological and one adopted.",
            "Vice-President of the Portuguese Cinema Academy.",
            "She was first noticed by the public with the Tv series
A Viúva do
             Enforcado (1992).",
            "Made her theater debut in 1992 in the play Os
Processos of
             Dostolevsky.",
            "Did some workshops of dance: dance in movement with
Peter Diez and
             dance classes with Madalena Vitorino.",
            "Besides Portuguese, she speaks English and French."
        ],
        "birthnotes" : [
            "Lisbon, Portugal"
        ],
        "birthdate" : [
            "18 May 1973"
        ],
        "birthname" : [
            "Teixeira, Anabela Cristina Alves"
        ],
        "interviews" : [
            "TV Guia (Portugal), 1997, Iss. 960, pg. 24-25, by:
Pedro Teixeira",
            "A Capital (Portugal), 30 April 1998, pg. 55, by:
Helena Mata"
        ]
    }
}
```

2. **The birthplace of Steven Spielberg**

```
> db.moviepeople3000.find({"person-name":"Spielberg, Steven"},
{"_id":0,"info.birthnotes":1}).pretty()
```

Answer:

```
{ "info" : { "birthnotes" : [ "Cincinnati, Ohio, USA" ] } }
```

3. **The number of people born in Lisbon**

```
> db.moviepeople3000.find({"info.birthnotes":/Lisbon/}).count()
```

Answer:

```
142
```

4. **The people taller than 170 cm**

```
> db.moviepeople3000.find({$or:[{"info.height":/[1][7][1-9]/},
{"info.height":/[1][8-9][0-9]/},{"info.height":/[2][0-9][0-9]/}]},
{"_id":0, "person-name":1, "info.height":1})
```

Answer:

```
{ "person-name" : "O'Brien, Conan", "info" : { "height" : [ "193
cm" ] } }
{ "person-name" : "Kimmel, Jimmy", "info" : { "height" : [ "182.88
cm" ] } }
{ "person-name" : "MacFarlane, Seth", "info" : { "height" : [
"180.6 cm" ] }}
{ "person-name" : "Lovhaug, Lewis", "info" : { "height" : [ "183
cm" ] } }
{ "person-name" : "Filipe, Guilherme", "info" : { "height" : [ "175
cm" ] } }
{ "person-name" : "Catarré, João", "info" : { "height" : [ "184 cm"
] } }
{ "person-name" : "Samora, Rogério", "info" : { "height" : [ "182
cm" ] } }
```

```
{ "person-name" : "Rapazote, Pêpê", "info" : { "height" : [ "181
cm" ] } }
{ "person-name" : "Vaona, Federico", "info" : { "height" : [ "185
cm" ] } }
{ "person-name" : "Gálvez, Christian", "info" : { "height" : [ "184
cm" ] } }
{ "person-name" : "Athayde, Jessica", "info" : { "height" : [ "173
cm" ] } }
{ "person-name" : "Pereira, Ricardo", "info" : { "height" : [ "185
cm" ] } }
{ "person-name" : "Serrano, Fernanda", "info" : { "height" : [ "175
cm" ] } }
{ "person-name" : "Thornton, Kirk", "info" : { "height" : [ "190
cm" ] } }
{ "person-name" : "Schmidt, Harald", "info" : { "height" : [ "191
cm" ] } }
{ "person-name" : "Vincent, Sam", "info" : { "height" : [ "178 cm"
] } }
{ "person-name" : "Fidalgo, José", "info" : { "height" : [ "180 cm"
] } }
{ "person-name" : "Brooks, Darin", "info" : { "height" : [ "178 cm"
] } }
{ "person-name" : "Jerónimo, Albano", "info" : { "height" : [ "191
cm" ] } }
{ "person-name" : "Wood, Mark", "info" : { "height" : [ "191 cm" ]
} }
Type "it" for more
```

5. **The names of people whose information contains "Opera"**

   Method1: Using the function of JavaScript

```
> db.moviepeople3000.find(function(){
... var text = JSON.stringify(this['info']);
... var obj = JSON.parse(text);
... for(i in obj){
...     var content = obj[i];
...     var str_content = JSON.stringify(content);
...     expr = "Opera";
...     if(str_content.includes(expr) == true){
...         return this.info;
...     }
... }
... },{"_id":0, "person-name":1})
```

   Method2: Using the index of full text

```
> db.moviepeople3000.ensureIndex({"$**":"text"})
> db.moviepeople3000.find({$text: {$search:"Opera"}},
{"_id":0,"person-name":1})
```

Answer:

```
{ "person-name" : "Bell, William J." }
{ "person-name" : "Griffin, Merv" }
{ "person-name" : "Nixon, Agnes" }
{ "person-name" : "Flynn, Christopher" }
{ "person-name" : "Lang, Katherine Kelly" }
{ "person-name" : "Moss, Ronn" }
{ "person-name" : "Rolfe, James" }
{ "person-name" : "Davidson, Doug" }
{ "person-name" : "Vaona, Federico" }
{ "person-name" : "Braeden, Eric" }
{ "person-name" : "Wyndham, Victoria" }
{ "person-name" : "Hall, Deidre" }
{ "person-name" : "Ashford, Matthew" }
{ "person-name" : "Mascolo, Joseph" }
{ "person-name" : "Sweeney, Alison" }
{ "person-name" : "Ripa, Kelly" }
{ "person-name" : "Slezak, Erika" }
{ "person-name" : "Linder, Kate" }
{ "person-name" : "Case, Sharon" }
{ "person-name" : "Scott, Melody Thomas" }
Type "it" for more
```

6. **For each movie person whose birthplace is known, find the latitude, longitude and population of that city (if that information exists in the city document)**

   1) Take the name and full information of address of "moviepeople3000" to new collection "people1";

   2) Take the name, population and the information of location of "cities" to new collection "cityinfo";

   3) Pick up the name of city from "people1" and output to new collection "people2";

   4) Join "people2" and "cityinfo" with the key of city name and show the infomation that we need in the console.

```
> db.moviepeople3000.aggregate([
{"$project":
{"_id":0,
"name":"$person-name",
"birthnotes":{"$arrayElemAt":["$info.birthnotes",0]}}},
{"$out": "people1"}
```

```
])

> db.cities.aggregate([
{"$project":
{"_id":0,"name":1,
"population":1,
"location":1}},
{"$out": "cityinfo"}
])

> db.people1.aggregate([
{$project:
{"_id":0,
"name": "$name",
"city":{$arrayElemAt:[{$split:["$birthnotes",","]},0]}}},
{"$out":"people2"}
])

> db.people2.aggregate([
{$lookup:
{from:"cityinfo",
localField:"city",
foreignField:"name",
as: "peoplecity"}},
{$project:
{"_id":0 ,
"personName": "$name",
"cityName":"$city",
"population":"$peoplecity.population",
"location":"$peoplecity.location"}}
])
```

Output:

```
{ "personName" : "Warren, Tony", "cityName" : "Eccles",
"population" : [ 37275 ], "location" : [ { "latitude" : 53.48333,
"longitude" : -2.33333 } ] }
{ "personName" : "Barker, Bob", "cityName" : "Darrington",
"population" : [ 1446 ], "location" : [ { "latitude" : 48.25539,
"longitude" : -121.60151 } ] }
{ "personName" : "Stewart, Jon", "cityName" : "New York City",
"population" : [ 8008278 ], "location" : [ { "latitude" : 40.71427,
"longitude" : -74.00597 } ] }
{ "personName" : "Friedman, Harry", "cityName" : "Omaha",
"population" : [ 390007 ], "location" : [ { "latitude" : 41.25861,
"longitude" : -95.93779 } ] }
{ "personName" : "Trebek, Alex", "cityName" : "Sudbury",
"population" : [ 17343 ], "location" : [ { "latitude" : 42.38343,
"longitude" : -71.41617 } ] }
...
...
...
```

MongoDB replication

- **Create working directories for 3 MongoDB servers**

```
mkdir ./mongo1 ./mongo2 ./mongo3
```

- **Create a replication set for a collection named small-movie**

```
mkdir ./small-movie
```

- **Launch 3 MongoDB servers (in different shells). The server program is mongod. Leave those shells alone.**

```
sudo mongod --replSet small-movie --dbpath ./mongo1 --port 27011
sudo mongod --replSet small-movie --dbpath ./mongo2 --port 27012
sudo mongod --replSet small-movie --dbpath ./mongo3 --port 27013
```

- **Connect a client (mongo) to one server. Through the client, initialize the replication: add the other replica server, and the arbiter.**

  Connect a client (mongo) to one server.

```
mongo localhost:27011
```

  Output of server **mongo1(27011)**:

```
2018-10-11T17:24:59.968+0200 I NETWORK  [listener] connection accepted
from 127.0.0.1:64100 #1 (1 connection now open)
2018-10-11T17:24:59.969+0200 I NETWORK  [conn1] received client
metadata from 127.0.0.1:64100 conn1: { application: { name: "MongoDB
Shell" }, driver: { name: "MongoDB Internal Client", version: "4.0.2"
}, os: { type: "Darwin", name: "Mac OS X", architecture: "x86_64",
version: "17.7.0" } }
```

  Initialize the replication, we make the server of 27011 master, make 27012 the replica server, and make 27013 the arbiter.

```
> rs.initiate({_id: 'small-movie',
members: [ {_id: 1, host: 'localhost:27011',priority:2},
           {_id: 2, host: 'localhost:27012',priority:1},
           {_id: 3, host: 'localhost:27013',arbiterOnly:true} ] })
```

  Output of server **mongo1(27011)**:

```
2018-10-11T17:33:36.630+0200 I NETWORK  [conn11] received client
metadata from 127.0.0.1:64184 conn11: { driver: { name:
"NetworkInterfaceTL", version: "4.0.2" }, os: { type: "Darwin", name:
"Mac OS X", architecture: "x86_64", version: "17.7.0" } }
2018-10-11T17:33:36.668+0200 I REPL     [replexec-0] Member
localhost:27012 is now in state STARTUP2
2018-10-11T17:33:36.669+0200 I REPL     [replexec-0] Member
localhost:27013 is now in state ARBITER
2018-10-11T17:33:37.172+0200 I REPL     [replexec-0] Member
localhost:27012 is now in state SECONDARY
```

  Output of server **mongo2(27012)**:

```
2018-10-11T17:33:36.443+0200 I NETWORK  [conn7] received client
metadata from 127.0.0.1:64181 conn7: { driver: { name:
"NetworkInterfaceTL", version: "4.0.2" }, os: { type: "Darwin", name:
"Mac OS X", architecture: "x86_64", version: "17.7.0" } }
2018-10-11T17:33:36.443+0200 I ASIO     [Replication] Connecting to
localhost:27013
2018-10-11T17:33:36.445+0200 I REPL     [replexec-0] New replica set
config in use: { _id: "small-movie", version: 1, protocolVersion: 1,
writeConcernMajorityJournalDefault: true, members: [ { _id: 1, host:
"localhost:27011", arbiterOnly: false, buildIndexes: true, hidden:
false, priority: 2.0, tags: {}, slaveDelay: 0, votes: 1 }, { _id: 2,
host: "localhost:27012", arbiterOnly: false, buildIndexes: true,
hidden: false, priority: 1.0, tags: {}, slaveDelay: 0, votes: 1 }, {
_id: 3, host: "localhost:27013", arbiterOnly: true, buildIndexes: true,
hidden: false, priority: 0.0, tags: {}, slaveDelay: 0, votes: 1 } ],
settings: { chainingAllowed: true, heartbeatIntervalMillis: 2000,
heartbeatTimeoutSecs: 10, electionTimeoutMillis: 10000,
catchUpTimeoutMillis: -1, catchUpTakeoverDelayMillis: 30000,
getLastErrorModes: {}, getLastErrorDefaults: { w: 1, wtimeout: 0 },
replicaSetId: ObjectId('5bbf6d4ee1f5bfea0edc0c04') } }
2018-10-11T17:33:36.445+0200 I REPL     [replexec-0] This node is
localhost:27012 in the config
2018-10-11T17:33:36.445+0200 I REPL     [replexec-0] transition to
STARTUP2 from STARTUP
2018-10-11T17:33:36.445+0200 I REPL     [replexec-0] Starting
replication storage threads
2018-10-11T17:33:36.446+0200 I REPL     [replexec-1] Member
localhost:27011 is now in state SECONDARY
2018-10-11T17:33:36.446+0200 I REPL     [replexec-2] Member
localhost:27013 is now in state ARBITER
```

Output of server **mongo3(27013)**:

```
2018-10-11T17:33:34.383+0200 I NETWORK  [conn2] received client
metadata from 127.0.0.1:64174 conn2: { driver: { name:
"NetworkInterfaceTL", version: "4.0.2" }, os: { type: "Darwin", name:
"Mac OS X", architecture: "x86_64", version: "17.7.0" } }
2018-10-11T17:33:34.384+0200 I ASIO     [Replication] Connecting to
localhost:27011
2018-10-11T17:33:36.396+0200 I NETWORK  [listener] connection accepted
from 127.0.0.1:64180 #6 (2 connections now open)
2018-10-11T17:33:36.396+0200 I NETWORK  [conn6] end connection
127.0.0.1:64180 (1 connection now open)
2018-10-11T17:33:36.397+0200 I STORAGE  [replexec-0] createCollection:
local.system.replset with generated UUID: 1def72ee-eb4c-4e44-886a-
ebd341f049e5
2018-10-11T17:33:36.441+0200 I REPL     [replexec-0] Stopping key
manager
2018-10-11T17:33:36.441+0200 I REPL     [replexec-0] New replica set
config in use: { _id: "small-movie", version: 1, protocolVersion: 1,
writeConcernMajorityJournalDefault: true, members: [ { _id: 1, host:
"localhost:27011", arbiterOnly: false, buildIndexes: true, hidden:
false, priority: 2.0, tags: {}, slaveDelay: 0, votes: 1 }, { _id: 2,
host: "localhost:27012", arbiterOnly: false, buildIndexes: true,
hidden: false, priority: 1.0, tags: {}, slaveDelay: 0, votes: 1 }, {
_id: 3, host: "localhost:27013", arbiterOnly: true, buildIndexes: true,
hidden: false, priority: 0.0, tags: {}, slaveDelay: 0, votes: 1 } ],
settings: { chainingAllowed: true, heartbeatIntervalMillis: 2000,
heartbeatTimeoutSecs: 10, electionTimeoutMillis: 10000,
catchUpTimeoutMillis: -1, catchUpTakeoverDelayMillis: 30000,
getLastErrorModes: {}, getLastErrorDefaults: { w: 1, wtimeout: 0 },
replicaSetId: ObjectId('5bbf6d4ee1f5bfea0edc0c04') } }
2018-10-11T17:33:36.441+0200 I REPL     [replexec-0] This node is
localhost:27013 in the config
2018-10-11T17:33:36.442+0200 I REPL     [replexec-0] transition to
ARBITER from STARTUP
2018-10-11T17:33:36.442+0200 I ASIO     [Replication] Connecting to
localhost:27012
2018-10-11T17:33:36.442+0200 I REPL     [replexec-1] Member
localhost:27011 is now in state SECONDARY
2018-10-11T17:33:36.443+0200 I REPL     [replexec-0] Member
localhost:27012 is now in state STARTUP
```

- **Identify the master from the outputs in the servers' shell and by requesting replica set information from the servers.**

  From output of **mongo2(27012)**:

  We can know that 27013 is an arbiter.

```
2018-10-11T17:46:55.440+0200 I REPL     [replexec-3] Member
localhost:27013 is now in state ARBITER
```

From output of **mongo3(27013)**:

We can know that 27011 is the master and 27012 is the replica server

```
2018-10-11T17:46:57.438+0200 I REPL     [replexec-0] Member
localhost:27012 is now in state SECONDARY
2018-10-11T17:47:05.451+0200 I REPL     [replexec-0] Member
localhost:27011 is now in state PRIMARY
```

By requesting:

```
> rs.status()
```

Result:

We get the same information than above.

```
{
    "set" : "small-movie",
    "date" : ISODate("2018-10-11T15:55:36.064Z"),
    "myState" : 1,
    "term" : NumberLong(1),
    "syncingTo" : "",
    "syncSourceHost" : "",
    "syncSourceId" : -1,
    "heartbeatIntervalMillis" : NumberLong(2000),
    "optimes" : {
        "lastCommittedOpTime" : {
            "ts" : Timestamp(1539273326, 1),
            "t" : NumberLong(1)
        },
        "readConcernMajorityOpTime" : {
            "ts" : Timestamp(1539273326, 1),
            "t" : NumberLong(1)
        },
        "appliedOpTime" : {
            "ts" : Timestamp(1539273326, 1),
            "t" : NumberLong(1)
        },
        "durableOpTime" : {
            "ts" : Timestamp(1539273326, 1),
            "t" : NumberLong(1)
        }
    },
```

```
    "lastStableCheckpointTimestamp" : Timestamp(1539273306, 1),
    "members" : [
        {
            "_id" : 1,
            "name" : "localhost:27011",
            "health" : 1,
            "state" : 1,
            "stateStr" : "PRIMARY",
            "uptime" : 553,
            "optime" : {
                "ts" : Timestamp(1539273326, 1),
                "t" : NumberLong(1)
            },
            "optimeDate" : ISODate("2018-10-11T15:55:26Z"),
            "syncingTo" : "",
            "syncSourceHost" : "",
            "syncSourceId" : -1,
            "infoMessage" : "",
            "electionTime" : Timestamp(1539272824, 1),
            "electionDate" : ISODate("2018-10-11T15:47:04Z"),
            "configVersion" : 1,
            "self" : true,
            "lastHeartbeatMessage" : ""
        },
        {
            "_id" : 2,
            "name" : "localhost:27012",
            "health" : 1,
            "state" : 2,
            "stateStr" : "SECONDARY",
            "uptime" : 522,
            "optime" : {
                "ts" : Timestamp(1539273326, 1),
                "t" : NumberLong(1)
            },
            "optimeDurable" : {
                "ts" : Timestamp(1539273326, 1),
                "t" : NumberLong(1)
            },
            "optimeDate" : ISODate("2018-10-11T15:55:26Z"),
            "optimeDurableDate" : ISODate("2018-10-11T15:55:26Z"),
            "lastHeartbeat" : ISODate("2018-10-11T15:55:35.808Z"),
            "lastHeartbeatRecv" : ISODate("2018-10-11T15:55:34.557Z"),
            "pingMs" : NumberLong(0),
            "lastHeartbeatMessage" : "",
            "syncingTo" : "localhost:27011",
            "syncSourceHost" : "localhost:27011",
            "syncSourceId" : 1,
            "infoMessage" : "",
```

```
            "configVersion" : 1
        },
        {
            "_id" : 3,
            "name" : "localhost:27013",
            "health" : 1,
            "state" : 7,
            "stateStr" : "ARBITER",
            "uptime" : 522,
            "lastHeartbeat" : ISODate("2018-10-11T15:55:35.808Z"),
            "lastHeartbeatRecv" : ISODate("2018-10-11T15:55:34.295Z"),
            "pingMs" : NumberLong(0),
            "lastHeartbeatMessage" : "",
            "syncingTo" : "",
            "syncSourceHost" : "",
            "syncSourceId" : -1,
            "infoMessage" : "",
            "configVersion" : 1
        }
    ],
    "ok" : 1,
    "operationTime" : Timestamp(1539273326, 1),
    "$clusterTime" : {
        "clusterTime" : Timestamp(1539273326, 1),
        "signature" : {
            "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
            "keyId" : NumberLong(0)
        }
    }
}
```

- **Import moviepeople-10.json through the master; note the output of the two other servers.**

  When the synchronization has finished, stop (ctrl-c) the master; note the output of the two other servers.

  Import moviepeople-10.json through the master

```
mongoimport --db tp --collection moviepeople10 --port 27011 --file
/Users/hchen/Desktop/DK/Architectures/lab2/labJSON/moviepeople-10.json
```

  Output of mongo1:

```
2018-10-11T18:12:02.439+0200 I NETWORK  [listener] connection accepted
from 127.0.0.1:64434 #14 (5 connections now open)
2018-10-11T18:12:02.445+0200 I STORAGE  [conn14] createCollection:
tp.moviepeople10 with generated UUID: 0d7fdaf3-8e8b-48a0-8622-
68205a84c96f
2018-10-11T18:12:02.522+0200 I NETWORK  [conn14] end connection
127.0.0.1:64434 (4 connections now open)
```

Output of mongo2:

```
2018-10-11T18:12:02.443+0200 I NETWORK  [listener] connection accepted
from 127.0.0.1:64436 #12 (3 connections now open) 2018-10-
11T18:12:02.481+0200 I STORAGE  [repl writer worker 6]
createCollection: tp.moviepeople10 with provided UUID: 0d7fdaf3-8e8b-
48a0-8622-68205a84c96f 2018-10-11T18:12:02.521+0200 I NETWORK  [conn12]
end connection 127.0.0.1:64436 (2 connections now open)
```

Output of mongo3:

```
2018-10-11T18:14:12.741+0200 I NETWORK  [LogicalSessionCacheRefresh]
Starting new replica set monitor for small-
movie/localhost:27011,localhost:27012
```

When the synchronization has finished, stop (ctrl-c) the master.

```
ctrl-c
```

The 2 other servers will be disconnected from the master server 27011 but they will try to connect it constantly.

Output of mongo2:

```
2018-10-11T18:31:32.466+0200 I ASIO     [Replication] Failed to connect
to localhost:27011 - HostUnreachable: Error connecting to
localhost:27011 (127.0.0.1:27011) :: caused by :: Connection refused
2018-10-11T18:31:32.466+0200 I ASIO     [Replication] Dropping all
pooled connections to localhost:27011 due to HostUnreachable: Error
connecting to localhost:27011 (127.0.0.1:27011) :: caused by ::
Connection refused
2018-10-11T18:31:32.466+0200 I REPL_HB  [replexec-14] Error in
heartbeat (requestId: 2345) to localhost:27011, response status:
HostUnreachable: Error connecting to localhost:27011 (127.0.0.1:27011)
:: caused by :: Connection refused 2018-10-11T18:31:32.466+0200 I ASIO
    [Replication] Connecting to localhost:27011
```

Output of mongo3:

```
2018-10-11T18:31:46.198+0200 I ASIO     [Replication] Failed to connect
to localhost:27011 - HostUnreachable: Error connecting to
localhost:27011 (127.0.0.1:27011) :: caused by :: Connection refused
2018-10-11T18:31:46.198+0200 I ASIO     [Replication] Dropping all
pooled connections to localhost:27011 due to HostUnreachable: Error
connecting to localhost:27011 (127.0.0.1:27011) :: caused by ::
Connection refused
2018-10-11T18:31:46.198+0200 I REPL_HB  [replexec-1] Error in heartbeat
(requestId: 1360) to localhost:27011, response status: HostUnreachable:
Error connecting to localhost:27011 (127.0.0.1:27011) :: caused by ::
Connection refused
2018-10-11T18:31:48.203+0200 I ASIO     [Replication] Connecting to
localhost:27011
```

MongoDB sharding

- **Start two shard servers; shard the cities by the country.**

  Start the shard servers:

  ```
  mkdir ./mongo4 ./mongo5
  ```

  ```
  mongod --shardsvr --dbpath ./mongo4 --port 27014
  mongod --shardsvr --dbpath ./mongo5 --port 27015
  ```

  Start the config server:

  ```
  mkdir ./mongoconfig
  mongod --configsvr --replSet conf --dbpath ./mongoconfig --port 27100
  ```

  Make a replica of config server:

  ```
  mongo localhost:27100
  ```

  ```
  > use admin
  > rs.initiate({"id":"conf", "configsvr":true, "members":[{"id":0,
  "host": "localhost:27100"}]})
  ```

  Output:

```
{
 "ok" : 1,
 "operationTime" : Timestamp(1539368457, 1),
 "$gleStats" : {
  "lastOpTime" : Timestamp(1539368457, 1),
  "electionId" : ObjectId("000000000000000000000000")
 },
 "lastCommittedOpTime" : Timestamp(0, 0),
 "$clusterTime" : {
  "clusterTime" : Timestamp(1539368457, 1),
  "signature" : {
   "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
   "keyId" : NumberLong(0)
  }
 }
}
```

Use MongoDB Shell to log in to mongos.

```
mongos --configdb localhost:27100 --port 40000
mongo localhost:40000/admin
```

Add Shard nodes and shard the cities by the country.

```
> db.runCommand( {addshard: "localhost:27014" })
{"shardAdded" : "shard0000", "ok" : 1 }

> db.runCommand( {addshard: "localhost:27015" })
{"shardAdded" : "shard0001", "ok" : 1}

> db.runCommand({enablesharding: "test"})
{ "ok" : 1}

> db.runCommand({shardcollection:"test.cities",key:{name:1}})
{"collectionsharded":{"collectionsharded" :"test.cities", "ok" : 1 }
```