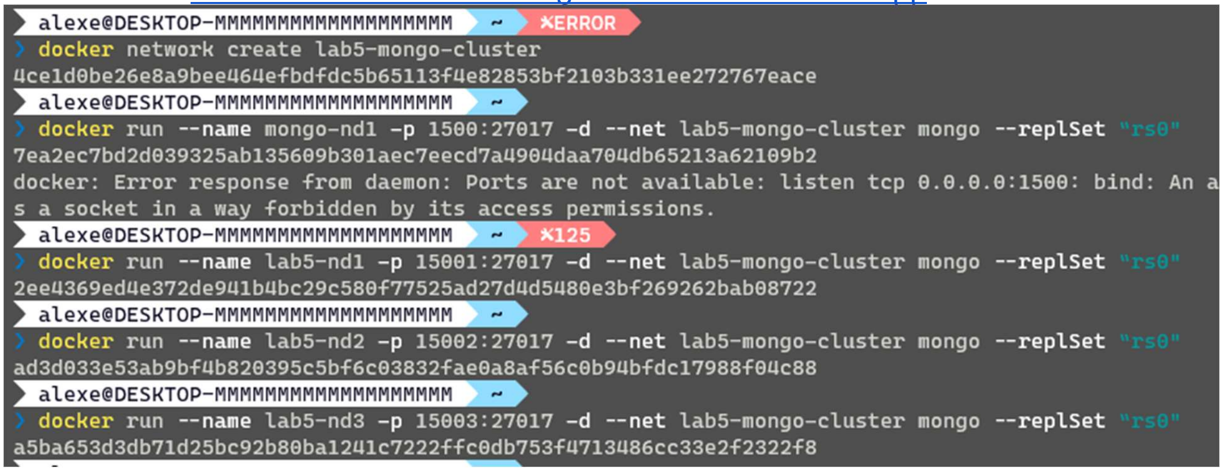
Task 5 - Налаштування реплікації та перевірка відмовостійкості MongoDB

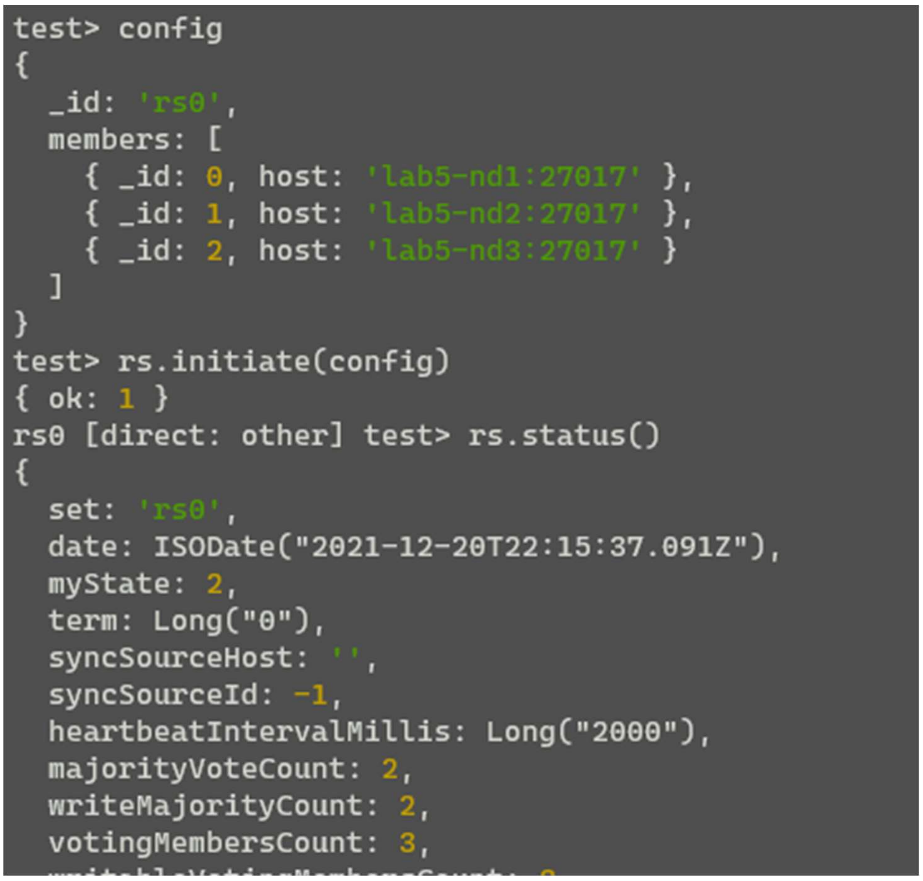
Ознайомтесь з реплікацію даних в MongoDB
http://docs.mongodb.org/manual/core/replication-introduction/

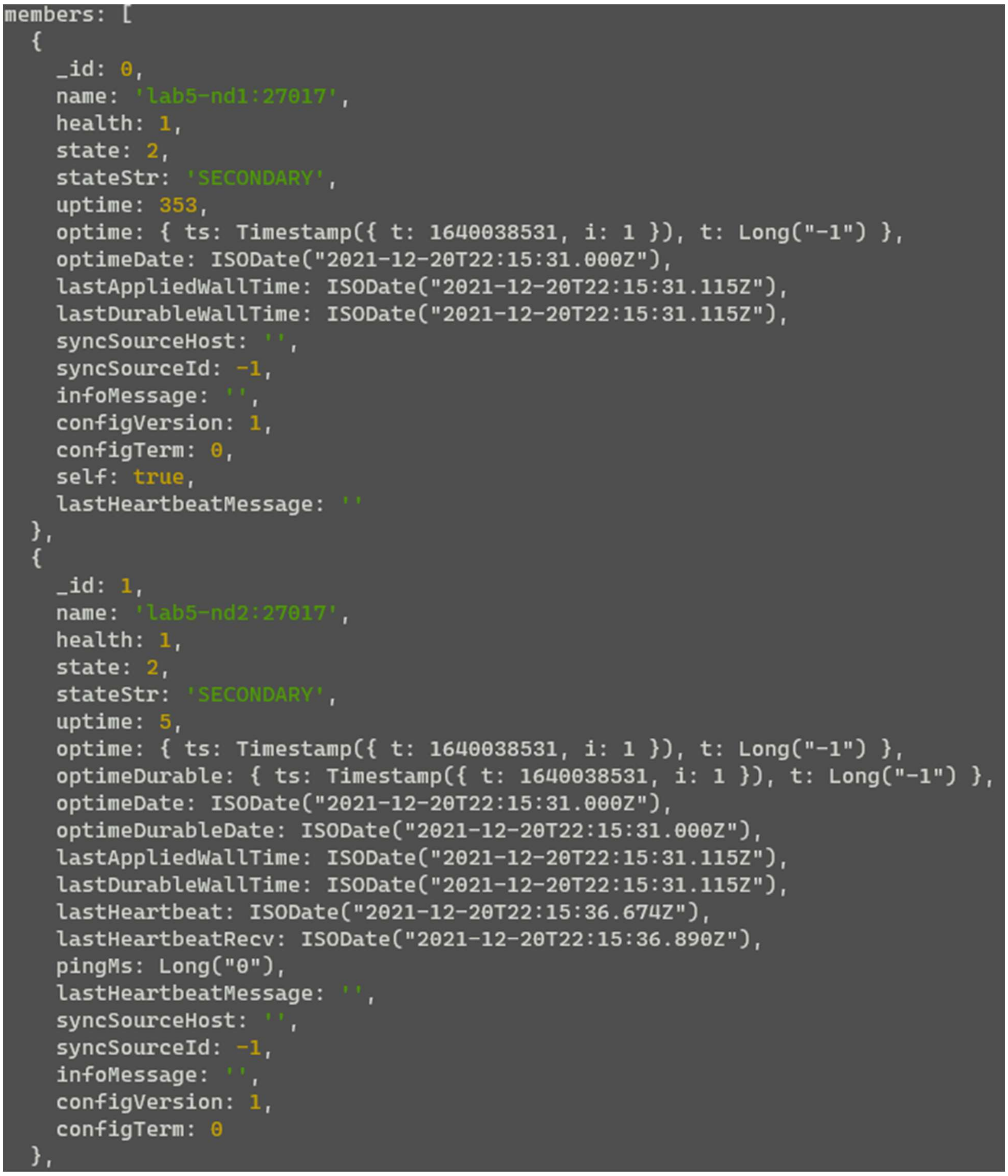
Завдання:

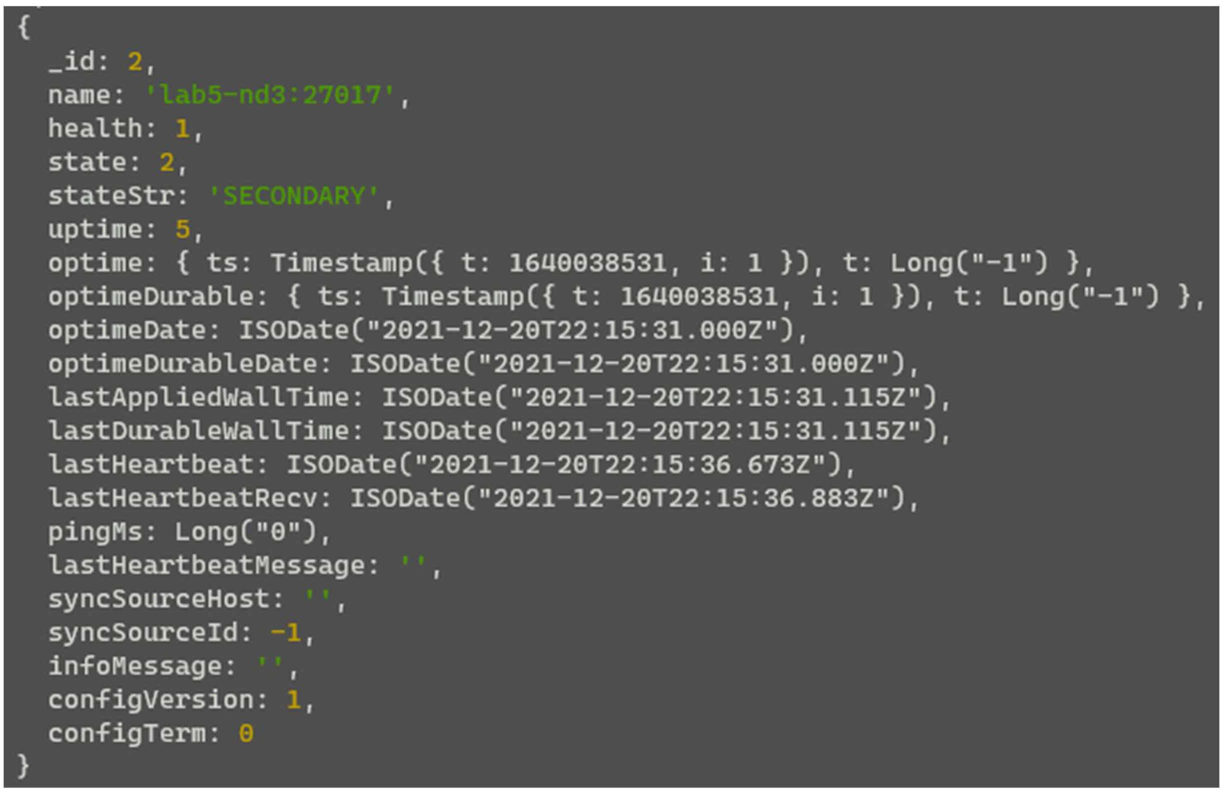
1. Налаштувати реплікацію в конфігурації: Primary with Two Secondary Members (всі ноди можуть бути запущені як окремі процеси або у Docker контейнерах) - http://docs.mongodb.org/manual/core/replica-set-architecture-three-members/
   - Deploy a Replica Set for Testing and Development- http://docs.mongodb.org/manual/tutorial/deploy-replica-set-for-testing/
   - http://www.tugberkugurlu.com/archive/setting-up-a-mongodb-replica-set-with-docker-and-connecting-to-it-with-a--net-core-app



```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~  ✘ERROR
❯ docker network create lab5-mongo-cluster
4ce1d0be26e8a9bee464efbdfdc5b65113f4e82853bf2103b331ee272767eace
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker run --name mongo-nd1 -p 1500:27017 -d --net lab5-mongo-cluster mongo --replSet "rs0"
7ea2ec7bd2d039325ab135609b301aec7eecd7a4904daa704db65213a62109b2
docker: Error response from daemon: Ports are not available: listen tcp 0.0.0.0:1500: bind: An a
s a socket in a way forbidden by its access permissions.
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~  ✘125
❯ docker run --name lab5-nd1 -p 15001:27017 -d --net lab5-mongo-cluster mongo --replSet "rs0"
2ee4369ed4e372de941b4bc29c580f77525ad27d4d5480e3bf269262bab08722
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker run --name lab5-nd2 -p 15002:27017 -d --net lab5-mongo-cluster mongo --replSet "rs0"
ad3d033e53ab9bf4b820395c5bf6c03832fae0a8af56c0b94bfdc17988f04c88
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker run --name lab5-nd3 -p 15003:27017 -d --net lab5-mongo-cluster mongo --replSet "rs0"
a5ba653d3db71d25bc92b80ba1241c7222ffc0db753f4713486cc33e2f2322f8
```

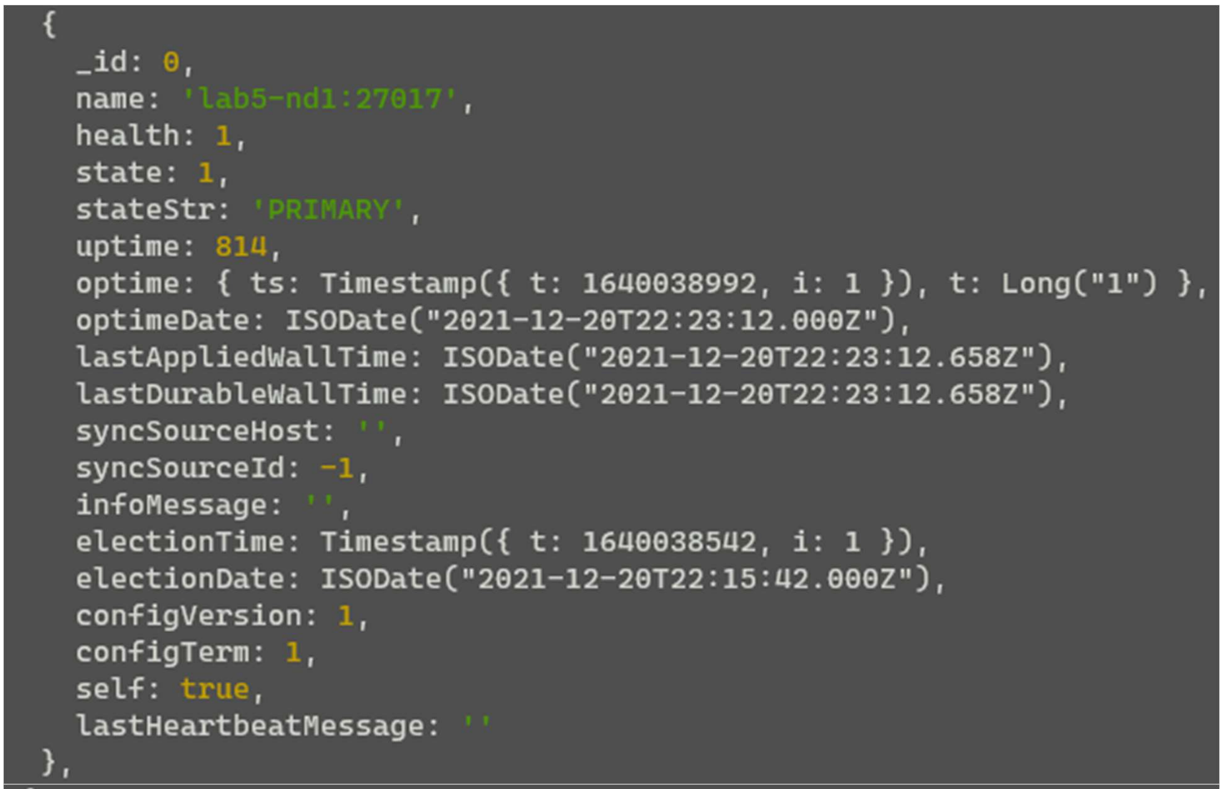

```
test> config
{
  _id: 'rs0',
  members: [
    { _id: 0, host: 'lab5-nd1:27017' },
    { _id: 1, host: 'lab5-nd2:27017' },
    { _id: 2, host: 'lab5-nd3:27017' }
  ]
}
test> rs.initiate(config)
{ ok: 1 }
rs0 [direct: other] test> rs.status()
{
  set: 'rs0',
  date: ISODate("2021-12-20T22:15:37.091Z"),
  myState: 2,
  term: Long("0"),
  syncSourceHost: '',
  syncSourceId: -1,
  heartbeatIntervalMillis: Long("2000"),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
```


```
members: [
  {
    _id: 0,
    name: 'lab5-nd1:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 353,
    optime: { ts: Timestamp({ t: 1640038531, i: 1 }), t: Long("-1") },
    optimeDate: ISODate("2021-12-20T22:15:31.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T22:15:31.115Z"),
    lastDurableWallTime: ISODate("2021-12-20T22:15:31.115Z"),
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    configVersion: 1,
    configTerm: 0,
    self: true,
    lastHeartbeatMessage: ''
  },
  {
    _id: 1,
    name: 'lab5-nd2:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 5,
    optime: { ts: Timestamp({ t: 1640038531, i: 1 }), t: Long("-1") },
    optimeDurable: { ts: Timestamp({ t: 1640038531, i: 1 }), t: Long("-1") },
    optimeDate: ISODate("2021-12-20T22:15:31.000Z"),
    optimeDurableDate: ISODate("2021-12-20T22:15:31.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T22:15:31.115Z"),
    lastDurableWallTime: ISODate("2021-12-20T22:15:31.115Z"),
    lastHeartbeat: ISODate("2021-12-20T22:15:36.674Z"),
    lastHeartbeatRecv: ISODate("2021-12-20T22:15:36.890Z"),
    pingMs: Long("0"),
    lastHeartbeatMessage: '',
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    configVersion: 1,
    configTerm: 0
  },
```

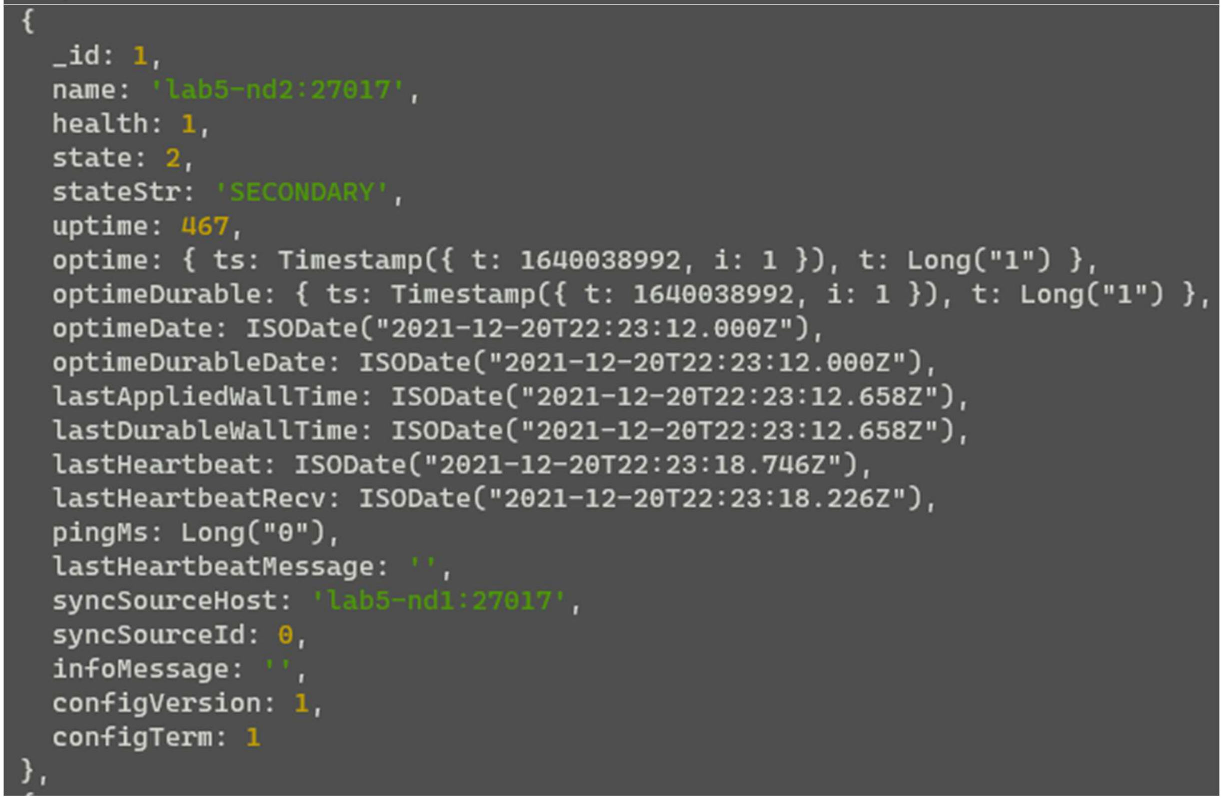


```
{
  _id: 2,
  name: 'lab5-nd3:27017',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 5,
  optime: { ts: Timestamp({ t: 1640038531, i: 1 }), t: Long("-1") },
  optimeDurable: { ts: Timestamp({ t: 1640038531, i: 1 }), t: Long("-1") },
  optimeDate: ISODate("2021-12-20T22:15:31.000Z"),
  optimeDurableDate: ISODate("2021-12-20T22:15:31.000Z"),
  lastAppliedWallTime: ISODate("2021-12-20T22:15:31.115Z"),
  lastDurableWallTime: ISODate("2021-12-20T22:15:31.115Z"),
  lastHeartbeat: ISODate("2021-12-20T22:15:36.673Z"),
  lastHeartbeatRecv: ISODate("2021-12-20T22:15:36.883Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '',
  syncSourceId: -1,
  infoMessage: '',
  configVersion: 1,
  configTerm: 0
}
```

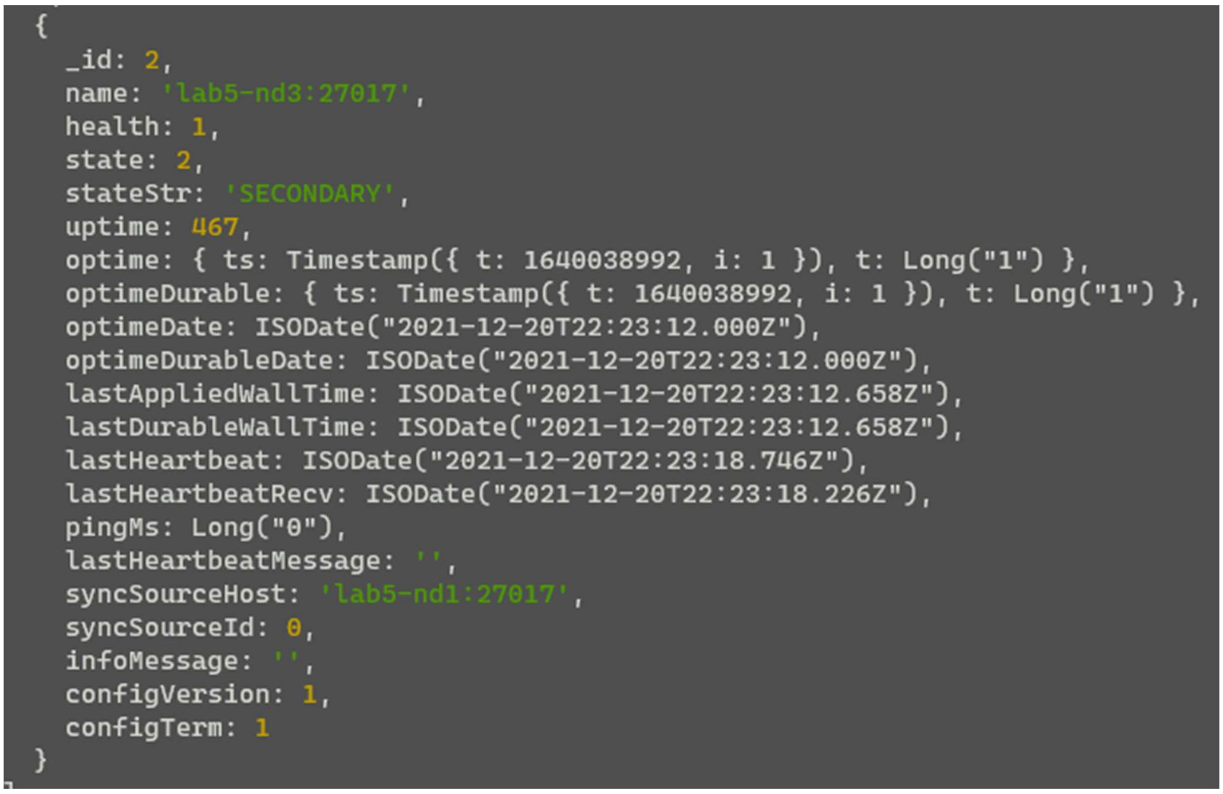
After, like, 5 minutes:



```
 {
   _id: 0,
   name: 'lab5-nd1:27017',
   health: 1,
   state: 1,
   stateStr: 'PRIMARY',
   uptime: 814,
   optime: { ts: Timestamp({ t: 1640038992, i: 1 }), t: Long("1") },
   optimeDate: ISODate("2021-12-20T22:23:12.000Z"),
   lastAppliedWallTime: ISODate("2021-12-20T22:23:12.658Z"),
   lastDurableWallTime: ISODate("2021-12-20T22:23:12.658Z"),
   syncSourceHost: '',
   syncSourceId: -1,
   infoMessage: '',
   electionTime: Timestamp({ t: 1640038542, i: 1 }),
   electionDate: ISODate("2021-12-20T22:15:42.000Z"),
   configVersion: 1,
   configTerm: 1,
   self: true,
   lastHeartbeatMessage: ''
 },
```



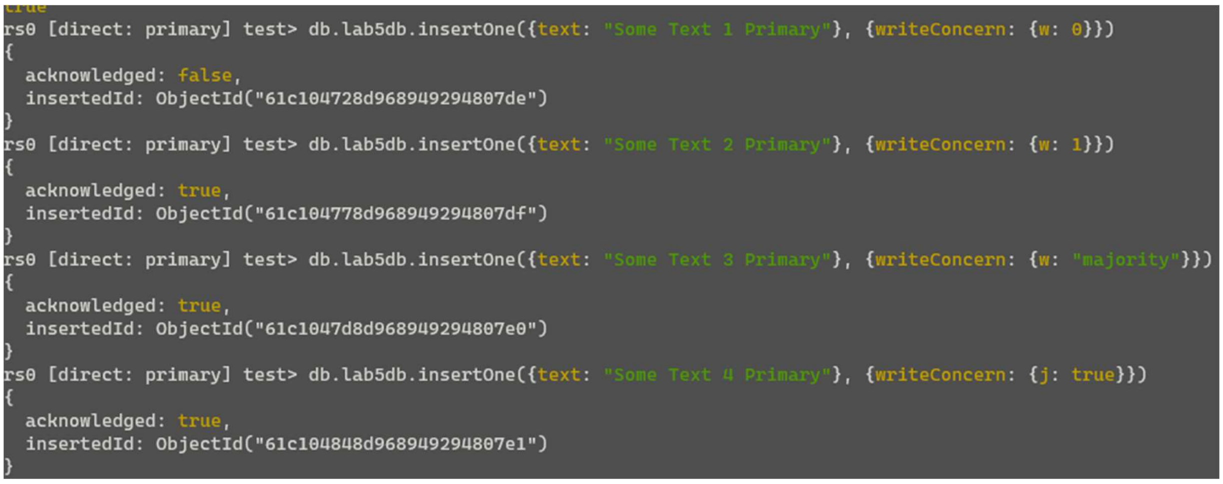
```
{
  _id: 1,
  name: 'lab5-nd2:27017',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 467,
  optime: { ts: Timestamp({ t: 1640038992, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1640038992, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2021-12-20T22:23:12.000Z"),
  optimeDurableDate: ISODate("2021-12-20T22:23:12.000Z"),
  lastAppliedWallTime: ISODate("2021-12-20T22:23:12.658Z"),
  lastDurableWallTime: ISODate("2021-12-20T22:23:12.658Z"),
  lastHeartbeat: ISODate("2021-12-20T22:23:18.746Z"),
  lastHeartbeatRecv: ISODate("2021-12-20T22:23:18.226Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: 'lab5-nd1:27017',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 1,
  configTerm: 1
},
```



```
  {
    _id: 2,
    name: 'lab5-nd3:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 467,
    optime: { ts: Timestamp({ t: 1640038992, i: 1 }), t: Long("1") },
    optimeDurable: { ts: Timestamp({ t: 1640038992, i: 1 }), t: Long("1") },
    optimeDate: ISODate("2021-12-20T22:23:12.000Z"),
    optimeDurableDate: ISODate("2021-12-20T22:23:12.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T22:23:12.658Z"),
    lastDurableWallTime: ISODate("2021-12-20T22:23:12.658Z"),
    lastHeartbeat: ISODate("2021-12-20T22:23:18.746Z"),
    lastHeartbeatRecv: ISODate("2021-12-20T22:23:18.226Z"),
    pingMs: Long("0"),
    lastHeartbeatMessage: '',
    syncSourceHost: 'lab5-nd1:27017',
    syncSourceId: 0,
    infoMessage: '',
    configVersion: 1,
    configTerm: 1
  }
```

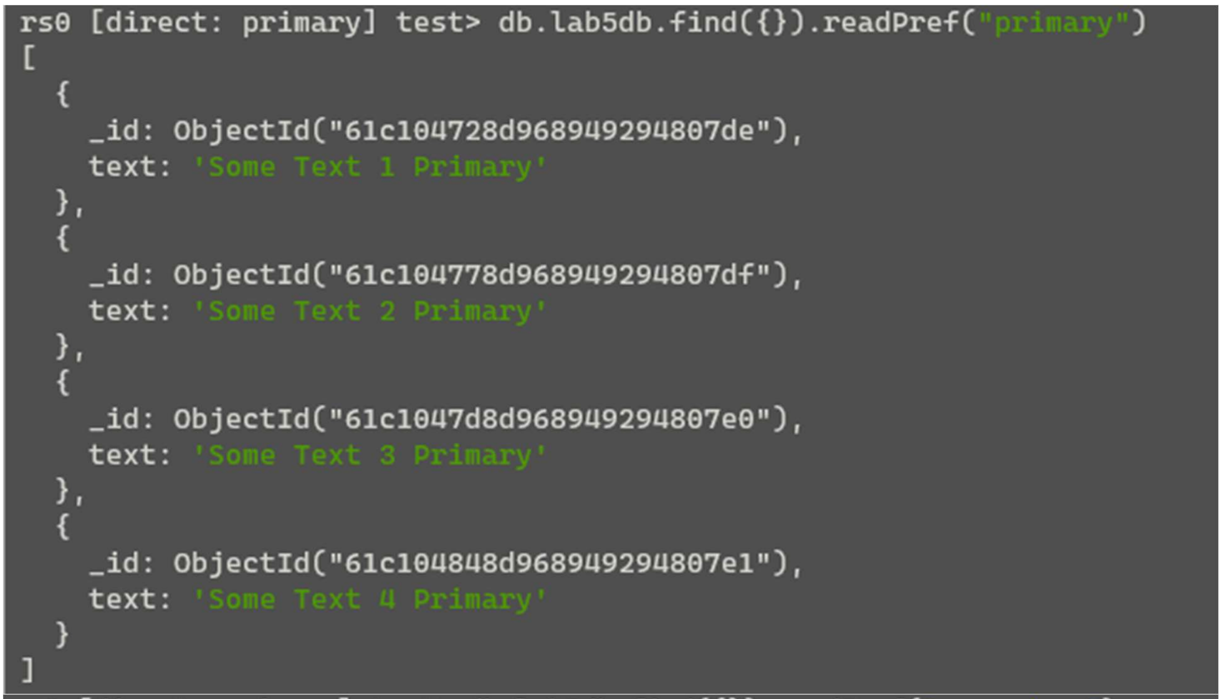
2.	Продемонструвати запис даних на *primary node* з різними Write Concern Levels (http://docs.mongodb.org/manual/core/write-concern/):

- Unacknowledged
- Acknowledged
- Journaled
- AcknowledgedReplica (http://docs.mongodb.org/manual/core/replica-set-write-concern/)

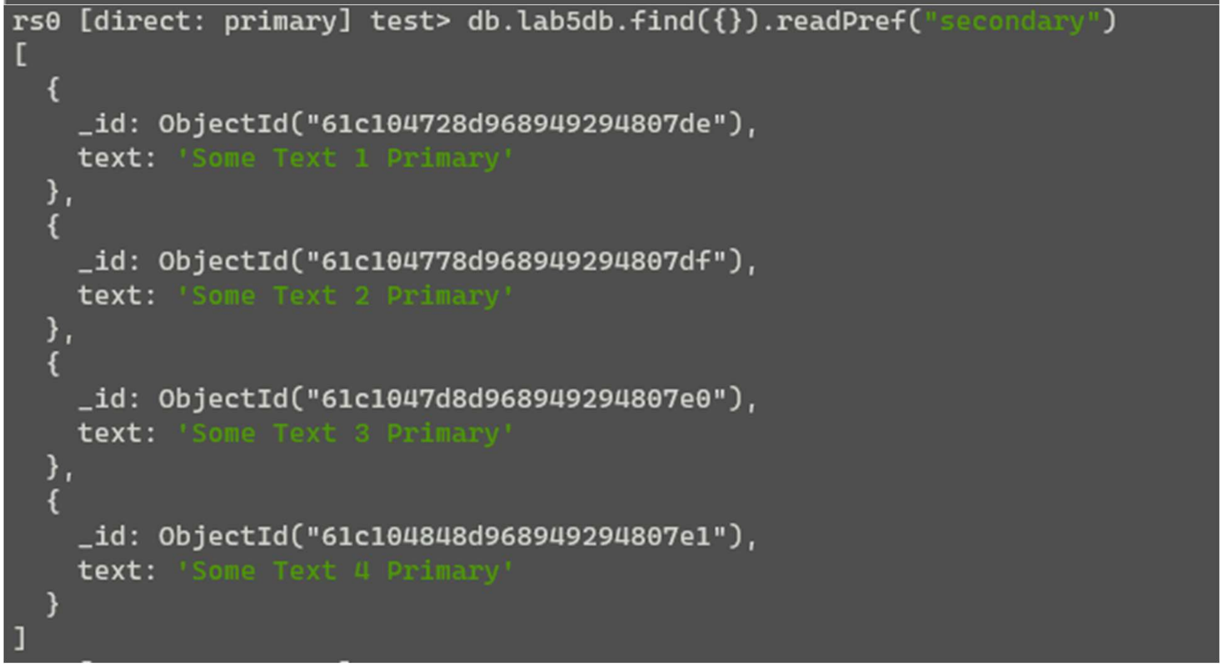


```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 1 Primary"}, {writeConcern: {w: 0}})
{
  acknowledged: false,
  insertedId: ObjectId("61c104728d968949294807de")
}
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 2 Primary"}, {writeConcern: {w: 1}})
{
  acknowledged: true,
  insertedId: ObjectId("61c104778d968949294807df")
}
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 3 Primary"}, {writeConcern: {w: "majority"}})
{
  acknowledged: true,
  insertedId: ObjectId("61c1047d8d968949294807e0")
}
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 4 Primary"}, {writeConcern: {j: true}})
{
  acknowledged: true,
  insertedId: ObjectId("61c104848d968949294807e1")
}
```

3.	Продемонструвати Read Preference Modes: читання з *primary* і *secondary* node (http://docs.mongodb.org/manual/core/read-preference/)

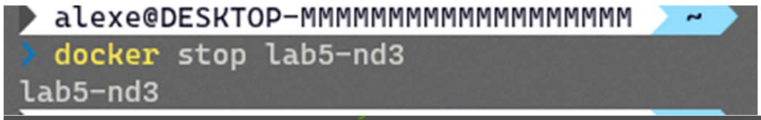


```
rs0 [direct: primary] test> db.lab5db.find({}).readPref("primary")
[
  {
    _id: ObjectId("61c104728d968949294807de"),
    text: 'Some Text 1 Primary'
  },
  {
    _id: ObjectId("61c104778d968949294807df"),
    text: 'Some Text 2 Primary'
  },
  {
    _id: ObjectId("61c1047d8d968949294807e0"),
    text: 'Some Text 3 Primary'
  },
  {
    _id: ObjectId("61c104848d968949294807e1"),
    text: 'Some Text 4 Primary'
  }
]
```

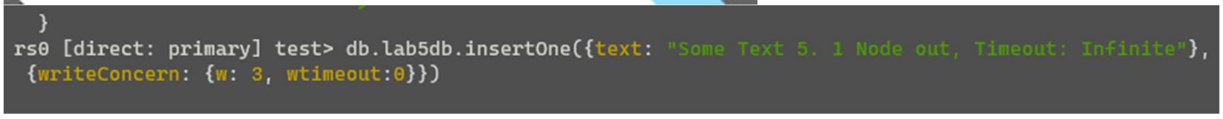


```
rs0 [direct: primary] test> db.lab5db.find({}).readPref("secondary")
[
  {
    _id: ObjectId("61c104728d968949294807de"),
    text: 'Some Text 1 Primary'
  },
  {
    _id: ObjectId("61c104778d968949294807df"),
    text: 'Some Text 2 Primary'
  },
  {
    _id: ObjectId("61c1047d8d968949294807e0"),
    text: 'Some Text 3 Primary'
  },
  {
    _id: ObjectId("61c104848d968949294807e1"),
    text: 'Some Text 4 Primary'
  }
]
```

4. Спробувати зробити запис з однією відключеною нодою та *write concern* рівнім 3 та нескінченім таймаутом. Спробувати під час таймаута включити відключену ноду

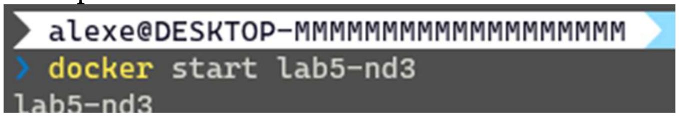


```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯ docker stop lab5-nd3
lab5-nd3
```

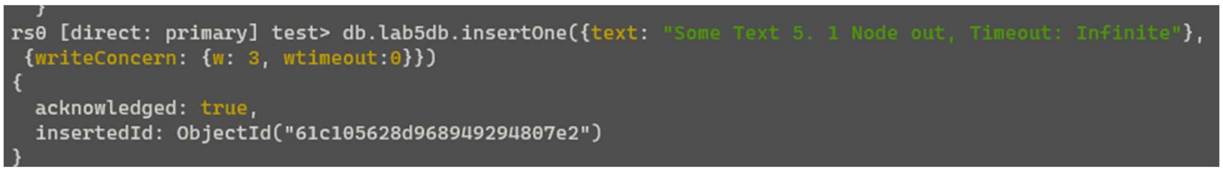


```
  }
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 5. 1 Node out, Timeout: Infinite"},
 {writeConcern: {w: 3, wtimeout:0}})
```

As our timeout set to infinity: we are waiting infinite amount of time till we will have 3 nodes in our replica set.



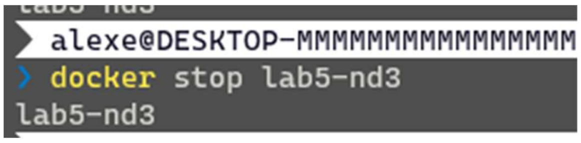
```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM
❯ docker start lab5-nd3
lab5-nd3
```



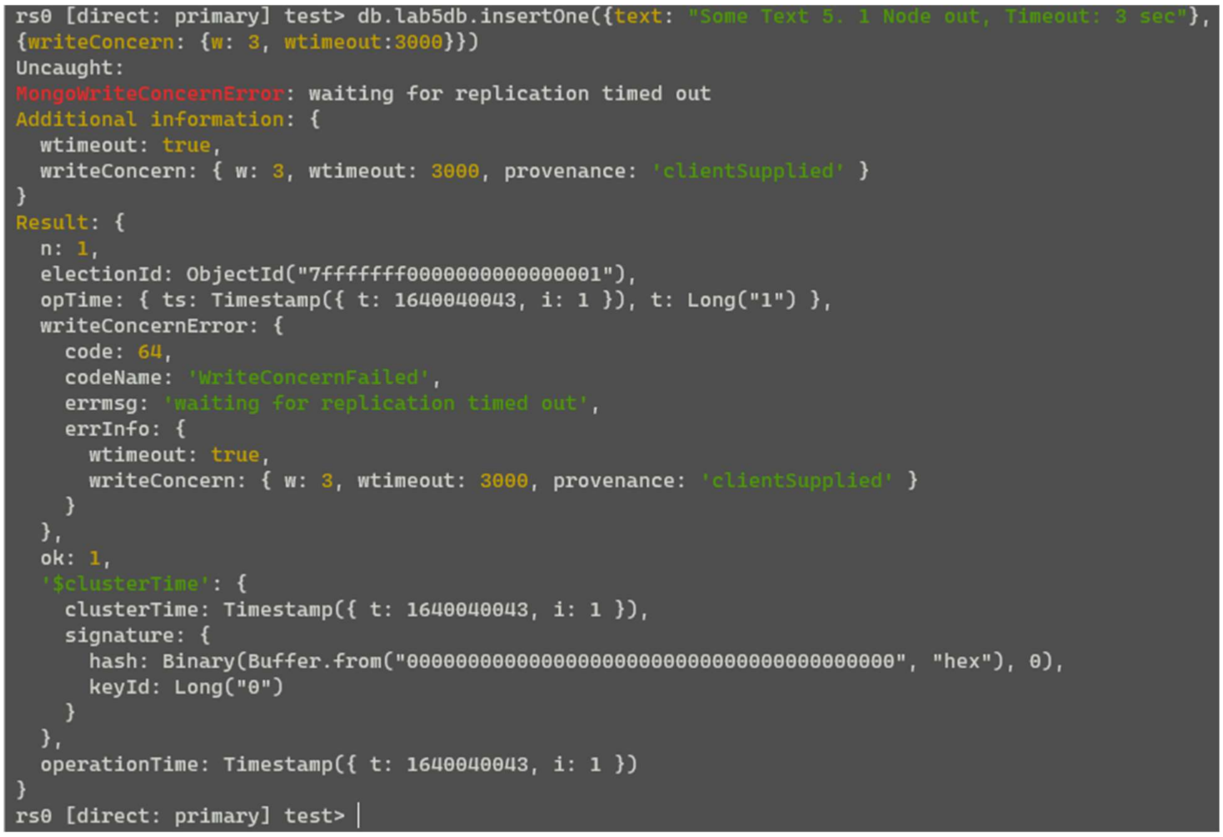
```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 5. 1 Node out, Timeout: Infinite"},
 {writeConcern: {w: 3, wtimeout:0}})
{
  acknowledged: true,
  insertedId: ObjectId("61c105628d968949294807e2")
}
```

Row inserted as soon as it received third node

5. Аналогічно попередньому пункту, але задати скінченний таймаут та дочекатись його закінчення. Перевірити чи данні записались і чи доступні на чиіання з рівнем *readConcern: “majority”*

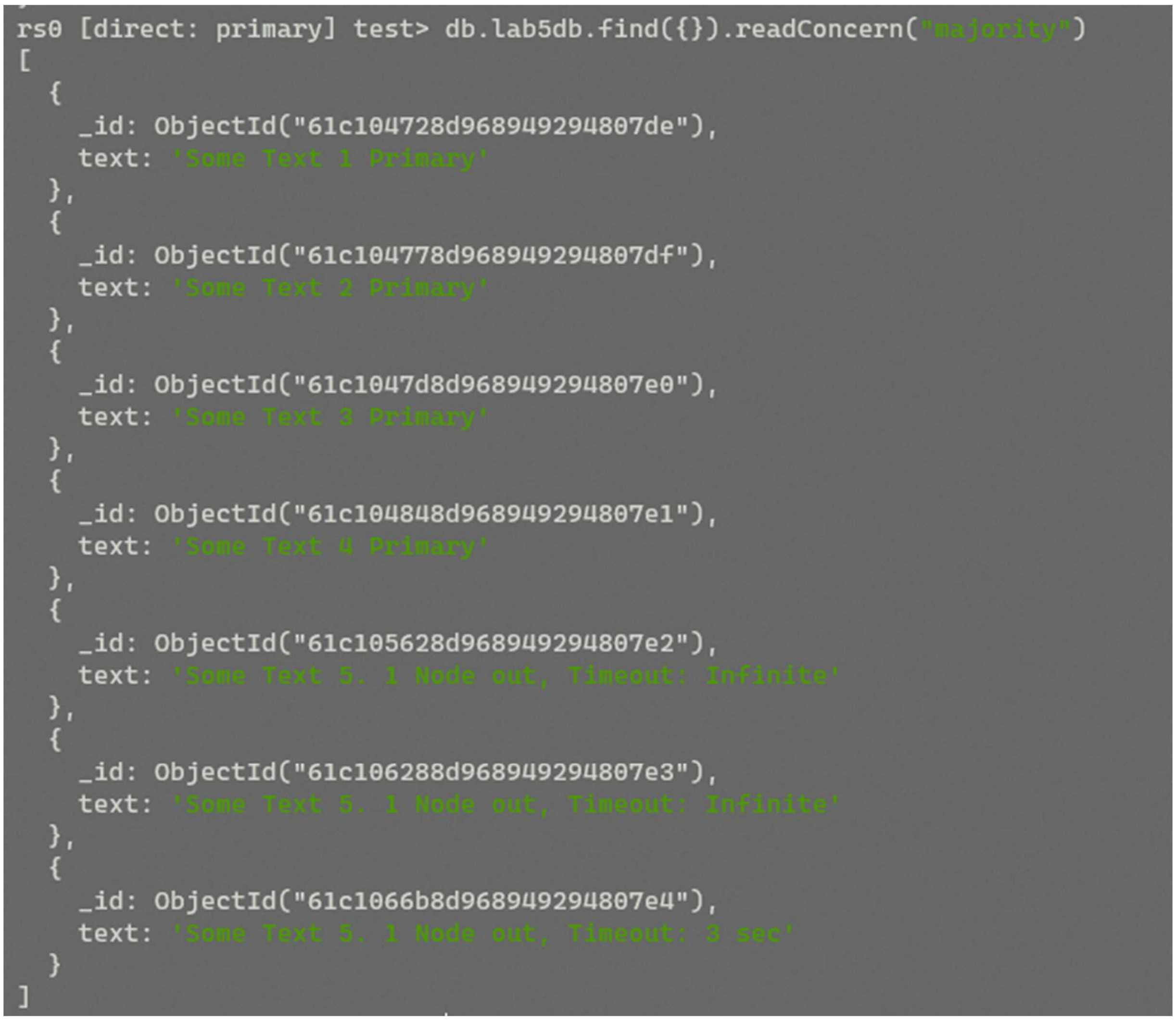


```
alexe@DESKTOP-MMMMMMMMMMMMMMMM
❯ docker stop lab5-nd3
lab5-nd3
```

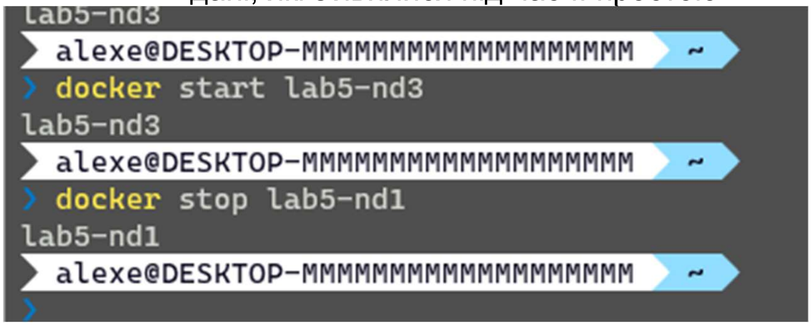


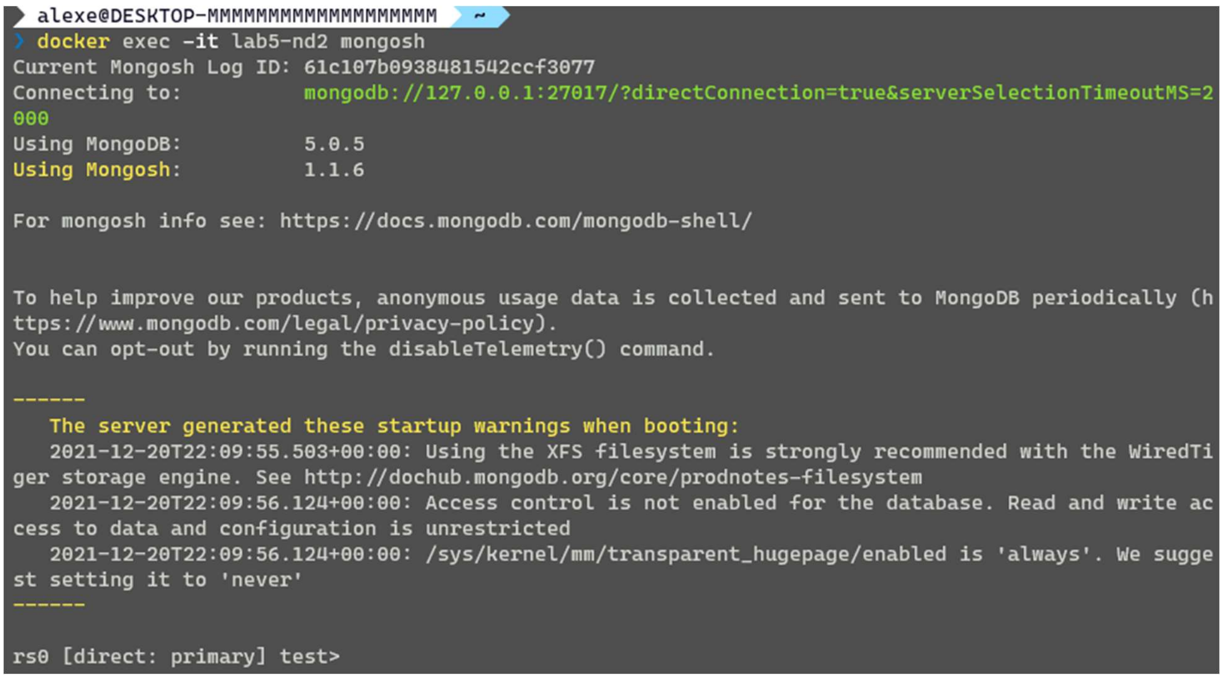
```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text 5. 1 Node out, Timeout: 3 sec"},
{writeConcern: {w: 3, wtimeout:3000}})
Uncaught:
MongoWriteConcernError: waiting for replication timed out
Additional information: {
  wtimeout: true,
  writeConcern: { w: 3, wtimeout: 3000, provenance: 'clientSupplied' }
}
Result: {
  n: 1,
  electionId: ObjectId("7fffffff0000000000000001"),
  opTime: { ts: Timestamp({ t: 1640040043, i: 1 }), t: Long("1") },
  writeConcernError: {
    code: 64,
    codeName: 'WriteConcernFailed',
    errmsg: 'waiting for replication timed out',
    errInfo: {
      wtimeout: true,
      writeConcern: { w: 3, wtimeout: 3000, provenance: 'clientSupplied' }
    }
  },
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1640040043, i: 1 }),
    signature: {
      hash: Binary(Buffer.from("0000000000000000000000000000000000000000", "hex"), 0),
      keyId: Long("0")
    }
  },
  operationTime: Timestamp({ t: 1640040043, i: 1 })
}
rs0 [direct: primary] test>
```

As we can see from the output we got time-out exception. But it also said: ”ok: 1”. This means that server has written this value into itself and replicated to available nodes:


```
rs0 [direct: primary] test> db.lab5db.find({}).readConcern("majority")
[
  {
    _id: ObjectId("61c104728d968949294807de"),
    text: 'Some Text 1 Primary'
  },
  {
    _id: ObjectId("61c104778d968949294807df"),
    text: 'Some Text 2 Primary'
  },
  {
    _id: ObjectId("61c1047d8d968949294807e0"),
    text: 'Some Text 3 Primary'
  },
  {
    _id: ObjectId("61c104848d968949294807e1"),
    text: 'Some Text 4 Primary'
  },
  {
    _id: ObjectId("61c105628d968949294807e2"),
    text: 'Some Text 5. 1 Node out, Timeout: Infinite'
  },
  {
    _id: ObjectId("61c106288d968949294807e3"),
    text: 'Some Text 5. 1 Node out, Timeout: Infinite'
  },
  {
    _id: ObjectId("61c1066b8d968949294807e4"),
    text: 'Some Text 5. 1 Node out, Timeout: 3 sec'
  }
]
```
Why we can access them with ‘majority’? Well because our available ‘majority’ are 2 nodes only.

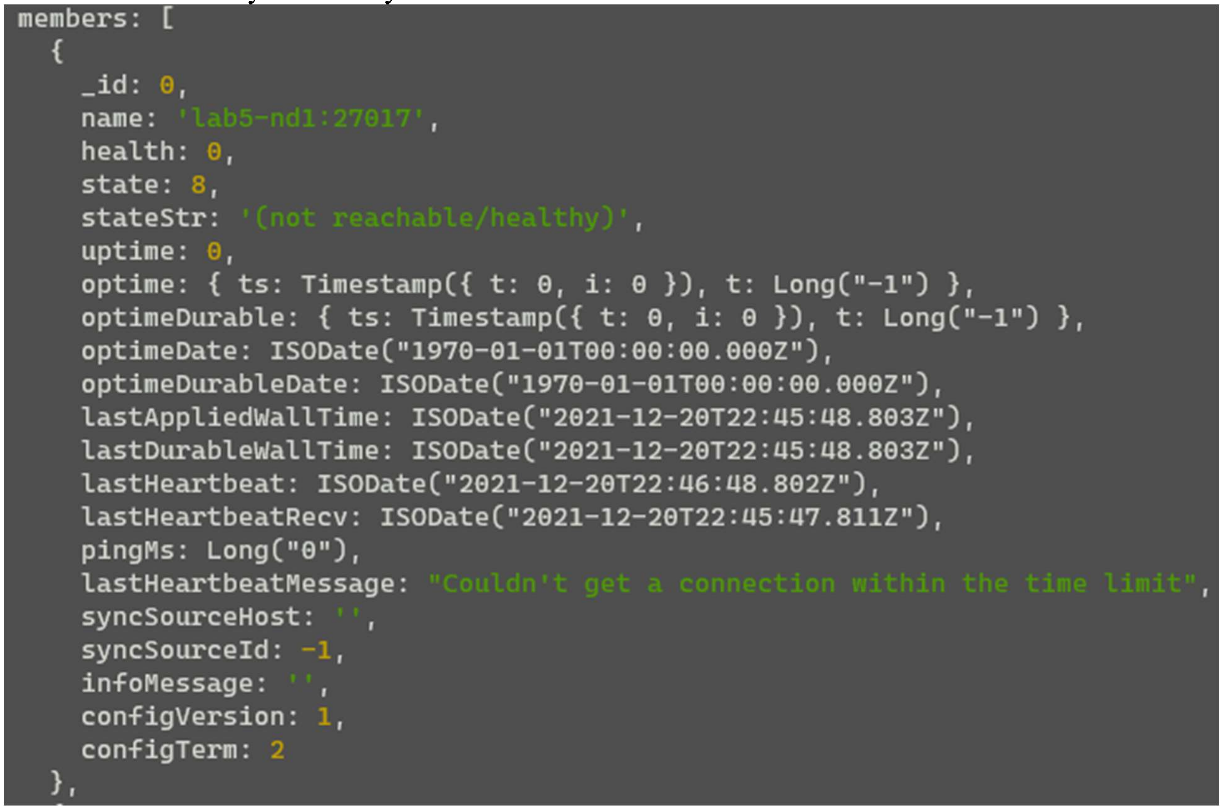
6. Продемонстрував перевибори primary node в відключивши поточний primary (Replica Set Elections) - http://docs.mongodb.org/manual/core/replica-set-elections/
   - і що після відновлення роботи старої primary на неї реплікуються нові дані, які з'явилися під час її простою


```
lab5-nd3
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯ docker start lab5-nd3
lab5-nd3
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯ docker stop lab5-nd1
lab5-nd1
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯
```



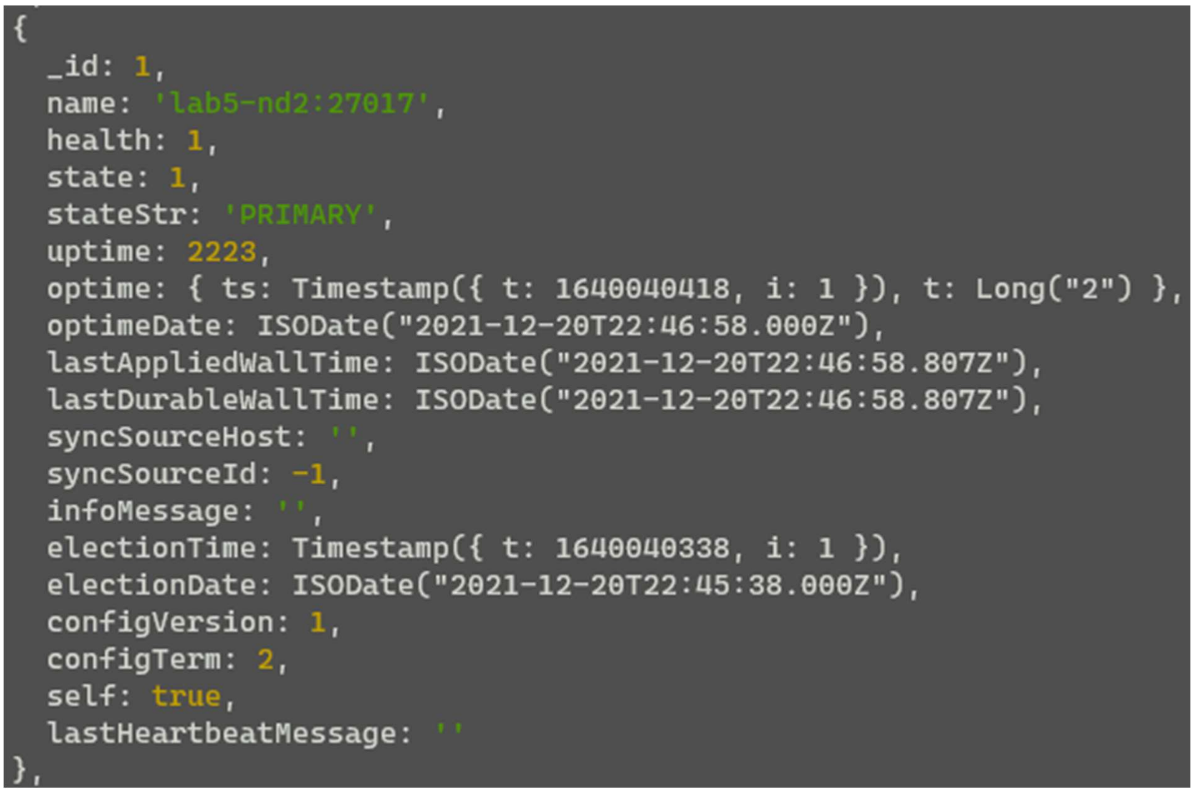
```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker exec -it lab5-nd2 mongosh
Current Mongosh Log ID: 61c107b0938481542ccf3077
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2
000
Using MongoDB:          5.0.5
Using Mongosh:          1.1.6

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (h
ttps://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting:
   2021-12-20T22:09:55.503+00:00: Using the XFS filesystem is strongly recommended with the WiredTi
ger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2021-12-20T22:09:56.124+00:00: Access control is not enabled for the database. Read and write ac
cess to data and configuration is unrestricted
   2021-12-20T22:09:56.124+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We sugge
st setting it to 'never'
------

rs0 [direct: primary] test>
```

Second node already is Primary node



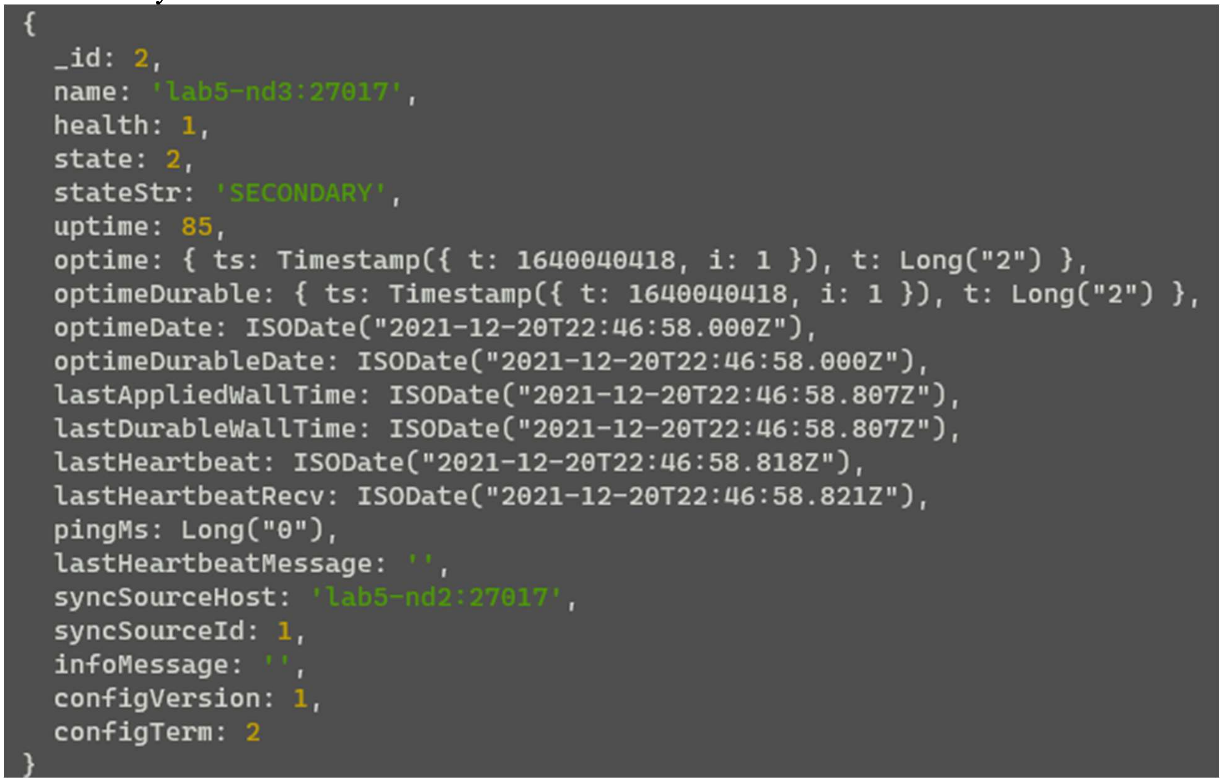
```
members: [
  {
    _id: 0,
    name: 'lab5-nd1:27017',
    health: 0,
    state: 8,
    stateStr: '(not reachable/healthy)',
    uptime: 0,
    optime: { ts: Timestamp({ t: 0, i: 0 }), t: Long("-1") },
    optimeDurable: { ts: Timestamp({ t: 0, i: 0 }), t: Long("-1") },
    optimeDate: ISODate("1970-01-01T00:00:00.000Z"),
    optimeDurableDate: ISODate("1970-01-01T00:00:00.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T22:45:48.803Z"),
    lastDurableWallTime: ISODate("2021-12-20T22:45:48.803Z"),
    lastHeartbeat: ISODate("2021-12-20T22:46:48.802Z"),
    lastHeartbeatRecv: ISODate("2021-12-20T22:45:47.811Z"),
    pingMs: Long("0"),
    lastHeartbeatMessage: "Couldn't get a connection within the time limit",
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    configVersion: 1,
    configTerm: 2
  },
```

Disconnected old 'PRIMARY' node1



```
{
  _id: 1,
  name: 'lab5-nd2:27017',
  health: 1,
  state: 1,
  stateStr: 'PRIMARY',
  uptime: 2223,
  optime: { ts: Timestamp({ t: 1640040418, i: 1 }), t: Long("2") },
  optimeDate: ISODate("2021-12-20T22:46:58.000Z"),
  lastAppliedWallTime: ISODate("2021-12-20T22:46:58.807Z"),
  lastDurableWallTime: ISODate("2021-12-20T22:46:58.807Z"),
  syncSourceHost: '',
  syncSourceId: -1,
  infoMessage: '',
  electionTime: Timestamp({ t: 1640040338, i: 1 }),
  electionDate: ISODate("2021-12-20T22:45:38.000Z"),
  configVersion: 1,
  configTerm: 2,
  self: true,
  lastHeartbeatMessage: ''
},
```

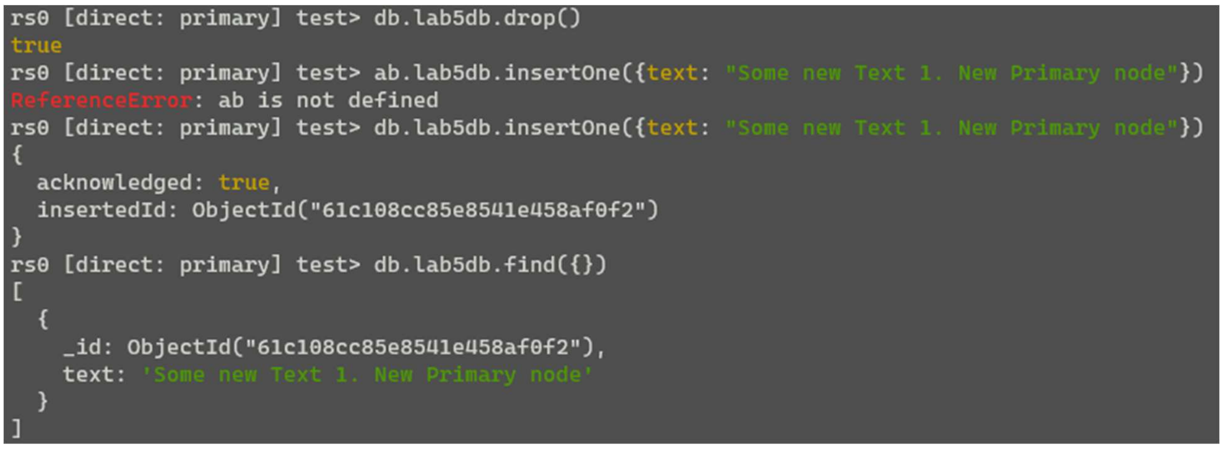
New Primary



```
 {
   _id: 2,
   name: 'lab5-nd3:27017',
   health: 1,
   state: 2,
   stateStr: 'SECONDARY',
   uptime: 85,
   optime: { ts: Timestamp({ t: 1640040418, i: 1 }), t: Long("2") },
   optimeDurable: { ts: Timestamp({ t: 1640040418, i: 1 }), t: Long("2") },
   optimeDate: ISODate("2021-12-20T22:46:58.000Z"),
   optimeDurableDate: ISODate("2021-12-20T22:46:58.000Z"),
   lastAppliedWallTime: ISODate("2021-12-20T22:46:58.807Z"),
   lastDurableWallTime: ISODate("2021-12-20T22:46:58.807Z"),
   lastHeartbeat: ISODate("2021-12-20T22:46:58.818Z"),
   lastHeartbeatRecv: ISODate("2021-12-20T22:46:58.821Z"),
   pingMs: Long("0"),
   lastHeartbeatMessage: '',
   syncSourceHost: 'lab5-nd2:27017',
   syncSourceId: 1,
   infoMessage: '',
   configVersion: 1,
   configTerm: 2
 }
```

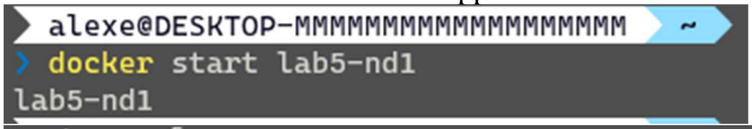
Only syncSourceHost changed to new Primary

Lets clean our db and write new data:

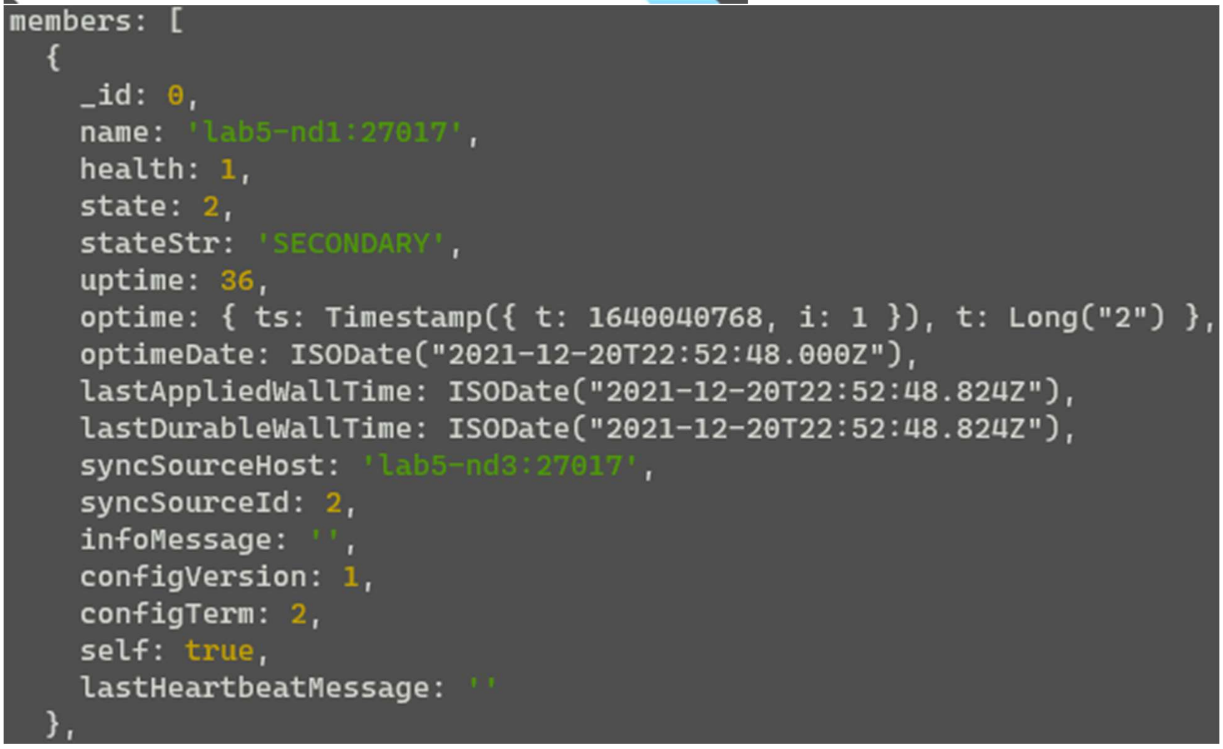


```
rs0 [direct: primary] test> db.lab5db.drop()
true
rs0 [direct: primary] test> ab.lab5db.insertOne({text: "Some new Text 1. New Primary node"})
ReferenceError: ab is not defined
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some new Text 1. New Primary node"})
{
  acknowledged: true,
  insertedId: ObjectId("61c108cc85e8541e458af0f2")
}
rs0 [direct: primary] test> db.lab5db.find({})
[
  {
    _id: ObjectId("61c108cc85e8541e458af0f2"),
    text: 'Some new Text 1. New Primary node'
  }
]
```

Lets start our node1 and see what happens:

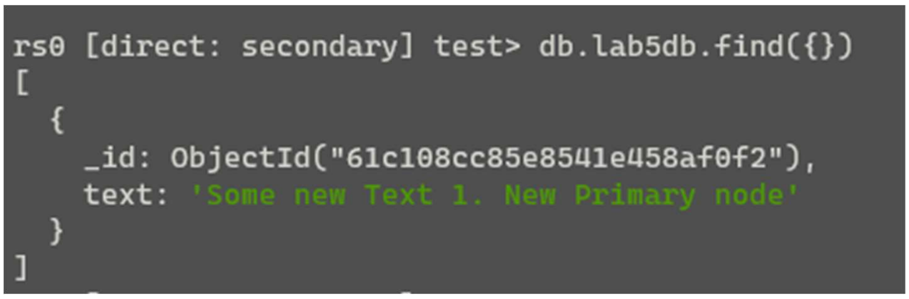


```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker start lab5-nd1
lab5-nd1
```



```
members: [
  {
    _id: 0,
    name: 'lab5-nd1:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 36,
    optime: { ts: Timestamp({ t: 1640040768, i: 1 }), t: Long("2") },
    optimeDate: ISODate("2021-12-20T22:52:48.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T22:52:48.824Z"),
    lastDurableWallTime: ISODate("2021-12-20T22:52:48.824Z"),
    syncSourceHost: 'lab5-nd3:27017',
    syncSourceId: 2,
    infoMessage: '',
    configVersion: 1,
    configTerm: 2,
    self: true,
    lastHeartbeatMessage: ''
  },
```

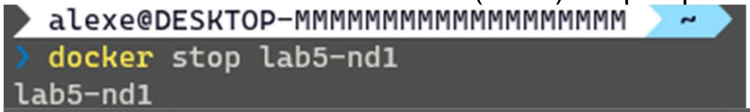
It successfully deleted and inserted all data:



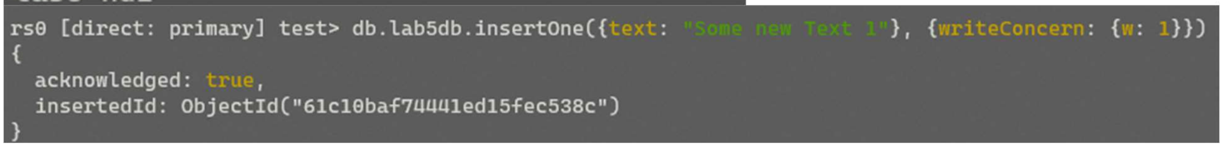
```
rs0 [direct: secondary] test> db.lab5db.find({})
[
  {
    _id: ObjectId("61c108cc85e8541e458af0f2"),
    text: 'Some new Text 1. New Primary node'
  }
]
```

7. Привести кластер до неконсистентного стану користуючись моментом часу коли *primary node* не відразу помічає відсутність *secondary node*

- відключити останню *secondary node* та протягом 5 сек. на мастері записати значення (з w:1) і перевірити, що воно записалось



```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯ docker stop lab5-nd1
lab5-nd1
```

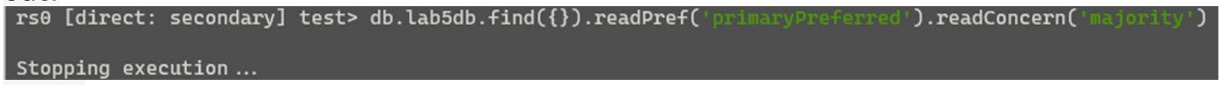


```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some new Text 1"}, {writeConcern: {w: 1}})
{
  acknowledged: true,
  insertedId: ObjectId("61c10baf74441ed15fec538c")
}
```

Written with instant success

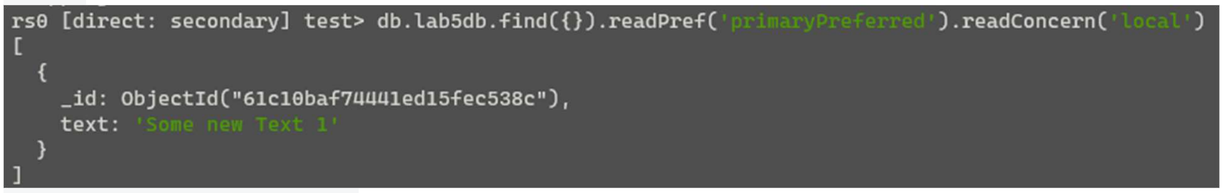
- спробувати зчитати це значення з різними рівнями *read concern* - readConcern: {level: <"majority"|"local"| "linearizable">}

Majority don't return any result. The result is seemed to be like we have infinite time-out:



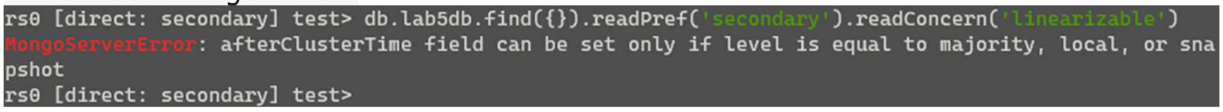
```
rs0 [direct: secondary] test> db.lab5db.find({}).readPref('primaryPreferred').readConcern('majority')

Stopping execution...
```

Local:



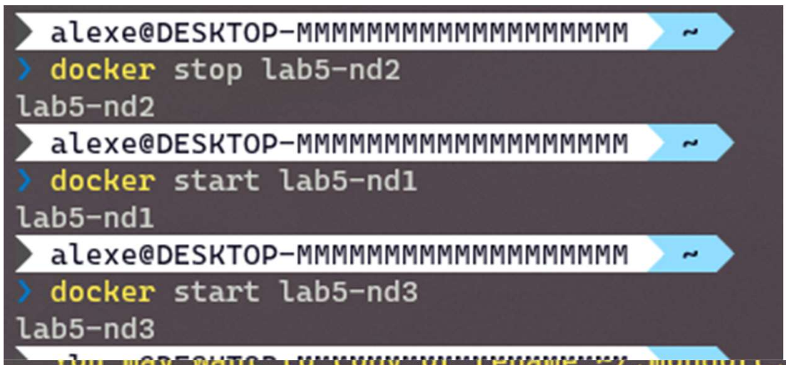
```
rs0 [direct: secondary] test> db.lab5db.find({}).readPref('primaryPreferred').readConcern('local')
[
  {
    _id: ObjectId("61c10baf74441ed15fec538c"),
    text: 'Some new Text 1'
  }
]
```

Linearizable we got error:

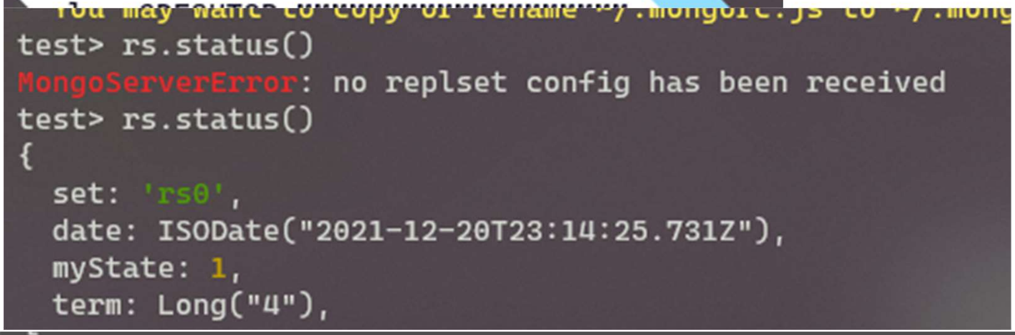


```
rs0 [direct: secondary] test> db.lab5db.find({}).readPref('secondary').readConcern('linearizable')
MongoServerError: afterClusterTime field can be set only if level is equal to majority, local, or sna
pshot
rs0 [direct: secondary] test>
```

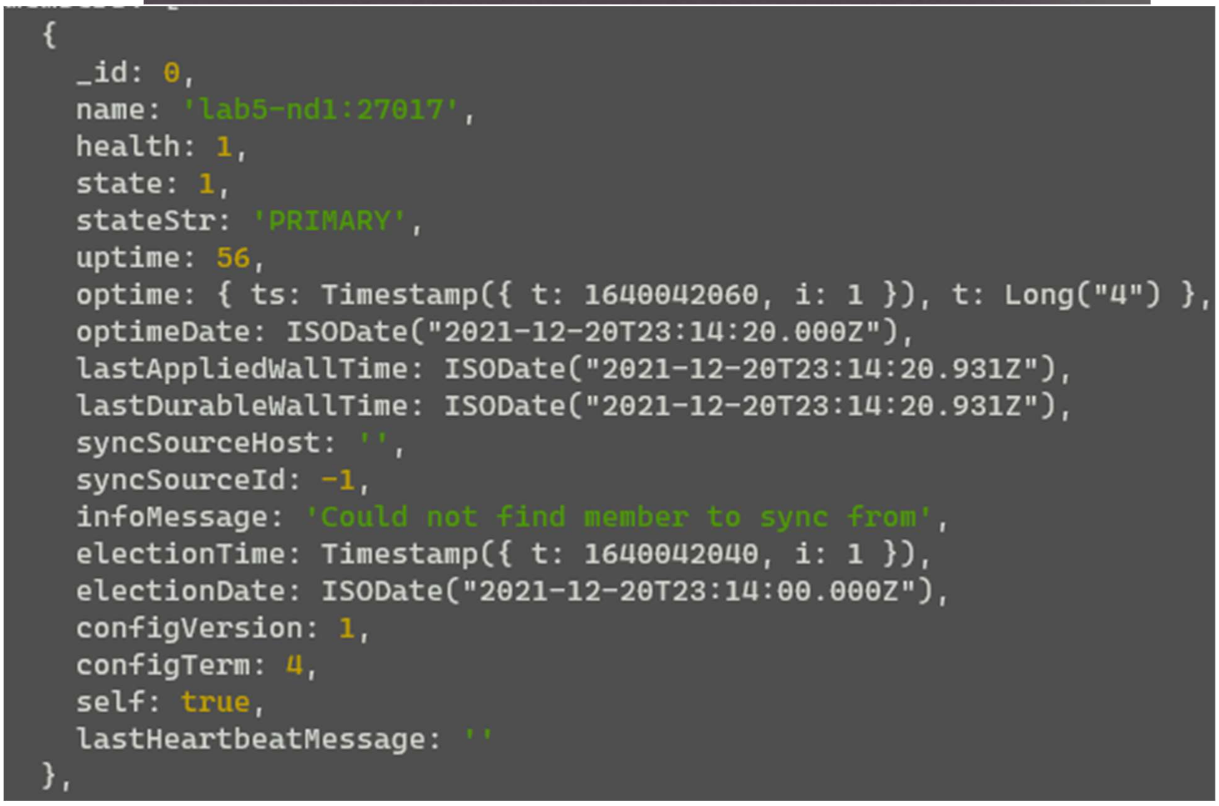
- включити дві інші ноди таким чином, щоб вони не бачили попереднього мастера (його можна відключити) і дочекатись поки вони оберуть нового мастера



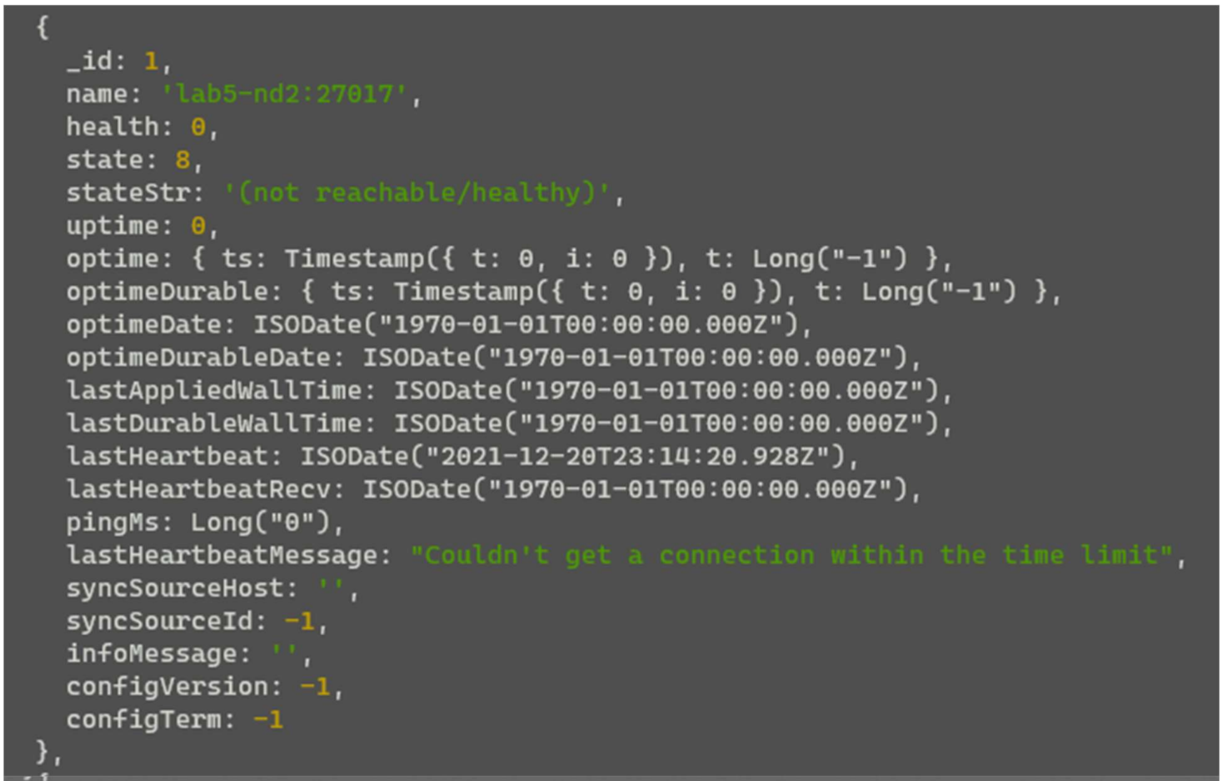
```
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker stop lab5-nd2
lab5-nd2
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker start lab5-nd1
lab5-nd1
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker start lab5-nd3
lab5-nd3
```



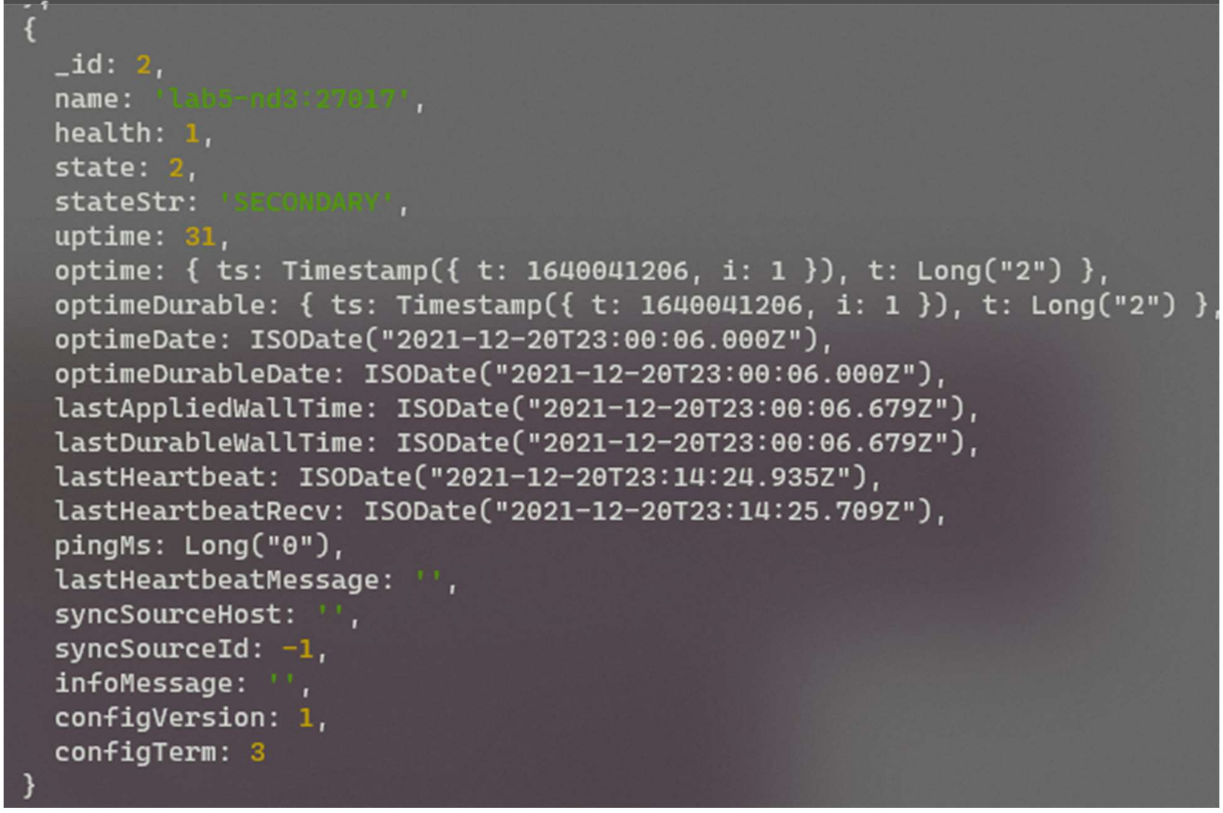
```
  You may want to copy or rename ~/.mongorc.js to ~/.mong
test> rs.status()
MongoServerError: no replset config has been received
test> rs.status()
{
  set: 'rs0',
  date: ISODate("2021-12-20T23:14:25.731Z"),
  myState: 1,
  term: Long("4"),
```



```
  {
    _id: 0,
    name: 'lab5-nd1:27017',
    health: 1,
    state: 1,
    stateStr: 'PRIMARY',
    uptime: 56,
    optime: { ts: Timestamp({ t: 1640042060, i: 1 }), t: Long("4") },
    optimeDate: ISODate("2021-12-20T23:14:20.000Z"),
    lastAppliedWallTime: ISODate("2021-12-20T23:14:20.931Z"),
    lastDurableWallTime: ISODate("2021-12-20T23:14:20.931Z"),
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: 'Could not find member to sync from',
    electionTime: Timestamp({ t: 1640042040, i: 1 }),
    electionDate: ISODate("2021-12-20T23:14:00.000Z"),
    configVersion: 1,
    configTerm: 4,
    self: true,
    lastHeartbeatMessage: ''
  },
```

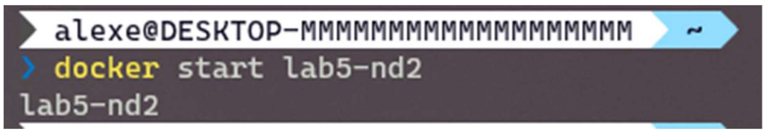


```
{
  _id: 1,
  name: 'lab5-nd2:27017',
  health: 0,
  state: 8,
  stateStr: '(not reachable/healthy)',
  uptime: 0,
  optime: { ts: Timestamp({ t: 0, i: 0 }), t: Long("-1") },
  optimeDurable: { ts: Timestamp({ t: 0, i: 0 }), t: Long("-1") },
  optimeDate: ISODate("1970-01-01T00:00:00.000Z"),
  optimeDurableDate: ISODate("1970-01-01T00:00:00.000Z"),
  lastAppliedWallTime: ISODate("1970-01-01T00:00:00.000Z"),
  lastDurableWallTime: ISODate("1970-01-01T00:00:00.000Z"),
  lastHeartbeat: ISODate("2021-12-20T23:14:20.928Z"),
  lastHeartbeatRecv: ISODate("1970-01-01T00:00:00.000Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: "Couldn't get a connection within the time limit",
  syncSourceHost: '',
  syncSourceId: -1,
  infoMessage: '',
  configVersion: -1,
  configTerm: -1
},
```



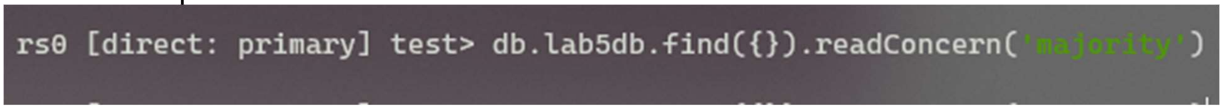
```
{
  _id: 2,
  name: 'lab5-nd3:27017',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 31,
  optime: { ts: Timestamp({ t: 1640041206, i: 1 }), t: Long("2") },
  optimeDurable: { ts: Timestamp({ t: 1640041206, i: 1 }), t: Long("2") },
  optimeDate: ISODate("2021-12-20T23:00:06.000Z"),
  optimeDurableDate: ISODate("2021-12-20T23:00:06.000Z"),
  lastAppliedWallTime: ISODate("2021-12-20T23:00:06.679Z"),
  lastDurableWallTime: ISODate("2021-12-20T23:00:06.679Z"),
  lastHeartbeat: ISODate("2021-12-20T23:14:24.935Z"),
  lastHeartbeatRecv: ISODate("2021-12-20T23:14:25.709Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '',
  syncSourceId: -1,
  infoMessage: '',
  configVersion: 1,
  configTerm: 3
}
```

- підключити (включити) попередню primary-ноду до кластеру і подивитись, що сталось зі значенням яке було на неї записано



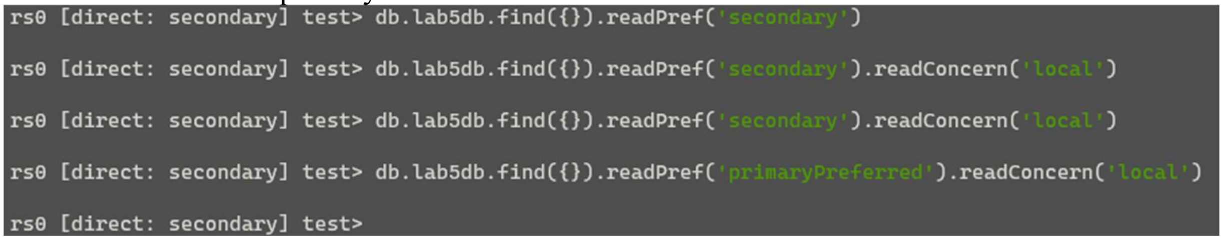
```
alexe@DESKTOP-MMMMMMMMMMMMMMMMMM  ~
❯ docker start lab5-nd2
lab5-nd2
```

No value on prime node:



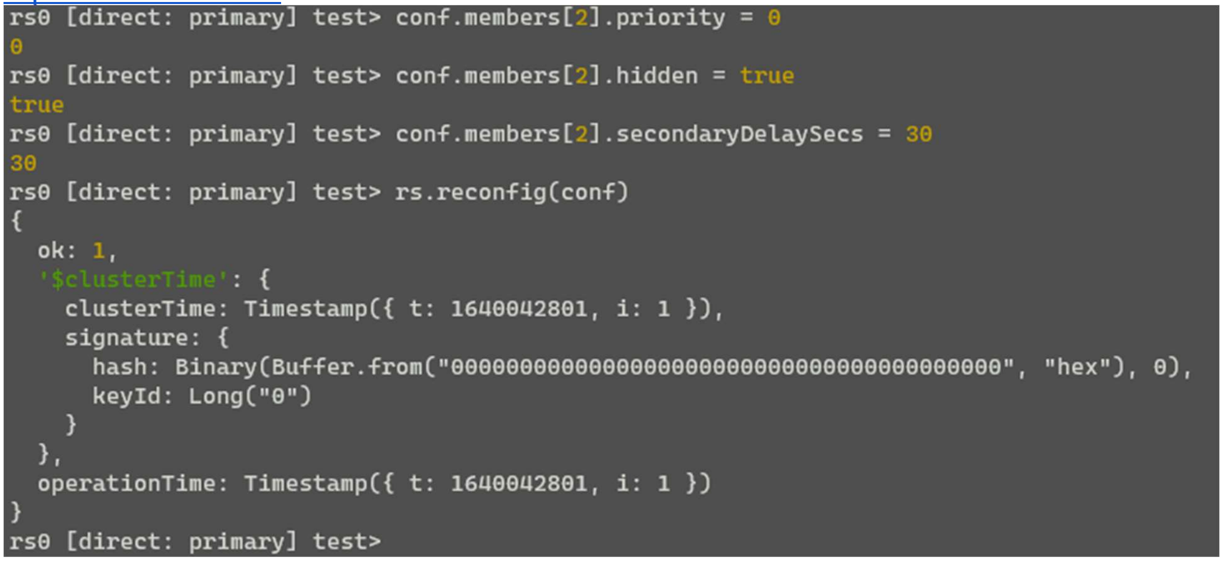
```
rs0 [direct: primary] test> db.lab5db.find({}).readConcern('majority')
```

No local value on old primary



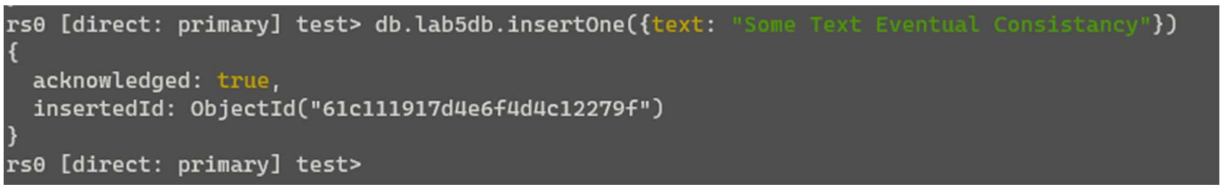
```
rs0 [direct: secondary] test> db.lab5db.find({}).readPref('secondary')

rs0 [direct: secondary] test> db.lab5db.find({}).readPref('secondary').readConcern('local')

rs0 [direct: secondary] test> db.lab5db.find({}).readPref('secondary').readConcern('local')

rs0 [direct: secondary] test> db.lab5db.find({}).readPref('primaryPreferred').readConcern('local')

rs0 [direct: secondary] test>
```

8.	Земулювати eventual consistency за допомогою установки затримки реплікації для репліки https://docs.mongodb.com/manual/tutorial/configure-a-delayed-replica-set-member/



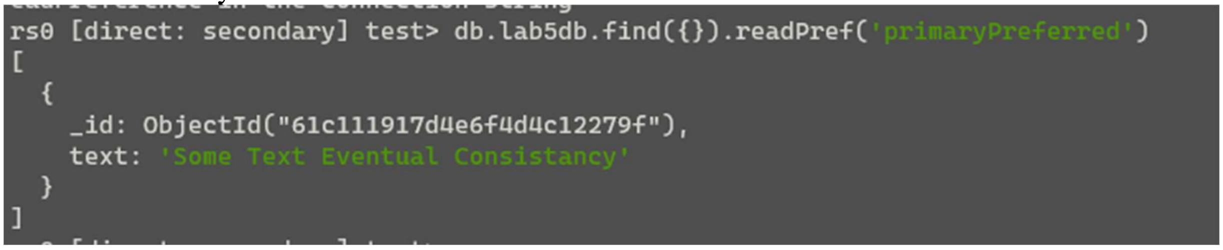
```
rs0 [direct: primary] test> conf.members[2].priority = 0
0
rs0 [direct: primary] test> conf.members[2].hidden = true
true
rs0 [direct: primary] test> conf.members[2].secondaryDelaySecs = 30
30
rs0 [direct: primary] test> rs.reconfig(conf)
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1640042801, i: 1 }),
    signature: {
      hash: Binary(Buffer.from("0000000000000000000000000000000000000000", "hex"), 0),
      keyId: Long("0")
    }
  },
  operationTime: Timestamp({ t: 1640042801, i: 1 })
}
rs0 [direct: primary] test>
```

Insert new value



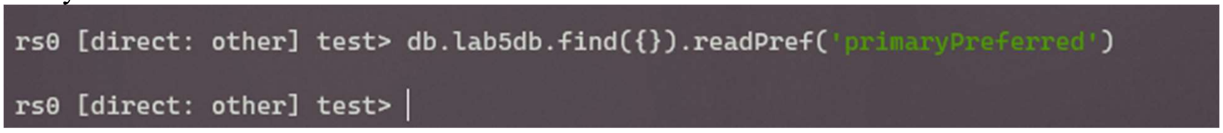
```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some Text Eventual Consistancy"})
{
  acknowledged: true,
  insertedId: ObjectId("61c111917d4e6f4d4c12279f")
}
rs0 [direct: primary] test>
```

Normal secondary:



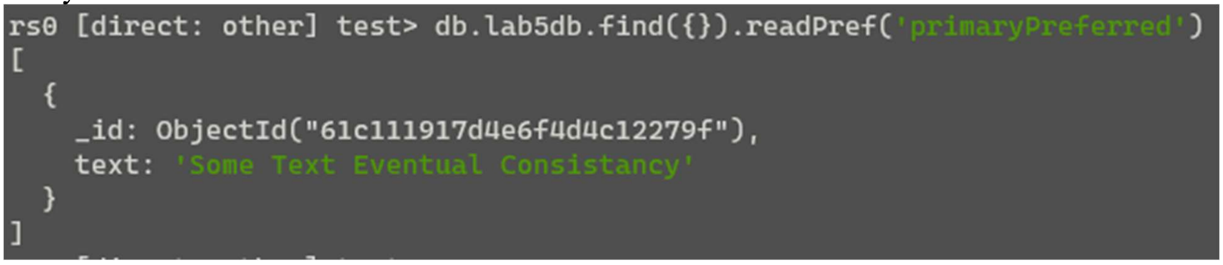
```
rs0 [direct: secondary] test> db.lab5db.find({}).readPref('primaryPreferred')
[
  {
    _id: ObjectId("61c111917d4e6f4d4c12279f"),
    text: 'Some Text Eventual Consistancy'
  }
]
```

Delayed node:



```
rs0 [direct: other] test> db.lab5db.find({}).readPref('primaryPreferred')

rs0 [direct: other] test> |
```

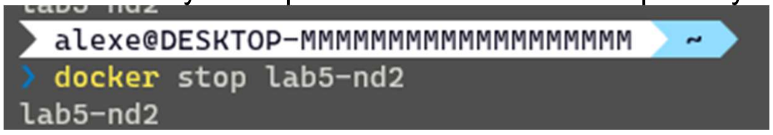
Delayed node after 30 sec:



```
rs0 [direct: other] test> db.lab5db.find({}).readPref('primaryPreferred')
[
  {
    _id: ObjectId("61c111917d4e6f4d4c12279f"),
    text: 'Some Text Eventual Consistancy'
  }
]
```

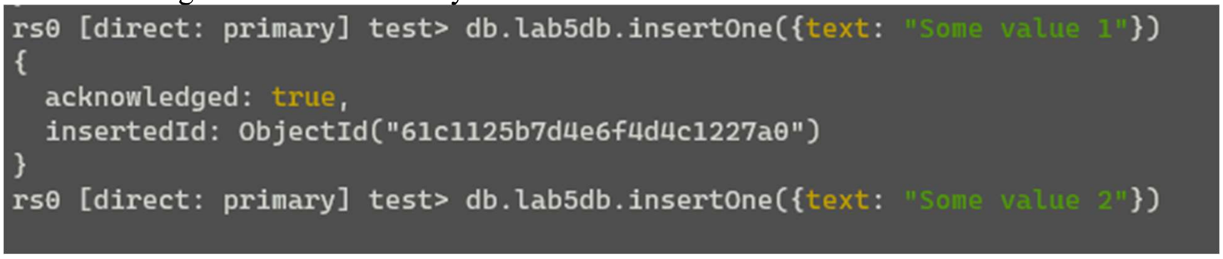
9.   Лишити *primary* та *secondary* для якої налаштована затримка реплікації. Записати декілька значень. Спробувати прочитати значення з `readConcern: {level: "linearizable"}`

Має бути затримка поки значення не реплікуються на більшість нод



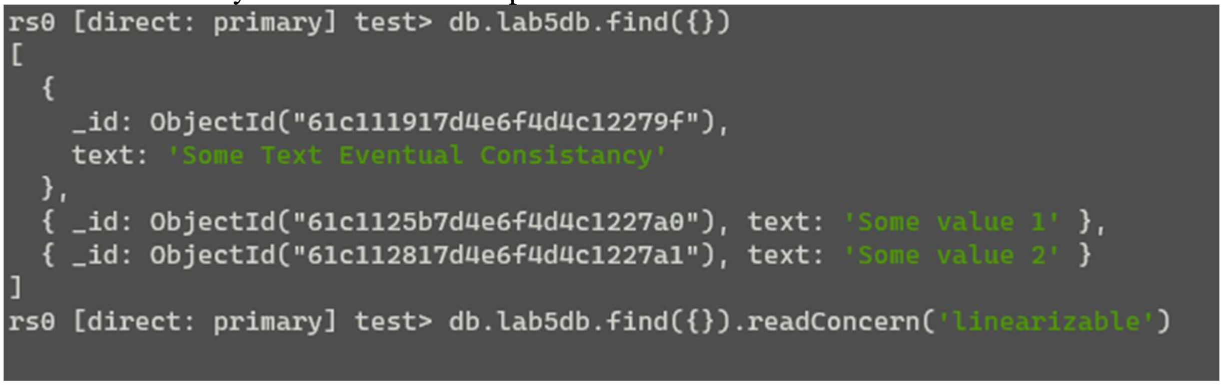
```
 alexe@DESKTOP-MMMMMMMMMMMMMMMMMMM  ~
❯ docker stop lab5-nd2
lab5-nd2
```

We are waiting 30 sec cause of delay.



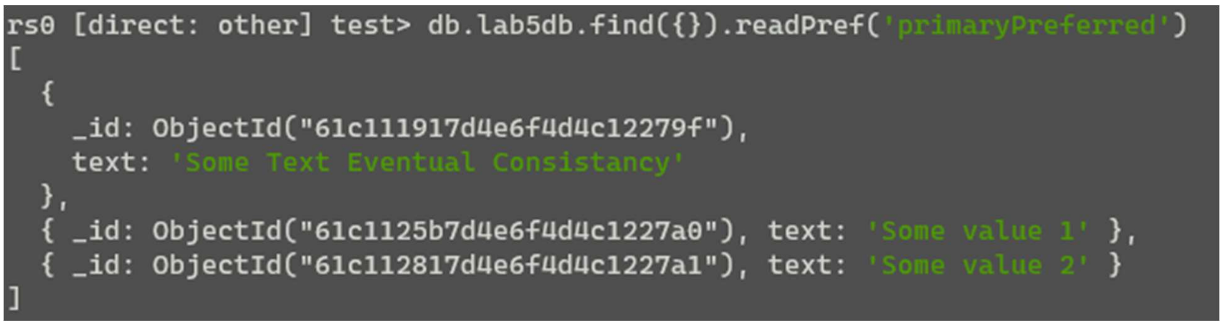
```
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some value 1"})
{
  acknowledged: true,
  insertedId: ObjectId("61c1125b7d4e6f4d4c1227a0")
}
rs0 [direct: primary] test> db.lab5db.insertOne({text: "Some value 2"})
```

Also we have delay with ‘linearizable’ option



```
rs0 [direct: primary] test> db.lab5db.find({})
[
  {
    _id: ObjectId("61c111917d4e6f4d4c12279f"),
    text: 'Some Text Eventual Consistancy'
  },
  { _id: ObjectId("61c1125b7d4e6f4d4c1227a0"), text: 'Some value 1' },
  { _id: ObjectId("61c112817d4e6f4d4c1227a1"), text: 'Some value 2' }
]
rs0 [direct: primary] test> db.lab5db.find({}).readConcern('linearizable')
```

But non-linearizable reading on secondary node is fast. Maybe because it uses ‘local’ option for those purposes



```
rs0 [direct: other] test> db.lab5db.find({}).readPref('primaryPreferred')
[
  {
    _id: ObjectId("61c111917d4e6f4d4c12279f"),
    text: 'Some Text Eventual Consistancy'
  },
  { _id: ObjectId("61c1125b7d4e6f4d4c1227a0"), text: 'Some value 1' },
  { _id: ObjectId("61c112817d4e6f4d4c1227a1"), text: 'Some value 2' }
]
```

Опис додаткових команд Replication Reference
- http://docs.mongodb.org/manual/reference/replication/

Вимогу до оформлення протоколу:
Завдання здається особисто без протоколу, або надсилається протокол який має містити:
- команди та результати їх виконання