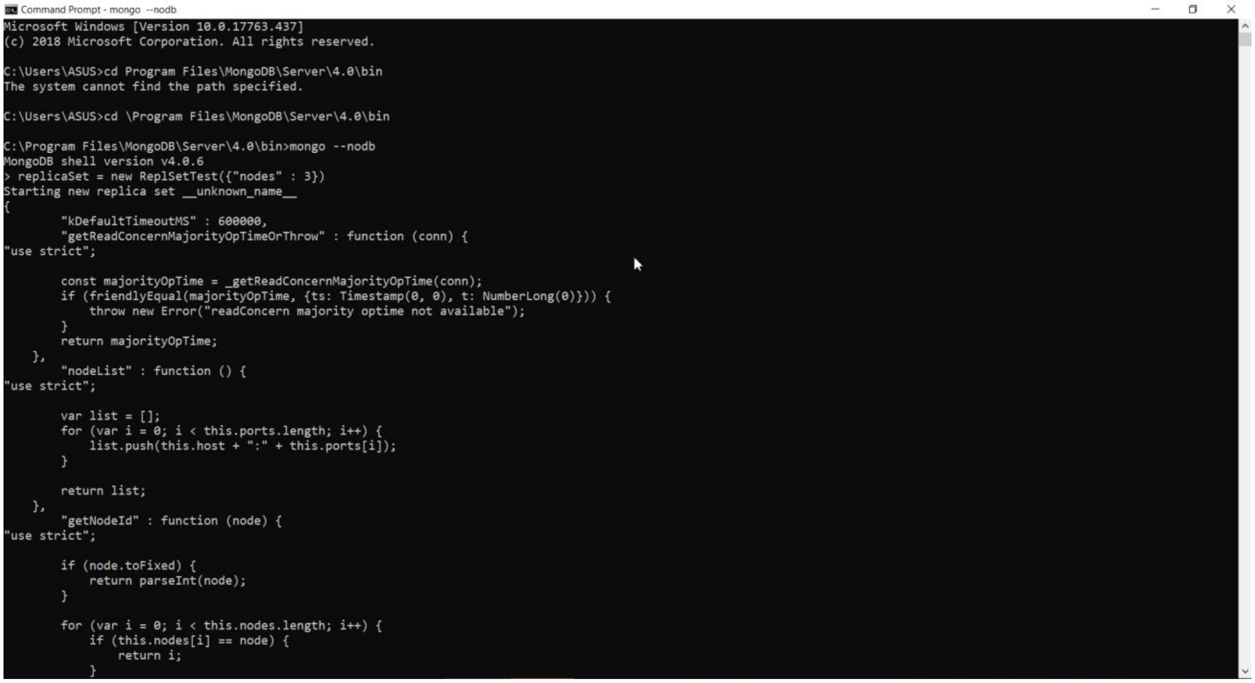
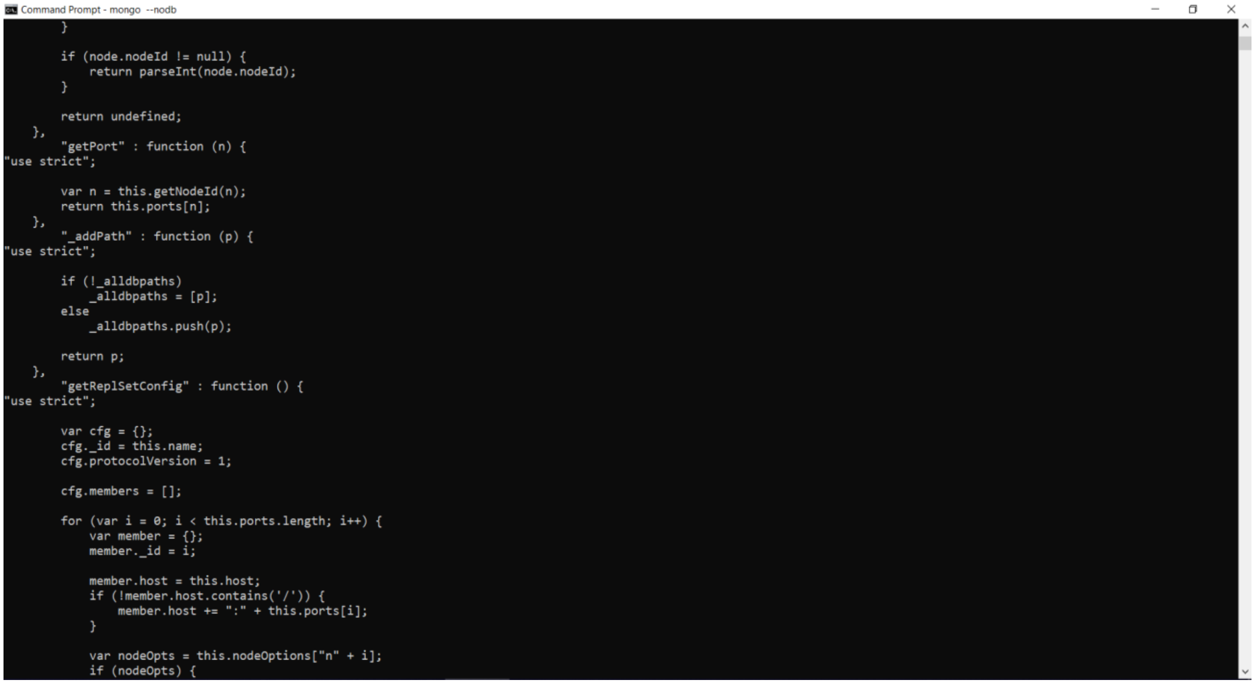
NANA : TONY BASKORO
NIM : 17050623016
KELAS : MI 2017

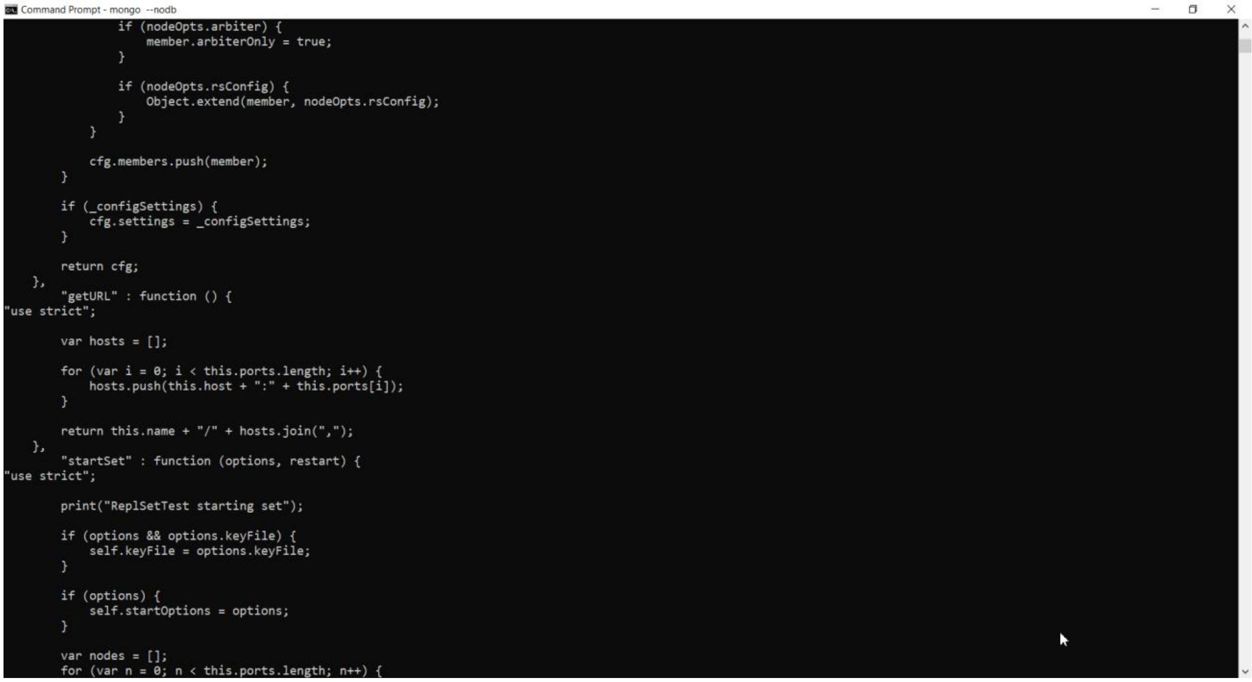
REPLICATION MONGODB



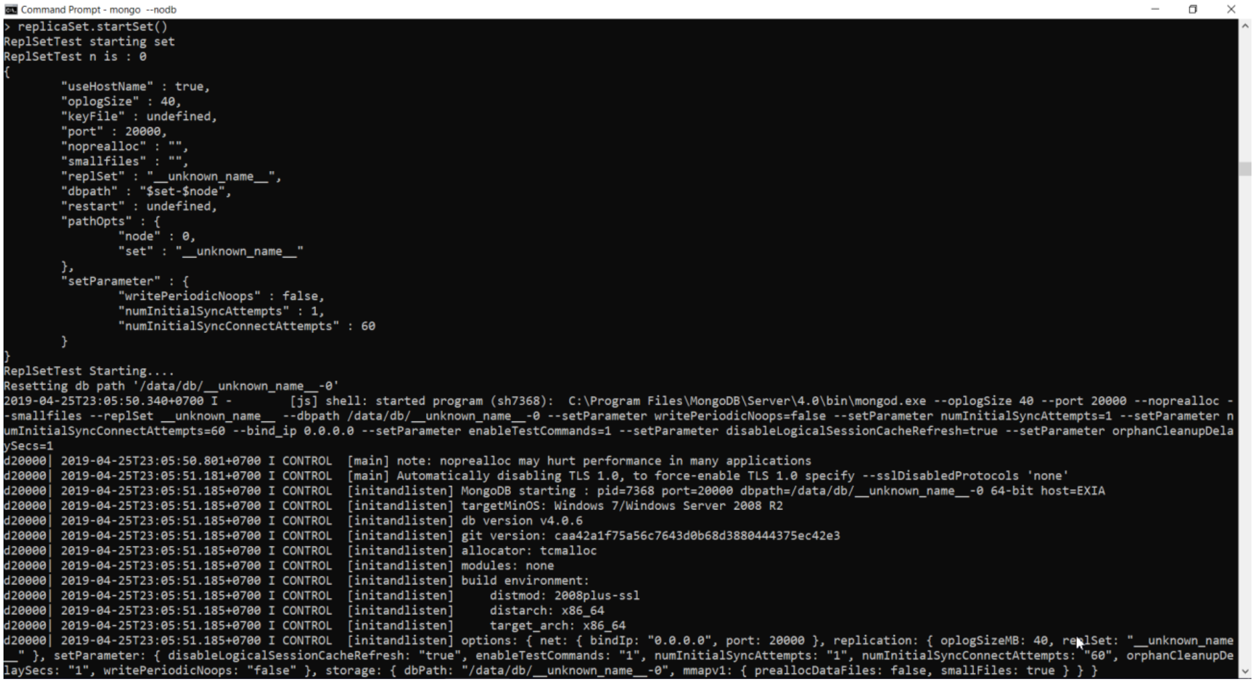
```
Microsoft Windows [Version 10.0.17763.437]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>cd Program Files\MongoDB\Server\4.0\bin
The system cannot find the path specified.

C:\Users\ASUS>cd \Program Files\MongoDB\Server\4.0\bin

C:\Program Files\MongoDB\Server\4.0\bin>mongo --nodb
MongoDB shell version v4.0.6
> replicaSet = new ReplSetTest({"nodes" : 3})
Starting new replica set __unknown_name__
{
        "kDefaultTimeoutMS" : 600000,
        "getReadConcernMajorityOpTimeOrThrow" : function (conn) {
"use strict";

        const majorityOpTime = _getReadConcernMajorityOpTime(conn);
        if (friendlyEqual(majorityOpTime, {ts: Timestamp(0, 0), t: NumberLong(0)})) {
            throw new Error("readConcern majority optime not available");
        }
        return majorityOpTime;
    },
        "nodeList" : function () {
"use strict";

        var list = [];
        for (var i = 0; i < this.ports.length; i++) {
            list.push(this.host + ":" + this.ports[i]);
        }

        return list;
    },
        "getNodeId" : function (node) {
"use strict";

        if (node.toFixed) {
            return parseInt(node);
        }

        for (var i = 0; i < this.nodes.length; i++) {
            if (this.nodes[i] == node) {
                return i;
            }
```



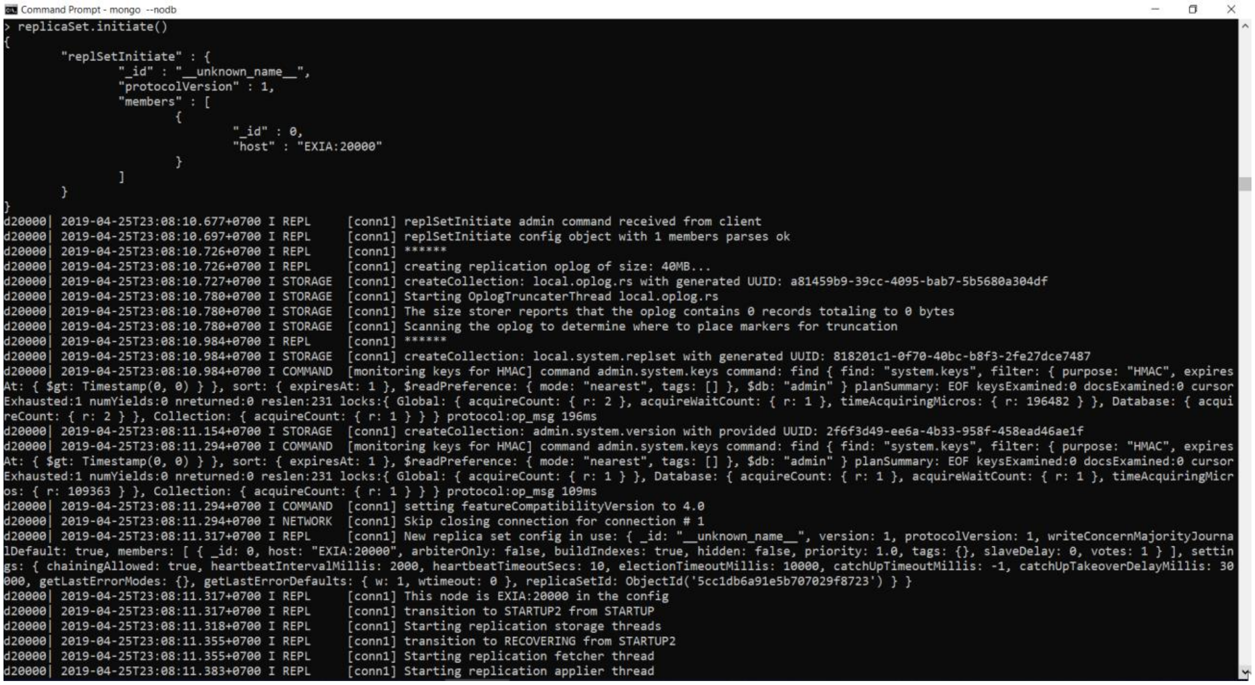
```
        }

        if (node.nodeId != null) {
            return parseInt(node.nodeId);
        }

        return undefined;
    },
        "getPort" : function (n) {
"use strict";

        var n = this.getNodeId(n);
        return this.ports[n];
    },
        "_addPath" : function (p) {
"use strict";

        if (!_alldbpaths)
            _alldbpaths = [p];
        else
            _alldbpaths.push(p);

        return p;
    },
        "getReplSetConfig" : function () {
"use strict";

        var cfg = {};
        cfg._id = this.name;
        cfg.protocolVersion = 1;

        cfg.members = [];

        for (var i = 0; i < this.ports.length; i++) {
            var member = {};
            member._id = i;

            member.host = this.host;
            if (!member.host.contains('/')) {
                member.host += ":" + this.ports[i];
            }

            var nodeOpts = this.nodeOptions["n" + i];
            if (nodeOpts) {
```



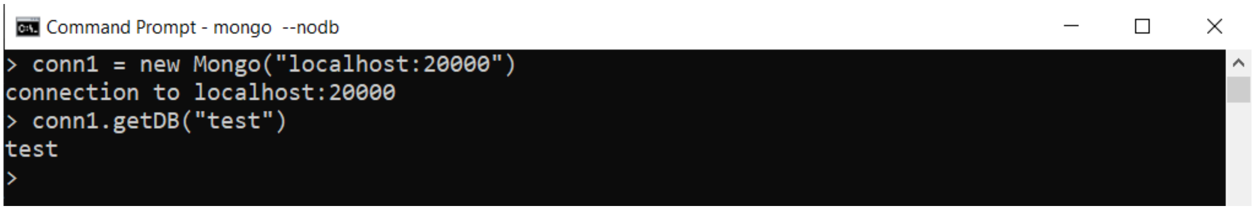
```
                if (nodeOpts.arbiter) {
                    member.arbiterOnly = true;
                }

                if (nodeOpts.rsConfig) {
                    Object.extend(member, nodeOpts.rsConfig);
                }
            }

            cfg.members.push(member);
        }

        if (_configSettings) {
            cfg.settings = _configSettings;
        }

        return cfg;
    },
        "getURL" : function () {
"use strict";

        var hosts = [];

        for (var i = 0; i < this.ports.length; i++) {
            hosts.push(this.host + ":" + this.ports[i]);
        }

        return this.name + "/" + hosts.join(",");
    },
        "startSet" : function (options, restart) {
"use strict";

        print("ReplSetTest starting set");

        if (options && options.keyFile) {
            self.keyFile = options.keyFile;
        }

        if (options) {
            self.startOptions = options;
        }

        var nodes = [];
        for (var n = 0; n < this.ports.length; n++) {
```



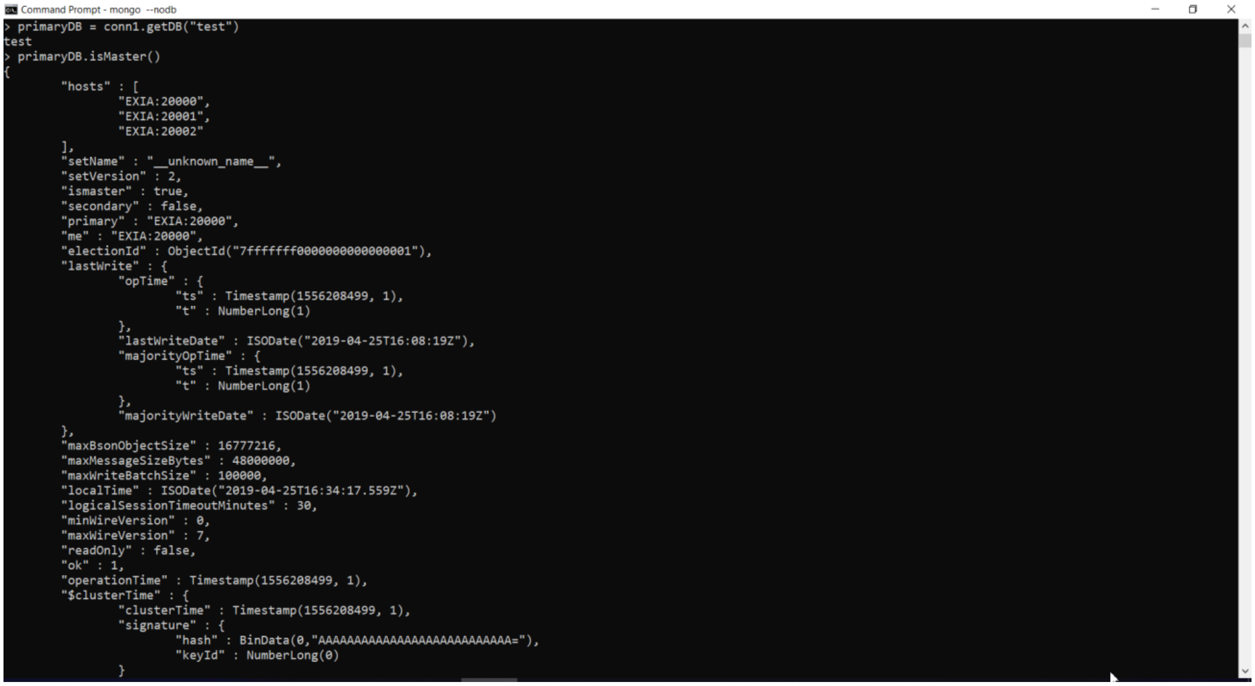
```
> replicaSet.startSet()
ReplSetTest starting set
ReplSetTest n is : 0
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20000,
        "noprealloc" : "",
        "smallfiles" : "",
        "replSet" : "__unknown_name__",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 0,
                "set" : "__unknown_name__"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncAttempts" : 1,
                "numInitialSyncConnectAttempts" : 60
        }
}
ReplSetTest Starting....
Resetting db path '/data/db/__unknown_name__-0'
2019-04-25T23:05:50.340+0700 I -        [js] shell: started program (sh7368):  C:\Program Files\MongoDB\Server\4.0\bin\mongod.exe --oplogSize 40 --port 20000 --noprealloc -
-smallfiles --replSet __unknown_name__ --dbpath /data/db/__unknown_name__-0 --setParameter writePeriodicNoops=false --setParameter numInitialSyncAttempts=1 --setParameter n
umInitialSyncConnectAttempts=60 --bind_ip 0.0.0.0 --setParameter enableTestCommands=1 --setParameter disableLogicalSessionCacheRefresh=true --setParameter orphanCleanupDela
ySecs=1
d20000| 2019-04-25T23:05:50.801+0700 I CONTROL  [main] note: noprealloc may hurt performance in many applications
d20000| 2019-04-25T23:05:51.181+0700 I CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] MongoDB starting : pid=7368 port=20000 dbpath=/data/db/__unknown_name__-0 64-bit host=EXIA
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] db version v4.0.6
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] git version: caa42a1f75a56c7643d0b68d3880444375ec42e3
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] allocator: tcmalloc
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] modules: none
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] build environment:
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten]     distmod: 2008plus-ssl
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten]     distarch: x86_64
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten]     target_arch: x86_64
d20000| 2019-04-25T23:05:51.185+0700 I CONTROL  [initandlisten] options: { net: { bindIp: "0.0.0.0", port: 20000 }, replication: { oplogSizeMB: 40, replSet: "__unknown_name
__" }, setParameter: { disableLogicalSessionCacheRefresh: "true", enableTestCommands: "1", numInitialSyncAttempts: "1", numInitialSyncConnectAttempts: "60", orphanCleanupDe
laySecs: "1", writePeriodicNoops: "false" }, storage: { dbPath: "/data/db/__unknown_name__-0", mmapv1: { preallocDataFiles: false, smallFiles: true } } }
```



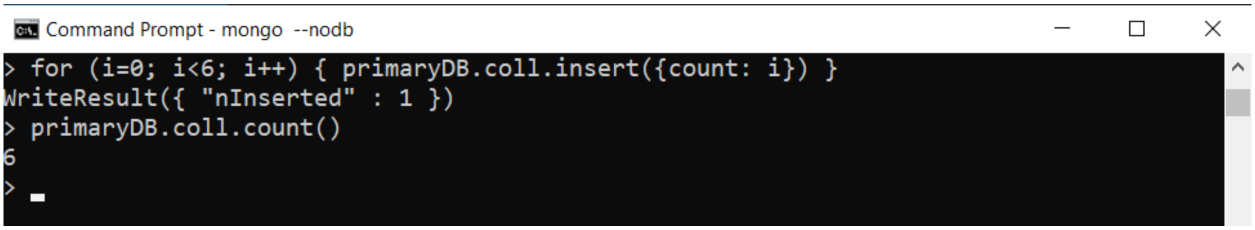
```
> replicaSet.initiate()
{
        "replSetInitiate" : {
                "_id" : "__unknown_name__",
                "protocolVersion" : 1,
                "members" : [
                        {
                                "_id" : 0,
                                "host" : "EXIA:20000"
                        }
                ]
        }
}
d20000| 2019-04-25T23:08:10.677+0700 I REPL     [conn1] replSetInitiate admin command received from client
d20000| 2019-04-25T23:08:10.697+0700 I REPL     [conn1] replSetInitiate config object with 1 members parses ok
d20000| 2019-04-25T23:08:10.726+0700 I REPL     [conn1] ******
d20000| 2019-04-25T23:08:10.726+0700 I REPL     [conn1] creating replication oplog of size: 40MB...
d20000| 2019-04-25T23:08:10.727+0700 I STORAGE  [conn1] createCollection: local.oplog.rs with generated UUID: a81459b9-39cc-4095-bab7-5b5680a304df
d20000| 2019-04-25T23:08:10.780+0700 I STORAGE  [conn1] Starting OplogTruncaterThread local.oplog.rs
d20000| 2019-04-25T23:08:10.780+0700 I STORAGE  [conn1] The size storer reports that the oplog contains 0 records totaling to 0 bytes
d20000| 2019-04-25T23:08:10.780+0700 I STORAGE  [conn1] Scanning the oplog to determine where to place markers for truncation
d20000| 2019-04-25T23:08:10.984+0700 I REPL     [conn1] ******
d20000| 2019-04-25T23:08:10.984+0700 I STORAGE  [conn1] createCollection: local.system.replset with generated UUID: 818201c1-0f70-40bc-b8f3-2fe27dce7487
d20000| 2019-04-25T23:08:10.984+0700 I COMMAND  [monitoring keys for HMAC] command admin.system.keys command: find { find: "system.keys", filter: { purpose: "HMAC", expires
At: { $gt: Timestamp(0, 0) } }, sort: { expiresAt: 1 }, $readPreference: { mode: "nearest", tags: [] }, $db: "admin" } planSummary: EOF keysExamined:0 docsExamined:0 cursor
Exhausted:1 numYields:0 nreturned:0 reslen:231 locks:{ Global: { acquireCount: { r: 2 }, acquireWaitCount: { r: 1 }, timeAcquiringMicros: { r: 196482 } }, Database: { acqui
reCount: { r: 2 } }, Collection: { acquireCount: { r: 1 } } } protocol:op_msg 196ms
d20000| 2019-04-25T23:08:11.154+0700 I STORAGE  [conn1] createCollection: admin.system.version with provided UUID: 2f6f3d49-ee6a-4b33-958f-458ead46ae1f
d20000| 2019-04-25T23:08:11.294+0700 I COMMAND  [monitoring keys for HMAC] command admin.system.keys command: find { find: "system.keys", filter: { purpose: "HMAC", expires
At: { $gt: Timestamp(0, 0) } }, sort: { expiresAt: 1 }, $readPreference: { mode: "nearest", tags: [] }, $db: "admin" } planSummary: EOF keysExamined:0 docsExamined:0 cursor
Exhausted:1 numYields:0 nreturned:0 reslen:231 locks:{ Global: { acquireCount: { r: 1 } }, Database: { acquireCount: { r: 1 }, acquireWaitCount: { r: 1 }, timeAcquiringMicr
os: { r: 109363 } }, Collection: { acquireCount: { r: 1 } } } protocol:op_msg 109ms
d20000| 2019-04-25T23:08:11.294+0700 I COMMAND  [conn1] setting featureCompatibilityVersion to 4.0
d20000| 2019-04-25T23:08:11.294+0700 I NETWORK  [conn1] Skip closing connection for connection # 1
d20000| 2019-04-25T23:08:11.317+0700 I REPL     [conn1] New replica set config in use: { _id: "__unknown_name__", version: 1, protocolVersion: 1, writeConcernMajorityJourna
lDefault: true, members: [ { _id: 0, host: "EXIA:20000", arbiterOnly: false, buildIndexes: true, hidden: false, priority: 1.0, tags: {}, slaveDelay: 0, votes: 1 } ], settin
gs: { chainingAllowed: true, heartbeatIntervalMillis: 2000, heartbeatTimeoutSecs: 10, electionTimeoutMillis: 10000, catchUpTimeoutMillis: -1, catchUpTakeoverDelayMillis: 30
000, getLastErrorModes: {}, getLastErrorDefaults: { w: 1, wtimeout: 0 }, replicaSetId: ObjectId('5cc1db6a91e5b707029f8723') } }
d20000| 2019-04-25T23:08:11.317+0700 I REPL     [conn1] This node is EXIA:20000 in the config
d20000| 2019-04-25T23:08:11.317+0700 I REPL     [conn1] transition to STARTUP2 from STARTUP
d20000| 2019-04-25T23:08:11.318+0700 I REPL     [conn1] Starting replication storage threads
d20000| 2019-04-25T23:08:11.355+0700 I REPL     [conn1] transition to RECOVERING from STARTUP2
d20000| 2019-04-25T23:08:11.355+0700 I REPL     [conn1] Starting replication fetcher thread
d20000| 2019-04-25T23:08:11.383+0700 I REPL     [conn1] Starting replication applier thread
```



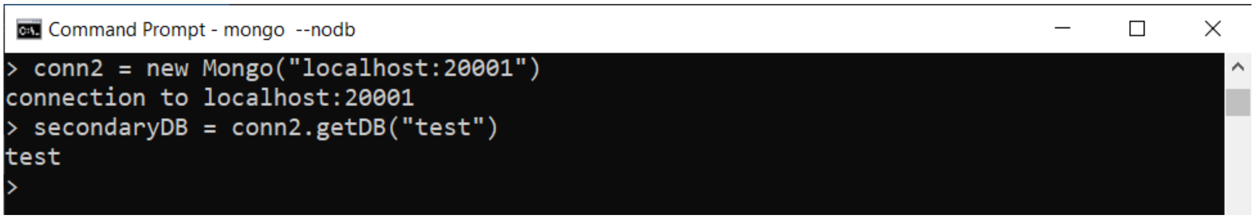
```
> conn1 = new Mongo("localhost:20000")
connection to localhost:20000
> conn1.getDB("test")
test
>
```



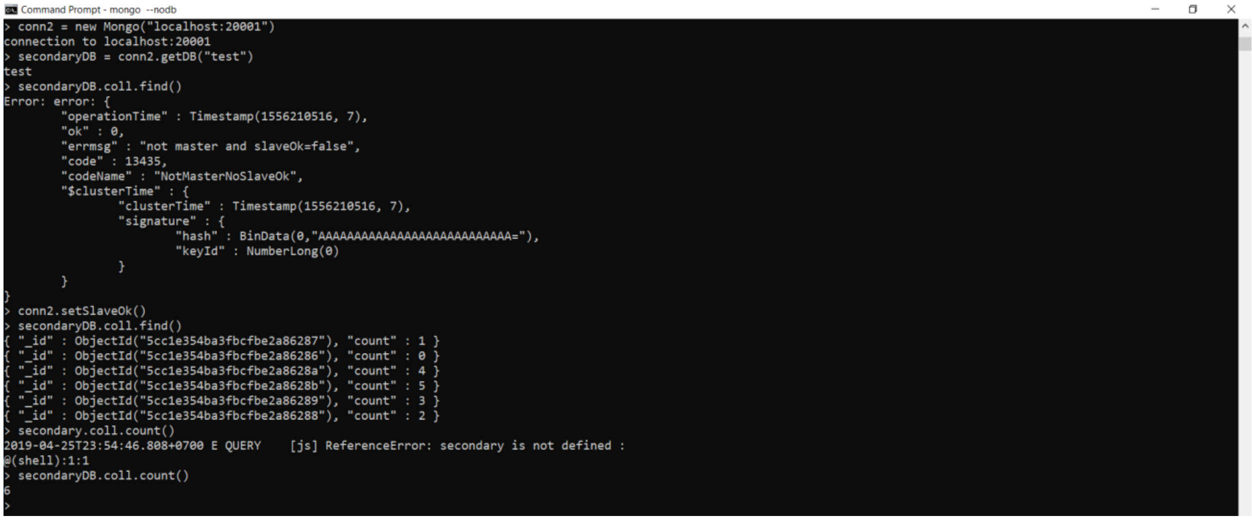
```
> primaryDB = conn1.getDB("test")
test
> primaryDB.isMaster()
{
        "hosts" : [
                "EXIA:20000",
                "EXIA:20001",
                "EXIA:20002"
        ],
        "setName" : "__unknown_name__",
        "setVersion" : 2,
        "ismaster" : true,
        "secondary" : false,
        "primary" : "EXIA:20000",
        "me" : "EXIA:20000",
        "electionId" : ObjectId("7fffffff0000000000000001"),
        "lastWrite" : {
                "opTime" : {
                        "ts" : Timestamp(1556208499, 1),
                        "t" : NumberLong(1)
                },
                "lastWriteDate" : ISODate("2019-04-25T16:08:19Z"),
                "majorityOpTime" : {
                        "ts" : Timestamp(1556208499, 1),
                        "t" : NumberLong(1)
                },
                "majorityWriteDate" : ISODate("2019-04-25T16:08:19Z")
        },
        "maxBsonObjectSize" : 16777216,
        "maxMessageSizeBytes" : 48000000,
        "maxWriteBatchSize" : 100000,
        "localTime" : ISODate("2019-04-25T16:34:17.559Z"),
        "logicalSessionTimeoutMinutes" : 30,
        "minWireVersion" : 0,
        "maxWireVersion" : 7,
        "readOnly" : false,
        "ok" : 1,
        "operationTime" : Timestamp(1556208499, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1556208499, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
```



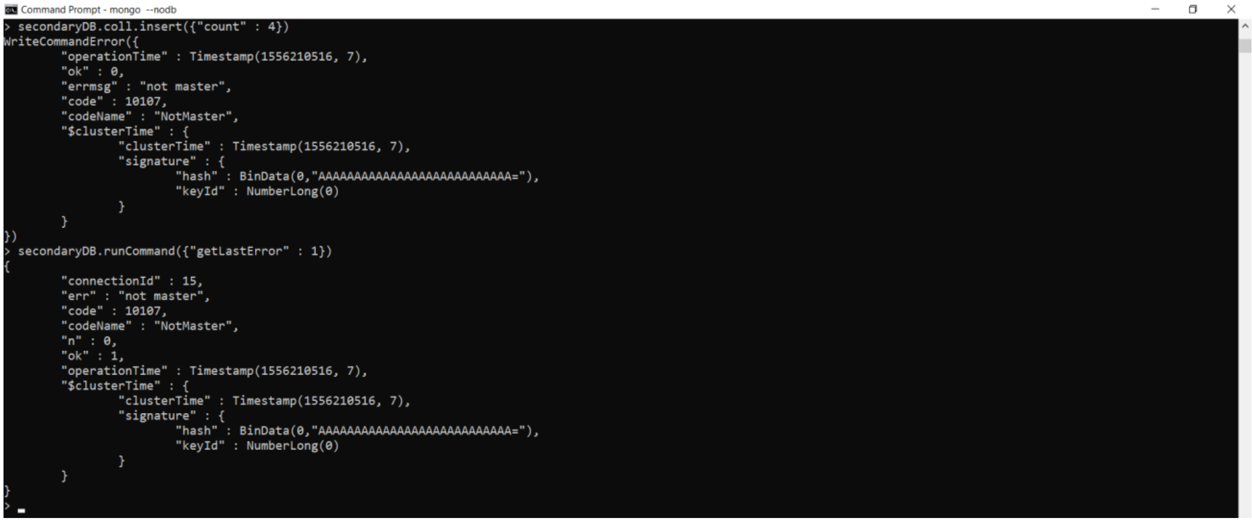
```
> for (i=0; i<6; i++) { primaryDB.coll.insert({count: i}) }
WriteResult({ "nInserted" : 1 })
> primaryDB.coll.count()
6
> _
```



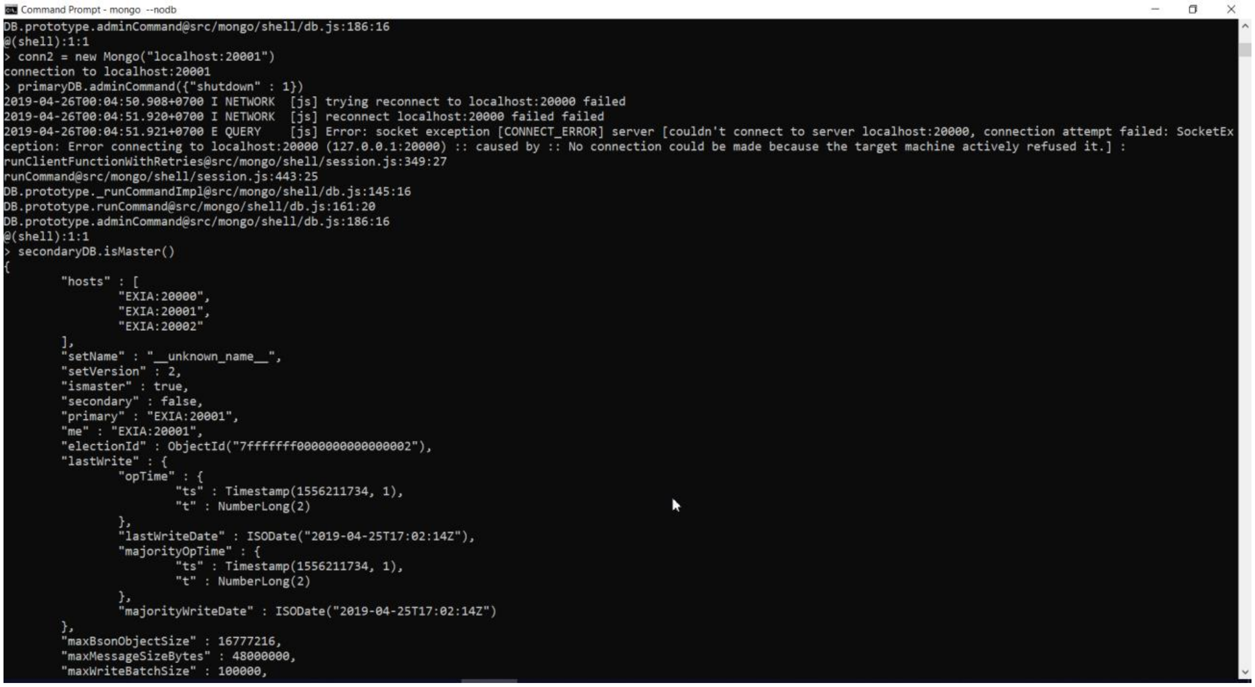
```
> conn2 = new Mongo("localhost:20001")
connection to localhost:20001
> secondaryDB = conn2.getDB("test")
test
>
```



```
> conn2 = new Mongo("localhost:20001")
connection to localhost:20001
> secondaryDB = conn2.getDB("test")
test
> secondaryDB.coll.find()
Error: error: {
        "operationTime" : Timestamp(1556210516, 7),
        "ok" : 0,
        "errmsg" : "not master and slaveOk=false",
        "code" : 13435,
        "codeName" : "NotMasterNoSlaveOk",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1556210516, 7),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
> conn2.setSlaveOk()
> secondaryDB.coll.find()
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a86287"), "count" : 1 }
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a86286"), "count" : 0 }
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a8628a"), "count" : 4 }
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a8628b"), "count" : 5 }
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a86289"), "count" : 3 }
{ "_id" : ObjectId("5cc1e354ba3fbcfbe2a86288"), "count" : 2 }
> secondary.coll.count()
2019-04-25T23:54:46.808+0700 E QUERY    [js] ReferenceError: secondary is not defined :
@(shell):1:1
> secondaryDB.coll.count()
6
>
```



```
> secondaryDB.coll.insert({"count" : 4})
WriteCommandError({
        "operationTime" : Timestamp(1556210516, 7),
        "ok" : 0,
        "errmsg" : "not master",
        "code" : 10107,
        "codeName" : "NotMaster",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1556210516, 7),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
})
> secondaryDB.runCommand({"getLastError" : 1})
{
        "connectionId" : 15,
        "err" : "not master",
        "code" : 10107,
        "codeName" : "NotMaster",
        "n" : 0,
        "ok" : 1,
        "operationTime" : Timestamp(1556210516, 7),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1556210516, 7),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
> _
```



```
DB.prototype.adminCommand@src/mongo/shell/db.js:186:16
@(shell):1:1
> conn2 = new Mongo("localhost:20001")
connection to localhost:20001
> primaryDB.adminCommand({"shutdown" : 1})
2019-04-26T00:04:50.908+0700 I NETWORK  [js] trying reconnect to localhost:20000 failed
2019-04-26T00:04:51.920+0700 I NETWORK  [js] reconnect localhost:20000 failed failed
2019-04-26T00:04:51.921+0700 E QUERY    [js] Error: socket exception [CONNECT_ERROR] server [couldn't connect to server localhost:20000, connection attempt failed: SocketEx
ception: Error connecting to localhost:20000 (127.0.0.1:20000) :: caused by :: No connection could be made because the target machine actively refused it.] :
runClientFunctionWithRetries@src/mongo/shell/session.js:349:27
runCommand@src/mongo/shell/session.js:443:25
DB.prototype._runCommandImpl@src/mongo/shell/db.js:145:16
DB.prototype.runCommand@src/mongo/shell/db.js:161:20
DB.prototype.adminCommand@src/mongo/shell/db.js:186:16
@(shell):1:1
> secondaryDB.isMaster()
{
        "hosts" : [
                "EXIA:20000",
                "EXIA:20001",
                "EXIA:20002"
        ],
        "setName" : "__unknown_name__",
        "setVersion" : 2,
        "ismaster" : true,
        "secondary" : false,
        "primary" : "EXIA:20001",
        "me" : "EXIA:20001",
        "electionId" : ObjectId("7fffffff0000000000000002"),
        "lastWrite" : {
                "opTime" : {
                        "ts" : Timestamp(1556211734, 1),
                        "t" : NumberLong(2)
                },
                "lastWriteDate" : ISODate("2019-04-25T17:02:14Z"),
                "majorityOpTime" : {
                        "ts" : Timestamp(1556211734, 1),
                        "t" : NumberLong(2)
                },
                "majorityWriteDate" : ISODate("2019-04-25T17:02:14Z")
        },
        "maxBsonObjectSize" : 16777216,
        "maxMessageSizeBytes" : 48000000,
        "maxWriteBatchSize" : 100000,
```