Authors: Daria Skarbek, Adam Pocheć, Igor Szemela

1. Description of a situation in which a MongoDB store may be used

Collections: Students, Teachers, Subjects, Houses

Our MongoDB store may be used for storing the data about students, teachers, subjects and houses that belong to given university. It is useful for all the administrative work that uni has to perform and is crucial for their smooth daily operation.

It may be helpful for the university council, it could execute queries that return info about students address and more personal info. It can track students' performance, assigned teachers and the house they belong to on the campus.

It may track what students live in a given house and how they perform as one entity. Surely it will help by assigning students to a given house, to prevent one from being overloaded.

It may help while assigning work to teachers.

2. List of operations adding at least 20 documents to the database

---- STUDENTS ---------------------------------------

```
db.students.insertMany([
{
        name : "Harry",
        surname : "Potter",
        date_of_birth : "2000-07-31",
        gender : "male",
        address : {
                country : "England",
                city : "London",
                street : "Baker Street",
                number : "201"
                },
        house : "Gryffindor",
        year : 2,
        classes : [
                {name : "transfiguration", grades : [3, 5, 3]},
                {name : "potions", grades : [2, 3, 2.5] },
                {name : "care of magical creatures", grades : [5, 5, 4]}
        ]
},
{
        name : "Ginny",
        surname : "Weasley",
        date_of_birth : "2001-12-11",
        gender : "female",
        address : {
            country : "England",
            city : "Manchester",
            street : "Long Street",
            number : "111"
            },
        house : "Gryffindor",
        year : 1,
        classes : [
            {name : "transfiguration", grades : [4, 4]},
            {name : "charms", grades : [4, 3] },
            {name : "defence against the dark arts", grades : [5]}
        ]
    },
    {
        name : "Cedric",
        surname : "Diggory",
        date_of_birth : "1998-05-05",
        gender : "male",
        address : {
            country : "Sweden",
            city : "Uppsala",
            street : "Rabyvagen",
            number : "9"
            },
        house : "Hufflepuff",
        year : 4,
        classes : [
            {name : "advanced transfiguration", grades : [4, 3]}
        ]
    },
    {
        name : "Luna",
        surname : "Lovegood",
        date_of_birth : "1995-05-17",
        gender : "female",
        address : {
            country : "Island",
            city : "Reykiavik",
            street : "Basendi",
            number : "3"
            },
        house : "Ravenclaw",
        year : 6,
        classes : [
            {name : "defence against the dark arts", grades : [4.5]},
            {name : "herbology", grades : [4, 3, 5] },
			{name : "advanced transfiguration", grades : [5]},
			{name : "advanced charms", grades : [3, 3.5]},
		]
},
{
		name : "Vincent",
		surname : "Crabbe",
		date_of_birth : "1994-08-23",
		gender : "male",
		address : {
			country : "England",
			city : "Scarborough",
			street : "Avenue Rd",
			number : "89"
			},
		house : "Slytherin",
		year : 7,
		classes : [
			{name : "defence against the dark arts", grades : [4, 3]},
			{name : "potions", grades : [4, 2] }
		]
}
])
```

---- TEACHERS --

```
db.teachers.insertMany([
{
		name : "Minerva",
		surname : "McGonagall",
		date_of_birth : "1960-10-10",
		gender : "female",
		subjects : [
			{name : "transfiguration"},
			{name : "advanced transfiguration"}
		]
},
{
		name : "Severus",
		surname : "Snape",
		date_of_birth : "19671-09-23",
		gender : "male",
		subjects : [
			{name : "potions"},
			{name : "defence against the dark arts"}
		]
},
{
        name : "Pomona",
        surname : "Sprout",
        date_of_birth : "1990-02-12",
        gender : "female",
        subjects : [
                {name : "herbology"}
        ]
},
{
        name : "Filius",
        surname : "Flitwick",
        date_of_birth : "1962-12-30",
        gender : "male",
        subjects : [
                {name : "charms"},
                {name : "advanced charms"}]
},
{
        name : "Cuthbert",
        surname : "Binns",
        date_of_birth : "1880-02-11",
        gender : "male",
        subjects : [
                {name : "history of magic"}]
},
{
        name : "Rubeus",
        surname : "Hagrid",
        date_of_birth : "1978-06-21",
        gender : "male",
        subjects : [
                {name : "care of magical creatures"}]
}
])
```

---- CLASSES ---------------------------------------

```
db.classes.insertMany([
{
        name : "transfiguration",
        teacher : { name : "Minerva", surname : "McGonagall" },
        year : [1, 2, 3],
        scheduled : [{
                day : "Tuesday",
                hour : "11:00",
                place : "room 11" }]
},
{
		name : "advanced transfiguration",
		teacher : { name : "Minerva", surname : "McGonagall" },
		year : [4, 5, 6, 7],
		scheduled : [
			{
			day : "Wednesday",
			hour : "09:00",
			place : "room 11" },
			{
			day : "Friday",
			hour : "13:00",
			place : "room 11" },
			]
	},
	{
		name : "potions",
		teacher : { name : "Severus", surname : "Snape" },
		year : [1, 2, 3, 4, 5],
		scheduled : [{
			day : "Monday",
			hour : "16:00",
			place : "room 9" }]
	},
	{
		name : "herbology",
		teacher : { name : "Pomona", surname : "Sprout" },
		year : [2, 3, 4, 5],
		scheduled : [{
			day : "Monday",
			hour : "10:00",
			place : "greenhouse 1" },
			{
			day : "Thursday",
			hour : "13:00",
			place : "greenhouse 2" }]
	},
	{
		name : "defence against the dark arts",
		teacher : { name : "Severus", surname : "Snape" },
		year : [1, 2, 3, 4, 5, 6, 7],
		scheduled : [{
			day : "Friday",
			hour : "09:00",
			place : "room 3" },
			{
			day : "Thursday",
			hour : "09:00",
			place : "room 3" }]
```

```
},
{
	name : "charms",
	teacher : { name : "Filius", surname : "Flitwick" },
	year : [1, 2, 3, 4],
	scheduled : [{
		day : "Wednesday",
		hour : "14:00",
		place : "room 10" }]
},
{
	name : "advanced charms",
	teacher : { name : "Filius", surname : "Flitwick" },
	year : [5, 6, 7],
	scheduled : [{
		day : "Wednesday",
		hour : "17:00",
		place : "room 10" }]
},
{
	name : "history of magic",
	teacher : { name : "Cuthbert", surname : "Binns" },
	year : [1, 2, 3],
	scheduled : [{
		day : "Tuesday",
		hour : "14:00",
		place : "room 7" }]
},
{
	name : "care of magical creatures",
	teacher : { name : "Rubeus", surname : "Hagrid" },
	year : [4, 5, 6],
	scheduled : [{
		day : "Tuesday",
		hour : "14:00",
		place : "Forbidden Forest" }]
}
])
```

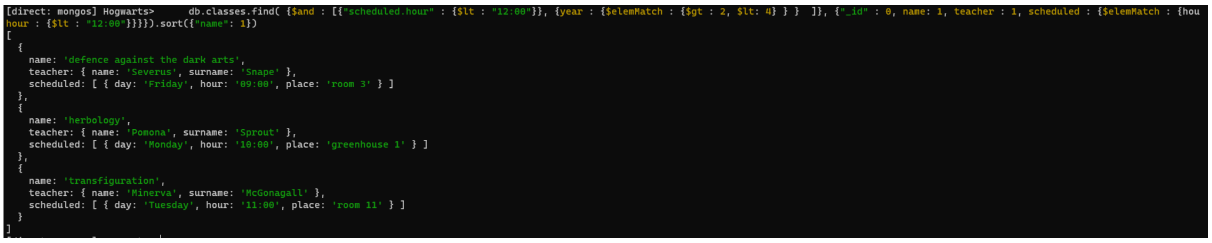
---- HOUSES ---------------------------------------

```
db.houses.insertMany([
{
	name : "Gryffindor",
	founder :
		{name : "Godryk", surname : "Gryffindor" },
	head :
		{name : "Minerva", surname : "McGonagall" },
        animal : "lion"
},
{
        name : "Slytherin",
        founder :
                {name : "Salazar", surname : "Slytherin" },
        head :
                {name : "Severus", surname : "Snape" },
        animal : "snake"
},
{
        name : "Hufflepuff",
        founder :
                {name : "Helga", surname : "Hufflepuff" },
        head :
                {name : "Pomona", surname : "Sprout" },
        animal : "badger"
},
{
        name : "Ravenclaw",
        founder :
                {name : "Rowena", surname : "Ravenclaw" },
        head :
                {name : "Filius", surname : "Flitwick" },
        animal : "eagle"
}
])
```

3. 4 queries retrieving the data according to envisioned usage (at least one of them being aggregation pipeline statement using at least two collections)

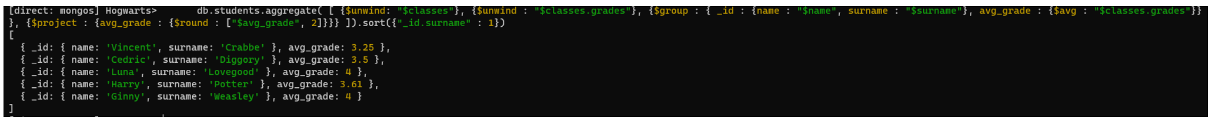
- Find classes for year 3 that start before noon (12:00) and sort them alphabetically

```
        db.classes.find( {$and : [{"scheduled.hour" : {$lt : "12:00"}}, {year : {$elemMatch : {$gt
: 2, $lt: 4} } }  ]}, {"_id" : 0, name: 1, teacher : 1, scheduled : {$elemMatch : {hour : {$lt :
"12:00"}}}}).sort({"name": 1})
```



```
[direct: mongos] Hogwarts>      db.classes.find( {$and : [{"scheduled.hour" : {$lt : "12:00"}}, {year : {$elemMatch : {$gt : 2, $lt: 4} } }  ]}, {"_id" : 0, name: 1, teacher : 1, scheduled : {$elemMatch : {hou
hour : {$lt : "12:00"}}}}).sort({"name": 1})
[
  {
    name: 'defence against the dark arts',
    teacher: { name: 'Severus', surname: 'Snape' },
    scheduled: [ { day: 'Friday', hour: '09:00', place: 'room 3' } ]
  },
  {
    name: 'herbology',
    teacher: { name: 'Pomona', surname: 'Sprout' },
    scheduled: [ { day: 'Monday', hour: '10:00', place: 'greenhouse 1' } ]
  },
  {
    name: 'transfiguration',
    teacher: { name: 'Minerva', surname: 'McGonagall' },
    scheduled: [ { day: 'Tuesday', hour: '11:00', place: 'room 11' } ]
  }
]
```

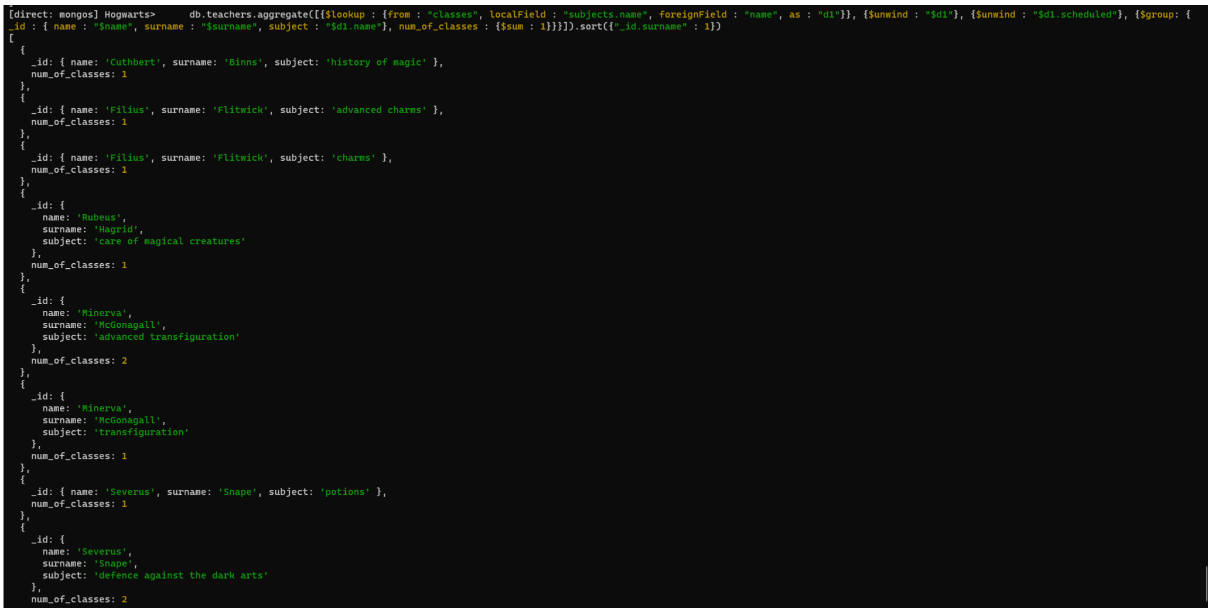
- For each student find their average grade from all subjects db.students.aggregate([{$unwind: "$classes"}, {$unwind : "$classes.grades"}, {$group : { _id : {name : "$name", surname : "$surname"}, avg_grade : {$avg : "$classes.grades"}}}, {$project : {avg_grade : {$round : ["$avg_grade", 2]}}}]).sort({"_id.surname" : 1})



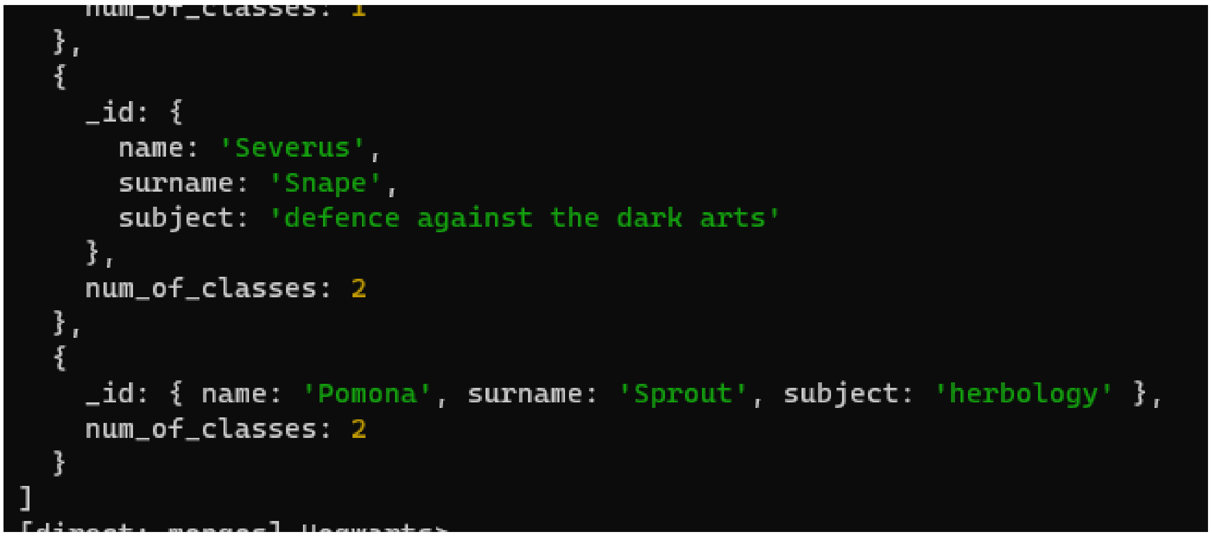
```
[direct: mongos] Hogwarts>      db.students.aggregate( [ {$unwind: "$classes"}, {$unwind : "$classes.grades"}, {$group : { _id : {name : "$name", surname : "$surname"}, avg_grade : {$avg : "$classes.grades"}}
}, {$project : {avg_grade : {$round : ["$avg_grade", 2]}}} ]).sort({"_id.surname" : 1})
[
  { _id: { name: 'Vincent', surname: 'Crabbe' }, avg_grade: 3.25 },
  { _id: { name: 'Cedric', surname: 'Diggory' }, avg_grade: 3.5 },
  { _id: { name: 'Luna', surname: 'Lovegood' }, avg_grade: 4 },
  { _id: { name: 'Harry', surname: 'Potter' }, avg_grade: 3.61 },
  { _id: { name: 'Ginny', surname: 'Weasley' }, avg_grade: 4 }
]
```

- For each teacher find subjects they teach (one teacher can teach many subjects) and for each subject find number of scheduled classes (one subjects can be scheduled for many days)

db.teachers.aggregate([{$lookup : {from : "classes", localField : "subjects.name", foreignField : "name", as : "d1"}}, {$unwind : "$d1"}, {$unwind : "$d1.scheduled"}, {$group: {_id : { name : "$name", surname : "$surname", subject : "$d1.name"}, num_of_classes : {$sum : 1}}}]).sort({"_id.surname" : 1})



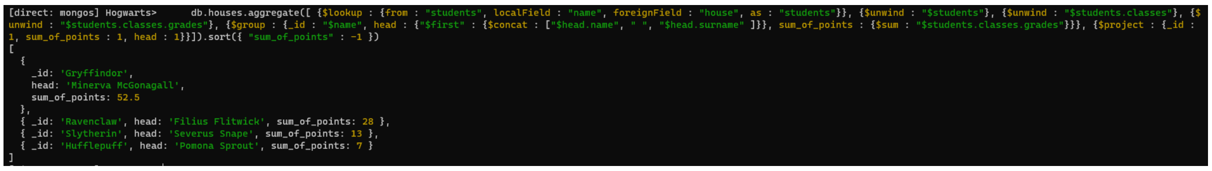
```
[direct: mongos] Hogwarts>      db.teachers.aggregate([{$lookup : {from : "classes", localField : "subjects.name", foreignField : "name", as : "d1"}}, {$unwind : "$d1"}, {$unwind : "$d1.scheduled"}, {$group: {
_id : { name : "$name", surname : "$surname", subject : "$d1.name"}, num_of_classes : {$sum : 1}}}]).sort({"_id.surname" : 1})
[
  {
    _id: { name: 'Cuthbert', surname: 'Binns', subject: 'history of magic' },
    num_of_classes: 1
  },
  {
    _id: { name: 'Filius', surname: 'Flitwick', subject: 'advanced charms' },
    num_of_classes: 1
  },
  {
    _id: { name: 'Filius', surname: 'Flitwick', subject: 'charms' },
    num_of_classes: 1
  },
  {
    _id: {
      name: 'Rubeus',
      surname: 'Hagrid',
      subject: 'care of magical creatures'
    },
    num_of_classes: 1
  },
  {
    _id: {
      name: 'Minerva',
      surname: 'McGonagall',
      subject: 'advanced transfiguration'
    },
    num_of_classes: 2
  },
  {
    _id: {
      name: 'Minerva',
      surname: 'McGonagall',
      subject: 'transfiguration'
    },
    num_of_classes: 1
  },
  {
    _id: { name: 'Severus', surname: 'Snape', subject: 'potions' },
    num_of_classes: 1
  },
  {
    _id: {
      name: 'Severus',
      surname: 'Snape',
      subject: 'defence against the dark arts'
    },
    num_of_classes: 2
```



```
    num_of_classes: 1
  },
  {
    _id: {
      name: 'Severus',
      surname: 'Snape',
      subject: 'defence against the dark arts'
    },
    num_of_classes: 2
  },
  {
    _id: { name: 'Pomona', surname: 'Sprout', subject: 'herbology' },
    num_of_classes: 2
  }
]
[direct: mongos] Hogwarts>
```

- For each house, retrieve the number of house points (sum of all grades of all students of that house) and information about the name of head of the house. Sort the houses from the biggest number of points.

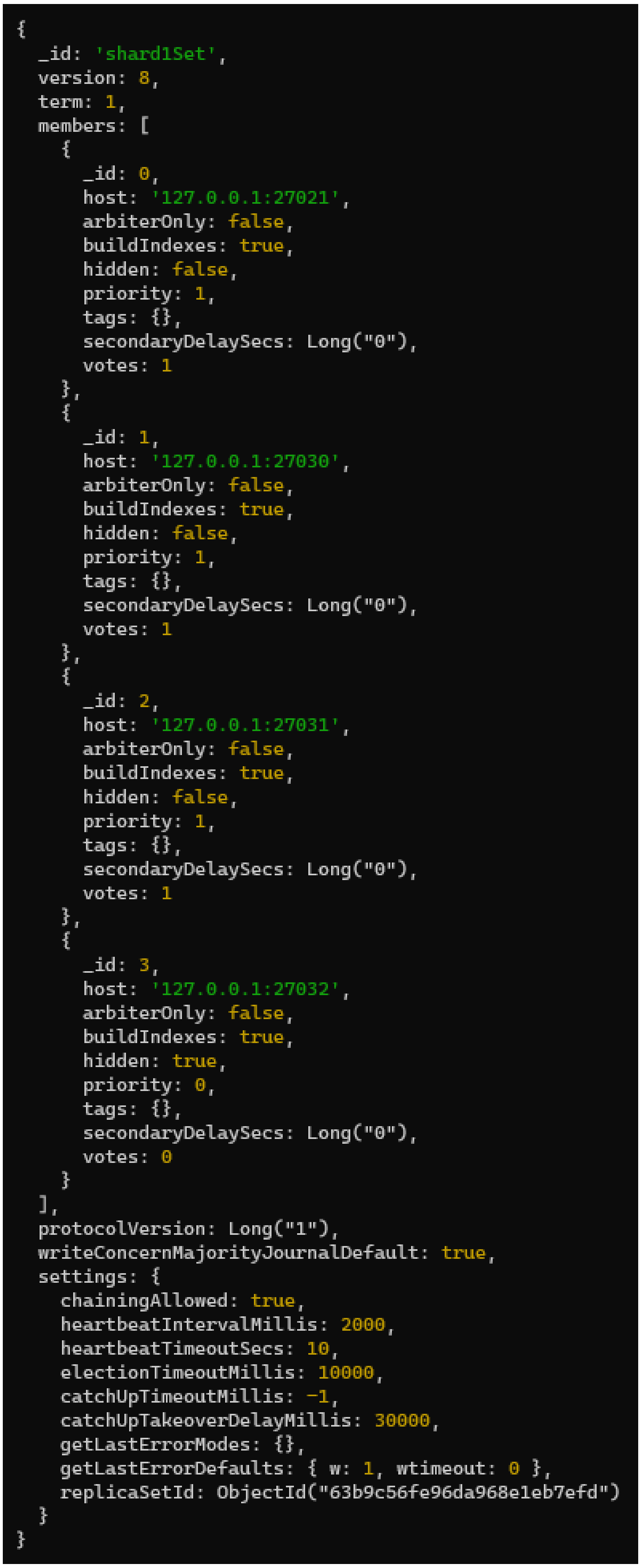
db.houses.aggregate([{$lookup : {from : "students", localField : "name", foreignField : "house", as : "students"}}, {$unwind : "$students"}, {$unwind : "$students.classes"}, {$unwind : "$students.classes.grades"}, {$group : {_id : "$name", head : {"$first" : {$concat : ["$head.name", " ", "$head.surname"]}}, sum_of_points : {$sum : "$students.classes.grades"}}}, {$project : {_id : 1, sum_of_points : 1, head : 1}}]).sort({ "sum_of_points" : -1 })



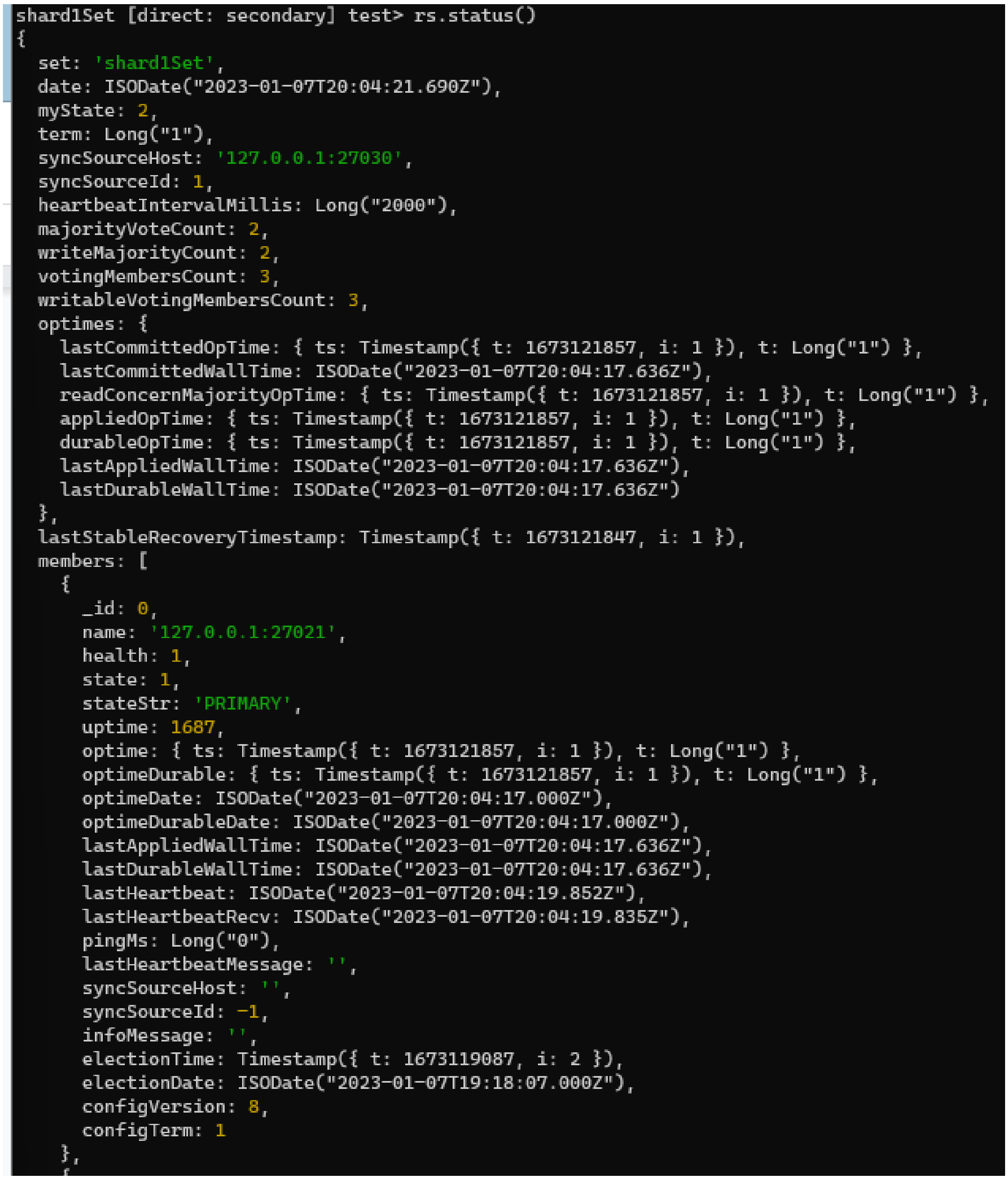
```
[direct: mongos] Hogwarts>      db.houses.aggregate([ {$lookup : {from : "students", localField : "name", foreignField : "house", as : "students"}}, {$unwind : "$students"}, {$unwind : "$students.classes"}, {$unwind : "$students.classes.grades"}, {$group : {_id : "$name", head : {"$first" : {$concat : ["$head.name", " ", "$head.surname" ]}}, sum_of_points : {$sum : "$students.classes.grades"}}}, {$project : {_id : 1, sum_of_points : 1, head : 1}}]).sort({ "sum_of_points" : -1 })
[
  {
    _id: 'Gryffindor',
    head: 'Minerva McGonagall',
    sum_of_points: 52.5
  },
  { _id: 'Ravenclaw', head: 'Filius Flitwick', sum_of_points: 28 },
  { _id: 'Slytherin', head: 'Severus Snape', sum_of_points: 13 },
  { _id: 'Hufflepuff', head: 'Pomona Sprout', sum_of_points: 7 }
]
```

4. Configuration and status of both replicationSets

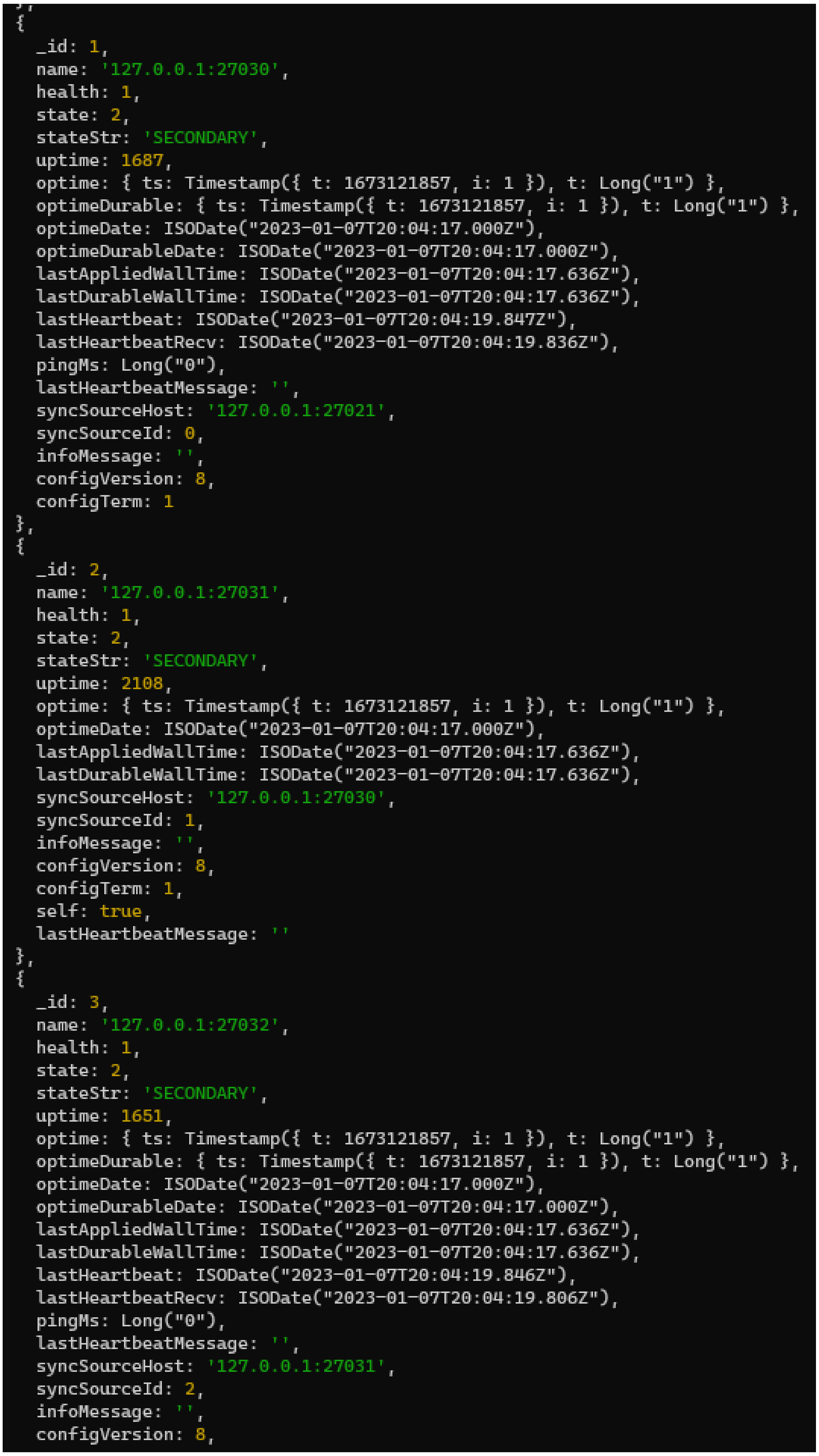
shard1Set [direct: secondary] test> rs.conf()


```
{
  _id: 'shard1Set',
  version: 8,
  term: 1,
  members: [
    {
      _id: 0,
      host: '127.0.0.1:27021',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 1,
      host: '127.0.0.1:27030',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 2,
      host: '127.0.0.1:27031',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 3,
      host: '127.0.0.1:27032',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: true,
      priority: 0,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 0
    }
  ],
  protocolVersion: Long("1"),
  writeConcernMajorityJournalDefault: true,
  settings: {
    chainingAllowed: true,
    heartbeatIntervalMillis: 2000,
    heartbeatTimeoutSecs: 10,
    electionTimeoutMillis: 10000,
    catchUpTimeoutMillis: -1,
    catchUpTakeoverDelayMillis: 30000,
    getLastErrorModes: {},
    getLastErrorDefaults: { w: 1, wtimeout: 0 },
    replicaSetId: ObjectId("63b9c56fe96da968e1eb7efd")
  }
}
```

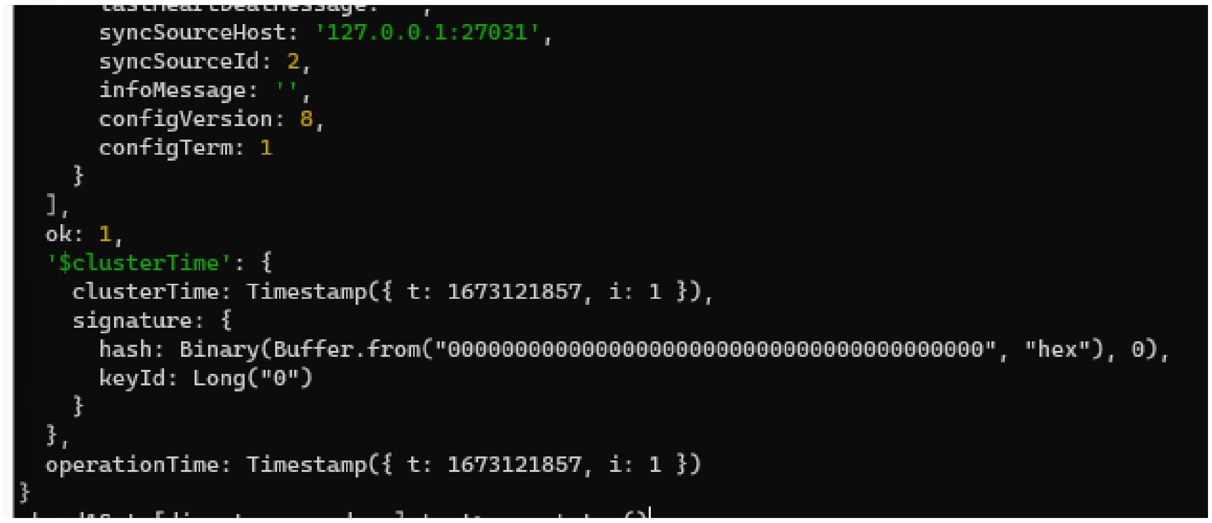
shard1Set [direct: secondary] test> rs.status()



```
shard1Set [direct: secondary] test> rs.status()
{
  set: 'shard1Set',
  date: ISODate("2023-01-07T20:04:21.690Z"),
  myState: 2,
  term: Long("1"),
  syncSourceHost: '127.0.0.1:27030',
  syncSourceId: 1,
  heartbeatIntervalMillis: Long("2000"),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
  writableVotingMembersCount: 3,
  optimes: {
    lastCommittedOpTime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
    lastCommittedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
    readConcernMajorityOpTime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
    appliedOpTime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
    durableOpTime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
    lastAppliedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
    lastDurableWallTime: ISODate("2023-01-07T20:04:17.636Z")
  },
  lastStableRecoveryTimestamp: Timestamp({ t: 1673121847, i: 1 }),
  members: [
    {
      _id: 0,
      name: '127.0.0.1:27021',
      health: 1,
      state: 1,
      stateStr: 'PRIMARY',
      uptime: 1687,
      optime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
      optimeDurable: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
      optimeDate: ISODate("2023-01-07T20:04:17.000Z"),
      optimeDurableDate: ISODate("2023-01-07T20:04:17.000Z"),
      lastAppliedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
      lastDurableWallTime: ISODate("2023-01-07T20:04:17.636Z"),
      lastHeartbeat: ISODate("2023-01-07T20:04:19.852Z"),
      lastHeartbeatRecv: ISODate("2023-01-07T20:04:19.835Z"),
      pingMs: Long("0"),
      lastHeartbeatMessage: '',
      syncSourceHost: '',
      syncSourceId: -1,
      infoMessage: '',
      electionTime: Timestamp({ t: 1673119087, i: 2 }),
      electionDate: ISODate("2023-01-07T19:18:07.000Z"),
      configVersion: 8,
      configTerm: 1
    },
```

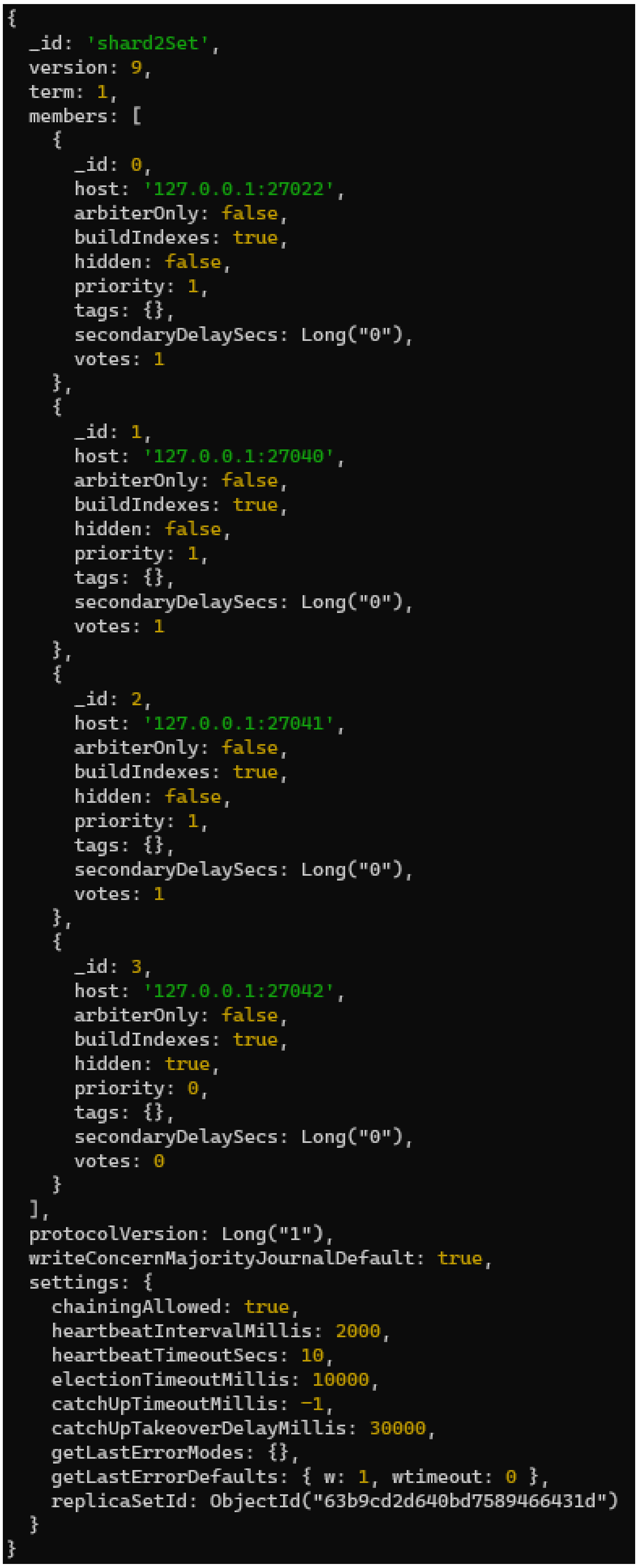


```
},
{
  _id: 1,
  name: '127.0.0.1:27030',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 1687,
  optime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:04:17.000Z"),
  optimeDurableDate: ISODate("2023-01-07T20:04:17.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  lastHeartbeat: ISODate("2023-01-07T20:04:19.847Z"),
  lastHeartbeatRecv: ISODate("2023-01-07T20:04:19.836Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '127.0.0.1:27021',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 8,
  configTerm: 1
},
{
  _id: 2,
  name: '127.0.0.1:27031',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 2108,
  optime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:04:17.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  syncSourceHost: '127.0.0.1:27030',
  syncSourceId: 1,
  infoMessage: '',
  configVersion: 8,
  configTerm: 1,
  self: true,
  lastHeartbeatMessage: ''
},
{
  _id: 3,
  name: '127.0.0.1:27032',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 1651,
  optime: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1673121857, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:04:17.000Z"),
  optimeDurableDate: ISODate("2023-01-07T20:04:17.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:04:17.636Z"),
  lastHeartbeat: ISODate("2023-01-07T20:04:19.846Z"),
  lastHeartbeatRecv: ISODate("2023-01-07T20:04:19.806Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '127.0.0.1:27031',
  syncSourceId: 2,
  infoMessage: '',
  configVersion: 8,
```



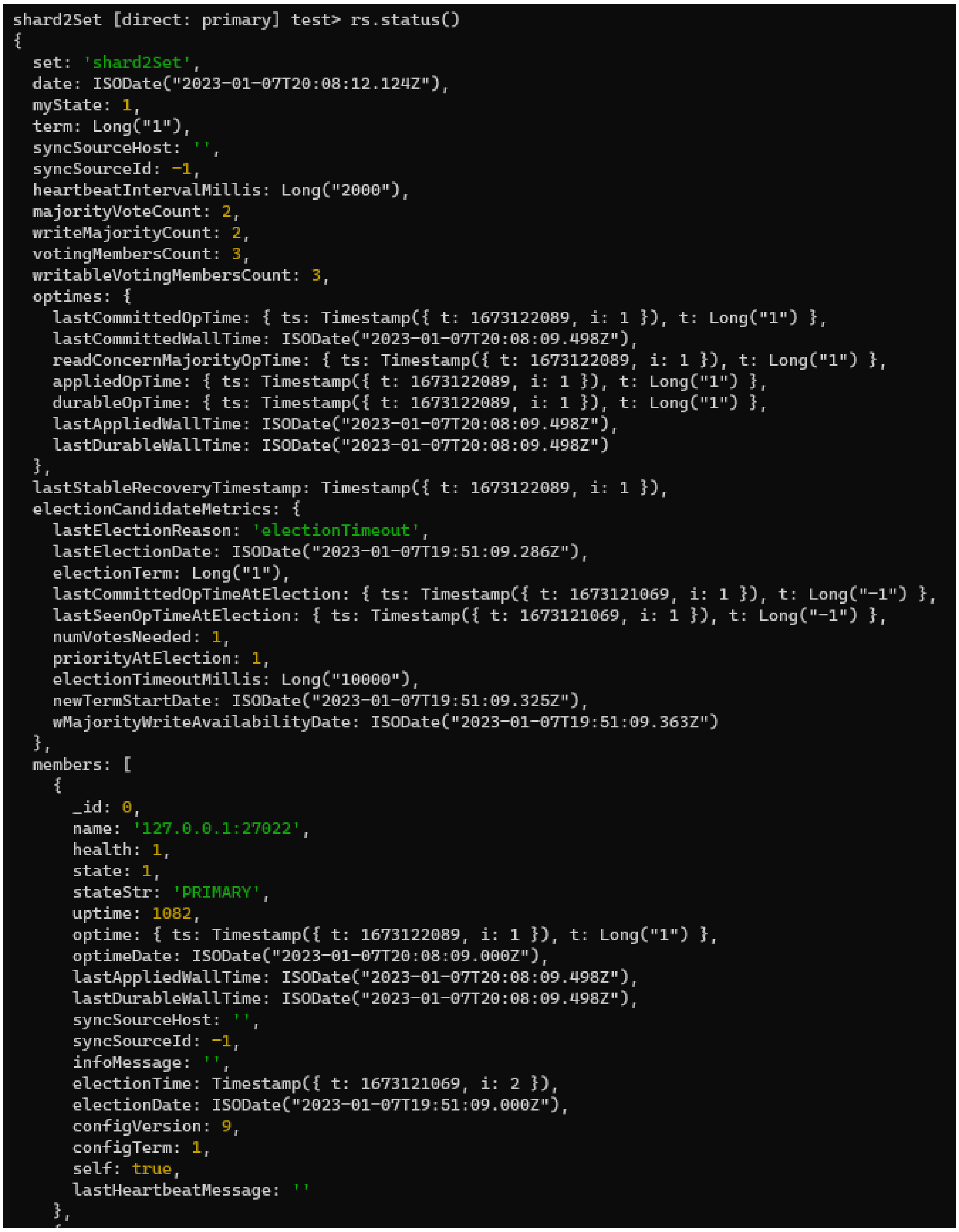
```
      lastHeartbeatMessage: '',
      syncSourceHost: '127.0.0.1:27031',
      syncSourceId: 2,
      infoMessage: '',
      configVersion: 8,
      configTerm: 1
    }
  ],
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1673121857, i: 1 }),
    signature: {
      hash: Binary(Buffer.from("0000000000000000000000000000000000000000", "hex"), 0),
      keyId: Long("0")
    }
  },
  operationTime: Timestamp({ t: 1673121857, i: 1 })
}
```

shard2Set [direct: primary] test> rs.conf()

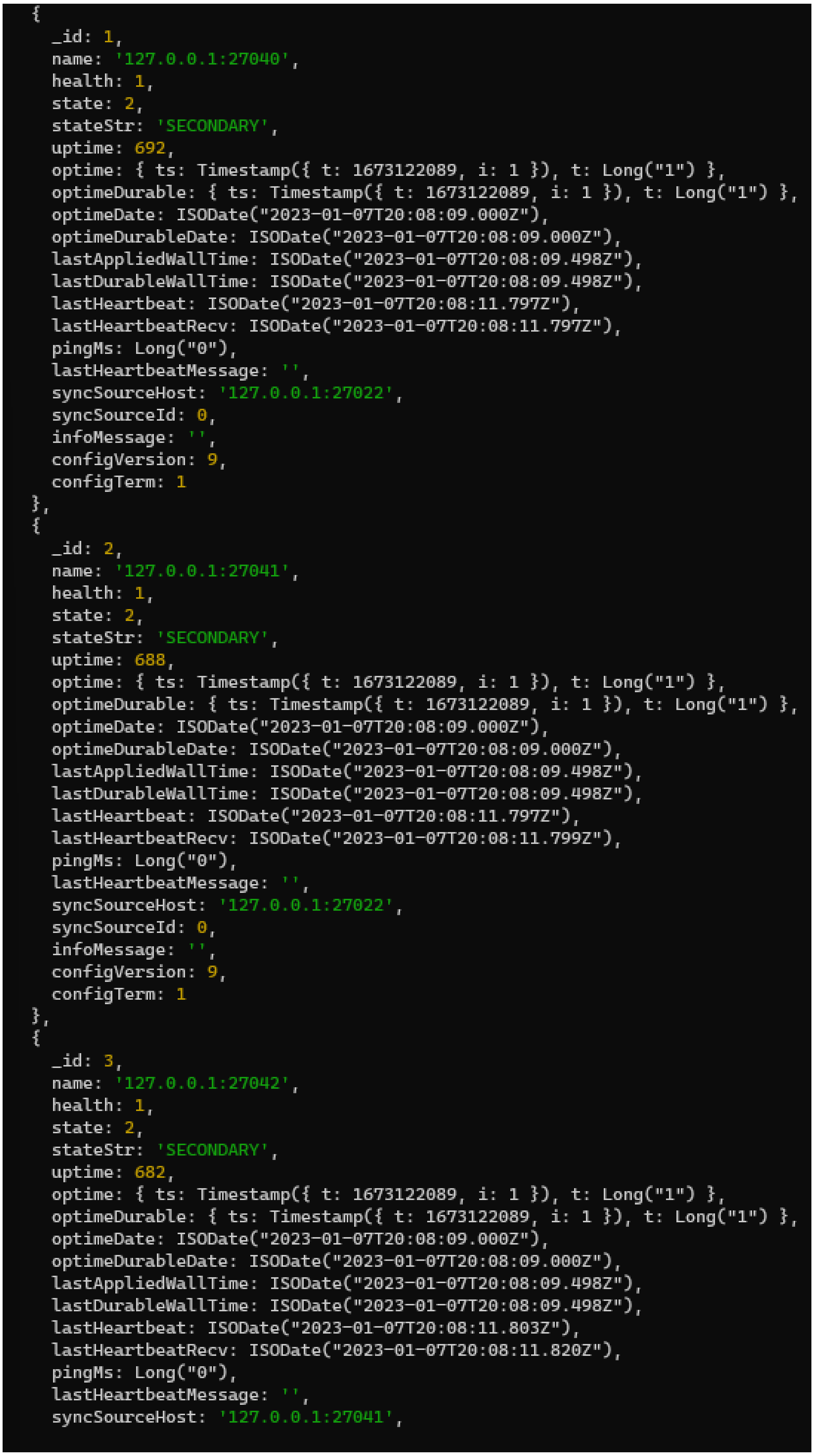


```
{
  _id: 'shard2Set',
  version: 9,
  term: 1,
  members: [
    {
      _id: 0,
      host: '127.0.0.1:27022',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 1,
      host: '127.0.0.1:27040',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 2,
      host: '127.0.0.1:27041',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
    {
      _id: 3,
      host: '127.0.0.1:27042',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: true,
      priority: 0,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 0
    }
  ],
  protocolVersion: Long("1"),
  writeConcernMajorityJournalDefault: true,
  settings: {
    chainingAllowed: true,
    heartbeatIntervalMillis: 2000,
    heartbeatTimeoutSecs: 10,
    electionTimeoutMillis: 10000,
    catchUpTimeoutMillis: -1,
    catchUpTakeoverDelayMillis: 30000,
    getLastErrorModes: {},
    getLastErrorDefaults: { w: 1, wtimeout: 0 },
    replicaSetId: ObjectId("63b9cd2d640bd7589466431d")
  }
}
```

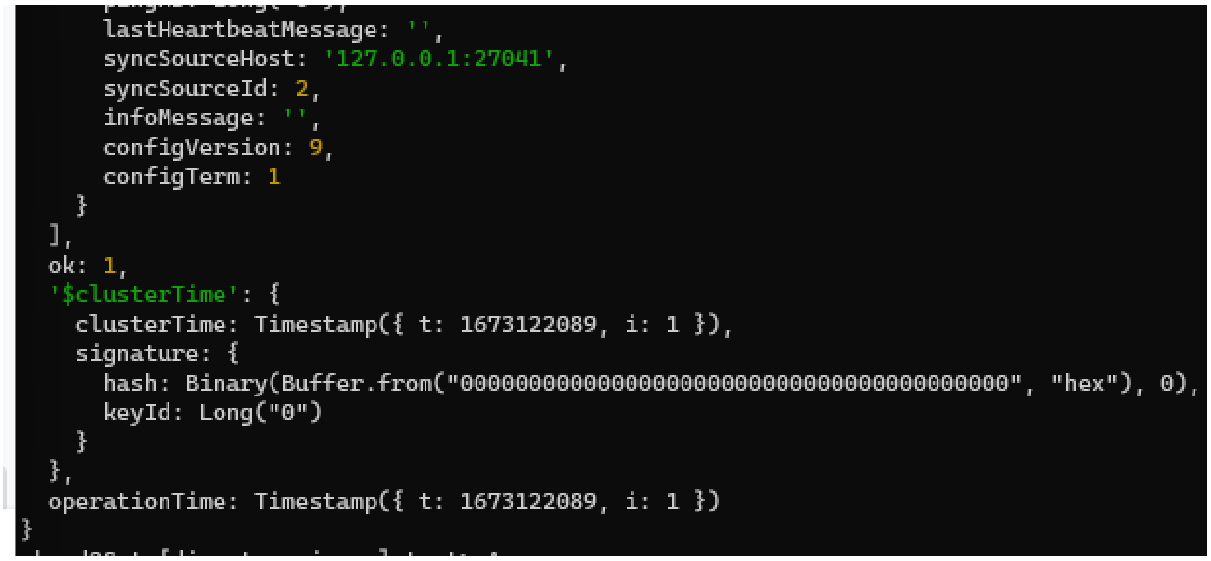
shard2Set [direct: primary] test> rs.status()



```
shard2Set [direct: primary] test> rs.status()
{
  set: 'shard2Set',
  date: ISODate("2023-01-07T20:08:12.124Z"),
  myState: 1,
  term: Long("1"),
  syncSourceHost: '',
  syncSourceId: -1,
  heartbeatIntervalMillis: Long("2000"),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
  writableVotingMembersCount: 3,
  optimes: {
    lastCommittedOpTime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
    lastCommittedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
    readConcernMajorityOpTime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
    appliedOpTime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
    durableOpTime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
    lastAppliedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
    lastDurableWallTime: ISODate("2023-01-07T20:08:09.498Z")
  },
  lastStableRecoveryTimestamp: Timestamp({ t: 1673122089, i: 1 }),
  electionCandidateMetrics: {
    lastElectionReason: 'electionTimeout',
    lastElectionDate: ISODate("2023-01-07T19:51:09.286Z"),
    electionTerm: Long("1"),
    lastCommittedOpTimeAtElection: { ts: Timestamp({ t: 1673121069, i: 1 }), t: Long("-1") },
    lastSeenOpTimeAtElection: { ts: Timestamp({ t: 1673121069, i: 1 }), t: Long("-1") },
    numVotesNeeded: 1,
    priorityAtElection: 1,
    electionTimeoutMillis: Long("10000"),
    newTermStartDate: ISODate("2023-01-07T19:51:09.325Z"),
    wMajorityWriteAvailabilityDate: ISODate("2023-01-07T19:51:09.363Z")
  },
  members: [
    {
      _id: 0,
      name: '127.0.0.1:27022',
      health: 1,
      state: 1,
      stateStr: 'PRIMARY',
      uptime: 1082,
      optime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
      optimeDate: ISODate("2023-01-07T20:08:09.000Z"),
      lastAppliedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
      lastDurableWallTime: ISODate("2023-01-07T20:08:09.498Z"),
      syncSourceHost: '',
      syncSourceId: -1,
      infoMessage: '',
      electionTime: Timestamp({ t: 1673121069, i: 2 }),
      electionDate: ISODate("2023-01-07T19:51:09.000Z"),
      configVersion: 9,
      configTerm: 1,
      self: true,
      lastHeartbeatMessage: ''
    },
```

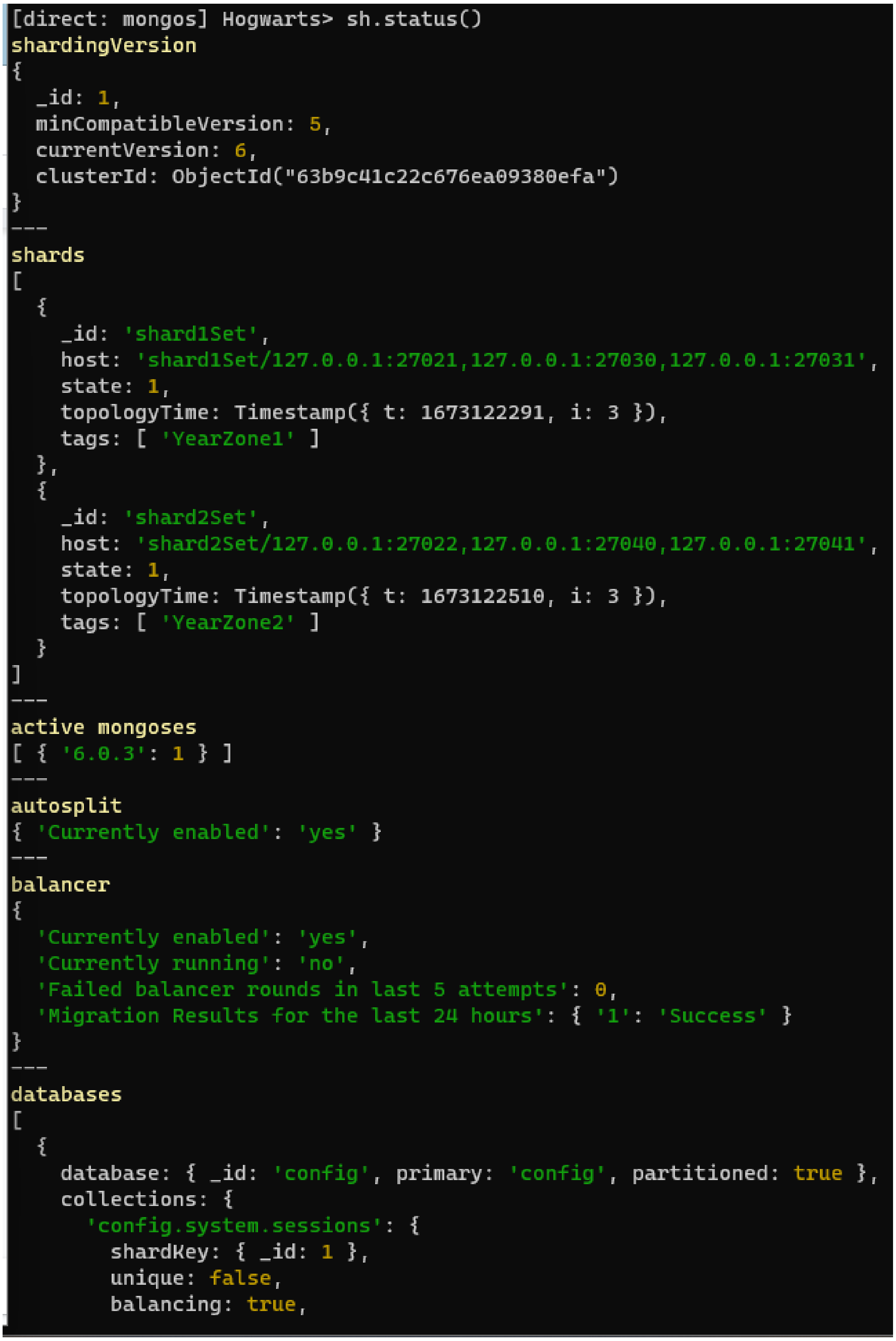


```
{
  _id: 1,
  name: '127.0.0.1:27040',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 692,
  optime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:08:09.000Z"),
  optimeDurableDate: ISODate("2023-01-07T20:08:09.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastHeartbeat: ISODate("2023-01-07T20:08:11.797Z"),
  lastHeartbeatRecv: ISODate("2023-01-07T20:08:11.797Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '127.0.0.1:27022',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 9,
  configTerm: 1
},
{
  _id: 2,
  name: '127.0.0.1:27041',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 688,
  optime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:08:09.000Z"),
  optimeDurableDate: ISODate("2023-01-07T20:08:09.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastHeartbeat: ISODate("2023-01-07T20:08:11.797Z"),
  lastHeartbeatRecv: ISODate("2023-01-07T20:08:11.799Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '127.0.0.1:27022',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 9,
  configTerm: 1
},
{
  _id: 3,
  name: '127.0.0.1:27042',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 682,
  optime: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDurable: { ts: Timestamp({ t: 1673122089, i: 1 }), t: Long("1") },
  optimeDate: ISODate("2023-01-07T20:08:09.000Z"),
  optimeDurableDate: ISODate("2023-01-07T20:08:09.000Z"),
  lastAppliedWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastDurableWallTime: ISODate("2023-01-07T20:08:09.498Z"),
  lastHeartbeat: ISODate("2023-01-07T20:08:11.803Z"),
  lastHeartbeatRecv: ISODate("2023-01-07T20:08:11.820Z"),
  pingMs: Long("0"),
  lastHeartbeatMessage: '',
  syncSourceHost: '127.0.0.1:27041',
```

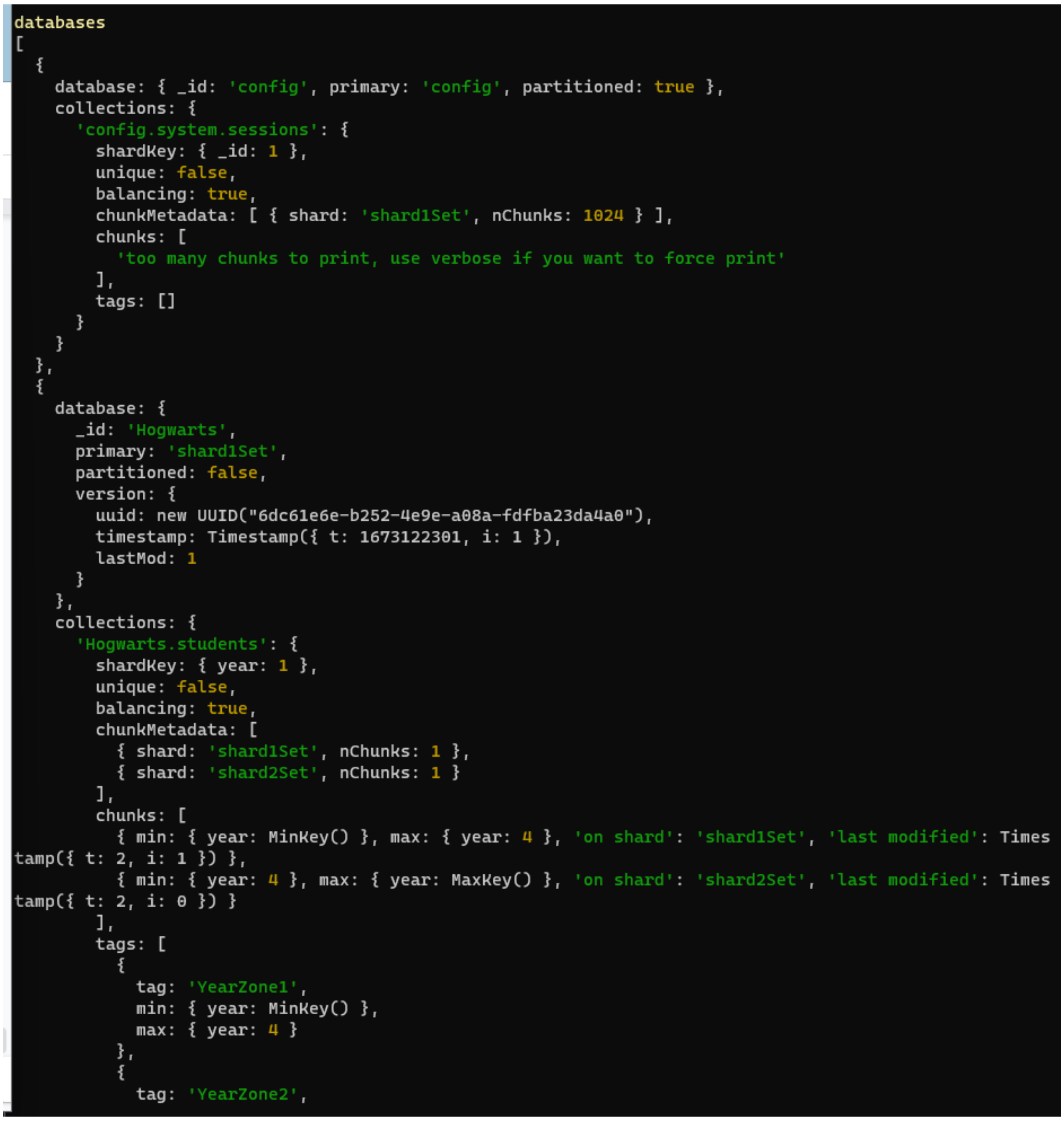


```
      lastHeartbeatMessage: '',
      syncSourceHost: '127.0.0.1:27041',
      syncSourceId: 2,
      infoMessage: '',
      configVersion: 9,
      configTerm: 1
    }
  ],
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1673122089, i: 1 }),
    signature: {
      hash: Binary(Buffer.from("0000000000000000000000000000000000000000", "hex"), 0),
      keyId: Long("0")
    }
  },
  operationTime: Timestamp({ t: 1673122089, i: 1 })
}
```

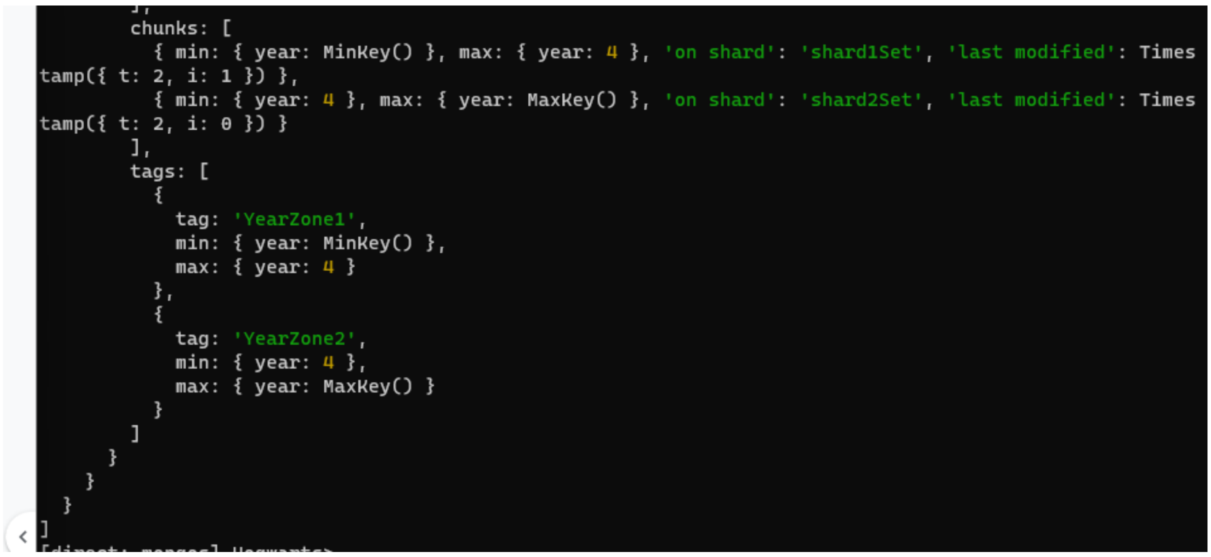
5. Status of sharding (after chosen collection has been sharded and shard-key ranges have been assigned to zones)



```
[direct: mongos] Hogwarts> sh.status()
shardingVersion
{
  _id: 1,
  minCompatibleVersion: 5,
  currentVersion: 6,
  clusterId: ObjectId("63b9c41c22c676ea09380efa")
}
---
shards
[
  {
    _id: 'shard1Set',
    host: 'shard1Set/127.0.0.1:27021,127.0.0.1:27030,127.0.0.1:27031',
    state: 1,
    topologyTime: Timestamp({ t: 1673122291, i: 3 }),
    tags: [ 'YearZone1' ]
  },
  {
    _id: 'shard2Set',
    host: 'shard2Set/127.0.0.1:27022,127.0.0.1:27040,127.0.0.1:27041',
    state: 1,
    topologyTime: Timestamp({ t: 1673122510, i: 3 }),
    tags: [ 'YearZone2' ]
  }
]
---
active mongoses
[ { '6.0.3': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Currently running': 'no',
  'Failed balancer rounds in last 5 attempts': 0,
  'Migration Results for the last 24 hours': { '1': 'Success' }
}
---
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
```



```
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'shard1Set', nChunks: 1024 } ],
        chunks: [
          'too many chunks to print, use verbose if you want to force print'
        ],
        tags: []
      }
    }
  },
  {
    database: {
      _id: 'Hogwarts',
      primary: 'shard1Set',
      partitioned: false,
      version: {
        uuid: new UUID("6dc61e6e-b252-4e9e-a08a-fdfba23da4a0"),
        timestamp: Timestamp({ t: 1673122301, i: 1 }),
        lastMod: 1
      }
    },
    collections: {
      'Hogwarts.students': {
        shardKey: { year: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [
          { shard: 'shard1Set', nChunks: 1 },
          { shard: 'shard2Set', nChunks: 1 }
        ],
        chunks: [
          { min: { year: MinKey() }, max: { year: 4 }, 'on shard': 'shard1Set', 'last modified': Times
tamp({ t: 2, i: 1 }) },
          { min: { year: 4 }, max: { year: MaxKey() }, 'on shard': 'shard2Set', 'last modified': Times
tamp({ t: 2, i: 0 }) }
        ],
        tags: [
          {
            tag: 'YearZone1',
            min: { year: MinKey() },
            max: { year: 4 }
          },
          {
            tag: 'YearZone2',
```



```
        ],
        chunks: [
          { min: { year: MinKey() }, max: { year: 4 }, 'on shard': 'shard1Set', 'last modified': Times
tamp({ t: 2, i: 1 }) },
          { min: { year: 4 }, max: { year: MaxKey() }, 'on shard': 'shard2Set', 'last modified': Times
tamp({ t: 2, i: 0 }) }
        ],
        tags: [
          {
            tag: 'YearZone1',
            min: { year: MinKey() },
            max: { year: 4 }
          },
          {
            tag: 'YearZone2',
            min: { year: 4 },
            max: { year: MaxKey() }
          }
        ]
      }
    }
  }
]
[direct: mongos] Hogwarts>
```