

Міністерство освіти і науки України  
НТУУ «Київський політехнічний інститут»  
Фізико-технічний інститут

**Проектування високонавантажених систем**

Лабораторна робота №7  
Налаштування шардінгу в MongoDB

**Виконав:**

Студент 4-го курсу  
групи ФІ-21

Климентьев Максим

**Перевірив:**

---

# **Зміст**

<b>1</b>	<b>Код реалізації</b>	<b>3</b>
<b>2</b>	<b>Результати</b>	<b>6</b>

# 1 Код реалізації

## all\_tests.bat

```
@echo off
echo [START] Running Setup...
call .\scripts\setup.bat

echo.
echo [START] Crashing Test Data..
call .\scripts\crash_test.bat

echo.
echo [DONE] All scripts executed.
```

## setup.bat

```
docker exec -it config-server mongosh --eval "rs.initiate({_id: 'rs_config', configsvr: true, members: [{_id: 0, host: 'config-server:27017'}]})"

docker exec -it shard1 mongosh --eval "rs.initiate({_id: 'rs_shard1', members: [{_id: 0, host: 'shard1:27017'}]})"
docker exec -it shard2 mongosh --eval "rs.initiate({_id: 'rs_shard2', members: [{_id: 0, host: 'shard2:27017'}]})"
docker exec -it shard3 mongosh --eval "rs.initiate({_id: 'rs_shard3', members: [{_id: 0, host: 'shard3:27017'}]})"

docker exec -it mongos mongosh -f "scripts/setup.js"
```

## crash\_test.bat

```
docker stop shard2
docker exec -it mongos mongosh -f "scripts/crash_test.js"
docker start shard2
docker exec -it mongos mongosh -f "scripts/crash_test.js"
```

## setup.js

```
var databaseName = "RLS";
var database = db.getSiblingDB(databaseName);
var NameUsers = databaseName + ".Users";
var NameTweets = databaseName + ".Tweets";

var rs1 = "rs_shard1";
var rs2 = "rs_shard2";
var rs3 = "rs_shard3";

var eu = "ZONE_EU";
var usa = "ZONE_USA";
var asia = "ZONE_ASIA";
var low = "LIKES_LOW";
var mid = "LIKES_MID";
var high = "LIKES_HIGH";

sh.addShard(rs1 + "/shard1:27017");
sh.addShard(rs2 + "/shard2:27017");
sh.addShard(rs3 + "/shard3:27017");

sh.enableSharding(databaseName);

try
{
    sh.removeShardFromZone(rs1, eu);
    sh.removeShardFromZone(rs2, usa);
    sh.removeShardFromZone(rs3, asia);

    sh.removeShardFromZone(rs1, low);
    sh.removeShardFromZone(rs2, mid);
    sh.removeShardFromZone(rs3, high);
}
catch(e) {}

print("[START] Tagging shards...");
sh.addShardToZone(rs1, eu);
sh.addShardToZone(rs2, usa);
sh.addShardToZone(rs3, asia);
```

```

sh.addShardToZone(rs1, low);
sh.addShardToZone(rs2, mid);
sh.addShardToZone(rs3, high);

print("[START] setting regions for Users...");
try { database.Users.drop(); } catch(e) {}
database.Users.createIndex({ region: 1, _id: 1 });
sh.shardCollection(NameUsers, { region: 1, _id: 1 });

sh.updateZoneKeyRange(NameUsers, { region: "EU", _id: MinKey }, { region: "EU", _id: MaxKey }, eu);
sh.updateZoneKeyRange(NameUsers, { region: "USA", _id: MinKey }, { region: "USA", _id: MaxKey }, usa);
sh.updateZoneKeyRange(NameUsers, { region: "Asia", _id: MinKey }, { region: "Asia", _id: MaxKey }, asia);

print("[START] setting Tweets...");
try { database.Tweets.drop(); } catch(e) {}
database.Tweets.createIndex({ likes: 1 });
sh.shardCollection(NameTweets, { likes: 1 });

sh.updateZoneKeyRange(NameTweets, { likes: 0 }, { likes: 101 }, low);
sh.updateZoneKeyRange(NameTweets, { likes: 101 }, { likes: 201 }, mid);
sh.updateZoneKeyRange(NameTweets, { likes: 201 }, { likes: MaxKey }, high);

print("[START] Cluster setup complete!");
print(sh.status());

print("[START] Inserting...");
var tweets = [
  { m: "Low1", l: 5 }, { m: "Low2", l: 50 }, { m: "Low3", l: 99 }, // -> Shard 1
  { m: "Mid1", l: 105 }, { m: "Mid2", l: 150 }, { m: "Mid3", l: 199 }, // -> Shard 2
  { m: "High1", l: 300 }, { m: "High2", l: 5000 }, { m: "High3", l: 9999 } // -> Shard 3
];
tweets.forEach(function(t) {
  database.Tweets.insertOne({ msg: t.m, likes: t.l });
});

database.Users.insertMany([
  { region: "Asia", name: "user_as_1" },
  { region: "Asia", name: "user_as_2" },
  { region: "Asia", name: "user_as_3" },
  { region: "Asia", name: "user_as_4" }
])
for (let i = 0; i < 5000; i++)
{
  database.Users.insertOne({
    region: i % 2 === 0 ? "EU" : "USA",
    name: "user_" + i
  })
}

print(database.Users.getShardDistribution());
print(database.Users.countDocuments());
try { database.Users.deleteMany({}); } catch(e) {}

db.adminCommand({ flushRouterConfig: "RLS.Users" });
db.adminCommand({ flushRouterConfig: "RLS.Tweets" });

print("[START] Setup complete!");

```

## crash\_test.js

```

var databaseName = "RLS";
var database = db.getSiblingDB(databaseName);

function testOperation(name, func) {
  print("[TEST] " + name);
  try {
    var start = new Date();
    var result = func();
    var time = (new Date() - start) + "ms";
    print("[SUCCESS] (" + time + "): " + JSON.stringify(result));
  } catch (e) {
    print("[ERROR] " + e.message);
  }
}

testOperation("INSERT in zone USA (Shard 2 - Down)", function() {
  return database.Users.insertOne({ name: "DeadUser", region: "USA" });
});

```

```
testOperation("INSERT in zone EU (Shard 1 - Up)", function() {
    return database.Users.insertOne({ name: "LiveUser", region: "EU" });
});

testOperation("FIND in zone USA (Shard 2 - Down)", function() {
    return database.Users.find({ region: "USA" }).toArray();
});

testOperation("FIND in zone EU (Shard 1 - Up)", function() {
    return database.Users.find({ region: "EU" }).allowPartialResults().toArray();
});

testOperation("RANGE FIND Likes 101-200 (Shard 2 - Down)", function() {
    return database.Tweets.find({ likes: { $gt: 100, $lt: 200 } }).toArray();
});

testOperation("RANGE FIND Likes 0-100 (Shard 1 - Up)", function() {
    return database.Tweets.find({ likes: { $gt: 0, $lt: 100 } }).allowPartialResults().toArray();
});

print(sh.status());
```

## 2 Результати

1. Сконфігуруйте 3 інстанси (сервери) MongoDB у якості шард. Ці шарди потім будуть зв'язані з Zones (<https://www.mongodb.com/docs/manual/core/zone-sharding/>)
  - Ranged Sharding (with Zone Ranges)
  - Zones
2. Створіть колекцію Users

Окрім інших атрибутів, записи User мають містити атрибут region, який може приймати одне з трьох значень region = EU, USA, Asia

1. Сконфігуруйте шардинг таким чином, щоб кожна з шард була проасоційована з одним з регіоном: Shard 1 - EU, Shard 2 - USA, Shard 3 - Asia

І записи які будуть вставлятись в колекцію Users мають зберігатись на відповідній шарді в залежності від значення region

1. Створіть колекцію Tweets

Серед інших атрибутів, Tweet має містити кількість лайків - атрибут likes

1. Сконфігуруйте шарди таким чином, щоб кожна з них була проасоційована з діапазоном кількості лайків (Ranged Sharding): Shard 1 - 0-100, Shard 2 - 101-200, Shard 3 - 200 - ...

І записи які будуть вставлятись в колекцію Tweets мають зберігатись на відповідній шарді в залежності від значення likes

1. Перевірте появу шард і зон командою sh.status()

```
shardingVersion
{
  _id: ObjectId('69431a48b58cf6b823272255'),
  clusterId: ObjectId('69431a48b58cf6b823272254')
}
---
shards
[
  {
    _id: 'rs_shard1',
    host: 'rs_shard1/shard1:27017',
    state: 1,
    topologyTime: Timestamp({ t: 1766005323, i: 32 }),
    replSetConfigVersion: Long('-1'),
    tags: [ 'ZONE_EU', 'LIKES_LOW' ]
  },
  {
    _id: 'rs_shard2',
    host: 'rs_shard2/shard2:27017',
    state: 1,
    topologyTime: Timestamp({ t: 1766005323, i: 89 }),
    replSetConfigVersion: Long('-1'),
    tags: [ 'ZONE_USA', 'LIKES_MID' ]
  }
]
```

```

},
{
  _id: 'rs_shard3',
  host: 'rs_shard3/shard3:27017',
  state: 1,
  topologyTime: Timestamp({ t: 1766005323, i: 144 }),
  replSetConfigVersion: Long('-1'),
  tags: [ 'ZONE_ASIA', 'LIKES_HIGH' ]
}
]
---
active mongoses
[ { '8.2.2': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Failed balancer rounds in last 5 attempts': 0,
  'Currently running': 'no',
  'Migration Results for the last 24 hours': 'No recent migrations'
}
---
shardedDataDistribution
[
  {
    ns: 'RLS.Users',
    shards: [
      {
        shardName: 'rs_shard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 0,
        ownedSizeBytes: 0,
        orphanedSizeBytes: 0
      }
    ]
  },
  {
    ns: 'RLS.Tweets',
    shards: [
      {
        shardName: 'rs_shard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 0,
        ownedSizeBytes: 0,
        orphanedSizeBytes: 0
      }
    ]
  }
]
---
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {}
  },
  {
    database: {
      _id: 'RLS',
      primary: 'rs_shard1',
      version: {
        uuid: UUID('38c823ff-2f01-4dd9-8cd3-9aa806cef7b3'),
        timestamp: Timestamp({ t: 1766005323, i: 172 }),
        lastMod: 1
      }
    },
    collections: {
      'RLS.Tweets': {
        shardKey: { likes: 1 },
        unique: false,
        balancing: true,
        allowMigrations: true,
        chunkMetadata: [ { shard: 'rs_shard1', nChunks: 1 } ],
        chunks: [
          { min: { likes: MinKey() }, max: { likes: MaxKey() }, 'on shard': 'rs_shard1', 'last modified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: [
    
```

```

        {
          tag: 'LIKES_LOW',
          min: { likes: 0 },
          max: { likes: 101 }
        },
        {
          tag: 'LIKES_MID',
          min: { likes: 101 },
          max: { likes: 201 }
        },
        {
          tag: 'LIKES_HIGH',
          min: { likes: 201 },
          max: { likes: MaxKey() }
        }
      ],
    },
    'RLS.Users': {
      shardKey: { region: 1, _id: 1 },
      unique: false,
      balancing: true,
      allowMigrations: true,
      chunkMetadata: [ { shard: 'rs_shard1', nChunks: 1 } ],
      chunks: [
        { min: { region: MinKey(), _id: MinKey() }, max: { region: MaxKey(), _id: MaxKey() }, 'on shard': 'rs_shard1', 'last modified': Timestamp({ t: 1, i: 0 }) }
      ],
      tags: [
        {
          tag: 'ZONE_ASIA',
          min: { region: 'Asia', _id: MinKey() },
          max: { region: 'Asia', _id: MaxKey() }
        },
        {
          tag: 'ZONE_EU',
          min: { region: 'EU', _id: MinKey() },
          max: { region: 'EU', _id: MaxKey() }
        },
        {
          tag: 'ZONE_USA',
          min: { region: 'USA', _id: MinKey() },
          max: { region: 'USA', _id: MaxKey() }
        }
      ]
    }
  ]
}

```

## 2. Продемонструйте роботу шардінгу (тобто що записи зберігаються на різних нодах):

```

print(database.Users.getShardDistribution());
print(database.Users.countDocuments());

Shard rs_shard3 at rs_shard3/shard3:27017
{
  data: '236B',
  docs: 4,
  chunks: 1,
  'estimated data per chunk': '236B',
  'estimated docs per chunk': 4
}

---
Shard rs_shard2 at rs_shard2/shard2:27017
{
  data: '141KiB',
  docs: 2500,
  chunks: 1,
  'estimated data per chunk': '141KiB',
  'estimated docs per chunk': 2500
}

---
Shard rs_shard1 at rs_shard1/shard1:27017
{
  data: '139KiB',
  docs: 3255,
  chunks: 5,
  'estimated data per chunk': '27KiB',
  'estimated docs per chunk': 651
}

```

```

---  

Totals  

{  

  data: '280KiB',  

  docs: 5759,  

  chunks: 7,  

  'Shard rs_shard3': [  

    '0.08 % data',  

    '0.06 % docs in cluster',  

    '59B avg obj size on shard'  

  ],  

  'Shard rs_shard2': [  

    '50.22 % data',  

    '43.41 % docs in cluster',  

    '57B avg obj size on shard'  

  ],  

  'Shard rs_shard1': [  

    '49.69 % data',  

    '56.52 % docs in cluster',  

    '56B avg obj size on shard'  

  ]  

}  

5004

```

- відключити одну з ноди

```
docker stop shard2
```

- спробувати додати записи зі значеннями shard key (Ranged та Zones), що потрапляють на відключенну ноду

```
[TEST] INSERT in zone USA (Shard 2 - Down)  

[ERROR] Write results unavailable from failing to target a host in the shard  

       rs_shard2 :: caused by :: Could not find host matching read preference {  

         mode: "primary" } for set rs_shard2
```

- спробувати додати записи зі значеннями shard key (Ranged та Zones), що потрапляють на працючу ноду

```
[TEST] INSERT in zone EU (Shard 1 - Up)  

[SUCCESS] (5ms): {"acknowledged":true,"insertedId":"69433156d8d1a447a29dc29e"}
```

- спробувати знайти всі записи з shard key для Zone, яка відповідає ноді яка працює/яка не працює

```
# Воно думає, що щось є на rs_shard2, тому падає повністю...  

  

[TEST] FIND in zone USA (Shard 2 - Down)  

[ERROR] Could not find host matching read preference { mode: "primary" } for  

       set rs_shard2  

[TEST] FIND in zone EU (Shard 1 - Up)  

[ERROR] Could not find host matching read preference { mode: "primary" } for  

       set rs_shard2
```

- спробувати знайти записи для shard key з певного проміжку, який входить до проміжку працюючої ноди для Ranged Sharding

```
# Воно думає, що щось є на rs_shard2, тому падає повністю...
[TEST] RANGE FIND Likes 0-100 (Shard 1 - Up)
[ERROR] Could not find host matching read preference { mode: "primary" } for
        set rs_shard2
```

- спробувати знайти записи для shard key з певного проміжку, який входить до проміжку ноди яка не працює для Ranged Sharding

```
[TEST] RANGE FIND Likes 101-200 (Shard 2 - Down)
[ERROR] Could not find host matching read preference { mode: "primary" } for
        set rs_shard2
```

### 3. Включити відключену ноду та перевірити працездатність запитів з попереднього пункту

```
docker start shard2

[TEST] INSERT in zone USA (Shard 2 - Down)
[SUCCESS] (1378ms): {"acknowledged":true,"insertedId":"694333aaf203c33bc09dc29d"}
[TEST] INSERT in zone EU (Shard 1 - Up)
[SUCCESS] (5ms): {"acknowledged":true,"insertedId":"694333acf203c33bc09dc29e"}
[TEST] FIND in zone USA (Shard 2 - Down)
[SUCCESS] (3ms): [{"_id":"694333aaf203c33bc09dc29d","name":"DeadUser","region":"USA"}]
[TEST] FIND in zone EU (Shard 1 - Up)
[SUCCESS] (3ms): [{"_id":"6943336d9bc5bf5c299dc29e","name":"LiveUser","region":"EU"}, {"_id":"694333acf203c33bc09dc29e","name":"LiveUser","region":"EU"}]
[TEST] RANGE FIND Likes 101-200 (Shard 2 - Down)
[SUCCESS] (11ms): [{"_id":"69433351712c168cf9dc2a0","msg":"Mid1","likes":105}, {"_id":"69433351712c168cf9dc2a1","msg":"Mid2","likes":150}, {"_id":"69433351712c168cf9dc2a2","msg":"Mid3","likes":199}]
[TEST] RANGE FIND Likes 0-100 (Shard 1 - Up)
[SUCCESS] (2ms): [{"_id":"69433351712c168cf9dc29d","msg":"Low1","likes":5}, {"_id":"69433351712c168cf9dc29e","msg":"Low2","likes":50}, {"_id":"69433351712c168cf9dc29f","msg":"Low3","likes":99}]
```

Всі завдання можуть бути виконані з командного рядка

Протокол має містити команди та кінцеві налаштування з sh.status()

```
shardingVersion
{
  _id: ObjectId('694330f36bea9194f99ed960'),
  clusterId: ObjectId('694330f36bea9194f99ed95f')
}
---
shards
[
  {
    _id: 'rs_shard1',
    host: 'rs_shard1/shard1:27017',
    state: 1,
    topologyTime: Timestamp({ t: 1766011126, i: 32 }),
    replSetConfigVersion: Long('1'),
    tags: [ 'ZONE_EU', 'LIKES_LOW' ]
  },
  {
    _id: 'rs_shard2',
```

```

host: 'rs_shard2/shard2:27017',
state: 1,
topologyTime: Timestamp({ t: 1766011126, i: 89 }),
replicaSetConfigVersion: Long('1'),
tags: [ 'ZONE_USA', 'LIKES_MID' ]
},
{
  _id: 'rs_shard3',
  host: 'rs_shard3/shard3:27017',
  state: 1,
  topologyTime: Timestamp({ t: 1766011126, i: 144 }),
  replicaSetConfigVersion: Long('1'),
  tags: [ 'ZONE_ASIA', 'LIKES_HIGH' ]
}
]
---
active mongoses
[ { '8.2.2': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Failed balancer rounds in last 5 attempts': 0,
  'Currently running': 'no',
  'Migration Results for the last 24 hours': { '36': 'Success' }
}
---
shardedDataDistribution
[
  {
    ns: 'config.system.sessions',
    shards: [
      {
        shardName: 'rs_shard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 142,
        ownedSizeBytes: 14058,
        orphanedSizeBytes: 0
      }
    ]
  },
  {
    ns: 'RLS.Tweets',
    shards: [
      {
        shardName: 'rs_shard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 3,
        ownedSizeBytes: 141,
        orphanedSizeBytes: 0
      },
      {
        shardName: 'rs_shard3',
        numOrphanedDocs: 0,
        numOwnedDocuments: 3,
        ownedSizeBytes: 144,
        orphanedSizeBytes: 0
      },
      {
        shardName: 'rs_shard2',
        numOrphanedDocs: 6,
        numOwnedDocuments: 3,
        ownedSizeBytes: 141,
        orphanedSizeBytes: 282
      }
    ]
  },
  {
    ns: 'RLS.Users',
    shards: [
      {
        shardName: 'rs_shard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 2,
        ownedSizeBytes: 112,
        orphanedSizeBytes: 0
      },
      {
        shardName: 'rs_shard3',
        numOrphanedDocs: 0,
        numOwnedDocuments: 2,
        ownedSizeBytes: 112,
        orphanedSizeBytes: 0
      }
    ]
  }
]

```

```

        shardName: 'rs_shard3',
        numOrphanedDocs: 0,
        numOwnedDocuments: 0,
        ownedSizeBytes: 0,
        orphanedSizeBytes: 0
    },
    {
        shardName: 'rs_shard2',
        numOrphanedDocs: 2048,
        numOwnedDocuments: 1,
        ownedSizeBytes: 56,
        orphanedSizeBytes: 114688
    }
]
]
---  

databases  

[  

{
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
        'config.system.sessions': {
            shardKey: { _id: 1 },
            unique: false,
            balancing: true,
            allowMigrations: true,
            chunkMetadata: [ { shard: 'rs_shard1', nChunks: 1 } ],
            chunks: [
                { min: { _id: MinKey() }, max: { _id: MaxKey() }, 'on shard': 'rs_shard1', 'last modified': Timestamp({ t: 1, i: 0 }) }
            ],
            tags: []
        }
    }
},
{
    database: {
        _id: 'RLS',
        primary: 'rs_shard2',
        version: {
            uuid: UUID('ad96755e-8b05-4b0e-b8e2-096840b04e3e'),
            timestamp: Timestamp({ t: 1766011126, i: 172 }),
            lastMod: 1
        }
    },
    collections: {
        'RLS.Tweets': {
            shardKey: { likes: 1 },
            unique: false,
            balancing: true,
            allowMigrations: true,
            chunkMetadata: [
                { shard: 'rs_shard1', nChunks: 1 },
                { shard: 'rs_shard2', nChunks: 2 },
                { shard: 'rs_shard3', nChunks: 1 }
            ],
            chunks: [
                { min: { likes: MinKey() }, max: { likes: 0 }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 3, i: 1 }) },
                { min: { likes: 0 }, max: { likes: 101 }, 'on shard': 'rs_shard1', 'last modified': Timestamp({ t: 2, i: 0 }) },
                { min: { likes: 101 }, max: { likes: 201 }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 1, i: 3 }) },
                { min: { likes: 201 }, max: { likes: MaxKey() }, 'on shard': 'rs_shard3', 'last modified': Timestamp({ t: 3, i: 0 }) }
            ],
            tags: [
                { tag: 'LIKES_LOW', min: { likes: 0 }, max: { likes: 101 } },
                { tag: 'LIKES_MID',
                    min: { likes: 101 },
                    max: { likes: 201 }
                },
                { tag: 'LIKES_HIGH',
                    min: { likes: 201 },
                    max: { likes: MaxKey() }
                }
            ]
        }
    }
}
]

```

```

'RLS.Users': {
    shardKey: { region: 1, _id: 1 },
    unique: false,
    balancing: true,
    allowMigrations: true,
    chunkMetadata: [
        { shard: 'rs_shard1', nChunks: 1 },
        { shard: 'rs_shard2', nChunks: 5 },
        { shard: 'rs_shard3', nChunks: 1 }
    ],
    chunks: [
        { min: { region: MinKey(), _id: MinKey() }, max: { region: 'Asia', _id: MinKey() }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 3, i: 1 }) },
        { min: { region: 'Asia', _id: MinKey() }, max: { region: 'Asia', _id: MaxKey() }, 'on shard': 'rs_shard3', 'last modified': Timestamp({ t: 2, i: 0 }) },
        { min: { region: 'Asia', _id: MaxKey() }, max: { region: 'EU', _id: MinKey() }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 1, i: 3 }) },
        { min: { region: 'EU', _id: MinKey() }, max: { region: 'EU', _id: MaxKey() }, 'on shard': 'rs_shard1', 'last modified': Timestamp({ t: 3, i: 0 }) },
        { min: { region: 'EU', _id: MaxKey() }, max: { region: 'USA', _id: MinKey() }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 1, i: 5 }) },
        { min: { region: 'USA', _id: MinKey() }, max: { region: 'USA', _id: MaxKey() }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 1, i: 6 }) },
        { min: { region: 'USA', _id: MaxKey() }, max: { region: MaxKey(), _id: MaxKey() }, 'on shard': 'rs_shard2', 'last modified': Timestamp({ t: 1, i: 7 }) }
    ],
    tags: [
        {
            tag: 'ZONE_ASIA',
            min: { region: 'Asia', _id: MinKey() },
            max: { region: 'Asia', _id: MaxKey() }
        },
        {
            tag: 'ZONE_EU',
            min: { region: 'EU', _id: MinKey() },
            max: { region: 'EU', _id: MaxKey() }
        },
        {
            tag: 'ZONE_USA',
            min: { region: 'USA', _id: MinKey() },
            max: { region: 'USA', _id: MaxKey() }
        }
    ]
},
{
    database: {
        _id: 'test',
        primary: 'rs_shard3',
        version: {
            uid: UUID('4da15db4-7310-490d-89ce-db43660d5704'),
            timestamp: Timestamp({ t: 1766012174, i: 7 }),
            lastMod: 1
        }
    },
    collections: {}
}
]

```