Actividad 3 - Conceptos y comandos básicos de la replicación en bases de datos NoSQL

Brayan Steven Bonilla Castellanos

Juan Carlos Monsalve Gómez

Corporación Universitaria Iberoamericana

Ingeniería de Software

Bases de datos avanzadas

Requerimientos no funcionales

- La base de datos debe permitir realizar todas las operaciones de consulta necesarias.
- La base de datos debe estar disponible 24/7
- Se debe garantizar la seguridad de la información.
- Debe permitir el acceso de varios usuarios al mismo tiempo (concurrencia)
- Se debe garantizar la fiabilidad de la información.
- Debe permitir la escalabilidad horizontal para el crecimiento del negocio.

Enlace Repositorio GIT

https://github.com/jcmonsalveg/Actividad-3---ReplicacionNoSQL

Enlace Video

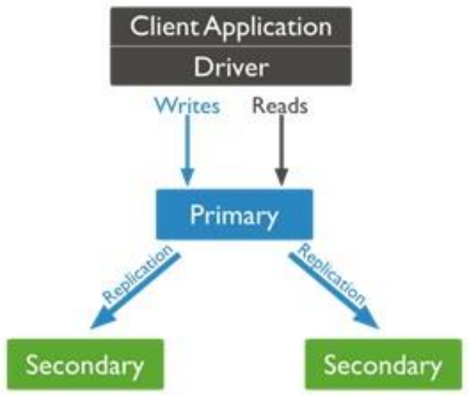
https://youtu.be/B2pZj4jVjT0

Réplicas DB

La replicación es la acción de sincronizar la información de una base de datos a través de servidores múltiples, de tal manera que pueda garantizar la redundancia y la disponibilidad de los datos.

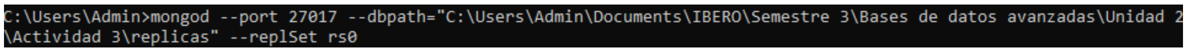
En el conjunto de réplicas existen varios nodos, de los cuales hay uno primario o maestro y varios esclavos o secundarios. Dentro del conjunto de nodos se puede tener un árbitro, este es un nodo que participa en las elecciones para la elección del nodo primario frente a una posible caída del nodo primario. la configuración mínima de una réplica establecida pide, entonces, un nodo primario y dos secundarios.

Entre los nodos creados existe una comunicación constante en la cual se verifica la existencia de los mismos (heartbeat). Frente a una caída de un servidor (donde reposa uno de los nodos), los demás nodos garantizan que no existe pérdida de la información, por ejemplo si se cae el nodo primario sucede una elección entre los nodos existentes y uno de los que quedan vivos asumir el nodo de primaria. Cuando el nodo caído se levante, se levantará como nodo secundario, ocurrirá una resincronización y se estabilizará el servicio.



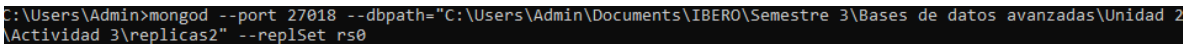
Réplica MongoDB

Creamos el Nodo 1 (Debe ser el principal, puerto 27017)



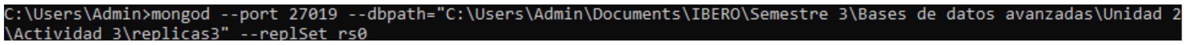
```
C:\Users\Admin>mongod --port 27017 --dbpath="C:\Users\Admin\Documents\IBERO\Semestre 3\Bases de datos avanzadas\Unidad 2
\Actividad 3\replicas" --replSet rs0
```

Creamos el Nodo 2



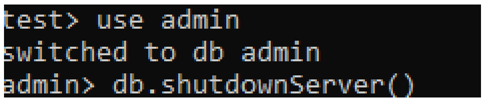
```
C:\Users\Admin>mongod --port 27018 --dbpath="C:\Users\Admin\Documents\IBERO\Semestre 3\Bases de datos avanzadas\Unidad 2
\Actividad 3\replicas2" --replSet rs0
```

Creamos el Nodo 3



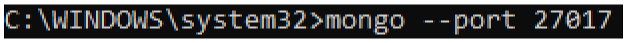
```
C:\Users\Admin>mongod --port 27019 --dbpath="C:\Users\Admin\Documents\IBERO\Semestre 3\Bases de datos avanzadas\Unidad 2
\Actividad 3\replicas3" --replSet rs0
```

Cerramos las conexiones



```
test> use admin
switched to db admin
admin> db.shutdownServer()
```

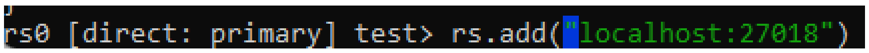
Nos conectamos al Nodo principal desde la terminal de Mongosh



```
C:\WINDOWS\system32>mongo --port 27017
```

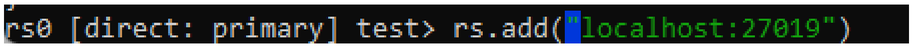
Ejecutamos el comando
rs.initiate()

Agregamos el nodo 2 al nodo primario



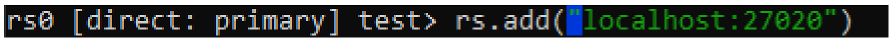
```
rs0 [direct: primary] test> rs.add("localhost:27018")
```

Agregamos el nodo 3 al nodo primario



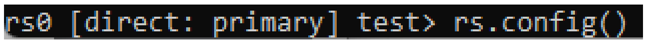
```
rs0 [direct: primary] test> rs.add("localhost:27019")
```

Agregamos el nodo 4 al nodo primario

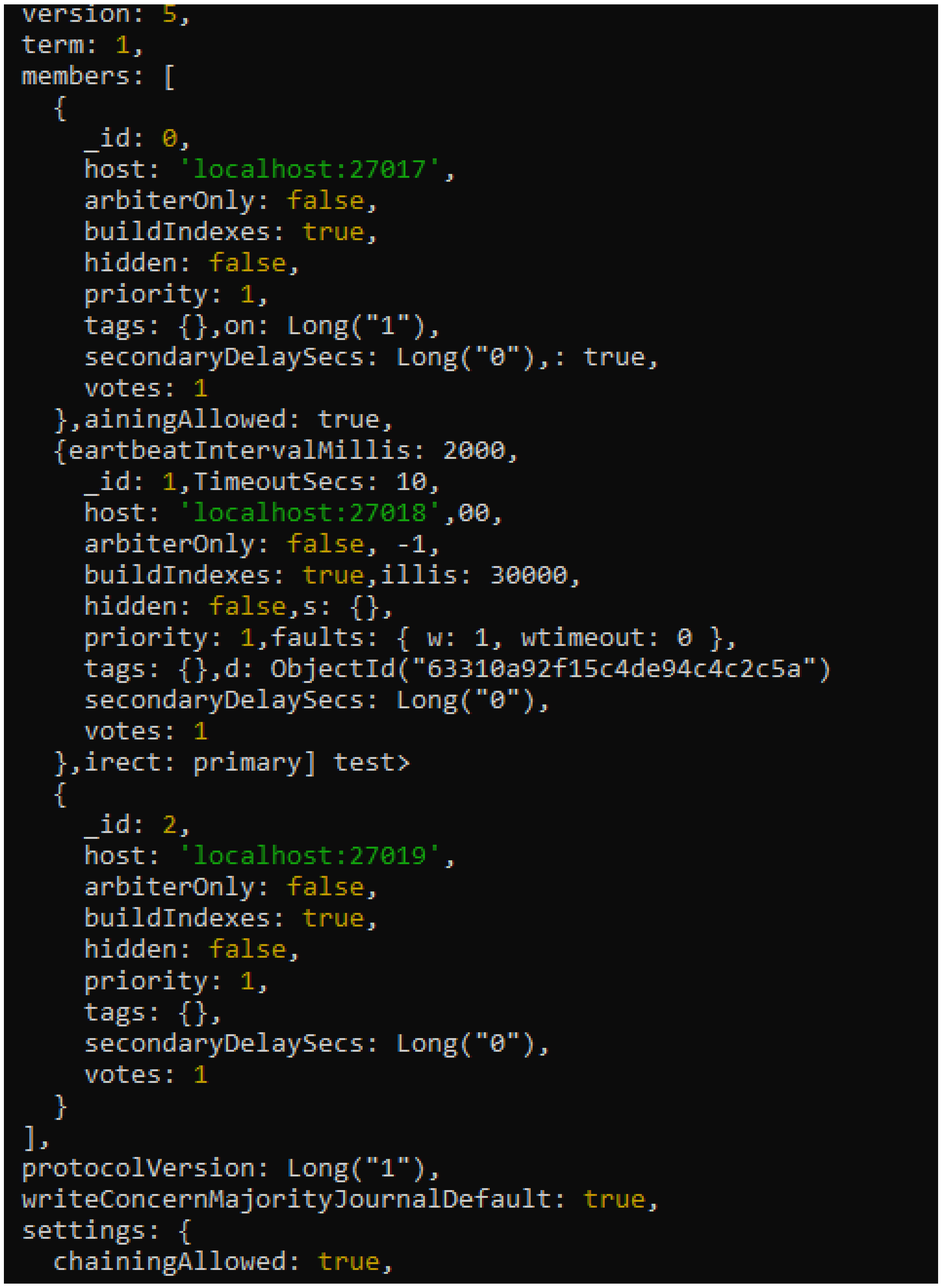


```
rs0 [direct: primary] test> rs.add("localhost:27020")
```

Verificamos la configuración

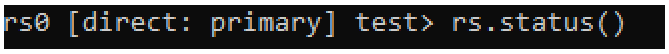


```
rs0 [direct: primary] test> rs.config()
```

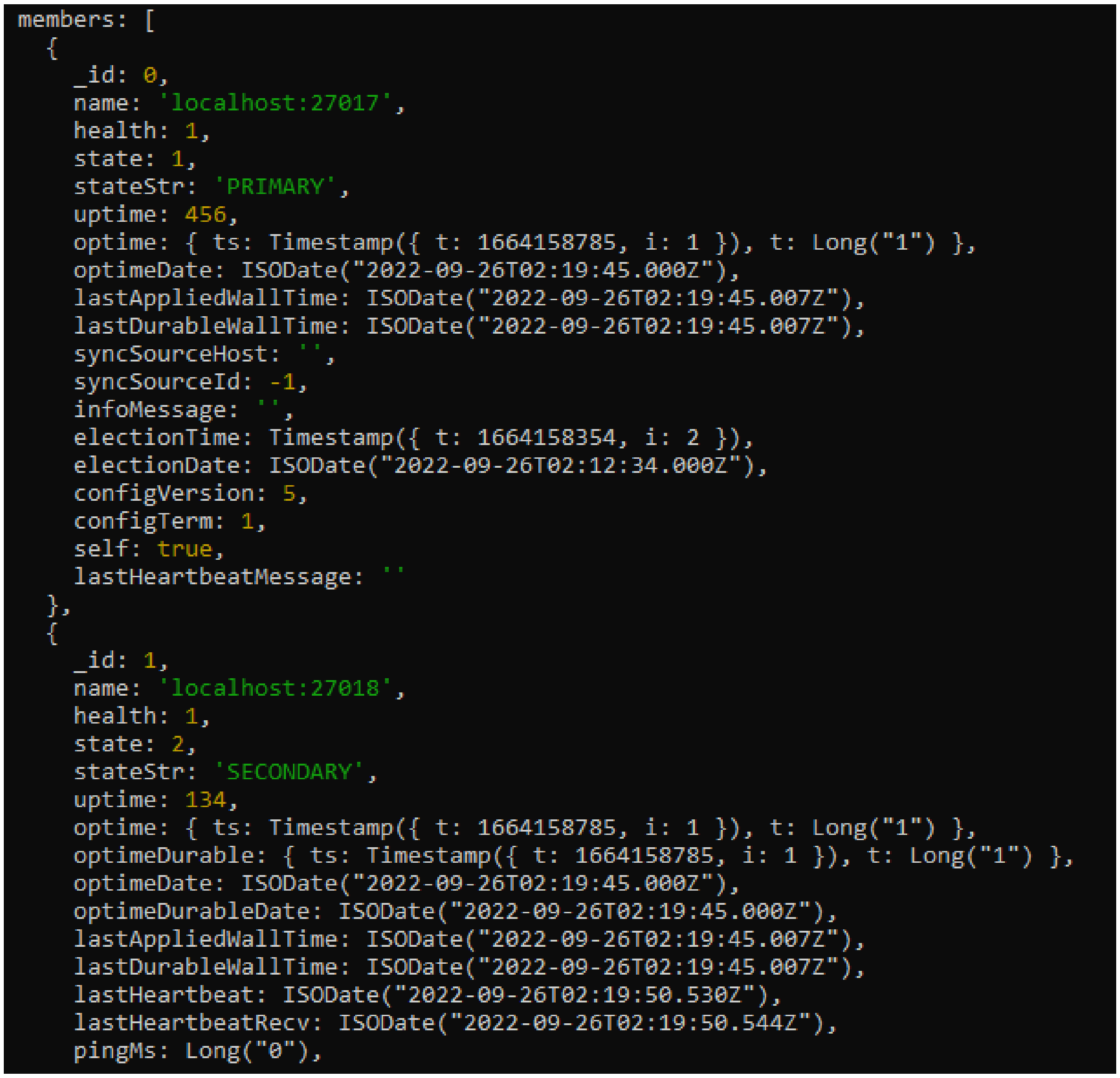


```
  version: 5,
  term: 1,
  members: [
    {
      _id: 0,
      host: 'localhost:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},on: Long("1"),
      secondaryDelaySecs: Long("0"),: true,
      votes: 1
    },ainingAllowed: true,
    {eartbeatIntervalMillis: 2000,
      _id: 1,TimeoutSecs: 10,
      host: 'localhost:27018',00,
      arbiterOnly: false, -1,
      buildIndexes: true,illis: 30000,
      hidden: false,s: {},
      priority: 1,faults: { w: 1, wtimeout: 0 },
      tags: {},d: ObjectId("63310a92f15c4de94c4c2c5a")
      secondaryDelaySecs: Long("0"),
      votes: 1
    },irect: primary] test>
    {
      _id: 2,
      host: 'localhost:27019',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    }
  ],
  protocolVersion: Long("1"),
  writeConcernMajorityJournalDefault: true,
  settings: {
    chainingAllowed: true,
```

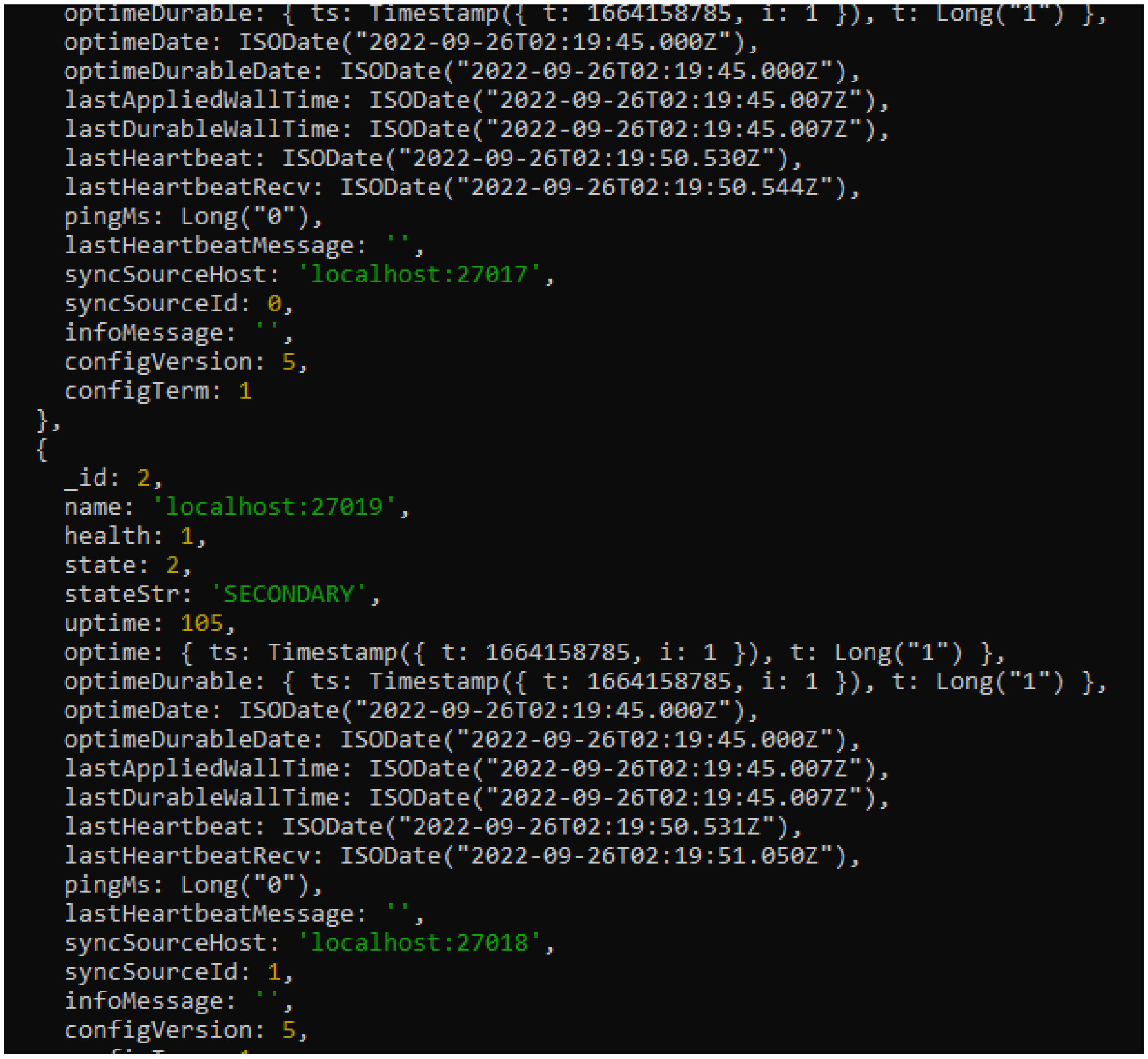
Verificamos el estado



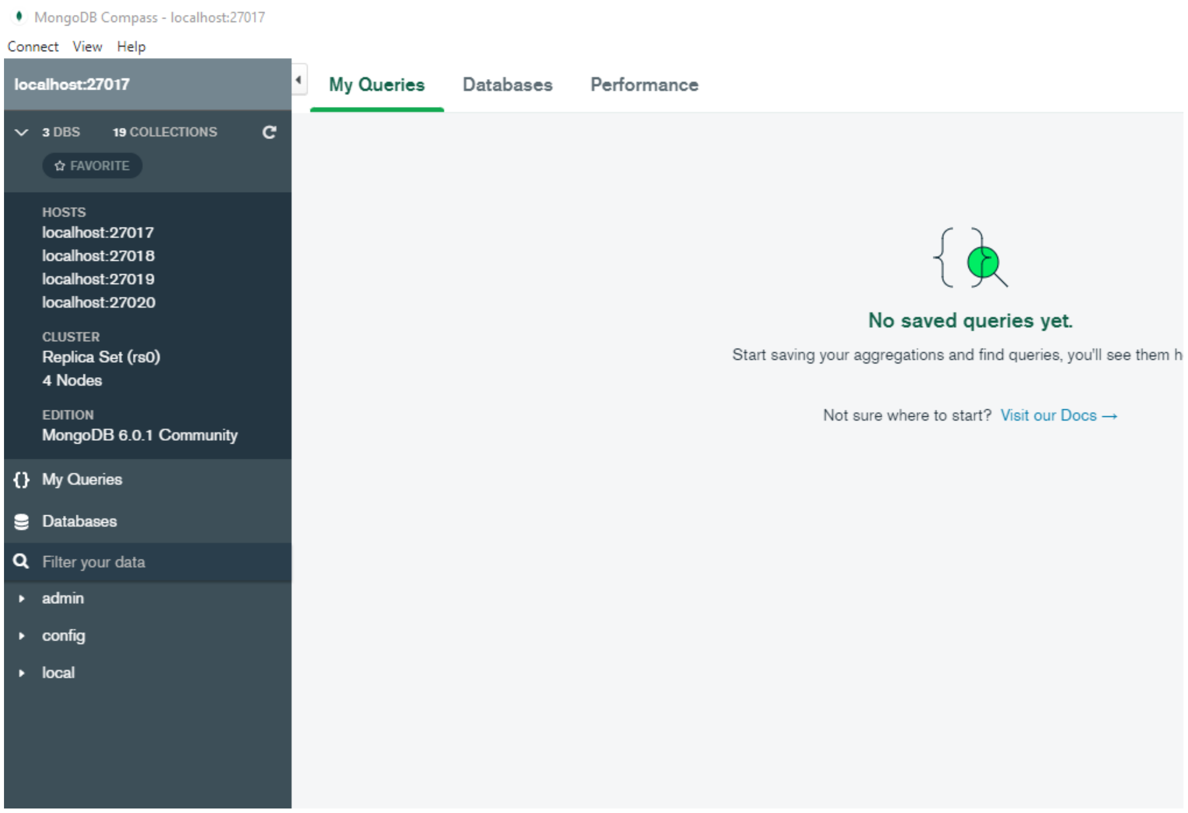
```
rs0 [direct: primary] test> rs.status()
```



```
members: [
  {
    _id: 0,
    name: 'localhost:27017',
    health: 1,
    state: 1,
    stateStr: 'PRIMARY',
    uptime: 456,
    optime: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDate: ISODate("2022-09-26T02:19:45.000Z"),
    lastAppliedWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastDurableWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    electionTime: Timestamp({ t: 1664158354, i: 2 }),
    electionDate: ISODate("2022-09-26T02:12:34.000Z"),
    configVersion: 5,
    configTerm: 1,
    self: true,
    lastHeartbeatMessage: ''
  },
  {
    _id: 1,
    name: 'localhost:27018',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 134,
    optime: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDurable: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDate: ISODate("2022-09-26T02:19:45.000Z"),
    optimeDurableDate: ISODate("2022-09-26T02:19:45.000Z"),
    lastAppliedWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastDurableWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastHeartbeat: ISODate("2022-09-26T02:19:50.530Z"),
    lastHeartbeatRecv: ISODate("2022-09-26T02:19:50.544Z"),
    pingMs: Long("0"),
```


```
    optimeDurable: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDate: ISODate("2022-09-26T02:19:45.000Z"),
    optimeDurableDate: ISODate("2022-09-26T02:19:45.000Z"),
    lastAppliedWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastDurableWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastHeartbeat: ISODate("2022-09-26T02:19:50.530Z"),
    lastHeartbeatRecv: ISODate("2022-09-26T02:19:50.544Z"),
    pingMs: Long("0"),
    lastHeartbeatMessage: '',
    syncSourceHost: 'localhost:27017',
    syncSourceId: 0,
    infoMessage: '',
    configVersion: 5,
    configTerm: 1
  },
  {
    _id: 2,
    name: 'localhost:27019',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 105,
    optime: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDurable: { ts: Timestamp({ t: 1664158785, i: 1 }), t: Long("1") },
    optimeDate: ISODate("2022-09-26T02:19:45.000Z"),
    optimeDurableDate: ISODate("2022-09-26T02:19:45.000Z"),
    lastAppliedWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastDurableWallTime: ISODate("2022-09-26T02:19:45.007Z"),
    lastHeartbeat: ISODate("2022-09-26T02:19:50.531Z"),
    lastHeartbeatRecv: ISODate("2022-09-26T02:19:51.050Z"),
    pingMs: Long("0"),
    lastHeartbeatMessage: '',
    syncSourceHost: 'localhost:27018',
    syncSourceId: 1,
    infoMessage: '',
    configVersion: 5,
```

Verificamos la creación correcta de los nodos en la instancia principal



MongoDB Compass - localhost:27017

Connect View Help

localhost:27017

3 DBS 19 COLLECTIONS

FAVORITE

HOSTS
localhost:27017
localhost:27018
localhost:27019
localhost:27020

CLUSTER
Replica Set (rs0)
4 Nodes

EDITION
MongoDB 6.0.1 Community

My Queries
Databases
Filter your data
admin
config
local

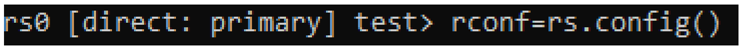
My Queries | Databases | Performance

No saved queries yet.

Start saving your aggregations and find queries, you'll see them h

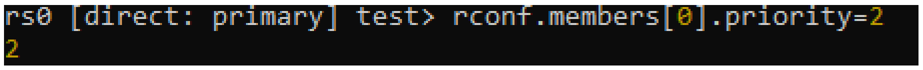
Not sure where to start? Visit our Docs →

Asignamos la configuración a una variable



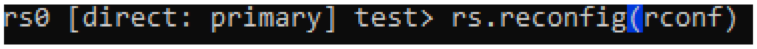
```
rs0 [direct: primary] test> rconf=rs.config()
```

Cambiamos la prioridad al nodo principal para tolerancia a fallos pueda ser instanciado de nuevo como principal



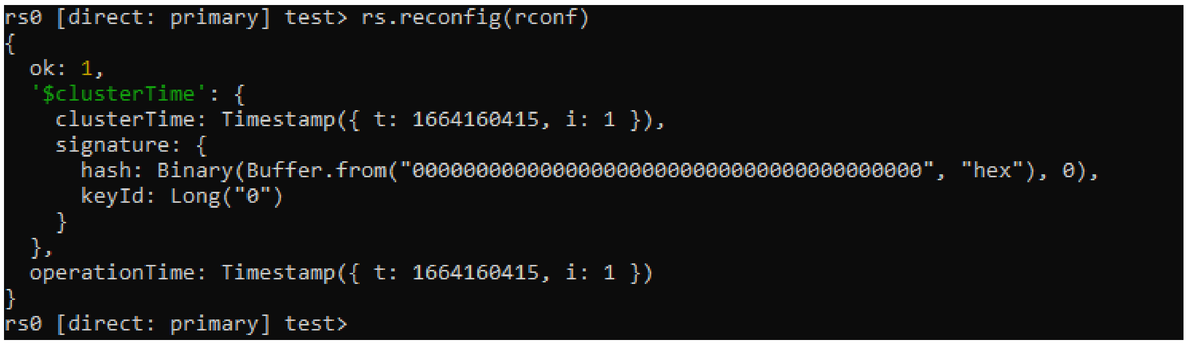
```
rs0 [direct: primary] test> rconf.members[0].priority=2
2
```

Sobrescribimos la configuración

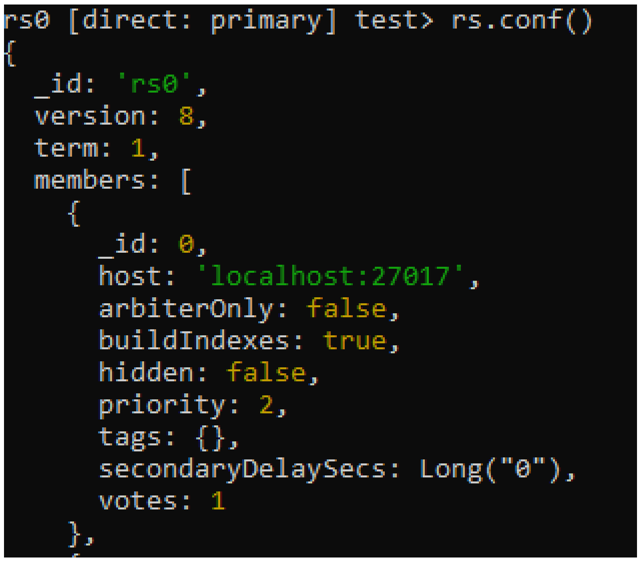


```
rs0 [direct: primary] test> rs.reconfig(rconf)
```

Verificamos los cambios realizados se apliquen de forma correcta

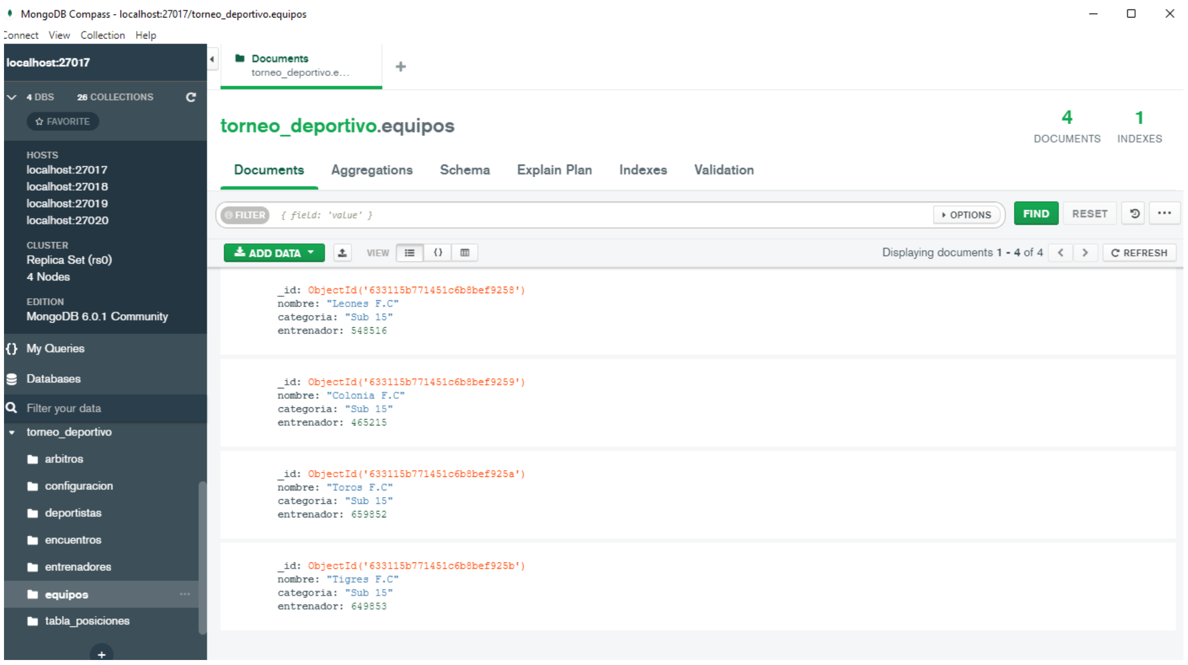


```
rs0 [direct: primary] test> rs.reconfig(rconf)
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1664160415, i: 1 }),
    signature: {
      hash: Binary(Buffer.from("0000000000000000000000000000000000000000", "hex"), 0),
      keyId: Long("0")
    }
  },
  operationTime: Timestamp({ t: 1664160415, i: 1 })
}
rs0 [direct: primary] test>
```



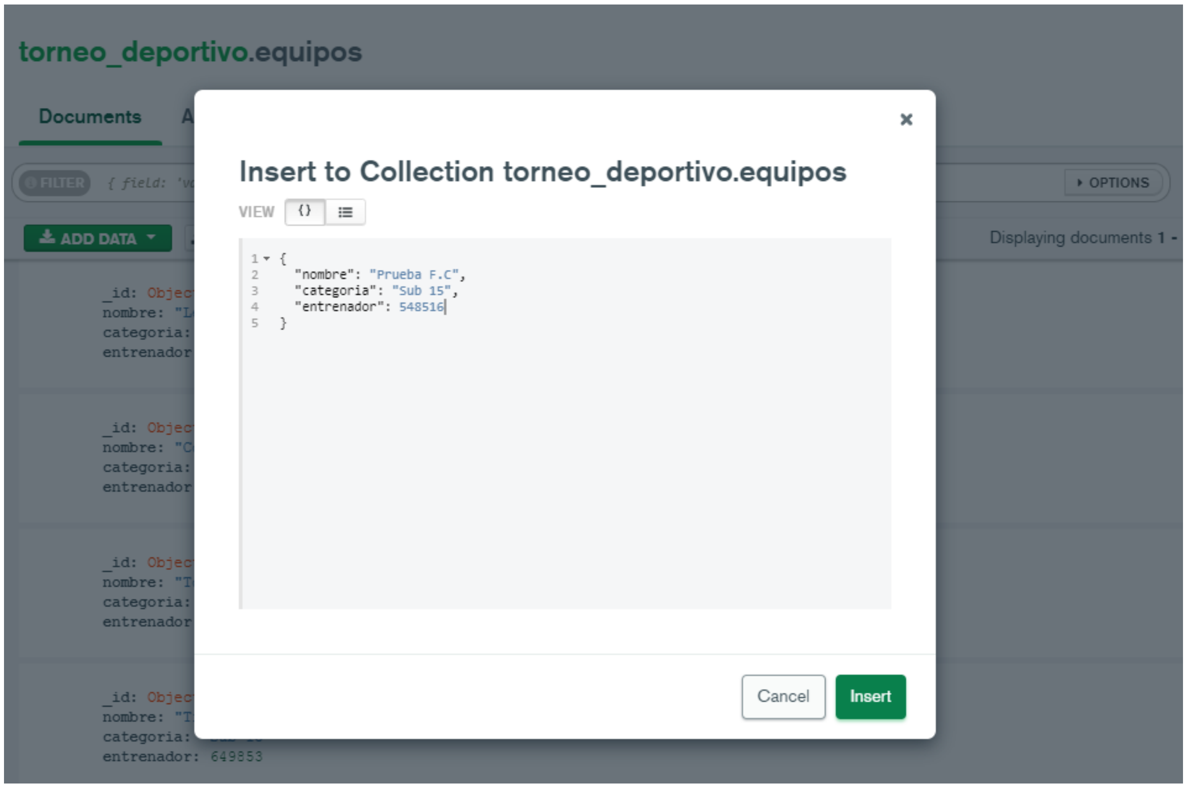
```
rs0 [direct: primary] test> rs.conf()
{
  _id: 'rs0',
  version: 8,
  term: 1,
  members: [
    {
      _id: 0,
      host: 'localhost:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 2,
      tags: {},
      secondaryDelaySecs: Long("0"),
      votes: 1
    },
```

Consultamos los equipos



```
_id: ObjectId('633115b771451c6b8bef9258')
nombre: "Leones F.C"
categoria: "Sub 15"
entrenador: 548516

_id: ObjectId('633115b771451c6b8bef9259')
nombre: "Colonia F.C"
categoria: "Sub 15"
entrenador: 465215

_id: ObjectId('633115b771451c6b8bef925a')
nombre: "Toros F.C"
categoria: "Sub 15"
entrenador: 659852

_id: ObjectId('633115b771451c6b8bef925b')
nombre: "Tigres F.C"
categoria: "Sub 15"
entrenador: 649853
```

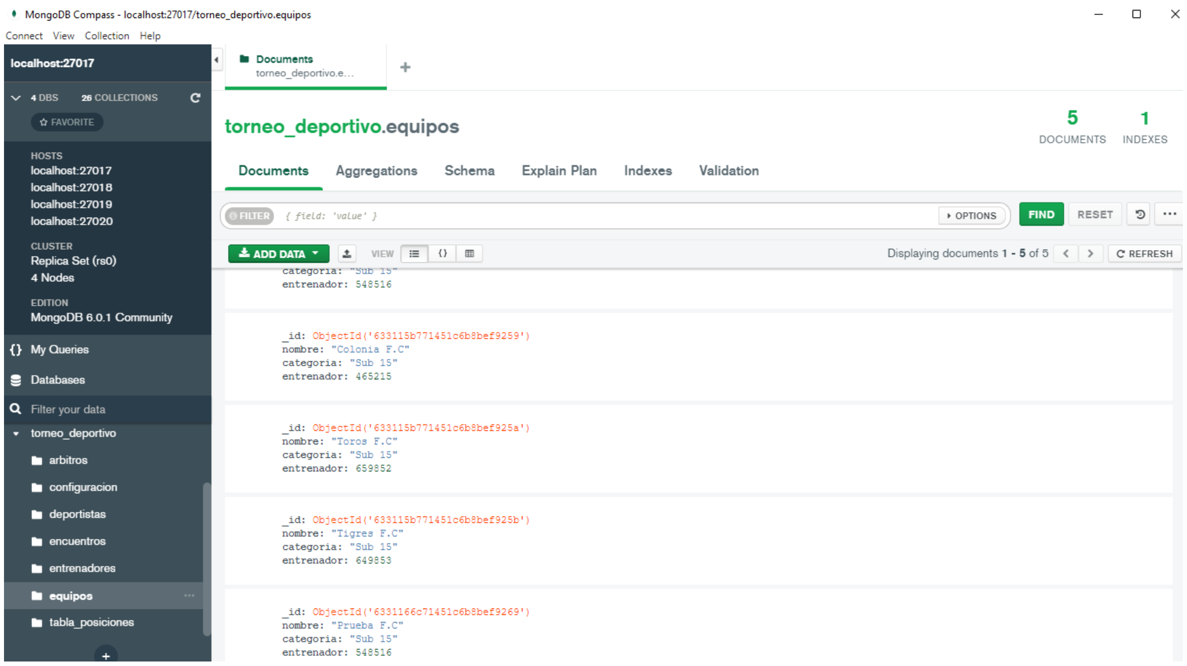
Insertamos un nuevo equipo



Insert to Collection torneo_deportivo.equipos

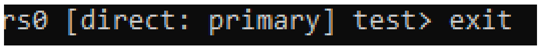
```
{
  "nombre": "Prueba F.C",
  "categoria": "Sub 15",
  "entrenador": 548516
}
```

Ahora tenemos 5 registros



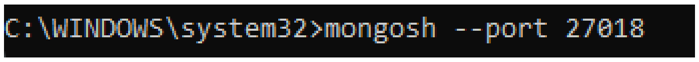
```
categoria: "Sub 15"
entrenador: 548516

_id: ObjectId('633115b771451c6b8bef9259')
nombre: "Colonia F.C"
categoria: "Sub 15"
entrenador: 465215

_id: ObjectId('633115b771451c6b8bef925a')
nombre: "Toros F.C"
categoria: "Sub 15"
entrenador: 659852

_id: ObjectId('633115b771451c6b8bef925b')
nombre: "Tigres F.C"
categoria: "Sub 15"
entrenador: 649853

_id: ObjectId('6331166c71451c6b8bef9269')
nombre: "Prueba F.C"
categoria: "Sub 15"
entrenador: 548516
```

Finalizamos la conexión al nodo principal

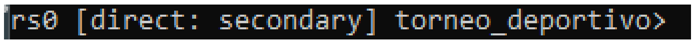


```
rs0 [direct: primary] test> exit
```

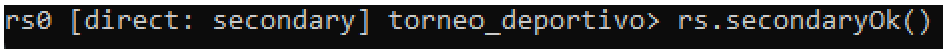
Nos conectamos al nodo secundario


```
C:\WINDOWS\system32>mongosh --port 27018
```

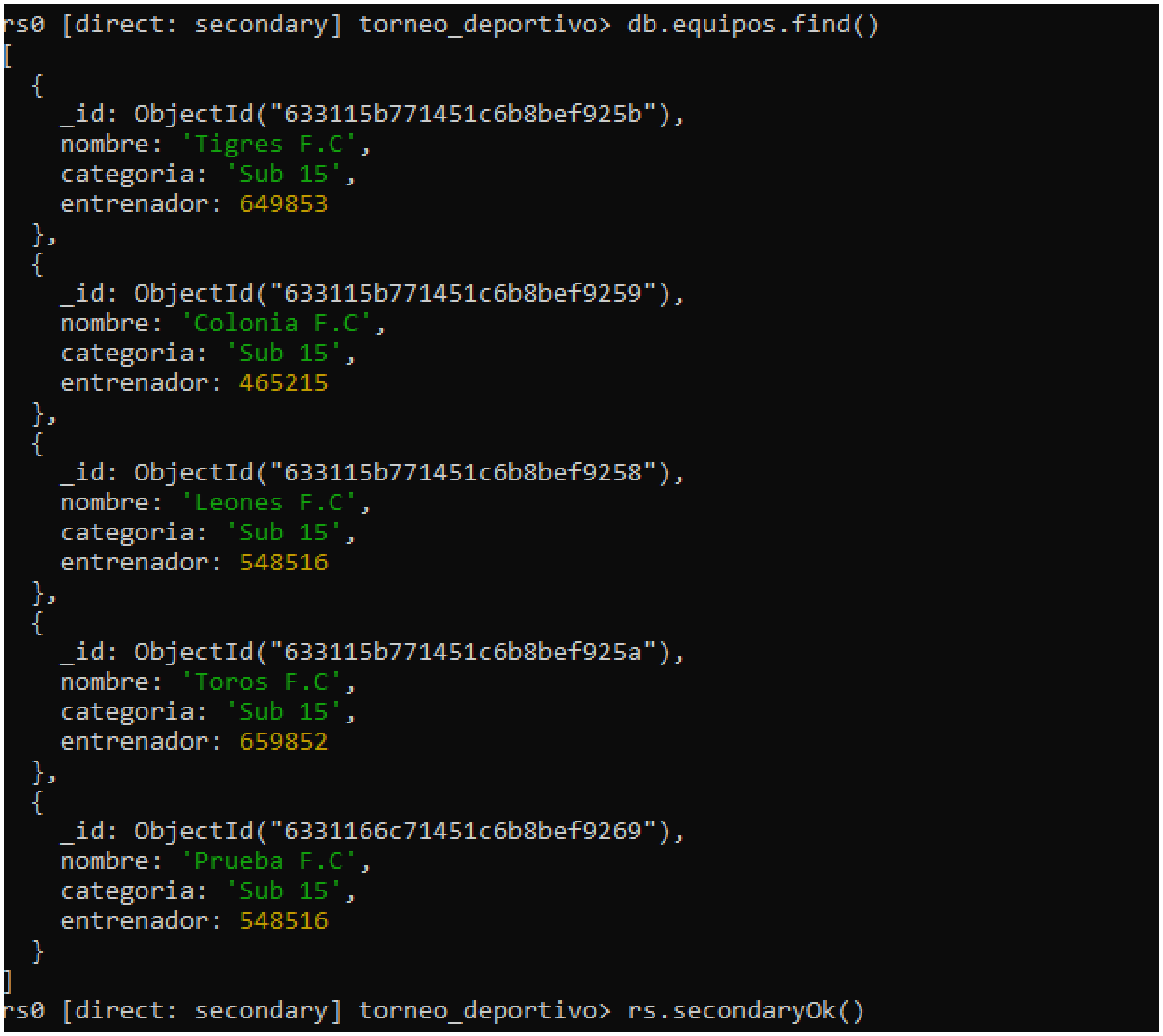
Seleccionamos la base de datos


```
rs0 [direct: secondary] torneo_deportivo>
```

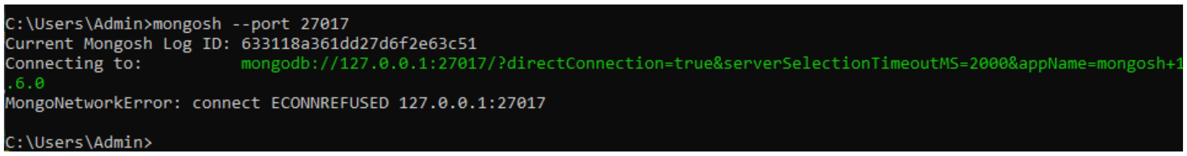
Asignamos permisos de lectura para el usuario


```
rs0 [direct: secondary] torneo_deportivo> rs.secondaryOk()
```

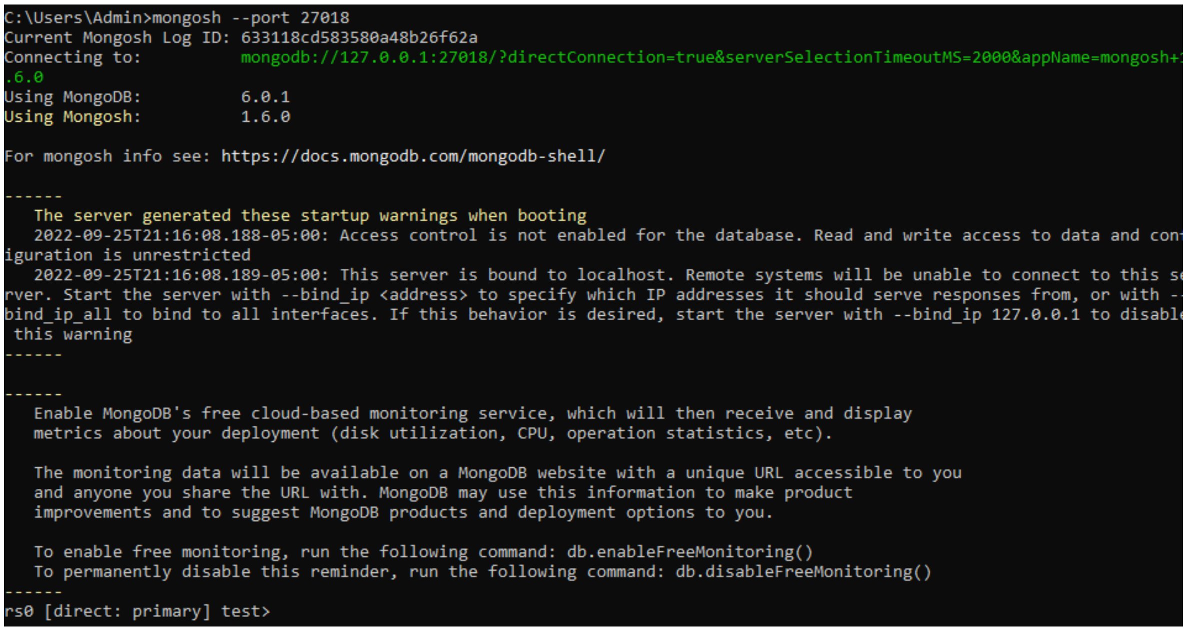
Realizamos la consulta para verificar la réplica de la información del nodo principal


```
rs0 [direct: secondary] torneo_deportivo> db.equipos.find()
[
  {
    _id: ObjectId("633115b771451c6b8bef925b"),
    nombre: 'Tigres F.C',
    categoria: 'Sub 15',
    entrenador: 649853
  },
  {
    _id: ObjectId("633115b771451c6b8bef9259"),
    nombre: 'Colonia F.C',
    categoria: 'Sub 15',
    entrenador: 465215
  },
  {
    _id: ObjectId("633115b771451c6b8bef9258"),
    nombre: 'Leones F.C',
    categoria: 'Sub 15',
    entrenador: 548516
  },
  {
    _id: ObjectId("633115b771451c6b8bef925a"),
    nombre: 'Toros F.C',
    categoria: 'Sub 15',
    entrenador: 659852
  },
  {
    _id: ObjectId("6331166c71451c6b8bef9269"),
    nombre: 'Prueba F.C',
    categoria: 'Sub 15',
    entrenador: 548516
  }
]
rs0 [direct: secondary] torneo_deportivo> rs.secondaryOk()
```

Cerramos la conexión del nodo principal

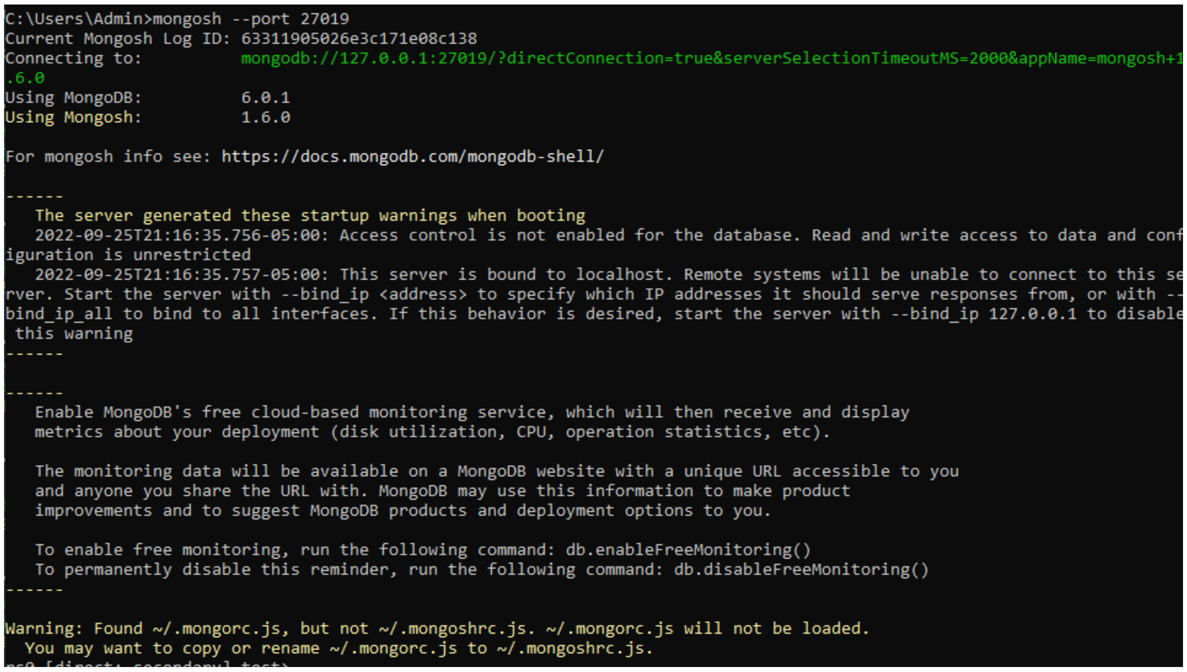

```
C:\Users\Admin>mongosh --port 27017
Current Mongosh Log ID: 633118a361dd27d6f2e63c51
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
.6.0
MongoNetworkError: connect ECONNREFUSED 127.0.0.1:27017

C:\Users\Admin>
```

Verificamos cual nodo tomo el lugar de primario (Tolerancia a fallos), el cual fue el siguiente



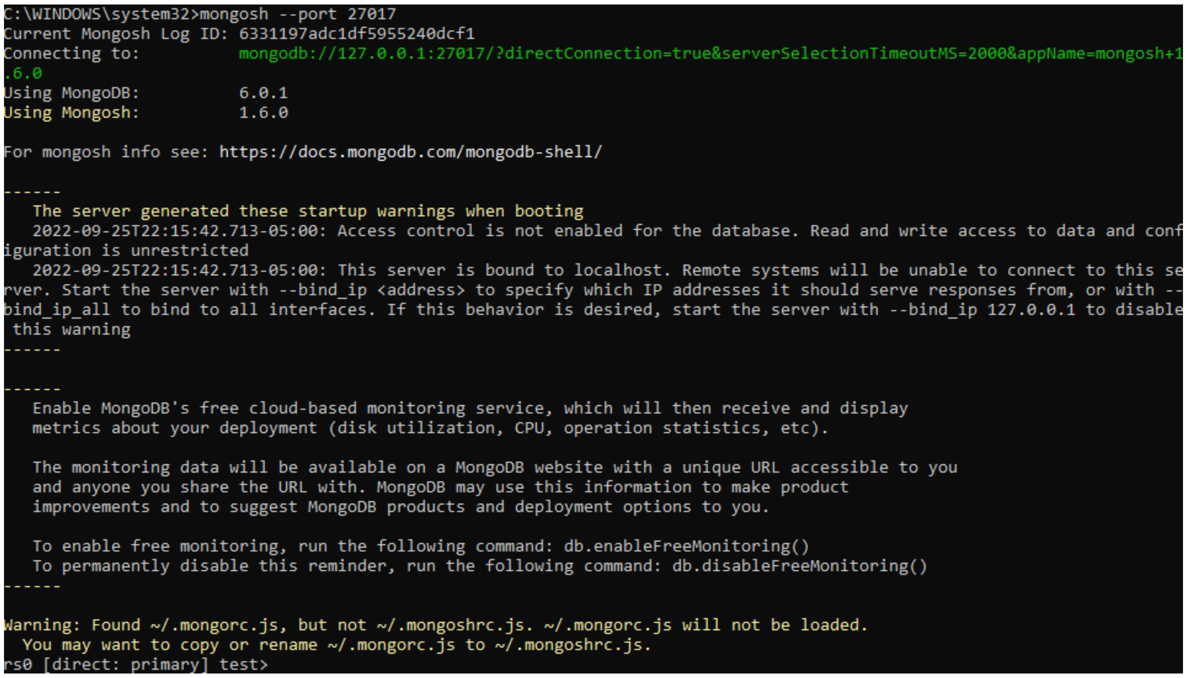
```
C:\Users\Admin>mongosh --port 27018
Current Mongosh Log ID: 633118cd583580a48b26f62a
Connecting to:          mongodb://127.0.0.1:27018/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
.6.0
Using MongoDB:          6.0.1
Using Mongosh:          1.6.0

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

------
   The server generated these startup warnings when booting
   2022-09-25T21:16:08.188-05:00: Access control is not enabled for the database. Read and write access to data and con
iguration is unrestricted
   2022-09-25T21:16:08.189-05:00: This server is bound to localhost. Remote systems will be unable to connect to this se
rver. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --
bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable
 this warning
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------
rs0 [direct: primary] test>
```

Verificamos que el nodo restante quedo como secundario



```
C:\Users\Admin>mongosh --port 27019
Current Mongosh Log ID: 63311905026e3c171e08c138
Connecting to:          mongodb://127.0.0.1:27019/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
.6.0
Using MongoDB:          6.0.1
Using Mongosh:          1.6.0

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

------
   The server generated these startup warnings when booting
   2022-09-25T21:16:35.756-05:00: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
   2022-09-25T21:16:35.757-05:00: This server is bound to localhost. Remote systems will be unable to connect to this se
rver. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --
bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable
 this warning
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.
rs0 [direct: secondary] test>
```

Reestablecemos la instancia del nodo principal y verificamos que es primario



```
C:\WINDOWS\system32>mongosh --port 27017
Current Mongosh Log ID: 6331197adc1df5955240dcf1
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
.6.0
Using MongoDB:          6.0.1
Using Mongosh:          1.6.0

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

------
   The server generated these startup warnings when booting
   2022-09-25T22:15:42.713-05:00: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
   2022-09-25T22:15:42.713-05:00: This server is bound to localhost. Remote systems will be unable to connect to this se
rver. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --
bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable
 this warning
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.
rs0 [direct: primary] test>
```