



INSTALAR MONGODB

Actividad 4



CRISTÓBAL SUÁREZ ABAD
ADMINISTRACIÓN DE SISTEMAS GESTORES DE BASES DE DATOS
2º ASIR

Introducción:

Sigue los siguientes pasos para instalar un cluster de MongoDB:

<https://serverspace.io/es/support/help/mongodb-cluster-configuration/>

- En el paso 1 de configuración del cluster sustituyelo por esto

Añade estas líneas en /etc/mongod.conf en todos los nodos.

replication:

replicaSetName: "rs0"

net:

bindIp: localhost, domainname

--> domainname sera el nombre de dominio del propio nodo

- Una vez realizado esto
 - Instala un cliente en tu equipo local y conéctate a la base de datos de mongo
 - Crea una nueva colección e inserta datos de prueba
 - Apaga el nodo primario y comprueba que algunos de los nodos secundario se ha transformado en primario.

Configuración de los servidores.

Detalles de los servidores utilizados:

1. Servidor primario: nombre — mongodb01, IP 10.2.17.80
2. Servidor secundario: nombre — mongodb02, IP 10.2.17.85
3. Servidor secundario: nombre — mongodb03, IP 10.2.17.90

Les cambiamos el nombre con el comando:

```
hostnamectl set-hostname mongodb01
```

En el archivo “/etc/hosts” ponemos las direcciones completas de cada uno, en cada uno de los servidores.

```
Tu Nombre jueves 6 noviembre 2025 10:00
[usuario@mongodb01 ~]$cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 mongodb01.suarez.abad mongodb01
10.2.17.80 mongodb01.suarez.abad mongodb01
10.2.17.85 mongodb02.suarez.abad mongodb02
10.2.17.90 mongodb03.suarez.abad mongodb03

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
[Tu Nombre jueves 6 noviembre 2025 10:00]
```

Instalamos mongo.

- 1) `sudo apt update`
- 2) `sudo apt install -y gnupg curl`
- 3) `curl -fsSL https://pgp.mongodb.com/server-7.0.asc | \`
`sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg --dearmor`
`echo "deb [signed-by=/usr/share/keyrings/mongodb-server-7.0.gpg]`
`\ https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0`
`multiverse" |`
`\ sudo tee /etc/apt/sources.list.d/mongodb-org-7.0.list`
- 4) `sudo apt update`
- 5) `sudo apt install -y mongodb-org`

Configuración del Cluster.

Una vez que tenemos instalado MongoDB en cada uno de los servidores, debemos realizar los siguientes pasos.

Podemos habilitarlo para que se inicie el servicio de MongoDB cada vez que se inicia el servidor.

systemctl enable mongod

systemctl restart mongod

systemctl status mongod

Si lo activas ahora tendrás que hacerle un “restart” cuando acabes de modificarlo.

Ahora, en el archivo “/etc/mongod.conf” de cada servidor debemos introducir la información que se mencionó en la introducción del trabajo. En nuestro caso lo hemos hecho así.

“replication:

replSetName: "rs0"

net:

port: 27017

bindIp: 127.0.0.1, 10.2.17.80

```
▶ port: 27017
▶ bindIp: 127.0.0.1

replication:
  replSetName: "rs0"

net:
  port: 27017
  bindIp: 127.0.0.1, 10.2.17.80
```

Cada servidor debe poner su IP.

Ahora nos metemos en la “Shell” de MongoDB usando el comando “mongosh”.

```

Tu Nombre jueves 6 noviembre 2025 10:10
[usuario@mongodb01 ~]$mongosh
Current Mongosh Log ID: 690c741b15723ca68a9dc29c
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionT
Using MongoDB:     7.0.25
Using Mongosh:     2.5.9

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

-----
The server generated these startup warnings when booting
2025-11-06T09:59:08.278+00:00: Using the XFS filesystem is strongly recommended with t
es-filesystem
2025-11-06T09:59:09.986+00:00: Access control is not enabled for the database. Read an
2025-11-06T09:59:09.987+00:00: vm.max_map_count is too low
-----

```

Ahí debemos introducir el siguiente código:

```

rs.initiate({
  _id: "rs0",
  members: [
    { _id: 0, host: "10.2.17.80:27017" },
    { _id: 1, host: "10.2.17.85:27017" },
    { _id: 2, host: "10.2.17.90:27017" }
  ]
})

```

“rs0” es el nombre del clúster y el resto de “id” identifican a los servidores. Nos debe salir algo como lo de la imagen de abajo.

```

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
test> rs.initiate({
...   _id: "rs0",
...   members: [
...     { _id: 0, host: "10.2.17.80:27017" },
...     { _id: 1, host: "10.2.17.85:27017" },
...     { _id: 2, host: "10.2.17.90:27017" }
...   ]
... })
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1762377823, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA='),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1762377823, i: 1 })
}
rs0 [direct: other] test> 

```

Si ahora usamos el comando “test> rs.conf()”, podremos obtener más información sobre el resto de servidores.

```
rs0 [direct: other] test> rs.conf()
{
  _id: 'rs0',
  version: 1,
  term: 1,
  members: [
    {
      _id: 0,
      host: '10.2.17.80:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long('0'),
      votes: 1
    },
    {
      _id: 1,
      host: '10.2.17.85:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long('0'),
      votes: 1
    },
    {
      _id: 2,
      host: '10.2.17.90:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
```

Cambiar la prioridad:

Usamos los siguientes comandos:

```
conf.members[0].priority = 1
```

```
conf.members[1].priority = 2
```

```
conf.members[2].priority = 3
```

```
rs0 [direct: primary] test> conf.members[0].priority = 1
1
rs0 [direct: primary] test> conf.members[1].priority = 2
2
rs0 [direct: primary] test> conf.members[2].priority = 3
3
```

Luego usamos “rs.reconfig(conf)” para que se aplique esta nueva configuración.

```
rs0 [direct: primary] test> rs.reconfig(conf)
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1762379007, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA='),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1762379007, i: 1 })
}
```

Ahora podemos verificar el estado de la configuración con el comando “rs.status()”.

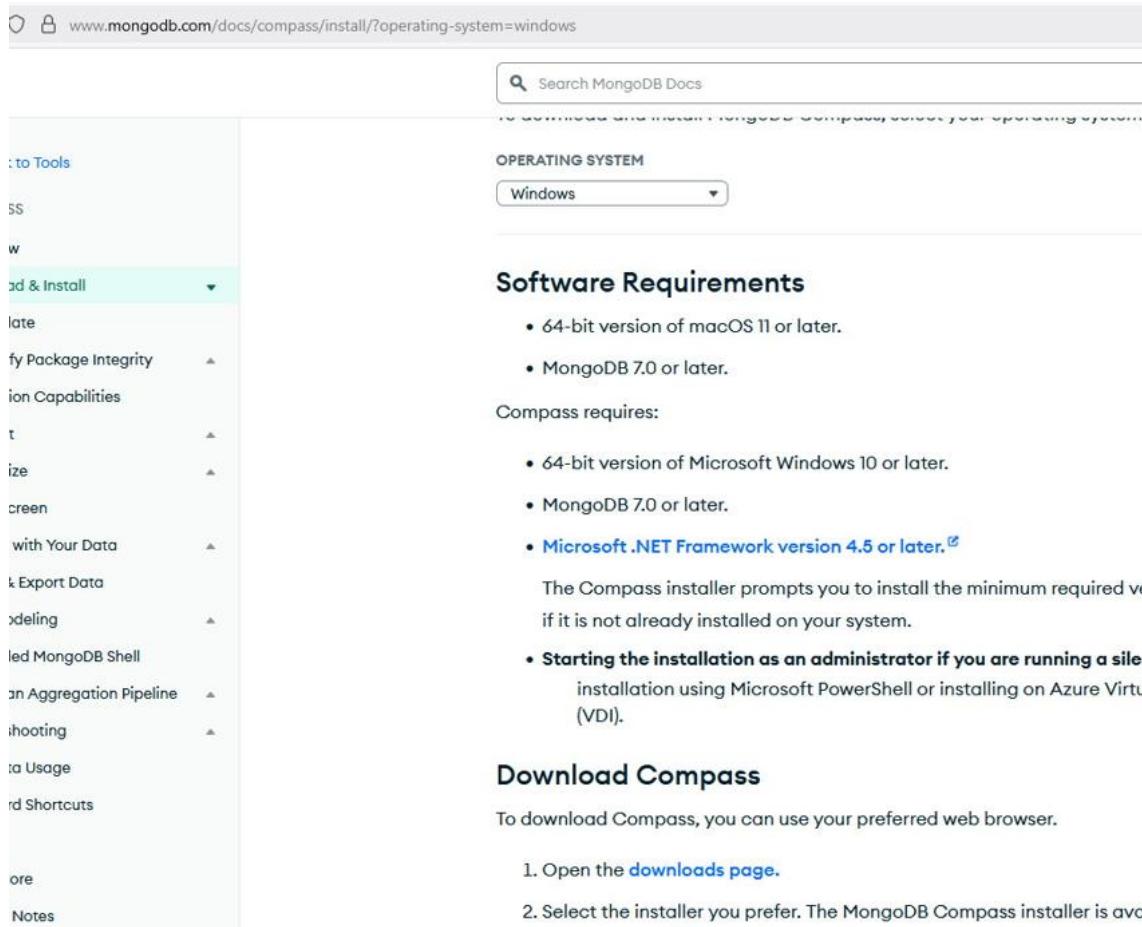
```
rs0 [direct: secondary] test> rs.status()
{
  set: 'rs0',
  date: ISODate('2025-11-05T21:48:14.817Z'),
  myState: 2,
  term: Long('2'),
  syncSourceHost: '10.2.17.90:27017',
  syncSourceId: 2,
  heartbeatIntervalMillis: Long('2000'),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
  writableVotingMembersCount: 3,
  optimes: {
    lastCommittedOpTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    lastCommittedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    readConcernMajorityOpTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    appliedOpTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    durableOpTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    lastAppliedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastDurableWallTime: ISODate('2025-11-05T21:48:08.157Z')
  },
  lastStableRecoveryTimestamp: Timestamp({ t: 1762379258, i: 1 }),
  electionParticipantMetrics: {
    votedForCandidate: true,
    electionTerm: Long('2'),
    lastVoteDate: ISODate('2025-11-05T21:43:38.123Z'),
    electionCandidateMemberId: 2,
    voteReason: '',
    lastAppliedOpTimeAtElection: { ts: Timestamp({ t: 1762379007, i: 1 }), t: Long('1') },
    maxAppliedOpTimeInSet: { ts: Timestamp({ t: 1762379007, i: 1 }), t: Long('1') }
  },
  priorityAtElection: 1,
  newTermStartDate: ISODate('2025-11-05T21:43:38.141Z'),
  newTermAppliedDate: ISODate('2025-11-05T21:43:38.148Z')
},
members: [
  {
    _id: '10.2.17.90:27017',
    host: '10.2.17.90:27017',
    port: 27017,
    state: 2,
    term: Long('2'),
    opTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    durableOpTime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    lastAppliedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastDurableWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastStableRecoveryTimestamp: Timestamp({ t: 1762379258, i: 1 })
  }
]
```

Ahí nos deben salir información sobre cada uno de los servidores. Nos importa en este caso el valor de la prioridad, que puede ser PRIMARY o SECONDARY.

```
members: [
  {
    _id: 0,
    name: '10.2.17.80:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 1732,
    optime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    optimeDate: ISODate('2025-11-05T21:48:08.000Z'),
    lastAppliedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastDurableWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    syncSourceHost: '10.2.17.90:27017',
    syncSourceId: 2,
    infoMessage: '',
    configVersion: 2,
    configTerm: 2,
    self: true,
    lastHeartbeatMessage: ''
  },
  {
    _id: 1,
    name: '10.2.17.85:27017',
    health: 1,
    state: 2,
    stateStr: 'SECONDARY',
    uptime: 1471,
    optime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    optimeDurable: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    optimeDate: ISODate('2025-11-05T21:48:08.000Z'),
    optimeDurableDate: ISODate('2025-11-05T21:48:08.000Z'),
    lastAppliedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastDurableWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastHeartbeat: ISODate('2025-11-05T21:48:14.805Z'),
    lastHeartbeatRecv: ISODate('2025-11-05T21:48:14.231Z'),
    pingMs: Long('0'),
    lastHeartbeatMessage: '',
    syncSourceHost: '10.2.17.80:27017',
    syncSourceId: 0,
    infoMessage: ''
  },
  {
    _id: 2,
    name: '10.2.17.90:27017',
    health: 1,
    state: 1,
    stateStr: 'PRIMARY',
    uptime: 1471,
    optime: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    optimeDurable: { ts: Timestamp({ t: 1762379288, i: 1 }), t: Long('2') },
    optimeDate: ISODate('2025-11-05T21:48:08.000Z'),
    optimeDurableDate: ISODate('2025-11-05T21:48:08.000Z'),
    lastAppliedWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastDurableWallTime: ISODate('2025-11-05T21:48:08.157Z'),
    lastHeartbeat: ISODate('2025-11-05T21:48:14.806Z'),
    lastHeartbeatRecv: ISODate('2025-11-05T21:48:14.271Z'),
    pingMs: Long('0'),
    lastHeartbeatMessage: '',
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    electionTime: Timestamp({ t: 1762379018, i: 1 }),
    electionDate: ISODate('2025-11-05T21:43:38.000Z'),
    configVersion: 2,
    configTerm: 2
  }
],
```

Instala un cliente en tu equipo local y conéctate a la base de datos de mongo.

Nos dirigimos a la página oficial de mongo y elegimos el sistema operativo donde lo vamos a instalar: <https://www.mongodb.com/docs/compass/install/?operating-system=windows> En nuestro caso, Windows.

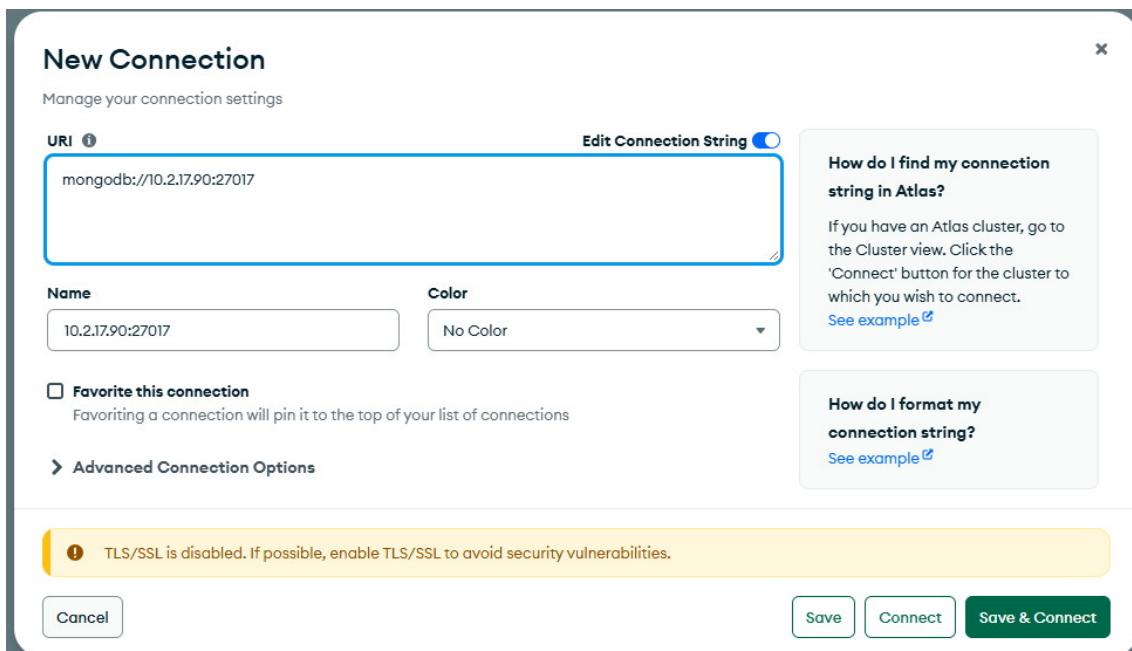


The screenshot shows a web browser displaying the MongoDB Compass documentation for Windows. The URL in the address bar is www.mongodb.com/docs/compass/install/?operating-system=windows. The page has a sidebar on the left with various links related to MongoDB and Compass. The main content area is titled "Software Requirements" and includes two bulleted lists: one for MongoDB requirements (64-bit version of macOS 11 or later, MongoDB 7.0 or later) and one for Compass requirements (64-bit version of Microsoft Windows 10 or later, MongoDB 7.0 or later, Microsoft .NET Framework version 4.5 or later). Below this, there is a section about starting the installation as an administrator if running silently. The "Download Compass" section at the bottom provides instructions for downloading the software using a web browser.

Le damos al archivo ejecutable y una vez abierto el programa nos vamos a añadir nueva conexión.



Introducimos el nombre o IP de nuestros servidores:



The screenshot shows the main MongoDB Compass interface. At the top, it says 'MongoDB Compass - 10.2.17.90:27017/Databases'. Below that is a navigation bar with 'Connections', 'Edit', 'View', and 'Help'. The main area is titled 'Compass' and has a sidebar with 'My Queries' and 'Data Modeling' buttons. On the left, there is a 'CONNECTIONS (3)' section with a search bar. The connections are listed as follows:

- 10.2.17.80:27017
 - admin
 - config
 - local
- 10.2.17.85:27017
 - admin
 - config
 - local
- 10.2.17.90:27017
 - admin
 - config
 - local

Crea una nueva colección e inserta datos de prueba.

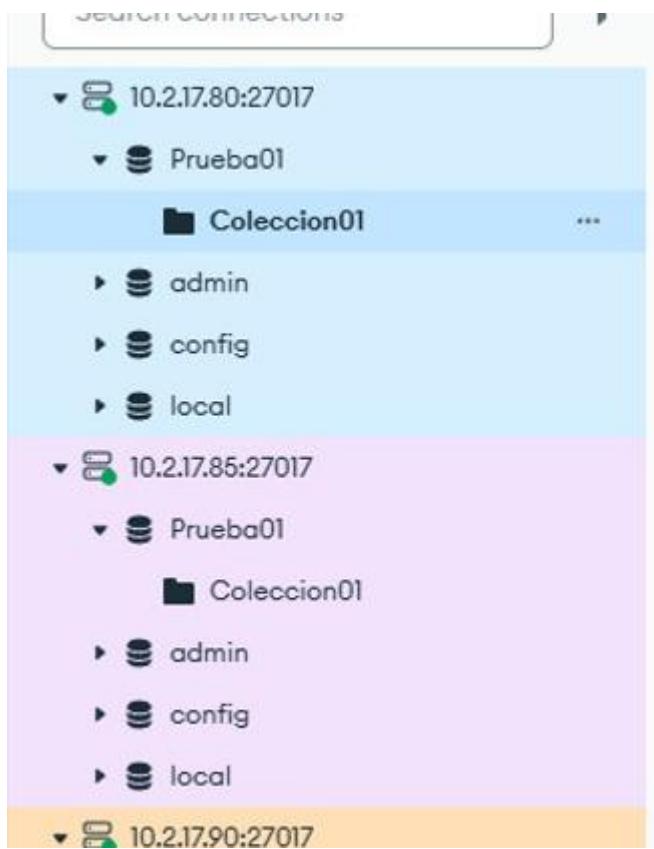
Seleccionamos un servidor y en este caso le vamos a crear una nueva base de datos y dentro de ella una nueva colección.

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar titled 'Connections' with a list of available servers: '10.2.17.80:27017', '10.2.17.90:27017', and '10.2.17.85:27017'. The '10.2.17.90:27017' server is selected. The main area displays the databases 'admin', 'config', and 'local'. A modal window titled 'Create Database' is open, prompting for a 'Database Name' (set to 'Prueba01') and a 'Collection Name' (set to 'Coleccion01'). There are also checkboxes for 'Time-Series' and 'Additional preferences'. At the bottom right of the modal are 'Cancel' and 'Create Database' buttons.

Seleccionamos la colección creada y le damos a "Add Data". Ahí nos da la opción de modificar una plantilla genérica para meter un documento. Como en este caso no se ha especificado nada sobre el contenido de la información, se ha decidido simplemente meter la plantilla.

The screenshot shows the MongoDB Compass interface for the 'Coleccion01' collection within the 'Prueba01' database. The top navigation bar includes tabs for 'Welcome', '10.2.17.80:27017', '10.2.17.90:27017' (which is highlighted in orange), 'Prueba01', 'Coleccion01' (highlighted in yellow), and 'config'. Below the tabs, the path '10.2.17.90:27017 > Prueba01 > Coleccion01' is displayed. The main area has tabs for 'Documents' (with 1 item), 'Aggregations', 'Schema', 'Indexes' (with 1 item), and 'Validation'. Under the 'Documents' tab, there's a search bar with a dropdown icon and the placeholder 'Type a query: { field: 'value' } or [Generate query](#) ✨'. Below the search bar are four buttons: '+ ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. A preview pane shows a single document with the '_id' field set to 'ObjectId('690c6f7f3ef7036f5fd9f5c2')'.

Si actualizamos el estado de los otros dos servidores, podemos ver como se ha replicado la base de datos y la colección (y también la información insertada).



Apaga el nodo primario y comprueba que algunos de los nodos secundario se han transformado en primario.

Vamos a apagar el servidor “mongodb03” que en nuestro caso es el primario.

```
Tu Nombre jueves 6 noviembre 2025 09:46
[root@mongodb03 usuario]$mongosh
Current Mongosh Log ID: 690c700933f3e4ae409dc29c
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=1
Using MongoDB:     7.0.25
Using Mongosh:      2.5.9

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

-----
The server generated these startup warnings when booting
2025-11-06T09:38:02.944+00:00: Using the XFS filesystem is strongly
recommended
2025-11-06T09:38:04.613+00:00: Access control is not enabled for the
system
2025-11-06T09:38:04.613+00:00: vm.max_map_count is too low
-----

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

Una vez que lo apagamos, el servidor “mongodb02” se convierte en primario.

```
Tu Nombre jueves 6 noviembre 2025 09:54
[root@mongodb02 usuario]$mongosh
Current Mongosh Log ID: 690c704113bfd9b139dc29c
Connecting to:      mongodb://127.0.0.1:27017/?directConnec
Using MongoDB:     7.0.25
Using Mongosh:      2.5.9

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

To help improve our products, anonymous usage data is collected
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2025-11-06T09:38:00.042+00:00: Using the XFS filesystem is recommended
2025-11-06T09:38:01.666+00:00: Access control is not enabled
2025-11-06T09:38:01.667+00:00: vm.max_map_count is too low
-----

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