

# **CORPORACION UNIVERSITARIA IBEROAMERICANA**

**Bases de datos avanzadas**

**Conceptos y comandos básicos de la replicación en bases de datos NoSQL**

**Juan Camilo Ballesteros Carmona**

**24/05/2024**

## TABLA DE CONTENIDOS

<b>Introducción .....</b>	<b>3</b>
<b>Contenido.....</b>	<b>4</b>
<b>Conclusion .....</b>	<b>28</b>
<b>Referencias.....</b>	<b>29</b>

## **Introducción**

Este documento especifica los requisitos no funcionales relacionados con la redundancia y la disponibilidad 24x7 del sistema de almacenamiento de información de la Copa del Mundo de fútbol. Estos requisitos aseguran que el sistema sea confiable y esté siempre accesible, permitiendo a los usuarios consultar información en cualquier momento.

## Contenido

### Documento de requerimientos no funcionales

- El requisito principal es mantener la operación ininterrumpida de la base de datos Torneo\_Copa\_Del\_Mundo y sus colecciones (árbitros, deportistas, encuentrosDeportivos, entrenadores y equipos).
- Se debe implementar la replicación de la base de datos, que incluirá tres nodos, los cuales contendrán información de gran importancia.
- Los nodos dispuestos para el proyecto imitarán la función de tres servidores, los cuales mantendrán la integridad de los datos y los almacenarán.
- Los nodos secundarios contarán con los respectivos permisos de lectura para la verificación de la integridad de los datos que están siendo replicados.
- Si el nodo maestro se encuentra inhabilitado por alguna circunstancia se postulará uno de los nodos secundarios para que ocupe la posición de nodo primario y brinde acceso a la información sin causar interrupciones.
- El sistema debe asegurar que la entrada, consulta o modificación de información se realice de la manera más eficiente posible.

Con base en lo anterior, se garantiza la redundancia y la disponibilidad del sistema. Asegurando que el sistema pueda operar de manera continua y confiable, minimizando el riesgo de interrupciones y asegurando la accesibilidad constante de la información del torneo de la copa del mundo de fútbol.

Enlace repositorio GITHUB

<https://github.com/Juan2033/torneoCopaDelMundo.git>

Enlace al video (Es un poco extenso, se sugiere ver en velocidad 1.5)

[https://drive.google.com/file/d/141X4e650nnzy4Id45vkLEZRADHzv\\_-Sk/view?usp=sharing](https://drive.google.com/file/d/141X4e650nnzy4Id45vkLEZRADHzv_-Sk/view?usp=sharing)

## Paso 1 – Creación del ReplicaSet

Para verificar el mecanismo de réplica de MongoDB, será necesario establecer varias instancias de MongoDB como servidores y configurarlas para que todas mantengan una copia sincronizada de los datos. Usaremos el comando `MiejsReplicaSet = new ReplSetTest ({name: "replicaSetTorneoCopaMundo", nodes: 3})`. Al ejecutar el comando, por consola nos imprimirá el contenido del objeto `ReplSetTest` que acabamos de crear.

```

MongoDB shell version v4.2.25
connecting to: mongod://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
Implicit session: session { "id" : UUID("2c283feb-aaaf-4b1d-b033-3eb03b0017d6") }
MongoDB server version: 4.2.25
Server has startup warnings:
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten]
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten] **           Read and write access to data and configuration is unrestricted.
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten]
---
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()
---
> MiejsReplicaSet = new ReplSetTest ({name: "replicaSetTorneoCopaMundo", nodes: 3})
Starting new replica set replicaSetTorneoCopaMundo
{
  "kDefaultTimeoutMS" : 600000,
  "getReadConcernMajorityOpTimeOrThrow" : function(conn) {
    const majorityOpTime = _getReadConcernMajorityOpTime(conn);
    if (friendlyEqual(majorityOpTime, {ts: Timestamp(0, 0), t: NumberLong(0)})) {
      throw new Error("readConcern majority optime not available");
    }
    return majorityOpTime;
  },
  "nodeList" : function() {
    var list = [];
    for (var i = 0; i < this.ports.length; i++) {
      list.push(this.host + ":" + this.ports[i]);
    }
    return list;
  },
  "getNodeId" : function(node) {

```

## Paso 2 – Arrancar los procesos de MongoDB de la replica

Hasta el momento, solo hemos configurado el conjunto de réplicas. Ahora procederemos a iniciar las instancias de los nodos que forman parte de la réplica. Para poner en marcha los nodos del conjunto de réplicas, ejecutaremos la función `startSet()` en el objeto que representa dicho conjunto. `MiejReplicaSet.startSet()` Al ejecutar este comando, veremos la configuración de cada nodo del conjunto de réplicas y los mensajes de registro que indican que cada una de las instancias está iniciándose.

```

> MieiReplicaSet.startSet()
RepSetTest starting set
RepSetTest n is : 0
{
  "useHostName" : true,
  "oplogSize" : 40,
  "keyFile" : undefined,
  "port" : 20000,
  "replSet" : "replicaSetTorneoCopaMundo",
  "dbpath" : "$set-$node",
  "restart" : undefined,
  "pathOpts" : {
    "node" : 0,
    "set" : "replicaSetTorneoCopaMundo"
  },
  "setParameter" : {
    "writePeriodicNoops" : false,
    "numInitialSyncConnectAttempts" : 60
  }
}
RepSetTest Starting....
Resetting db path '/data/db/replicaSetTorneoCopaMundo-0'
2024-05-23T09:35:16.614-0500 I - [js] shell: started program (sh64908): C:\Program Files\MongoDB\Server\4.2\bin\mongod.exe --oplogSize 40 --port 20000 --replSet replicaSetTorneoCopaMundo --dbpath /data/db/replicaSetTorneoCopaMundo-0 --setParameter writePeriodicNoops=false --setParameter numInitialSyncConnectAttempts=60 --bind_ip 0.0.0.0 --setParameter enableTestCommands=1 --setParameter disableLogicalSessionCacheRefresh=true --setParameter minNumChunksForSessionsCollection=1 --setParameter orphanCleanupDelaySecs=1
d20000 2024-05-23T09:35:17.056-0500 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
d20000 2024-05-23T09:35:17.059-0500 W ASIO [main] No TransportLayer configured during NetworkInterface startup
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] MongoDB starting : pid=64908 port=20000 dbpath=/data/db/replicaSetTorneoCopaMundo-0 64-bit host=DESKTOP-SJS0E4P
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] db version v4.2.25
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] git version: 41b59c2bfb5121e66f18cc3ef40055a1b5fb6c2e
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] allocator: tcmalloc
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] modules: none
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] build environment:
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] distmod: 2012plus
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] distarch: x86_64
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] target_arch: x86_64
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] options: { net: { bindIp: '0.0.0.0', port: 20000 }, replication: { oplogSizeMB: 40, replSet : "replicaSetTorneoCopaMundo" }, setParameter: { disableLogicalSessionCacheRefresh: "true", enableTestCommands: "1", minNumChunksForSessionsCollection: "1",

```

### Paso 3 – Iniciar el proceso de replica

En este punto, hemos configurado los tres procesos de servicio mongod que componen nuestro conjunto de réplicas, pero aún no hemos activado la funcionalidad de replicación de datos. Para habilitar la replicación, debemos llamar a la función `initiate()` en el objeto `ReplSetTest`. **MiejReplSet.initiate()**. Al ejecutar esta función, veremos en la salida de la consola la configuración de los miembros del conjunto de réplicas y luego se activará la funcionalidad de replicación en el grupo.

```

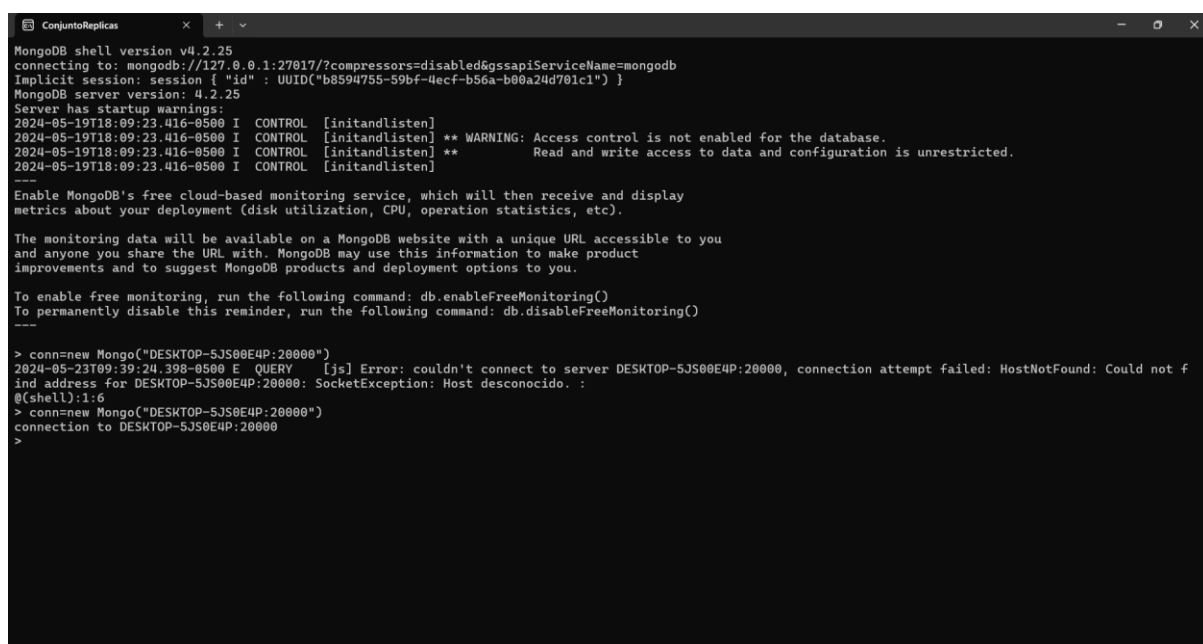
> MijReplSet.startSet()
ReplSetTest starting set
ReplSetTest n is : 0
{
  "useHostName" : true,
  "oplogSize" : 40,
  "keyFile" : undefined,
  "port" : 20000,
  "replSet" : "replicaSetTorneoCopaMundo",
  "dbpath" : "$set-$node",
  "restart" : undefined,
  "pathOpts" : {
    "node" : 0,
    "set" : "replicaSetTorneoCopaMundo"
  },
  "setParameter" : {
    "writePeriodicNoops" : false,
    "numInitialSyncConnectAttempts" : 60
  }
}
ReplSetTest Starting....
Resetting db path '/data/db/replicaSetTorneoCopaMundo-0'
2024-05-23T09:35:16.614-0500 I - [js] shell: started program (sh64900): C:\Program Files\MongoDB\Server\4.2\bin\mongod.exe --oplogSize 40 --port 20000 --replSet replicaSetTorneoCopaMundo --dbpath /data/db/replicaSetTorneoCopaMundo-0 --setParameter writePeriodicNoops=false --setParameter numInitialSyncConnectAttempts=60 --bind_ip 0.0.0.0 --setParameter enableTestCommands=1 --setParameter disableLogicalSessionCacheRefresh=true --setParameter minNumChunksForSessionsCollection=1 --setParameter orphanCleanupDelaySecs=1
d20000 2024-05-23T09:35:17.056-0500 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
d20000 2024-05-23T09:35:17.059-0500 W ASIO [main] No TransportLayer configured during NetworkInterface startup
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] MongoDB starting : pid=64908 port=20000 dbpath=/data/db/replicaSetTorneoCopaMundo-0 64-bit host=DESKTOP-SJ58E4P
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] db version v4.2.25
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] git version: 41b59c2bfb5121e66f18cc3ef40055a1b5fb6c2e
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] allocator: tcmalloc
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] modules: none
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] build environment:
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] distmod: 2012plus
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] distarch: x86_64
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] target_arch: x86_64
d20000 2024-05-23T09:35:17.061-0500 I CONTROL [initandlisten] options: { net: { bindIp: '0.0.0.0', port: 20000 }, replication: { oplogSizeMB: 40, replSet : "replicaSetTorneoCopaMundo" }, setParameter: { disableLogicalSessionCacheRefresh: "true", enableTestCommands: "1", minNumChunksForSessionsCollection: "1",

```

## Paso 4

Conexión al nodo primario del grupo de réplica. En la segunda consola digitamos la siguiente cadena de conexión:

**> conn=new Mongo("DESKTOP-5JS0E4P:20000")**



```

MongoDB shell version v4.2.25
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("b8594755-59bf-4ecf-b56a-b00a24d701c1") }
MongoDB server version: 4.2.25
Server has startup warnings:
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten]
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten] **           Read and write access to data and configuration is unrestricted.
2024-05-19T18:09:23.416-0500 I CONTROL [initandlisten]
---
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()
---
> conn=new Mongo("DESKTOP-5JS0E4P:20000")
2024-05-23T09:39:24.398-0500 E QUERY [js] Error: couldn't connect to server DESKTOP-5JS0E4P:20000, connection attempt failed: HostNotFound: Could not f
ind address for DESKTOP-5JS0E4P:20000: SocketException: Host desconocido. :
@(shell):1:6
> conn=new Mongo("DESKTOP-5JS0E4P:20000")
connection to DESKTOP-5JS0E4P:20000
>

```



## Paso 5

Una vez obtenida la conexión, obtenemos la BD sobre la que realizaremos la prueba. En nuestro caso, para esta prueba vamos a utilizar la base de datos **Torneo\_Copa\_Del\_Mundo** que contiene las colecciones: árbitros, deportistas, encuentrosDeportivos, entrenadores y equipos.

```
> testDB=conn.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
>
```

## Paso 6

Como último paso sobre la BD seleccionada, preguntaremos si es el nodo primario, utilizando la función `isMaster()`:

```
ConjuntoReplicas
> testDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : true,
  "secondary" : false,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20000",
  "electionId" : ObjectId("7fffffff0000000000000001"),
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716475027, 1),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T14:37:07Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716475027, 1),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T14:37:07Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T14:45:29.169Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 26,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716475027, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
```

Del objeto devuelto, nos fijaremos en la propiedad `ismaster` y `secondary`. Si el nodo al que nos hemos conectado no es el primario, volveremos a conectarnos al siguiente nodo del grupo de réplica, hasta que encontremos el nodo primario.

## **Paso 7 – Insertar un conjunto de datos en el nodo primario**

Una vez que ya estamos conectados al nodo primario, vamos a ejecutar una inserción de un conjunto de datos en las colecciones correspondientes, en este caso son **árbitros**, **deportistas**, **encuentrosDeportivos**, **entrenadores** y **equipos**, veamos:

### **Insertando documentos a la colección “árbitros”**

```
> testDB.arbitros.insert(
... {
... id_editorial: " 60aaf1f88206310001528051 ",
... Nombre : " Howard Webb ",
... Pais: " Inglaterra ",
... experiencia: " 15 "
... });
WriteResult({ "nInserted" : 1 })
> testDB.arbitros.insert (
... {
... id_editorial: " 60aaf1f88206310001528052 ",
... Nombre: " Pierluigi Collina ",
... Pais: " Italia ",
... experiencia: " 10 "
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.arbitros.insert (
... {
... id_editorial: "60aaf1f88206310001528053",
... Nombre: "Nestor Pitana ",
... Pais: " Argentina ",
... experiencia: " 8 "
... });
WriteResult({ "nInserted" : 1 })
>
```

### **Insertando documentos a la colección “deportistas”**

```
> testDB.deportistas.insert(
... {
... id_editorial: " 60aaf1f8820631000152803a ",
... nombre: "Lionel Messi",
... edad: 34,
... posicion: "Delantero",
... equipo: "Argentina"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.deportistas.insert(
... {
... id_editorial: " 60aaf1f8820631000152803c ",
... nombre: "Cristiano Ronaldo",
... edad: 36,
... posicion: "Delantero",
... equipo: "Portugal"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.deportistas.insert(
... {
... id_editorial: " 60aaf1f8820631000152803e ",
... nombre: "Neymar Jr",
... edad: 29,
... posicion: "Delantero",
... equipo: "Brasil"
... });
WriteResult({ "nInserted" : 1 })
>
```

## Insertando documentos a la colección “encuentrosDeportivos”

```
> testDB.encuentrosDeportivos.insert(
... {
... id: "60aaaf1f8820631000152805a",
... equipo_local: "Brasil",
... equipo_visitante: "Croacia",
... arbitro_principal: "Pierluigi Collina"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.encuentrosDeportivos.insert (
... {
... id: "60aaaf1f8820631000152805b",
... equipo_local: "Países Bajos",
... equipo_visitante: "Croacia",
... arbitro_principal: "Nestor Pitana"
... });
```

## Insertando documentos en la colección “entrenadores”

```
> testDB.entrenadores.insert(
... {
... id: "60aaaf1f8820631000152804a",
... nombre: "Fernando Santos",
... pais: "Portugal",
... experiencia: 12
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.entrenadores.insert(
... {
... id: "60aaaf1f8820631000152804b",
... nombre: "Tite",
... pais: "Brasil",
... experiencia: 8
... });
WriteResult({ "nInserted" : 1 })
>
```

## Insertando documentos en la colección “equipos”

```
> testDB.equipos.insert(
... {
... id: "60aaaf1f8820631000152803b",
... pais: "Argentina",
... entrenador: "Lionel Scaloni"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.equipos.insert(
... {
... id: "60aaaf1f8820631000152803d",
... pais: "Portugal",
... entrenador: "Fernando Santos"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.equipos.insert(
... {
... id: "60aaaf1f8820631000152803f",
... pais: "Brasil",
... entrenador: "Tite"
... });
WriteResult({ "nInserted" : 1 })
>
```

Por último, comprobamos que se han almacenado los registros en la colección por medio de los siguientes comandos:

```
> testDB.arbitros.count();
```

```
> testDB.deportistas.count();
```

### Colección “arbitros”

```
> testDB.arbitros.count();
3
>
```

### Colección “deportistas”

```
> testDB.deportistas.count()
3
>
```

### Colección “encuentrosDeportivos”

```
> testDB.encuentrosDeportivos.count()
5
>
```

### Colección “entrenadores”

```
> testDB.entrenadores.count()
3
>
```

### Colección “equipos”

```
> testDB.equipos.count()
3
>
```

## Paso 8 - Comprobación de la réplica sobre los nodos secundarios

Una vez que hemos insertado los datos a través del nodo primario, vamos a conectarnos a alguno de los nodos secundarios, y comprobar si se han replicado los datos a ese nodo.

Empezamos por conectarnos a uno de los nodos secundarios, obtener la conexión y comprobamos que, efectivamente, el nodo si es secundario:

```
> connSecondary = new Mongo("DESKTOP-5JS0E4P:20001")
```

```
> connSecondary = new Mongo("DESKTOP-5JS0E4P:20001")
connection to DESKTOP-5JS0E4P:20001
>
```

```
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
```

```
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
>
```

Comprobamos si este nodo es el master o no lo es:

```
> secondaryTestDB.isMaster()
```

```
ConjuntoReplicas
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20001",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T15:22:03Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T15:22:03Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T15:32:52.842Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 18,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
```

Ahora que estamos ya conectados a la base de datos en el nodo secundario dentro del grupo de réplica, vamos a comprobar si los datos que hemos insertado en el nodo primario se han replicado en este secundario.

Para ello hacemos una consulta a la colección:

**> secondaryTestDB.arbitros.count();**

```
> secondaryTestDB.arbitros.count();
2024-05-23T10:35:13.416-0500 E QUERY [js] uncaught exception: Error: count failed: {
  "operationTime" : Timestamp(1716477723, 4),
  "ok" : 0,
  "errmsg" : "not master and slaveOk=false",
  "code" : 13435,
  "codeName" : "NotPrimaryNoSecondaryOk",
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
} :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DBQuery.prototype.count@src/mongo/shell/query.js:376:11
DBCollection.prototype.count@src/mongo/shell/collection.js:1401:12
@(shell):1:1
>
```

En este caso obtenemos un error porque por defecto, tal y como comentamos en los nodos secundarios en un grupo de réplica, no admiten operaciones ni de escritura ni de lectura. Todas las operaciones deben realizarse siempre sobre el nodo principal.

Sin embargo, podemos activar el permiso para realizar operaciones de lectura sobre un nodo secundario.

**> connSecondary.setSecondaryOk()**

```
> connSecondary.setSecondaryOk()
>
```

Volvemos a hacer la consulta sobre la colección “**arbitros**”:

**> secondaryTestDB.arbitros.count();**

```
> secondaryTestDB.arbitros.count();
3
>
> secondaryTestDB.arbitros.find().pretty();
{
  "_id" : ObjectId("664f5a0735ee3a886c9dc662"),
  "id_editorial" : " 60aaf1f88206310001528051 ",
  "Nombre" : " Howard Webb ",
  "Pais" : " Inglaterra ",
  "experiencia" : " 15 "
}
{
  "_id" : ObjectId("664f5a1235ee3a886c9dc663"),
  "id_editorial" : " 60aaf1f88206310001528052 ",
  "Nombre" : " Pierluigi Collina ",
  "Pais" : " Italia ",
  "experiencia" : " 10 "
}
{
  "_id" : ObjectId("664f5a1435ee3a886c9dc664"),
  "id_editorial" : "60aaf1f88206310001528053",
  "Nombre" : "Nestor Pitana ",
  "Pais" : " Argentina ",
  "experiencia" : " 8 "
}
}
```

En este caso, comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.

Volvemos a hacer la consulta sobre la colección “**deportistas**”:

```
> secondaryTestDB.deportistas.count();
3
>
> secondaryTestDB.deportistas.find().pretty();
{
  "_id" : ObjectId("664f5b1a35ee3a886c9dc665"),
  "id_editorial" : " 60aaf1f8820631000152803a ",
  "nombre" : "Lionel Messi",
  "edad" : 34,
  "posicion" : "Delantero",
  "equipo" : "Argentina"
}
{
  "_id" : ObjectId("664f5b1a35ee3a886c9dc666"),
  "id_editorial" : " 60aaf1f8820631000152803c ",
  "nombre" : "Cristiano Ronaldo",
  "edad" : 36,
  "posicion" : "Delantero",
  "equipo" : "Portugal"
}
{
  "_id" : ObjectId("664f5b1a35ee3a886c9dc667"),
  "id_editorial" : " 60aaf1f8820631000152803e ",
  "nombre" : "Neymar Jr",
  "edad" : 29,
  "posicion" : "Delantero",
  "equipo" : "Brasil"
}
}
```

Comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.

Volvemos a hacer la consulta sobre la colección “**encuentrosDeportivos**”:

```
> secondaryTestDB.encuentrosDeportivos.count();
5
>
> secondaryTestDB.encuentrosDeportivos.find().pretty();
{
  "_id" : ObjectId("664f5ca335ee3a886c9dc668"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc669"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66a"),
  "id" : "60aaf1f88206310001528059",
  "equipo_local" : "Argentina",
  "equipo_visitante" : "Portugal",
  "arbitro_principal" : "Howard Webb"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66b"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
},
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66c"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}
}
```

Comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.

Volvemos a hacer la consulta sobre la colección “**entrenadores**”:

```
> secondaryTestDB.entrenadores.count();
3
>
> secondaryTestDB.entrenadores.find().pretty();
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66d"),
  "id" : "60aaf1f88206310001528049",
  "nombre" : "Lionel Scaloni",
  "pais" : "Argentina",
  "experiencia" : 3
},
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66e"),
  "id" : "60aaf1f8820631000152804a",
  "nombre" : "Fernando Santos",
  "pais" : "Portugal",
  "experiencia" : 12
},
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66f"),
  "id" : "60aaf1f8820631000152804b",
  "nombre" : "Tite",
  "pais" : "Brasil",
  "experiencia" : 8
}
}
```

Comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.



Volvemos a hacer la consulta sobre la colección “equipos”:

```
> secondaryTestDB.equipos.count();
3
>
> secondaryTestDB.equipos.find().pretty();
{
  "_id" : ObjectId("664f5f1b35ee3a886c9dc670"),
  "id" : "60aaf1f8820631000152803b",
  "pais" : "Argentina",
  "entrenador" : "Lionel Scaloni"
}
{
  "_id" : ObjectId("664f5f1b35ee3a886c9dc671"),
  "id" : "60aaf1f8820631000152803d",
  "pais" : "Portugal",
  "entrenador" : "Fernando Santos"
}
{
  "_id" : ObjectId("664f5f1b35ee3a886c9dc672"),
  "id" : "60aaf1f8820631000152803f",
  "pais" : "Brasil",
  "entrenador" : "Tite"
}
}
```

Comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.

## Paso 9 – Comprobación en el tercer nodo

```
ConjuntoReplicas x + v
> connSecondary = new Mongo("DESKTOP-SJS0E4P:20002")
connection to DESKTOP-SJS0E4P:20002
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-SJS0E4P:20000",
    "DESKTOP-SJS0E4P:20001",
    "DESKTOP-SJS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-SJS0E4P:20000",
  "me" : "DESKTOP-SJS0E4P:20002",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T15:22:03Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T15:22:03Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T15:54:18.065Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 19,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
```

```

ConjuntoReplicas
},
"bsonObjectSize" : 16777216,
"maxMessageSizeBytes" : 48000000,
"maxWriteBatchSize" : 100000,
"localTime" : ISODate("2024-05-23T15:54:18.065Z"),
"logicalSessionTimeoutMinutes" : 30,
"connectionId" : 19,
"minWireVersion" : 0,
"maxWireVersion" : 8,
"readOnly" : false,
"ok" : 1,
"$clusterTime" : {
  "clusterTime" : Timestamp(1716477723, 4),
  "signature" : {
    "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
    "keyId" : NumberLong(0)
  }
},
"operationTime" : Timestamp(1716477723, 4)
}
> secondaryTestDB.arbitros.count();
2024-05-23T10:55:36.066-0500 E QUERY [js] uncaught exception: Error: count failed: {
  "operationTime" : Timestamp(1716477723, 4),
  "ok" : 0,
  "errmsg" : "not master and slaveOk=false",
  "code" : 13435,
  "codeName" : "NotPrimaryNoSecondaryOk",
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
} :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DBQuery.prototype.count@src/mongo/shell/query.js:376:11
DBCollection.prototype.count@src/mongo/shell/collection.js:1401:12
@(shell):1:1
> connSecondary.setSecondaryOk()
>
> secondaryTestDB.arbitros.count();
3
> secondaryTestDB.arbitros.find().pretty();
{
  "_id" : ObjectId("664f5a0735ee3a886c9dc662"),
  "id_editorial" : " 60aaf1f88206310001528051 ",
  "Nombre" : " Howard Webb ",
  "Pais" : " Inglaterra ",
  "experiencia" : " 15 "
}
{
  "_id" : ObjectId("664f5a1235ee3a886c9dc663"),
  "id_editorial" : " 60aaf1f88206310001528052 ",
  "Nombre" : " Pierluigi Collina ",
  "Pais" : " Italia ",
  "experiencia" : " 10 "
}
{
  "_id" : ObjectId("664f5a1435ee3a886c9dc664"),
  "id_editorial" : " 60aaf1f88206310001528053 ",
  "Nombre" : " Nestor Pitana ",
  "Pais" : " Argentina ",
  "experiencia" : " 8 "
}
}
>
> secondaryTestDB.deportistas.count();
3
> secondaryTestDB.encuentrosDeportivos.count();
5
> secondaryTestDB.entrenadores.count();
3
> secondaryTestDB.equipos.count();
3
>

```

Comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario (Tercer Nodo). Los pasos son iguales a la verificación del nodo #2, la diferencia radica en ingresar al servidor 20002

```
> connSecondary = new Mongo("DESKTOP-5JS0E4P:20002")
```

## Casos de pruebas en replicación bajo MongoDB

Tipo de prueba	Objetivo
Verificación la replicación multimaestro	Se verificará que la replicación multimaestro esté correctamente configurada y funcionando según lo especificado en el documento de requerimientos no funcionales
Disponibilidad de información	Debemos asegurarnos de que se ingresen al menos dos documentos en las colecciones especificadas en el documento de requerimientos en el nodo maestro. Luego, verifica que todas las instancias tengan una réplica de los registros insertados
Prueba de integridad de datos	Verificar que los datos replicados están consistentes en todas las réplicas y no se producen inconsistencias o corrupciones.

## Ejecución casos de prueba

### Verificación de la replicación multimaestro:

Vamos a conectarnos a uno de los nodos secundarios para verificar si se realizo correctamente el proceso de **replicación**

**> connSecondary = new Mongo("DESKTOP-5JS0E4P:20001")**

```
> connSecondary = new Mongo("DESKTOP-5JS0E4P:20001")
connection to DESKTOP-5JS0E4P:20001
>
```

Ahora accedemos a la base de datos

**> secondaryTestDB = connSecondary.getDB("Torneo\_Copa\_Del\_Mundo")**

```
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
>
```

Comprobamos si este nodo es el master o no lo es:

**> secondaryTestDB.isMaster()**

```

ConjuntoReplicas
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20001",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T15:22:03Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T15:22:03Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T15:32:52.842Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 18,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}

```

Ahora que estamos ya conectados a la base de datos en el nodo secundario dentro del grupo de réplica, vamos a comprobar si los datos que hemos insertado en el nodo primario se han replicado en este secundario.

Ahora vamos a activar el permiso para realizar operaciones de lectura sobre un nodo secundario.

**> connSecondary.setSecondaryOk()**

```

> connSecondary.setSecondaryOk()
>

```

Luego de estar activado el permiso para realizar operaciones de lectura, vamos a realizar la consulta sobre la colección “**arbitros**”:

**> secondaryTestDB.arbitros.count();**

```
> secondaryTestDB.arbitros.count();
3
>
> secondaryTestDB.arbitros.find().pretty();
{
  "_id" : ObjectId("664f5a0735ee3a886c9dc662"),
  "id_editorial" : " 60aaf1f88206310001528051 ",
  "Nombre" : " Howard Webb ",
  "Pais" : " Inglaterra ",
  "experiencia" : " 15 "
}
{
  "_id" : ObjectId("664f5a1235ee3a886c9dc663"),
  "id_editorial" : " 60aaf1f88206310001528052 ",
  "Nombre" : " Pierluigi Collina ",
  "Pais" : " Italia ",
  "experiencia" : " 10 "
}
{
  "_id" : ObjectId("664f5a1435ee3a886c9dc664"),
  "id_editorial" : " 60aaf1f88206310001528053 ",
  "Nombre" : " Nestor Pitana ",
  "Pais" : " Argentina ",
  "experiencia" : " 8 "
}
}
```

**En este caso, comprobamos que los datos que insertamos en el nodo principal se han replicado en el nodo secundario.** Este mismo proceso lo realizaremos para cada una de las colecciones, este proceso ya se ha realizado previamente y es válido afirmar que la información del nodo primario está siendo replicada en los otros nodos.

**Caso de prueba: Aprobado**

## Disponibilidad de la información:

Accedemos al nodo maestro

```
ConjuntoReplicas
> conn=new Mongo("DESKTOP-SJS0E4P:20000")
connection to DESKTOP-SJS0E4P:20000
>
```

Una vez obtenida la conexión, vamos a obtener la base de datos donde realizaremos la prueba

```
> testDB=conn.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
>
```

Luego, preguntamos si nos encontramos en el nodo maestro

```
ConjuntoReplicas
> testDB.isMaster()
{
  "hosts" : [
    "DESKTOP-SJS0E4P:20000",
    "DESKTOP-SJS0E4P:20001",
    "DESKTOP-SJS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : true,
  "secondary" : false,
  "primary" : "DESKTOP-SJS0E4P:20000",
  "me" : "DESKTOP-SJS0E4P:20000",
  "electionId" : ObjectId("7fffffff0000000000000001"),
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T15:22:03Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716477723, 4),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T15:22:03Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T18:44:54.484Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 31,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716477723, 4),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
```

Ahora, vamos a insertar un conjunto de datos:

```

> testDB.arbitros.insert(
... {
... id: "60aa1f88206310001528058",
... nombre: "Björn Kuipers",
... pais: "Países Bajos",
... experiencia: 11
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.arbitros.insert(
... {
... id: "60aa1f88206310001528056",
... nombre: "Néstor Fabián",
... pais: "Mexico",
... experiencia: 10
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.equipos.insert(
... {
... id: "60aa1f88206310001528047",
... pais: "Belgica",
... entrenador: "Roberto Martinez"
... });
WriteResult({ "nInserted" : 1 })
>
> testDB.equipos.insert(
... {
... id: "60aa1f88206310001528048",
... pais: "España",
... entrenador: "Luis Enrique"
... });
WriteResult({ "nInserted" : 1 })
>
>

```

Ahora, vamos a verificar que las instancias tengan una replica de los registros insertados.

Vamos a conectarnos a alguno de los nodos secundarios:

```

> connSecondary = new Mongo("DESKTOP-5JS0E4P:20001")
connection to DESKTOP-5JS0E4P:20001
>

```

Accedemos a la base de datos

```

> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
>

```

Comprobamos si este nodo es el master o no lo es

```

ConjuntoReplicas
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20001",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T18:53:15Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T18:53:15Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T18:55:59.021Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 22,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716490395, 2),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}

```

Vamos a activar el permiso para realizar operaciones de lectura sobre un nodo secundario.

```
> connSecondary.setSecondaryOk()
>
```

Podemos observar que el proceso de replicación se cumple, previamente teníamos 3 documentos y sumado a los 2 que acabamos de registrar, daría un total de 5 documentos en la colección **arbitros**

```
> secondaryTestDB.arbitros.count();
5
>
```

Ahora vamos al otro nodo secundario (**Tercer Nodo**)

Accedemos al nodo secundario, posteriormente ingresamos a la base de datos y comprobamos si este nodo o es el master, finalmente damos permisos para realizar operaciones de lectura y finalmente consultamos en la coleccion

```
> connSecondary = new Mongo("DESKTOP-5JS0E4P:20002")
connection to DESKTOP-5JS0E4P:20002
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
ConjuntoReplicas x + v
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20002",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T18:53:15Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T18:53:15Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T19:01:38.348Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 21,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716490395, 2),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
> connSecondary.setSecondaryOk()
> secondaryTestDB.arbitros.count();
5
>
```

Es posible confirmar que el proceso de replicación se cumple, la colección arbitros tiene 5 documentos, eso quiere decir que todas las instancias tienen una réplica de los registros insertados.

**Caso de prueba: Aprobado**



## Prueba de integridad de datos:

Ingresamos al nodo master, luego accedemos a la base de datos, posteriormente verificamos si nos encontramos en el nodo master y procedemos a verificar la cantidad de documentos que tienen las colecciones **encuentrosDeportivos**, **entrenadores** y el contenido de estas colecciones.

```

ConjuntoReplicas
}
> connPrimary = new Mongo("DESKTOP-5JS0E4P:20000")
connection to DESKTOP-5JS0E4P:20000
> primaryDB = connPrimary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
> testDB.isMaster()
{
  "hosts" : [
    "DESKTOP-5JS0E4P:20000",
    "DESKTOP-5JS0E4P:20001",
    "DESKTOP-5JS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : true,
  "secondary" : false,
  "primary" : "DESKTOP-5JS0E4P:20000",
  "me" : "DESKTOP-5JS0E4P:20000",
  "electionId" : ObjectId("7fffffff0000000000000001"),
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T18:53:15Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T18:53:15Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T19:54:32.036Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 31,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
}

> testDB.encuentrosDeportivos.count()
5
> testDB.encuentrosDeportivos.find().pretty()
{
  "_id" : ObjectId("664f5ca335ee3a886c9dc668"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc669"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66a"),
  "id" : "60aaf1f88206310001528059",
  "equipo_local" : "Argentina",
  "equipo_visitante" : "Portugal",
  "arbitro_principal" : "Howard Webb"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66b"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
}
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66c"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}
}
>

```

Ahora, vamos a dirigirnos a nuestra replicas para verificar la calidad y confiabilidad de la información registrada en ellas

## Nodo secundario (Nodo Dos)

```

Mongo-Shell
> connSecondary = new Mongo("DESKTOP-SJS0E4P:20001")
connection to DESKTOP-SJS0E4P:20001
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-SJS0E4P:20000",
    "DESKTOP-SJS0E4P:20001",
    "DESKTOP-SJS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-SJS0E4P:20000",
  "me" : "DESKTOP-SJS0E4P:20001",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T18:53:15Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T18:53:15Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T20:10:15.708Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 25,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1716490395, 2),

```

```

Mongo-Shell
> secondaryTestDB.encuentrosDeportivos.count();
5
> secondaryTestDB.encuentrosDeportivos.find().pretty();
{
  "_id" : ObjectId("664f5ca335ee3a886c9dc668"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc669"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66a"),
  "id" : "60aaf1f88206310001528059",
  "equipo_local" : "Argentina",
  "equipo_visitante" : "Portugal",
  "arbitro_principal" : "Howard Webb"
}
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66b"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
}
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66c"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}
}

```

## Nodo secundario (Nodo Tres)

```

Mongo-Shell
> connSecondary = new Mongo("DESKTOP-SJS0E4P:20002")
connection to DESKTOP-SJS0E4P:20002
> secondaryTestDB = connSecondary.getDB("Torneo_Copa_Del_Mundo")
Torneo_Copa_Del_Mundo
> secondaryTestDB.isMaster()
{
  "hosts" : [
    "DESKTOP-SJS0E4P:20000",
    "DESKTOP-SJS0E4P:20001",
    "DESKTOP-SJS0E4P:20002"
  ],
  "setName" : "replicaSetTorneoCopaMundo",
  "setVersion" : 2,
  "ismaster" : false,
  "secondary" : true,
  "primary" : "DESKTOP-SJS0E4P:20000",
  "me" : "DESKTOP-SJS0E4P:20002",
  "lastWrite" : {
    "opTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "lastWriteDate" : ISODate("2024-05-23T18:53:15Z"),
    "majorityOpTime" : {
      "ts" : Timestamp(1716490395, 2),
      "t" : NumberLong(1)
    },
    "majorityWriteDate" : ISODate("2024-05-23T18:53:15Z")
  },
  "maxBsonObjectSize" : 16777216,
  "maxMessageSizeBytes" : 48000000,
  "maxWriteBatchSize" : 100000,
  "localTime" : ISODate("2024-05-23T20:17:02.924Z"),
  "logicalSessionTimeoutMinutes" : 30,
  "connectionId" : 22,
  "minWireVersion" : 0,
  "maxWireVersion" : 8,
  "readOnly" : false,
  "ok" : 1,
  "$clusterTime" : {
}

Mongo-Shell
> connSecondary.setSecondaryOk()
> secondaryTestDB.encuentrosDeportivos.count();
5
> secondaryTestDB.encuentrosDeportivos.find().pretty();
{
  "_id" : ObjectId("664f5ca335ee3a886c9dc668"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc669"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66a"),
  "id" : "60aaf1f88206310001528059",
  "equipo_local" : "Argentina",
  "equipo_visitante" : "Portugal",
  "arbitro_principal" : "Howard Webb"
},
{
  "_id" : ObjectId("664f5dec35ee3a886c9dc66b"),
  "id" : "60aaf1f8820631000152805a",
  "equipo_local" : "Brasil",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Pierluigi Collina"
},
{
  "_id" : ObjectId("664f5e9735ee3a886c9dc66c"),
  "id" : "60aaf1f8820631000152805b",
  "equipo_local" : "Países Bajos",
  "equipo_visitante" : "Croacia",
  "arbitro_principal" : "Nestor Pitana"
}

```

De esta manera es posible determinar que la integridad de los datos es la adecuada, además fue posible verificar el número de documentos de cada colección y corresponde con la información registrada en el nodo master, asegurándonos que la réplica funciona adecuadamente. Esta verificación nos permite detectar y corregir cualquier inconsistencia o corrupción de datos de manera proactiva, asegurando así la fiabilidad y consistencia de nuestros datos en el entorno de MongoDB.

## Caso de prueba: Aprobado

## **Conclusión**

Estos requerimientos no funcionales son esenciales para asegurar que el sistema de almacenamiento de información de la Copa del Mundo de fútbol sea altamente disponible y redundante, garantizando así un servicio continuo y confiable para los usuarios en todo el mundo.

## Referencias

Rootstack. (s/f). *Replicación en MongoDB*. Rootstack. Recuperado el 24 de mayo de 2024, de <https://rootstack.com/es/blog/replicacion-en-mongodb>

Holcombe, J. (2023, marzo 16). *Construye un Robusto Conjunto de Réplicas MongoDB en Tiempo Récord (4 Métodos)*. Kinsta®; Kinsta. <https://kinsta.com/es/blog/conjunto-de-replicas-mongodb/>