

Materia Bases de datos avanzadas

PROFESOR: WILLIAM RUIZ

Universidad Iberoamericana

Ingeniería en Ciencia de Datos

Actividad 3 - Conceptos y Comandos básicos del
particionamiento en bases de datos NoSQL

Nelson Fernando Nopssa Castro

ID 100169351

Descripción del Escenario

La base de datos de la Fórmula 1 contiene información sobre los circuitos, ganadores, tiempos y fechas de las carreras. Para manejar el aumento continuo de datos, se ha implementado Sharding en MongoDB. Este proceso distribuye los datos en múltiples shards, asegurando escalabilidad y disponibilidad.

Casos de Pruebas

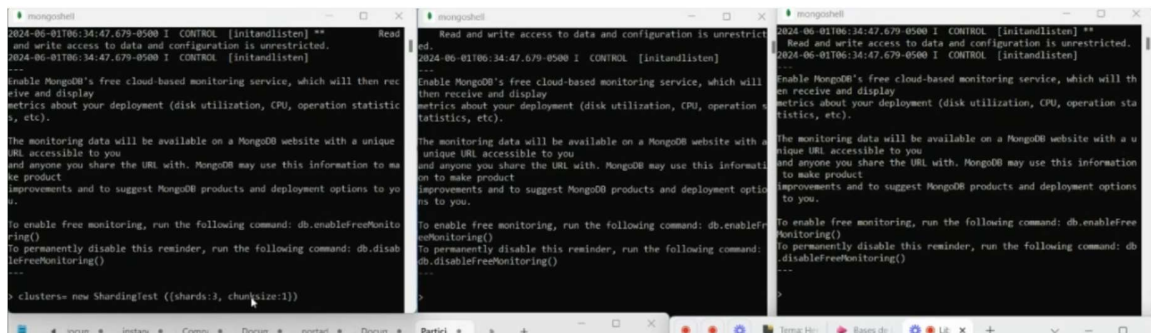
Caso de Prueba 1: Verificación de la Creación del Clúster de Sharding

Descripción: Confirmar que el clúster de Sharding se ha creado correctamente con al menos tres shards.

Pasos:

Iniciar el clúster de Sharding con tres shards y un tamaño de chunk de 1 MB en la consola 1:

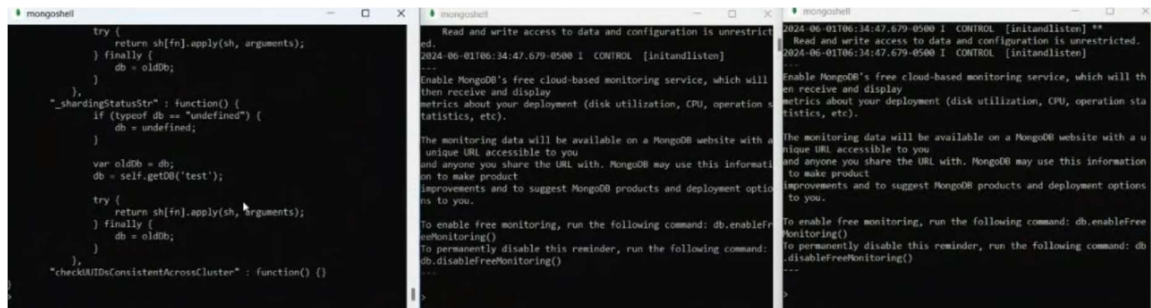
```
clusters = new ShardingTest({shards: 3, chunksize: 1});
```



```
2024-06-01T06:34:47.679-0500 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2024-06-01T06:34:47.679-0500 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()
...
> clusters = new ShardingTest({shards: 3, chunksize: 1})
```

Resultado Esperado: El clúster de Sharding debe ser inicializado correctamente con tres shards.

TERMINACION EN LA CONSOLA DEL COMANDO



```
try {
  return sh[fn].apply(sh, arguments);
} finally {
  db = oldDb;
}
},
"_shardingStatusStr" : function() {
  if (typeof db == "undefined") {
    db = undefined;
  }
  var oldDb = db;
  db = self.getDB('test');
  try {
    return sh[fn].apply(sh, arguments);
  } finally {
    db = oldDb;
  }
},
"checkAllIDsConsistentAcrossCluster" : function() {}
}
```



```

for (i = 0; i < 150000; i++) {
    db.Resultados.insert({
        circuito: "Circuit de Monaco" + i,
        ganador: "Max Verstappen" + i,
        tiempo: "1:44:27" + i,
        fecha: "2024-06-26" + i
    });
}

```

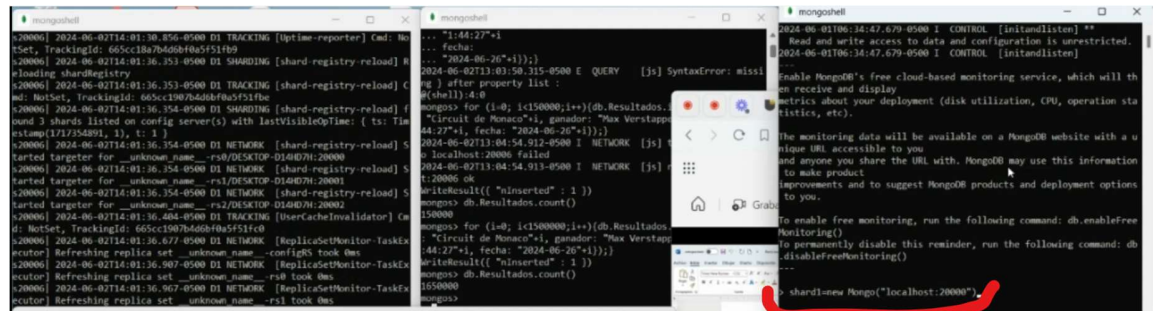
Verificar la cantidad de documentos insertados:

FINALIZADO EL PROCESO

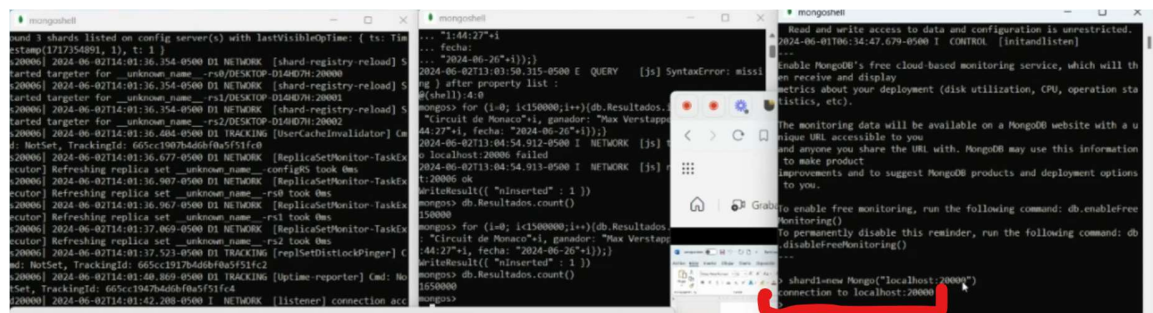
db.Resultados.count();

Pasos:

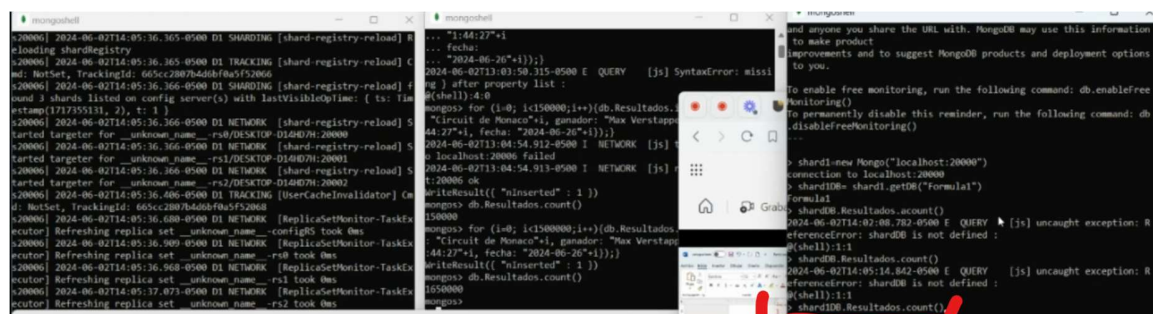
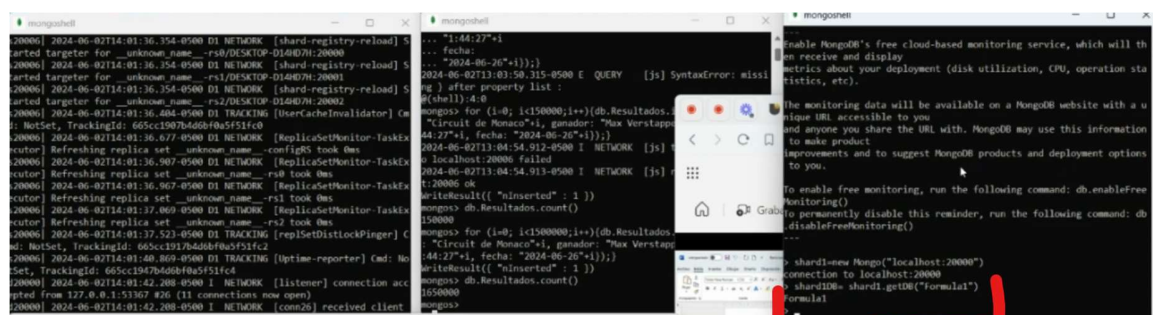
Conectar a cada shard y verificar el conteo de documentos:



```
shard1 = new Mongo("localhost:20000");
```



```
shard1DB = shard1.getDB("Formula1");
```




```
shard2 = new Mongo("localhost:20001");  
shard2DB = shard2.getDB("Formula1");  
shard2DB.Resultados.count();
```

```
shard3 = new Mongo("localhost:20002");  
shard3DB = shard3.getDB("Formula1");  
shard3DB.Resultados.count();
```

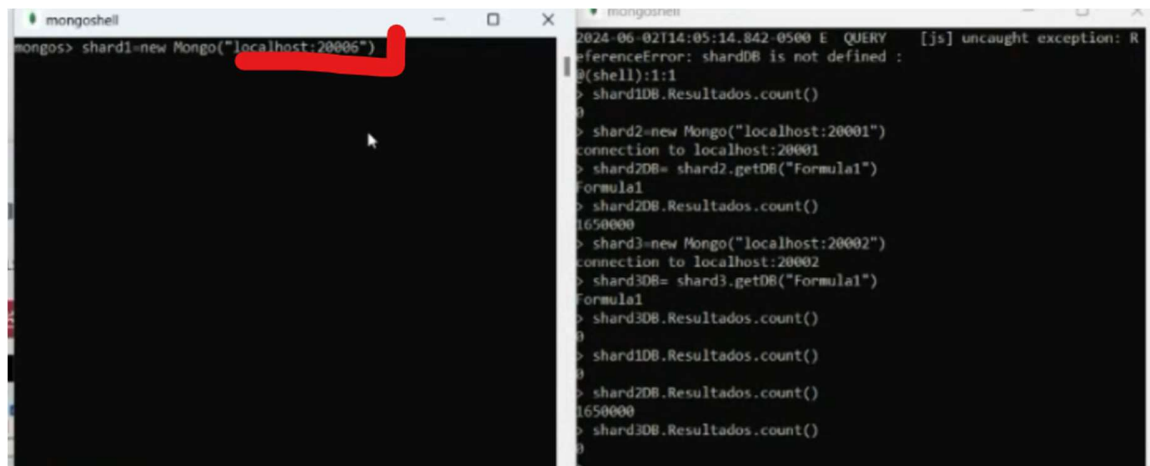
Resultado Esperado: Los documentos deben estar distribuidos entre los shards de manera equitativa.

Caso de Prueba 4: Verificación del Estado y Balanceo de Carga

Descripción: Validar el estado del clúster y el funcionamiento del balanceador.

Pasos:

Verificar el estado del clúster y habilitar el Sharding en la base de datos:



```
mongoshell  
mongos> shard1=new Mongo("localhost:20006")  
  
mongosnet  
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R  
eferenceError: shardDB is not defined :  
9(shell):1:1  
-> shard1DB.Resultados.count()  
0  
-> shard2=new Mongo("localhost:20001")  
connection to localhost:20001  
-> shard2DB= shard2.getDB("Formula1")  
Formula1  
-> shard2DB.Resultados.count()  
1650000  
-> shard3=new Mongo("localhost:20002")  
connection to localhost:20002  
-> shard3DB= shard3.getDB("Formula1")  
Formula1  
-> shard3DB.Resultados.count()  
0  
-> shard1DB.Resultados.count()  
0  
-> shard2DB.Resultados.count()  
1650000  
-> shard3DB.Resultados.count()  
0
```

```
shard1 = new Mongo("localhost:20006");
```



```
mongos> shard1=new Mongo("localhost:20006")
connection to localhost:20006
mongos>
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
referenceError: shardDB is not defined :
0(shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
```

sh.status();

```
mongoshell
mongos> shard1=new Mongo("localhost:20006")
connection to localhost:20006
mongos> sh.status()
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
referenceError: shardDB is not defined :
0(shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
```

```
shards:
  { "_id" : "unknown_name__rs0", "host" : "unknown_
name__rs0/DESKTOP-D14HD7H:20000", "state" : 1 }
  { "_id" : "unknown_name__rs1", "host" : "unknown_
name__rs1/DESKTOP-D14HD7H:20001", "state" : 1 }
  { "_id" : "unknown_name__rs2", "host" : "unknown_
name__rs2/DESKTOP-D14HD7H:20002", "state" : 1 }
active mongoses:
  "4.2.25" : 1
autosplit:
  Currently enabled: no
balancer:
  Currently enabled: no
  Currently running: no
  Failed balancer rounds in last 5 attempts: 0
  Migration Results for the last 24 hours:
    No recent migrations
databases:
  { "_id" : "Formula1", "primary" : "unknown_name__rs
1", "partitioned" : false, "version" : { "uuid" : UUID("d4a22
ef7-0281-4620-996d-c6a4169ad2f8"), "lastMod" : 1 } }
  { "_id" : "config", "primary" : "config", "partitio
```

sh.enableSharding("Formula1");

```
mongoshell
balancer:
  Currently enabled: no
  Currently running: no
  Failed balancer rounds in last 5 attempts: 0
  Migration Results for the last 24 hours:
    No recent migrations

databases:
  { "_id": "Formula1", "primary": "__unknown_name__-rs0", "partitioned": false, "version": { "uuid": "d4a22ef7-0281-4620-996d-c6a4169ad2f8", "lastMod": 1 } }
  { "_id": "config", "primary": "config", "partitioned": true }

config.system.sessions
  shard key: { "_id" : 1 }
  unique: false
  balancing: true
  chunks:
    __unknown_name__-rs0 1
    { "_id" : { "$minKey" : 1 } } --> { "_id" : { "$maxKey" : 1 } } on : __unknown_name__-rs0 Timestamp(1, 0)

mongos> sh.enableSharding("Formula1");
```

```
mongoshell
shard key: { "_id" : 1 }
unique: false
balancing: true
chunks:
  __unknown_name__-rs0 1
  { "_id" : { "$minKey" : 1 } } --> { "_id" : { "$maxKey" : 1 } } on : __unknown_name__-rs0 Timestamp(1, 0)

mongos> sh.enableSharding("Formula1");
{
  "ok" : 1,
  "operationTime" : Timestamp(1717356817, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1717356817, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
```

db.Resultados.ensureIndex({circuito: 1});

```
mongoshell
shard key: { "_id" : 1 }
unique: false
balancing: true
chunks:
  __unknown_name__-rs0 1
  { "_id" : { "$minKey" : 1 } } --> { "_id" : { "$maxKey" : 1 } } on : __unknown_name__-rs0 Timestamp(1, 0)

mongos> sh.enableSharding("Formula1");
{
  "ok" : 1,
  "operationTime" : Timestamp(1717356817, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1717356817, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
```

```
{
  "raw" : {
    "__unknown_name__-rs1/DESKTOP-D14HD7H:20001" : {
      "createdCollectionAutomatically" : false,
      "numIndexesBefore" : 1,
      "numIndexesAfter" : 2,
      "ok" : 1
    },
    "ok" : 1,
    "operationTime" : Timestamp(1717356917, 1),
    "$clusterTime" : {
      "clusterTime" : Timestamp(1717356917, 1),
      "signature" : {
        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAA"),
        "keyId" : NumberLong(0)
      }
    }
  }
}
mongos>
```

```
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
referenceError: shardDB is not defined :
@ (shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
```

sh.shardCollection("Formula1.Resultados", {circuitito: 1});

```
mongos> sh.shardCollection("Formula1.Resultados", {circuitito:1})
```

```
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
referenceError: shardDB is not defined :
@ (shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
```

```
Seleccionar mongoshell
```

```
{
  "circuitito" : { "$minKey" : 1 } --> {
    "circuitito" : "Circuit de Monaco1337493" } on : __unknown_name__-rs1 Timestamp(1, 0)
  } --> {
    "circuitito" : "Circuit de Monaco324995" } on : __unknown_name__-rs1 Timestamp(1, 1)
  } --> {
    "circuitito" : "Circuit de Monaco662495" } on : __unknown_name__-rs1 Timestamp(1, 2)
  } --> {
    "circuitito" : "Circuit de Monaco662495" } on : __unknown_name__-rs1 Timestamp(1, 3)
  } --> {
    "$maxKey" : 1 } on : __unknown_name__-rs0 Timestamp(1, 0)
  }
}
config.system.sessions
shard key: { "_id" : 1 }
unique: false
balancing: true
chunks:
  __unknown_name__-rs0 1
  { "_id" : { "$minKey" : 1 } } --> { "_id" : { "$maxKey" : 1 } } on : __unknown_name__-rs0 Timestamp(1, 0)
```

```
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
referenceError: shardDB is not defined :
@ (shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
```

sh.status();

Verificar y activar el balanceador:

```
sh.getBalancerState();
```

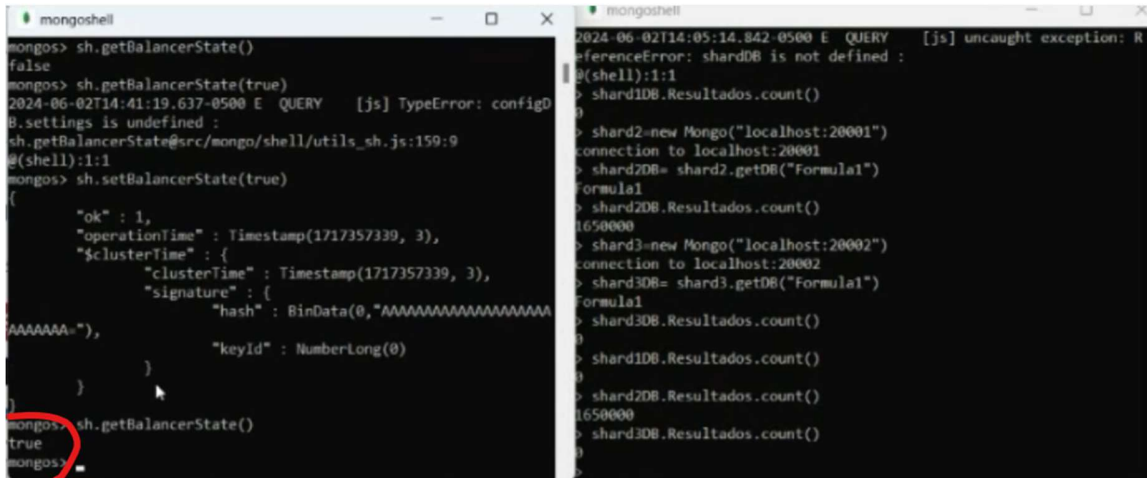
```
sh.setBalancerState(true);
```

```
unique: false
balancing: true
chunks:
  __unknown_name__-rs0 1
  { "_id" : { "$minKey" : 1 } } --> { "_id" : { "$maxKey" : 1 } } on : __unknown_name__-rs0 Timestamp(1, 0)

mongos> sh.getBalancerState()
false
mongos> sh.getBalancerState(true)
2024-06-02T14:40:56.387-0500 E QUERY [js] TypeError: configD
0.settings is undefined :
sh.getBalancerState@src/mongo/shell/utils_sh.js:159:9
@(shell):1:1
mongos> sh.getBalancerState()
false
mongos> sh.getBalancerState(true)
2024-06-02T14:41:19.637-0500 E QUERY [js] TypeError: configD
0.settings is undefined :
sh.getBalancerState@src/mongo/shell/utils_sh.js:159:9
@(shell):1:1
mongos> sh.setBalancerState(true)
```

```
sh.getBalancerState();
```

```
sh.getBalancerState@src/mongo/shell/utils_sh.js:159:9
@(shell):1:1
mongos> sh.getBalancerState()
false
mongos> sh.getBalancerState(true)
2024-06-02T14:41:19.637-0500 E QUERY [js] TypeError: configD
0.settings is undefined :
sh.getBalancerState@src/mongo/shell/utils_sh.js:159:9
@(shell):1:1
mongos> sh.setBalancerState(true)
{
  "ok" : 1,
  "operationTime" : Timestamp(1717357339, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1717357339, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAA
AAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
mongos> sh.getBalancerState()
```


```
mongosh> sh.getBalancerState()
false
mongosh> sh.getBalancerState(true)
2024-06-02T14:41:19.637-0500 E QUERY [js] TypeError: configD
B.settings is undefined :
sh.getBalancerState@src/mongo/shell/utils_sh.js:159:9
@(shell):1:1
mongosh> sh.setBalancerState(true)
{
  "ok" : 1,
  "operationTime" : Timestamp(1717357339, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1717357339, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  }
}
mongosh> sh.getBalancerState()
true
mongosh>
```

```
2024-06-02T14:05:14.842-0500 E QUERY [js] uncaught exception: R
eferenceError: shardDB is not defined :
@(shell):1:1
> shard1DB.Resultados.count()
0
> shard2=new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB= shard2.getDB("Formula1")
Formula1
> shard2DB.Resultados.count()
1650000
> shard3=new Mongo("localhost:20002")
connection to localhost:20002
> shard3DB= shard3.getDB("Formula1")
Formula1
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
0
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
>
```

sh.isBalancerRunning();

Resultado Esperado: El balanceador debe estar activo y distribuyendo la carga equitativamente entre los shards.

Caso de Prueba 5: Verificación de Operaciones Sin Contratiempos



```
> shard2DB.Resultados.count()
1650000
> shard3DB.Resultados.count()
0
> shard1DB.Resultados.count()
412500
> shard2DB.Resultados.count()
1264464
> shard3DB.Resultados.count()
412499
> shard1DB.Resultados.count()
412500
> shard3DB.Resultados.count()
412499
> shard2DB.Resultados.count()
949629
> shard1DB.Resultados.count()
```

Descripción: Asegurar que las operaciones de inserción, consulta y modificación de datos se realizan sin interrupciones.

Pasos:

Realizar operaciones de inserción, consulta y modificación en la base de datos.

Verificar que todas las operaciones se completan con éxito y sin errores.

Resultado Esperado: Todas las operaciones deben completarse sin errores y con un tiempo de respuesta adecuado.