Actividad 6: Pruebas de particionamiento de 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-6---PruebasParticionamientoMongo

Enlace Video

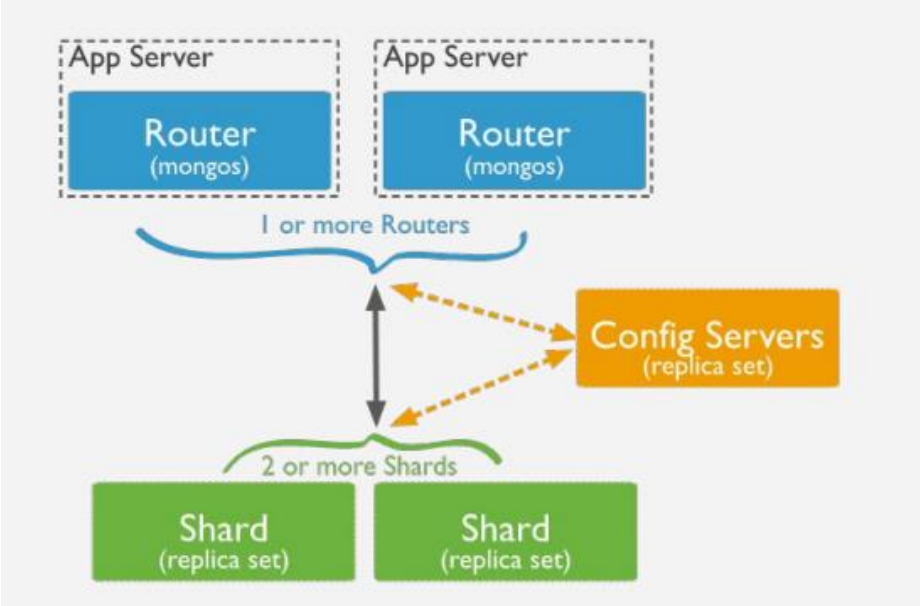
https://youtu.be/T3e2GhMXEso

Sharding

Fragmento: cada fragmento contiene un subconjunto de los datos fragmentados. Cada fragmento se puede implementar como un conjunto de réplicas.

● mongos : mongos actúa como un enrutador de consultas, proporcionando una interfaz entre las aplicaciones cliente y el clúster fragmentado. A partir de MongoDB 4.4, mongos puede admitir lecturas de cobertura para minimizar las latencias.

● Servidores de configuración: los servidores de configuración almacenan metadatos y ajustes de configuración para el clúster.



Casos de prueba Sharding MongoDB

Deben existir 3 archivos de configuracion para los servidores

Puertos: 28017 28117 28217

cfgsvr1.cfg

```
net:
   bindIp: 127.0.0.1
   port: 28017
sharding:
   clusterRole: configsvr
replication:
   replSetName: "replicaConfSvr"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/cfgsvr1/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/cfgsvr1.log"
   logAppend: true
```

cfgsvr2.cfg

```
net:
   bindIp: 127.0.0.1
   port: 28117
sharding:
   clusterRole: configsvr
replication:
   replSetName: "replicaConfSvr"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/cfgsvr2/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/cfgsvr2.log"
   logAppend: true
```

cfgsvr3.cfg

```
net:
   bindIp: 127.0.0.1
   port: 28217
sharding:
   clusterRole: configsvr
replication:
   replSetName: "replicaConfSvr"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/cfgsvr3/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/cfgsvr3.log"
   logAppend: true
```

Deben existir al menos 2 Shard Replica Set, compuestos por su nodo primario, secundario y arbitro.

Puertos: 29017 29117 29217

shard1pri.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29017
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard1"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard1pri/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard1pri.log"
   logAppend: true
```

shard1sec.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29117
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard1"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard1sec/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard1sec.log"
   logAppend: true
```

shard1arb.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29217
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard1"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard1arb/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard1arb.log"
   logAppend: true
```

Se inician los 3 Servers del Shard1

```
mongod --config "C:\Program Files\MongoDB\Server\3.4\etc\shard1pri.cfg" --install --serviceName Mongosh1pri
--serviceDisplayName "MongoDB Shard 1 Primary"
mongod --config "C:\Program Files\MongoDB\Server\3.4\etc\shard1sec.cfg" --install --serviceName Mongosh1sec
--serviceDisplayName "MongoDB Shard 1 Secondary"
mongod --config "C:\Program Files\MongoDB\Server\3.4\etc\shard1arb.cfg" --install --serviceName Mongosh1arb
--serviceDisplayName "MongoDB Shard 1 Arbiter"
```

Conectarse a la interface localhost de uno de los servers del shard1 para inicializar el replica set de este shard.

Con esta configuración he querido forzar cual es el primario por defecto, cual el secundario y cual es árbitro.

```
rs.initiate(
  {
    _id: "replicaShard1",
    members: [
      { _id : 0, host : "localhost:29017", priority : 20 },
      { _id : 1, host : "localhost:29117", priority : 10 },
      { _id : 2, host : "localhost:29217", arbiterOnly : true }
    ]
  }
)
```

Puertos: 29317 29417 29517

shard2pri.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29317
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard2"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard2pri/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard2pri.log"
   logAppend: true
```

shard2sec.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29417
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard2"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard2sec/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard2sec.log"
   logAppend: true
```

shard2arb.cfg

```
net:
   bindIp: 127.0.0.1
   port: 29517
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard2"
storage:
   dbPath: "C:/Program Files/MongoDB/Server/3.4/data/cluster/shard2arb/"
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/shard2arb.log"
   logAppend: true
```

Se inician los 3 Servers del Shard2

```
mongod --config shard2pri.cfg
mongod --config shard2sec.cfg
mongod --config shard2arb.cfg
```

Debe existir un archivo de configuracion para los routers

router1.cfg

```
net:
   bindIp: 127.0.0.1
   port: 27017
sharding:
   configDB: replicaConfSvr/localhost:28017,localhost:28117,localhost:28217
systemLog:
   destination: file
   path: "C:/Program Files/MongoDB/Server/3.4/log/router1.log"
   logAppend: true
```

Debe existir un usuario para la conexión al router

Conectarse a la interface localhost del router

```
admin = db.getSiblingDB("admin")
admin.createUser(
  {
    user: "admin",
    pwd: "Password1",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" }, { role: "clusterAdmin", db: "admin" }, { role:
    "readAnyDatabase", db: "admin" }, { role: "readWriteAnyDatabase", db: "admin" }, { role:
    "userAdminAnyDatabase", db: "admin" } ]
  }
)
```

La replica creada debe pertenecer al fragmento del clúster

Para continuar, debe estar conectado a mongos y autenticado como usuario administrador del clúster para el clúster fragmentado.

```
db.getSiblingDB("admin").auth("admin", "Password1" )
```

```
sh.addShard( "replicaShard1/localhost:29017")
```

Debe estar habilitada la fragmentación para la base de datos del torneo

```
sh.enableSharding("torneo_deportivo")
```

Debe estar Fragmentada una colección

```
sh.shardCollection("torneo_deportivo.deportistas",{"nombre":"Juan"})
```