**100. Creando un Sharded Cluster**

Planificar: ¿Cuantos **Shards** queremos inicialmente en nuestro cluster?

¿Cuales el **Replication Factor** que buscamos en nuestro cluster?

En el ejemplo siguiente, montaremos un **Sharded Cluster** con 2 **Shards** y un **Replication Factor** de 2.

Tened en cuenta que esto lo vamos a montar sobre una unica maquina fisica, pero en **Produccion** deberiamos tener tantas maquinas como procesos **mongod portadores de datos** vayamos a necesitar (Los **Config Servers**, los **mongos** y los **Arbiters** pueden levantarse en maquinas donde ya esten estos procesos) *Vamos a necesitar para este ejemplo:*

• Servidores **mongod portadores de datos** → **ReplicationFactor x ShardsNumber**, en nuestro caso **2 x 2**, es decir, **4**

• Varios procesos **mongos** (por ejemplo 1, son procesos muy ligeros, no hay problema, pero como solo tengo una maquina)

• **Config Servers**, vamos a levantar 2, tambien son muy ligeros, asi que no nos preocupemos. Sobra decir que esta configuracion tiene un fin pedagogico, y no deberia ser aplicada en **Produccion**, ya que no asegura la **disponibilidad**.

Para todo esto, elegiremos una ruta de trabajo (por ejemplo **mongoDB\shardTest**).

• 1 → Creamos los directorios para nuestros **Replica Sets** de cada **Shard** y nuestros **Config Servers**.

Windows y Unix mkdir s0\_0 s0\_1 s1\_0 s1\_1 config\_0 config\_1

• 2 → Levantamos los **Config Servers** (Ojo, a partir de **Mongo 3.2** se pueden crear como **Replica Set**, vamos a hacerlo asi)

Windows start /b mongod --configsvr --dbpath config\_0 --port 26050 --logpath log.cfg0 --logappend --replSet configRS

start /b mongod --configsvr --dbpath config\_1 --port 26051 --logpath log.cfg1 --logappend --replSet configRS

Unix mongod --configsvr --dbpath config\_0 --port 26050 --logpath log.cfg0 –logappend --replSet configRS --fork

mongod --configsvr --dbpath config\_1 --port 26051 --logpath log.cfg1 –logappend --replSet configRS –fork

3 → Levantamos los **Shards**, levantando cada miembro del **Replica Set** de cada **Shard**

Windows start /b mongod --shardsvr --replSet S0 --dbpath s0\_0 --logpath log.s0\_0 --port 27000 -logappend --smallfiles --oplogSize 50

start /b mongod --shardsvr --replSet S0 --dbpath s0\_1 --logpath log.s0\_1 --port 27001--logappend --smallfiles --oplogSize 50

start /b mongod --shardsvr --replSet S1 --dbpath s1\_0 --logpath log.s1\_0 --port 27100 --logappend --smallfiles --oplogSize 50

start /b mongod --shardsvr --replSet S1 --dbpath s1\_1 --logpath log.s1\_1 --port 27101 --logappend --smallfiles --oplogSize 50

Unix mongod --shardsvr --replSet S0 --dbpath s0\_0 --logpath log.s0\_0 --port 27000 --logappend --smallfiles --oplogSize 50 --fork

mongod --shardsvr --replSet S0 --dbpath s0\_1 --logpath log.s0\_1 --port 27001 --logappend --smallfiles --oplogSize 50 --fork

mongod --shardsvr --replSet S1 --dbpath s1\_0 --logpath log.s1\_0 --port 27100 --logappend --smallfiles --oplogSize 50 --fork

mongod --shardsvr --replSet S1 --dbpath s1\_1 --logpath log.s1\_1 --port 27101 --logappend --smallfiles --oplogSize 50 –fork

• 4 → Arrancamos los procesos **mongos**, dando las direcciones de los **Config Servers**. Evitar usar **localhost**. Es muy recomendable levantar los **mongos** en el puerto por defecto de **MongoDB** , el **27017**, ya que no queremos que nadie se conecte por error a un **mongod** (salvo administradores)

Windows start /b mongos --configdb configRS/localhost:26050, localhost:26051 –logpath log.mongos0 --logappend --port 27017

Unix mongos --configdb configRS/localhost:26050,localhost:26051 --logpath log.mongos0 --logappend --port 27017 –fork

• 5 → Iniciamos el **Replica Set** de nuestro **Config Server** (nos conectamos a uno de los 2 y anado el otro. Acordarse de sustituir el **host**)

• 5.1 → Conectarse a un **mongod** del **Config Server** mongo --port 26050

• 5.2 → Iniciar el **Replica Set** y anadir el otro nodo rs.initiate() ; rs.add("localhost:26051")

• 6 → Inicializar uno de los **Shards**• 6.1 → desde el shell **mongo** conectado a **mongos**, nos conectamos a uno de los **Shards**

db=connect("localhost:27000/test")

6.2 → Iniciamos el Replica Set y anadimos el otro miembro del Shard (Ojo, revisar que el hostname que nos ha dado por defecto es correcto, si no, cambiarlo). rs.initiate() //esta parte es solo para cambiar el nombre del host por defecto

cfg=rs.config()

cfg.members[0].host="localhost:27000"

rs.reconfig(cfg) //Fin de cambiar el nombre del host por defecto

rs.add("localhost:27001")

6.3 → Volvemos a conectarnos al **mongos** (tambien vale salir y volver a conectarse), y anadimos el **Shard**

db=connect("localhost:27017/test")

sh.addShard("S0/localhost:27000");

• 7 → Repetimos para el otro **Shards**

7.1 → desde el shell **mongo** conectado a **mongos**, nos conectamos a uno de los **Shards**

db=connect("localhost:27100/test")

7.2 → Iniciamos el Replica Set y anadimos el otro miembro del Shard.

rs.initiate() //esta parte es solo para cambiar el nombre del host por defecto

cfg=rs.config()

cfg.members[0].host="localhost:27100"

rs.reconfig(cfg) //Fin de cambiar el nombre del host por defecto

rs.add("localhost:27101")

7.3 → Volvemos a conectarnos al **mongos** (tambien vale salir y volver a conectarse), y anadimosel **Shard**

db=connect("localhost:27017/test")

sh.addShard("S1/localhost:27100");

Una vez hecho esto, ya tenemos nuestro **Sharded Cluster** levantado y funcionando, aunque **NO** estamos usandolo correctamente, puesto que para ello hay que decir que bases de datos admiten.**Sharding**, que colecciones vamos a trocear.

Podemos confirmar que no estamos haciendo nada porque las siguientes queries no devuelven datos:

use config ; db.chunks.find() ; db.shards.find()

Recordemos que desde **mongos** podemos consultar la configuracion de nuestro **Sharded Cluster** siempre a traves de dicha base de datos, **config**. Las colecciones mas interesantes son **chunks**, **databases**, **locks**, **mongos**, **shards**, **settings**…