Práctica – Conjuntos Réplica en MongoDB

Pasos previos:

a) Abrir un terminal, al que a partir de ahora llamaremos T0. Situarse en una carpeta que nos permita crear subcarpetas (es decir que tenga permisno de escritura).

b) (OJO: lo que sigue esto borrará, si existen, las carpetas data1, data2, data3, data4...asegurarse de que no tienen nada que nos interese). Teclear en T0
rm -rf data1 data2 data3; mkdir data1 data2 data3

c) Abrir 3 terminales adicionales, a los que llamaremos T1, T2, T3, y situarnos en el mismo directorio en el que está T0

1.- [2] En T1, T2, T3 crear servidores parte de un conjunto de réplica con nombre practReplicaxxx con xxx el número de puesto en el que se está trabajando. Además:
- El terminal Ti creará el servidor en el puesto 2700i
- El terminal Ti usará la carpeta ./datai como carpeta de datos.

Desde T0 entrar en el shell mongo con puerto 27001 e iniciar el conjunto réplica (ver apuntes).
Si todo va bien tras unos segundos el shell nos indicará que estamos en el primario de la réplica (al principio puede mostrar “OTHER” o “SECONDARY”; dar enter hasta que aparezca “PRIMARY”).

Copiar en la solución: las instrucciones que han permitido crear los 3 servidores, y la configuración que se ha utilizado desde el shell de mongo en T0 para crear la réplica.

Nota: Al iniciar el conjunto réplica es necesario saber el nombre que se debe usar como “host”. Para tener idea de los nombres una posibilidad es preguntar desde un shell de mongo por el nombre del host con : getHostName(). A este nombre habrá que añadirle “:” y en nombre del puerto.

Solución:
Instrucciones que permiten crear los 3 servidores:
mongod --port 27001 --replSet practReplica001 --dbpath ./data1 --logappend --oplogSize 50
mongod --port 27002 --replSet practReplica001 --dbpath ./data2 --logappend --oplogSize 50
mongod --port 27003 --replSet practReplica001 --dbpath ./data3 --logappend --oplogSize 50

Configuración para crear la réplica:

```
config = {
              _id: "practReplica001",
              members: [ {_id:0, host:"puck.local:27001" },
                         {_id:1, host:"puck.local:27002" },
                         {_id:2, host:"puck.local:27003" }]
          }
```

2.- [1] En el terminal T3 parar el servidor con Ctr-C. ¿Cómo se puede saber desde T0 (shell asociado a 27001, primario) que T3 no está activo? ¿Y desde el sistema operativo linux? ¿Y observando los mensajes del terminal T2? Indicar el comando y la parte de la respuesta que muestra esta información en cada caso (Nota: en el comando del sistema operativo lo que se ve es que falta el proceso).

Solución:

Instrucción en el shell de mongo:
`rs.status()`
Parte de la respuesta que muestra que 27003 está caído:
“name" : "puck.local:27003",
“health" : 0,
"state" : 8,
"stateStr" : "(not reachable/healthy)",

Instrucción en linux:
`lsof -iTCP -sTCP:LISTEN | grep mongo`

¿Cómo se sabe por la respuesta en linux que está caído 27003?
Porque no aparece en la lista

¿Qué mensaje en la consola de T2 muestra que 27003 está caído? (una frase)
`Request to puck.local:27003; HostUnreachable Connection refused`

3.- [1] Seguimos con T3 “caído”. Desde el terminal T0 (primario) escribimos:
use pruebas
db.tururu.insert({dato:"tuturu"},{ writeConcern: { w:3, wtimeout: 1500 }})

Explica en la solución por qué se obtiene el resultado que se puede ver como respuesta.
Solución
Explicación: se obtiene un error porque tras 1.5 segundos no se logra asegurar la escritura en 3 servidores, ya que hay uno caído (T2)

4.- [1] Volvemos a activar T3. Ahora vamos a provocar un fallo en el primario. Sigamos estos pasos. desde el T0, en un shell conectado a 27001 (primario)
a) Ejecutar
use pruebas
db.tururu.drop()
for (i=0;i<1000;i++){db.tururu.insert({dato:"tururu"+i});}
b) Esperar a que se hagan las 1000 inserciones. Comprobar que está bien con db.tururu.find().count() y también con db.tururu.find().pretty()
c) Salir de la shell en T0 (Ctr-d)
d) En T1 acabar con el servidor (Ctr-C).
e) Para simular un fallo de disco buscar en el explorador de archivos la carpeta data1 y borrarla (no hacerlo desde el terminal, mejor desde el explorador).
g) De nuevo en T0 entrar en el shell de mongo, pero esta vez conectando al puerto 27002 y/o al 27003 (probar ambos). ¿Se puede ver desde alguno la colección tururu? ¿Está completa? Da una breve explicación. (Ayuda: no olvidar hacer “`use pruebas"` al entrar)

Solución
Se puede ver la colección tururu de la base de datos pruebas desde 27002 o 27003 (Sí/No)?.Sí
¿Por qué? Al desaparecer el primario, un secundario se ha convertido en primario y permite acceder a los datos, ya que le ha dado tiempo a hacer una réplica.

5) [2]
a) Desde T1 hacer mkdir data1
b) Volvemos a activar el servidor en T1 (puerto 27001).
c) Desde T0, salir a la línea de comandos del terminal
d) Ahora conectar al shell de mongo en 27001
¿Se puede ver la colección directamente tururu de la base de datos pruebas desde 27001?
¿En caso negativo, qué hay que hacer?

Solución
¿Se puede ver la colección tururu de la base de datos pruebas desde 27001? (Sí/No y una frase explicando por qué) No el cliente (shell) solo puede leer en un secundario si este ha indicado que acepta que los datos pueden ser inconsistentes.
¿Cómo se puede habilitar? Con rs.slaveOk().

6) [1] Antes de este paso debemos poner de nuevo los 3 servidores activos, con el del puerto 27001 como primario Desde una consola de mongo (por ejemplo desde el primario), Queremos ejecutar una consulta db.tururu.find(), pero queremos que se ejecute solo sobre cualquier secundario, y si esto no fuera posible sobre el primario. ¿Qué debemos escribir? Copia la instrucción en la solución.

Solución
```
Instrucción: db.pruebas.find().readPref("secondaryPreferred")
```

7) [1] Queremos añadir un nuevo servidor a nuestro conjunto réplica. Para ello:
- Crear una nueva carpeta de datos, data4
- Abrir un nuevo terminal y teclear la instrucción para crear el nuevo servidor, siguiendo las mismas ideas que en el ejercicio 1, pero con un nuevo puerto, por ejemplo 27004.
- Ahora desde el shell de mongo añadimos el nuevo servidor. Copiar esta última instrucción en la solución.
Nota: Para ver el nombre que debemos usar conviene usar rs.status() y fijarse en el “name”, pero cambiando el número de puerto.

Solución
rs.add("puck:27007")

8) [1] Ahora queremos quitar de forma segura el primario. Escribir en la solución la secuencia de instrucciones a seguir, indicando además en qué consola de mongo se ejecutan.

Solución
Desde la consola de 27001: rs.stepDown()
Desde la consola del nuevo primario (en mi caso 27002): rs.remove(“puck:27001”)