IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA
P.J. No. 0428 del 28 de Enero 1982 - MEN | VIGILADA MINEDUCACIÓN

ACTIVIDAD 2 – CUANTIFICADORES Y ESTRUCTURAS MATEMATICAS

NILZON RAMIREZ VILLARREAL

INGENIERA DE SOFTWARE

MATEMATICAS DISCRETAS

2023

REQUERIMIENTOS NO FUNCIONALES

- El sistema debe garantizar siempre la disponibilidad de los datos
- Independientemente de los fallos el sistema siempre deberá mantener disponible la información
- Cada uno de los nodos tendrá acceso a la BD “Participantes”.
- El sistema de replicación como mínimo debe poseer 3 nodos.
- En caso de que el primero nodo genere inaccesibilidad, el sistema generara entre los 2 nodos secundarios, uno primario.
- Para los nodos de réplica contarán con el mismo nombre de servidor (hostname), pero contara con puertos diferentes para el acceso.

Creación de los 3 nodos

```
> MiejReplicaSet = new ReplSetTest ({name: "Participantes", nodes: 3}); print("hecho")
Starting new replica set Participantes
hecho
```

Se arranca los procesos

```
> MiejReplicaSet.startSet()
ReplSetTest starting set
ReplSetTest n is : 0
```

```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20000,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 0,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20001,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 1,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20002,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 2,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

```
connection to LA001TOBSIS010:20000,
connection to LA001TOBSIS010:20001,
connection to LA001TOBSIS010:20002
```

Arranque del proceso de replicación

```
> MiejReplicaSet.initiate()
```

```
{
        "replSetReconfig" : {
                "_id" : "Participantes",
                "protocolVersion" : 1,
                "members" : [
                        {
                                "_id" : 0,
                                "host" : "LA001TOBSIS010:20000"
                        },
                        {
                                "_id" : 1,
                                "host" : "LA001TOBSIS010:20001"
                        },
                        {
                                "_id" : 2,
                                "host" : "LA001TOBSIS010:20002"
                        }
                ],
                "version" : 2
        }
```

Se realiza la prueba del grupo de replica

```
> conn=new Mongo("LA001TOBSIS010:20000")
connection to LA001TOBSIS010:20000
```

Después realizamos la conexión con la BD

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

Se consulta si es el nodo primario

```
> testDB.isMaster()
{
        "hosts" : [
                "LA001TOBSIS010:20000",
                "LA001TOBSIS010:20001",
                "LA001TOBSIS010:20002"
        ],
        "setName" : "Participantes",
        "setVersion" : 2,
        "ismaster" : true,
        "secondary" : false,
        "primary" : "LA001TOBSIS010:20000",
        "me" : "LA001TOBSIS010:20000",
        "electionId" : ObjectId("7fffffff0000000000000001"),
        "lastWrite" : {
                "opTime" : {
                        "ts" : Timestamp(1680355970, 1),
                        "t" : NumberLong(1)
                },
                "lastWriteDate" : ISODate("2023-04-01T13:32:50Z"),
                "majorityOpTime" : {
                        "ts" : Timestamp(1680355970, 1),
                        "t" : NumberLong(1)
                },
```

Insertar un conjunto de datos en el nodo primario.

```
}
> testDB.Participantes.insert({id:"0001",Nombre:"Alejandro",Apellido:"Ramirez Garzon",Edad:"20",Equipo:"Escorpiones"});
WriteResult({ "nInserted" : 1 })
> testDB.Participantes.insert({id:"0001",Nombre:"Alvaro",Apellido:" Garzon Nuñez ",Edad:"22",Equipo:"Escorpiones"});
WriteResult({ "nInserted" : 1 })
```

Se consulta que se haya almacenado los registros

```
> testDB.Participantes.count()
2
```

Comprobación de la replica sobre los nodos secundarios

```
> connSecondary=new Mongo("LA001TOBSIS010:20001")
connection to LA001TOBSIS010:20001
> secondaryTestDB=connSecondary.getDB("Participantes")
Participantes
```

Se comprueba si este nodo es el master o no

```
> secondaryTestDB.isMaster()
{
        "hosts" : [
                "LA001TOBSIS010:20000",
                "LA001TOBSIS010:20001",
                "LA001TOBSIS010:20002"
        ],
        "setName" : "Participantes",
        "setVersion" : 2,
        "ismaster" : false,
        "secondary" : true,
        "primary" : "LA001TOBSIS010:20000",
        "me" : "LA001TOBSIS010:20001",
        "lastWrite" : {
                "opTime" : {
                        "ts" : Timestamp(1680465456, 1),
                        "t" : NumberLong(1)
                },
                "lastWriteDate" : ISODate("2023-04-02T19:57:36Z"),
                "majorityOpTime" : {
                        "ts" : Timestamp(1680465456, 1),
                        "t" : NumberLong(1)
                },
                "majorityWriteDate" : ISODate("2023-04-02T19:57:36Z")
        },
```

Consulta de la información del segundo nodo

```
> connSecondary.setSecondaryOk()
> secondaryTestDB.Participantes.count();
2
> secondaryTestDB.Participantes.findOne()
{
        "_id" : ObjectId("6429de085de00a6795147adc"),
        "id" : "0001",
        "Nombre" : "Alejandro",
        "Apellido" : "Ramirez Garzon",
        "Edad" : "20",
        "Equipo" : "Escorpiones"
}
>
```

Se valida que si se replicó.

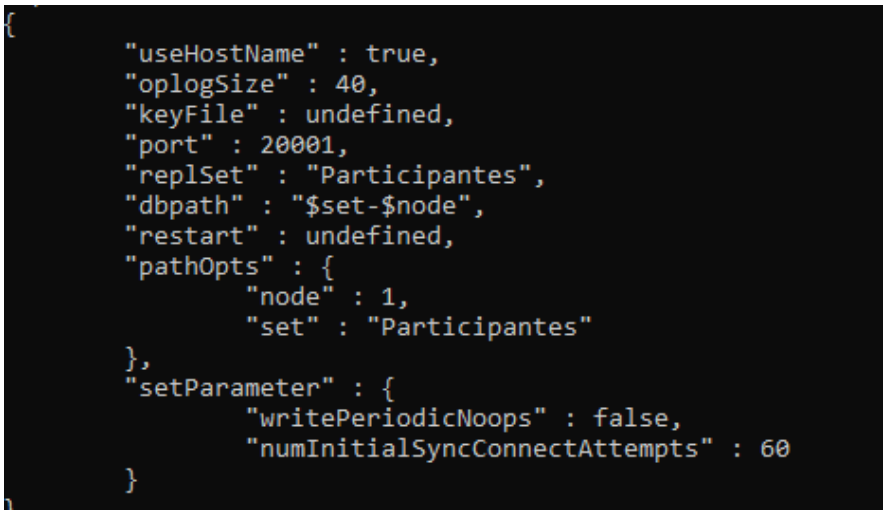
CASOS DE PRUEBA

Replicación: verificar que se hayan creado el nodo primario y los secundarios.

- Nodo primario

```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20000,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 0,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

- Nodos secundarios

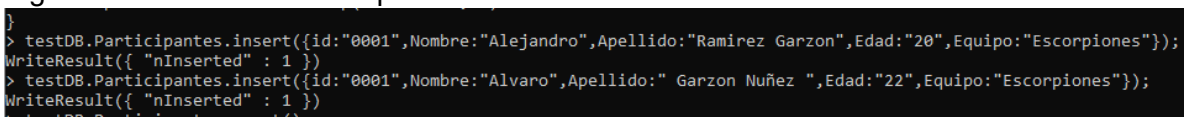


```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20001,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 1,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

```
{
        "useHostName" : true,
        "oplogSize" : 40,
        "keyFile" : undefined,
        "port" : 20002,
        "replSet" : "Participantes",
        "dbpath" : "$set-$node",
        "restart" : undefined,
        "pathOpts" : {
                "node" : 2,
                "set" : "Participantes"
        },
        "setParameter" : {
                "writePeriodicNoops" : false,
                "numInitialSyncConnectAttempts" : 60
        }
}
```

Disponibilidad: Ingresar 2 o más documentos em las colecciones propuestas en el documento de requerimientos en el nodo maestro y verificar que todas las instancias tienen una réplica de los registros insertados.

- Ingreso de datos en el nodo primario



```
}
> testDB.Participantes.insert({id:"0001",Nombre:"Alejandro",Apellido:"Ramirez Garzon",Edad:"20",Equipo:"Escorpiones"});
WriteResult({ "nInserted" : 1 })
> testDB.Participantes.insert({id:"0001",Nombre:"Alvaro",Apellido:" Garzon Nuñez ",Edad:"22",Equipo:"Escorpiones"});
WriteResult({ "nInserted" : 1 })
```

- Consulta de datos en los nodos secundarios

```
> connSecondary.setSecondaryOk()
> secondaryTestDB.Participantes.count();
2
> secondaryTestDB.Participantes.findOne()
{
        "_id" : ObjectId("6429de085de00a6795147adc"),
        "id" : "0001",
        "Nombre" : "Alejandro",
        "Apellido" : "Ramirez Garzon",
        "Edad" : "20",
        "Equipo" : "Escorpiones"
}
>
```

TOLERANCIA A FALLOS: Prueba de desconexión del nodo primario y promoción de algunos de los nodos secundarios a primario.

```
> primaryDB.adminCommand({shotdown:1});
```

DISPONIBILIDAD: Verificar cuál de los nodos secundarios es ahora el nodo primario

```
    "hosts" : [
            "LA001TOBSIS010:20000",
            "LA001TOBSIS010:20001",
            "LA001TOBSIS010:20002"
    ],
    "setName" : "Participantes",
    "setVersion" : 2,
    "ismaster" : false,
    "secondary" : true,
    "primary" : "LA001TOBSIS010:20001",
```

Nuevo nodo primario puerto: 20001