Replica SET.

En MongoDB las réplicas son un grupo de procesos de mongod que mantiene el mismo grupo de datos. El objetivos de las réplicas es poder ofrecer redundancia y alta disponibilidad de servicio, algo totalmente necesario para los entornos de producción.

Redundancia y alta disponibilidad.

Las replicaciones son un mecanismo para proveer redundancia y alta disponibilidad de los datos, gracias a las múltiples copias de los servidores de bases de datos, aumentando significativamente la tolerancia a fallos y pérdidas de conexiones.

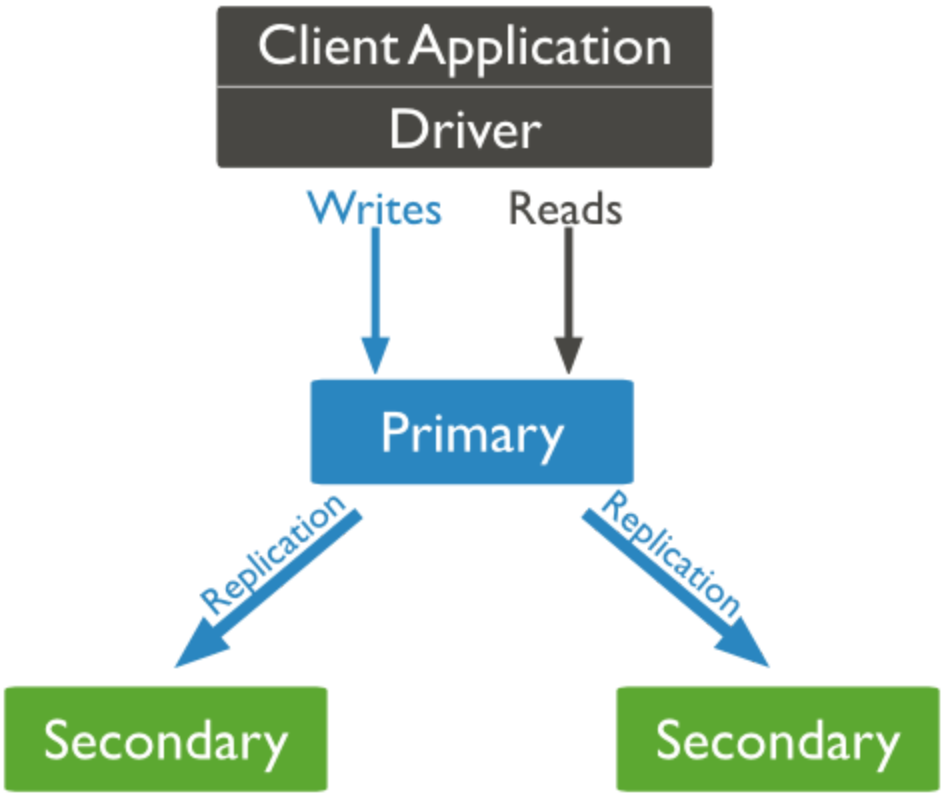
En algunos casos la réplica puede aumentar significativamente la capacidad y velocidad de los procesos de lectura, utilizando distintos servidores para dividir las operaciones en distintos datacenters, además de ofrecer puntos de restauración para la eventual pérdida de la información.

Replicas en MongoDB.

En MongoDB un replicar set es un conjunto de instancias de mongod como bien hemos mencionado anterior, el cual cuenta con una instancia primario y un número de instancias secundarias (al menos 2).

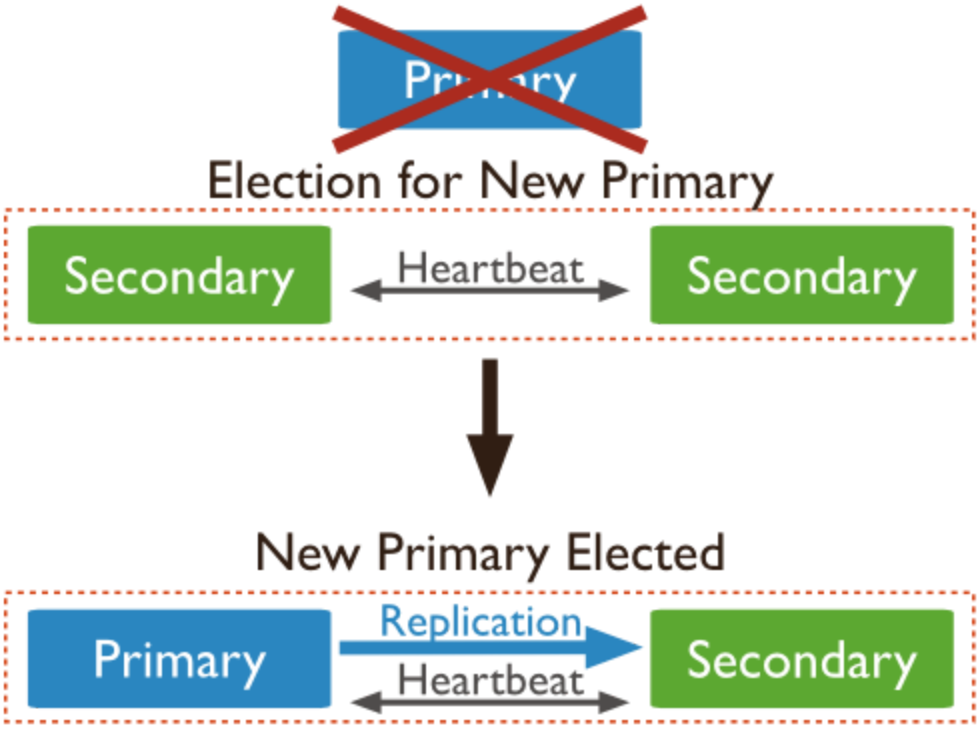
Primary.

La instancia primario recibe todas las operaciones de escritura. Solo podemos tener una instancia primaria por réplica en este modelo. Las operaciones de lectura se hacen sobre el nodo primario o los secundarios (aunque posiblemente no estén con los set de datos actualizados).



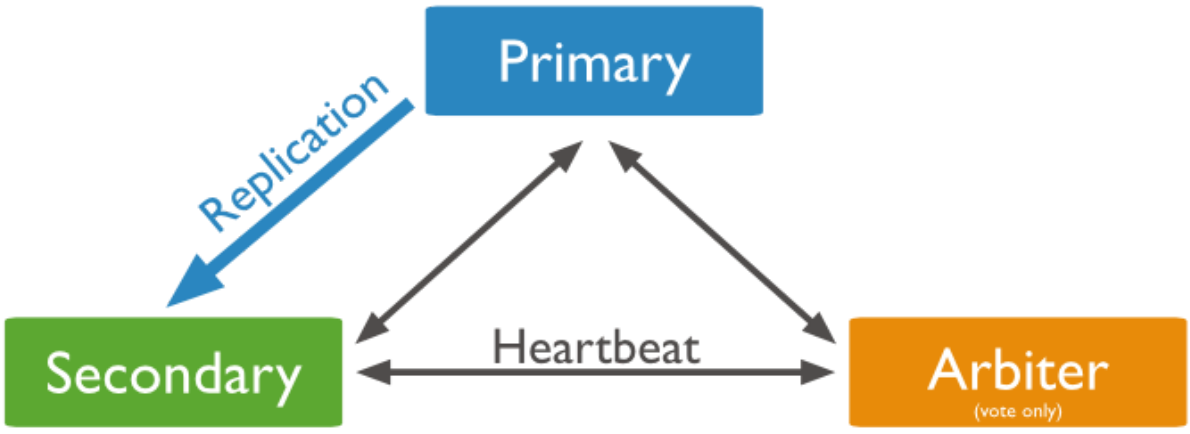
Secundary.

Las instancias secundarias con aquellos miembros del replica set que reciben y mantiene una copia de los datos.
Jamás debemos enviar información a almacenar a las instancias secundarias.



Árbitros.

Los árbitros son una instancia de mongod que tienen como objetivo decidir qué miembro del replica set está en mejores condiciones de ser el nuevo Primary.
Este tipo de instancias no tiene una copia de los datos, su función solo se limita a la de mantener un miembro primario dentro del replica set.



Replicación asincrónica.

La réplica de datos desde el primary a los miembros secundarios se realiza de forma asincrónica, por lo que es importante tenerlo en cuenta a la hora de establecer la lectura sobre un miembro secundario.

Oplog.

El Oplog (Operations log) es una collection del tipo capped que mantiene un registro continuo de todas las operaciones que se fueron dando en el replica set.
Todos los miembros del replicat set toman y aplican esos cambios del Oplog en forma asincrónica. Los miembros mantienen una copia de Oplog en la collection 'local.oplog.rs'.

Para facilitar el proceso de la réplica, todos los miembros envían un ping en forma cruzada, y pueden importar los registro de Oplog desde cualquier miembro del replica set.

Cuando iniciamos un nuevo replica set, MongoDB establece el tamaño de la collection para Oplog.

Storage Engine	Default Oplog Size	Lower Bound	Upper Bound
In-Memory Storage Engine	5% of physical memory	50 MB	50 GB
WiredTiger Storage Engine	5% of free disk space	990 MB	50 GB
MMAPv1 Storage Engine	5% of free disk space	990 MB	50 GB

RS.

El objeto 'rs' de MongoDB es el que nos permite poder crear, configurar y ver el status de un replica set.

Inicio.

Iniciando mongod para replicar set.
bin$ mongod --replSet myrs --dbpath rs1 --port 27501
bin$ mongod --replSet myrs --dbpath rs2 --port 27502
bin$ mongod --replSet myrs --dbpath rs3 --port 27503
bin$ mongo 127.0.0.1:27501
mongo> rs.initiate();
mongo> rs.status();

```
config = {_id: "myrs", members: [
           { _id: 0, host: "localhost:27500"},
           { _id: 1, host: "localhost:27501"},
           { _id: 2, host: "localhost:27502"}
                                ]
         }

rs.initiate(config)
rs.status()
```

Status

Para obtener el status de un replica set:

db.getReplicationInfo()

Para obtener el reporte formateado usar: rs.printReplicationInfo()

Agregar miembro.

Solo recibe 2 argumentos, el primero es el host y puerto, el segundo es si el miembro del réplica set será solo árbitro.

rs.add('host:port', [false/true]);

Eliminar un miembro.

```
rs.remove('host:port')
```