GESTIÓN DE MONGODB

Seguridad en MongoDB – Consideraciones

1. Habilitar el control de acceso y especifique el mecanismo de autenticación.
2. Configurar el control de acceso basado en roles.
3. Cifre la comunicación (TLS/SSL)
4. Cifre y proteja los datos.
5. Limite la exposición de la red.
6. Aplique frecuentemente un sistema de auditoría.
7. Ejecute MongoDB con un usuario dedicado.
8. Ejecute MongoDB con opciones de configuración seguras.
9. Es posible solicitar a MongoDB un STIG (Guía de implementación técnica de seguridad)

Seguridad – Control de Acceso y Autenticación

- Definir un usuario **userAdmin** o un **userAdminAnyDatabase** con el rol de administrador base de datos.
- Este usuario podrá administrar usuarios y roles: crear usuarios, otorgar o revocar roles de usuarios y crear o modificar roles aduaneros.

- **Procedimiento**:

Crear usuario *admin* → **Iniciar control de acceso** → **Pruebe el acceso** → **Cree más usuarios** → **Pruebe los nuevos usuarios**

Seguridad – Control de Acceso y Autenticación

- Definir un usuario **userAdmin** o un **userAdminAnyDatabase** (rol administrador base de datos).

```
1 mongod --port 27017 --dbpath /var/lib/mongodb
2 mongo --port 27017
3 use admin
4 db.createUser(
    {
      user: "myUserAdmin",
      pwd: passwordPrompt(), // pedir contraseña (ponerla directamente)
      roles: [ { role: "userAdminAnyDatabase", db: "admin" },
               "readWriteAnyDatabase" ]
    }
  )
```

Levantar el servidor

```
5 db.adminCommand( { shutdown: 1 } )
6 mongod --auth --port 27017 --dbpath /var/lib/mongodb
7 security:
      authorization: enabled
```

Seguridad – Control de Acceso y Autenticación

- Definir un usuario **myTest** con permisos específicos en la base de datos.

```
mongo --port 27017  --authenticationDatabase "admin" -u "myUserAdmin" -p
```

8

```
mongo --port 27017

use admin
db.auth("myUserAdmin", passwordPrompt()) // pedir contraseña
```

9

```
use test                                    mongo --port 27017 -u "myTester" --authenticationDatabase "test" -p
db.createUser(
  { user: "myTester",
    pwd:  passwordPrompt(),   // or cleartext password
    roles: [ { role: "readWrite", db: "test" },
             { role: "read", db: "reporting" } ]
  }
)
```

read
readWrite
dbAdmin
dbOwner
userAdmin
clusterAdmin
clusterManager
clusterMonitor
hostManager
backup
restore
readAnyDatabase
readWriteAnyDatabase
userAdminAnyDatabase
dbAdminAnyDatabase

Respaldo de la BD

- **Mongodump** lee datos de una base de datos MongoDB y crea archivos BSON de alta fidelidad que la herramienta **Mongorestore** puede usar para completar una base de datos MongoDB.
- **Mongodump** y **Mongorestore** son simples y eficientes para realizar copias de seguridad y restaurar implementaciones pequeñas de MongoDB. No son ideales para capturar copias de seguridad de sistemas más grandes.
- Operan contra un proceso *mongod* en ejecución y pueden manipular los archivos de datos subyacentes directamente.
- **mongodump** solo captura los documentos en la base de datos. La copia de seguridad resultante ahorra espacio, pero **mongorestore** o mongod deben reconstruir los índices después de restaurar los datos.

Respaldo de la BD

- Cuando se conecta a una instancia de MongoDB, **mongodump** puede afectar negativamente al rendimiento de *mongod*. Si sus datos son más grandes que la memoria del sistema, las consultas llevarán al conjunto de trabajo fuera de la memoria, causando así fallos de página.
- Para conjuntos de réplicas, **mongodump** proporciona la opción **--oplog** para incluir en su salida las entradas de registro de operaciones que ocurren durante la operación de **mongodump**. Esto permite que la operación de **mongorestore** correspondiente reproduzca el registro de operaciones capturado.
- Para restaurar una copia de seguridad creada con **--oplog**, use **mongorestore** con la opción **--oplogReplay**.

Respaldo de la BD

- Exportar una base de datos:

```
mongoexport --collection=<name-collection> --db=<name-db> --out=<name-file>.json
```

- Restaurar la base de datos:

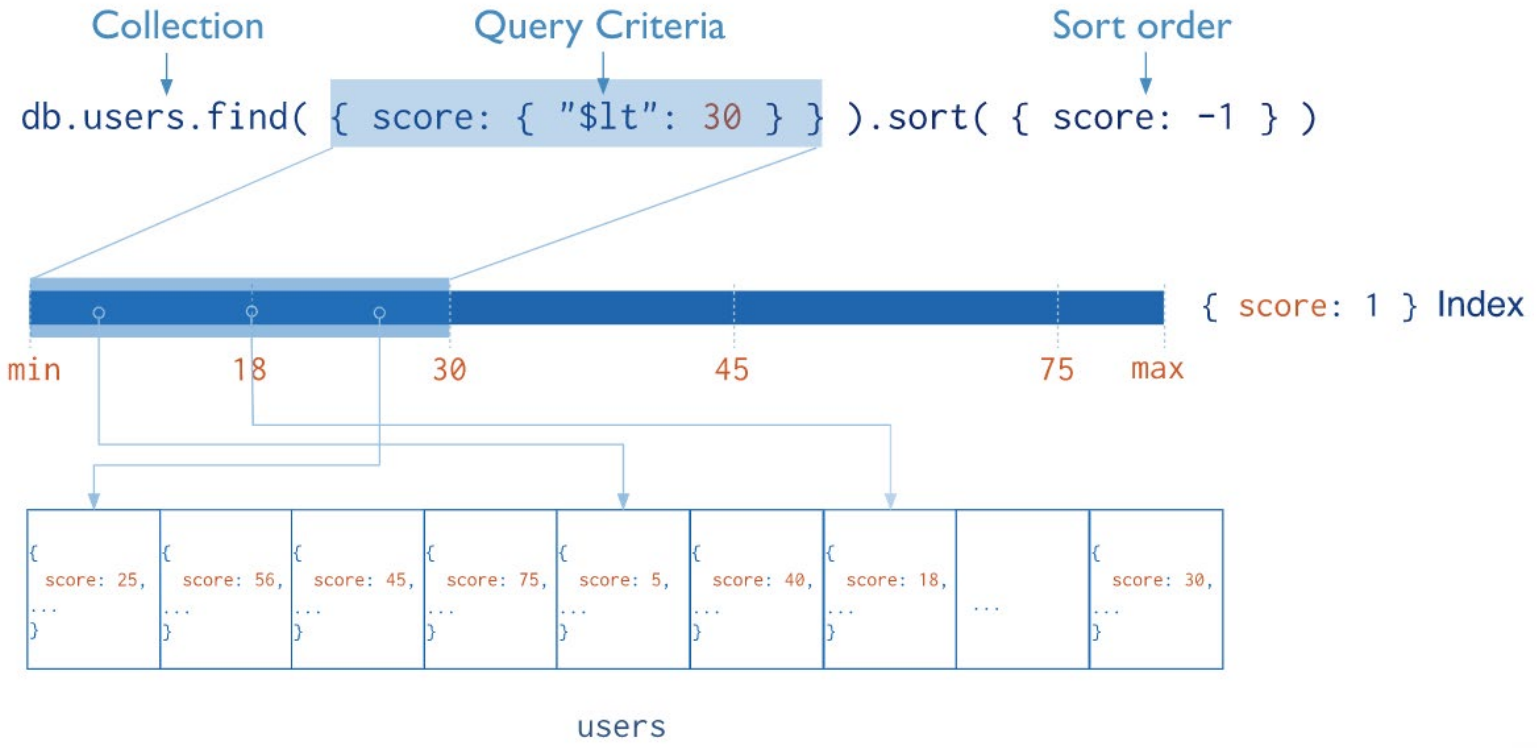
```
mongorestore dump/
mongorestore --nsInclude=test.purchaseorders dump/
mongorestore --db=test --collection=purchaseorders dump/test/purchaseorders.bson
mongorestore --nsInclude='transactions.*' --nsExclude='transactions.*_dev' dump/
mongorestore --nsInclude='data.*' --nsFrom='data.$prefix$_$customer$' -- nsTo='$customer$.$prefix$'
```

- sales_customer1
- sales_customer2
- sales_customer3
- users_customer1
- users_customer2
- users_customer3

Rendimiento - Índices

- Los índices permiten la ejecución eficiente de consultas en MongoDB.
- Sin índices, MongoDB tendría que escanear la colección completa, es decir, recorrer cada documento de una colección, para seleccionar aquellos que coincidan con la declaración de la consulta.
- Si existe un índice apropiado para una consulta, MongoDB puede usar el índice para limitar la cantidad de documentos que debe inspeccionar.
- Los índices son estructuras de datos especiales que almacenan una pequeña porción del conjunto de datos de la colección en una forma fácil de recorrer.
- El índice almacena el valor de un campo específico o conjunto de campos, ordenados por su valor.
- El orden de las entradas de índice admite coincidencias de igualdad eficientes y operaciones de consulta basadas en rangos.

Rendimiento - Índices



Rendimiento - Índices

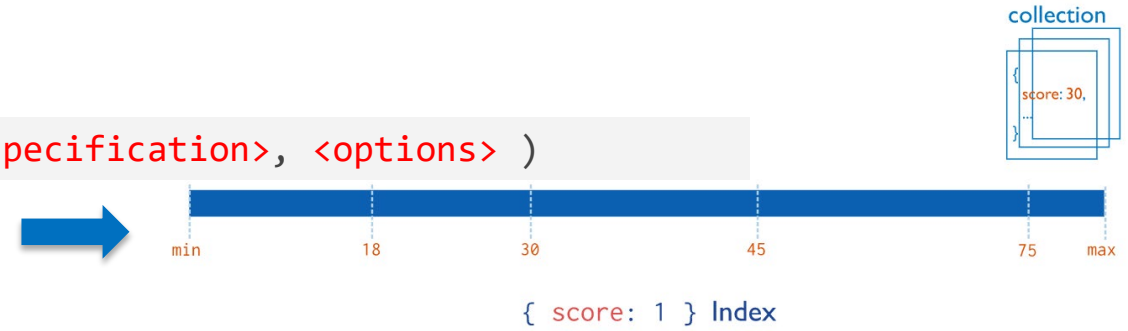
- Los índices en MongoDB son similares a los índices en otros sistemas de bases de datos.
- MongoDB define índices a nivel de colección y admite índices en cualquier campo o subcampo de los documentos en una colección MongoDB.
- MongoDB crea un índice único en el campo _id durante la creación de una colección.
- El índice _id evita que los clientes inserten dos documentos con el mismo valor para el campo _id.
- En clústeres, si no se usa el campo _id como clave de fragmento, su aplicación debe garantizar la unicidad de los valores en el campo _id para evitar errores.
- Debido a lo anterior, una recomendación es utilizar *ObjectId* estándar generado automáticamente.

Rendimiento – Índices Simples

- Crear un índice

```
> db.<collection>.createIndex( <key and index type specification>, <options> )
```

```
> db.<collection>.createIndex( { score: 1 } )
```



- El índice se crea solo si este no existe.
- El nombre predeterminado para un índice es la concatenación de las claves indexadas y la dirección de cada clave en el índice (es decir, 1 o -1) utilizando guiones bajos como separador.
  - ✓ Ejemplo: { item: 1, quantity: -1}, el índice de ítem será item_1_quantity_-1
- Es posible definir nombres de índices personalizados.

```
> db.products.createIndex(
    { item: 1, quantity: -1 },
    { name: "query for inventory" }
  )
> db.products.getIndexes()
```

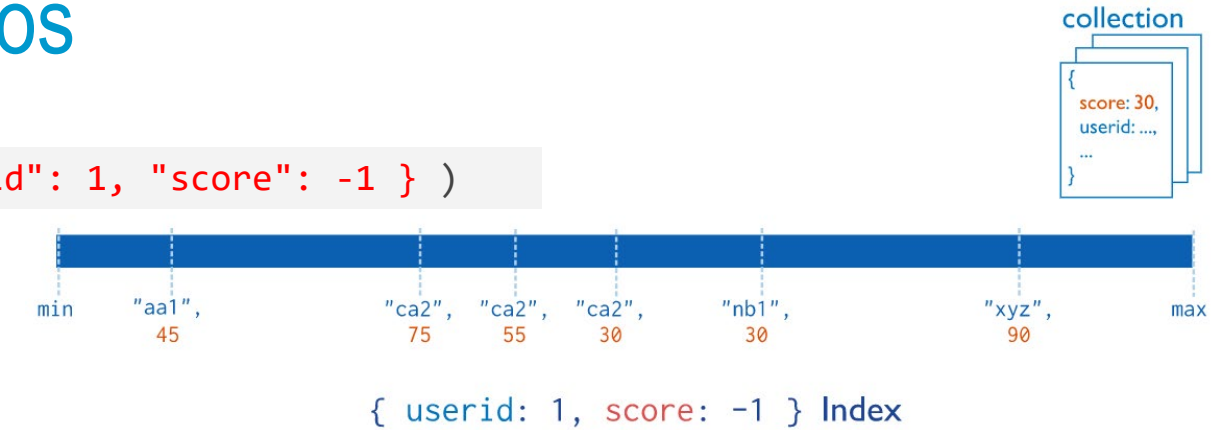
No puede cambiar el nombre de un índice una vez creado. En su lugar, debe borrar y volver a crear el índice con un nuevo nombre.

Rendimiento – Índices Compuestos

- Creación

```
> db.<collection>.createIndex( { “userid": 1, "score": -1 } )
```

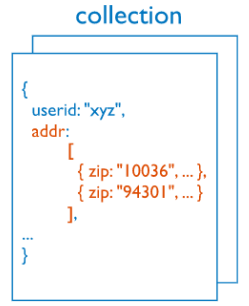
```
> db.users.find({userid: "aa1"})
> db.users.find({userid: "aa1", score:{$gt: 75}})
```



- ✓ El orden de los campos enumerados en un índice compuesto es importante.
- ✓ El índice contendrá referencias a documentos ordenados primero por los valores del campo del artículo y, dentro de cada valor del campo del artículo, ordenados por valores del campo de inventario.
- ✓ Además de admitir consultas que coinciden en todos los campos de índice, los índices compuestos pueden admitir consultas que coinciden en el prefijo de los campos de índice. Es decir, el índice admite consultas en el campo de artículo, así como en los campos de artículo y stock.

Rendimiento – Índices Multikey

- Para indexar un campo que contiene un valor de array, MongoDB crea una clave de índice para cada elemento del array.
- Estos índices multiclaves admiten consultas eficientes contra campos array.
- Los índices Multikey se pueden construir sobre arrays que contienen valores escalares (por ejemplo, cadenas, números) y documentos anidados.




```
> db.<collection>.createIndex({ addr.zip: 1 })
```

{ "addr.zip": 1 } Index

- $expr no soporta los índices multikey.
- MongoDB crea un índice multikey si algún campo indexado es un array.
- Cada documento indexado puede tener máximo un campo array.

Rendimiento – Índices Texto

- MongoDB proporciona índices de texto para admitir consultas de búsqueda de texto en contenido de cadena.
- Los índices de texto pueden incluir cualquier campo cuyo valor sea una cadena o un array de cadenas.
- Una colección puede tener como máximo un índice de texto.
- Para indexar un campo que contiene una cadena o un array de cadenas, incluya el campo y especifique el "*texto*" literal de cadena en el documento de índice.

```
> db.<collection>.createIndex( { comments: "text" }, { default_language: "spanish" } )
```

- Puede indexar múltiples campos para el índice de texto.

```
> db.<collection>.createIndex( { subject: “text”, comments: “text” } )
```

Rendimiento – Índices Consideraciones

Cree índices para respaldar sus consultas:

- Un índice admite una consulta cuando este contiene los campos utilizados por la consulta.

Use índices para ordenar los resultados de la consulta.

- Para admitir consultas eficientes, use como estrategia especificar el orden secuencial y el orden de los campos de índice.

Asegure que los índices encajen en la RAM:

```
> db.<collection>.totalIndexSize()
```

- Cuando su índice encaja en la RAM, el sistema puede evitar leer el índice del disco y obtener el procesamiento más rápido.

Cree consultas que garanticen la selectividad: capacidad de una consulta para limitar los resultados utilizando el índice.

Sharding

Es un método para distribuir datos en varias máquinas. MongoDB usa Sharding para admitir implementaciones con conjuntos de datos muy grandes y operaciones de alto rendimiento.

Hay dos métodos para abordar el crecimiento del sistema: escalamiento vertical y horizontal.

- **El escalado vertical** implica aumentar la capacidad de un solo servidor, como usar una CPU más potente, agregar más RAM o aumentar la cantidad de espacio de almacenamiento.
- **El escalado horizontal** implica dividir el conjunto de datos del sistema y la carga en varios servidores, agregando servidores adicionales para aumentar la capacidad según sea necesario.
- MongoDB admite escalado horizontal mediante Sharding. Conjunto de datos persistente utilizado por un sistema de software.

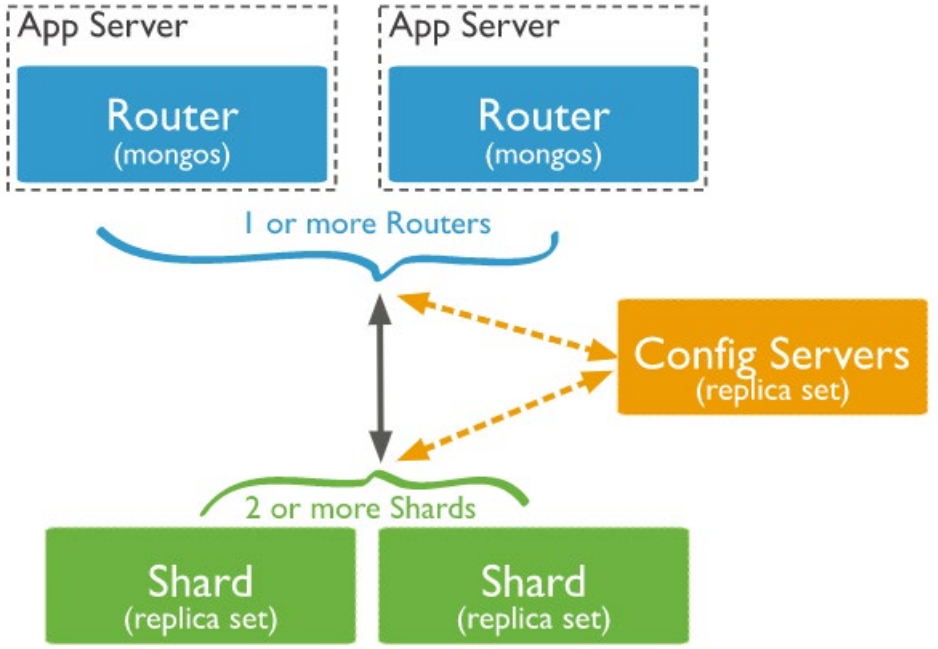
Sharding cluster

Un **sharded clúster** de MongoDB consta de los siguientes componentes:

- **Shard**: 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 pueden admitir lecturas de cobertura para minimizar las latencias.
- **Servidores de configuración**: los servidores de configuración almacenan metadatos y ajustes de configuración para el clúster.

Sharding clúster

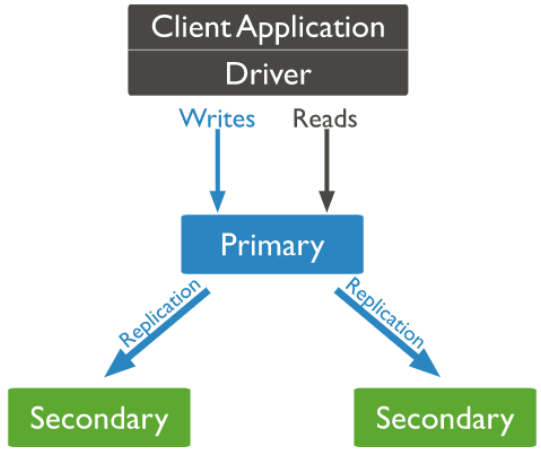
MongoDB fragmenta los datos en el nivel de recopilación, distribuyendo los datos de recopilación entre los fragmentos del clúster.



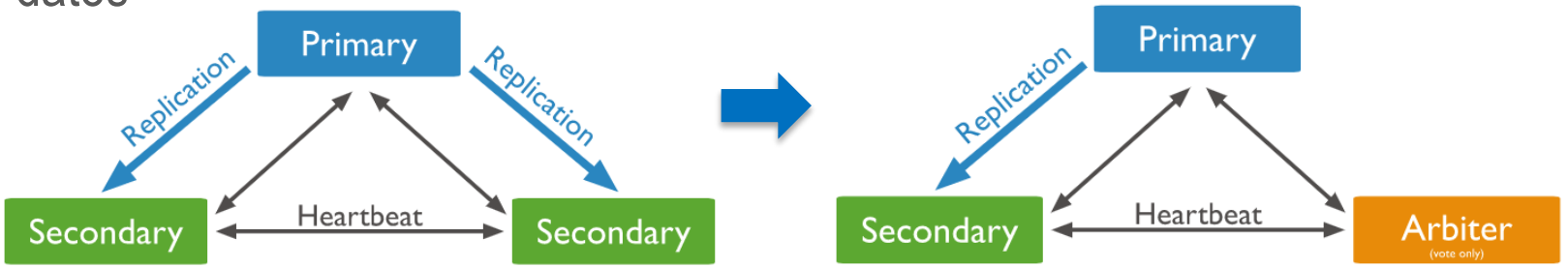
Replica Set

- Un conjunto de réplicas en MongoDB es un grupo de procesos mongod que mantienen el mismo conjunto de datos.
- Los conjuntos de réplicas proporcionan redundancia y alta disponibilidad, y son la base para todas las implementaciones de producción.
- Con múltiples copias de datos en diferentes servidores de bases de datos, la replicación proporciona un nivel de tolerancia a fallas contra la pérdida de un solo servidor de bases de datos.
- De los nodos portadores de datos, uno y solo un miembro se considera el nodo principal, mientras que los otros nodos se consideran nodos secundarios.
- El nodo principal recibe todas las operaciones de escritura.
- Un conjunto de réplicas solo puede tener un primario capaz de confirmar escrituras con preocupación de escritura {w: "majority"}.

Replica Set



- Los secundarios replican el registro de operaciones primario y aplican las operaciones a sus conjuntos de datos.
- Si la primaria no está disponible, una secundaria elegible llevará a cabo una elección para elegir la nueva primaria.
- En algunas circunstancias puede optar por agregar una instancia mongod a un conjunto de réplicas como árbitro.
- Un árbitro siempre será un árbitro, participa en las elecciones pero no retiene datos



Muchas gracias por tu atención