



Performance en las Queries

MongoDB



Historial de revisiones

VERSIÓN	FECHA	AUTOR – PRINCIPALES CAMBIOS
1.0	10/29/2015	Mvargas – Versión inicial

Índice

1	Objetivos del Documento	3
2	¿Qué es MongoDB?	4
3	Instalación en Windows	5
3.1	Correr MongoDB en Windows	5
3.2	Conectarse al motor	6
4	Optimizando Queries	7
4.1	Crear índices para soportar las queries	7
4.2	Algunas estrategias a la hora de generar índices	8
4.3	Ordenar el resultado de una Query – Sort	9
4.4	Limitar el Número de resultados	10
4.5	Forzando a MongoDB a usar un índice particular	10

1 Objetivos del Documento

Con este documento se intenta proporcionar un acercamiento al motor de Base de Datos MongoDB y conocer a grandes rasgos sus características más importantes. Sin embargo, el objetivo principal por el cual se confecciona este documento es para tener una noción detallada de cuales con los puntos a tener en cuenta a la hora de ser conscientes de realizar queries perforantes contra el motor.

2 ¿Qué es MongoDB?

MongoDB es una base de datos documental de código abierto diseñada para facilitar el proceso de desarrollo proveyendo alta performance, alta disponibilidad y fácil escalabilidad.

El motor almacena n cantidad de bases de datos, cada base de datos almacena un set de colecciones y cada colección almacena un set de documentos. Un documento es un par clave-valor.

Los documentos son dinámicos, lo que significa que los documentos almacenados en una colección no necesariamente deben tener el mismo set de campos ni que deben compartir la misma estructura e incluso los campos comunes almacenados en distintos documentos no necesariamente tienen que ser del mismo tipo.

Un documento en MongoDB es un registro cuya estructura de datos se compone de pares clave-valor. Estos documentos son similares a un objeto JSON. Los valores de los campos pueden incluir otros documentos, listas y/o listas de documentos.

El siguiente es un ejemplo de un documento:

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```



← field: value
← field: value
← field: value
← field: value

3 Instalación en Windows

1. **Descargar el instalador del motor:** para ello, acceder al siguiente link <https://www.mongodb.org/downloads? ga=1.217638486.1765182487.1446143360#production>.
 1. Instalar el motor:
 2. Aceptar los términos y condiciones.
 3. Seleccionar instalación completa.

NOTA: El instalador no solicitará mayor información.

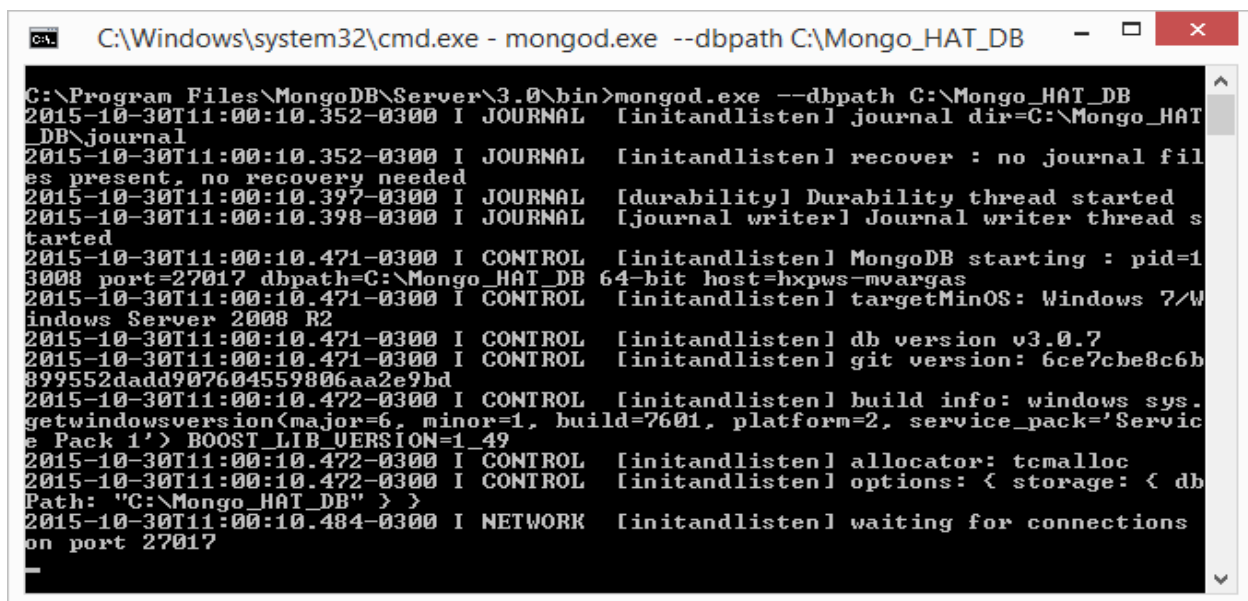
3.1 Correr MongoDB en Windows

2. Crear un ambiente:
3. Mongo requiere de la creación de un directorio donde almacenar los datos. Mongo posee un directorio por defecto, pero también es posible especificar uno:

```
C:\Program Files\MongoDB\Server\3.0\bin>mongod.exe --dbpath C:\Mongo_HAT_DB
```

O bien, si la ruta tiene espacios:

```
C:\Program Files\MongoDB\Server\3.0\bin>mongod.exe --dbpath "C:\Mongo DB"
```

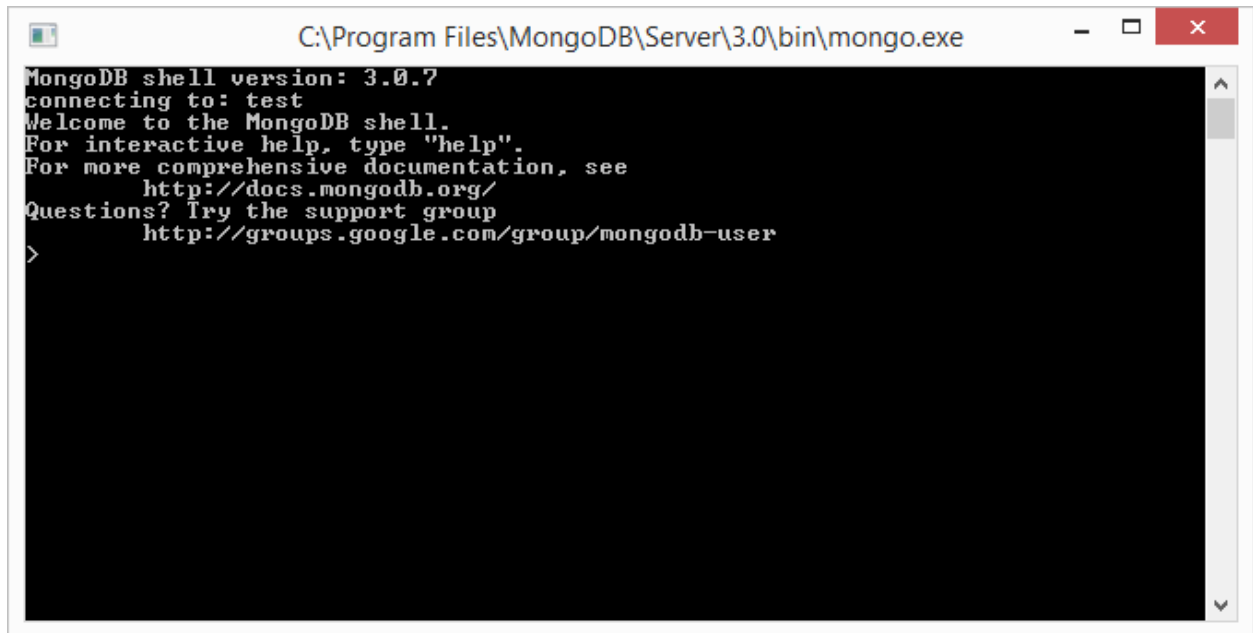


```
C:\Windows\system32\cmd.exe - mongod.exe --dbpath C:\Mongo_HAT_DB

C:\Program Files\MongoDB\Server\3.0\bin>mongod.exe --dbpath C:\Mongo_HAT_DB
2015-10-30T11:00:10.352-0300 I JOURNAL [initandlisten] journal dir=C:\Mongo_HAT
_DB\journal
2015-10-30T11:00:10.352-0300 I JOURNAL [initandlisten] recover : no journal fil
es present, no recovery needed
2015-10-30T11:00:10.397-0300 I JOURNAL [durability] Durability thread started
2015-10-30T11:00:10.398-0300 I JOURNAL [journal writer] Journal writer thread s
tarted
2015-10-30T11:00:10.471-0300 I CONTROL [initandlisten] MongoDB starting : pid=1
3008 port=27017 dbpath=C:\Mongo_HAT_DB 64-bit host=hxpws-mvargas
2015-10-30T11:00:10.471-0300 I CONTROL [initandlisten] targetMinOS: Windows 7/W
indows Server 2008 R2
2015-10-30T11:00:10.471-0300 I CONTROL [initandlisten] db version v3.0.7
2015-10-30T11:00:10.471-0300 I CONTROL [initandlisten] git version: 6ce7cbe8c6b
899552dadd907604559806aa2e9bd
2015-10-30T11:00:10.472-0300 I CONTROL [initandlisten] build info: windows sys.
getwindowsversion(major=6, minor=1, build=7601, platform=2, service_pack='Servic
e Pack 1') BOOST_LIB_VERSION=1_49
2015-10-30T11:00:10.472-0300 I CONTROL [initandlisten] allocator: tcmalloc
2015-10-30T11:00:10.472-0300 I CONTROL [initandlisten] options: { storage: { db
Path: "C:\Mongo_HAT_DB" } }
2015-10-30T11:00:10.484-0300 I NETWORK [initandlisten] waiting for connections
on port 27017
```

3.2 Conectarse al motor

Simplemente correr el archivo **mongo.exe**.

A screenshot of a Windows command prompt window titled "C:\Program Files\MongoDB\Server\3.0\bin\mongo.exe". The window has a black background with white text. The text inside the window reads: "MongoDB shell version: 3.0.7", "connecting to: test", "Welcome to the MongoDB shell.", "For interactive help, type 'help'.", "For more comprehensive documentation, see", "http://docs.mongodb.org/", "Questions? Try the support group", "http://groups.google.com/group/mongodb-user", and a prompt character ">".

```
C:\Program Files\MongoDB\Server\3.0\bin\mongo.exe
MongoDB shell version: 3.0.7
connecting to: test
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
http://docs.mongodb.org/
Questions? Try the support group
http://groups.google.com/group/mongodb-user
>
```

4 Optimizando Queries

Se crearon dos colecciones en MongoDB **HAT_TEST_INDEXED** y **HAT_TEST_NOT_INDEXED** cada uno posee 2 millones de documentos (registros). Esto es con el objeto de realizar comparaciones con las queries que se lanzan al motor.

4.1 Crear índices para soportar las queries

Buscar un índice es más rápido que buscar por elementos en la colección. La estructura de índices es más pequeña que la referencia a los documentos y además están ordenados.

Query sin indexar por el campo **AGE**:

```
Script 1 x
1 db.HAT_TEST_NOT_INDEXED.find({age: 54}).explain('executionStats')

Result
22 "executionStats" : {
23   "executionSuccess" : true,
24   "nReturned" : 77079,
25   "executionTimeMillis" : 54625,
26   "totalKeysExamined" : 0,
27   "totalDocsExamined" : 2000000,
28   "executionStages" : {
29     "stage" : "COLLSCAN",
30     "filter" : {
31       "age" : {
32         "$eq" : 54
33       }
34     },
35     "nReturned" : 77079,
36     "executionTimeMillisEstimate" : 52430,
37     "works" : 2000077,
```

Query indexado por el campo **AGE**:

```
Script 1 x
1 db.HAT_TEST_INDEXED.find({age: 54}).explain('executionStats')

Result
30 "executionStats" : {
31   "executionSuccess" : true,
32   "nReturned" : 77094,
33   "executionTimeMillis" : 156,
34   "totalKeysExamined" : 77094,
35   "totalDocsExamined" : 77094,
36   "executionStages" : {
37     "stage" : "FETCH",
38     "nReturned" : 77094,
39     "executionTimeMillisEstimate" : 130,
40     "works" : 77095,
```


NOTA: recordar que MongoDB no maneja múltiples índices, esto significa que para una query, solo un índice será usado. Por lo que está bien pensar en índices compuestos de varios campos. Así mismo es altamente aconsejable tener la menor cantidad posible de índices, por lo que hay que armar los mismos entendiendo bien el tipo de queries que se corren.

4.2 Algunas estrategias a la hora de generar índices

- > **Index Early:** Los índices merecen la consideración de primer orden en el proceso de diseño. Eficiencia en el nivel de acceso a datos históricamente se ha descargado a un rol DBA que impulsa una estrategia de optimización en capas post-diseño. Para el caso de las bases de datos orientadas a documentos todavía tiene la oportunidad de evitar.
- > **Index Often:** Las queries que ha sido indexadas tienen mejor performance que aquellas que no debido al orden de magnitud, incluso para pequeños bloques de datos. Esto es así siempre y cuando el índice esté debidamente creado.
- > **Index Fully:** Las queries hacen uso de los índices de izquierda a derecha. Un índice sólo se puede utilizar en la medida en que una query utiliza todos los campos en el índice sin omitir ninguna.
- > **Index Sorts:** Si las queries tienen una clausula **sort** u **orderby**, es aconsejable que el campo por el que se está ordenando este indexado.
- > **Comandos:** Siempre, ante una nueva query es aconsejable correr alguno de los siguientes comandos.
 - `.explain()` – muestra entre otras cosas, qué índice (si aplica) se está usando para una query determinada.
 - `.ensureIndex()` – crea un índice en base a los campos que se seleccionan.
 - `.getIndexes()` or `.getIndexKeys()` – lista los índices que actualmente han sido creados.

NOTA: Recordar que la creación de demasiados índices traerá problemas de performance, lo que supone un problema. Es necesario tener siempre presente que optimizar la creación de índices es la mejor opción.

El orden de los campos en un índice tiene importancia. Se sugiere tener en cuenta el siguiente orden:

- > Campos sobre los que se harán queries buscando por valores exactos.
- > Campos sobre los que se va a ordenar.
- > Campos sobre los que se harán queries buscando por rangos de valores.

4.3 Ordenar el resultado de una Query – Sort

El ordenamiento puede resultar una operación costosa. Para que esto sea lo más óptimo posible, hay que tener en cuenta que el ultimo campo que aparece en la query, debería formar parte del índice por el cual se está ordenando.

Query indexado por el campo **AGE**:

```
Script 1 ✖
1 db.HAT_TEST_INDEXED.find({'name.full': 'Kassandra Bohm'}).sort({'age':1}).explain('executionStats')

Result
62 "executionStats" : {
63   "executionSuccess" : true,
64   "nReturned" : 7841,
65   "executionTimeMillis" : 72771,
66   "totalKeysExamined" : 7841,
67   "totalDocsExamined" : 7841,
68   "executionStages" : {
69     "stage" : "SORT",
70     "nReturned" : 7841,
71     "executionTimeMillisEstimate" : 4640,
72     "works" : 22017,
```

Query sin indexar por el campo **AGE**:

```
Script 1 ✖
1 db.HAT_TEST_NOT_INDEXED.find({'name.full': 'Kassandra Bohm'}).sort({'age':1}).explain('executionStats')

Result
28 "executionStats" : {
29   "executionSuccess" : true,
30   "nReturned" : 7864,
31   "executionTimeMillis" : 15422,
32   "totalKeysExamined" : 0,
33   "totalDocsExamined" : 2000000,
34   "executionStages" : {
35     "stage" : "SORT",
36     "nReturned" : 7864,
37     "executionTimeMillisEstimate" : 14900,
38     "works" : 2007923,
```

4.4 Limitar el Número de resultados

MongoDB retorna los resultados en grupos de múltiples documentos. Si se sabe de antemano la cantidad de resultados que se desean, se puede reducir la demanda de recursos simplemente usando el método **limit()**

Query indexado por el campo **AGE**:

```
Script 1 x
1 db.HAT_TEST_INDEXED.find({'name.full': 'Kassandra Bohm'}).limit(1000).explain('executionStats')

Result
34   "executionStats" : {
35     "executionSuccess" : true,
36     "nReturned" : 101,
37     "executionTimeMillis" : 0,
38     "totalKeysExamined" : 101,
39     "totalDocsExamined" : 101,
40     "executionStages" : {
41       "stage" : "LIMIT",
42       "nReturned" : 101,
43       "executionTimeMillisEstimate" : 0,
44       "works" : 102,
```

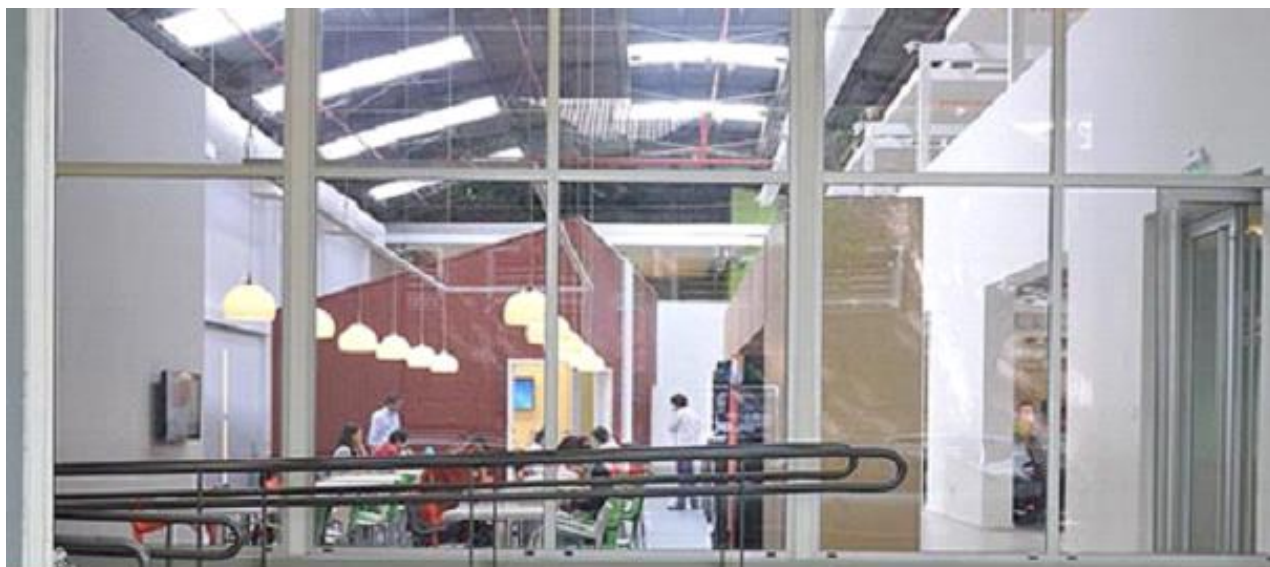
Query sin indexar por el campo **AGE**:

```
Script 1 x
1 db.HAT_TEST_NOT_INDEXED.find({'name.full': 'Kassandra Bohm'}).limit(1000).explain('executionStats')

Result
26   "executionStats" : {
27     "executionSuccess" : true,
28     "nReturned" : 101,
29     "executionTimeMillis" : 32,
30     "totalKeysExamined" : 0,
31     "totalDocsExamined" : 24096,
32     "executionStages" : {
33       "stage" : "LIMIT",
34       "nReturned" : 101,
35       "executionTimeMillisEstimate" : 20,
36       "works" : 24098,
```

4.5 Forzando a MongoDB a usar un índice particular

En la mayoría de los casos, el **query optimizer** (procesador de queries) selecciona el índice más óptimo para una operación específica. Sin embargo, se puede forzar al motor a que utilice un índice específico llamando al método **hint()**. Es aconsejable usarlo cuando un campo está en más de un índice o cuando se está buscando un campo específico.



 HexactaArg

 @Hexacta



www.hexacta.com