

Empezando con MongoDB: una base de datos noSQL de documentos

Profesor: Héctor Gómez Gauchía

0.- IMPORTANTE : tus datos de las BDs las almacena en disco local, **que se borra todos los días**

- Recuerda salvar tu BD antes de irte del laboratorio: ver al final del documento: mongodump y mongorestore
- Hay una alternativa: usar un servidor público: pero tu BD la ve todo el mundo y la puede modificar.

1.- Inicia el servidor ejecutando en el Laboratorio: (cada PC tiene un servidor)

(ver al final si lo haces desde tu portatil)

Ejecuta Programación -> BBDD -> Servidor MongoDB

Se abre una ventana cmd de Windows.

Esperar a que aparezca en los mensajes, una línea para estar preparado:

"... waiting for connections on port 2717" y otros mensajes (no hay todavía conexiones)

2.- Inicia el Cliente ejecutando en el Laboratorio: dos modos

a) **(usar este)** Modo gráfico: ejecuta Robo3T en: Programación -> BBDD -> Robo3T

- Permite visualizar las BDs y Colecciones en una ventana
- Valida sintaxis. Completa comandos (también la línea de comandos)
- Aunque todavía debes usar comandos de línea

b) Modo Línea comando : el CMD de Cliente con el intérprete

Todos los programas -> BBDD -> Entorno MongoDB

Se abre otra ventana cmd de Windows con el prompt: c:\hlocal>

3.- Conectar Cliente al servidor ejecutando en el Laboratorio. Dos modos:

a) **(usar este)** En Robo3T:

1.- *Crear una conexión* en ventana "mongo Connections"

- o "Create" : abre ventana con parámetros, dejar los que tiene por defecto
- o Se añade a la lista de conexiones
- o Escoge una de esas conexiones si quiere conectarte: haz click en *connect*

2.- *Abrir una Shell* (botón dcho sobre la conexión): permite las operaciones vistas en teoría

→ es un intérprete de JavaScript para la BD

→ Completa comandos pulsando TAB

3.- *Ejecutar un comando* después de escribirlo:

→ Para obtener el resultado en una sola ventana:

- selecciona el comando y ejecútalo con CTRL + RET
- Si NO lo seleccionas: ejecuta todos los comandos escritos en el buffer y abre una ventana para cada resultado:
 - Si haces: click en el marco de arriba deja solo esa ventana

b) Desde dentro del CMD del Cliente (paso 2.b): c:\hlocal> mongo

Responde:

MongoDB shell version: N.N.N → es un intérprete de JavaScript para la BD
(YA NO CONECTA POR DEFECTO)

connecting to: test // conecta a la BD "test"
// en la ventana del servidor dice : accept 1 conexión
// Completa comandos pulsando TAB

4.- Consultar y Crear una BD.

En la variable **db** se almacena la BD actual, es un objeto database
MongoDB siempre se inicia con la BD “test” por defecto

```
> show dbs // da los nombres de todas las BDs que hay
show collections // da los nombres de todas las colecciones en la BD actual (en db) SOLO si tienen docs.
➔ LINK interesante: documentación http://docs.mongodb.org/manual/

> db.getName() // devuelve el nombre de la BD actual sobre la que ejecuta los comandos
> db.help(); << // Ayuda de todos los métodos aplicables al objeto en var. db
> db.test.help(); << // Ayuda de todos los métodos aplicables al objeto colección “test”

> db.getMongo(); // devuelve el objeto connection: tiene métodos útiles asociados a él

> use midb // crear una BD con ese nombre y la asigna como “actual”
> miconn= db.getMongo(); // obtener la conexión en una variable para poder llamar a su métodos
// también se puede usar como objeto “db”
> nuevaDB=miconn.getDB("midb"); // asignamos la Base de Datos “midb” a la variable: es un objeto database
// También dispone de métodos útiles

> nuevaDB.getName(); // nos da el nombre de la BD que tiene en la var nuevaDB
```

SIGUE página siguiente ➔

5.- Crear un objeto collection (dentro de una BD)

Para almacenar datos dentro de la BD que apunta la variable *db*

```
> use testData // si ya existe la coloca como actual. Si no existe, la crea y asigna como actual
> db.createCollection("palabras"); // crea una colección (las comillas deben ser verticales ", no de abrir o cerrar: " ")
> coll = db.getCollection("palabras");// almacena la referencia a la colección en la var. coll
test.palabras

> miscoles = db.getCollectionNames(); // almacena un objeto con los nombres de colecciones
> printjson(miscoles); //imprime los nombre
[ "palabras" ]

> db.createCollection("nuevaColeA"); // crea otra colección
{ "ok" : 1 }
> db.createCollection("nuevaColeB");
{ "ok" : 1 }
> print("Después de crear colección:"); // imprime texto
Después de crear colección

> collections = db.getCollectionNames(); // responde con una lista de objetos
[ "nuevaColeA", "nuevaColeB", "palabras", "system.indexes" ]

> printjson(collections); // imprime json de los objetos
[ "nuevaColeA", "nuevaColeB", "palabras", "system.indexes" ]

> db.createCollection("nuevaColeA"); // intenta crear una col. que ya existe: da un obj con mensaje
{ "ok" : 0, "errmsg" : "collection already exists" }

> printjson(db.createCollection("nuevaColeB")); // imprime json del contenido del objeto de error
{ "ok" : 0, "errmsg" : "collection already exists" }

> print("Después de crear últimas colecciones");
Después de crear últimas colecciones

> miscoles = db.getCollectionNames(); // devuelve un objeto con los nombres
[ "nuevaColeA", "nuevaColeB", "palabras", "system.indexes" ]

> printjson(miscoles); // imprime el contenido del objeto
[ "nuevaColeA", "nuevaColeB", "palabras", "system.indexes" ]
```

6.- Crear documentos en la colección “palabras” de la db

Ayuda para escribir bien la sintaxis es validar json en Robomongo:

botón dcho sobre una colección + “Insert Document”

Se abre una ventana editor de texto. Escribes el documento { . . . } y botón “validate” (antes de salvar “save”)

```
> db.palabras.insert( // creamos un documento con la palabra “tomate”
{ "_id":{"str":"52d87454483398c8f24222"}, // si no das tú el valor, genera un valor único (índice y clave únicos)
  "palabra":"tomate",
  "primera":"t",
  "ultima":"e",
  "tamaño":6,
  "letras":["t", "o", "m", "a", "t", "e"],
  "estadis": { "vocales":3, "consonantes":3 },
  "caractsets":[
    { "tipo":"consonantes", "caracts":["t", "m"] },
    { "tipo":"vocales", "caracts":["e", "a"] }
  ]
} )
```

```
// responde : insertado un registro

> // SI REPETIMOS el insert
E11000 duplicate key error collection: midb.palabras index: _id_ dup key: { : { str: "52d87454483398c8f24222" } }

> db.palabras.count(); // consulta cuántos docs hay en la colección "palabras"
1

db.palabras.insert(
{ "_id":{"str":"52d87454483398c8f2429277"},
  "palabra":"the",
  "primera":"t",
  "ultima":"e",
  "tamaño":3,
  "letras":["t","h","e"],
  "estadis": {"vocales":1, "consonantes":2},
  "caractsets":[
    {"tipo":"consonantes","caracts":["t","h"]},
    {"tipo":"vocales","caracts":["e"]}
  ]
} )
```

➔ Cómo puedo validar la sintaxis JSON de un documento que he escrito a mano?

En el Robo3T, encima de la colección que quieras:

Hacer “insert document”: se abre una ventana par escribir el documento

Abajo tiene el botón de “validate”

7. Buscar documentos

ANTES hemos ejecutado esta instrucción para almacenar la referencia a la colección en la var. coll

```
coll = db.getCollection("palabras");
```

```
> coll.find() // da todos los objetos en la colección apuntada por coll, en este caso "palabras"

> sale= coll.find ( { "tamaño" : 3, "ultima" : "e" }, {}); // "the" tiene de tamaño 3

> sale= coll.find ( { "tamaño" : 6, "ultima" : "e" }, {}); // "tomate" tiene de tamaño 6

> sale= coll.find ( { "tamaño" : { $gt: 0, $lt: 4 }, "ultima" : 'e' }, {});

> sale= coll.find ( { "tamaño" : { $gt: 0, $lt: 4 }, "ultima" : 'e' }, {palabra: 1}); // da solo el campo "palabra"

> if (sale) {print("fue bien")} ; // compruebo si ha ido bien la instrucción anterior
```

➔ Mas ejercicios: Puedes probar búsquedas con la BD aficion-restaurantes.json de la práctica
➔ antes, ver como importarla a continuación

8.- Para llevar a tu casa la BD entera: mongodump y mongorestore

8.0- Salvar Toda la BD "test" al directorio A:\mongoWork

```
mongodump --db test --out A:\mongoWork
```

----- crea un directorio con dos ficheros, uno es el de metadatos

```
C:\Users\misuario> mongodump --db test --out A:\mongoWork
```

```
2016-05-06T17:52:14.384+0200 writing test.esp-restaurantes to
2016-05-06T17:52:14.387+0200 done dumping test.esp-restaurantes (15 documents)
```

→ *Algunos parámetros útiles:*
si queremos que cree un archivo solo --archive <nombreFichero>
Para comprimir --gzip

8.1- mongorestore : para incluir en Mongoddb un BD o colección generada con mongodump

- mongorestore crea una BD o añade a la que ponemos de destino
- NO actualiza los documentos si coinciden sus _Id únicos
- SI reconstruye los índices

8.2.- Recupera toda la BD de dump del path indicado en la bd indicada

```
mongorestore --db midb A:\mongoWork\test
```

```
C:\Users\misuario> mongorestore --db midb A:\mongoWork\test
```

```
2016-05-06T18:10:20.105+0200 building a list of collections to restore from A:\mongoWork\test dir
2016-05-06T18:10:20.108+0200 reading metadata for midb.esp-restaurantes from A:\mongoWork\test\esp-
restaurantes.metadata.json
2016-05-06T18:10:20.191+0200 restoring midb.esp-restaurantes from A:\mongoWork\test\esp-restaurantes.bson
2016-05-06T18:10:20.195+0200 restoring indexes for collection midb.esp-restaurantes from metadata
2016-05-06T18:10:20.196+0200 finished restoring midb.esp-restaurantes (15 documents)
2016-05-06T18:10:20.198+0200 done
```

8.3.- Recupera una sola colección en la bd indicada

```
C:\Users\misuario> mongorestore --collection dump-restaurantes --db nada A:\mongoWork\test\esp-restaurantes.bson
```

```
2016-05-06T18:06:24.548+0200 checking for collection data in A:\mongoWork\test\esp-restaurantes.bson
2016-05-06T18:06:24.551+0200 reading metadata for nada.dump-restaurantes from A:\mongoWork\test\esp-
restaurantes.metadata.json
2016-05-06T18:06:24.637+0200 restoring nada.dump-restaurantes from A:\mongoWork\test\esp-restaurantes.bson
2016-05-06T18:06:24.699+0200 restoring indexes for collection nada.dump-restaurantes from metadata
2016-05-06T18:06:24.727+0200 finished restoring nada.dump-restaurantes (15 documents)
2016-05-06T18:06:24.728+0200 done
```

8.4.- Importar y exportar en la BD una colección restaurantes (mejor usar el dump y restore)

- **Exportar desde el CMD, sin cliente:** (NO desde dentro del intérprete MongoDB)

```
mongoexport --db test --collection esp-restaurantes --out misrestaurantesx.json
```

- **Importar, En el dir donde están tus datos de la BD**

```
mongoimport --db test --collection restaurantes --drop --file E:\ABD-campusCV\misrestaurantes.json
```

```
mongoimport --db otraBD --collection restaurantes --drop --file D:\robomongo\rombodatos\misrestaurantesx.json
```

9.- VARIOS

----- Constructores: Comandos y funciones útiles (DENTRO DE LA SHELL)

-- Nativos del MS-DOS

- cd(): cambia el working directory. EJ: Cambio al dir local donde tengo mis datos
cd("H:/robomongo/rombodatos")
- pwd() para confirmar
- load() carga y ejecuta un fich JavaScript
- mkdir()
- pwd() muestr el work. dir.
- removeFile borra fichero

Fuciones Propias de MongoDB

- Date() crea el obj, date con fecha de hoy
 - db.traffic.insert({ _id: 1, volume: NumberLong('2980000'), date: new Date() })
- ObjectId.toString()
- \$dateToString

Uso de un aggregate para devolver el campo fecha en forma de strings formateadas usando \$dateToString

Si tengo este documento en db.micole

```
{
  "_id" : 1,
  "nombre" : "abc",
  "precio" : 10,
  "cantidad" : 2,
  "miFecha" : ISODate("2014-01-01T08:15:39.736Z")
}
```

Puedo formatear partes de la fecha:

```
db.micole.aggregate(
[
  {
    $project: {
      AñoMesDía: { $dateToString: { format: "%Y-%m-%d", miFecha: "$date" } },
      hora: { $dateToString: { format: "%H:%M:%S:%L", miFecha: "$date" } }
    }
  }
]
)
```

===== PASOS para INSTALAR MONGODB y ROBOMONGO (Actualmente ROBO 3T) en WIN =====

- 1-) Descargamos MongoDB de la página oficial, asegurarnos de bajar la versión para nuestra versión de Windows. Es el servidor de BD.
- 2-) Descargamos Robomongo de la página oficial de Robomongo (Actualmente ROBO 3T), es el entorno desarrollo que usamos. Es un cliente del servidor de MongoDB
- 3-) Instalamos MongoDB y Robomongo. Acordarnos de la ruta donde instalamos MongoDB.
- 4-) A partir de aquí seguir los PASOS para TRABAJAR en TU PORTATIL

===== PASOS para TRABAJAR en TU PORTATIL =====

- Para tu Bases de datos puedes tener un directorio de Trabajo que escoges, ej.: A:\mongoWork
- Asignas el camino de la instalación (en CMD): te sitúas en el dir que has escogido A:\mongoWork
set path=C:\Program Files\MongoDB\Server\N.N\bin\;%path%
- Arrancar el servidor. Ejecutar (en CMD) >
mongod.exe --dbpath A:\mongoWork // el dir. que quieras
- Arrancar cliente, una conexión al servidor, dos formas:
 - a).- En Robo3T hacer una conexión
 - b).- Ejecutar desde otra ventana del CMD : mongo
Antes: debes haber asignado el *path* correcto como arriba
- Si quieres importar alguna base de datos:
 - La creas dentro de MongoDB, ej. : use “aficiones”
 - En una ventana CMD: asignas el path y luego ejecutas lo indicado en secciones anteriores