Práctica 1 – Primer contacto con MongoDB

Nuestro primer objetivo es conseguir que MongoDB esté accesible y funcione.
Comenzamos arrancando en **Linux→usuario local**

1. Nos movemos al área de trabajo 2 (ctrl+Alt+ →)
2. Abrir un terminal (arriba a la derecha; al lado del símbolo de Chrome; si ya hay un terminal y queremos otro, dar botón derecho sobre el terminal y elegir "Ventana Nueva")
3. Crear un directorio donde se grabarán los datos: rm -rf datos; *mkdir datos*
4. Tecleamos mongod *--dbpath datos*
5. Volvemos al área de trabajo 1 ((ctrl+Alt+ ←)
6. Ahora vamos a importar los datos que usaremos en la práctica:
    1. Bajar del campus el fichero
    2. Desde la consola de linux teclear

        ```
        mongoimport -d lv -c comments --drop --file /home/hlocal/Descargas/lv.json
        ```

        debe decirnos que ha importado 121319 documentos

7. En el mismo terminal tecleamos *mongo lv* sin más...ya estamos en el shell de mongo. Podemos teclear show collections y debemos ver que la colección comments está. Y escribir db.comments.findOne() para mirar un poco la estructura de la colección. Se trata de un fichero de comentarios de un foro de noticias del periódico La Vanguardia.

A partir de ahora, cuando lleguemos al laboratorio solo habrá que repetir estos pasos para empezar las prácticas.

1) Consulta para mostrar el comentario que ocupa el puesto 101 en la colección comments.

Solución
db.comments.find().skip(100).limit(1)

2) Número de comentarios escritor por el autor con 'authorNick' igual a 'UROBOROS3'.
Nota1 : para saber cuántos elementos devuelve una consulta, añadir count() tras el find(…)
Nota2: observar que 'authorNick' es parte de clave 'content'
Copiar la consulta (solo la consulta) en el fichero de texto de la solución

Solución
db.comments.find({'content.authorNick':'UROBOROS3'}).count()

3) Autor (authorNick) del usuario que ha emitido el comentario

```
"_id" : { "collection" : "238449142","id" : "811961925"  }
```

Solución

```
db.comments.find({"_id" : { "collection" : "238449142","id" : "811961925"
}},{'content.authorNick':1,_id:0})
```

4) Para buscar documentos que incluyen un cierto valor en un array se utiliza la misma sintaxis que ya conocemos. Por ejemplo, copiar y ejecutar este código:

```
db.pru.drop()
db.pru.insert({_id:1, lista: [2,3,4]})
db.pru.insert({_id:2, lista: [1,3,4]})
db.pru.insert({_id:3, lista: [5,2]})
```

Entonces, la consulta

```
db.pru.find({lista:2})
```

devolverá

{ "_id" : 1, "lista" : [2, 3, 4] }
{ "_id" : 3, "lista" : [5, 2] }
Escribir una consulta que cuente el número de comentarios a los que ha dado like el usuario 'Don_Fulgencio'

Solución
db.comments.find({'content.likedBy':'Don_Fulgencio'}).count()

5) [2] La sintaxis db.comments.find(…).forEach(function(doc){ … }) permite ejecutar un código para cada documento de la consulta. Escribir una consulta de esta forma que permita copiar en una nueva colección 'user' todos los comentarios del usuario 'La.tralla'. Por tanto los primeros puntos suspensivos corresponderán a la consulta para seleccionar los comentarios de este usuario y los segundos será la operación de inserción en user. En la solución completar el código

db.user.drop(); db.comments.find(…).forEach(function(doc){ … }) ; db.user.count()

que escribirá el valor 1004.

Solución:
db.user.drop(); db.comments.find({'content.authorNick':
"La.tralla"}).forEach(function(doc){ db.user.insert(doc) })}) ; db.user.count()

6) El valor *comunidad* de la clave *content* indica si el usuario ha sido clasificado como no independentista '0' o independentista '1'. Escribir una expresión que calcule el tanto por ciento de independentistas en la colección *comments*.

Solución:
db.comments.distinct('content.authorNick', {'content.comunidad':'1'}).length * 100 /
db.comments.distinct('content.authorNick').length

7) Queremos obtener un listado de todos los usuarios (authorNick) pero sin que se repitan. Buscar información sobre el operador *distinct* en Mongo, y escribir la consulta correspondiente. La salida será de la forma:

```
[
        "cs62.8",
        "Kemedices",
        "jgarcia",
        "Towanda returns",
        "catalania",
        "indiketasam",
...
]
```

Solución:
db.comments.distinct("content.authorNick")

8) Vamos a crear una colección con las siguientes instrucciones:

```
db.pru.drop()
db.pru.insert([{a:1,b:1},{a:1,b:2},{a:2,b:1},{a:2,b:2},{a:3,b:1},{a:3,b:2}])
```

Escribir una consulta que devuelva:

{ "a" : 3, "b" : 1 }
{ "a" : 2, "b" : 1 }
{ "a" : 1, "b" : 1 }
{ "a" : 3, "b" : 2 }
{ "a" : 2, "b" : 2 }
{ "a" : 1, "b" : 2 }

Solución: db.pru.find({},{_id:0}).sort({b:1,a:-1})

9) Tenemos una colección con documentos que contienen un array a, como en el siguiente insert:
db.pru2.insert({a:[10,20,30,40,50]})
queremos una consulta que devuelva los dos últimos valores del a para todos los documentos. En el caso anterior mostraría

{ "a" : [40, 50] }

Debe ser un find normal, no se permite usar bucles en javascript ni nada similar. Buscar en internet soluciones para mostrar parte de un array

Solución
db.pru2.find({},{_id:0,a:{$slice:-2}})