MISION TIC2022

CICLO 1 DE FORMACIÓN: Fundamentos de Programación

Bases de Datos NoSQL - MONGO

Elizabeth León G, Universidad Nacional de Colombia

Notebook con información de instalación de mongo e interacción con mongo usando sentencias CRUD por consola.

MongoDB

- Base de datos NoSQL orientada a documentos
- Un registro en MongoDB es un documento, el cuál es una estructura de datos de pares campo/valor.
- Los documentos tienen una estructura similar a JSON
- Almacena una colección de documentos

```
{
  nombre : “Juan Perez",
  direccion : {
        calle : “Cra 26 # 44-20",
        ciudad : “Bogotá", bold text
        pais : “Colombia“
  }
}
```

Instalación

Linux

1. Descargue MongoDB: https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-3.2.9.tgz
2. Descomprima y cambie al directorio creado:

```
$ tar xfz mongodb-linux-x86_64-3.2.9.tgz
$ cd mongodb-linux-x86_64-3.2.9
```

3. Crear el directorio de datos:

```
$ mkdir ./data
```

4. Inicie el servidor de MongoDB:

```
$ ./bin/mongod —-dbpath ./data
```

5. Inicie el interprete de comandos (Mongo shell):

```
$ ./bin/mongo
```

Windows

https://www.mongodb.com/download-center#community

Mac

OPCION 1 Se recomienda instalar MongoDB a través de Homebrew.

1. Abrir una consola y actualizar Homebrew.

```
$ brew update
```

2. Una vez actualizado Homebrew, instalar MongoDB con el comando:

```
$ brew install mongodb
```

   Este comando instala los archivos binarios necesarios para ejecutar un servidor de MongoDB.

3. Crear el directorio donde el servidor de MongoDB guardará los datos. Desde la consola del sistema ejecuta:

```
$ mkdir -p /data/db
```

4. Asignar permisos necesarios para que el servidor pueda escribir en ese directorio.

```
$ sudo chown -R /data/db
```

% Puede solicitar la contraseña, eso es normal ya que se usa el comando sudo.

5. Abrir dos consolas, en una subir el servidor:

```
$ mongod
```

y en la otra, iniciar el interprete de comandos (Mongo shell):

```
$ mongo
```

El servidor de MongoDB (**mongod**) se "baja" con ctrl + c y el shell de Mongo (**mongo**) se cierra con quit().

OPCION 2 Bajar archivo .tgz e instalar

1. Bajar el tgz de https://www.mongodb.com/try/download/community
2. Después de descargar Mongo, mover el archivo comprimido .tar (archivo con la extensión .tgz que se ha descargado) a la carpeta donde se desee instalar Mongo. Se puede mover a la carpeta de inicio, con los siguientes comandos (o usar el manejador de archivos gráfico):

```
cd Downloads
mv mongodb-macos-x86_64-4.4.1.tgz ~/
```

3. Si se usa el manejador de archivos gráfico, dar doble click en el archivo para descomprimir. Si se esta trabajando por consola ejecutar:

```
cd ~/
tar -zxvf mongodb-macos-x86_64-4.4.1.tgz
```

4. Cambiar el nombre del directorio por mongodb

```
mv mongodb-macos-x86_64-4.4.1  mongodb
```

5. Crear el directorio donde mongo almacenará los datos en la carpeta de inicio. Por consola:

```
mkdir -p /data/db
```

6. Darle permisos al directorio

```
sudo chown -R `id -un` /data/db
```

7. Iniciar el servicio de mongo. Para esto ejecutar *mongod* (demonio de mongo) en una terminal. Esto debería iniciar el servidor de Mongo.

```
~/mongodb/bin/mongod
```

8. Para iniciar el intérprete de comandos de mongo (el shell de mongo) y empezar a usar mongo, abrir otra terminal (sin cerrar la anterior) y ejecutar el comando *mongo*.

```
~/mongodb/bin/mongod
```

   El servidor de MongoDB (**mongod**) se "baja" con ctrl + c y el shell de Mongo (**mongo**) se cierra con quit().

Interactuando con Mongo

Práctica con ejecución de sentencias CRUD.

En la consola de mongo seleccionar una base de datos. Como no hay creamos una con use (el comado use crea la bases de dtaos si esta no existe)

```
> use restaurante
```

Insertar documento. (**C**reation)

1. Comando: **db.producto.insert()**

   Al insertar un elemento y colocar el nombre de la colección, esta se crea. A continuación se inserta un producto en la colección producto:

```
> db.producto.insert({
_id : 1,
nombre : "BigBurger",
precio : 25000,
existencias : 30,
categoría : "Hamburguesa",
ingredientes : {
 pan : "arabe",
 salsa : ["tomate", "mayonesa"],
 vegetales : ["lechuga", "tomate"]
}
})
```

2. Insertar otros documentos tipo producto

```
> db.producto.insert({
_id : 2,
nombre : "Malteada Rosa",
precio : 15000,
existencias : 50,
categoría : "Bebida",
ingredientes : {
 helado : "fresa",
 fruta : "fresa"
}
})
```

```
> db.producto.insert({
_id : 3,
nombre : "CaseraH",
precio : 20000,
existencias : 45,
categoría : "Hamburguesa",
ingredientes : {
 pan : "con ajonjolí",
 salsa : ["tomate", "mayonesa", "mostaza"],
 vegetales : ["lechuga", "tomate", "cebolla"]
```

```
}
})
```

3. Insertar muchos documentos en una sola vez. Se utiliza una lista.

```
db.product.insertMany( [
{ "_id": 4,  "nombre": "Malteada de Maracuya", "precio" : 15000, "existencia
{ "_id": 5,  "nombre": "Limonada Natural", "precio" : 5000,"existencias" : 2
{ "_id": 6,  "nombre": "Criolla", "precio" : 35000,"existencias" : 10, "cate
] );
```

Ejercicio:

Insertar dos documentos más. Tener cuidado en asignar valores diferentes para los identificadores (_id). Deben ser únicos para cada producto.

Consultas (**R**ead)

1. Listar todos los documentos en la colección

```
> db.producto.find()
```

Si se desea ver de manera bonita el resultado, adicionar al final de la instrucción *.pretty()*:

```
> db.producto.find().pretty()
```

2. Buscar documentos que contengan la pareja campo:valor

```
> db.producto.find({nombre : "Malteada Rosa"})
```

3. Buscar los productos de categoría Hamburguesa, pero que solo devuelva el nombre y precio de las Hamburguesas

```
> db.producto.find(
{categoría : "Hamburguesa"},
{nombre: 1, precio: 1}).pretty()
```

La respuesta de la consulta debe ser:

```
{ "_id" : 1, "nombre" : "BigBurger", "precio" : 25000 }
{ "_id" : 3, "nombre" : "CaseraH", "precio" : 20000 }
```

4. Buscar documentos con condiciones AND (precio = 20000 Y (AND) categoría = "Hamburguesa")

```
> db.producto.find({precio : 20000, categoría : "Hamburguesa"})
```

5. Buscar documentos con documentos embebidos. Buscar los productos que tienen lechuga en sus ingredientes vegetales.

```
> db.producto.find({ "ingredientes.vegetales" : "lechuga" }).pretty()
```

6. Buscar documentos que contengan alguno de los valores en una lista

```
> db.producto.find({"ingredientes.salsa" : {$in : ["tomate", "mostaza"]}},
{nombre: 1, precio: 1, "ingredientes.salsa": 1}).pretty()
```

Ejercicio

Realizar las siguientes consultas:

1. Recuperar el precio del producto llamado "BigBurger". Solo el precio.
2. Recuperar las Hamburguesas de precio mayor a $25000
3. Recuperar las malteadas que tengan fresa
4. Recuperar las hamburguesas que no tengan cebolla
5. Definir dos consultas interesantes y realizarlas

Actualizaciones (**U**pdate)

Comando: **db.producto.insert()**

1. Actualiza un documento

```
> db.producto.update(
{ "nombre": "BigBurger"},
```

```
  { "$set": { precio: 35000 } },
{ multi: true }
)
```

2. A continuación, se actualiza la categoría de los productos que el nombre inicie con la letra "M".

```
db.producto.update(
{ "nombre": { "$regex": "^M" } },
{ "$set": { "categoría": "Malteada" } },
{ multi: true }
)
db.producto.find(}).pretty()
```

3. Incrementa el precio en $8000 a las hamburguesas:

```
db.producto.update(
{ "categoría":"Hamburguesa"},
{ "$inc": {"precio": 8000 } },
{ multi: true }
)
db.producto.find(}).pretty()
```

4. El comando *pull* remueve un elemento de una lista. A continuación se remueve la salsa de tomate del producto con _id = 6

```
db.producto.update(
{ "_id": 3},
{ $pull: {"ingredientes.salsa": "tomate"} },
{ multi: true }
)
db.producto.find(}).pretty()
```

Ejercicio

Realizar las siguientes actualizaciones:

1. Aumentar el precio en 1000 a los productos que tengan helado de maracuya.
2. Restar en 10 las cantidades de la Malteada con _id 2.

3. Cambiar el ingrediente cebolla por pepinillos en todas las Hamburguesas.

Borrado (**D**elete)

1. Borra un documento

```
db.producto.deleteOne({"nombre": "BigBurger"})
```

2. Borra varios documentos

```
db.producto.deleteMany({"categoría": "Malteada"})
```

Ejercicio

Realizar los siguientes borrados:

1. Borrar el producto con _id 3.
2. Borrar los productos que tengan pepinillos en los vegetales
3. Borrar los productos con precio menor a 20000

Referencias

Manual de referencia: https://docs.mongodb.com/manual/