Laboratorio Nro 09 - Instalación y Gestión de una base de datos MongoDB

Universidad Privada de Tacna

Escuela Profesional de Ingeniería de Sistemas

Base de DAtos II

Robles Flores , Anthony Richard (2016056192)

Tacna, Perú

---

1. INFORMACIÓN GENERAL

Objetivos:

- Realizar la instalacion de Base de Datos NoSQL MongoDB
- Importar un archivo JSON con colecciones de datos a MongoDB
- Realizar consultas de registros con MongoDB

Requerimientos
Hardware

- Virtualization activada en el BIOS.
- CPU SLAT capable feature.
- Al menos 4GB de RAM.

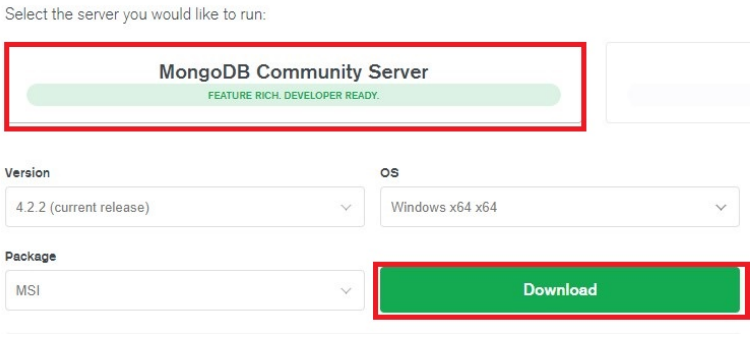
Software

- Windows 10 64 bit
- Docker Desktop

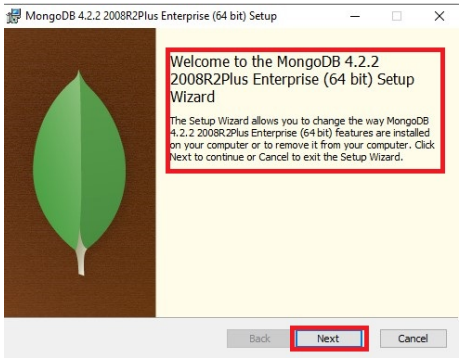
2. DESARROLLO DE LABORATORIO

Parte I : Instalación de MongoDB (seguir los pasos)

a) Ingresar a la URL (https://www.mongodb.com/download-center/community) para descargar el instalador que nos ofrece en la pagina.

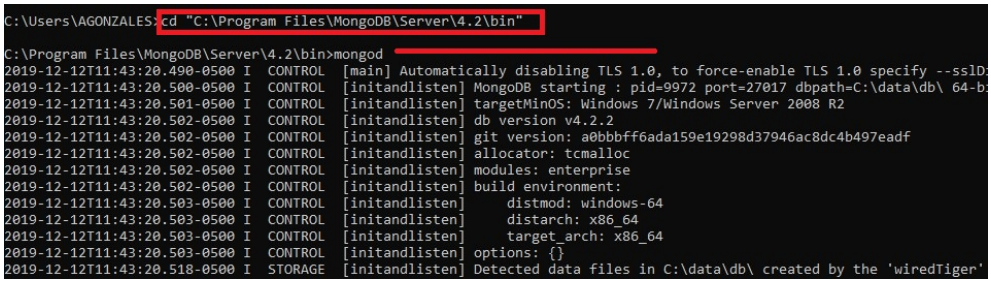


b) Iniciar el instalador como administrador de windows.



c) Crear las carpetas de almacenamiento y configuración de MongoDB. Dar permisos escritura y lectura a estas carpetas.Tal como se ve en la imagen C:/data - C:/data/db

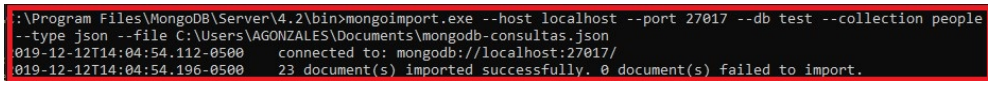
d) Iniciar el servidor del servicio de MongoDB: mongod. Primero dirigir a la siguiente ruta (C:/Program Files/MongoDB/Server/4.2/bin). Y ejecutar el comando mongod:



e) Iniciar el shell de mongo : mongo.exe

Parte II: Importar Datos

f) Para importar datos utilizaremos el comandoimport. Este comando es capaz de importar datos en distintos formatos a nuestra base de datos. (mongoimport.exe –host localhost –port 27017 –db test –collection people –type json –file C:/Users/AGONZALES/Documents/mongodb-consultas.json)

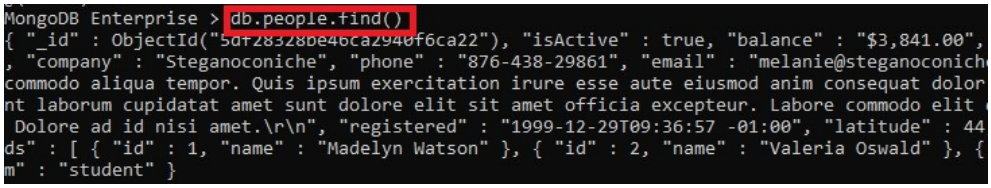


g) Con este comando, lo que haremos será incorporar los datos a nuestro servidor, que está en el puerto 27017, concretamente a una base de datos llamada test y a una colección que hemos llamado people.

Parte III : Consultas

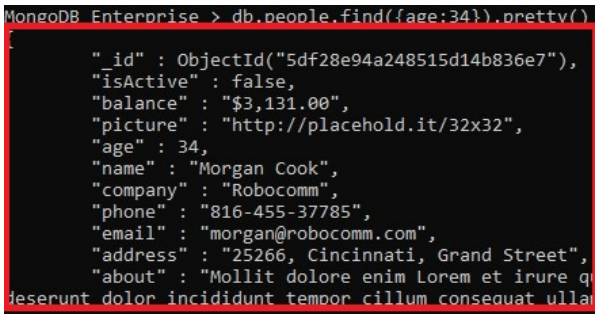
h) Iniciar el shell de mongo mongo.exe.

i) Para realizar consultas a la base de datos, deberemos usar el comando db.nombredecoleccion.find(). Este comando puede recibir dos parámetros: una consulta y una proyección. Ambos comandos son opcionales por lo que si ejecutamos el comando: 'db.people.find()'.



j) Obtendremos una larga lista con los 20 elementos de la colección. Al ejecutar el comando podemos ver que el resultado no está demasiado formateado, por lo que es muy difícil leerlo. Para solucionar este problema podemos usar el modificador pretty que nos devolverá un resultado mucho más legible 'db.people.find().pretty()'.

k) Ahora vamos a añadir la consulta al comando find, para que filtre los elementos según nuestras necesidades. Para ello especificaremos un objeto JSON como primer parámetro del comando, con los campos por los que queremos filtrar:
db.people.find(age:34).pretty()



l) Con ese comando obtendremos las personas cuya edad es de 34 años. Podemos añadir tantos filtros como queramos.

db.people.find(age:34,isActive:true).pretty()

```
MongoDB Enterprise > db.people.find({age:34,isActive:true}).pretty()
{
        "_id" : ObjectId("5df297add08dd97f3e32aa8c"),
        "isActive" : true,
        "balance" : "$1,066.00",
        "picture" : "http://placehold.it/32x32",
        "age" : 34,
        "name" : "Julia Young",
        "company" : "Genland",
        "phone" : "831-524-30334",
        "email" : "julia@genland.com",
        "address" : "18282, ChulaVista, Bleecker Street",
        "about" : "Nisi excepteur Lorem mollit amet amet aliqua non c
uptate aliquip sit voluptate nisi cillum pariatur. Incididunt amet co
        "registered" : "2005-10-27T05:30:21 -02:00",
        "latitude" : -18,
```

m) En este caso filtramos por age y por isActive. Como se ve los resultados nos muestran todos los campos de cada elemento. Es como si hubiésemos utilizado el asterisco en una consulta SELECT. Si queremos solo algunos de los campos, deberemos utilizar el segundo parámetro de la consulta find para definir una proyeccion.
db.people.find(age:34,isActive:true, name:1,age:1,isActive:1).pretty()

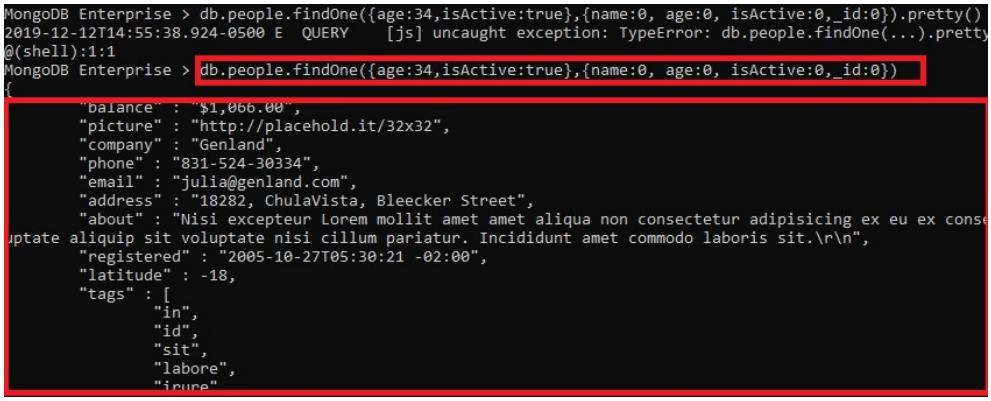
```
MongoDB Enterprise > db.people.find({age:34,isActive:true},{name:1, age:1, isActive:1}).pretty()
{
        "_id" : ObjectId("5df297add08dd97f3e32aa8c"),
        "isActive" : true,
        "age" : 34,
        "name" : "Julia Young"
}
```

n) Que no devuelve solo los campos que se requieren, además del id. El id por defecto se muestra siempre, así que si queremos ocultarlo hay que especificarlo en la proyección.
db.people.find(age:34,isActive:true, name:1,age:1,isActive:1,id:0

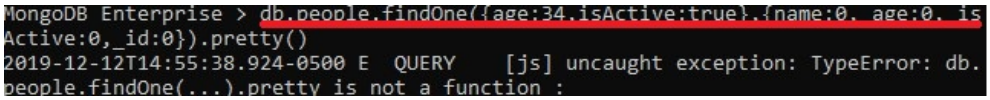
```
MongoDB Enterprise > db.people.find({age:34,isActive:true},{name:1, age:1, isActive:1, id:0}).pretty()
{ "isActive" : true, "age" : 34, "name" : "Julia Young" }
```

ñ) Si queremos mostrar todos los campos, pero quitando solo algunos, lo que haremos será desactivar los no deseados en la proyección:
db.people.find(
age:34,isActive:true,
name:0,age:0,isActive:0,id:0).pretty()

o) El comando findOne tiene el mismo funcionamiento que el comando find, con la diferencia de que, si el comando encuentra más de un resultado que cumpla las condiciones de la consulta, tan solo nos devolverá el primero.
db.people.findOne(age:34,isActive:true,name:0,age:0,isActive:0,id:0)



p) Además findOne no acepta pretty, pero ya devuelve el resultado formateado.



q) En las consultas, en cambio, no se ha especificado dichas comillas. Esto es porque el motor JavaScript de MongoDB se encarga de añadirlas. Esto nos facilita la escritura de consultas, ya que no son obligatorias. De hecho, la siguiente consulta, funcionará perfectamente:
db.people.findOne(
“age”:34,”isActive”:true,

"name":0,"age":0,"isActive":0,"id":0)