Trabajo Práctico – Introducción a MongoDB

Consigna

El objetivo de este trabajo es familiarizarse con MongoDB, utilizando Mongo Shell (mongosh) y Mongo Compass. Se deberán crear múltiples bases de datos, colecciones y documentos, además de aplicar operaciones básicas (insertar, consultar, actualizar y eliminar).
El alumno deberá entregar:
- Un documento con capturas de pantalla de los comandos ejecutados y sus resultados.

Parte 1 – Preguntas de repaso

1. Conceptos generales

a. ¿Qué es un documento en MongoDB y en qué formato se almacena?

Es la unidad básica de datos en MongoDB (pares clave–valor, anidables).
Se almacena en BSON (Binary JSON). En la consola lo ves como JSON, pero físicamente es BSON. Siempre tiene un campo _id único.

b. **¿Qué es una colección y en qué se diferencia de una tabla en una base de datos relacional?**
Una colección es un conjunto de documentos, posee un esquema flexible (los documentos pueden tener distintos campos) y una tabla son filas con esquema fijo (mismas columnas).

2. Comandos básicos

a. **¿Qué hace el comando use <nombreBD>?**
Cambia el contexto a esa base. La BD no se crea realmente hasta que haya una escritura.
b. **¿Qué comando se utiliza para listar todas las bases de datos disponibles?**
show dbs
c. ¿Por qué una base de datos recién creada no aparece en la lista de dbs hasta que se inserta un dato?
Porque use <bd> solo cambia el contexto: MongoDB crea las bases de forma diferida (lazy) y solo las materializa cuando hay alguna escritura (p. ej., insertar un documento o crear una colección/índice).
show dbs lista únicamente las bases con tamaño > 0 en disco; si no hubo escritura, la BD no tiene datos y no aparece.

3. Inserción de datos

a. **Explica con tus palabras la diferencia entre insertOne() e insertMany().**
   insertOne() inserta un documento.
   insertMany() inserta varios documentos
b. **¿Qué ventajas tiene insertMany() cuando queremos cargar grandes volúmenes de datos?**
   La ventaja de insertMany() es que permite insertar múltiples documentos en una sola operación, lo que lo hace más rápido y eficiente que ejecutar varios insertOne(), reduciendo el número de solicitudes al servidor y mejorando el rendimiento al cargar grandes volúmenes de datos.

4. Consultas

a. **¿Qué diferencia hay entre find() y find().pretty()?**
   Ambos devuelven lo mismo; pretty() solo formatea la salida en la consola para que sea legible (sangrías, saltos de línea).

5. Eliminaciones

a. **¿Qué diferencia hay entre deleteOne(), drop() y dropDatabase()?**
   deleteOne(filtro): elimina un documento que cumpla el filtro (existe deleteMany para varios).
   drop(): elimina la colección completa (y sus índices).
   dropDatabase(): elimina toda la base de datos (todas las colecciones e índices).

b. **Si eliminas una colección con drop(), ¿sigue existiendo la base de datos?**
   Sí. La BD permanece. Si queda sin colecciones, puede no listarse hasta que vuelvas a insertar algo.

6.**Colecciones y administración**

a. **¿Qué hace el comando db.createCollection() y en qué caso puede ser útil, si MongoDB crea colecciones automáticamente al insertar datos?**

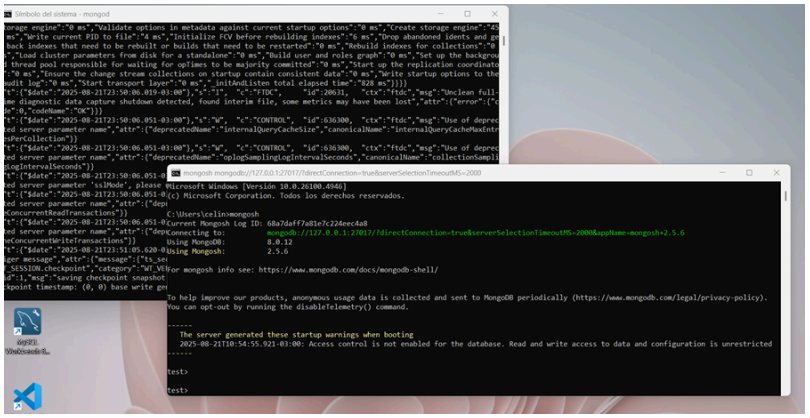
El comando db.createCollection("nombreColeccion") crea explícitamente una colección en la base de datos actual.
dEl comando b.createCollection() es útil en casos específicos, como cuando necesitas:

- Definir opciones avanzadas de la colección, por ejemplo:
  - Tamaño máximo de la colección (capped).
  - Límite de documentos.
  - Validación de esquemas (con reglas sobre los campos).
- Crear colecciones vacías pero con restricciones antes de empezar a insertar datos.

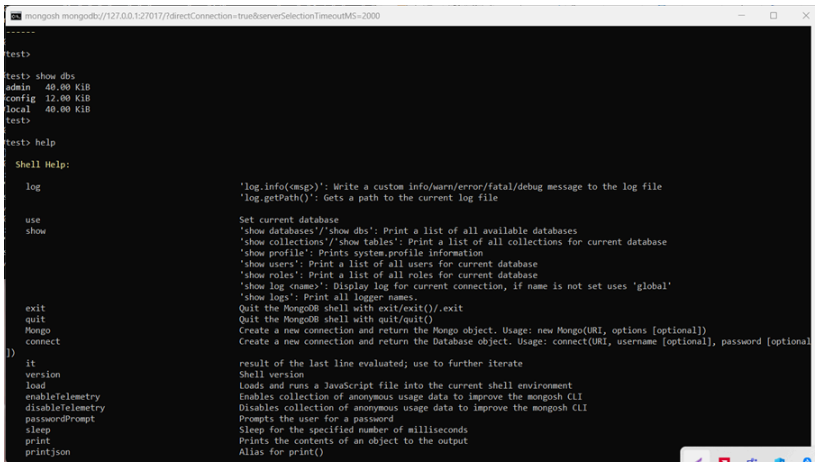
b. **¿Qué sucede si intentamos insertar un documento sin definir campos (ejemplo: {})?}**
Si insertamos algo como db.clientes.insertOne({}) MongoDB inserta un documento vacío, pero le asigna automáticamente un campo _id (que es obligatorio y único).
El resultado sería algo como: { "_id": ObjectId("64e2fbd3a1c3d2...") }
No es muy útil insertar {} porque no guarda información relevante.

Parte 2 – Conexión y exploración inicial

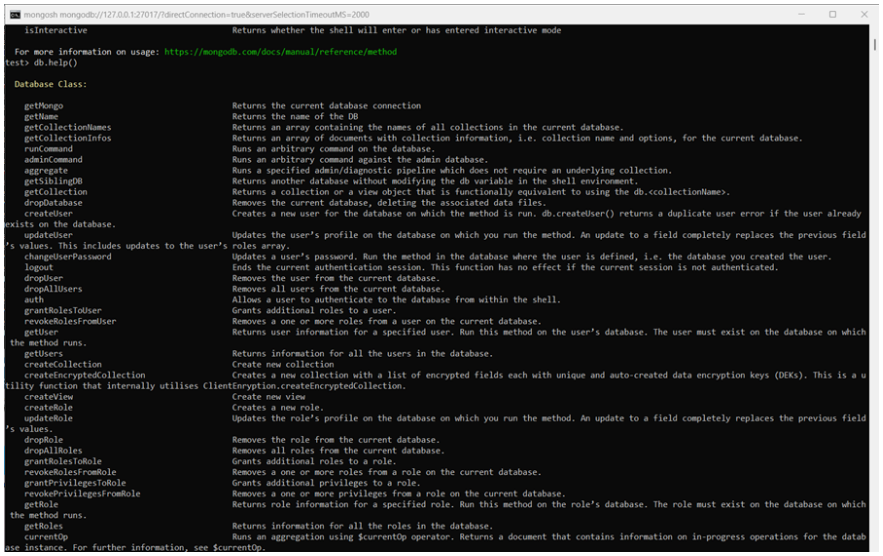
1. Abrir una terminal y ejecutar:
Mongosh



2. Ejecutar los siguientes comandos de exploración:
show dbs
help



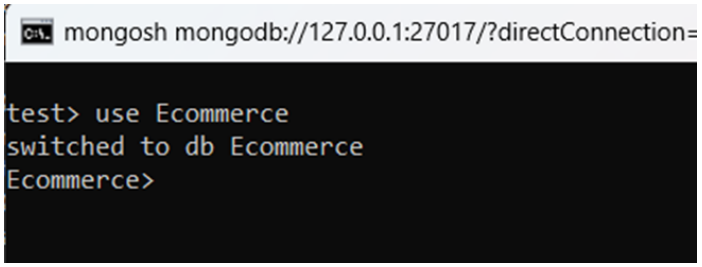
db.help()



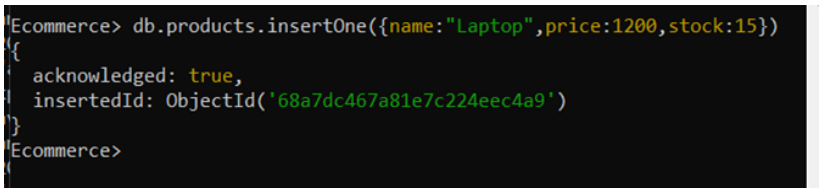
Db

Parte 3 – Creación de bases de datos y colecciones

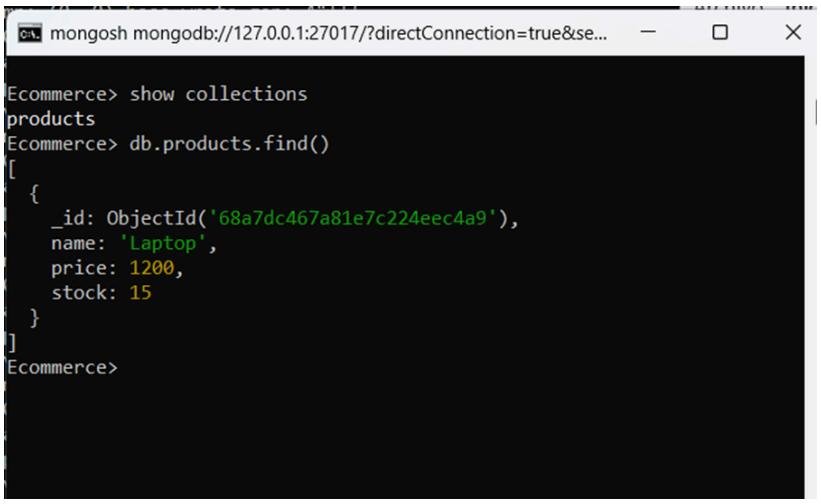
1. Crear una base de datos llamada Ecommerce:
use Ecommerce



2. Crear una colección products e insertar un documento:
db.products.insertOne({ name: "Laptop", price: 1200, stock: 15 })



3. Verificar con:
show collections
db.products.find()



Parte 3 – Inserción de múltiples datos

1. Insertar varios productos:

```
db.products.insertMany([
  { name: "Monitor", price: 400, stock: 25 },
  { name: "Teclado", price: 50, stock: 100 },
  { name: "Mouse", price: 30, stock: 200 }
])
```

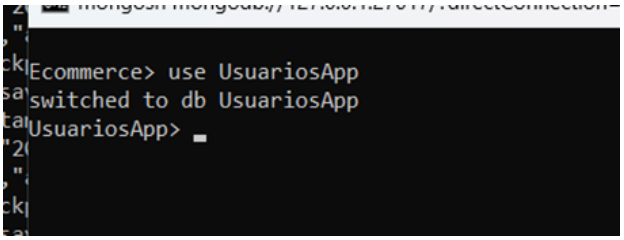
2. Consultar:

```
db.products.find()
```

db.products.find({ name: "Monitor" })

Parte 4 – Otra base de datos con usuarios

1. Crear una base de datos llamada UsuariosApp:
use UsuariosApp

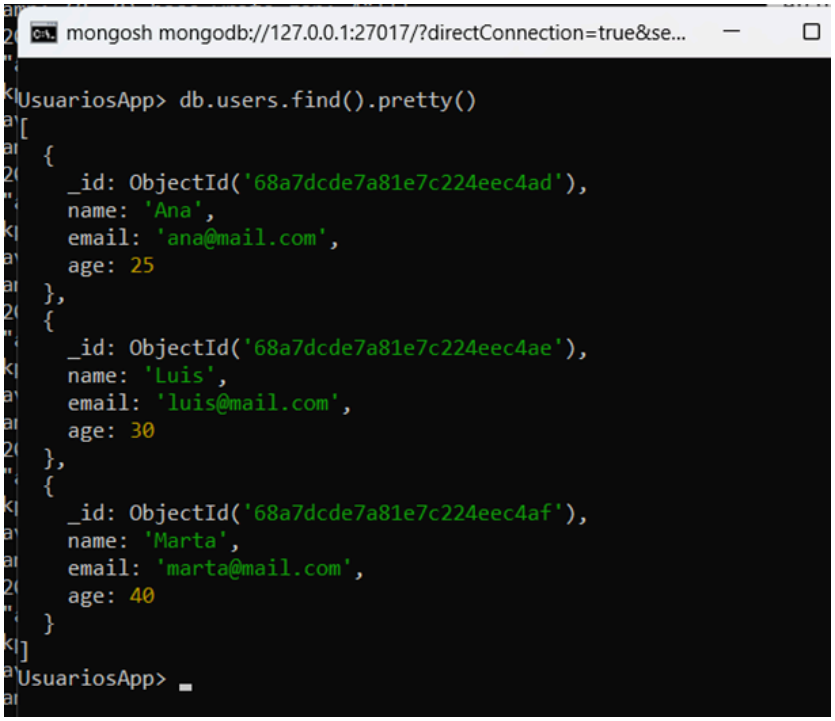


2. Insertar documentos en una colección users:

```
db.users.insertMany([
  { name: "Ana", email: "ana@mail.com", age: 25 },
  { name: "Luis", email: "luis@mail.com", age: 30 },
  { name: "Marta", email: "marta@mail.com", age: 40 }
])
```

3. Consultar todos los usuarios con:

```
db.users.find().pretty()
```

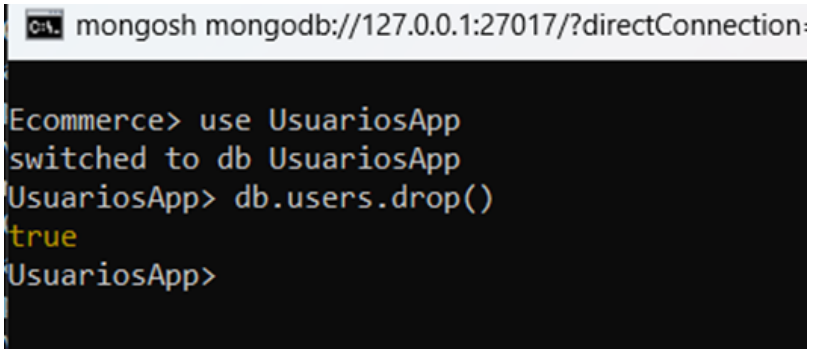


Parte 5 – Actualizaciones y eliminaciones

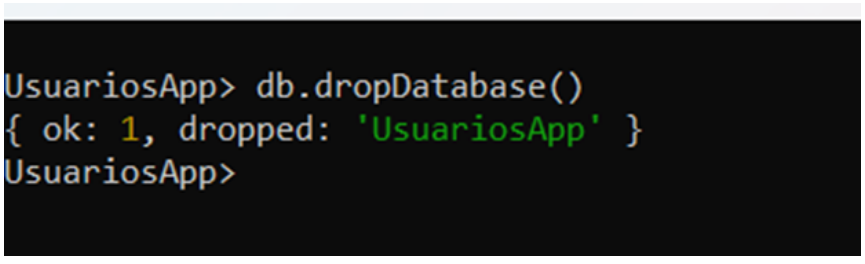
1. Actualizar el stock de un producto:
use Ecommerce
db.products.updateOne({ name: "Teclado" }, { $set: { stock: 80 } })

2. Eliminar un producto:
db.products.deleteOne({ name: "Mouse" })

3. Eliminar toda la colección users:
use UsuariosApp
db.users.drop()

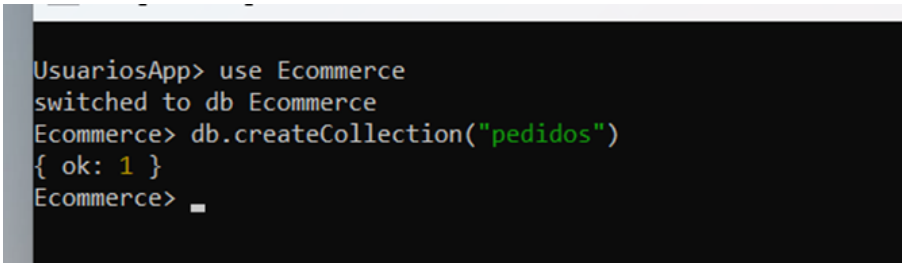


4. Eliminar la base de datos UsuariosApp:
db.dropDatabase()



Parte 6 – Extra: Crear una colección explícita

1. Crear la colección pedidos en la base Ecommerce:
use Ecommerce
db.createCollection("pedidos")



2. Insertar algunos pedidos:
db.pedidos.insertMany([
{ user: "Ana", product: "Laptop", quantity: 1 },
{ user: "Luis", product: "Monitor", quantity: 2 }
])

3. Consultar los pedidos con:
db.pedidos.find().pretty()