

Trabajo Tema 4

Otros modelos de datos para SI.
MongoDB



**UNIVERSIDAD
DE GRANADA**

Grupo: D1 Five Stars

Juan Andrés Mauricio Martín
Daniel Alconchel Vázquez
Luis Crespo Orti
Laura Lázaro Soraluze
Ximo Sanz Tornero

1. Descarga e instalación.

Para instalar MongoDB vamos a usar el tutorial:

<https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-20-04-es>

Antes que nada, debemos importar la clave pública GPG para la versión indicada en la página web. En este caso, la **4.4**. Lo haremos usando el siguiente comando:

```
curl -fsSL https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
```

Vemos que tras ejecutar esta orden, se nos devuelve un OK, lo que confirma que se habrá hecho correctamente.

En este momento, la instalación APT no sabe dónde encontrar el paquete mongodb-org.

Para ello ejecutamos:

```
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list
```

Tras ello, ejecutamos:

```
sudo apt update.
```

Y ahora finalmente, instalamos **MongoDB**

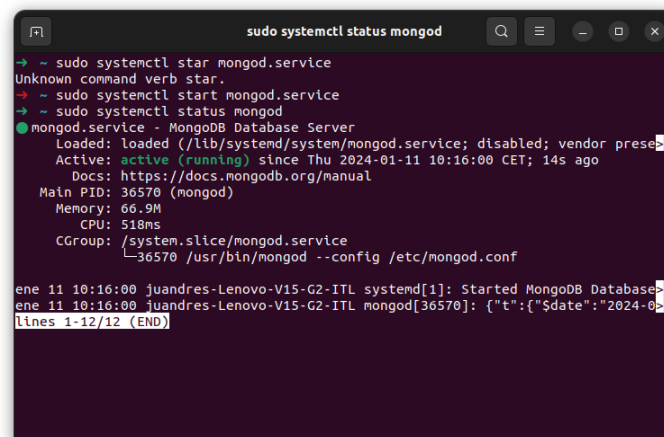
```
sudo apt install mongodb-org
```

Cabe destacar que nos dio un problema de dependencias entre paquetes que resolvimos con este [tutorial](#).

El proceso de instalación descrito en el paso anterior configura automáticamente MongoDB para que se ejecute como un daemon controlado por systemd, lo que significa que puede administrar MongoDB usando los diferentes comandos de systemctl. Sin embargo, este procedimiento de instalación no inicia automáticamente el servicio. Iniciemos el servicio con el siguiente comando,

```
sudo systemctl start mongod.service
```

Veremos que está **running**



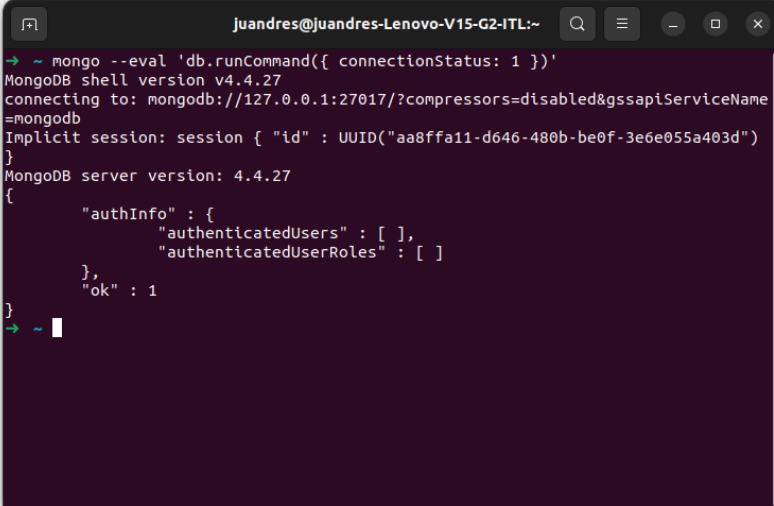
```
sudo systemctl status mongod
→ ~ sudo systemctl star mongod.service
Unknown command verb star.
→ ~ sudo systemctl start mongod.service
→ ~ sudo systemctl status mongod
● mongod.service - MongoDB Database Server
   Loaded: loaded (/lib/systemd/system/mongod.service; disabled; vendor prese
   Active: active (running) since Thu 2024-01-11 10:16:00 CET; 14s ago
     Docs: https://docs.mongodb.org/manual
   Main PID: 36570 (mongod)
    Memory: 66.9M
       CPU: 518ms
    CGroup: /system.slice/mongod.service
           └─36570 /usr/bin/mongod --config /etc/mongod.conf

ene 11 10:16:00 juandres-Lenovo-V15-G2-ITL systemd[1]: Started MongoDB Database
ene 11 10:16:00 juandres-Lenovo-V15-G2-ITL mongod[36570]: {"t":{"$date":"2024-0
lines 1-12/12 (END)
```

Ahora, habilitaremos el servicio para que se inicie en el arranque:

```
sudo systemctl enable mongod
```

Ahora, podemos verificar que la base de datos está operativa conectando con el servidor de la base de datos y ejecutando un comando de diagnóstico.

A terminal window with a dark purple background. The title bar shows 'Juandres@Juandres-Lenovo-V15-G2-ITL:~'. The prompt is '~ mongo --eval 'db.runCommand({ connectionStatus: 1 })''. The output shows 'MongoDB shell version v4.4.27', 'connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb', 'Implicit session: session { "id" : UUID("aa8ffa11-d646-480b-be0f-3e6e055a403d") }', 'MongoDB server version: 4.4.27', and a JSON object: {'authInfo': {'authenticatedUsers': [], 'authenticatedUserRoles': []}, 'ok': 1}. The prompt returns to '~ '.

Por último, instalaremos y configuraremos la **shell** de MongoDB para poder trabajar con ella. Nos la descargamos de la [página web](#). Podremos acceder a la shell, con el comando *mongosh*.

2. Descripción DDL y DML

Debemos tener en cuenta que MongoDB está orientado a documentos, lo que quiere decir, que almacena colecciones de documentos de formato libre, llamados **colecciones**. MongoDB, siendo una base de datos NoSQL, tiene un enfoque diferente al de las bases de datos relacionales tradicionales en cuanto a la manipulación y definición de datos.

DML (Data Manipulation Language) en MongoDB:

Insertar Documentos:

- *insertOne()*: Inserta un único documento en una colección.
- *insertMany()*: Permite insertar múltiples documentos en una sola operación.

Actualizar Documentos:

- *updateOne()*, *updateMany()*: Actualizan uno o varios documentos que cumplen con un criterio específico.
- *replaceOne()*: Reemplaza un documento completo.

Eliminar Documentos:

- *deleteOne()*, *deleteMany()*: Eliminan documentos que coinciden con un criterio dado.

Consultar Documentos:

- *find()*: Para recuperar documentos. Puede incluir filtros, ordenamiento, limitación de resultados, etc.

Indexación:

- Aunque no es manipulación directa de los datos, la creación de índices es crucial para la optimización de consultas.

DDL (Data Definition Language) en MongoDB:

El DDL en las bases de datos relacionales se refiere a las instrucciones que definen la estructura de la base de datos, como crear, alterar y eliminar tablas. En MongoDB, el enfoque es diferente:

Creación de Colecciones:

- *createCollection()*: Crea una nueva colección. En MongoDB, las colecciones son análogas a las tablas en las bases de datos relacionales.

Modificación de Colecciones:

- Aunque no es común en MongoDB modificar la estructura de las colecciones, se pueden realizar ciertas operaciones, como cambiar la configuración de validación de documentos.

Eliminar Colecciones o Bases de Datos:

- *drop()*: Elimina una colección.
- Para eliminar una base de datos, se usa el comando *db.dropDatabase()*.

Manejo de Índices:

- Creación, eliminación y modificación de índices.

Veamos un ejemplo general:

Un documento para un trabajador tendría el siguiente formato:

```
{
  "_id": "123456789012",
  "nombre": "Juan Andres",
  "apellidos": "Mauricio Martin",
  "fechaNacimiento": ISODate("2001-07-03T00:00:00Z"),
  "area": "Administración",
  "fechaFinContrato": ISODate("2025-01-11T00:00:00Z")
}
```

Podríamos crear la colección **Trabajadores**, de la siguiente forma:

```
db.createCollection("Trabajadores")
```

Para insertar un nuevo trabajador:

```
db.Trabajadores.insertOne({
  "_id": "123456789012",
  "nombre": "Juan Andres",
  "apellidos": "Mauricio Martin",
  "fechaNacimiento": ISODate("2001-07-03T00:00:00Z"),
  "area": "Administración",
  "fechaFinContrato": ISODate("2025-01-11T00:00:00Z")
})
```

Para actualizarlo:

```
db.Trabajadores.updateOne(
  { "_id": "123456789012" },
  { $set: { "area": "Recursos Humanos" } }
)
```

Para eliminar un trabajador

```
db.Trabajadores.deleteOne({ "_id": "123456789012" })
```

Para encontrar a un trabajador o un grupo de trabajadores:

```
db.Trabajadores.find({ "area": "Administración" })
```

Además, para consultas, también podríamos usar los siguientes operadores:

- *\$eq*: Los valores son iguales.
- *\$ne*: Los valores no son iguales.
- *\$gt*: El valor es mayor que otro valor.
- *\$gte*: El valor es mayor o igual a otro valor.
- *\$lt*: El valor es menor que otro valor.
- *\$lte*: El valor es menor o igual a otro valor.
- *\$in*: El valor coincide con cualquier valor dentro de un array.

3. Sentencias

Lo primero que haremos en MongoDB será crear nuestra nueva base de datos
use *FiveStars*

Añadiremos las respectivas tablas de Actividades, Clientes y Trabajadores

```
mongosh mongodb://127.0.0.1:27017/?directConnection=tru...
FiveStars> db.createCollection("Actividades")
{ ok: 1 }
FiveStars> db.createCollection("Clientes")
{ ok: 1 }
FiveStars> db.createCollection("Trabajadores")
{ ok: 1 }
FiveStars> show collections
Actividades
Clientes
Trabajadores
FiveStars>
```

Vamos a insertar algunos elementos:

```
FiveStars> db.actividades.insertOne({
...   nombre: "Yoga Matutino",
...   lugar: "Salón Comunal",
...   tipo: 2,
...   fecha: new Date("2024-05-15"),
...   hora_inicio: new Date("2024-05-15T07:00:00Z"),
...   hora_fin: new Date("2024-05-15T08:00:00Z"),
...   num_max: 15,
...   id_trabajador: "123456789012"
... });
{
  acknowledged: true,
  insertedId: ObjectId('659fc199c58a14680c1d3e6e')
}
FiveStars>
```

```
FiveStars> db.clientes.insertOne({
...   _id: "123456789012",
...   nombre: "Ana",
...   apellidos: "Pérez",
...   numeroTelefono: "123456789",
...   correoElectronico: "ana.perez@email.com",
...   tipoCliente: 1,
...   importe: 120.50
... });
{ acknowledged: true, insertedId: '123456789012' }
```

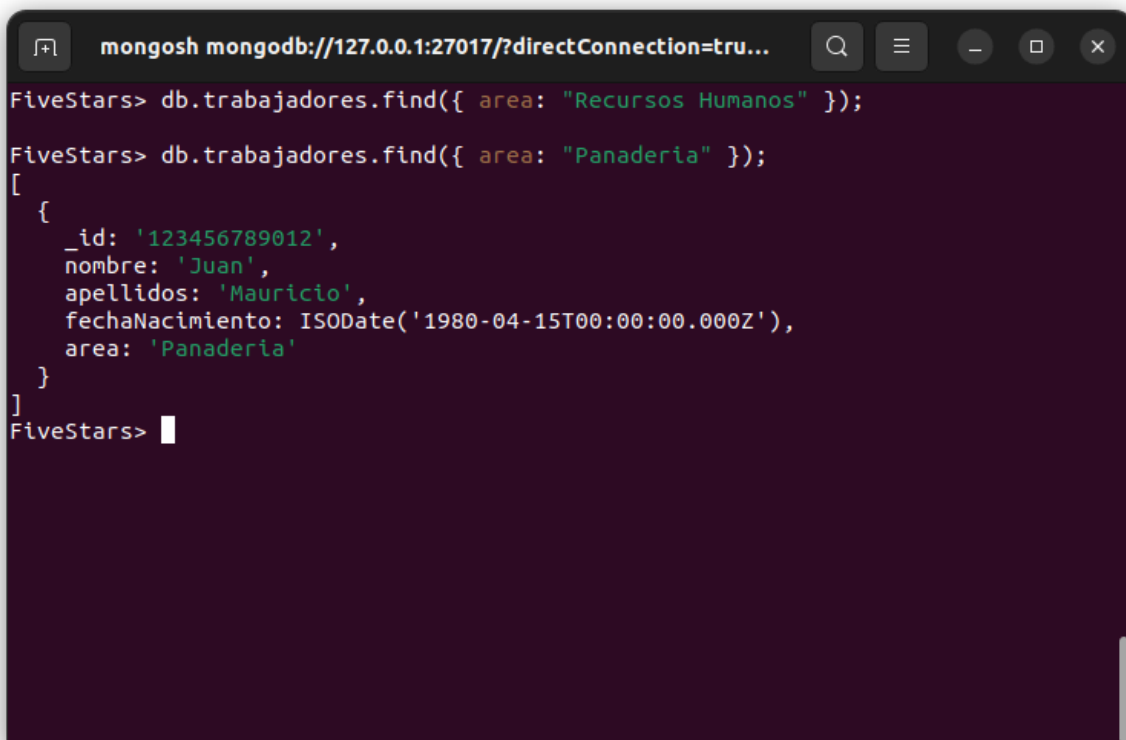
```
FiveStars> db.trabajadores.insertOne({
...   _id: "123456789012",
...   nombre: "Juan",
...   apellidos: "Mauricio",
...   fechaNacimiento: new Date("1980-04-15"),
...   area: "Administración"
... });
```

Veamos cómo podemos cambiar el área de un trabajador

```
FiveStars> db.trabajadores.updateOne(
... { _id: "123456789012"},
... {$set: {area: "Panaderia"}}
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

FiveStars> db.trabajadores.find()
[
  {
    _id: '123456789012',
    nombre: 'Juan',
    apellidos: 'Mauricio',
    fechaNacimiento: ISODate('1980-04-15T00:00:00.000Z'),
    area: 'Panaderia'
  }
]
FiveStars>
```

Ahora haremos una consulta



```
mongosh mongodb://127.0.0.1:27017/?directConnection=tru...
FiveStars> db.trabajadores.find({ area: "Recursos Humanos" });
FiveStars> db.trabajadores.find({ area: "Panaderia" });
[
  {
    _id: '123456789012',
    nombre: 'Juan',
    apellidos: 'Mauricio',
    fechaNacimiento: ISODate('1980-04-15T00:00:00.000Z'),
    area: 'Panaderia'
  }
]
FiveStars>
```

4. Mecanismo de conexión

En esta sección, vamos a ver como nos conectaremos a MongoDB a través de Python. Lo primero que debemos hacer es instalar la librería *pymongo*.

pip install pymongo

Una vez instalada la librería, ya nos podemos conectar a la base de datos. Para ello, debemos crear un script que contenga lo siguiente.

```
import pymongo

MONGODB_HOST = '127.0.0.1'
MONGODB_PORT = '27017'
MONGODB_TIMEOUT = 1000

URI_CONNECTION = "mongodb://" + MONGODB_HOST + ":" + MONGODB_PORT + "/"

try:
    client = pymongo.MongoClient(URI_CONNECTION, serverSelectionTimeoutMS=MONGODB_TIMEOUT)
    client.server_info()
    print('OK -- Conectado al servidor MongoDB en ' + MONGODB_HOST)
    client.close()
except pymongo.errors.ServerSelectionTimeoutError as error:
    print('Error con la conexión por TimeOut: ' + str(error))
except pymongo.errors.ConnectionFailure as error:
    print('Error -- No nos hemos podido conectar a MongoDB: ' + str(error))
```

Con este código ya nos podríamos conectar a la base de datos.

5. ¿Es adecuado para nuestra práctica?

Es claro que pasar de un lenguaje SQL a uno NoSQL será complicado. Sin embargo, no es imposible. Usando MongoDB de una forma específica, podríamos llegar a implementar nuestra SGBD. Los puntos más importantes a considerar serían los siguientes:

- OracleSQL se basa en relaciones fijas y normalizadas para evitar la **redundancia de datos**. Esto no ocurre en MongoDB donde se utilizan documentos BSON que pueden contener estructuras anidadas y listas, donde además la normalización no es algo importante y se prefiere mejorar el rendimiento a normalizar.
- En MongoDB las **validaciones** son posibles pero no tan estrictas. Mientras que Oracle nos permite tener una base de datos rígida gracias a claves foráneas, checks o triggers.
- MongoDB no es adecuado para implementar un **modelo de transacciones ACID**