Taller Mongo

Base de datos Masivas

Brayhan Alejandro Ortiz Ballesteros
NRC: 863617

Docente
Ing. William Matallana Porras

Corporación universitaria Minuto de Dios
Ingeniería de sistemas
Zipaquirá
2025

Tabla de contenido

Introducción

MongoDB es una base de datos NoSQL ampliamente utilizada debido a su flexibilidad y escalabilidad. A diferencia de las bases de datos relacionales como MySQL, MongoDB emplea un modelo orientado a documentos en formato BSON, lo que permite almacenar datos estructurados, semiestructurados y no estructurados sin la necesidad de un esquema fijo. Esto facilita la gestión y manipulación de grandes volúmenes de información, lo que lo hace ideal para aplicaciones modernas como Big Data, IoT y microservicios. En este documento, se explorarán las diferencias clave entre MongoDB y MySQL, el almacenamiento de datos en MongoDB y los comandos esenciales para realizar operaciones CRUD.

Objetivos

- Explicar qué es MongoDB y en qué se diferencia de las bases de datos relacionales como MySQL.
- Comprender el modelo de almacenamiento de MongoDB y su formato BSON.
- Identificar las ventajas de MongoDB en términos de escalabilidad y flexibilidad.
- Conocer los comandos esenciales para la administración de bases de datos, colecciones y usuarios en MongoDB.
- Aprender a realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) en MongoDB de manera eficiente.

Desarrollo

¿Qué tipo de base de datos es MongoDB y en qué se diferencia de una base de datos relacional como MySQL?

MongoDB es una base de datos NoSQL de código abierto. Como base de datos no relacional, puede procesar datos estructurados, semiestructurados y no estructurados. Utiliza un modelo de datos no relacional orientado a documentos y un lenguaje de consulta no estructurado. Mongo es una base de datos de

documentos que ofrece una gran escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado.

No requiere un sistema de gestión de bases de datos relacionales (RDBMS), por lo que proporciona un modelo de almacenamiento de datos flexible que permite a los usuarios almacenar y consultar tipos de datos multivariados con facilidad. Esto no únicamente simplifica la gestión de bases de datos para los desarrolladores, sino que también crea un entorno **altamente escalable** para aplicaciones y servicios multiplataforma.

MongoDB frente a MySQL

MySQL (enlace externo a IBM) utiliza un **lenguaje de consulta estructurado** para acceder a los datos almacenados. En este formato, los esquemas se utilizan para crear estructuras de bases de datos, utilizando tablas como una forma de estandarizar los tipos de datos para que los valores se puedan buscar y consultar correctamente. **MySQL**, una solución madura, es útil para una variedad de situaciones, incluidas bases de datos de sitios web, aplicaciones y manejo de productos comerciales.

Característica	MySQL	MongoDB
Modelo	Tablas relacionales	Documentos flexibles
Esquema	Rígido	Dinámico (schema-less)
Escalabilidad	Vertical	Horizontal (sharding)
Consultas	SQL	Métodos/operadores propios
Transacciones	ACID (multi-tabla)	ACID (multi-documento desde v4.0+)
Fortalezas	Integridad y relaciones	Velocidad y escalabilidad

¿Qué es una colección en MongoDB y en qué se diferencia de una tabla en SQL?

Los documentos o colecciones de documentos de MongoDB son las unidades básicas de datos. Con un formato JSON binario (notación de objetos de secuencia de comandos de Java), estos documentos pueden almacenar **varios tipos de datos y distribuirse en diferentes sistemas**. Dado que MongoDB está diseñado bajo un esquema dinámico, los usuarios tienen una flexibilidad de, al crear registros de datos, consultar colecciones de documentos a través de la agregación de MongoDB y analizar grandes cantidades de información.

En MongoDB, una colección es un grupo de documentos (similares a registros o filas en SQL) almacenados en formato flexible **BSON** (una variante binaria de **JSON**). Se diferencia de una tabla en **SQL** en varios aspectos clave:

Aspecto	Colección (MongoDB)	Tabla (SQL)
Esquema	Flexible (schema-less)	Rígido (predefinido)
Relaciones	Datos embebidos o referencias	Claves foráneas + JOINs
Escalabilidad	Horizontal (sharding automático)	Vertical o horizontal compleja
Flexibilidad	Alta (campos variables)	Baja (requiere migraciones)
Casos de uso	Datos no estructurados/volátiles	Datos estructurados y relaciones complejas

Operaciones comunes

Operación	MongoDB (Colección)	SQL (Tabla)
Insertar	db.usuarios.insertOne({ nombre: "Pedro" })	INSERT INTO usuarios (nombre) VALUES ('Pedro');
Consultar	db.usuarios.find({ edad: { $gt: 25 } })	SELECT * FROM usuarios WHERE edad > 25;
Actualizar	db.usuarios.updateOne({ _id: 1 }, { $set: { edad: 30 } })	UPDATE usuarios SET edad = 30 WHERE id = 1;

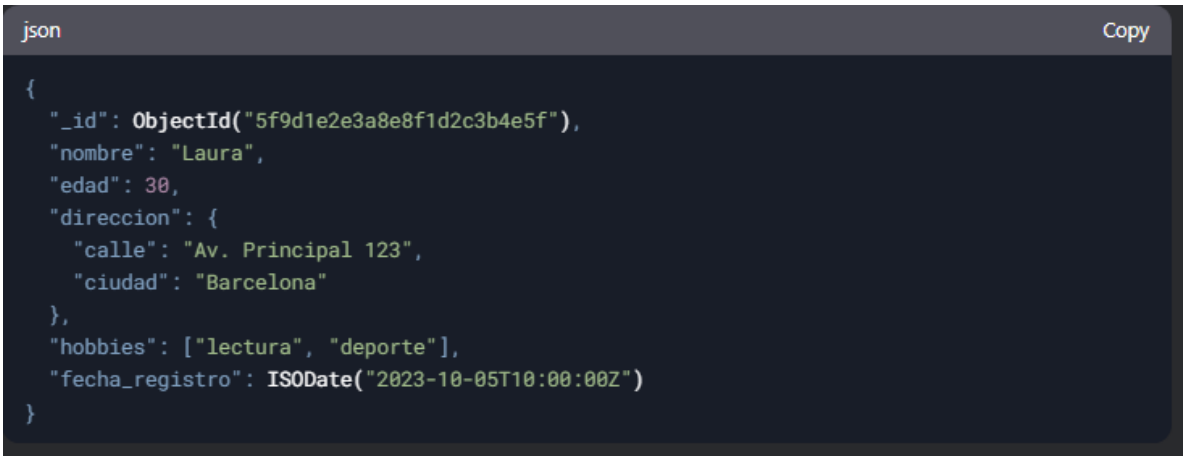
Una tabla en SQL es una estructura rígida diseñada para datos altamente normalizados, mientras que una colección en MongoDB es un contenedor flexible para documentos dinámicos.

¿Cómo se almacena la información en MongoDB y qué formato utiliza?

En MongoDB, la información se almacena en documentos BSON (una versión binaria de JSON) organizados en colecciones.

A medida que las empresas escalan sus operaciones, es fundamental obtener acceso a métricas clave y conocimientos comerciales a partir de grandes grupos de datos. MongoDB maneja la conversión de JSON y documentos similares a JSON, como BSON, en objetos Java **sin esfuerzo**, lo que hace que la lectura y escritura de datos en MongoDB sea **rápida e increíblemente eficiente** al analizar información en tiempo real en múltiples entornos de desarrollo.

- BSON (Binary JSON):
    - Es un formato binario ligero derivado de JSON, diseñado para ser eficiente en almacenamiento y procesamiento.
    - Soporta más tipos de datos que JSON estándar, como:
        - Date (fechas),
        - Binary (datos binarios, como imágenes),
        - ObjectId (identificador único de MongoDB),
        - Timestamp (marca de tiempo),
        - Geospatial (coordenadas geográficas).



Explica la diferencia entre JSON y BSON en MongoDB.

Binario vs. texto: BSON es un formato de codificación binaria, mientras que JSON es un formato de texto. Esto significa que BSON es compacto para la transmisión por red, mientras que JSON es legible y más fácil de usar en diversos contextos.

Compatibilidad con datos ampliada: JSON se limita a tipos de datos de JavaScript, como cadenas, números, booleanos, nulos, objetos y matrices. Estos tipos de datos pueden combinarse para representar tipos de datos complejos. BSON admite tipos de datos adicionales (como datos binarios y tipos de fecha) que no son compatibles con JSON.

Compatible con: BSON solo es compatible de forma nativa con MongoDB. JSON, por otro lado, tiene amplia compatibilidad y se puede usar con sistemas de bases de datos distribuidas , lenguajes de programación y plataformas.

Huella: En algunas situaciones, los documentos BSON pueden ser más grandes que los documentos JSON equivalentes porque incluyen metadatos adicionales e información de tipo que no está presente en JSON. Esto puede afectar los tiempos de transmisión y los requisitos de almacenamiento, especialmente para conjuntos de datos grandes. Tanto BSON como JSON pueden beneficiarse de la compresión.

Complejidad y compatibilidad: BSON es más complejo que JSON, lo que dificulta su uso en ciertos contextos. Los desarrolladores podrían necesitar aprender nuevos tipos de datos y métodos de codificación/decodificación para trabajar con

BSON eficazmente. También pueden surgir problemas de compatibilidad al intercambiar datos entre sistemas que no son totalmente compatibles con los tipos adicionales de BSON.

Característica	JSON	BSON
Definición	Formato de texto basado en JavaScript	Formato binario extendido de JSON
Eficiencia	Más fácil de leer y escribir, pero menos eficiente en almacenamiento y transmisión	Más rápido de procesar debido a su formato binario compacto
Soporte de Tipos de Datos	Soporta tipos básicos como cadenas, números, booleanos, arreglos y objetos	Soporta tipos adicionales como fechas, binarios, ObjectId, y decimales
Uso en MongoDB	Se usa en consultas y exportaciones de datos	Se usa internamente para almacenamiento y transmisión de datos
Tamaño	Puede ser más grande debido a etiquetas explícitas de claves y valores	Más eficiente en tamaño debido a codificación binaria

Estructura de los archivos json

La estructura de los datos JSON se basa en dos elementos: una colección de pares clave / valor y una lista ordenada de valores. Esto es similar a los tipos de datos utilizados en muchos lenguajes de programación, como los objetos y los arrays.

Reglas básicas del formato JSON

1. **Las claves deben estar entre comillas dobles** ("clave").
2. **Los valores pueden ser**:
    - Cadenas ("texto")
    - Números (10, 3.14)
    - Booleanos (true, false)

- null (sin valor)
- Objetos {} (estructuras anidadas)
- Arreglos [] (listas de valores)

3. **No se pueden usar comentarios** (a diferencia de otros formatos como YAML).

¿Qué ventajas tiene MongoDB sobre una base de datos relacional en términos de escalabilidad y flexibilidad?

MongoDB no tiene un esquema, lo que proporciona más flexibilidad y le permite trabajar con datos estructurados, semiestructurados y no estructurados. MySQL tiene un esquema rígido que funciona bien con datos estructurados. MongoDB usa los certificados Kerberos, X. 509 y LDAP para autenticar a los usuarios.

La mayor ventaja con respecto a las bases de datos relacionales, no obstante, la encontramos con la flexibilidad. Las bases de datos NoSQL permiten cambiar el esquema de datos de manera más ágil, incluso trabajar con elementos con distintos esquemas en la misma estructura.

MongoDB es una excelente opción para aplicaciones que necesitan gran escalabilidad, manejo flexible de datos y alta velocidad en consultas, como Big Data, IoT, aplicaciones en la nube y microservicios. Sin embargo, en sistemas que requieren alta consistencia y transacciones complejas, SQL sigue siendo una mejor opción.

Comandos para realizar CRUD en Mongo

Sintaxis del comando

```
db.runCommand ( { hello: 1 } )
```

Los comandos más importantes de MongoDB para empezar

Los comandos básicos de MongoDB te ayudarán a trabajar eficazmente con el software. A continuación, te mostramos los comandos que probablemente sean los más importantes para empezar:

Comandos de MongoDB	Descripción
db.help()	Este comando muestra todos los comandos disponibles de MongoDB.
mongo --version	Utiliza este comando para averiguar qué versión de MongoDB estás utilizando. Ejecuta el comando en la terminal de Linux o macOS. Si estás utilizando Windows, usa el símbolo del sistema CMD. A continuación, se mostrará la versión del shell que estás utilizando y el servidor MongoDB correspondiente.

Comandos para bases de datos

Para poder almacenar tus datos, necesitas bases de datos. Los siguientes comandos de MongoDB te resultarán particularmente importantes para trabajar con ellas:

Comandos de MongoDB	Descripción
show dbs	Muestra todas las bases de datos en forma de lista.
use DATABASE_NAME	Crea una nueva base de datos cuyo nombre puedes elegir libremente.
db	Pregunta qué base de datos has seleccionado.
db.dropDatabase()	Elimina la base de datos actualmente seleccionada.

Comandos para las colecciones

Mientras que las bases de datos relacionales como MySQL se basan en tablas, MongoDB utiliza colecciones. Los siguientes comandos de MongoDB se refieren al manejo de colecciones:

Comandos de MongoDB	Descripción
db.createCollection (Name, Options)	Crea una colección simple y especifica tu nombre (si es necesario, también otras opciones). La colección puede ser limitada.
show collections	Muestra y enumera todas las colecciones disponibles.
collectionName.drop()	Elimina una colección. Si la colección se ha eliminado con éxito, el sistema lo confirma con “true”. Si hay un error, aparece como “false”.

Administración de usuarios

Para trabajar con diferentes usuarios en una base de datos, tienes que crear perfiles de usuario y gestionarlos. Los siguientes comandos de MongoDB, entre otros, te ayudarán a hacerlo:

Comandos de MongoDB	Descripción
createUser (user, writeConcern)	Crea un nuevo usuario. Utiliza “writeConcern” para establecer un nivel de autorización.
dropUser	Elimina un usuario individual de la base de datos.
dropAllUsersFromDatabase	Elimina todos los usuarios depositados para una base de datos.
usersInfo	Muestra toda la información disponible sobre un usuario.
updateUser	Actualiza los datos de un usuario.
grantRolesToUser	Da a un usuario ciertos derechos o roles.
revokeRolesFromUser	Elimina ciertos derechos o roles de un usuario.

Comandos para roles

Puedes asignar a los usuarios determinados derechos o funciones. Para gestionar, especificar o eliminarlos puedes usar los siguientes comandos de MongoDB:

Comandos de MongoDB	Descripción
createRole	Crea un rol y define sus derechos y deberes.
rolesInfo	Consulta las especificaciones de un rol en particular.
updateRole	Actualiza un rol y la información existente.
dropRole	Elimina un rol específico.
dropAllRolesFromDatabase	Elimina todos los roles de una base de datos.
grantPrivilegesToRole	Añade privilegios claramente definidos a un rol.
revokePrivilegesFromRole	Elimina los privilegios individuales de un rol.
grantRolesToRole	Define los roles cuyos privilegios se transfieren a otro rol.
revokeRolesFromRole	Elimina los roles heredados.
invalidateUserCache	Borra la caché de usuarios y elimina la información sobre los roles.

Añadir y gestionar documentos

Para llenar las colecciones, asígnales documentos específicos o crea otras nuevas usando los siguientes comandos de MongoDB:

Comandos de MongoDB	Descripción
insert	Añade un documento (o varios) a una colección.
update	Actualiza uno o varios documentos.
delete	Elimina los documentos de una colección.
find	Selecciona y muestra documentos específicos de una colección.
findAndModify	Muestra y modifica un documento específico.

Comandos de MongoDB	Descripción
getMore	Da salida a los documentos seleccionados con el cursor.
getLastError	Muestra el estado de la última operación realizada.

Agrupar y clasificar

Para poder clasificar aún mejor los documentos, la base de datos ofrece los llamados comandos de agregación. La agrupación se realiza con los siguientes comandos:

Comandos de MongoDB	Descripción
aggregate	Documentos agrupados.
count	Cuenta los diferentes documentos de una colección.
distinct	Muestra los valores definidos y determina la frecuencia con la que aparecen en una colección.
mapReduce	Se utiliza para grandes conjuntos de datos y los reduce u ordena.

Comandos de MongoDB relevantes para la seguridad

MongoDB también es muy adecuado para trabajar con datos sensibles, porque el sistema permite limitar los accesos y ofrece opciones para proteger los registros de datos mediante autenticación. Para usar estas opciones debes conocer los siguientes comandos:

Comandos de MongoDB	Descripción
authenticate	Inicia una sesión autentificada que requiere un nombre de usuario y una contraseña.
getnonce	Genera una contraseña única para un inicio de sesión protegido.
logout	Finaliza la sesión protegida actual.

Comandos para las sesiones

Desde la versión 3.6 o 4, MongoDB también ofrece comandos para sesiones específicas. Estos comandos pueden ser interesantes para tu trabajo:

Comandos de MongoDB	Descripción
startSession	Inicia una nueva sesión.
refreshSessions	Actualiza las sesiones inactivas.
endSessions	Termina las sesiones antes de la hora prevista.
killSessions	Detiene las sesiones específicas y fijas.
killAllSessions	Detiene todas las sesiones inmediatamente.
killAllSessionsByPattern	Detiene todas las sesiones que coinciden con ciertos parámetros definidos.
commitTransaction	Realiza una transacción.
abortTransaction	Cancela una transacción.

Otros comandos administrativos

MongoDB tiene otros comandos para facilitar el trabajo administrativo. Algunas de las más importantes se enumeran en la siguiente tabla por orden alfabético:

Comandos de MongoDB	Descripción
cloneCollectionAsCapped	Copia una colección sin límite como una nueva colección con límite.
collMod	Añade opciones a una colección.
compact	Desfragmenta una colección y rediseña los índices.
convertToCapped	Convierte una colección sin límite en una colección con límite.
createIndexes	Añade uno o más índices a una colección.
getParameter	Muestra las opciones de configuración.

Comandos de MongoDB	Descripción
listIndexes	Enumera todos los índices disponibles de una colección.
setParameter	Modifica las opciones de configuración.
shutdown	Interrumpe el proceso de MongoDB o Mongos.

Diagnóstico y seguimiento

MongoDB también proporciona comandos para la supervisión y el diagnóstico. Estos son algunos de ellos:

Comandos de MongoDB	Descripción
dbStats	Proporciona estadísticas sobre el tipo y la utilización de una base de datos específica.
features	Enumera todas las funciones disponibles.
serverStatus	Indica el estado del servidor en uso.
buildInfo	Muestra toda la información disponible sobre la compilación actual de MongoDB.
connectionStatus	Proporciona información sobre la conexión actual.
dataSize	Muestra el tamaño de un archivo o de una serie seleccionada de archivos diferentes.
setFreeMonitoring	Permite o prohíbe el monitoreo libre durante el tiempo de ejecución.

Consulta que arroja Chat GPT

1. CREATE (Insertar datos)

🔹 Insertar un solo documento

```
db.usuarios.insertOne({ nombre: "Juan", edad: 25, ciudad: "Bogotá" })
```

🔻 Insertar múltiples documentos

```
db.usuarios.insertMany([
  { nombre: "María", edad: 30, ciudad: "Lima" },
  { nombre: "Carlos", edad: 28, ciudad: "Buenos Aires" }
])
```

2. READ (Leer datos)

🔻 Obtener todos los documentos

```
db.usuarios.find()
```

🔻 Filtrar documentos con una condición

```
db.usuarios.find({ ciudad: "Bogotá" })
```

🔻 Mostrar solo algunos campos

```
db.usuarios.find({ ciudad: "Bogotá" }, { nombre: 1, _id: 0 })
```

🔻 Obtener un solo documento

```
db.usuarios.findOne({ nombre: "Juan" })
```

3. UPDATE (Actualizar datos)

🔻 Actualizar un solo documento

```
db.usuarios.updateOne(
  { nombre: "Juan" },
  { $set: { edad: 26 } }
)
```

🔻 Actualizar múltiples documentos

```
db.usuarios.updateMany(
  { ciudad: "Lima" },
  { $set: { pais: "Perú" } }
)
```

4. DELETE (Eliminar datos)

🔻 Eliminar un solo documento

```
db.usuarios.deleteOne({ nombre: "Carlos" })
```

🔻 Eliminar múltiples documentos

```
db.usuarios.deleteMany({ ciudad: "Lima" })
```

Extras

🔻 Contar documentos

```
db.usuarios.countDocuments()
```

🔻 Ordenar resultados

```
db.usuarios.find().sort({ edad: 1 })    // 1: Ascendente, -1: Descendente
```

🔻 Limitar resultados

```
db.usuarios.find().limit(5)
```

Estos son los comandos esenciales para manejar bases de datos en **MongoDB**. 🚀

Cómo se pueden relacionar datos en Mongo sin usar joins como en sql

Podemos unir documentos en colecciones en MongoDB mediante la función $lookup (Agregación). $lookup (Agregación) crea una unión externa izquierda con otra colección y ayuda a filtrar datos de los datos fusionados.

Descargar imagen de mongo en docker

Descargar la imagen

docker pull mongo:latest

```
C:\WINDOWS\system32\CMD.exe
Microsoft Windows [Versión 10.0.19045.5608]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Brayhan>docker pull mongo:latest
latest: Pulling from library/mongo
5a7813e071bf: Pull complete
d67c4ebf9460: Pull complete
7afa02f8c09e: Pull complete
4e7ca17a42bd: Pull complete
342a4f4728ff: Pull complete
d5bafd14fbe8: Pull complete
0c492c8e8cfd: Pull complete
734719e891c0: Pull complete
Digest: sha256:7bd28e5eea1c5766a084d5818254046f3ebe3b8f20a65e3a274640189e296667
Status: Downloaded newer image for mongo:latest
docker.io/library/mongo:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview mongo:latest

C:\Users\Brayhan>
```

Herramientas similares a Workbench para visualizar los datos de mongo.

MongoDB Compass

La herramienta oficial de MongoDB proporciona una interfaz gráfica para explorar y analizar sus datos, permitiendo visualizar la estructura de la base de datos, ejecutar consultas y optimizar el rendimiento.

Robo 3T (anteriormente Robomongo)

Ofrece una interfaz ligera y fácil de usar para interactuar con bases de datos MongoDB, permitiendo la edición y visualización de documentos, así como la ejecución de consultas.

DBeaver

Esta aplicación de código abierto es compatible con múltiples bases de datos, incluyendo MongoDB. Proporciona una interfaz unificada para administrar diferentes sistemas de bases de datos, lo que facilita el trabajo con múltiples fuentes de datos.

Wikipedia, la enciclopedia libre

Navicat for MongoDB

Una herramienta de gestión de bases de datos diseñada para profesionales que trabajan con MongoDB. Su interfaz intuitiva y características avanzadas facilitan la gestión de grandes conjuntos de datos y flujos de trabajo complejos.

DronaHQ

phpMoAdmin

Similar a phpMyAdmin pero para MongoDB, es una interfaz web ligera que permite

administrar bases de datos MongoDB desde el navegador, ofreciendo funcionalidades básicas de gestión y consulta.

Conclusiones

MongoDB es una base de datos NoSQL que ofrece una alternativa flexible y escalable a los sistemas tradicionales de bases de datos relacionales. Su capacidad para manejar datos sin un esquema rígido y su modelo de almacenamiento basado en documentos permiten optimizar el rendimiento en aplicaciones que requieren alta velocidad y adaptabilidad. Además, el uso de BSON mejora la eficiencia del almacenamiento y procesamiento de datos. Al comprender las diferencias con MySQL y dominar los comandos esenciales para la gestión de bases de datos, colecciones y usuarios, los desarrolladores pueden aprovechar al máximo las capacidades de MongoDB en entornos dinámicos y distribuidos.

Bibliografía

Mongo:

https://www.mongodb.com/es/company/what-is-mongodb

https://www.ibm.com/mx-es/topics/mongodb#:~:text=MongoDB%20(enlace%20externo%20a%20IBM,almacenar%20diversas%20formas%20de%20datos.

Bson:

https://www.couchbase.com/resources/concepts/json-vs-bson/#:~:text=Binary%20vs.,work%20with%20in%20various%20contexts.

Json:

https://www.arsys.es/blog/formato-json-que-es-y-para-que-sirve#:~:text=La%20estructura%20de%20los%20datos,los%20objetos%20y%20los%20arrays.

Ventajas:

https://www.arsys.es/blog/bases-de-datos-nosql-que-son-tipos-y-ventajas#:~:text=La%20mayor%20ventaja%20con%20respecto,esquemas%20en%20la%20misma%20estructura.

Comandos:

https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/mongodb-commands/

Join Mongo:

https://hevodata.com/learn/mongodb-join-two-collections/#:~:text=We%20can%20join%20documents%20on,filter%20data%20from%20merged%20data.