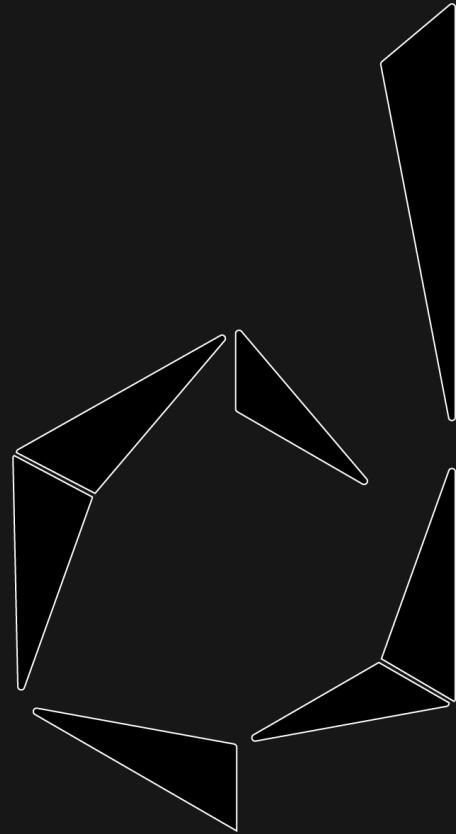


INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

PROGRAMACIÓN: DESARROLLO BACKEND/BASES DE DATOS

NoSQL

Bases de Datos No Relacionales
Basadas en Documentos: MongoDB

Contenido

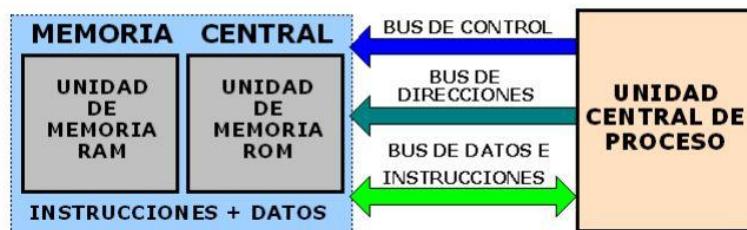
Introducción a las Bases de Datos	2
Tipos de Bases de Datos.....	2
Representación de las Bases de Datos: Nomenclatura de Chen	3
Bases de Datos No Relacionales	3
Jerarquía de Datos de las Bases de Datos No Relacionales basadas en Documentos.....	5
Tipos de Escalamiento: Horizontal y Vertical	7
Réplicas o Sharding (Cluster)	8
Instalación de MongoDB Cloud - Mongo Atlas	8
NRDBMS (Non-Relational DB Management System) - Mongo Compass	15
Consultas NoSQL en MongoDB - Mongo en VSCode con Lenguajes de Programación.....	20
.gitignore y .editorconfig - Mongo Query Language en Visual Studio Code	24
Carpetas src/playground - Mongo Query Language	25
¿Qué es Docker 🐳 ?	27
Instalación de Docker en Windows con WSL (Windows Subsystem for Linux).....	27
Ejemplo de Imagen: MongoDB en Docker 🐳	33
Referencias.....	35



Introducción a las Bases de Datos

Las **bases de datos** ayudan a complementar la **arquitectura de Von Neumann**, que es utilizada en ordenadores, la cual a diferencia de la arquitectura **Harvard** usada en microcontroladores, **utiliza una sola memoria para realizar sus funciones y guardar sus datos**. La necesidad de extender la capacidad de la memoria central es la de conservar los datos más allá de la memoria RAM o ROM, ya que en la arquitectura **Von Neumann** si se contempla el procesamiento de datos, pero no el almacenamiento de datos persistentes, por lo que es de suma importancia la utilización de las **databases (DB)**.

ARQUITECTURA VON NEUMANN



Para resolver esta situación, donde se busca que de una forma fácil se puedan **guardar y extraer datos** de información, se obtuvieron dos soluciones:

- **Bases de datos basadas en archivos:** Este **método de almacenamiento de datos persistentes** consiste en **guardar información en un archivo de texto plano, hojas de cálculo, etc.** usualmente separados por comas o de alguna otra forma ordenada.
- **Bases de datos basadas en documentos:** En este tipo de **base de datos**, la unidad básica de almacenamiento es el **documento**, que **puede contener datos en forma de texto, números, listas, objetos JSON (JavaScript Object Notation)** y a veces incluso otros documentos anidados.

Tipos de Bases de Datos

Los diferentes tipos de bases de datos existentes son los siguientes:

- **Relacionales o RDB:** Son **bases de datos basadas en documentos** y están gobernadas por las **12 reglas de Edgar Codd**, que dan como resultado el álgebra relacional, a través de las cuales se indican las reglas con las que los **datos** de las **RDB (Relational Databases)** se pueden relacionar.
 - **Privadas:** Microsoft SQL Server, **Oracle**, etc.
 - **Open Source:** **PostgreSQL**, MySQL, MariaDB, etc.

Ejemplos de bases de datos relacionales



- **No relacionales o NRDB:** Hay varios tipos de **bases de datos no relacionales**, todas ellas pueden ser muy distintas unas de otras, pero se engloban dentro de la misma categoría de **NRDB (Non Relational Databases)** porque utilizan lenguajes **NoSQL (Not Only SQL)** para sus consultas. Los diferentes tipos a grandes rasgos son:
 - Basadas en Clave-Valor, en Documentos, en Grafos, en Memoria, Optimizadas para Búsquedas, etc. Algunos ejemplos de ellas son:
 - Memcached, Cassandra (Facebook), DynamoDB, ElasticSearch, BigQuery, Neo4j (GraphQL), **MongoDB**, **Firestore (Firebase)**.

Bases de datos no relacionales



Representación de las Bases de Datos: Nomenclatura de Chen

- **Entidad:** Se refiere a una **tabla** que almacena datos sobre un tipo de objeto o elemento del mundo real.
 - Cada **fila** en la **tabla** representa una **instancia individual** de esa **entidad**.
 - Cada **columna** en la **tabla** representa un **atributo o característica** de esa **entidad**.
- **Atributo:** Son las **columnas de una tabla** que representan las **características o propiedades** de la **entidad** que está siendo modelada, todas ellas tienen un **nombre y tipo de dato** asociado.
- **Registro:** Representa una **fila perteneciente a una tabla**. También es conocido como "**tupla**" y contiene los **valores** de los **atributos** correspondientes a una **instancia** específica de una **entidad**.

Bases de Datos No Relacionales

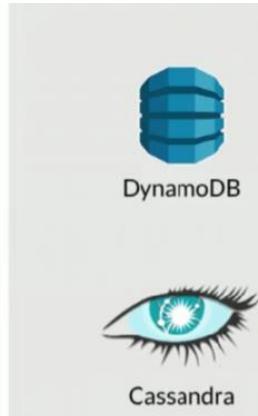
Como ya se había mencionado previamente, las **bases de datos no relacionales** o **NRDB (Non Relational Data Bases)** **no se conforman de un solo tipo de bases de datos**, sino de **varias**, y aunque puedan ser muy distintas unas de otras, todas se engloban dentro de la misma categoría de **base de datos no relacional**, ya que utilizan lenguajes **NoSQL (Not Only SQL)**. Los diferentes tipos de **databases no relacionales** son:

- **NRDB Basadas en Clave-Valor:** Estas **bases de datos no relacionales** están diseñadas para **almacenar y recuperar información** de manera rápida mediante el uso de una **clave**. **Cada clave** **está asociada a un valor**, que puede ser un **dato simple**, **un objeto** o un **conjunto de datos**. Las consultas se realizan utilizando estas **claves únicas** y su principal **característica** es su **alta velocidad y eficiencia en operaciones de lectura y escritura de datos**.

- Algunos ejemplos de estas bases de datos no relacionales son **DynamoDB de AWS**, Cassandra de Facebook y Redis.

Clave - valor

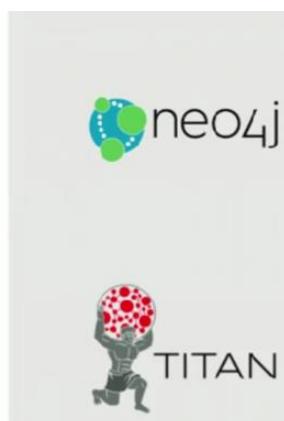
Son ideales para almacenar y extraer datos con una clave única. Manejan los diccionarios de manera excepcional.



- **NRDB Basadas en Grafos:** Los grafos se componen de **nodos** o **entidades (tablas)** que tienen **relaciones** muy complejas unas con otras y generalmente se **conectan** entre sí, creando redes de **datos**, se usan para crear **inteligencias artificiales (redes neuronales) o redes sociales**.
 - Algunos ejemplos de estas **bases de datos no relacionales** son Neo4j y Titan.

Basadas en grafos

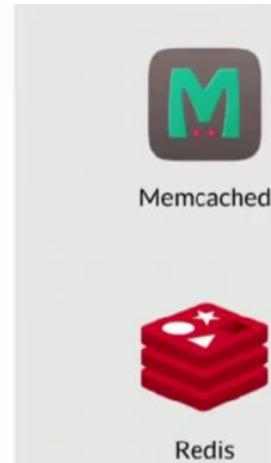
Basadas en teoría de grafos sirven para entidades que se encuentran interconectadas por múltiples relaciones. Ideales para almacenar relaciones complejas.



- **NRDB Basadas en Memoria:** Este tipo de **bases de datos** son sumamente rápidas, pero tienen la gran desventaja de que son volátiles, osea que **su memoria no es duradera**.

En memoria

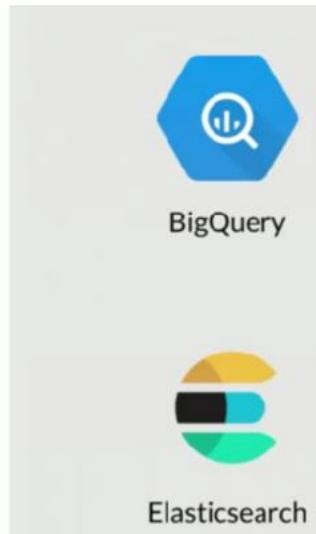
Pueden ser de estructura variada, pero su ventaja radica en la velocidad, ya que al vivir en memoria la extracción de datos es casi inmediata.



- **NRDB Optimizadas para Búsquedas:** Este tipo de **base de datos** pueden ejecutar Queries muy complejas de forma muy rápida y a grandes **repositorios de datos históricos** que almacenan un gran volumen de información, son muy utilizadas en aplicaciones de **business intelligence y machine learning**.
 - Algunos ejemplos de estas bases de datos no relacionales son **BigQuery de Google** y Elasticsearch.

Optimizadas para búsqueda

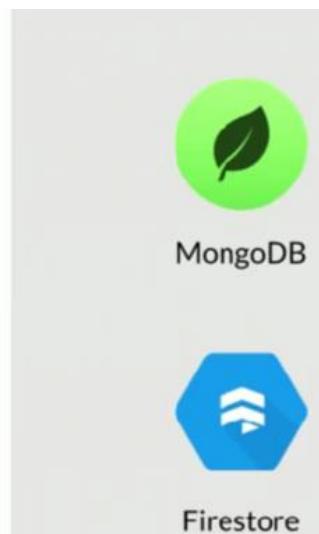
Pueden ser de diversas estructuras, su ventaja radica en que se pueden hacer queries y búsquedas complejas de manera sencilla.



- **NRDB Basadas en Documentos:** En estas **bases de datos no relacionales** se empareja cada **clave** con una estructura de **datos** llamada **Documento** que es mayormente utilizado para referirnos a **archivos de tipo JSON (JavaScript Object Notation) o XML**.
 - Algunos ejemplos de estas bases de datos no relacionales son **MongoDB**, **FireStore de Google**, Couchbase, etc.

Basados en documentos

Son una implementación de clave valor que varía en la forma semiestructurada en que se trata la información. Ideal para almacenar datos JSON y XML.



Jerarquía de Datos de las Bases de Datos No Relacionales basadas en Documentos

En las bases de **datos no relacionales basadas en documentos**, en vez de que se cuente con **tablas (entidades)**, **atributos (columnas)**, **relaciones (conexiones)**, **etc.** su estructura se basa en **colecciones de**

datos, que son el equivalente a las **entidades**, las cuales clasifican los distintos **documentos** que contienen estructuras JSON y estos de forma interna asocian cada **valor** de la DB con una **clave**.



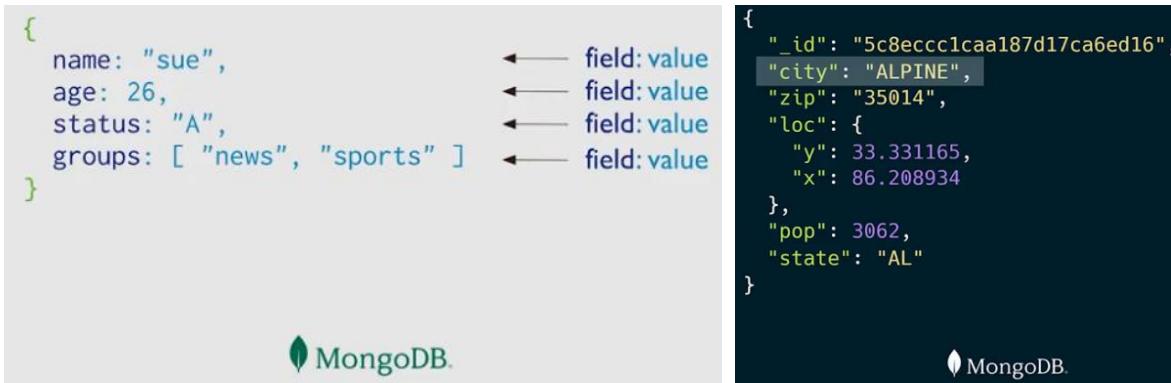
Cuando trabajamos con bases de datos basadas en documentos como **MongoDB** o **FireStore**, cambiaremos el concepto de las **tablas** por las **colecciones** y las **tuplas (filas)** por los **documentos**.

- **Entidad o Tabla** → **Colección**.
- **Tuplas o Filas** → **Documento**, que almacena sus **datos** en forma de **clave:valor**, donde cada **clave** corresponde a las **columnas** de la **tabla** y su valor a cada **instancia** específica.

Además, en el contexto de las **colecciones**, identificamos dos categorías principales:

- Las "**Top level collections**" o **Colecciones de nivel superior**.
- Y las "**subcollections**" o **subcolecciones**, que se incorporan dentro de otra **colección**, **así como en las estructuras JSON, podemos meter un JSON dentro de otro**, de igual forma se puede introducir un **documento** dentro de otro o una **colección** dentro de otra.

No existe una regla estricta para determinar si la **colección** debe ser de **nivel superior** o una **subcolección** al crear una **base de datos basada en documentos**; más bien, esta decisión depende del caso de uso específico.



Una consideración clave al diseñar la **database no relacional** es anticipar cómo se **extraerán los datos**. En el contexto de una aplicación, es útil pensar en términos de las vistas que se mostrarán en un momento específico. En otras palabras, al estructurar la **DB**, debemos asegurarnos de que refleje o contenga, al menos, todos los **datos necesarios** para satisfacer los requisitos visuales de nuestra **aplicación** en un momento dado.

Esta regla se aplica salvo algunas excepciones, como cuando se tiene una entidad que necesita existir y modificarse de manera constante e independiente de otras colecciones.

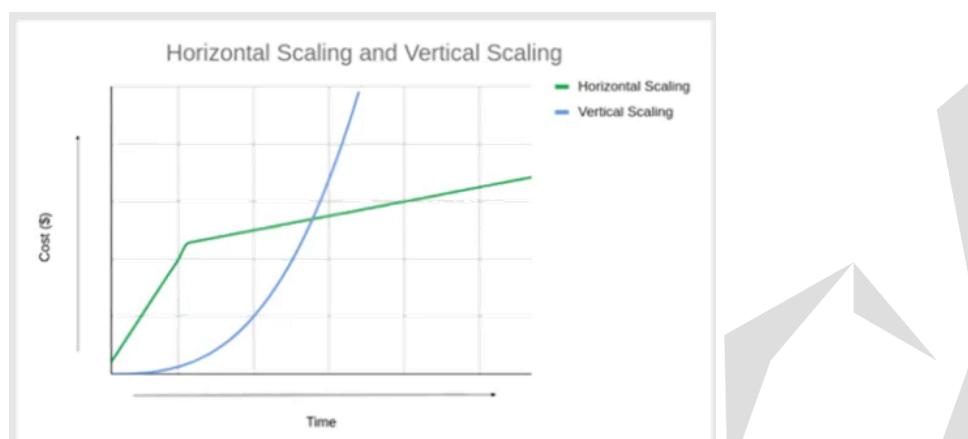
Tipos de Escalamiento: Horizontal y Vertical

El **escalamiento** se refiere a la habilidad de nuestras **bases de datos** o sus **servidores (ordenadores donde se encuentran almacenadas las DB)** de adaptarse a las necesidades que requiera la aplicación o el usuario. Existen dos tipos de escalamiento:

- **Escalamiento vertical:** Este escalamiento se refiere a **aumentar las capacidades del servidor** donde se encuentra montada la **base de datos**, incrementando los límites de sus recursos como **su procesador, memoria RAM, espacio de almacenamiento, etc.**
- **Escalamiento horizontal:** Este escalamiento se refiere a crear varias **réplicas (o nodos)** de las **bases de datos** en **diferentes servidores**, lo cual hace que, **con recursos limitados de los servidores, nos aseguremos que la DB posea alta disponibilidad de datos, copias de seguridad, un sistema en conjunto que responda de forma simultánea, etc.** sin necesidad de aumentar los recursos de las máquinas.



Las **bases de datos no relacionales** son más propensas a permitir el **escalamiento horizontal** y aunque el **escalamiento vertical** en un inicio es más fácil implementarlo, a la larga es más costoso, mientras que el escalamiento horizontal es más costoso en un inicio, pero si aumentan los requerimientos del sistema, su costo se mantiene.



Réplicas o Sharding (Cluster)

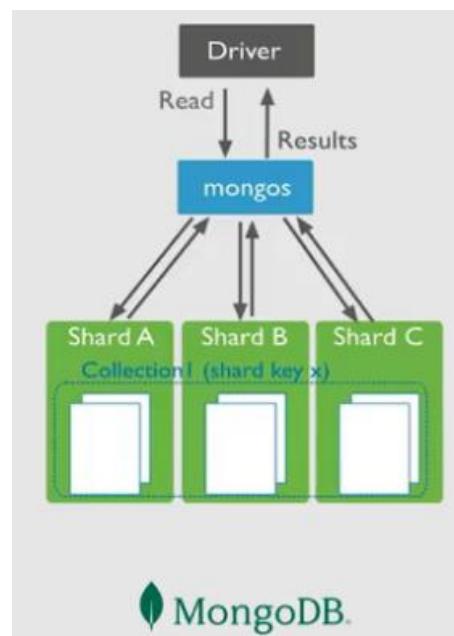
Réplica o Shard se refiere al proceso de **copiar y mantener actualizada una copia de la base de datos en un servidor diferente**.

Esta técnica es utilizada para evitar problemas de **entrada y salida de datos** en los sistemas operativos, ya que, si nuestra aplicación realiza **peticiones de lectura y escritura de datos** de forma exponencial; como **no se puede leer y escribir datos** en una **tabla** al mismo tiempo, esta se bloquea durante dicho proceso, pero cuando se tienen manejo de datos múltiples, existen límites electrónicos o físicos, que restringen la capacidad de procesamiento del CPU, por lo que una **petición podría tardar minutos en ejecutarse** y esto es catastrófico.

Por eso es tan importante el uso de **réplicas**, donde al menos se cuenta con **dos servidores distintos (aunque pueden ser más de 2)**, uno como **master** y el otro es la **réplica**:

- Se tiene un **servidor** con un **database principal llamado master**, donde solo se realizan las **entradas o modificaciones de datos**.
- Y otro **servidor** con una **base de datos secundaria** (que es la **réplica**), donde solo se realiza la **lectura de datos**.

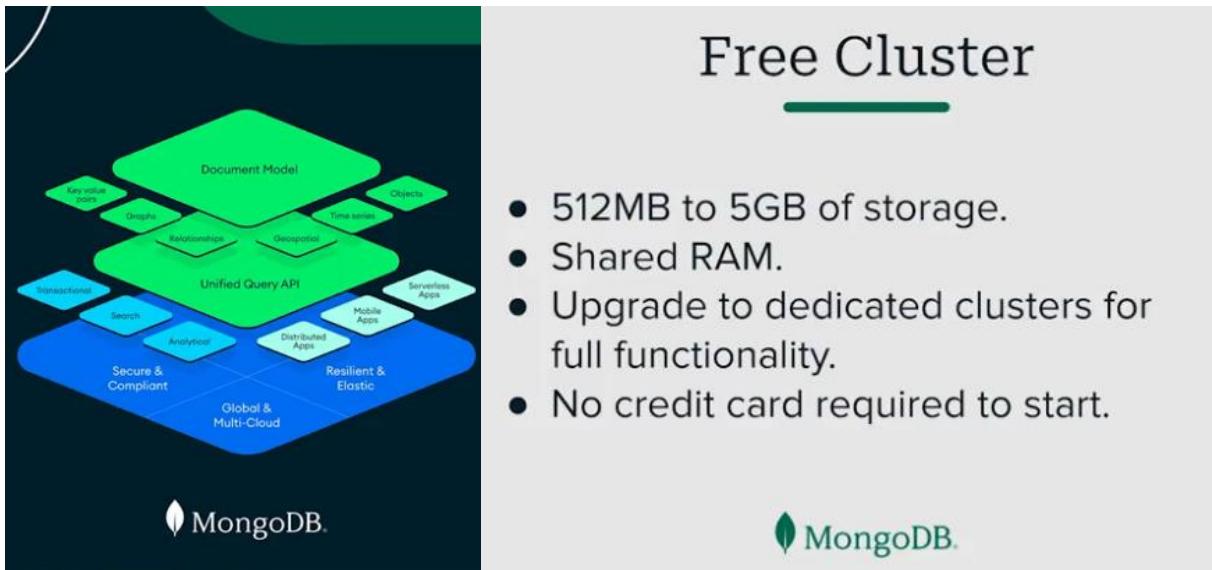
Un **cluster** en **MongoDB** es un **conjunto de servidores que trabajan juntos** para proporcionar alta disponibilidad, escalabilidad y redundancia (duplicación para crear una copia de seguridad de los datos) dirigidos para soportar las necesidades de una **base de datos no relacional**.



Instalación de MongoDB Cloud - Mongo Atlas

Mongo Atlas es un servicio en la nube que nos permite administrar nuestra **base de datos no relacional** con un motor de **MongoDB**. Este **como parte de su servicio incluye un sistema de replicación y**

clusterización de datos, permitiendo así la creación de un modelo de escalamiento muy fácil de implementar. Para ello usaremos el plan Free Cluster, que proporciona ciertas características de almacenamiento gratuitas dentro del servicio de nube de **Mongo Atlas**.

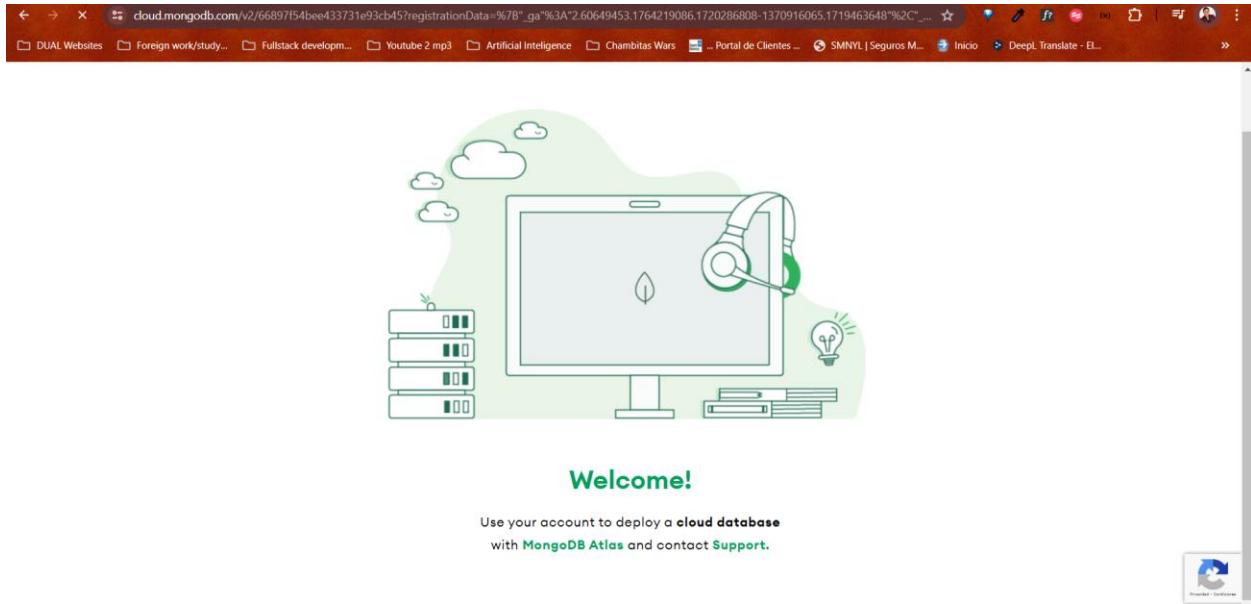
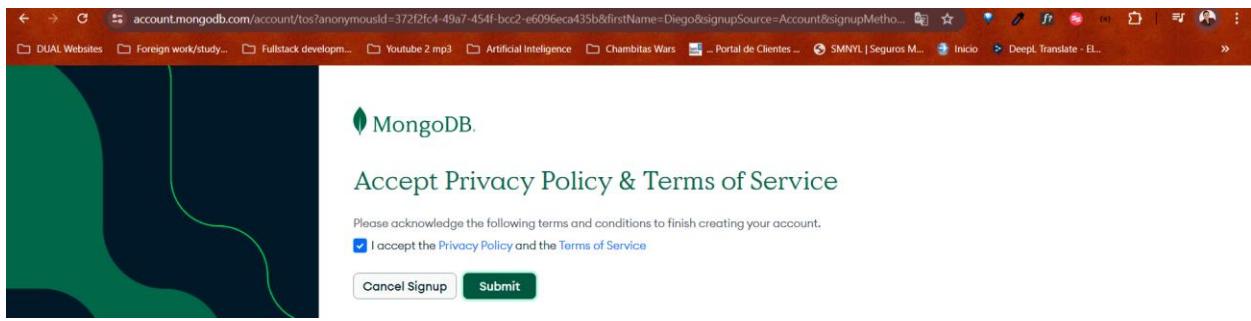


Para ello ingresaremos a la página oficial de **MongoDB**, nos registraremos con nuestro correo o con una cuenta de **GitHub**.

<https://www.mongodb.com/>

The screenshot shows the MongoDB homepage with the following elements:

- Header:** MongoDB logo, navigation menu (Products, Resources, Solutions, Company, Pricing), search bar, Eng dropdown, Support, Sign In, Try Free button.
- Middle Section:** Headline "Loved by developers. Built for Time Series Data". Subtext: "You don't need a separate database to support transactions, rich search, or gen AI. The world's most popular document database is now the world's most versatile developer data platform." Call-to-action buttons: "Try Atlas Free" and "Deploy your way".
- Right Side:** A grid of five cards: "Row wise time", "Market Price", "Commodity Data", "Regular Markets", and a "Contact Us" button.
- Bottom Section:** A GitHub OAuth authorization dialog titled "Authorize MongoDB Atlas". It shows a user profile picture, the text "MongoDB Atlas by 10gen wants to access your discord account", and a "Personal user data" section. Buttons: "Cancel" and "Authorize 10gen". Below the dialog, it says "Authorizing will redirect to http://auth.mongodb.com". At the bottom, there are checkboxes for "Not created or operated by GitHub" and "More than 1K GitHub users".



Luego se nos realizará un cuestionario para personalizar la creación de la **base de datos no relacional** dependiendo de la función que vaya a llevar a cabo, el tipo de proyecto, los **tipos de datos** que almacenará, el lenguaje de programación utilizado, nuestra experiencia con **MongoDB**, etc.

The screenshot shows a web browser window with the URL cloud.mongodb.com/v2/66897f54bee433731e93cb45#setup/personalization. It displays two main sections: "GETTING TO KNOW YOU" and "GETTING TO KNOW YOUR PROJECT".

- GETTING TO KNOW YOU:**
 - What is your primary goal? (dropdown: Learn MongoDB)
 - How long have you been developing software with MongoDB? (dropdown: I've never developed software with MongoDB before)
- GETTING TO KNOW YOUR PROJECT:**
 - What programming language are you primarily building on MongoDB with? (dropdown: Python)
 - What type(s) of data will your project use? (dropdown: Not sure...)
 - Will your application include any of the following architectural models? (dropdown: Not sure...)

A large, semi-transparent 3D geometric shape (hexagon) is overlaid on the right side of the form. A "Finish" button is located at the bottom right of the form area.

En esta parte es donde elegiremos el plan de almacenamiento, que en este caso será el **M0 Free Cluster**, que nos permite almacenar hasta 10,000 documentos.

Deploy your cluster

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

Plan	Cost
M10	\$0.08/hour
Serverless	Up to 1TB, Auto-scale
M0	Free

M0 cluster details:
For learning and exploring MongoDB in a cloud environment.
STORAGE: 512 MB, RAM: Shared, vCPU: Shared

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Name: Cluster0

Luego se asignará el nombre del **cluster MongoDB**, el **proveedor de servicios de nube** que puede ser **AWS**, **Google Cloud** o **Azure** y finalmente la ubicación geográfica del servidor de nube (la cual se elegirá dependiendo de la cercanía a nuestra propia localización) y daremos clic en el botón de **Create Cluster**.

Name: 1-ExampleMongoDB

Automate security setup (checked)

Preload sample dataset (checked)

Provider: AWS

Region: N. Virginia (us-east-1)

Deployment buttons: I'll do this later, Go to Advanced Configuration, Create Deployment

Ahora indicaremos nuestro **usuario administrador** de la **base de datos** y su contraseña.

Connect to 1-ExampleMongoDB

You can't connect yet. Set up your firewall access and user security permissions first. Then you'll be able to see your connection options.

Create a database user

This first user will have **atlasAdmin** permissions for this project. You'll need your database user's credentials to connect to your cluster. We autogenerated an username and password. You can use this or create your own. You can edit, delete, or add database users in [Database Access](#).

Username: di_cero

Password: [Copy](#)

Create User

Add a connection IP address

Your current IP address (38.186.31.214) has been added to enable local connectivity. You can edit, delete, or add IP addresses in [Network Access](#).

Finish setting up your security access

You'll need to set up both firewall access and user security permissions to view your connection string.

Resources

- Get started with the Python Driver
- Python Starter Sample App
- Access your Database Users

Aquí es donde entra en juego el lenguaje de programación que hayamos indicado previamente durante la configuración de **MongoDB**, ya que se nos preguntará cómo es que nos queremos conectar al **cluster** de la **DB** que acabamos de crear:

- Se nos indicará que el **usuario** que acabamos de crear tendrá **permisos de super user**.
- Se asigna por defecto una **dirección IP de conexión**, aunque estas se pueden agregar, editar o eliminar al dar clic en el enlace de Network Access.
- Se indica la librería y forma de conexión que se debe realizar dentro del código con el lenguaje de programación que hayamos indicado durante la configuración para poder acceder, borrar o insertar información de la base de datos y la línea de código que se debe agregar en el programa para establecer su conexión.

The screenshot shows two tabs of a MongoDB connection configuration interface. Both tabs have a similar layout with sections for selecting a database user, client category, driver, and version. The top tab is titled "Connect to 1-ExampleMongoDB" and contains a "Customized instructions based on your inputs" section with steps for installing the driver and adding a connection string. It also includes a "Sample Code" section with Python code for connecting to the database. The bottom tab has a similar structure but lacks the "Customized instructions" and "Sample Code" sections. Both tabs include a "Close" button in the top right corner.

Cuando hayamos terminado todo este proceso, damos clic en el botón de close.

Ahora podremos cargar **datos** a la **database**, realizarles Queries, etc.

También podemos ver que dentro del dashboard se indican las **organizaciones (que existan previamente en GitHub)** a las que se puede asociar cada proyecto. Las organizaciones se pueden configurar o crear nuevas al dar clic en el engrane que se encuentra a su lado derecho y se pueden crear nuevos proyectos dentro de cada organización al dar clic en la opción de + New Project.

Ahora dentro de los **clusters**, daremos clic derecho en los 3 puntos de la esquina superior derecha y seleccionaremos la opción de **View all clusters** o directo en **DEPLOYMENT → Database**.

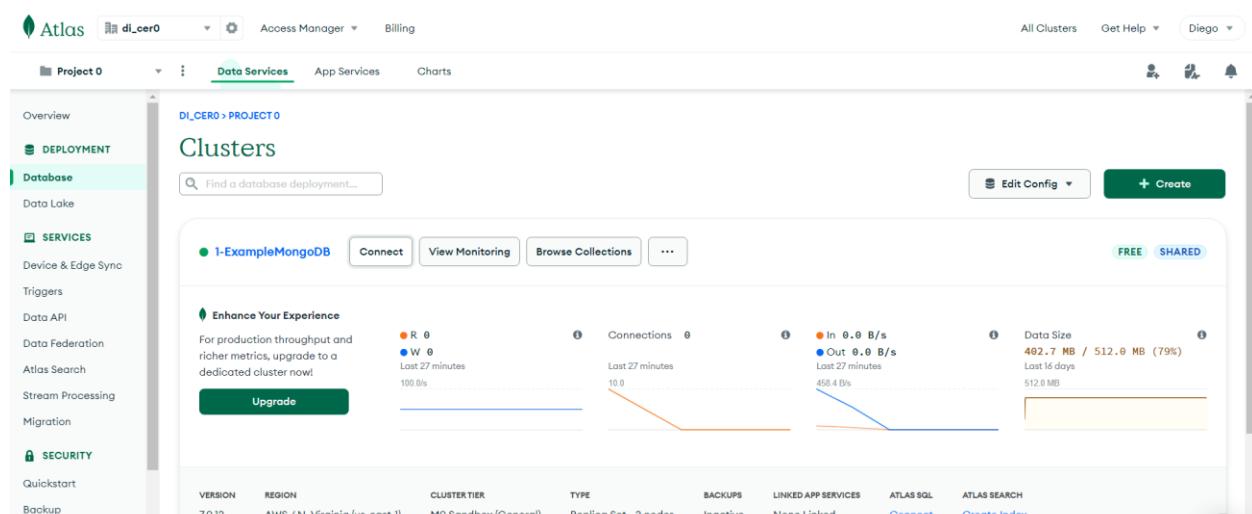
Después podremos ver ciertas características del **cluster** que creamos y para poder agregar datos de muestra a él para realizar pruebas seleccionaremos la opción de ... → Load Sample Data Set → Load Sample Data Set → Browse Collections → Aquí dentro se deberían ver todas las **colecciones** de **datos** de muestra insertadas en la **base de datos** de **MongoDB**, donde se destaca una **Colección de Airbnb**. La documentación de estos datos de muestra se encuentra en el siguiente enlace:

<https://www.mongodb.com/docs/atlas/sample-data/>

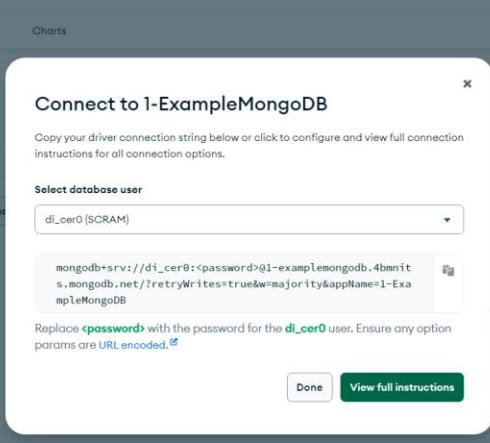
NRDBMS (Non-Relational DB Management System) - Mongo Compass

Ahora para que podamos realizar **consultar a la información** de nuestra **base de datos** usaremos un software creado por **MongoDB** llamado **Mongo Compass**, el cual es un **NRDBMS (Non Relational DataBase Management System)** que podrá conectarse a la plataforma cloud remota **Mongo Atlas**, pero de igual manera nos podría permitir consultar información que tengamos en una base de datos local (que no se encuentre montada en un servicio de nube).

Cuando ya hayamos creado nuestra base de datos remota en **Mongo Atlas**, los pasos a seguir con los que podremos realizar la instalación del **NRDBMS Mongo Compass** es al seleccionar la opción de: DEPLOYMENT → Database → Connect → View Full Instructions → Select Client Category → Developer Tools → Choose a Tool → **Compass (GUI)** → Select your operating system → Windows 64-bit (10+) → Costumized instructions Based on your inputs → **Download Compass** → Ejecutar .exe descargado → Abrir **MongoDBCompass** → New Connection + → Copiar y pegar el connection string que se encuentra en el punto número 2 → Poner mi password en la URI copiada → Connect.



The screenshot shows the MongoDB Atlas interface. On the left, there's a sidebar with options like Overview, Deployment (selected), Database, Services, and Security. The main area is titled 'Clusters' and shows a single cluster named '1-ExampleMongoDB'. It has tabs for 'Connect', 'View Monitoring', 'Browse Collections', and '...'. Below the tabs, there's a section titled 'Enhance Your Experience' with a button to 'Upgrade'. To the right, there are four performance metrics: R: 0, W: 0 (Last 27 minutes), Connections: 0 (Last 27 minutes), and In: 0.0 B/s, Out: 0.0 B/s (Last 27 minutes). At the bottom, there's a table with columns: VERSION, REGION, CLUSTER TIER, TYPE, BACKUPS, LINKED APP SERVICES, ATLAS SQL, and ATLAS SEARCH. The cluster details are: VERSION 7.0.12, REGION AWS / N. Virginia (us-east-1), CLUSTER TIER M0 Sandbox (General), TYPE Replica Set - 3 nodes, BACKUPS Inactive, LINKED APP SERVICES None Linked, ATLAS SQL Connect, and ATLAS SEARCH Create Index.



This screenshot shows a modal dialog titled 'Connect to 1-ExampleMongoDB'. It contains instructions to copy the driver connection string or view full connection instructions. A dropdown menu 'Select database user' shows 'di_cero (SCRAM)'. Below it is a code snippet of a MongoDB connection URI: `mongodb+srv://di_cero:<password>@1-examplemongodb.4bmnit.mongodb.net/?retryWrites=true&w=majority&appName=1-ExampleMongoDB`. A note says to replace '<password>' with the password for the 'di_cero' user. At the bottom are 'Done' and 'View full instructions' buttons.

Connect to 1-ExampleMongoDB

Configure your connection below and follow the instructions on the left. Instructions will be updated based on your configurations.

Select database user: `di_cero (SCRAM)`

Select client category:

- Drivers: Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)
- Developer Tools: Access your Atlas data through tools like Compass, Shell, VS Code

Choose a tool:

- MongoDB Shell
- Compass (GUI)**
- MongoDB for VS Code

I have installed Compass

Select your operating system: Windows 64-bit (10+)

Customized Instructions based on your inputs:

- Download MongoDB Compass and version: [Download Compass \(1.43.4\)](#) or [Copy download URL](#)
Compass is an interactive tool for querying, optimizing, and analyzing your MongoDB data.
- Copy the connection string, then open MongoDB Compass:
`mongodb+srv://di_cero:<password>@1-examplemongodb.4bmnts.mongodb.net/`
Replace `<password>` with the password for the `di_cero` user. Ensure any option params are URL encoded.

Resources:

- Connect with Compass
- Import and Export Data
- Access your Database Users
- Troubleshoot Connections

CLOSE

← Todo Aplicaciones Documentos Web Configuración → 50 ⚡ 🧑 ... 🌈

Mejor coincidencia

MongoDBCompass Aplicación

Aplicaciones

mongodb-compass-1.43.4-win32-x64.exe

Buscar en Internet

mong - Ver más resultados de la búsqueda

MongoDB

Mongolia

MongoDBCompass Aplicación

- Abrir
- Ejecutar como administrador
- Abrir ubicación del archivo
- Anclar a Inicio
- Anclar a la barra de tareas
- Desinstalar

MongoDB Compass

Connect Edit View Help

Compass

New connection +

Saved connections

Recents

New Connection

Connect to a MongoDB deployment

URI: `mongodb+srv://di_cero:<password>@1-examplemongodb.4bmnts.mongodb.net/`

Edit Connection String

Advanced Connection Options

Save Save & Connect Connect

New to Compass and don't have a cluster? If you don't already have a cluster, you can create one for free using [MongoDB Atlas](#)

CREATE FREE CLUSTER

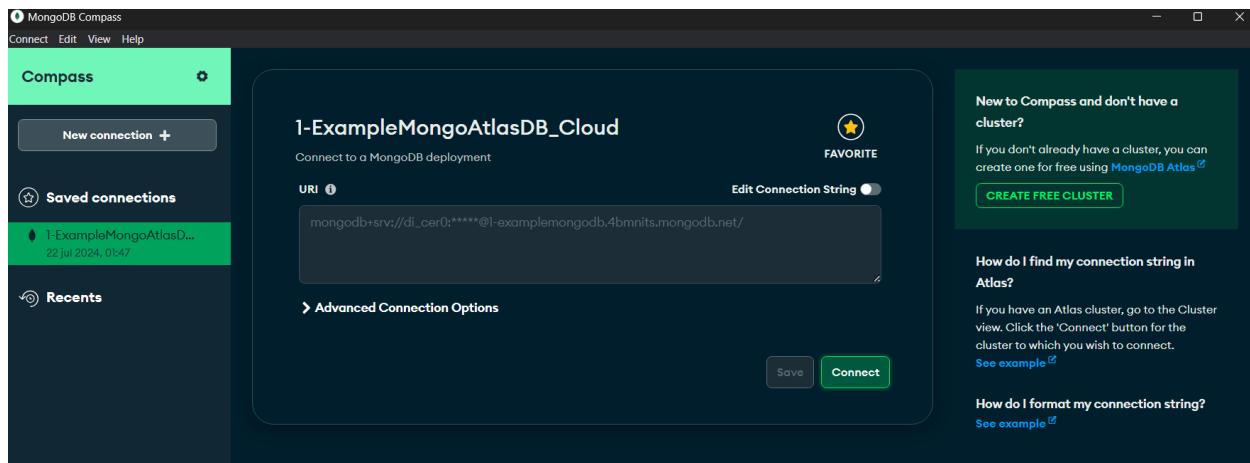
How do I find my connection string in Atlas? If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect. See example

How do I format my connection string? See example

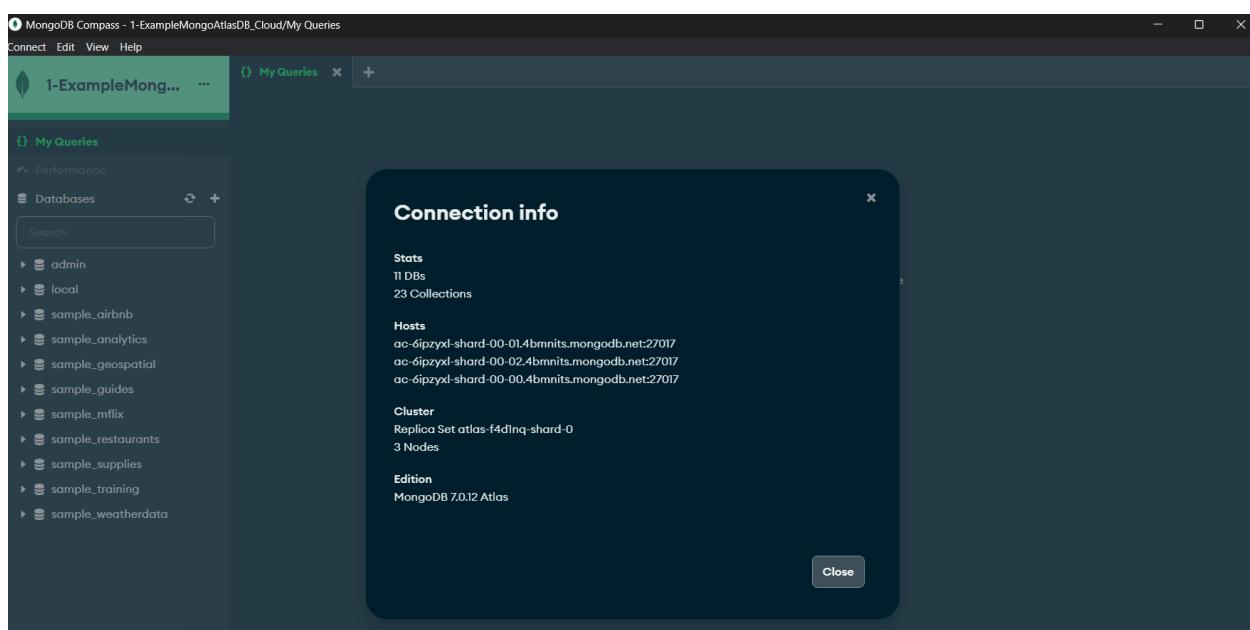
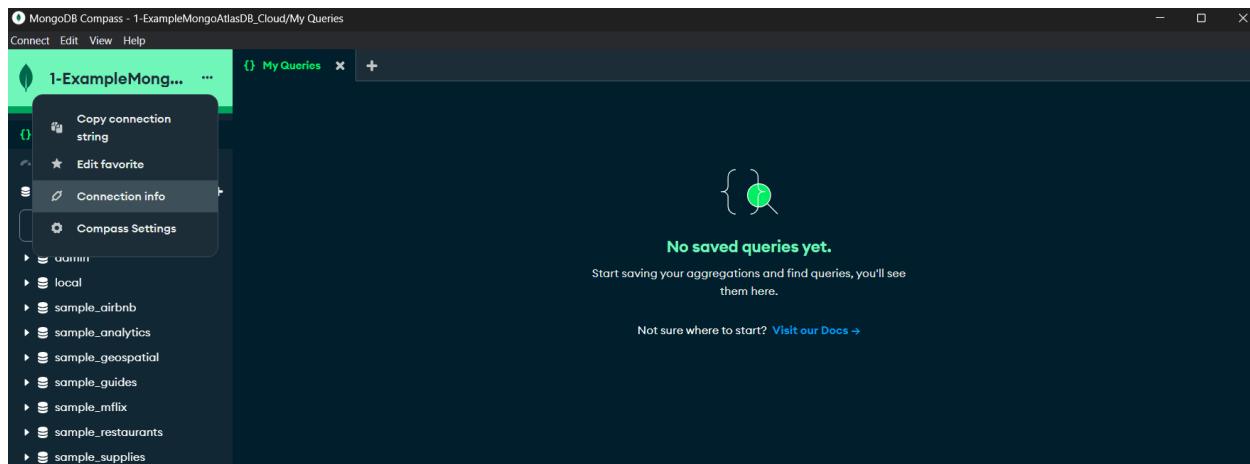
Ahora para ponerle un nombre a esta conexión deberemos dar clic en la opción de Connect → Disconnect → New Connection → Flechita de Editar → Nuevo Nombre de Conexión → Color del Fondo del espacio donde aparece el nombre de la conexión → Save → Saved Connections → Connect.

The screenshots illustrate the steps to save a new connection:

- Screenshot 1: My Queries View**
Shows the main interface with a sidebar of databases. The central area displays a message: "No saved queries yet." with a small icon of a person holding a query symbol.
- Screenshot 2: New Connection Dialog**
Shows the "New Connection" dialog. It includes fields for "URI" (set to "mongodb+srv://di_cer0:*****@l-examplemongodb.4bmnts.mongodb.net/"), "Edit Connection String" (checkbox), and "FAVORITE" (star icon). Below the dialog are "Save", "Save & Connect", and "Connect" buttons. To the right, there's a sidebar with help links for creating a cluster and finding connection strings.
- Screenshot 3: Save Connection Dialog**
Shows the "Save connection to favorites" dialog. It has a "Name" field containing "1-ExampleMongoAtlasDB_Cloud" and a "Color" picker with several color swatches. Buttons for "Cancel" and "Save" are at the bottom. The background shows the previous "New Connection" dialog.



Dentro de la conexión con **cluster** de la **base de datos** podremos ver sus características al seleccionar la opción de: Nombre de Conexión → ... → Connection info, que indica el número de **replicas** con las que cuenta (**Hosts**), Los **nodos** del **cluster**, la edición del motor de **MongoDB**, etc.



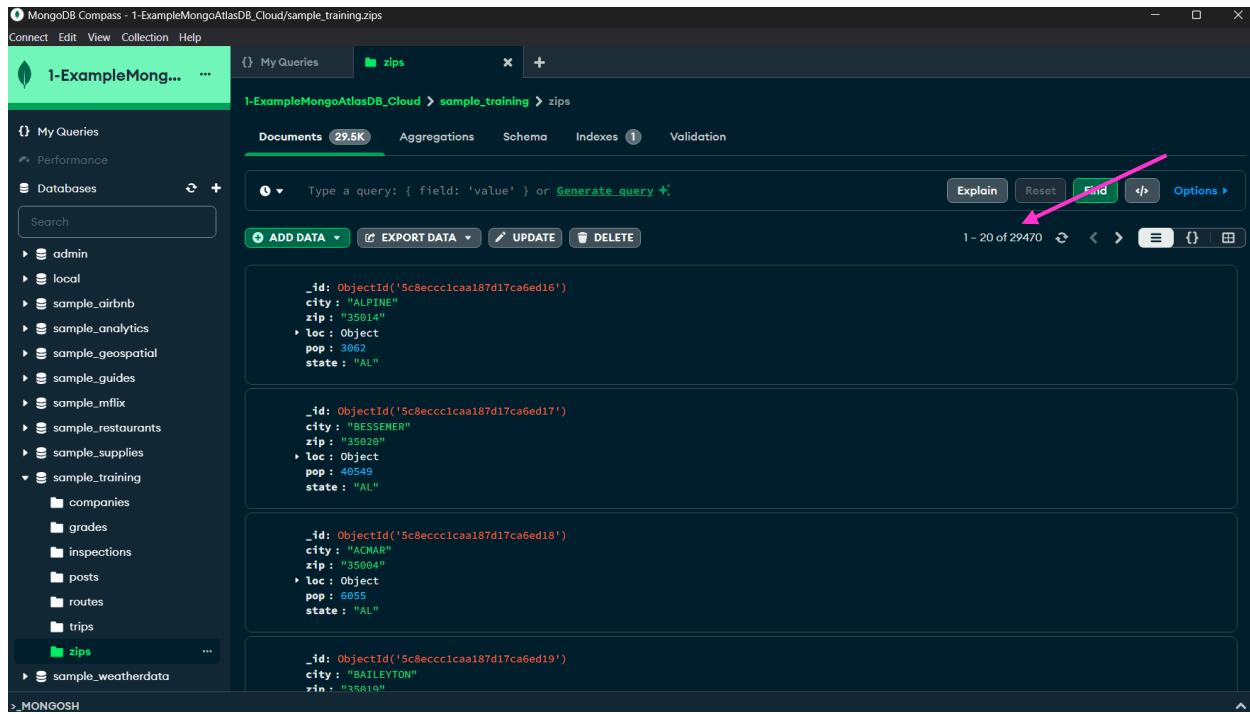
Dentro de la conexión de **Mongo Compass** recordemos que al trabajar con **bases de datos no relacionales basadas en documentos** como **MongoDB o FireStore**, cambiaremos el concepto de las **tablas** por las **colecciones** y las **tuplas (filas)** por los **documentos**.



- **Entidad o Tabla** → **Colección**.
- **Tuplas o Filas** → **Documento**, que almacena sus **datos** en forma de **clave:valor**, donde cada **clave** corresponde a las **columnas** de la **tabla** y su valor a cada **instancia** específica.

Dentro del **NRDBMS Mongo Compass** encontraremos del lado izquierdo todas las **bases de datos no relacionales** que existan dentro del **cluster**, pero al seleccionar cada una, podremos ver las **colecciones** que las conforman y dentro de las colecciones se podrá visualizar el número de **documentos** que le corresponde a cada una; aquí es donde ya se podrán **realizar consultas** a cada **colección (tabla)** de la **database** y al hacerlo podremos ver que el número de **documentos (filas o tuplas de datos)** cambia.

Collection	Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
companies	16.58 MB	9.5 K	3.84 kB	1	290.82 kB
grades	7.10 MB	100 K	233.00 B	1	3.86 MB
inspections	8.62 MB	80 K	276.00 B	1	2.32 MB
posts	4.24 MB	500	34.91 kB	1	32.77 kB
routes					



Consultas NoSQL en MongoDB - Mongo en VSCode con Lenguajes de Programación

Se puede utilizar la API de **MongoDB** para hacer consultas con el código **Mongo Query Language** directamente hacia **Mongo Atlas** o una **base de datos no relacional local** (almacenada en nuestra computadora) a través del IDE Visual Studio Code (VSCode), la extensión que permite hacer esto se llama **MongoDB for VSCode** y es obtenida directamente dentro de VSCode o a través de la siguiente opción en el sitio oficial de **MongoDB**: Products → View All Products → Scroll Down → VS Code → Download Now → Install. Y sirve como un mini **Mongo Compass** para que podamos hacer consultas, ejecutar rutinas de inserción de datos, etc.

<https://www.mongodb.com/>

The screenshot shows the MongoDB website's "Explore MongoDB offerings" section. It features a dark header with the MongoDB logo and navigation links like Products, Resources, Solutions, Company, and Pricing. A search bar and language selection (Eng) are also present. Below the header, a large title "Explore MongoDB offerings" is centered. To the left, a section titled "I want to..." lists several options with icons and descriptions:

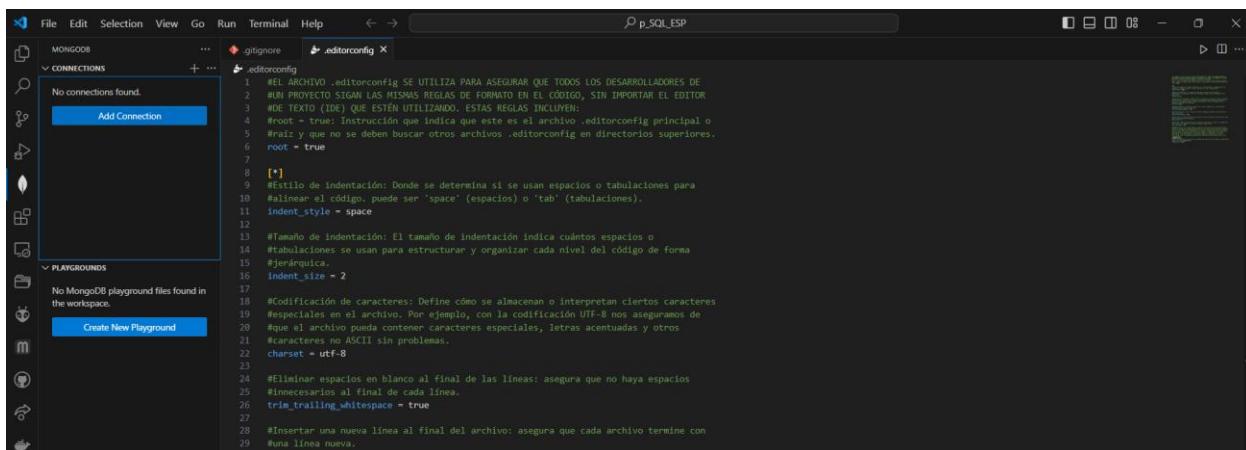
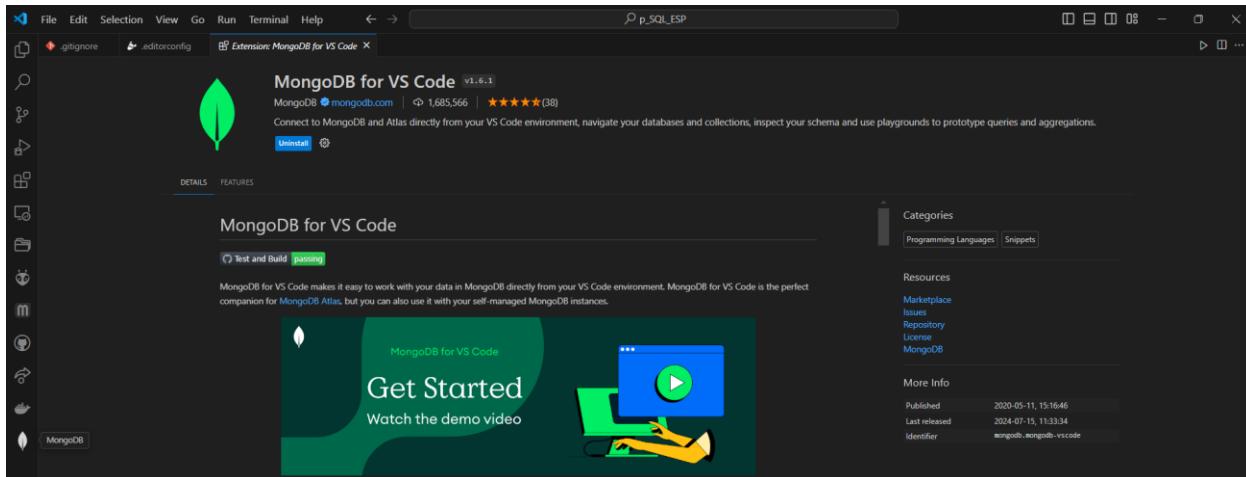
- Model, build, and interact with data**: Define document schemas, write queries, and interact with your data in MongoDB. [Explore developer tools >](#)
- Search, analyze, and visualize data**: Design engaging end-user applications, unlock AI-powered experiences, and discover insights from your data in MongoDB.
- Migrate to MongoDB**: Move your data from an existing database to MongoDB with tools and fully managed services.
- Run MongoDB**: Run MongoDB anywhere and move or sync data across environments for a single connected data layer.

On the right side, there are two more tool cards: "Compass" (Free GUI for querying, optimizing, and analyzing MongoDB data) and "Shell" (Javascript and Node.js REPL for querying, configuring, and executing automation scripts on MongoDB). A "Contact Us" button is located in the bottom right corner of the main content area.

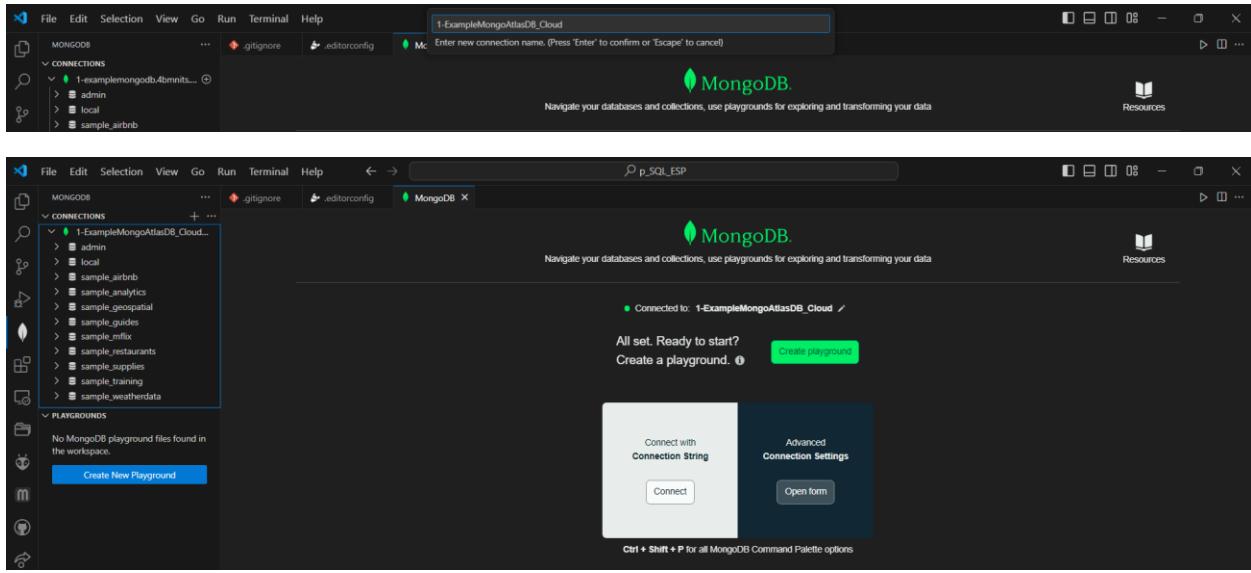
The screenshot shows the MongoDB VS Code extension landing page. The URL is "mongodb.com/products/tools/vs-code". The page has a dark header with the MongoDB logo and navigation links. A green banner at the top right states "Atlas Vector Search voted most loved vector database in 2024 Retool State of AI report [Read more >](#)". Below the header, a "TOOLS" section is visible. The main content features a large, bold title: "MongoDB for VS Code. Build Without Leaving Your IDE." A "Contact Us" button is in the bottom right. Below the title, a sub-headline says "Build applications and work with your data in MongoDB directly from your VS Code environment." Two buttons are at the bottom: "Download Now" and "Read documentation →".

The screenshot shows the MongoDB for VS Code extension page on the Visual Studio Marketplace. The URL is "VisualStudio | Marketplace". The page has a dark header with the Visual Studio Code logo and navigation links. A message box in the center says "Visual Studio Code is required to install this extension." Below it, a link says "Don't have Visual Studio Code? [Get it now.](#)". There is also a checkbox for "Don't show this message again" and a "Continue" button. The main content area shows the extension's details: "MongoDB for VS Code" by MongoDB, 1.685.887 installs, 5 stars (38 reviews), and Free. It describes the extension as connecting to MongoDB and Atlas directly from the VS Code environment. At the bottom, a note says "MongoDB for VS Code makes it easy to work with your data in MongoDB directly from your VS Code environment." Navigation links like Overview, Version History, Q & A, and Rate are at the bottom left. A "Test and Build" status bar at the bottom indicates "passing".

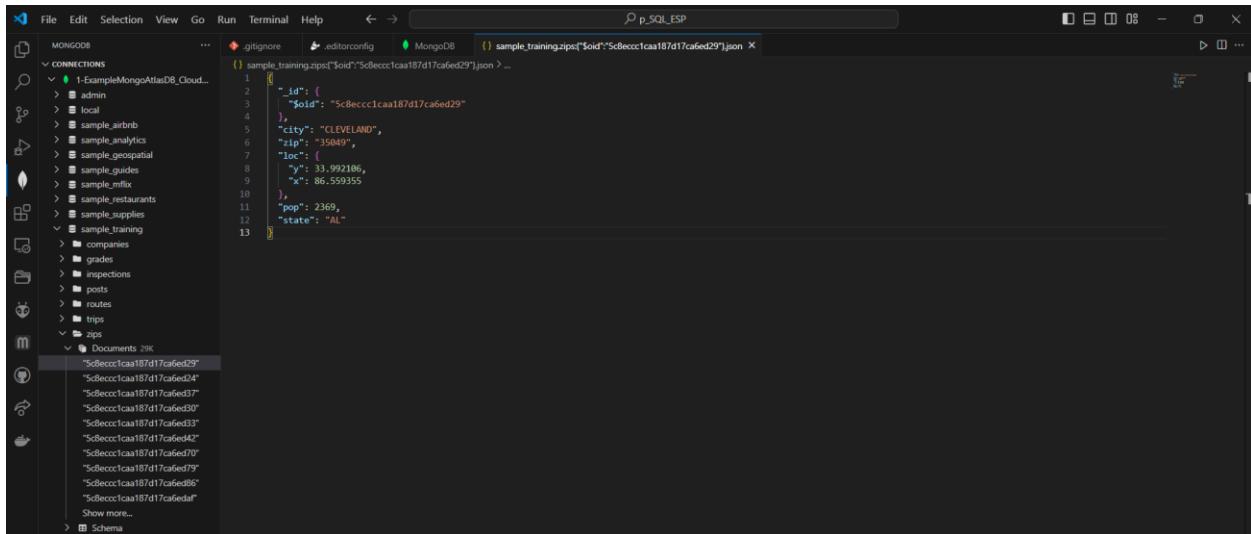
Al terminar de ejecutarse la instalación, aparecerá en el menú izquierdo la opción de MongoDB, ahora deberemos reiniciar VSCode y luego seleccionar la opción de: Menú Izquierdo de VSCode → MongoDB → Add Connection → Connect With Connection String → Connect → Pegar Connection String Copiado de la página de **Mongo Atlas**, la cual se obtiene al dar clic en la opción de: Mongo Altas → Connect → Copiar Connection String.



Dentro de VSCode podemos asignar un nombre a nuestra conexión de la misma forma como se hacía dentro del **NRDBMS (Non Relational DataBase Management System) Mongo Compass** al dar clic derecho y seleccionar la opción de Rename connection.



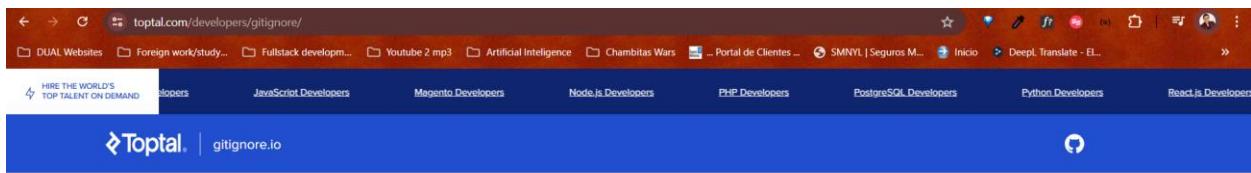
Dentro de estas conexiones podemos navegar en cada **colección (tabla)** de la **database** de **MongoDB** y ver sus **documentos (filas o tuplas de datos)** uno por uno.



.gitignore y .editorconfig - Mongo Query Language en Visual Studio Code

Es importante crear y añadir un archivo **.gitignore** dentro del repositorio de GitHub que estemos utilizando para realizar las consultas a través de código **Mongo Query Language** directamente hacia **Mongo Atlas** porque de esta manera evitamos que se suban archivos indeseables que solo utilicen espacio de forma innecesaria o que por seguridad no deben estar arriba de Git, para ello nos podemos apoyar de la siguiente página, que crea el .gitignore para ignorar archivos que se crean por defecto en los sistemas operativos Windows, Linux y Mac.

<https://www.toptal.com/developers/gitignore/>



gitignore.io

Create useful .gitignore files for your project



El archivo **.editorconfig** se utiliza para asegurar que todos los desarrolladores de un proyecto sigan las mismas reglas de formato en el código, sin importar el editor de texto (IDE) que usen. Estas reglas incluyen:

- **Estilo de indentación:** Donde se determina si se usan espacios o tabulaciones para alinear el código.
- **Tamaño de indentación:** El tamaño de indentación indica cuántos espacios o tabulaciones se usan para estructurar y organizar cada nivel del código de forma jerárquica.
- **Codificación de caracteres:** Define cómo se almacenan o interpretan ciertos caracteres especiales en el archivo. Por ejemplo, con la codificación UTF-8 nos aseguramos de que el archivo pueda contener caracteres especiales, letras acentuadas y otros caracteres no ASCII sin problemas.
- **Eliminación de espacios en blanco:** Borra los espacios innecesarios al final de cada línea.
- **Línea nueva al final del archivo:** Asegura que cada archivo termine con una línea en blanco.
- **Estilo de fin de línea:** Define qué carácter se usa para indicar el final de una línea (por ejemplo, LF para Unix/Linux).

Este archivo ayuda a mantener un código limpio y consistente, facilitando la colaboración entre los miembros del equipo.

Carpetas src/playground - Mongo Query Language

Cuando se quieran realizar consultas o manipulación de datos aplicados a una **base de datos MongoDB** a través de VSCode, se deben crear las siguientes carpetas que tendrán dentro archivos de código **Mongo Query Language** (que es una API basada en el lenguaje JavaScript) con extensión **.mongodb**:

- **src:** Esta carpeta contendrá otras carpetas llamadas playgrounds, donde se realizarán consultas parecidas entre sí, ya sea porque se aplican a una misma **DB**, obtienen datos relacionados, etc.
 - **playground:** Este folder incluirá todos los archivos con extensión **.mongodb** que ejecuten consultas (queries) hacia una **database MongoDB**, pero para ello previamente ya se debe haber creado un **cluster** en el servicio cloud **Mongo Atlas**, instalado la extensión **MongoDB for VSCode** en Visual Studio Code, creado una conexión entre ambas plataformas y utilizar el siguiente método perteneciente al

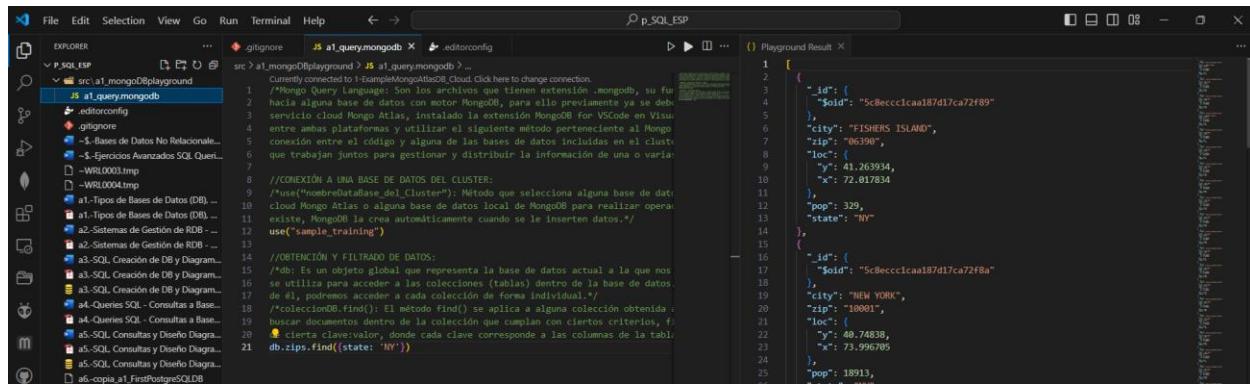
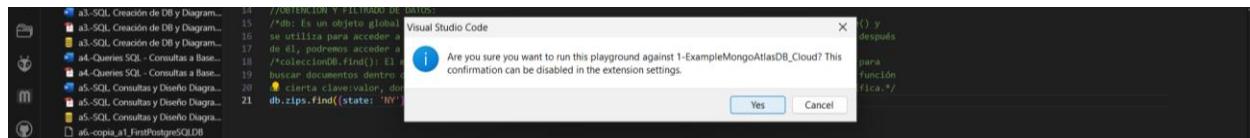
Mongo Query Language para establecer una conexión entre el código y alguna de las **bases de datos** incluidas en el **cluster**, el cual es un grupo de servidores que trabajan juntos para gestionar y distribuir la información de una o varias **databases**: `use("nombreDataBase_del_Cluster")`.

- Con las conexiones establecidas en el código podremos navegar en cada **colección (tabla)** de la **database** de **MongoDB** y acceder a sus **documentos (filas o tuplas de datos)**, pudiendo filtrar la información de sus **datos** a través de un filtro en forma de **clave:valor**, donde cada **clave** corresponde a las **columnas** de la **tabla** y su valor a cada **instancia** específica.

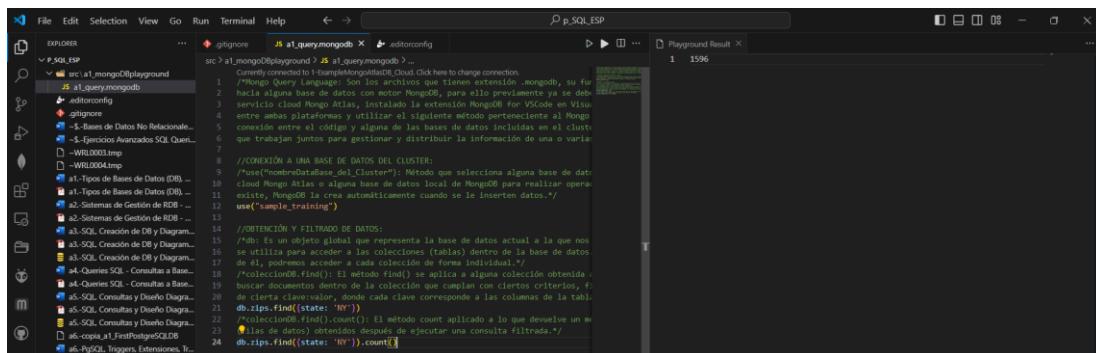
Ya que contemos con el código **Mongo Query Language** que realice una consulta, esta se podrá ejecutar al dar clic en el botón de Play que se encuentra en la esquina superior derecha del IDE VSCode.



Al ejecutar el query, aparecerá una advertencia a la que debemos dar clic en el botón de Sí y así podremos obtener los datos de la consulta en el editor de código.



No importando qué tipo de código estemos ejecutando, se mostrará el resultado del playground en el lado derecho del editor de código.



¿Qué es Docker ?

Docker 🚀 es una plataforma de código abierto que permite a los desarrolladores automatizar el despliegue, escalado y administración de aplicaciones tipo web, de escritorio, **bases de datos**, machine learning, data science, IoT, sistemas embebidos, etc. dentro de entornos ejecutables llamados contenedores.

Docker permite que sus aplicaciones se corran en **contenedores aislados**, lo que facilita su instalación, actualización y funcionamiento en cualquier entorno sin problemas de compatibilidad, ya que estos son entornos ligeros y portátiles que incluyen todo lo necesario para ejecutar una aplicación, desde el sistema operativo, las bibliotecas, las dependencias y el propio código. Algunas de sus características importantes son:

- **Puerto:** El que tiene asignado siempre por defecto es el **27017**.
- **Archivo de configuración:** Para configurar las características de un contenedor **Docker**, se debe crear un archivo llamado **docker-compose.yml**, cuya extensión significa **YAML (YAML Ain't Markup Language)**.
 - **Imagen:** Es un paquete que contiene todo lo necesario para ejecutar una aplicación específica, incluyendo el sistema operativo base, código, librerías y configuraciones. Es solo de lectura, lo que significa que no se puede modificar una vez creada (immutable). Las imágenes en **Docker** se utilizan para crear **contenedores**, que son instancias ejecutables de la **imagen**. Cada tipo de **imagen** para aplicaciones basadas en **MongoDB**, **Node.js**, **Django**, etc. se debe configurar de forma diferente, y esto se indica en la documentación de la página **Docker Hub**: <https://hub.docker.com/>

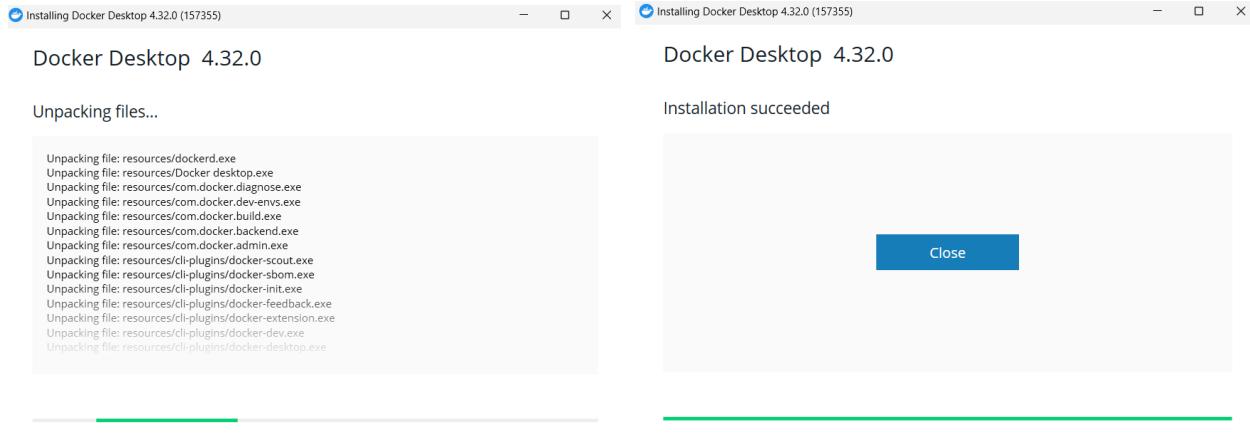
Instalación de Docker en Windows con WSL (Windows Subsystem for Linux)

Se debe descargar el instalador desde la página de Docker para Windows:

<https://docs.docker.com/desktop/install/windows-install/>

The screenshot shows the Docker documentation website for Docker Desktop on Windows. The main title is "Install Docker Desktop on Windows". A sidebar on the left lists options for Mac, Windows, and Linux. The Windows section is expanded, showing "Use the MSI installer [Early Access]" and "Understand permission requirements for Windows". The main content area includes a note about commercial use terms and download links for "Docker Desktop for Windows - x86_64" and "Docker Desktop for Windows - Arm (Beta)". The right sidebar contains links for "System requirements", "Install Docker Desktop on Windows", "Start Docker Desktop", and "Where to go next".

La instalación al ejecutar el archivo Docker Desktop Installer.exe es muy sencilla, cuando este acabe, se debe realizar cierta configuración dentro de Docker para que funcione sin problemas.



Ya que se haya instalado la aplicación de Docker Desktop, debemos abrir su interfaz y asegurarnos que la opción “Use the WSL 2 based engine” esté habilitada:



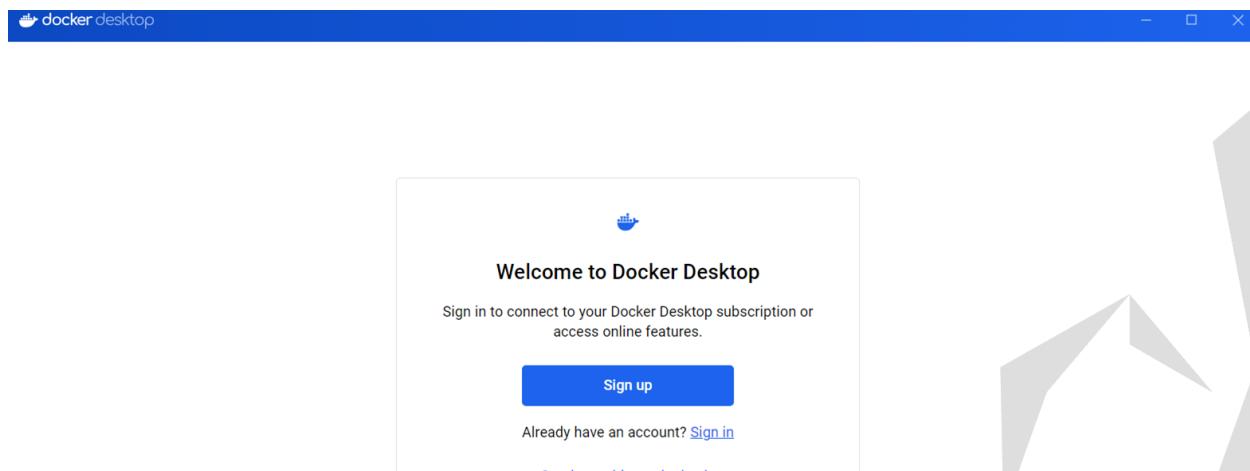
Complete the installation of Docker Desktop.

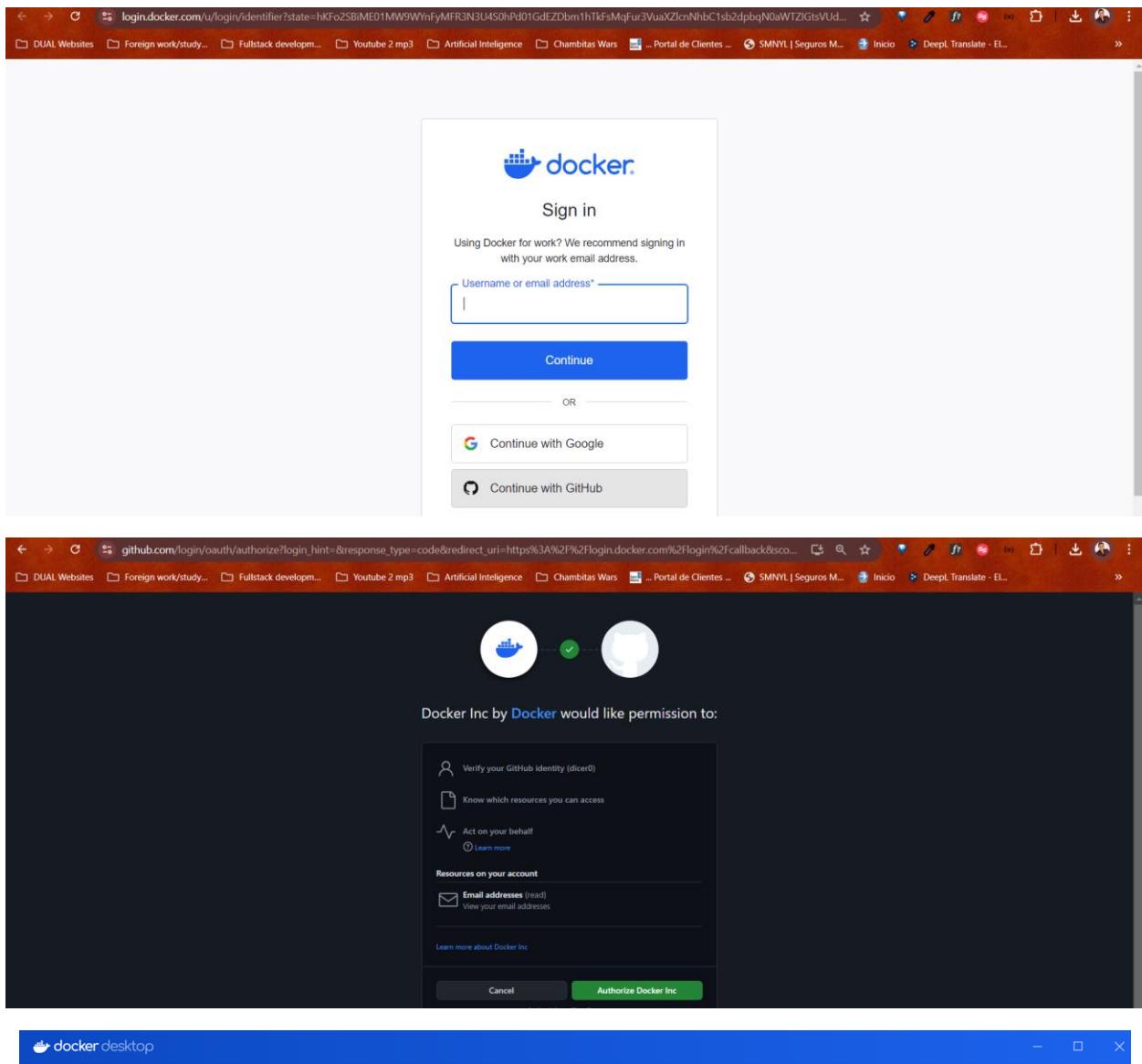
- Use recommended settings (requires administrator password)
Docker Desktop automatically sets the necessary configurations that work for most developers.
- Use advanced settings
You manually set your preferred configurations.

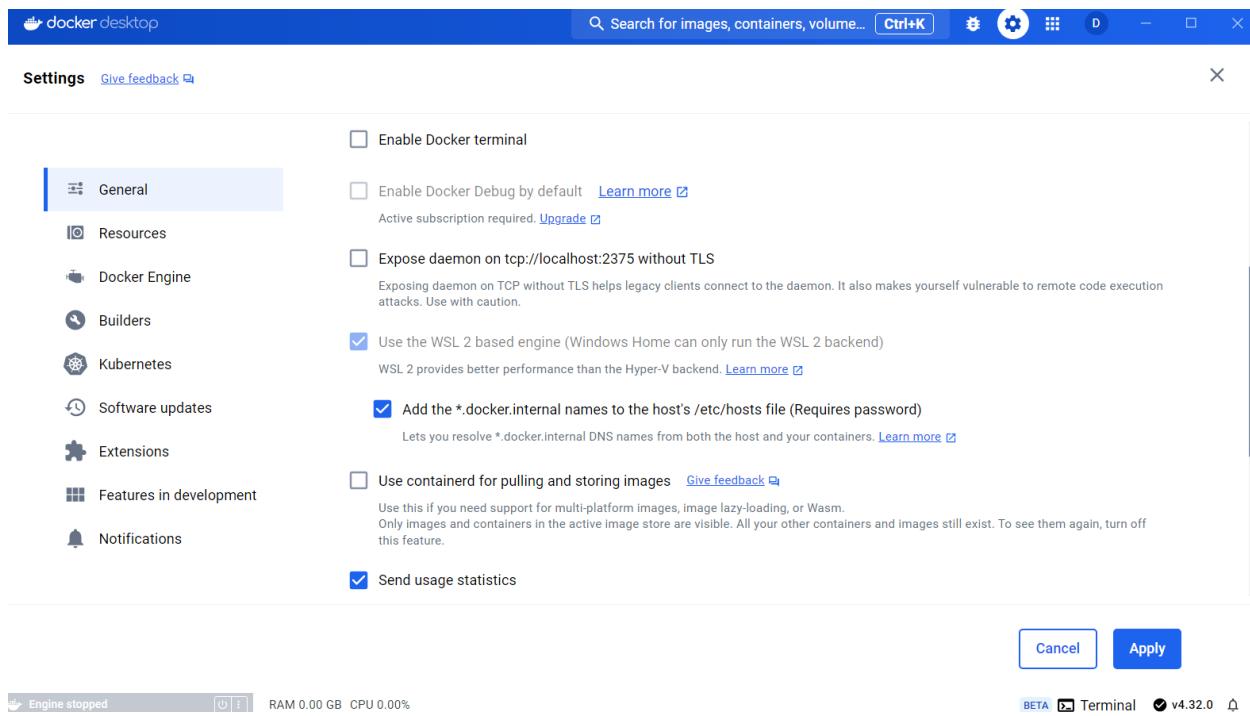
Choose your preferred WSL configuration: [Learn more](#)

- Update to the latest version of WSL 2 from Microsoft (Recommended - requires administrator password)
We run ‘wsl –update’ for you.
- Manually update WSL 2 myself
You'll need to manually update WSL 2 and then relaunch Docker Desktop to complete set up.

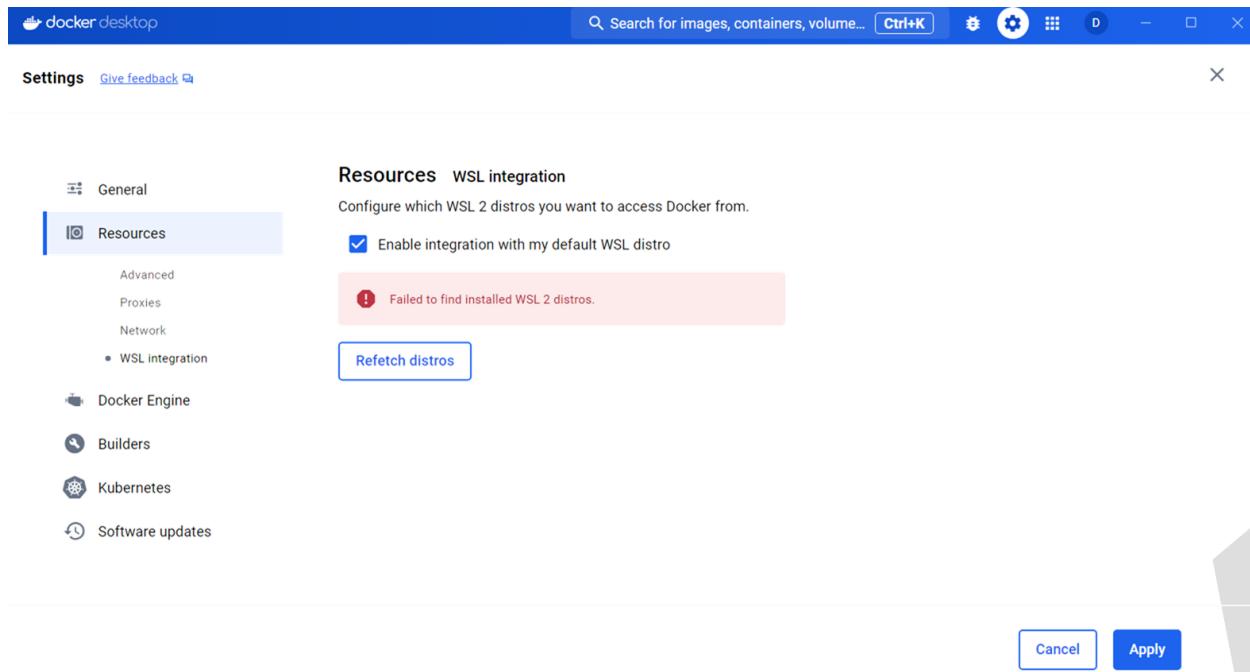
Finish



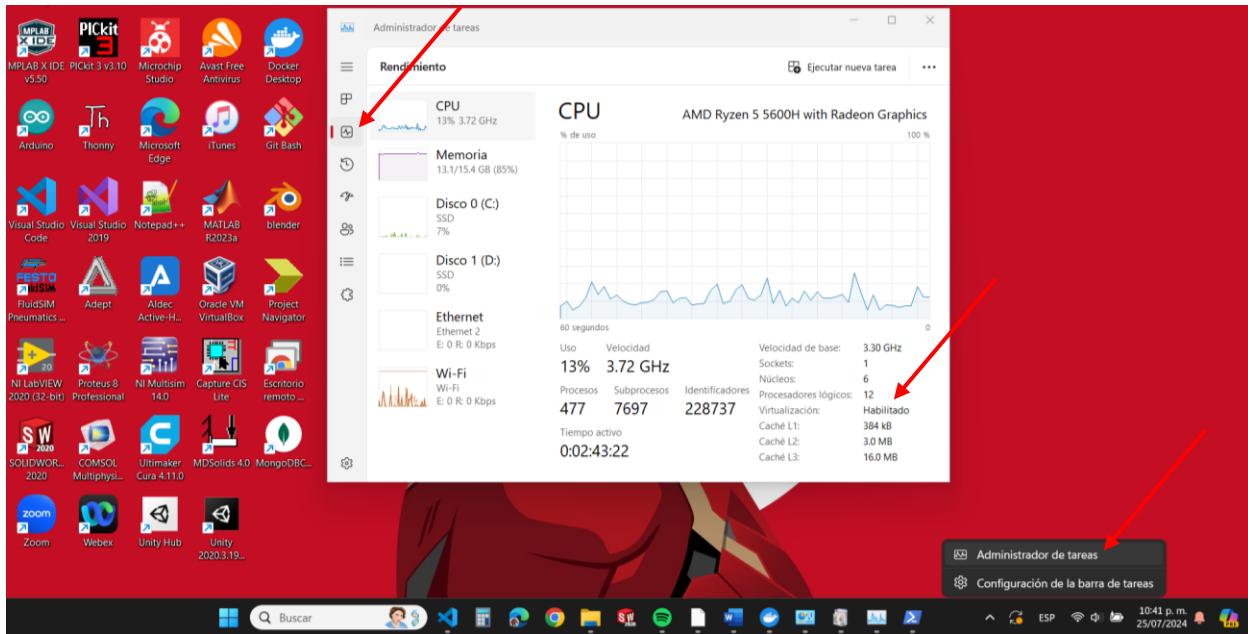




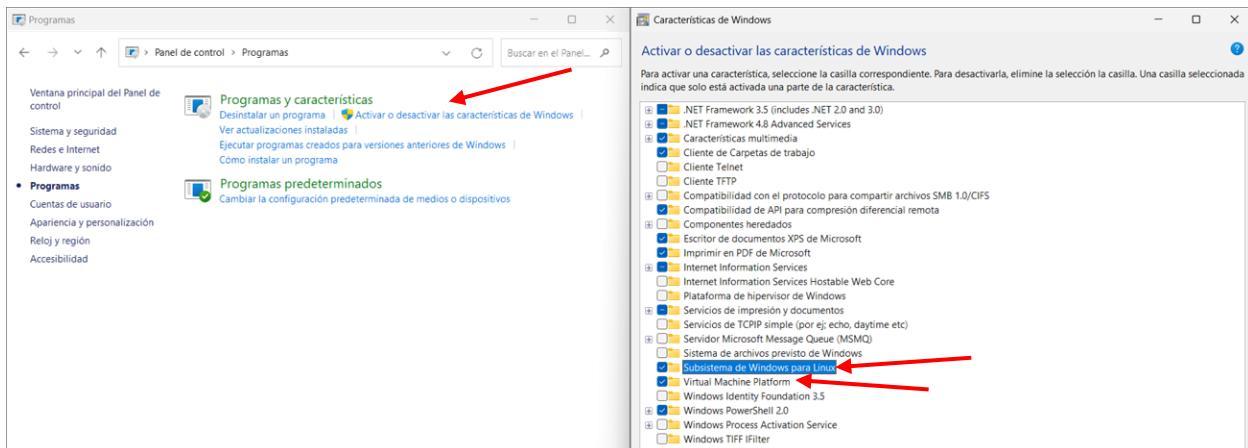
Luego seleccionamos la opción de Resources → WSL Integration → Enable integration with my default WSL distro.



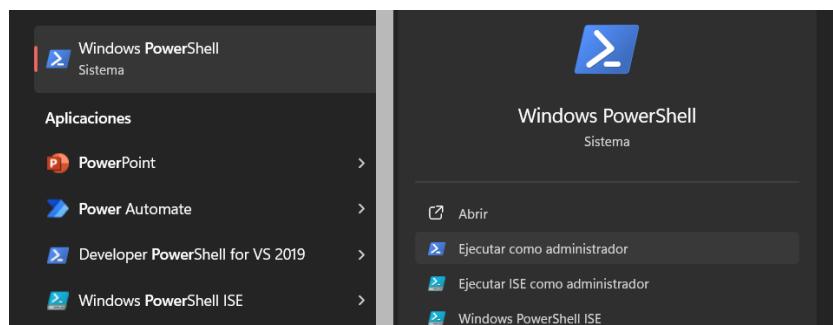
Cuando ya tengamos instalado y configurado Docker Desktop dentro de nuestro ordenador, una de las cosas que debemos tener en cuenta en la instalación con Windows es que el sistema debe tener instalada la versión Windows 10 de 64 Bits o superior, se debe tener habilitada la virtualización que se observa al seleccionar la opción de: Barra de Tareas → Clic Derecho → Administrador de Tareas → Rendimiento → Virtualización: Habilitado.



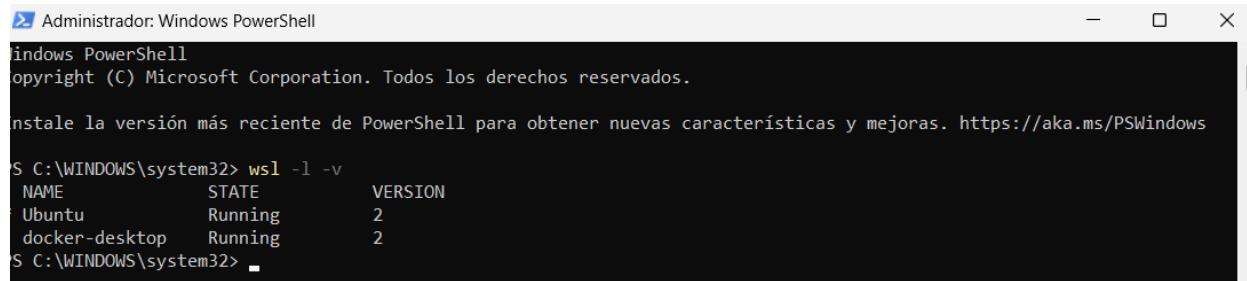
Luego debemos asegurarnos de que estén habilitadas las características de Windows de Subsistema para Linux y Virtual Machine Platform al seleccionar la opción de: Panel de Control → Activar o Desactivar las Características de Windows → Seleccionar las checkbox de Subsistema para Linux y Virtual Machine Platform.



Y finalmente introduciéndonos a la consola Windows PowerShell en modo administrador para ejecutar los siguientes comandos: wsl –install y wsl -l -v.



- `wsl --install`: Comando que sirve para instalar el WSL (Windows Subsystem for Linux).
- `wsl -l -v`: Comando que comprueba la versión del WSL, debe ser la versión 2.



```

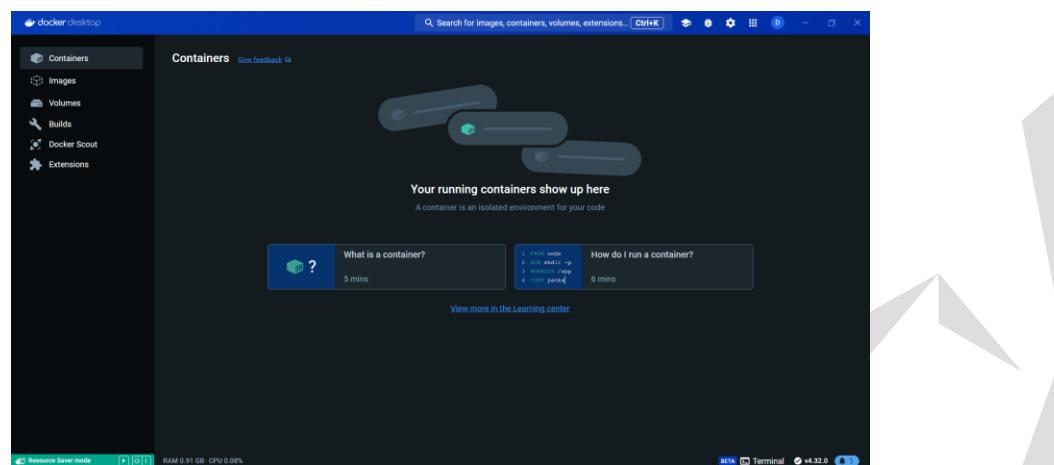
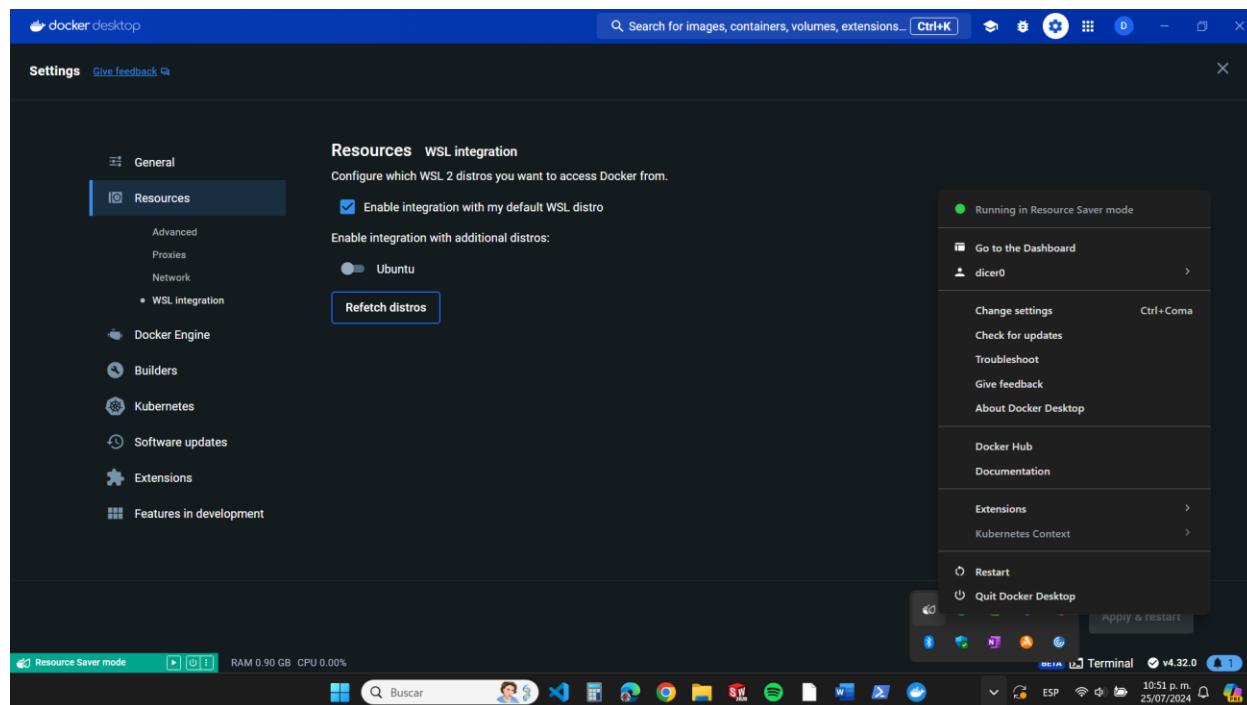
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

S: C:\WINDOWS\system32> wsl -l -v
NAME      STATE    VERSION
Ubuntu    Running   2
docker-desktop    Running   2
S: C:\WINDOWS\system32>

```

Al aplicar estos cambios, podremos visualizar que se ha activado el WSL de Ubuntu dentro del Docker Desktop y estará corriendo sin problemas, además aparecerá un servicio en la esquina inferior derecha donde se mostrará de igual forma que éste se encuentra corriendo.

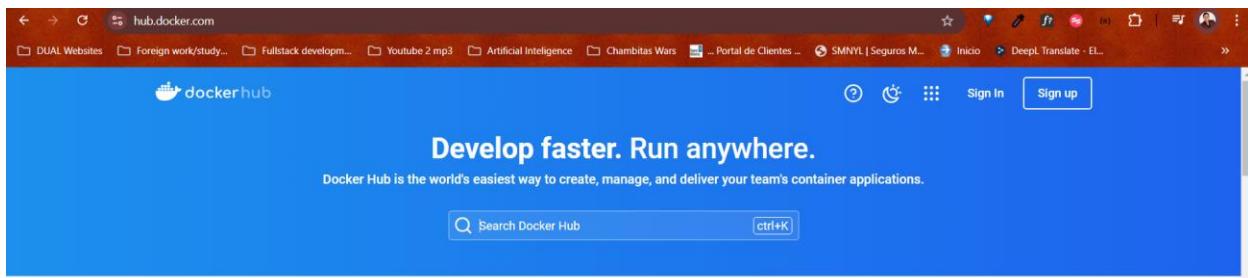


Ejemplo de Imagen: MongoDB en Docker

¿Por qué correr **MongoDB** en **Docker** y como se realiza esta ejecución de forma local? **Docker** es un sistema que nos permite correr cualquier aplicación (web, de escritorio, **bases de datos**, machine learning y data science, IoT, sistemas embebidos, etc.) dentro de un entorno ejecutable sin necesidad de tener problemas con instalaciones, conexiones, etc. Simplemente con levantar y ejecutar el contenedor, este contendrá todos los elementos necesarios para conectarnos y consultar cualquier **database**. Para ello se debe crear un nuevo archivo que se llame **docker-compose.yml**, el cual contendrá las características de la **imagen** utilizada en el proyecto (plantilla que contiene todo lo necesario para ejecutar una aplicación específica, incluyendo el sistema operativo base, código, librerías y configuraciones).

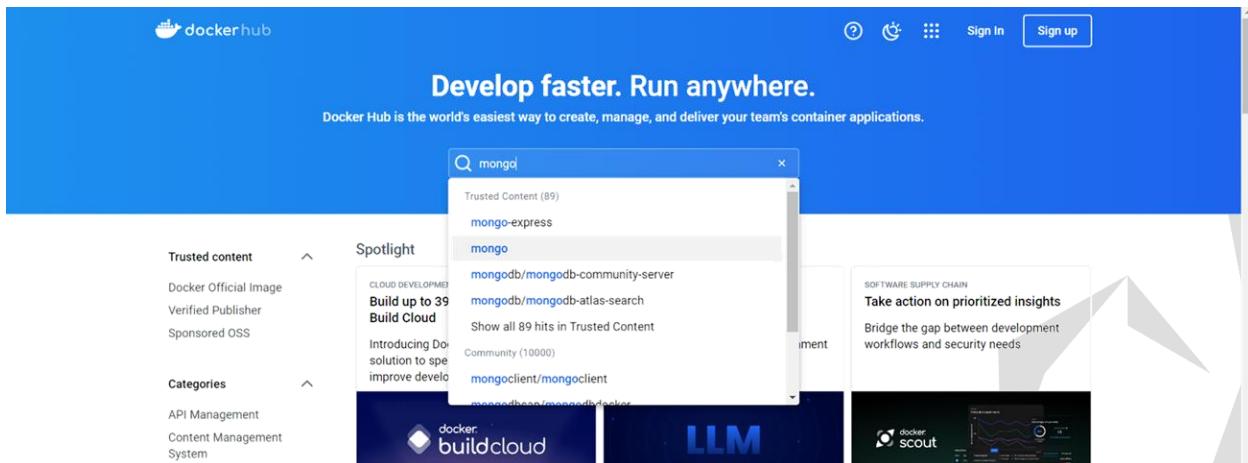
Las características añadidas en el archivo **docker-compose.yml** dependen del tipo de aplicación que se esté utilizando en el proyecto, y estas pueden ser consultadas en el siguiente enlace, donde se incluye información acerca de las características necesarias para cada tipo de **imagen** distinta:

<https://hub.docker.com/>



The screenshot shows the Docker Hub homepage with a blue header. The main heading is "Develop faster. Run anywhere." Below it, a subtext reads "Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications." A search bar with the placeholder "Search Docker Hub" and a "ctrl+K" keyboard shortcut is visible. On the left, there are navigation sections for "Trusted content" (Docker Official Image, Verified Publisher, Sponsored OSS) and "Categories" (API Management, Content Management System). In the center, there's a "Spotlight" section with three cards: "Build up to 39x faster with Docker Build Cloud", "LLM Everywhere: Docker and Hugging Face", and "Take action on prioritized insights".

Para trabajar con **MongoDB** en **Docker**, se cuenta con la siguiente documentación de **imagen**:



The screenshot shows the Docker Hub homepage with a search bar containing "mongo". A dropdown menu lists several Docker images related to MongoDB, including "mongo-express", "mongo", "mongodb/mongodb-community-server", "mongodb/mongodb-atlas-search", and "mongoclient/mongoclient". The rest of the page layout is similar to the previous screenshot, featuring the "Develop faster. Run anywhere." header, the "Spotlight" section, and the "Trusted content" and "Categories" sidebar.

Docker Hub search results for "mongo".

Filters:

- Products: Images, Extensions, Plugins
- Trusted Content: Docker Official Image, Verified Publisher, Sponsored OSS
- Categories: API Management

Search Results:

- mongo-express**: Updated 5 days ago, 100M+, 1.5K pulls. Description: Web-based MongoDB admin interface, written with Node.js and express.
- mongo**: Updated 13 days ago, 1B+, 10K+ pulls. Description: MongoDB document databases provide high availability and easy scalability.

Official Docker Image Details:

mongo Docker Official Image · 1B+ · 10K+
MongoDB document databases provide high availability and easy scalability.

Actions: docker pull mongo | Copy

Quick Reference:

- Maintained by: the Docker Community
- Where to get help: Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow

Recent Tags:

- 8.0.0-rc13-nanoserver-ltsv2022
- 8.0.0-rc13-nanoserver-1809 · 8.0.0-rc13-nanoserver
- 8.0.0-rc13-jammy · 8.0.0-rc13
- 8.0.0-rc-nanoserver-ltsv2022 · 8.0.0-rc-nanoserver-1809
- 8.0.0-rc-nanoserver · 8.0.0-rc-jammy · 8.0.0-rc

En el caso específico de **MongoDB con Docker**, se debe crear una carpeta local que se puede llamar como nosotros queramos. Esta es importante porque en ella se guardarán todos los datos de la **base de datos no relacional**, que dentro del contenedor se almacenan en un directorio llamado **/data/db** (lo cual se menciona en la documentación de **Docker Hub**). Y para que lo que se almacene en dicha carpeta,

se almacene de igual forma en nuestro folder local, se debe crear una igualdad entre ambas en la parte de **volumes** dentro de las características del archivo **docker-compose.yml**, guardando así el estado (los datos) del contenedor que maneja la **DB** de **MongoDB**.

Esta carpeta debe estar indicada dentro del archivo **.gitignore**, ya que contiene datos sensibles, como contraseñas.

Where to Store Data

Important note: There are several ways to store data used by applications that run in Docker containers. We encourage users of the `mongo` images to familiarize themselves with the options available, including:

- Let Docker manage the storage of your database data [by writing the database files to disk on the host system using its own internal volume management](#). This is the default and is easy and fairly transparent to the user. The downside is that the files may be hard to locate for tools and applications that run directly on the host system, i.e. outside containers.
- Create a data directory on the host system (outside the container) and [mount this to a directory visible from inside the container](#). This places the database files in a known location on the host system, and makes it easy for tools and applications on the host system to access the files. The downside is that the user needs to make sure that the directory exists, and that e.g. directory permissions and other security mechanisms on the host system are set up correctly.

WARNING (Windows & OS X): When running the Linux-based MongoDB images on Windows and OS X, the file systems used to share between the host system and the Docker container is not compatible with the memory mapped files used by MongoDB ([docs.mongodb.org](#) and related [jira.mongodb.org](#) bug). This means that it is not possible to run a MongoDB container with the data directory mapped to the host. To persist data between container restarts, we recommend using a local named volume instead (see `docker volume create`). Alternatively you can use the Windows-based images on Windows.

The Docker documentation is a good starting point for understanding the different storage options and variations, and there are multiple blogs and forum postings that discuss and give advice in this area. We will simply show the basic procedure here for the latter option above:

1. Create a data directory on a suitable volume on your host system, e.g. `/my/own/datadir`.
2. Start your `mongo` container like this:

```
$ docker run --name some-mongo -v /my/own/datadir:/data/db -d mongo Copy
```

Referencias

Platzi, Israel Vázquez, “Curso de Fundamentos de Bases de Datos”, 2018 [Online], Available: <https://platzi.com/new-home/clases/1566-bd/19781-bienvenida-conceptos-basicos-y-contexto-historico/>

Platzi, Nicolás Molina, “Curso de Introducción a MongoDB”, 2023 [Online], Available: <https://platzi.com/cursos/mongodb/>

