

INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

PROGRAMACIÓN: DESARROLLO BACKEND

SQL

Bases de Datos No Relacionales
Basadas en Documentos: MongoDB

Contenido

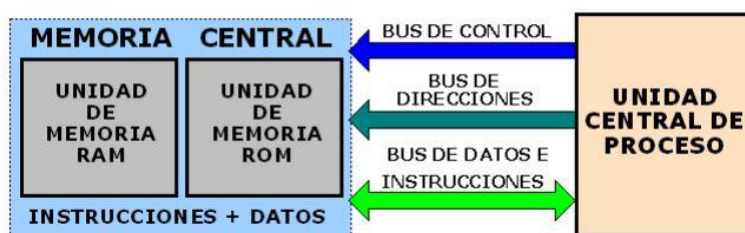
Introducción a las Bases de Datos	2
Tipos de Bases de Datos.....	2
Representación de las Bases de Datos: Nomenclatura de Chen	3
Bases de Datos No Relacionales	3
Jerarquía de Datos de las Bases de Datos No Relacionales basadas en Documentos.....	5
Tipos de Escalamiento: Horizontal y Vertical	7
Réplicas o Sharding (Cluster)	8
Instalación de MongoDB Cloud - Mongo Atlas	8
Referencias	15



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

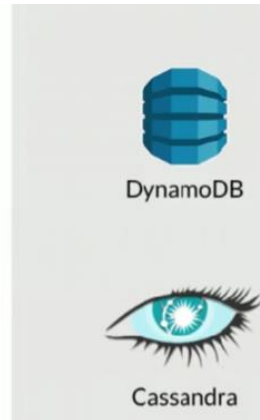
Como ya se había mencionado previamente, las **bases de datos no relacionales** o **NRDB (Non Relational Data Bases)** no se conforman de un solo tipo de **bases de datos**, sino de varias, y aunque puedan ser muy distintas unas de otras, todas se engloban dentro de la misma categoría de **base de datos 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

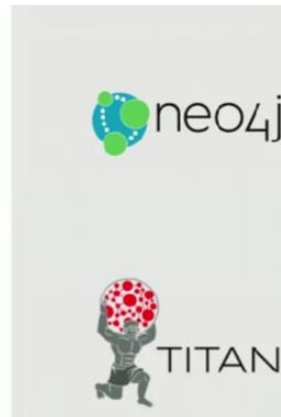
Son ideales para almacenar y extraer datos con una clave única. Manejan los diccionarios de manera excepcional.



- **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

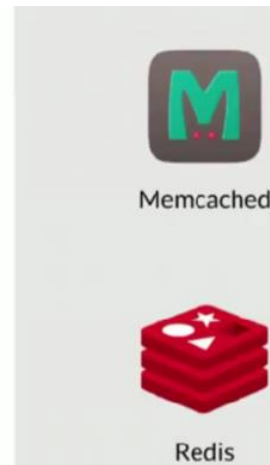
Basadas en teoría de grafos sirven para entidades que se encuentran interconectadas por múltiples relaciones. Ideales para almacenar relaciones complejas.



- **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.



BigQuery



Elasticsearch

- **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.



MongoDB



Firestore

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**.



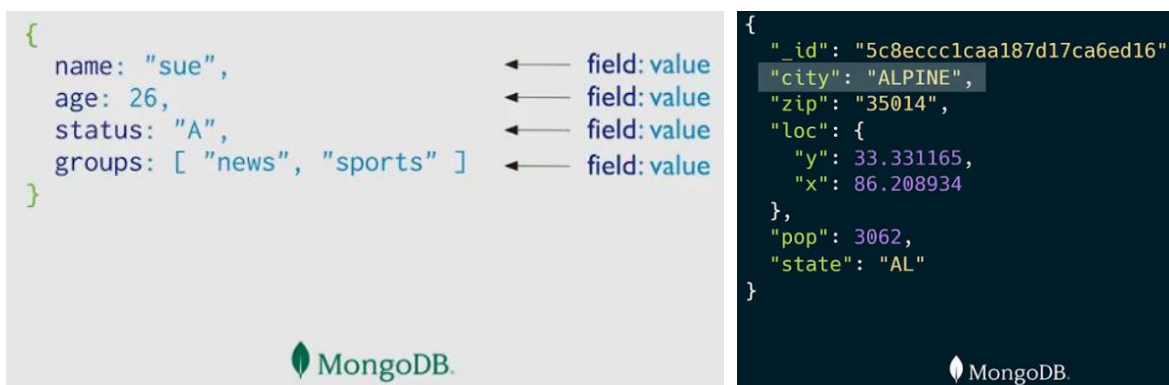
Cuando trabajamos con bases de datos basadas en documentos como **MongoDB** o **Firestore**, cambiaremos el concepto de las **tablas** por las **colecciones** y las **tuplas (filas)** por los **documentos**.

- **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.



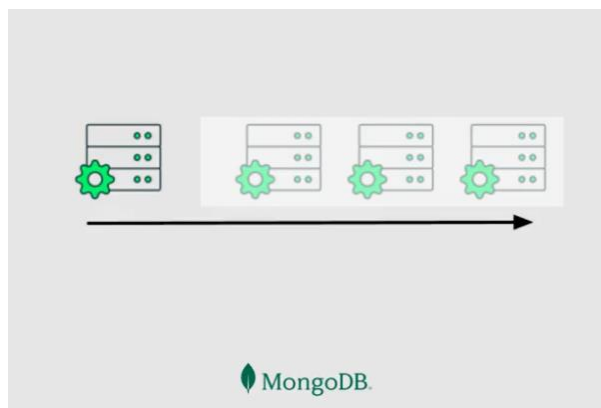
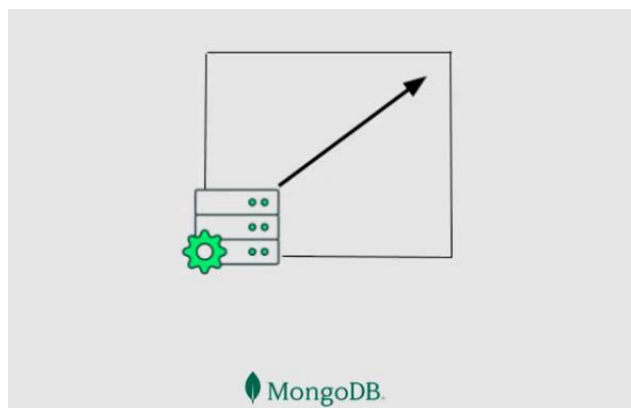
Una consideración clave al diseñar la **database no relacional** es anticipar cómo se **extraerán los datos**. En el contexto de una aplicación, es útil pensar en términos de las vistas que se mostrarán en un momento específico. En otras palabras, al estructurar la **DB**, **debemos asegurarnos de que 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.

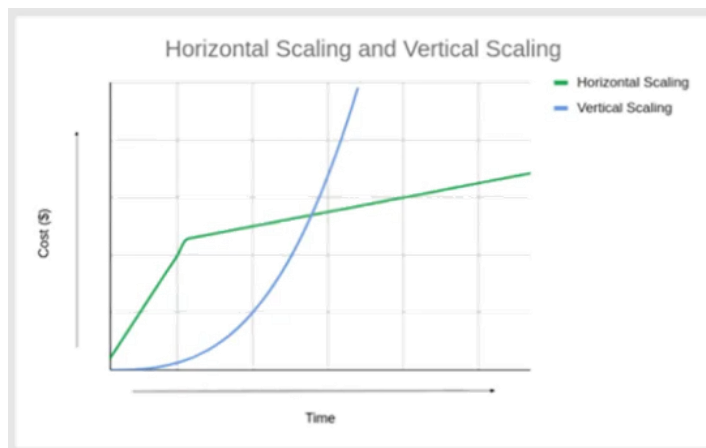
Tipos de Escalamiento: Horizontal y Vertical

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)

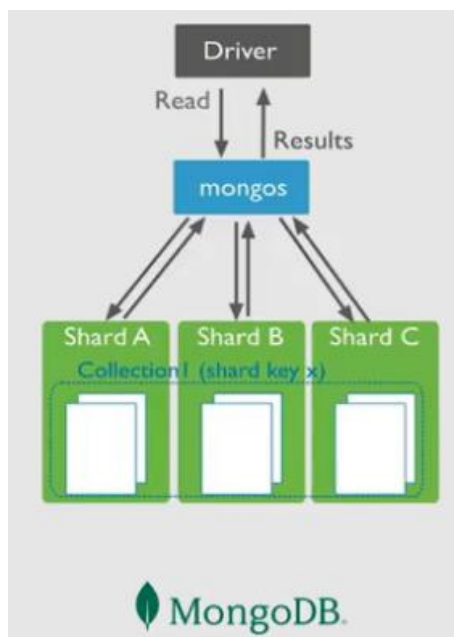
Réplica o Shard se refiere al proceso de **copiar y mantener actualizada una copia de la base de datos en un servidor diferente**.

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 una **database principal**, 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**.

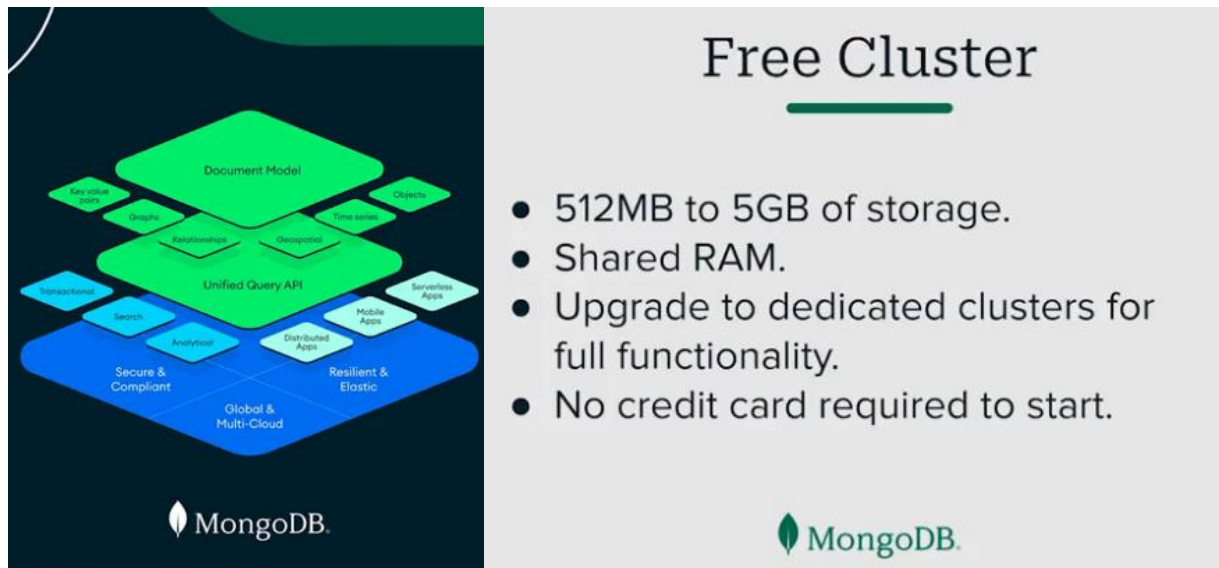
Un **cluster** en **MongoDB** es un **conjunto de servidores que trabajan juntos** para proporcionar alta disponibilidad, escalabilidad y **redundancia (duplicación para crear una copia de seguridad de los datos)** dirigidos para soportar las necesidades de una **base de datos no relacional**.



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**, además este **como parte de su servicio incluye un sistema de replicación y**

clusterización de datos, permitiendo así la creación de un modelo de escalamiento y replicación funcional y 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**.



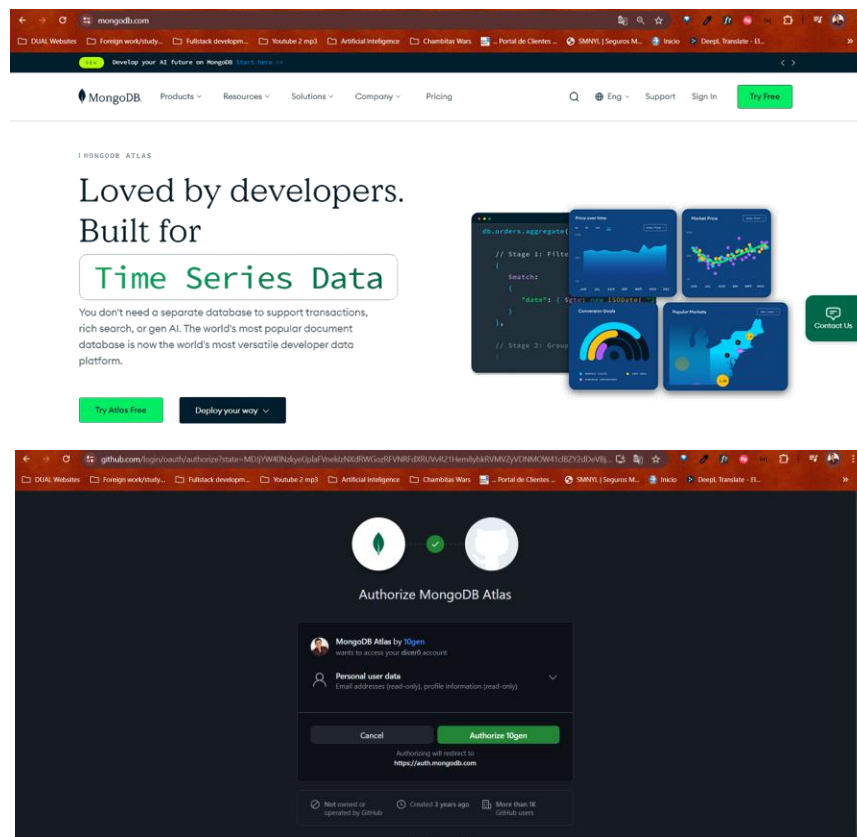
The image shows a diagram of the MongoDB architecture on the left and a list of features for the 'Free Cluster' on the right. The diagram illustrates a multi-tier architecture with a 'Document Model' at the top, followed by a 'Unified Query API' layer, and a 'Global & Multi-Cloud' layer at the bottom. Various application types like 'Key-value pairs', 'Graphs', 'Time series', 'Objects', 'Relationships', 'Geospatial', 'Transactional', 'Search', 'Analytical', 'Distributed Apps', 'Mobile Apps', and 'Serverless Apps' are shown interacting with the API. The 'Free Cluster' features are listed in a bulleted format.

Free Cluster

- 512MB to 5GB of storage.
- Shared RAM.
- Upgrade to dedicated clusters for full functionality.
- No credit card required to start.

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 image shows two screenshots. The top screenshot is the MongoDB Atlas website, which features the headline 'Loved by developers. Built for Time Series Data' and a 'Try Atlas Free' button. The bottom screenshot is the GitHub authorization page for MongoDB Atlas, showing a user named 'MongoDB Atlas by 10gen' and an 'Authorize 10gen' button.

The screenshot shows a web browser window at the URL cloud.mongodb.com. The address bar contains several tabs: DUAL Websites, Foreign work/study..., Fullstack develop..., Youtube 2 mp3, Artificial Intelligence, Chambito Wars, Portal de Clientes ..., SMNYL | Seguros M..., Inicio, and DeepL Translate - EL... The main content area features a green illustration of a computer monitor with headphones, a stack of servers, a lightbulb, and clouds. Below the illustration, the text reads: "Welcome!" followed by "Use your account to deploy a cloud database with MongoDB Atlas and contact Support." A small MongoDB logo is visible in the bottom right corner.

cloud.mongodb.com/v2/66697f54bee433731e93cb45#/setup/personalization

DUAL Websites Foreign work/study... Fullstack develop... Youtube 2 mp3 Artificial Intelligence Chamtibas Wars ... Portal de Clientes ... SMNYL | Seguros M... Inicio DeepL, Translate - EL...

GETTING TO KNOW YOU

What is your primary goal?

Learn MongoDB

How long have you been developing software with MongoDB?

I've never developed software with MongoDB before

GETTING TO KNOW YOUR PROJECT

What programming language are you primarily building on MongoDB with?

Python

What type(s) of data will your project use?

You can choose as many as you want

Not sure... X

Will your application include any of the following architectural models?

You can choose as many as you want

Not sure... X

Finish

En esta parte es donde elegiremos el plan de almacenamiento, que en este caso será el 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	Price	Storage	RAM	vCPU
<input type="radio"/> M10	\$0.08/hour	10 GB	2 GB	2 vCPUs
<input type="radio"/> Serverless		Up to 1 TB	Auto-scale	Auto-scale
<input checked="" type="radio"/> M0	Free	512 MB	Shared	Shared

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

Name

You cannot change the name once the cluster is created.

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

You cannot change the name once the cluster is created.

1-ExampleMongoDB

☒ Automate security setup

☒ Preload sample dataset

Provider

☒ AWS ☐ Google Cloud ☐ Azure

Region

☒ N. Virginia (us-east-1) ☐ Mumbai (ap-south-1) ☐ Singapore (ap-southeast-1)

North America

☐ N. Virginia (us-east-1) ☐ Oregon (us-west-2)

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

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

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_cer0

Password

Copy

Create User

Add a connection IP address

Your current IP address (138.186.31.214) 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 aquí 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.

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_cer0 (SCRAM)

You can edit, delete, or add users in [Database Access](#).

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

Driver: Python

Version: 3.12 or later

☐ I have installed my driver

Create a database user

The first user will have [atlasAdmin](#) permissions for this project. You'll need your database user's credentials to connect to your cluster. You can manage database users in [Database Access](#).

Add a connection IP address

Your current IP address (138.186.31.214) has been added to enable local connectivity. You can edit, delete, or add IP addresses in [Network Access](#).

Customized instructions based on your inputs:

1. Install your driver

Run the following on the command line

Note: Use appropriate Python 3 executable

```
python -m pip install "pymongo[srv]"
```

[View MongoDB Python Driver installation instructions.](#)

2. Add connection string into your application code

Close

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_cer0 (SCRAM)

You can edit, delete, or add users in [Database Access](#).

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

Driver: Python

Version: 3.12 or later

☐ I have installed my driver

String

Sample Code

Show Password

```
from pymongo.mongo_client import MongoClient
from pymongo.server_api import ServerApi

uri = "mongodb+srv://di_cer0:<password>@1-examplemongodb.4bm1ts."

# Create a new client and connect to the server
client = MongoClient(uri, server_api=ServerApi('1'))

# Send a ping to confirm a successful connection
try:
    client.admin.command('ping')
    print("Pinged your deployment. You successfully connected to MongoDB")
except Exception as e:
    print(e)
```

Replace [password](#) with the password for the [di_cer0](#) user. Ensure any option params are URL encoded.

Resources

[Get started with the Python Driver](#)

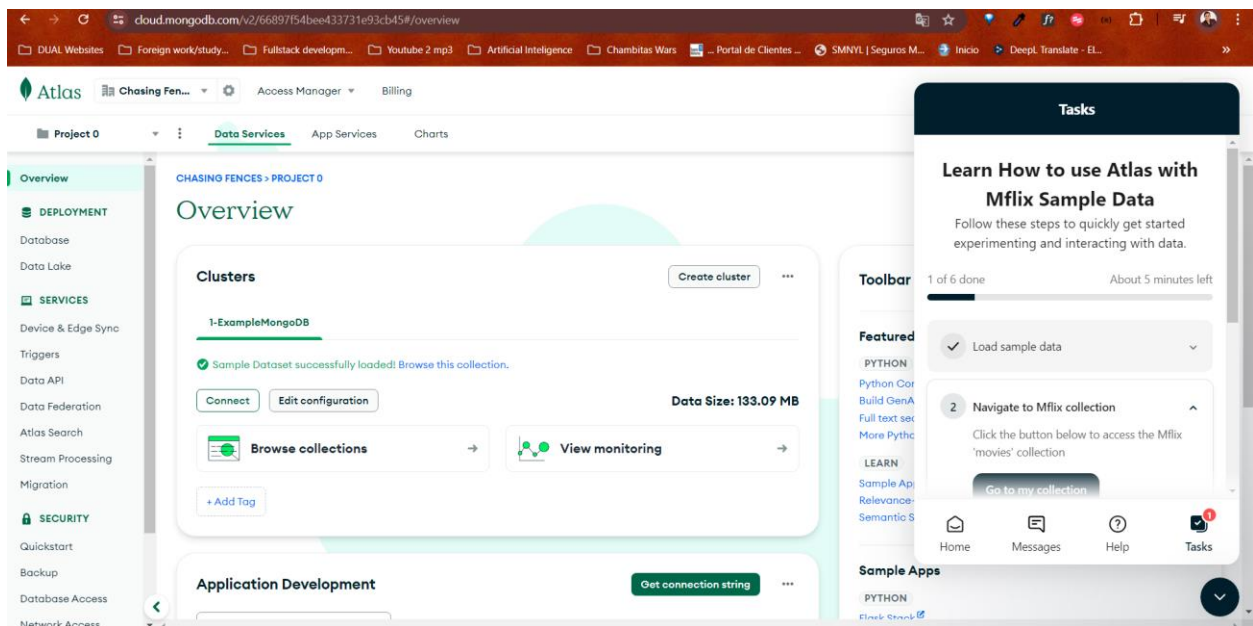
[Python Starter Sample App](#)

[Access your Database Users](#)

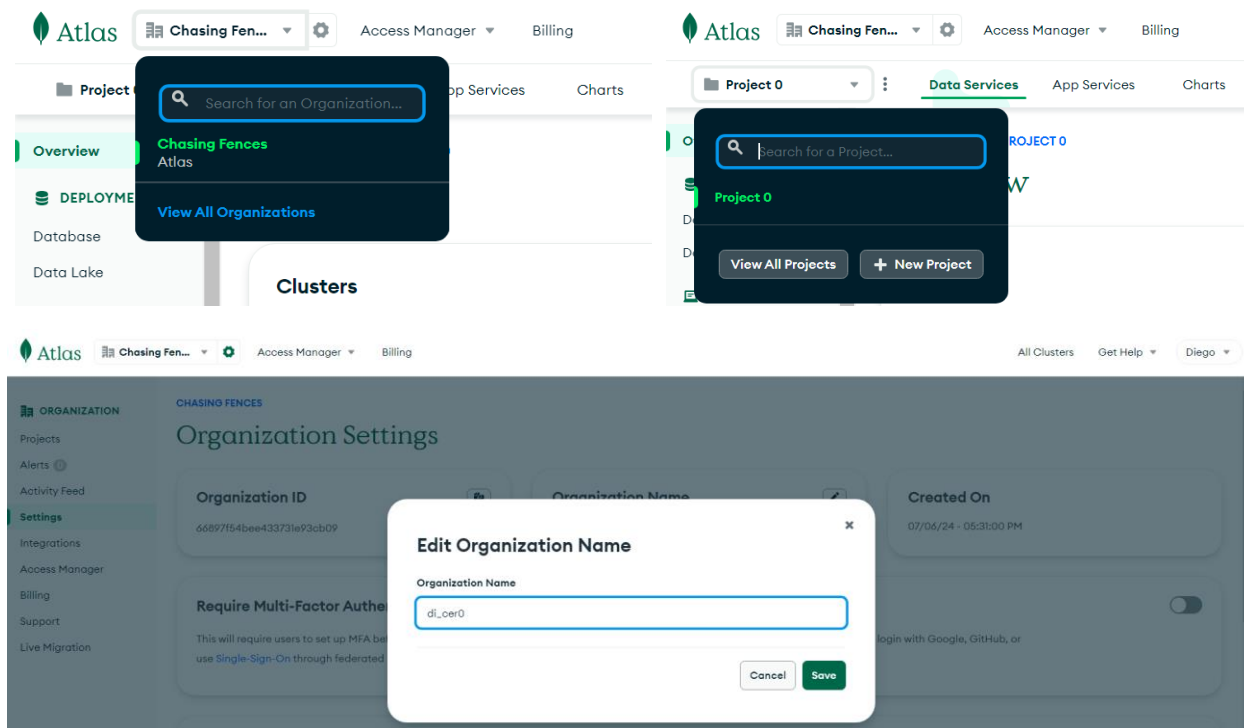
Close

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

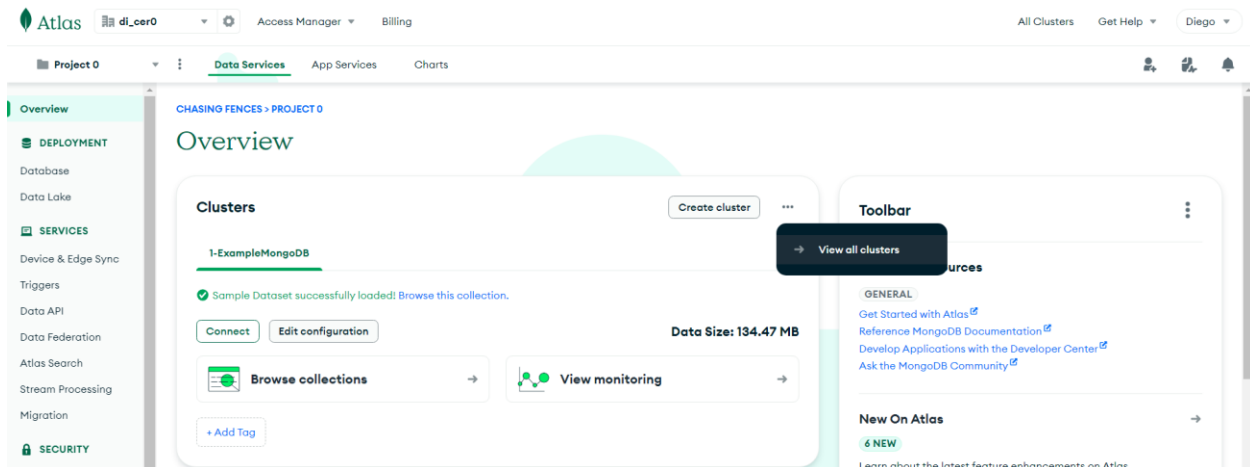
Ahora se me indica que puedo cargar datos a la **database**, realizarles Queries, etc.



También podremos 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.

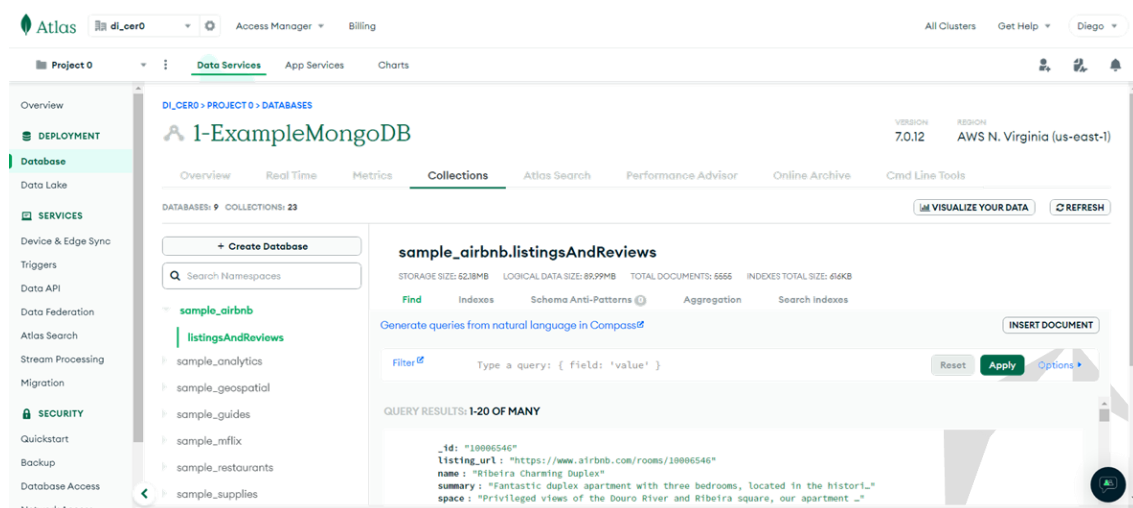
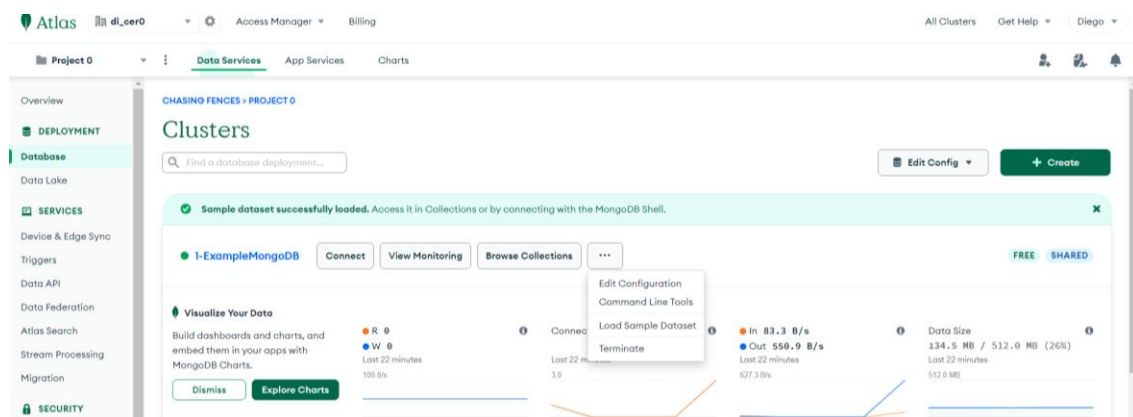


Ahora dentro de los clusters, daremos clic derecho en los 3 puntos de la esquina superior derecha y seleccionaremos la opción de View all clusters o directo en DEPLOYMENT → Database.



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 se encuentra en el siguiente enlace:

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



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-/](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/>

