

Benemérita Universidad Autónoma de Puebla

- MODELOS DE DESARROLLO WEB
- NRC: 12051
- PROYECTO FINAL
- PROF. ALFREDO GARCÍA SUÁREZ
- ALUMNO: VICENTE ZAVALETAA
SANCHEZ



Thunder Client

GET http://127.0.0.1:8000/laptops/mostrar/ Send

Query Headers ² Auth Body ¹ Tests Pre Run

Query Parameters

parameter value

Status: 202 Accepted Size: 1.34 KB Time: 104 ms

Response Headers ⁵ Cookies Results Docs

```
1 [  
2 {  
3   "id": "655e44f8ac4decea75f20981",  
4   "CPU": "Intel1",  
5   "RAM": "16GB",  
6   "Almacenamiento": "512GB",  
7   "Marca": "ACER",  
8   "SO": "Linux"  
9 },  
10 {  
11   "id": "655e44f8ac4decea75f20981",  
12   "CPU": "Intel1",  
13   "RAM": "16GB",  
14   "Almacenamiento": "512GB",  
15   "Marca": "ACER",  
16   "SO": "Linux"  
17 }]
```

GET http://127.0.0.1:8000/pasajeros/mostrar/ Send

Query Headers ² Auth Body ¹ Tests Pre Run

Query Parameters

parameter value

Status: 202 Accepted Size: 3.3 KB Time: 102 ms

Response Headers ⁵ Cookies Results Docs

```
1 [  
2 {  
3   "id": "654ecb0a13730345704d0f2b",  
4   "username": "FREDDY",  
5   "full_name": "Freddy Garcia",  
6   "email": "alfredo.garcias@alumno.buap.mx",  
7   "phone": 2224237171,  
8   "dni": "FREDDY",  
9   "password": "$2a$12$Px4  
10   /G9Onxs4m6QxjAwsbt0mqf4BFxkLUvn3F5PFPbWmhWLYEyGObG",  
11   "disabled": "False"  
12 },  
13 {  
14   "id": "654ecb0a13730345704d0f2c",  
15   "username": "YOSS",  
16 }
```

GET http://127.0.0.1:8000/usuarios/mostrar/ Send

Query Headers ² Auth Body ¹ Tests Pre Run

Query Parameters

parameter value

Status: 202 Accepted Size: 378 Bytes Time: 100 ms

Response Headers ⁵ Cookies Results Docs

```
1 [  
2 {  
3   "id": "65604388d2ff39af4877088b",  
4   "name": "Ejemplo1234",  
5   "Pclass": 1,  
6   "Survived": 1,  
7   "Sex": "Femenil12no",  
8   "Age": 25,  
9   "SibSp": 1,  
10  "Parch": 0,  
11  "Ticket": "12123345",  
12  "Fare": "50.00",  
13  "Cabin": "C21233",  
14  "Embarked": "S"
```

The screenshot shows a UI testing application interface with two requests made to the endpoint `http://127.0.0.1:8000/laptops/agregar/`.

Request 1 (Bad Request):

- Method: POST
- URL: `http://127.0.0.1:8000/laptops/agregar/`
- Status: 400 Bad Request
- Size: 48 Bytes
- Time: 101 ms
- Response:

```
1 {
2     "detail": "Ya existe un objeto con el mismo ID"
3 }
```

Request 2 (Success):

- Method: POST
- URL: `http://127.0.0.1:8000/laptops/agregar/`
- Status: 200 OK
- Size: 170 Bytes
- Time: 209 ms
- Response:

```
1 {
2     "mensaje": "La Laptop se agrego exitosamente",
3     "Laptop": {
4         "_id": "6560e8ea5cd7679d9b6a057b",
5         "CPU": "Intel",
6         "RAM": "16GB",
7         "Almacenamiento": "512GB",
8         "Marca": "ACER",
9         "SO": "Linux"
10    }
11 }
```

The screenshot shows a UI testing application interface with two separate request windows.

Request 1 (Top Window):

- Method:** POST
- URL:** http://127.0.0.1:8000/pasajeros/agregar/
- Status:** 400 Bad Request
- Size:** 48 Bytes
- Time:** 103 ms
- Response:**

```
1 {
2     "detail": "Ya existe un objeto con el mismo ID"
3 }
```

Request 2 (Bottom Window):

- Method:** POST
- URL:** http://127.0.0.1:8000/pasajeros/agregar/
- Status:** 201 Created
- Size:** 255 Bytes
- Time:** 204 ms
- Response:**

```
1 {
2     "mensaje": "El Pasajero se agrego exitosamente",
3     "Pasajero": {
4         "_id": "6560e9075cd7679d9b6a057d",
5         "name": "Ejemplo1234",
6         "Pclass": 1,
7         "Survived": 1,
8         "Sex": "Femeni112no",
9         "Age": 25,
10        "SibSp": 1,
11        "Parch": 0,
12        "Ticket": "12123345",
13        "Fare": "50.00",
14        "Cabin": "C21233",
15        "Embarked": "S"
16    }
17 }
```

The left sidebar contains various icons for file operations, search, and other tools. The bottom navigation bar includes tabs for 'master*', '0', '△ 0', '0', and '0'.

The screenshot shows a desktop environment with multiple Postman windows open, indicating a multi-instance setup. The main window in the foreground displays a failed POST request to `http://127.0.0.1:8000/usuarios/agregar/`. The response status is **400 Bad Request**, with a size of **45 Bytes** and a time of **102 ms**. The response body contains:

```
1 {
2     "detail": "Ya existe un Usuario con este id"
3 }
```

The JSON Content pane shows the request body sent to the server:

```
1 {
2     "id": "654ecb0a13730345704d0",
3     "username": "FREDDY",
4     "full_name": "Freddy García",
5     "email": "alfredo.garcias@alum.buap.mx",
6     "phone": 2224237171,
7     "dni": "FREDDY",
8     "password": "$2a$12$Px4
/G90nxs4m6QxjAwsbt0mqf4BFxI
hWLYEyGObG",
9     "disabled": "False"
10 }
```

The second instance of Postman in the background shows a successful POST request to the same endpoint, resulting in a **201 Created** response with a size of **296 Bytes** and a time of **433 ms**. The response body contains:

```
1 {
2     "mensaje": "El Usuario se agrego exitosamente",
3     "Usuario": {
4         "_id": "6560e9c85cd7679d9b6a057f",
5         "username": "Kevbc",
6         "full_name": "Kevin Bañuelos",
7         "email": "kevbc@alumno.buap.mx",
8         "phone": 2124124112,
9         "dni": "IMG",
10        "password": "$2b$12$.fx8PucdN.WzOfIQ1vYSPOr6BwYHctKL5f
/EJkdDSSnRANLnabKlq",
11        "disabled": "False"
12    }
13 }
```

The screenshot shows a REST client interface with three separate requests displayed in tabs.

Top Tab (laptops/mostrar/):

- Method: GET
- URL: http://127.0.0.1:8000/laptops/mostrar/
- Status: 202 Accepted
- Size: 1.45 KB
- Time: 102 ms
- Response Headers:
 - Content-Type: application/json
 - Content-Length: 160
 - Date: Mon, 12 Mar 2018 14:45:10 GMT
 - Server: Apache/2.4.10 (Ubuntu)
- Response Body:

```
91     "id": "6560e8ea5cd7679d9b6a057b",
92     "CPU": "Intel",
93     "RAM": "8GB",
94     "Almacenamiento": "256GB",
95     "Marca": "HP",
96     "SO": "Windows"
97   },
98   {
99     "id": "6560e8ea5cd7679d9b6a057b",
100    "CPU": "Intel",
101    "RAM": "16GB",
102    "Almacenamiento": "512GB",
103    "Marca": "ACER",
104    "SO": "Linux"
105 }
```

Middle Tab (pasajeros/mostrar/):

- Method: GET
- URL: http://127.0.0.1:8000/pasajeros/mostrar/
- Status: 202 Accepted
- Size: 573 Bytes
- Time: 105 ms
- Response Headers:
 - Content-Type: application/json
 - Content-Length: 573
 - Date: Mon, 12 Mar 2018 14:45:10 GMT
 - Server: Apache/2.4.10 (Ubuntu)
- Response Body:

```
9     "id": "6560e9075cd7679d9b6a057d",
10    "name": "Ejemplo1234",
11    "Pclass": 1,
12    "Survived": 1,
13    "Sex": "Femeni112no",
14    "Age": 25,
15    "SibSp": 1,
16    "Parch": 0,
17    "Ticket": "12123345",
18    "Fare": "50.00",
19    "Cabin": "C21233",
20    "Embarked": "S"
21  }
22 ]
```

Bottom Tab (usuarios/mostrar/):

- Method: GET
- URL: http://127.0.0.1:8000/usuarios/mostrar/
- Status: 202 Accepted
- Size: 3.54 KB
- Time: 101 ms
- Response Headers:
 - Content-Type: application/json
 - Content-Length: 3540
 - Date: Mon, 12 Mar 2018 14:45:10 GMT
 - Server: Apache/2.4.10 (Ubuntu)
- Response Body:

```
139    "password": "$2b$12$.fx8PucdN
140        .WzOfIQ1vYSP0r6BwYHctKL5f/EJkdDSSnRANLnabKlq",
141    "disabled": "False"
142  },
143  {
144    "id": "6560e9c85cd7679d9b6a057f",
145    "username": "Kevbc",
146    "full_name": "Kevin Bañuelos",
147    "email": "kevbc@alumno.buap.mx",
148    "phone": 2124124112,
149    "dni": "IMG",
150    "password": "$2b$12$.fx8PucdN
151        .WzOfIQ1vYSP0r6BwYHctKL5f/EJkdDSSnRANLnabKlq",
152    "disabled": "False"
153  }
154 ]
```

Analitica-de-Datos

PUT <http://127.0.0.1:8000/laptops/actualizar/6560e8ea5cd7> Send

Status: 200 OK Size: 44 Bytes Time: 110 ms

Query Headers 2 Auth Body 1 Tests Pre Run Response Headers 5 Cookies Results Docs { } 4

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {
2     "message": "Laptop modificado exitosamente"
3 }
```

Archivo Editar Selección Ver ... [Analitica-de-Datos](#) 08 - x

GET <http://127.0.0.1:8000/laptops/mostrar/> Send

Status: 202 Accepted Size: 1.45 KB Time: 101 ms

Query Headers 2 Auth Body Tests Pre Run Response Headers 5 Cookies Results Docs { } 4

Query Parameters

parameter value

```
91     "id": "6560320/2eb4/560/c9e1c5b",
92     "CPU": "Intel",
93     "RAM": "8GB",
94     "Almacenamiento": "256GB",
95     "Marca": "HP",
96     "SO": "Windows"
97 },
98 [
99     "id": "6560e8ea5cd7679d9b6a057b",
100    "CPU": "AMD",
101    "RAM": "8GB",
102    "Almacenamiento": "12GB",
103    "Marca": "ASUS",
104    "SO": "Linux"
105 ]
106 ]
```

master* 0 △ 0 ⚡ 0

Go Live Quokka

Analitica-de-Datos

PUT <http://127.0.0.1:8000/usuarios/actualizar/6560e9c85cd7679d9b6a057f> Send

Status: 200 OK Size: 47 Bytes Time: 417 ms

Query Headers 2 Auth Body 1 Tests Pre Run Response Headers 5 Cookies Results Docs { } ≡

JSON XML Text Form Form-encode GraphQL Binary

1 {
2 "mensaje": "Usuario actualizado correctamente"
3 }

JSON Content

Format

1 {
2 "id": "6560e9c85cd7679d9b6a057f",
3 "username": "KEVINBC",
4 "full_name": "Kevin Bañuelos Camaño",
5 "email": "kevbc@alumno.buap.mx",
6 "phone": 2124124112,
7 "dni": "IMG",
8 "password": "123",
9 "disabled": "False"
10 }

Archivo Editar Selección Ver ... ← → Analitica-de-Datos

GET <http://127.0.0.1:8000/usuarios/mostrar/> Send

Status: 202 Accepted Size: 3.55 KB Time: 106 ms

Query Headers 2 Auth Body Tests Pre Run Response Headers 5 Cookies Results Docs { } ≡

Query Parameters

parameter value

139 "password": "\$2b\$12\$L73vqVqy4Ty8EX0Slzv
140 /Ved9qz5MENooXNnJWpq7nApbcNUejkiHm",
141 "disabled": "False"
142 },
143 {
144 "id": "6560e9c85cd7679d9b6a057f",
145 "username": "KEVINBC",
146 "full_name": "Kevin Bañuelos Camaño",
147 "email": "kevbc@alumno.buap.mx",
148 "phone": 2124124112,
149 "dni": "IMG",
150 "password": "\$2b\$12\$L73vqVqy4Ty8EX0Slzv
151 /Ved9qz5MENooXNnJWpq7nApbcNUejkiHm",
152 "disabled": "False"

parameter value

master* ↻ ⊗ 0 △ 0 ⌂ 0

Go Live Quokka

The screenshot shows a desktop application with multiple instances of a REST client tool, likely Postman or similar, running simultaneously. The interface includes a header bar with tabs like Archivo, Editar, Selección, Ver, and a search bar labeled 'Analitica-de-Datos'. The main area displays two separate requests:

Request 1 (Top Instance):

- Method: DELETE
- URL: <http://127.0.0.1:8000/laptops/eliminar/6560e8ea5cd7679d9b6a057>
- Status: 200 OK
- Size: 44 Bytes
- Time: 106 ms
- Response Body:

```
1 {
2     "mensaje": "Laptop eliminada correctamente"
3 }
```

Request 2 (Bottom Instance):

- Method: GET
- URL: <http://127.0.0.1:8000/laptops/mostrar/>
- Status: 202 Accepted
- Size: 1.34 KB
- Time: 101 ms
- Response Body:

```
83     "id": "655ttccce15caat0/40266/38",
84     "CPU": "Intel Core i5",
85     "RAM": "8GB",
86     "Almacenamiento": "256GB SSD",
87     "Marca": "HP",
88     "SO": "Windows 10"
89 },
90 [
91     "id": "656032b72e6475607c9e1c56",
92     "CPU": "Intel",
93     "RAM": "8GB",
94     "Almacenamiento": "256GB",
95     "Marca": "HP",
96     "SO": "Windows"
97 ]
98 ]
```

The left sidebar contains various icons for file operations, search, filters, and other tools. The bottom navigation bar includes tabs for master*, Go Live, Quokka, and a gear icon.

Analitica-de-Datos

TC New Request TC New Request X TC New Request TC New Request TC New Request TC New Request ...

DELETE http://127.0.0.1:8000/pasajeros/eliminar Send

Status: 200 OK Size: 45 Bytes Time: 104 ms

Query Headers 2 Auth Body 1 Tests Pre Run Response Headers 5 Cookies Results Docs {} =

JSON XML Text Form Form-encode GraphQL B

message: "Pasajero eliminado exitosamente"

JSON Content

```
1 {  
2   "id": "6560e8ea5cd7679d9b6a0",  
3   "name": "EJEMPLO_PRUEBA",  
4   "Pclass": 1,  
5   "Survived": 1,  
6   "Sex": "F",  
7   "Age": 25,  
8   "SibSp": 1,  
9   "Parch": 0,  
10  "Ticket": "1214",  
11  "Fare": "50.00",  
12  "Cabin": "C21233",  
13  "Embarked": "S"  
14 }
```

Format

TC New Request TC New Request TC New Request TC New Request X TC New Request ...

GET http://127.0.0.1:8000/pasajeros/mostrar/ Send

Status: 202 Accepted Size: 378 Bytes Time: 100 ms

Query Headers 2 Auth Body Tests Pre Run Response Headers 5 Cookies Results Docs {} =

parameter value

Query Parameters

```
15 },  
16 {  
17   "id": "65604388d2ff39af4877088d",  
18   "name": "Ejemplo",  
19   "Pclass": 1,  
20   "Survived": 1,  
21   "Sex": "Femenino",  
22   "Age": 25,  
23   "SibSp": 1,  
24   "Parch": 0,  
25   "Ticket": "12345",  
26   "Fare": "50.00",  
27   "Cabin": "C23",  
28   "Embarked": "S"  
29 }  
30 ]
```

master* ↻ ⊗ 0 △ 0 ⌂ 0

Go Live Quokka

Analitica-de-Datos

TC 127.0.0.1:8000/login/ X TC 127.0.0.1:8000/users/me/

POST http://127.0.0.1:8000/login/ Send

Query Headers 2 Auth Body 1 Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

Form Fields

username FREDDY

password 1234

field name value

Status: 200 OK Size: 166 Bytes Time: 326 ms

Response Headers 5 Cookies Results Docs

```
1 {
2   "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
3 .eyJzdWIiOiJGUkVERFkiLCJleHAiOjE3MDA4NTEyMzR9
4 .ArZ_QzhdEUPjWncQ6RNjx8QaJswS4DRuAQ0aXlaIJyI",
5   "token_type": "bearer"
6 }
```

Analitica-de-Datos

TC 127.0.0.1:8000/login/ X TC 127.0.0.1:8000/users/me/

GET http://127.0.0.1:8000/users/me/ Send

Query Headers 2 Auth 1 Body Tests Pre Run

None Basic Bearer OAuth 2 NTLM AWS

Bearer Token

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJGUkVERFkiLCJleHAiOjE3MDA4NTEyMzR9.ArZ_QzhdEUPjWncQ6RNjx8QaJswS4DRuAQ0aXlaIJyI
```

Status: 401 Unauthorized Size: 54 Bytes Time: 4 ms

Response Headers 5 Cookies Results Docs

```
1 {
2   "detail": "Credenciales de autenticación inválidas"
3 }
```

Analitica-de-Datos

TC 127.0.0.1:8000/login/ X TC 127.0.0.1:8000/users/me/

Send

Status: 200 OK Size: 286.16 KB Time: 153 ms

Tests Pre Run

Response Headers 7 Cookies Results Docs

AWS



dWliOiJGUkVERFkiLCJleHAI
RNjx8QaJswS4DRuAQ0aXI

- NOMBRE: ALFREDO
- EMAIL: ALFREDO.MX
- TELÉFO

Swagger



FastAPI 0.1.0 OAS 3.1

/openapi.json

Authorize

default

GET /usuarios/mostrar/ Mostrar

POST /usuarios/agregar/ Agregar

PUT /usuarios/actualizar/{id} Actualizar

DELETE /usuarios/eliminar/{id} Eliminar

POST /login/ Login

GET /users/me/ Me

GET /laptops/mostrar/ Mostrar

POST /laptops/agregar/ Agregar Laptop

PUT /laptops/actualizar/{laptop_id} Actualizar Laptop

DELETE /laptops/eliminar/{id} Eliminar

GET /pasajeros/mostrar/ Mostrar

POST /pasajeros/agregar/ Agregar Pasajero

PUT /pasajeros/actualizar/{pasajero_id} Actualizar Pasajero

DELETE /pasajeros/eliminar/{pasajero_id} Eliminar Pasajero

GET / Imprimir

Schemas

Body_login_login__post > Expand all object

HTTPValidationError > Expand all object

Laptop > Expand all object

Laptop2 > Expand all object

Pasajeros > Expand all object

Pasajeros2 > Expand all object

User > Expand all object

UserM1 > Expand all object

ValidationError > Expand all object



Redocly

Search...

GET Mostrar

POST Agregar

PUT Actualizar

DEL Eliminar

POST Login

GET Me

GET Mostrar

POST Agregar Laptop

PUT Actualizar Laptop

DEL Eliminar

GET Mostrar

POST Agregar Pasajero

PUT Actualizar Pasajero

DEL Eliminar Pasajero

GET Imprimir

API docs by Redocly

FastAPI (0.1.0)

Download OpenAPI specification: [Download](#)

Mostrar

Responses

> 202 Se devolvio la lista correctamente

Agregar

REQUEST BODY SCHEMA: application/json

```
username
  string (Username)
  required
full_name
  string (Full Name)
  required
email
  string (Email)
  required
phone
  integer (Phone)
  required
```

GET /usuarios/mostrar/

Response samples

202

Content type
application/json

null

Copy

POST /usuarios/agregar/

Request samples

Payload

Content type
application/json

{
 "username": "string",
 "full_name": "string",
 "email": "string",
 "phone": 0}

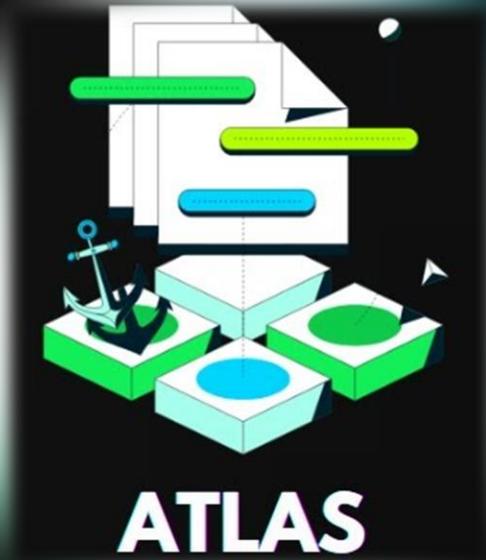
Copy

MongoDB Atlas y
MongoDB
Compass:
Ventajas,
Desventajas y
diferencias



MongoDB Atlas y MongoDB Compass

MongoDB Atlas es un servicio de base de datos en la nube que facilita el escalado y la administración rápida de bases de datos no relacionales. En cambio, MongoDB Compass es una herramienta de visualización y análisis de datos que proporciona una interfaz gráfica para explorar y manipular datos almacenados en la base de datos MongoDB.



Ventajas de MongoDB Atlas

- Escalabilidad horizontal sin interrupciones
- Recuperación ante desastres, automatizada y fácil de usar
- Seguridad avanzada y cumplimiento normativo



Desventajas de MongoDB Atlas

- **Costos**

MongoDB Atlas puede ser costoso en comparación con otras opciones de bases de datos no relacionales.

- **Complejidad**

MongoDB Atlas puede ser más complejo de configurar y mantener que otras opciones de bases de datos no relacionales.

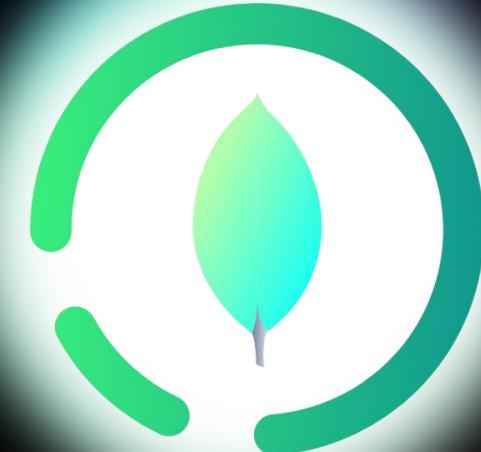
- **Limitaciones de la nube**

MongoDB Atlas tiene limitaciones en cuanto a la cantidad de almacenamiento y la capacidad de procesamiento en la nube.



Ventajas de MongoDB Compass

- Interfaz Gráfica Intuitiva
- Permite explorar y visualizar los datos de manera eficiente, facilitando la comprensión de la estructura y la distribución de los datos
- Operaciones CRUD Simplificadas mediante operaciones visuales en lugar de comandos de consola



Desventajas de MongoDB Compass

- Rendimiento

Puede ser menos eficiente en términos de rendimiento al utilizar la interfaz gráfica.

- Puede requerir más recursos del sistema en comparación con el uso de la línea de comandos
- Algunas funciones pueden requerir una conexión a Internet para descargar información adicional.



Diferencias entre MongoDB Atlas y MongoDB Compass



- **MongoDB Atlas**

MongoDB Atlas es una base de datos en la nube completamente administrada que funciona en AWS, Azure y Google Cloud. Permite escalar horizontalmente con facilidad y ofrece opciones de seguridad y cumplimiento.

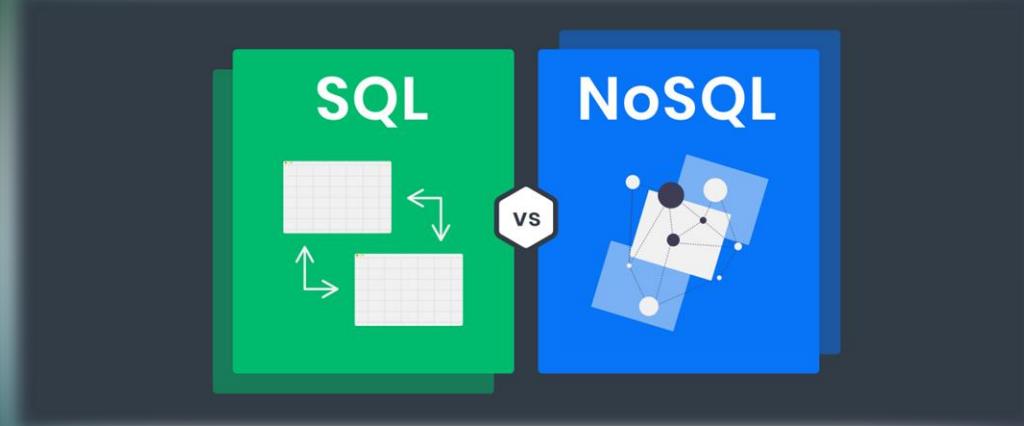
- **MongoDB Compass**

MongoDB Compass es una herramienta de visualización de datos de MongoDB que proporciona una interfaz gráfica para que los desarrolladores interactúen de manera más intuitiva y eficiente con sus datos, permitiendo la creación, análisis y modificación de datos en MongoDB.



Bases de Datos no Relacionales

Las bases de datos NoSQL son una alternativa a las bases de datos relacionales tradicionales. A diferencia de estas últimas, las NoSQL permiten almacenar y procesar eficientemente grandes cantidades de datos no estructurados o semiestructurados mediante la flexibilidad en los esquemas y la utilización de modelos de datos como documentos, grafos y clave-valor.



Comparación entre AWS, Azure, and Google Cloud



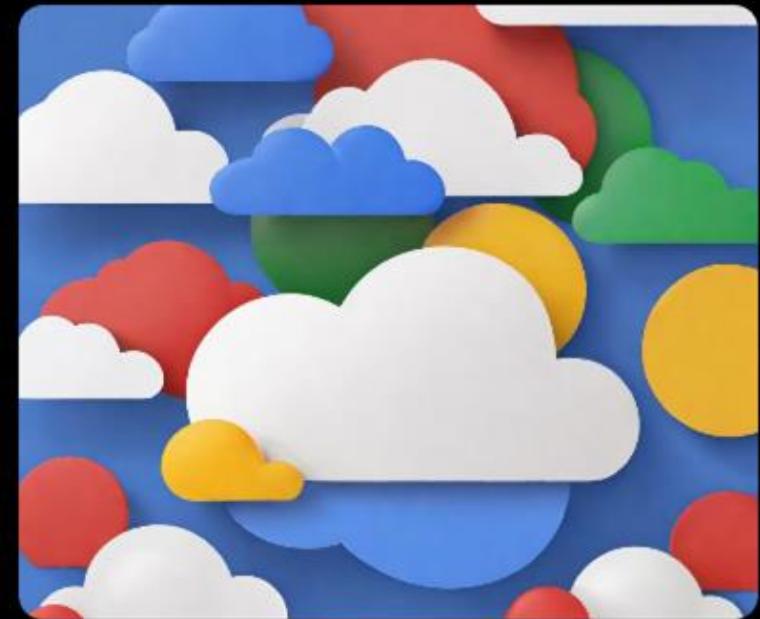
- **AWS**

AWS es conocido por su amplia gama de servicios, escalabilidad y seguridad. También es compatible con una gran cantidad de sistemas operativos y lenguajes de programación, lo que lo hace ideal para empresas que buscan una solución personalizada.



- **Azure**

Azure es una plataforma de nube híbrida que ofrece una amplia gama de servicios y una integración sin problemas con otros productos de Microsoft. También es conocido por su enfoque en la inteligencia artificial y la analítica de datos.



- **Google Cloud**

Google Cloud es conocido por su enfoque en la innovación y la tecnología de vanguardia. También es compatible con una amplia gama de lenguajes de programación y tiene una gran cantidad de herramientas de análisis de datos.