



Bases de datos NoSQL

Diego Iván Oliveros Acosta

Agenda

- Contexto y conceptos
- Big Data
- NoSQL
- **MongoDB**
- Instalación
- Uso
- Comandos comunes
- Servicios Web **Atlas**

La creación de datos, a punto de explotar

Cantidad real y prevista de datos generados en todo el mundo (en zettabytes)

Contexto


CADA DÍA CREAMOS
2,5 QUINTILLONES
DE BYTES DE
DATOS. (2,5 Exabytes)



JAN
2021

OVERVIEW OF GLOBAL INTERNET USE

A SNAPSHOT OF INTERNET USE AROUND THE WORLD

 INTERNET USER NUMBERS NO LONGER INCLUDE DATA SOURCED FROM SOCIAL MEDIA PLATFORMS, SO VALUES ARE **NOT COMPARABLE** WITH PREVIOUS REPORTS

TOTAL NUMBER
OF GLOBAL
INTERNET USERS



4.66
BILLION

INTERNET USERS AS A
PERCENTAGE OF TOTAL
GLOBAL POPULATION



59.5%

ANNUAL CHANGE
IN THE NUMBER OF
GLOBAL INTERNET USERS



+7.3%
+316 MILLION

AVERAGE DAILY TIME SPENT
USING THE INTERNET BY
EACH INTERNET USER



6H 54M

PERCENTAGE OF USERS
ACCESSING THE INTERNET
VIA MOBILE DEVICES



92.6%

SOURCES: KEPIOS (JAN 2021) BASED ON EXTRAPOLATIONS OF DATA PUBLISHED BY: THE ITU; LOCAL GOVERNMENT BODIES; GWI; GSMA INTELLIGENCE; EUROSTAT; APJII; CNNIC; THE UNITED NATIONS. DATA FOR TIME SPENT AND MOBILE INTERNET SHARE FROM GWI (Q3 2020). SEE [GLOBALWEBINDEX.COM](https://www.globalwebindex.com) FOR MORE DETAILS. **COMPARABILITY ADVISORY:** SOURCE AND BASE CHANGES. INTERNET USER NUMBERS NO LONGER INCLUDE DATA SOURCED FROM SOCIAL MEDIA PLATFORMS. FIGURES ARE **NOT COMPARABLE** WITH DATA PUBLISHED IN PREVIOUS REPORTS.

JAN
2022

OVERVIEW OF INTERNET USE

ESSENTIAL INDICATORS OF INTERNET ADOPTION AND USE



TOTAL
INTERNET
USERS



KEROS

4.95
BILLION

INTERNET USERS AS
A PERCENTAGE OF
TOTAL POPULATION



we
are
social

62.5%

YEAR-ON-YEAR CHANGE
IN THE NUMBER OF
INTERNET USERS



we
are
social

+4.0%
+192 MILLION

AVERAGE DAILY TIME SPENT
USING THE INTERNET BY
EACH INTERNET USER



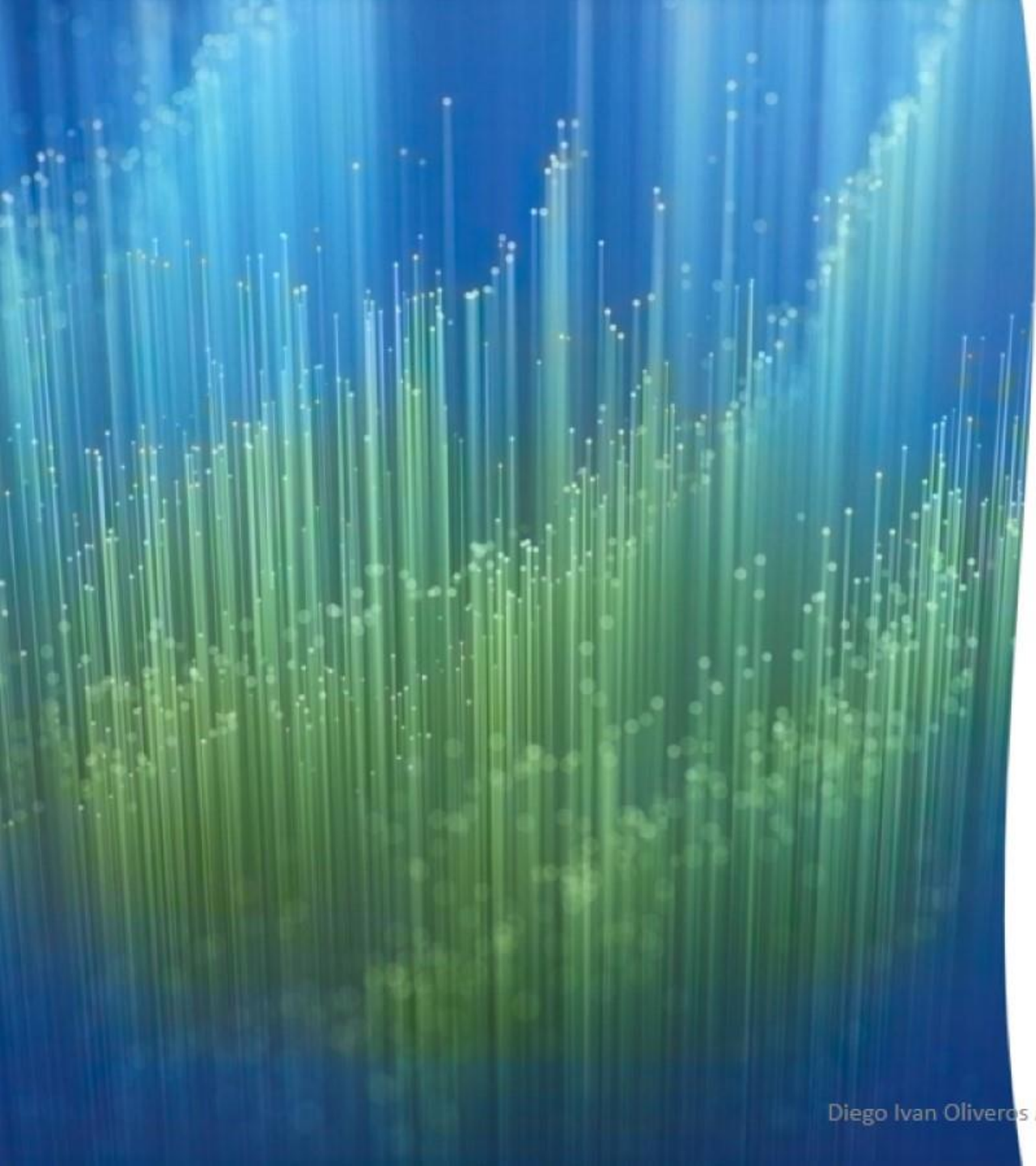
GWJ.

6H 58M
+1.0% (+4M)

PERCENTAGE OF USERS
ACCESSING THE INTERNET
VIA MOBILE PHONES



92.1%



PRINCIPALES CAMBIOS QUE SE PRODUJERON EN LA TECNOLOGÍA (últimos 20 años)

Los masificación uso de internet

Surgimiento de las redes sociales

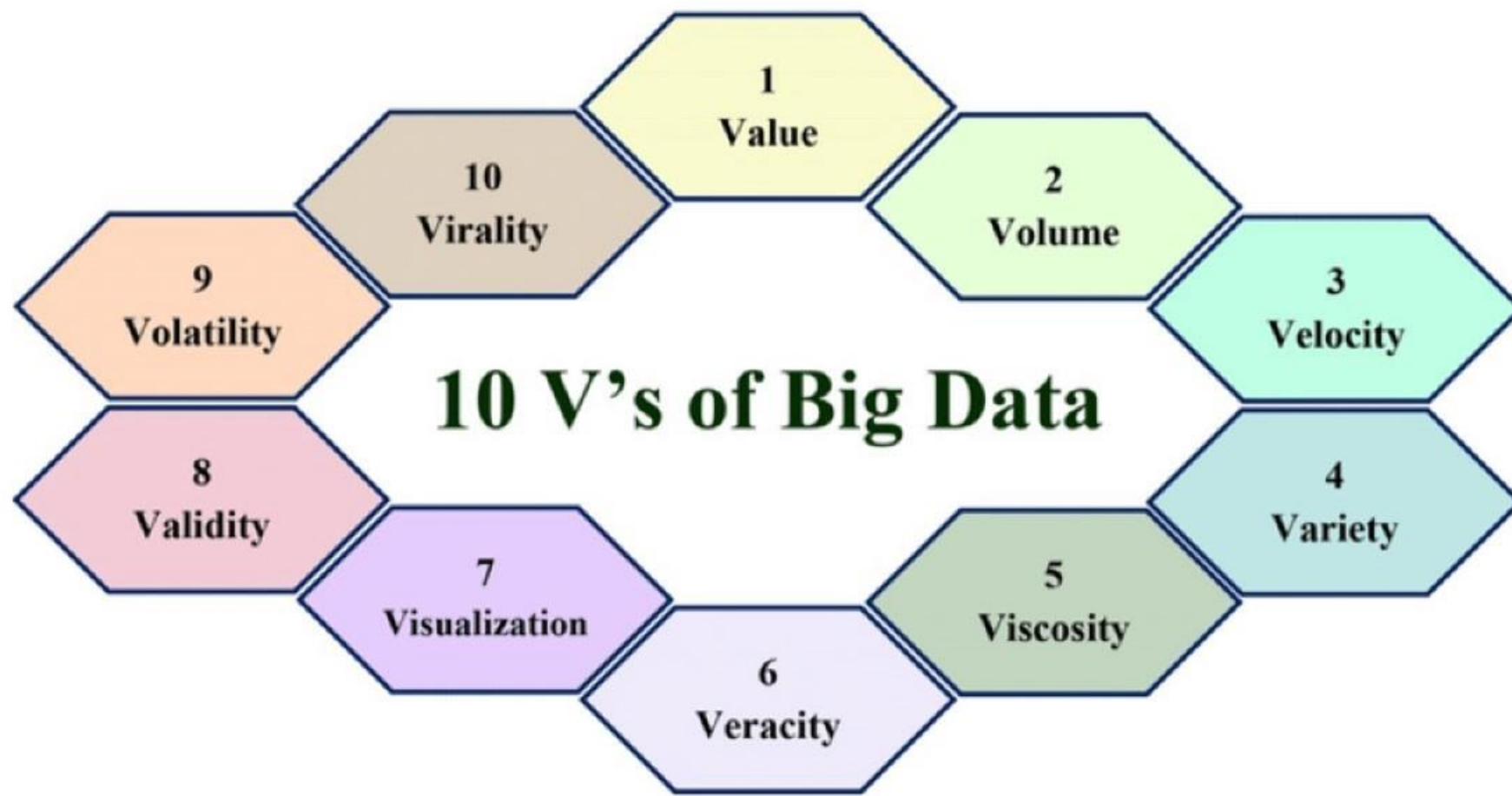
Crecimiento exponencial de dispositivos móviles

Interfaces de usuario más simples e intuitivas

Cambios en las formas de procesamiento

Reducción exponencial en el costo de almacenamiento

Big Data



End to End Platform Builds

Business & Data Analysis

Tools & Technologies

DevOps & Cloud
Computing

Automation

Advanced
Analytics

Stanford
NLP



Spark
MLlib



Data
Visualization



tableau
SOFTWARE



Libraries – D3.js

Data
Management



mongoDB



elasticsearch

Data
Integration/
Acquisition



Apache
Tika

talend*

alteryx



pentaho*
A Hitachi Group Company

Custom Apps

Cloud
Compute

Infrastructure as a
Service

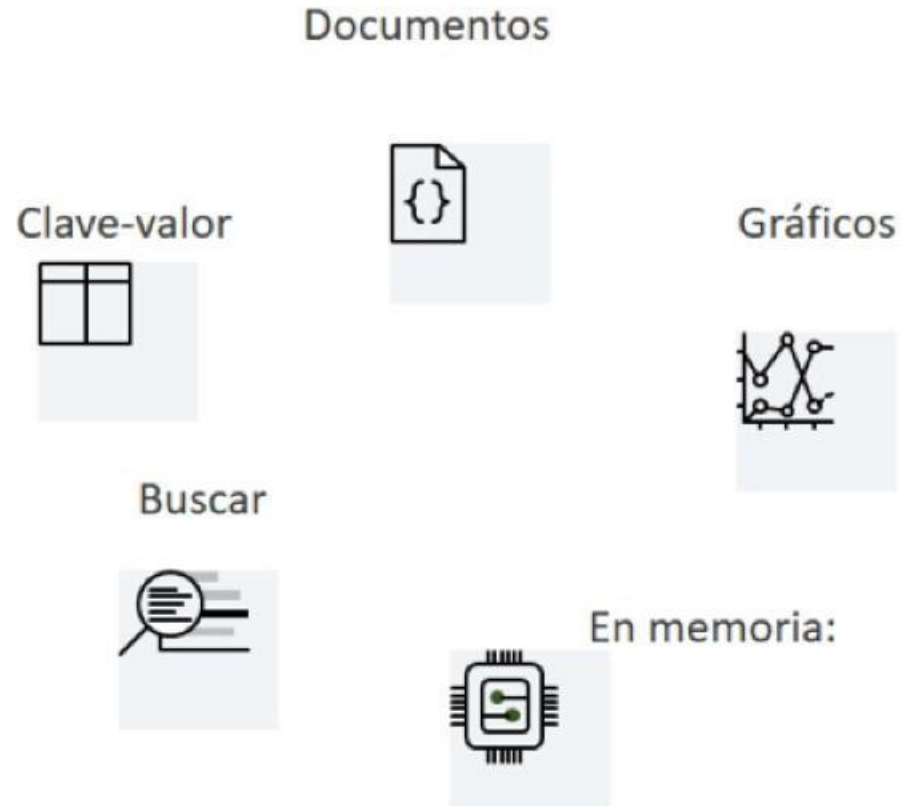
Data as a Service

Software as a
Service



NoSQL

- **Flexibilidad**
- **Escalabilidad**
- **Alto rendimiento**
- **Altamente funcional**



Trusted by thousands of companies, from startup to enterprise

 inVISION

 ebay

 Adobe

 Google



- De propósito general
- Open-source NoSQL
- Bases de datos mas populares
- Escalado horizontal
- Bases de datos de gran tamaño
- Datos no estructurados
- Para: app del día - día móviles y web



$$\text{Elasticidad (E)} = \frac{\% \text{ variación de X}}{\% \text{ variación de Y}}$$

Humongous
/hju:ˈmʌŋgəs/

- Escalabilidad
- Elasticidad



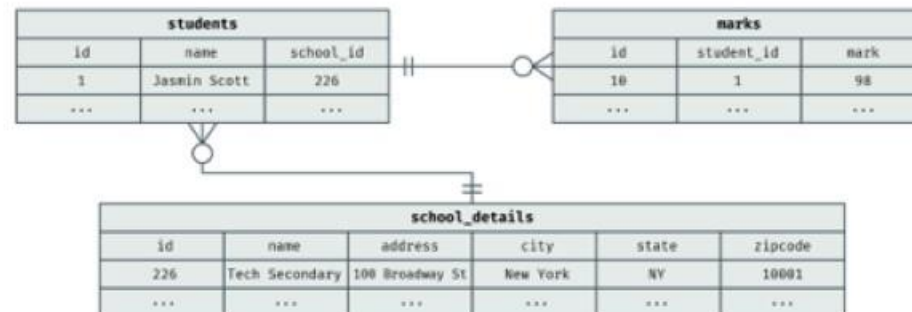
MongoDB

```
{
  "_id": 1,
  "student_name": "Jasmin Scott",
  "school": {
    "school_id": 226,
    "name": "Tech Secondary",
    "address": "100 Broadway St",
    "city": "New York",
    "state": "NY",
    "zipcode": "10001"
  },
  "marks": [98, 93, 95, 88, 100],
}
```

mongo

```
> db.students.find({"student_name":
  "Jasmin Scott"})
```

SQL



Results

name	mark	school_name	city
Jasmin Scott	98	Tech Secondary	New York
...

sql

```
SELECT s.name, m.mark, d.name as "school name",
d.city
FROM students s
INNER JOIN marks m ON s.id = m.student_id
INNER JOIN school_details d ON s.school_id = d.id
WHERE s.name = "Jasmin Scott";
```




Performance Charts.

Sidebar. Redesigned.

Visualize your Schema.

Build Geo Queries.

Interactive Document Editor.

Visual Explain Plans.

Index Management.

Schema Validation.

Improved CRUD

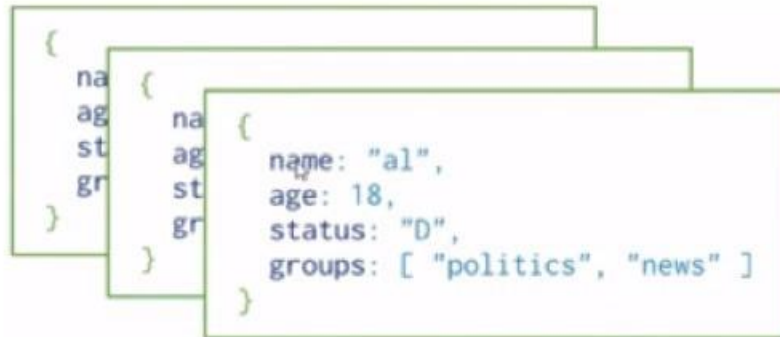
Deployment Awareness

Query History

Es orientado a documentos u
objetos de java script
Sin esquemas
Json



Colecciones

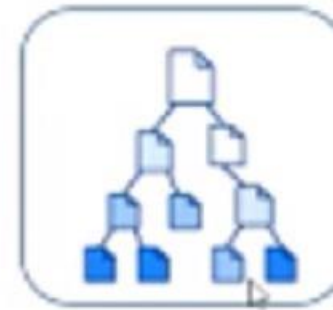


Documentos:

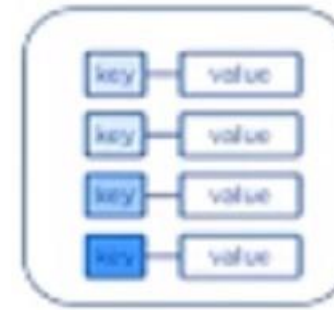
```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```

← field: value
← field: value
← field: value
← field: value

Base de datos:



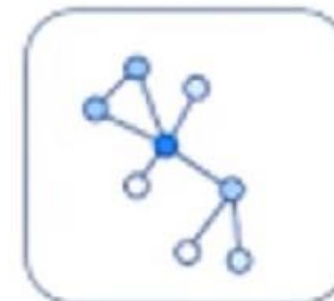
Document
Store



Key-Value
Store



Wide-Column
Store



Graph
Store

Características de MongoDB

- Podemos hacer búsqueda por campos, consultas de rangos y expresiones regulares. Además, estas consultas pueden devolver un campo específico del documento, pero también puede ser una función JavaScript definida por el usuario.
- **Replicación**. Del mismo modo, la replicación es un proceso básico en la gestión de bases de datos. MongoDB soporta el tipo de replicación primario-secundario. Mientras podemos realizar consultas con el primario, el secundario actúa como réplica de datos en solo lectura a modo copia de seguridad con la particularidad de que los nodos secundarios tienen la habilidad de poder elegir un nuevo primario en caso de que el primario actual deje de responder.
- **Balanceo de carga**. Resulta muy interesante cómo MongoDB puede escalar la carga de trabajo. MongoDB tiene la capacidad de ejecutarse de manera simultánea en múltiples servidores, ofreciendo un balanceo de carga o servicio de replicación de datos, de modo que podemos mantener el sistema funcionando en caso de un fallo del hardware.
- **Ejecución de JavaScript del lado del servidor**. MongoDB tiene la capacidad de realizar consultas utilizando JavaScript, haciendo que estas sean enviadas directamente a la base de datos para ser ejecutadas.

Consideraciones:

- ¿Tiene cargas de trabajo de gran volumen que requieren gran escala?
- ¿Los datos son dinámicos y cambian con frecuencia ?
- ¿Los datos se pueden expresar sin relaciones?
- ¿Necesita escrituras rápidas y la seguridad de escritura no es crítica?
- ¿La recuperación de datos es sencilla ?
- ¿Los datos requieren una distribución geográfica amplia?



Estructura de los datos

JSON

BSON

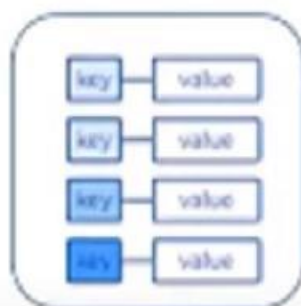
```
{ "BSON": [ "awesome", 5.05, 1986 ] }
```

```
\x31\x00\x00\x00
\x04BSON\x00
\x26\x00\x00\x00
\x02\x30\x00\x08\x00\x00\x00awesome\x00
\x01\x31\x00\x33\x33\x33\x33\x33\x33\x14\x40
\x10\x32\x00\xc2\x07\x00\x00
\x00
\x00
```


Administración de los datos



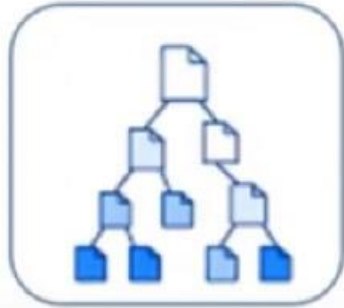
↳



Key-Value
Store

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```

← field: value
← field: value
← field: value
← field: value



Document
Store

```
{
  _id: <ObjectId>,
  username: "123xyz",
  contact: {
    phone: "123-456-7890",
    email: "xyz@example.com"
  },
  access: {
    level: 5,
    group: "dev"
  }
}
```

Embedded sub-
document

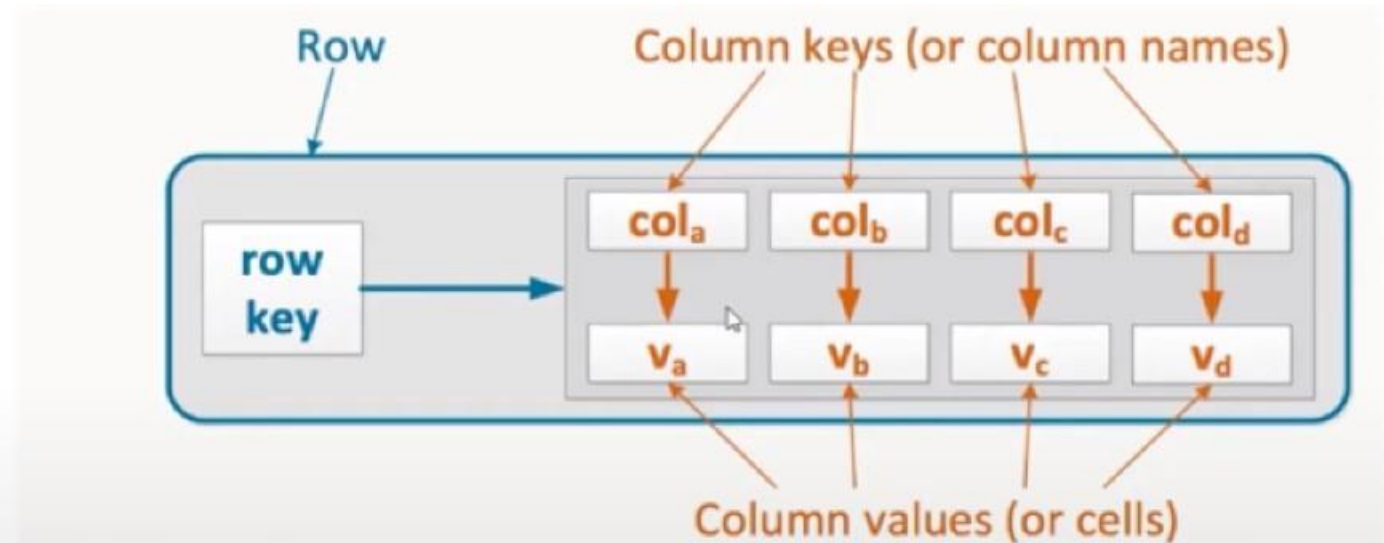
Embedded sub-
document

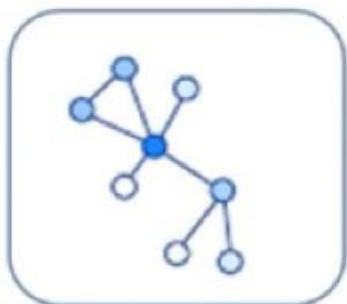
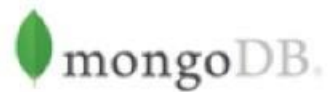
Document store

Administración de los **datos**

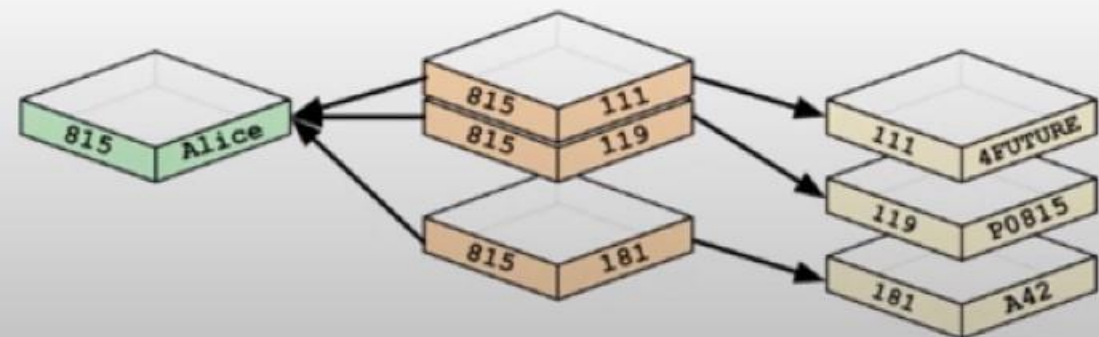
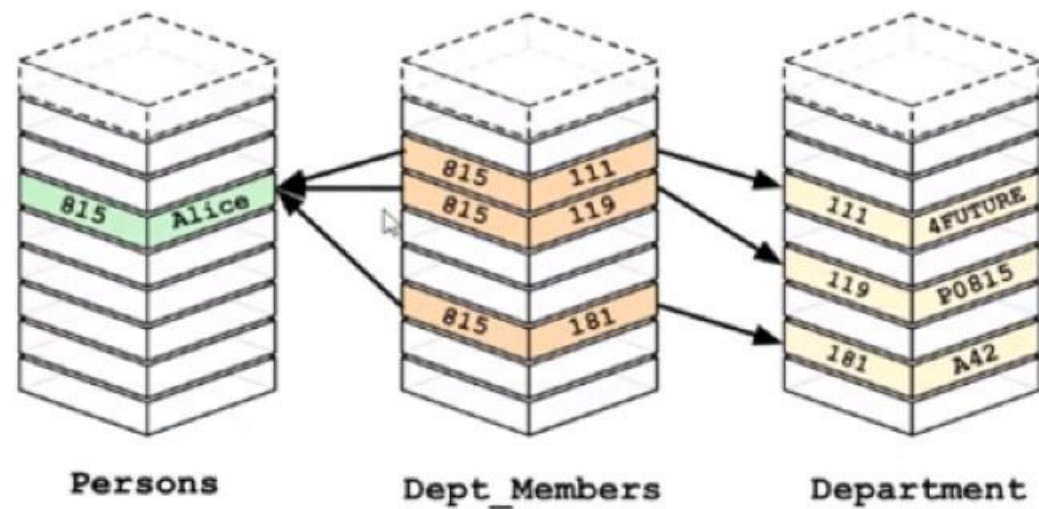


Wide-Column
Store

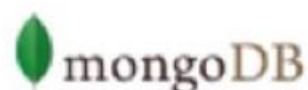




Graph
Store



Insertar Documentos



<https://docs.mongodb.com/manual/tutorial/insert-documents/>

<code>db.collection.insertOne()</code>	Inserts a single document into a collection.
<code>db.collection.insertMany()</code>	<code>db.collection.insertMany()</code> inserts multiple documents into a collection.
<code>db.collection.insert()</code>	<code>db.collection.insert()</code> inserts a single document or multiple documents into a collection.

`db.inventory.insertOne()`

```
{ item: 'canvas', qty: 100, tags: ['cotton'], size: { h: 28, w: 35.5, uom: 'cm' } }
```

`db.inventory.insertMany()`

```
{ item: 'journal', qty: 25, size: { h: 14, w: 21, uom: 'cm' }, status: 'A' },  
{ item: 'notebook', qty: 50, size: { h: 8.5, w: 11, uom: 'in' }, status: 'A' },  
{ item: 'paper', qty: 100, size: { h: 8.5, w: 11, uom: 'in' }, status: 'D' },  
{ item: 'planner', qty: 75, size: { h: 22.85, w: 30, uom: 'cm' }, status: 'D' }, { item: 'postcard', qty: 45, size: { h: 10, w: 15.25, uom: 'cm' }, status: 'A' }
```

`ObjectId(hexadecimal)`

Returns a new `ObjectId` value. The 12-byte `ObjectId` value consists of:

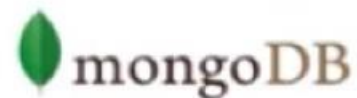
- a 4-byte timestamp value, representing the `ObjectId`'s creation, measured in seconds since the Unix epoch
- a 5-byte random value
- a 3-byte incrementing counter, initialized to a random value

While the BSON format itself is little-endian, the timestamp and counter values are big-endian, with the most significant bytes appearing first in the byte sequence.

`ObjectId()` can accept the following parameter:

Field	Type	Description
<code>hexadecimal</code>	String	Optional. Hexadecimal string value for the new <code>ObjectId</code> .

Buscar Documentos



<https://docs.mongodb.com/manual/reference/method/db.collection.find/index.html>

`db.inventory.find({})` → `SELECT * FROM inventory`

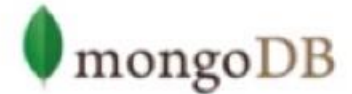
```
{ "_id": "apples", "qty": 5 }  
{ "_id": "bananas", "qty": 7 }  
{ "_id": "oranges", "qty": { "in stock": 8, "ordered": 12 } }  
{ "_id": "avocados", "qty": "fourteen" }
```

`$gt` = greater than

→ `db.collection.find({ qty: { $gt: 4 } })`

↓
{ "_id": "apples", "qty": 5 }
{ "_id": "bananas", "qty": 7 }

Buscar Documentos



<https://docs.mongodb.com/manual/reference/method/db.collection.find/index.html>

bios collection

`{ "_id": <value>,`

`"name": { "first": <string>, "last": <string> }, // embedded document`

`"birth": <ISODate>,`

`"death": <ISODate>,`

`"contribs": [<string>, ...], // Array of Strings`

`"awards": [{ "award": <string>, year: <number>, by: <string> } // Array of embedded documents ...]`

`db.bios.find({ "name.last": { $regex: /^N/ } })`

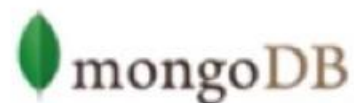
`db.bios.find({ "_id": { $in: [5, ObjectId("507c35dd8fada716c89d0013")] } })`

`db.bios.find({ "_id": 5 })`

`db.bios.find({ "name.last": "Hopper" })`

```
db.bios.find( {  
    birth: { $gt: new Date('1920-01-01') },  
  
    death: { $exists: false }  
})
```

Buscar Documentos (con funciones)



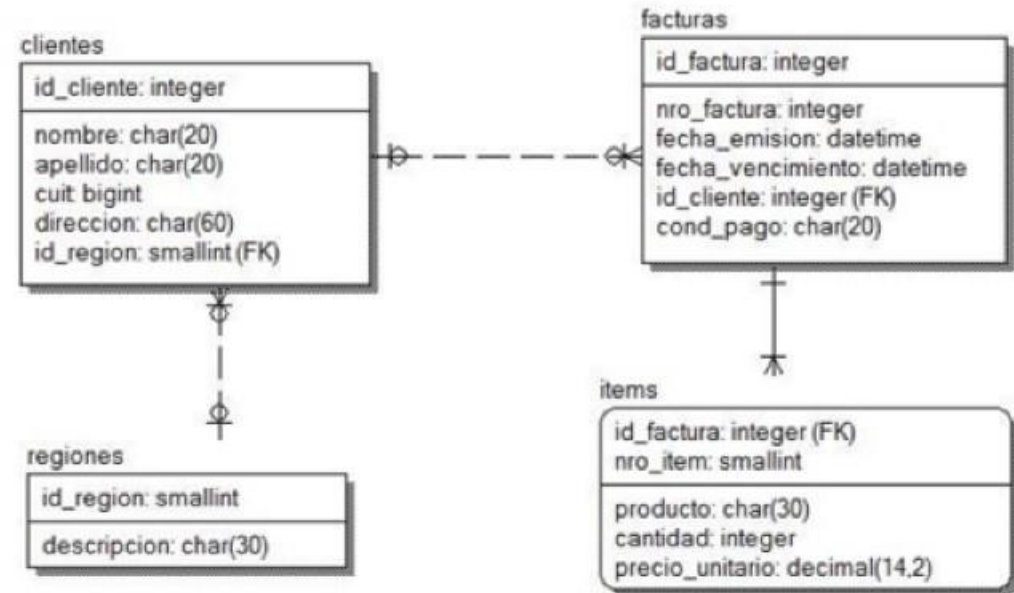
- `db.system.js.save({ _id: "markMapedItem", value : function(x) { return {...x, maped: true}; } })`
- `db.loadServerScripts()`
- `db.collection.find().map(markMapedItem)`

Instalación

1. Instalar MongoDB
2. Actualizar variable Path
 - Buscar el directorio donde se instaló MongoDB (por ejemplo C:\Program Files\MongoDB\Server\3.2\bin)
 - Añadir MongoDB a la variable Path (Inicio > Equipo > Propiedades del Sistema > Opciones avanzadas)
3. Crear una carpeta (con los permisos adecuados) para guardar la base de datos
 - C:/Data/Db es la carpeta default para MongoDB
 - Si quisiéramos tener otro path debemos ejecutar desde consola: `mongod --dbpath ruta/nueva-a/la-carpeta-db`
4. Abrir conexión desde Consola de Windows (Símbolo del Sistema) ejecutando el comando `mongod`
5. Mantener la consola de conexión abierta y abrir una nueva consola para operar sobre la base

Uso

- **Armar un modelo que contenga la información de las facturas y todos sus ítems, detallando el nombre, apellido, ciudad y región del cliente al que se le emitió la factura, para poder realizar consultas desde un portal de facturas de la forma optima posible.**



Comandos comunes

- `C:>mongod`
- `C:>mongo`
- `show dbs`
- `show collections`
- `db.facturas.help()`
- `db.facturas.find()`
- `db.facturas.count()`
- `db.facturas.dataSize()`
- `db.facturas.insert({"": ""})`
- `db.<collection_name>.remove({criterio_de Eliminación})`
- **`db.coleccion.update ()`**

Servicios en la nube

- Local o desplegado
- Mlab
- Studio 3t
- Mongo DB Atlas
- NoSqlBooster



NoSQLBooster

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

Diego Ivan Oliveros Acosta @scalapp.co

[Sign In](#)[Try Free](#)

Your Company (optional)

scalapp

How are you using MongoDB?

I'm learning MongoDB



Your Work Email

ceo@scalapp.co

First Name

Diego

Last Name

Oliveros

Password

✓ 8 characters minimum



I agree to the [terms of service](#) and [privacy policy](#).

[Get started free](#)



Great, now verify your email



Check your inbox at [<ceo@scalapp.co>](mailto:ceo@scalapp.co) and click the verification link inside to complete your registration. This link will expire shortly, so verify soon!

Don't see an email? Check your spam folder.

Link expired? [Resend verification email](#)

Diego Ivan Oliveros Acosta @scalapp.co



Welcome!

Use your account to deploy a **cloud database**
with **MongoDB Atlas** and contact **Support**.

PREVIEW

Serverless

For serverless applications that aren't critical with variable traffic. Minimal configuration required.

- ✓ Pay only for the operations you run
- ✓ Resources scale seamlessly to meet your workload
- ✓ Always-on security and backups

Create

Starting at
\$0.30/1M reads

[do this later](#)

Dedicated

For production applications with sophisticated workload requirements. Advanced configuration controls.

- ✓ Network isolation and fine-grained access controls
- ✓ On-demand performance advice
- ✓ Multi-region and multi-cloud options available

Create

Starting at
\$0.08/hr*
*estimated cost \$26.94/month

FREE

Shared

For learning and exploring MongoDB in a cloud environment. Basic configuration options.

- ✓ No credit card required to start
- ✓ Explore with sample datasets
- ✓ Upgrade to dedicated clusters for full functionality

Create

Starting at
FREE

[Advanced Configuration Options](#)

▼

7



-1



0



0



2

AWS



EUROPE

 N. Virginia (us-east-1) ★

🇩🇪 Frankfurt (eu-central-1) ★

 Murni

 Oregon (us-west-2) ★

🇮🇪 Ireland (eu-west-1) ★

Singh

AUSTRALIA

 Sydney (ap-southeast-2) ★

defeated action: promising 3 servers)

[✦ Create](#)

Business Collections

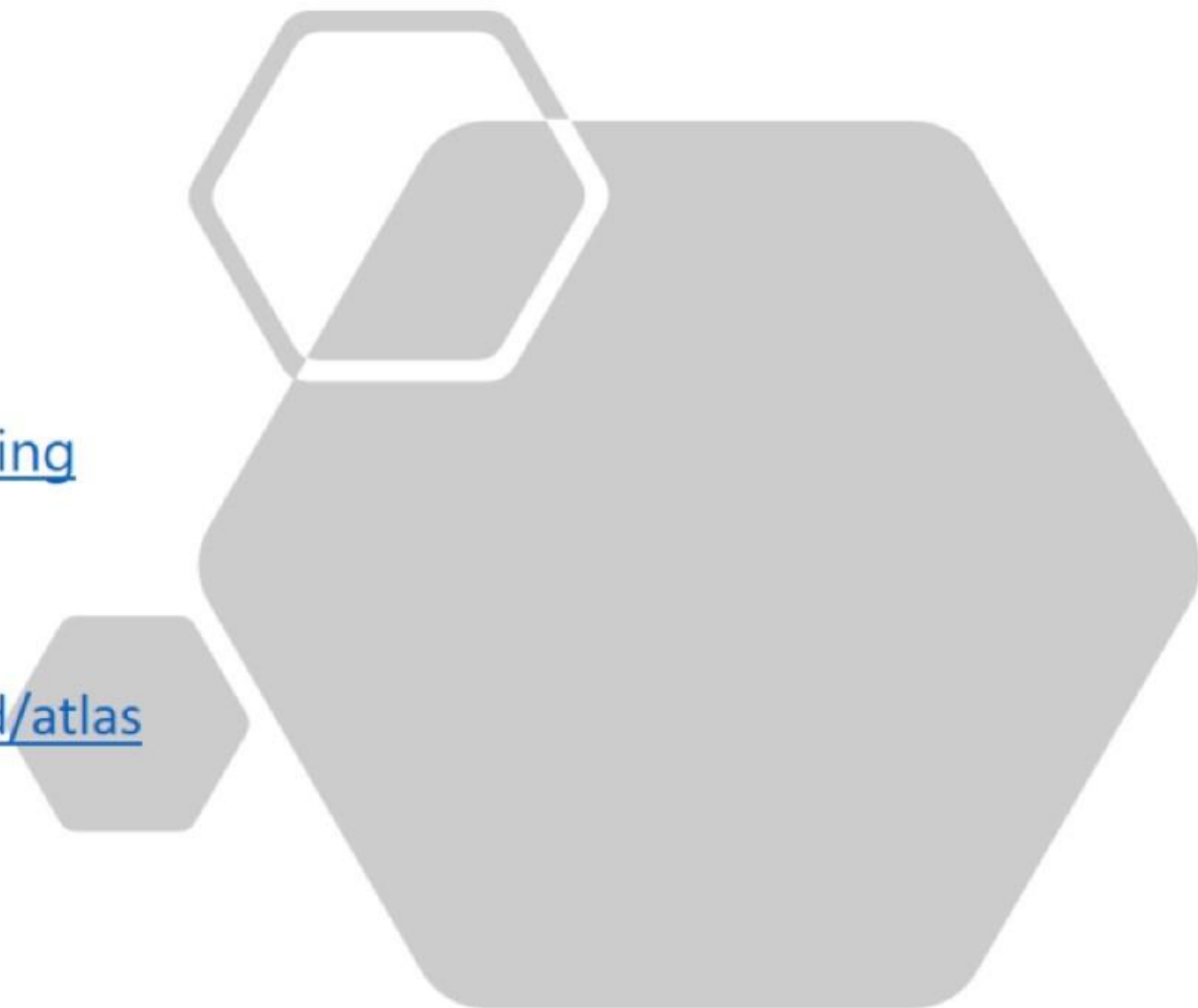
FREE

SPARE

Your cluster is being created.

Referencias

- <https://www.mongodb.com/>
- <http://db-engines.com/en/ranking>
- <https://nosqlbooster.com/>
- <https://mlab.com/>
- <https://www.mongodb.com/cloud/atlas>





Gracias