



Alicia D. Benítez  
Dpto. de Informática  
IES Francisco Ayala  
Granada

# Índice

- 1.- Bases de Datos NoSQL.
- 2.- MongoDB.
- 3.- Instalación.
- 4.- Borrar la base de datos del sistema.
- 5.- ¿Cómo funciona?
- 6.- Usando MongoDB.
- 7.- Crear una BD.
- 8.- Crear una colección e insertar un documento.
- 9.- Borrar la BD.
- 10.- Consultar elementos.
- 11.- Editar un documento.
- 12.- Eliminar un documento.
- 13.- Consultas.

# 1.- Base de Datos NoSQL

- Base de Datos (BD) NoSQL (Not Only SQL):
  - Sistema Gestor de Base de Datos.
  - No cumple el esquema Entidad-Relación.
  - No usa SQL para consultas, aunque pueden usarlo.
  - No usan tablas de almacenamiento.
  - No soportan operaciones JOIN.
  - Permiten trabajar con grandes volúmenes de datos (Google, Amazon, Twitter, Facebook).
  - Da más importancia al rendimiento en tiempo real que a la coherencia.
  - Escrita en C++.



# 1.- Base de Datos NoSQL (II)

- BD SQL vs NoSQL:

SQL	NoSQL
Permite unir tablas con JOIN	No permite el uso de JOINS o están muy limitados
Dificultad para distribuir los datos	Facilidad para distribuir los datos
Escalabilidad vertical (mejorar potencia del servidor para obtener mejores resultados)	Escalabilidad horizontal (repartir / distribuir base de datos en diferentes servidores)
Los datos se estructuran siguiendo los esquemas de las tablas	Los datos no siguen ningún esquema
Posibilidad de crear restricciones, triggers, claves foráneas... sobre los datos	Estas utilidades no suelen estar disponibles

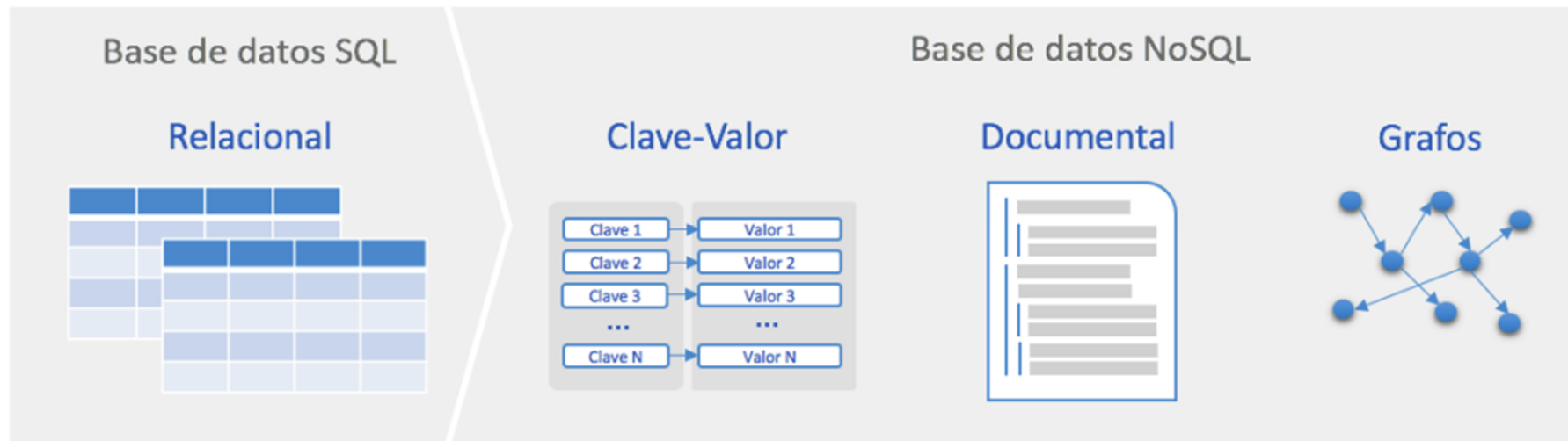
# 1.- Base de Datos NoSQL (III)

- BD SQL vs NoSQL :

 <b>SQL</b>	 <b>NoSQL</b>
<p>Cuando el volumen de mis datos no crece o lo hace poco a poco.</p>	<p>Cuando el volumen de mis datos crece muy rápidamente en momentos puntuales.</p>
<p>Cuando las necesidades de proceso se pueden asumir en un sólo servidor.</p>	<p>Cuando las necesidades de proceso no se pueden preveer.</p>
<p>Cuando no tenemos picos de uso del sistema por parte de los usuarios más allá de los previstos.</p>	<p>Cuando tenemos picos de uso del sistema por parte de los usuarios en múltiples ocasiones.</p>







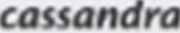


# 1.- Base de Datos NoSQL (IV)

- Tipos BD NoSQL :



# 1.- Base de Datos NoSQL (V)

- Ejemplos BD NoSQL:

Documentales	Datos semi-estructurados en documentos (XML, YAML, JSON y BSON)	 elastic  mongoDB
Grafo	Datos estructurados como nodos relacionados entre si	 Neo4j  OPENLINK SOFTWARE <small>Making Technology Work for You®</small>
Clave / valor	Datos estructurados como clave / valor	 redis  emCached
Familia de columnas	Datos estructurados en columnas donde cada fila puede tener una configuración diferente	 cassandra   hadoop



mongoDB

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

## 2.- MongoDB

- MongoDB:
  - “hum**ong**ous” que significa enorme .
  - Sistema de BD NoSQL orientado a documentos.
  - Open source.
  - Integración de datos en aplicaciones fácil y rápida.
  - Creada en 2007 por 10gen.
  - Disponible para Windows, Linux, OSX, Solaris.





mongoDB

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

## 2.- MongoDB (II)

- Características generales:
  - Soporta búsqueda por campos, consultas de rangos y expresiones regulares.
  - Cualquier campo puede ser indexado.
  - Balanceo de carga y replicación de datos en múltiples servidores.
  - Puede realizar consultas en JavaScript en la BD.
  - Permite almacenar ficheros: imágenes, vídeos...



mongoDB

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

## 2.- MongoDB (III)

- Desventajas:
  - Problemas de consistencia de datos con versiones obsoletas de documentos => Puede devolver datos incorrectos.
  - Pérdidas de información cuando no se ha escrito aún la información en el espacio de almacenamiento permanente.
  - Problemas de rendimiento cuando el volumen de datos supera los 100TB.

## 2.- MongoDB (IV)



mongoDB

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

- Empresas que usan MongoDB:





mongoDB

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

## 3.- Instalación

- Paso 1: Añadir MongoDB al repositorio.
  - Añadimos la clave del repositorio oficial.
    - `curl -sL "http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xA6A19B38D3D831EF" | sudo apt-key add`
    - `sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 -recv 2930ADAE8CAF5059EE73BB4B58712A2291FA4AD5`
  - Añadimos MongoDB al repositorio:
    - `echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.6 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.6.list`



mongoDB

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

## 3.- Instalación (II)

- Paso 2: Instalar MongoDB en Ubuntu 18.04.
  - `sudo apt-get update`
  - `sudo apt install -y mongodb-org`
- Paso 3: Manejo de la BD: parar, arrancar y activar MongoDB para arranque automático.
  - `sudo systemctl stop mongod.service`
  - `sudo systemctl start mongod.service`
  - `sudo systemctl enable mongod.service`



mongoDB

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

## 3.- Instalación (III)

– Testear que MongoDB se está ejecutando:

- `sudo systemctl status mongod`

```
alicia@rosetta:~$ sudo systemctl status mongod
[sudo] password for alicia:
● mongod.service - High-performance, schema-free document-oriented database
   Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2018-12-26 18:58:40 UTC; 1h 9min ago
     Docs: https://docs.mongodb.org/manual
    Main PID: 6721 (mongod)
      CGroup: /system.slice/mongod.service
              └─6721 /usr/bin/mongod --auth --config /etc/mongod.conf

dic 26 18:58:40 rosetta systemd[1]: Started High-performance, schema-free document-oriented database.
alicia@rosetta:~$
```

– Por defecto, escucha con el puerto 27017.

– Para conectar MongoDB a la shell:

- `mongo --host 127.0.0.1:27017`

```
alicia@rosetta:~$ mongo --host 127.0.0.1:27017
MongoDB shell version v3.6.9
connecting to: mongodb://127.0.0.1:27017/
Implicit session: session { "id" : UUID("37b48e56-7f86-4849-90b9-5116b6ab9b0c") }
MongoDB server version: 3.6.9
>
```



mongoDB

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

## 3.- Instalación (IV)

- Paso 4: Añadir un usuario administrador.
  - Crea un nuevo usuario administrador:
    - use admin
  - Ejecuta el comando para crear un nuevo administrador:

```
> use admin
switched to db admin
> db.createUser({user:"admin", pwd:"new_password_here", roles:[{role:"root", db:"admin"}]})
Successfully added user: {
  "user" : "admin",
  "roles" : [
    {
      "role" : "root",
      "db" : "admin"
    }
  ]
}
> db.createUser({user:"admin", pwd:"alicia", roles:[{role:"root", db:"admin"}]})
2018-12-26T18:52:11.995+0000 E QUERY [thread1] Error: couldn't add user: User "admin@admin" already exists :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DB.prototype.createUser@src/mongo/shell/db.js:1437:15
@shell):1:1
> db.changeUserPassword('admin','alicia')
> exit
bye
```



mongoDB

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

## 3.- Instalación (V)

- Paso 4:
  - Sal y continua con la autenticación del login, mediante la edición del fichero de configuración:
    - `sudo nano /lib/systemd/system/mongod.service`
  - Modifica la línea roja añadiendo la opción verde:

```
[Unit]
Description=High-performance, schema-free document-oriented database
After=network.target
Documentation=https://docs.mongodb.org/manual

[Service]
User=mongodb
Group=mongodb
ExecStart=/usr/bin/mongod --auth --config /etc/mongod.conf
PIDFile=/var/run/mongodb/mongod.pid
# file size
```

- Guarda los cambios y sal de nano.





mongoDB

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

## 3.- Instalación (VI)

- Paso 4:
  - Haz un restart a la base de datos para que se apliquen los cambios:
    - `sudo systemctl daemon-reload`
    - `sudo service mongod restart`
  - Ahora solo los usuarios autenticados tendrán acceso al servidor de la base de datos:
    - `mongo -u admin -p new_password_here --authenticationDatabase admin`
- FUENTE: <https://websiteforstudents.com/install-mongodb-on-ubuntu-18-04-lts-beta-server/>



mongoDB

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

- Tarea 1: Instalar MongoDB

## 4.- Borrar la base de datos del sistema



mongoDB

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

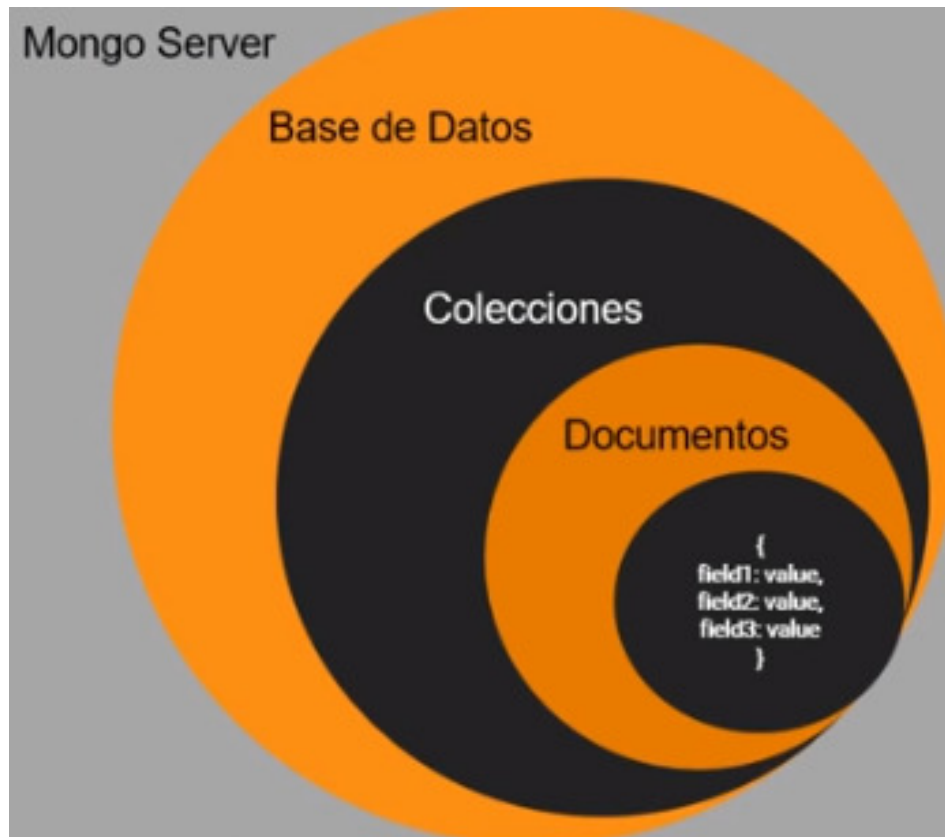
- Para el servidor de la base de datos:
  - `sudo systemctl stop mongod.service`
- Elimina los paquetes:
  - `sudo apt purge mongodb-org*`
- Borra las bases de datos y los ficheros de log:
  - `sudo rm -r /var/log/mongodb`
  - `sudo rm -r /var/lib/mongodb`



mongoDB

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

## 5.- ¿Cómo funciona?



- **Mongo Server**: Servidor de Mongo que contiene la BD.
- **Base de Datos**: Contiene colecciones.
- **Colecciones**: Análogo a las tablas en las BD SQL.
- **Documentos**: Análogo a los registros en una BD SQL .
- Cada **documento** son **pares de clave-valor**:
  - La **clave** es el nombre del campo. Son cadenas de texto.
  - El **valor** es el valor del campo. Son cadenas de texto, números, listas, fechas, valores booleanos, otros objetos...



mongoDB

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

## 5.- ¿Cómo funciona? (II)

- Definimos un documento para PERSONA en MongoDB:

{ (Los identificadores no llevan comillas)

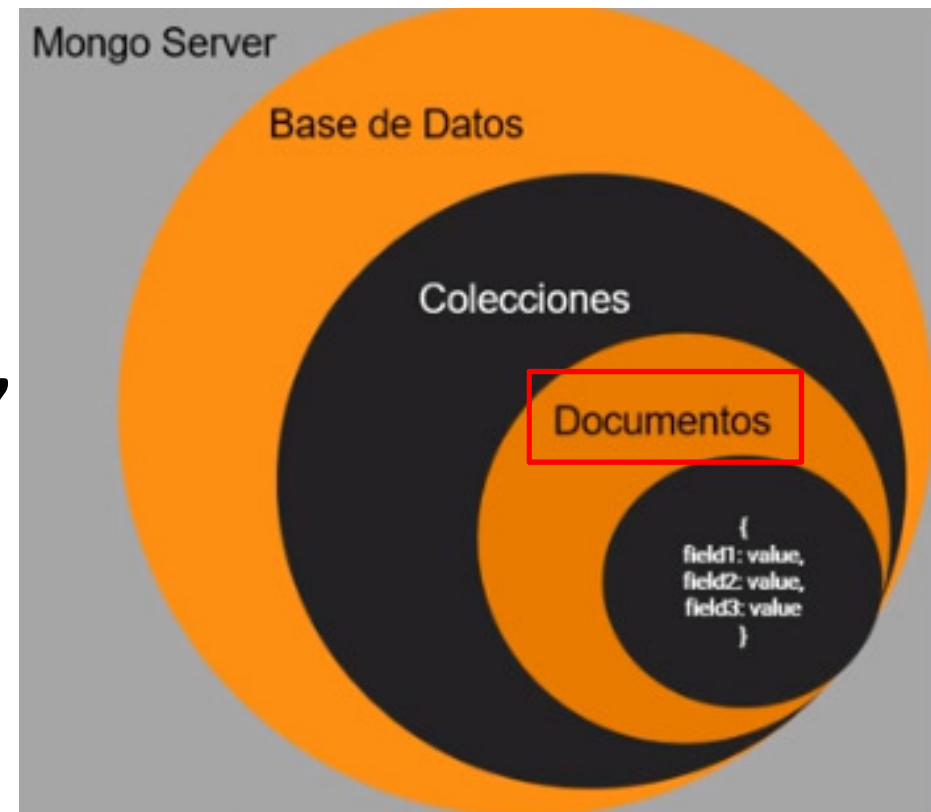
“nombre”: “Maria”,

“apellido”: “Rodriguez”,

“DNI”: “12345678C”,

“edad”: 38

}



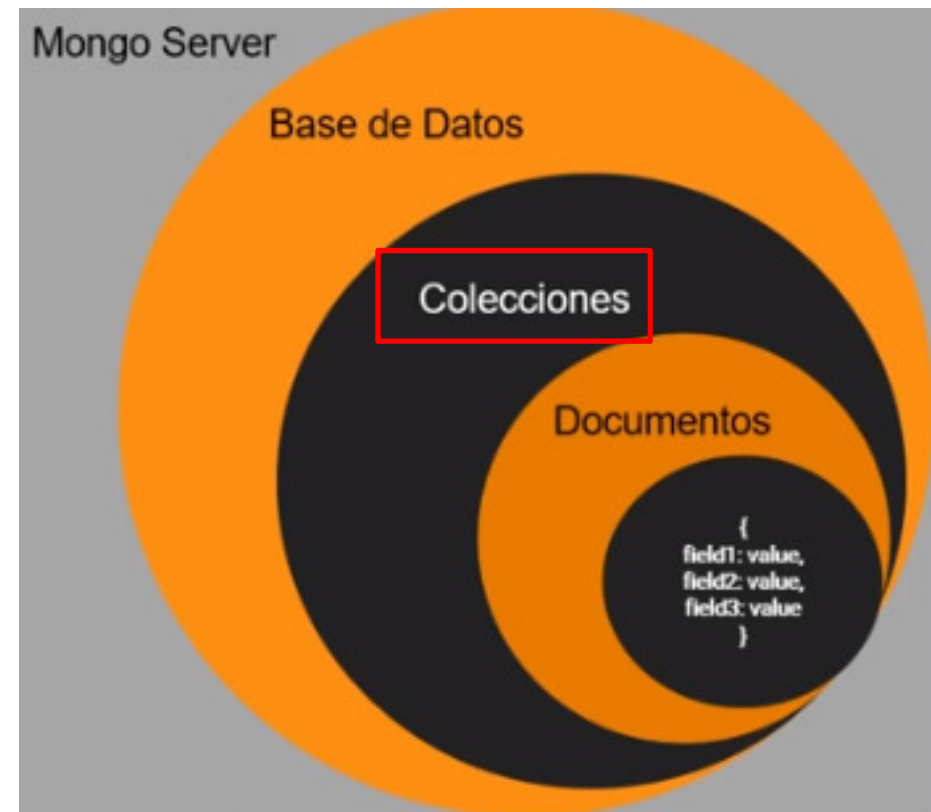


mongoDB

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

## 5.- ¿Cómo funciona? (III)

- **Definimos un colección:**
  - Conjunto de documentos de la misma entidad.  
Ejemplos: Todos los Usuarios, Personas, Productos, Aeropuertos, etc.





mongoDB

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

## 5.- ¿Cómo funciona? (IV)

- Las consultas se hacen pasando objetos JSON como parámetro.
- Ejecución en consola, consultas en JavaScript.
- Ejemplo: Buscar todos los clientes que se llamen Pedro:
  - `db.Clientes.find({Nombre:"Pedro"})`;
- Se puede trabajar con variables, funciones o bucles.



mongoDB

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

## 6.- Usando MongoDB

- 1) Abrir el servidor de MongoDB (mongod.service):
  - `sudo systemctl start mongod.service`
- 2) Abrir el cliente de MongoDB:
  - `mongo -u admin -p password --authenticationDatabase admin`





mongoDB

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

## 7.- Crear una BD

- Comando **use**: Crea una BD.
  - use <nombre\_BD>
- Comando **db**: Indica la BD en donde te encuentras.
- Comando **show dbs**: Muestra el nombre de las BD que he creado.

```
alicia@rosetta:~$ mongo -u admin -p alicia --authenticationDatabase admin
MongoDB shell version v3.6.9
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("75dd7545-543d-48d3-82c9-48c94a5eb75b") }
MongoDB server version: 3.6.9
Server has startup warnings:
2018-12-26T18:58:43.014+0000 I STORAGE [initandlisten]
2018-12-26T18:58:43.014+0000 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the
XFS file engine
2018-12-26T18:58:43.014+0000 I STORAGE [initandlisten] **           See http://dochub.mongodb.org/core/prodnotes-filesystem
> db
test
> use mitienda
switched to db mitienda
> db
mitienda
> db.mitienda.insert({nregistro: 1, DNI: "12345678", nombre: "Maria", apellidos: "Ruiz Martin", pais: "España"})
WriteResult({ "nInserted" : 1 })
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
mitienda   0.000GB
> 
```



mongoDB

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

## 8.- Crear una colección e insertar un documento

- Crear colección (implícita):  
`db.nombre_coleccion.insert({  
 campo1: "info1",  
 campo2: "info2",  
 campo3: "info3"  
})`

\*Los nombres de los campos se pueden o no poner entre comillas dobles

Ejemplo:

```
> db.usuarios.insert({nregistro: 1, DNI: "12345678", nombre: "Maria", apellidos: "Ruiz Martin", pais: "España"})
WriteResult({ "nInserted" : 1 })
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
mitienda   0.000GB
> █
```

\*La colección (tabla) usuarios está insertada en la BD mitienda



mongoDB

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

## 8.- Crear una colección e insertar un documento (II)

- Crear colección (explícita):

```
db.createCollection("nombreColeccion")
```

- Mostrar colecciones:

```
show collections
```

```
> db.createCollection("productos")
{ "ok" : 1 }
> show collections
productos
usuarios
> █
```

- Borrar una colección: `db.nombreColeccion.drop()`

```
> show collections
productos
usuarios
> db.productos.drop()
true
> show collections
usuarios
>
```



mongoDB

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

## 9.- Borrar la BD

- 1) Asegurarnos que estamos en la BD que vamos a borrar: use mitienda
- 2) Borrar la BD: db.dropDatabase()
- 3) Ver los cambios: show dbs

```
> use mitienda
switched to db mitienda
> db.dropDatabase()
{ "dropped" : "mitienda", "ok" : 1 }
> show dbs
admin    0.000GB
config  0.000GB
local    0.000GB
>
```



mongoDB

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

# 10.- Consultar elementos

- `db.productos.find().pretty()`

\*Pretty: función para mostrar formato legible

```
> db.productos.insert({id: 1, nombre:"camiseta", precio: 15.0, stock: 5})
WriteResult({ "nInserted" : 1 })
> db.productos.insert({id: 2, nombre:"vestido", precio: 30.0, stock: 7})
WriteResult({ "nInserted" : 1 })
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c26797e8dc94faab1cd51d7"),
  "id" : 1,
  "nombre" : "camiseta",
  "precio" : 15,
  "stock" : 5
}
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 30,
  "stock" : 7
}
>
```

- Sin pretty

```
> db.productos.find()
{ "_id" : ObjectId("5c26797e8dc94faab1cd51d7"), "id" : 1, "nombre" : "camiseta", "precio" : 15, "stock" : 5 }
{ "_id" : ObjectId("5c2679828dc94faab1cd51d8"), "id" : 2, "nombre" : "vestido", "precio" : 30, "stock" : 7 }
>
```



mongoDB

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

# 10.- Consultar elementos (II)

- Colección USUARIO

```
> db.usuarios.insert({nregistro: 1, DNI: "12345678", nombre: "Maria", apellidos: "Ruiz Martin", pais: "España"})
WriteResult({ "nInserted" : 1 })
> db.usuarios.insert({nregistro: 2, DNI: "23456781", nombre: "Pepe", apellidos: "Martin Gonzalez", pais: "España"})
WriteResult({ "nInserted" : 1 })
> db.usuarios.insert({nregistro: 3, DNI: "34567812", nombre: "Manuel", apellidos: "Varo Ortiz", pais: "Francia"})
WriteResult({ "nInserted" : 1 })
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c26797e8dc94faab1cd51d7"),
  "id" : 1,
  "nombre" : "camiseta",
  "precio" : 15,
  "stock" : 5
}
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 30,
  "stock" : 7
}
```



mongoDB

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

# 11.- Editar un documento

- `db.productos.update({  
    id: 1  
}, {  
    $set: {'precio': 100.0}  
})`

Registros que cumplan la condición

Campos con el nuevo valor

```
> db.productos.update({id:1},{ $set : {'precio':100.0}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c26797e8dc94faab1cd51d7"),
  "id" : 1,
  "nombre" : "camiseta",
  "precio" : 100,
  "stock" : 5
}
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 30,
  "stock" : 7
}
>
```

ACTUALIZA  
SOLO UN  
DOCUMENTO



mongoDB

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

# 11.- Editar un documento (II)

```
• db.productos.update({  
    id: 1  
}, {  
    $set: {'precio': 100.0}  
},  
{multi:true}  
)
```

Registros que cumplan la condición

Campos con el nuevo valor

ACTUALIZA TODOS LOS  
DOCUMENTOS QUE  
CUMPLEN CON LA  
CONDICIÓN





mongoDB

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

# 11.- Editar un documento (III)

ACTUALIZA TODOS LOS DOCUMENTOS QUE CUMPLEN CON LA CONDICIÓN:  
Inserto un nuevo documento

```
> db.productos.insert({id: 3, nombre:"camiseta M", precio: 30.0, stock: 3})
WriteResult({ "nInserted" : 1 })
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c26797e8dc94faab1cd51d7"),
  "id" : 1,
  "nombre" : "camiseta",
  "precio" : 100,
  "stock" : 5
}
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 30,
  "stock" : 7
}
{
  "_id" : ObjectId("5c267e8c8dc94faab1cd51dc"),
  "id" : 3,
  "nombre" : "camiseta M",
  "precio" : 30,
  "stock" : 3
}
```



mongoDB

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

# 11.- Editar un documento (IV)

ACTUALIZA TODOS LOS DOCUMENTOS QUE CUMPLEN CON LA CONDICIÓN:  
Actualizo en todos los documentos que cumplen que precio: 30 al valor de 100

```
> db.productos.update({precio:30},{ $set : {'precio':100.0}}, {multi:true})
WriteResult({ "nMatched" : 2, "nUpserted" : 0, "nModified" : 2 })
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c26797e8dc94faab1cd51d7"),
  "id" : 1,
  "nombre" : "camiseta",
  "precio" : 100,
  "stock" : 5
}
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 100,
  "stock" : 7
}
{
  "_id" : ObjectId("5c267e8c8dc94faab1cd51dc"),
  "id" : 3,
  "nombre" : "camiseta M",
  "precio" : 100,
  "stock" : 3
}
>
```



mongoDB

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

## 12.- Eliminar un documento

- `db.productos.deleteOne({`

`id: 1`

Registros que cumplan la condición

`})`

```
> db.productos.deleteOne({id:1})
{ "acknowledged" : true, "deletedCount" : 1 }
> db.productos.find().pretty()
{
  "_id" : ObjectId("5c2679828dc94faab1cd51d8"),
  "id" : 2,
  "nombre" : "vestido",
  "precio" : 100,
  "stock" : 7
}
{
  "_id" : ObjectId("5c267e8c8dc94faab1cd51dc"),
  "id" : 3,
  "nombre" : "camiseta M",
  "precio" : 100,
  "stock" : 3
}
```



mongoDB

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

## 13.- Consultas

- `db.productos.find().pretty()`: hace una consulta a la colección que le precede (productos). Ejemplo:
- `db.productos.find({  
    "precio": 100.0  
})`

Mostrar registros que cumplan la condición

```
> db.productos.find({precio:100.0})  
{ "_id" : ObjectId("5c2679828dc94faab1cd51d8"), "id" : 2, "nombre" : "vestido", "precio" : 100, "stock" : 7 }  
{ "_id" : ObjectId("5c267e8c8dc94faab1cd51dc"), "id" : 3, "nombre" : "camiseta M", "precio" : 100, "stock" : 3 }  
>
```



mongoDB

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

## 13.- Consultas (II)

- `db.productos.find().pretty()`: hace una consulta a la colección que le precede (productos). Ejemplo:
- `db.productos.find({`  
    `“precio”: {$lt: 100.0}`  
    `})`

Mostrar registros que cumplan la condición

```
> db.productos.find({precio:{$lt:100.0}})
{ "_id" : ObjectId("5c2683048dc94faab1cd51dd"), "id" : 4, "nombre" : "camiseta L", "precio" : 30, "stock" : 3 }
>
```



mongoDB

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

## 13.- Consultas (III)

Operacion	Ejemplo
Igualdad	<code>{ "stock": 0 }</code>
Menor Que	<code>{ "valor": { \$lt: 15.0 } }</code>
Menor o Igual Que	<code>{ "valor": { \$lte: 16.0 } }</code>
Mayor Que	<code>{ "valor": { \$gt: 18.0 } }</code>
Mayor o Igual Que	<code>{ "valor": { \$gte: 16.0 } }</code>
No es Igual	<code>{ "valor": { \$ne: 0 } }</code>
AND	<code>{ {key1: value1, key2:value2} }</code>



mongoDB

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

## 13.- Consultas (IV)

Operacion	Ejemplo
OR	<pre>{   \$or: [     {key1: value1}, {key2:value2}   ] }</pre>
AND + OR	<pre>{   key1: value1,   \$or: [ { key2: {\$lt: value2}, {key3: value3} } ] }</pre>



mongoDB

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

## 13.- Consultas (V)

- `db.productos.find().sort({valor:1})`: Devuelve la consulta ordenada:
  - Si valor vale 1: De menor a mayor.
  - Si valor vale -1: De mayor a menor.
- Ejemplo:

```
> db.productos.find().sort({id:-1})
{ "_id" : ObjectId("5c2683048dc94faab1cd51dd"), "id" : 4, "nombre" : "camiseta L", "precio" : 30, "stock" : 3 }
{ "_id" : ObjectId("5c267e8c8dc94faab1cd51dc"), "id" : 3, "nombre" : "camiseta M", "precio" : 100, "stock" : 3 }
{ "_id" : ObjectId("5c2679828dc94faab1cd51d8"), "id" : 2, "nombre" : "vestido", "precio" : 100, "stock" : 7 }
> 
```





mongoDB

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

## 13.- Consultas (VI)

- Mostrar en la consulta determinados campos:
- `db.productos.find({}, {nombre:1, _id:0})`
- Ejemplo:

```
> db.productos.find({}, {nombre:1, _id:0})
{ "nombre" : "vestido" }
{ "nombre" : "camiseta M" }
{ "nombre" : "camiseta L" }
>
```

- `db.productos.find({nregistro:{$ne:1}}, {nombre:1, _id:0})`

```
> db.usuarios.find()
{ "_id" : ObjectId("5c267ace8dc94faab1cd51d9"), "nregistro" : 1, "DNI" : "12345678", "nombre" : "Maria", "apellidos" : "Ruiz Martin", "pais" : "España" }
{ "_id" : ObjectId("5c267aec8dc94faab1cd51da"), "nregistro" : 2, "DNI" : "23456781", "nombre" : "Pepe", "apellidos" : "Martin Gonzalez", "pais" : "España" }
{ "_id" : ObjectId("5c267b128dc94faab1cd51db"), "nregistro" : 3, "DNI" : "34567812", "nombre" : "Manuel", "apellidos" : "Varo Ortiz", "pais" : "Francia" }
> db.usuarios.find({"nregistro":{$ne: 1}}, {nombre:1, _id:0})
{ "nombre" : "Pepe" }
{ "nombre" : "Manuel" }
>
```



mongoDB

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

- Tarea 2: Creación y manejo de bases de datos con MongoDB