

BD ORIENTADAS A DOCUMENTOS





- Este tipo de BD almacena la información como un documento.
- La precursora fue Lotus Notes.
- Utilizan estructuras simples como JSON o XML y donde se utiliza una clave única para cada registro.
- Buenas en modelado de datos natural
- Amigable al programador
- Orientadas a la web: CRUD
- Algunos ejemplos de este tipo son:
 - **MongoDB o CouchDB.**



EJEMPLO XML

```
<artist>
  <artistname>Iron Maiden</artistname>
  <albums>
    <album>
      <albumname>The Book of Souls</albumname>
      <datereleased>2015</datereleased>
      <genre>Hard Rock</genre>
    </album>
    <album>
      <albumname>Killers</albumname>
      <datereleased>1981</datereleased>
      <genre>Hard Rock</genre>
    </album>
    <album>
      <albumname>Powerslave</albumname>
      <datereleased>1984</datereleased>
      <genre>Hard Rock</genre>
    </album>
    <album>
      <albumname>Somewhere in Time</albumname>
      <datereleased>1986</datereleased>
      <genre>Hard Rock</genre>
    </album>
  </albums>
</artist>
```

EJEMPLO JSON

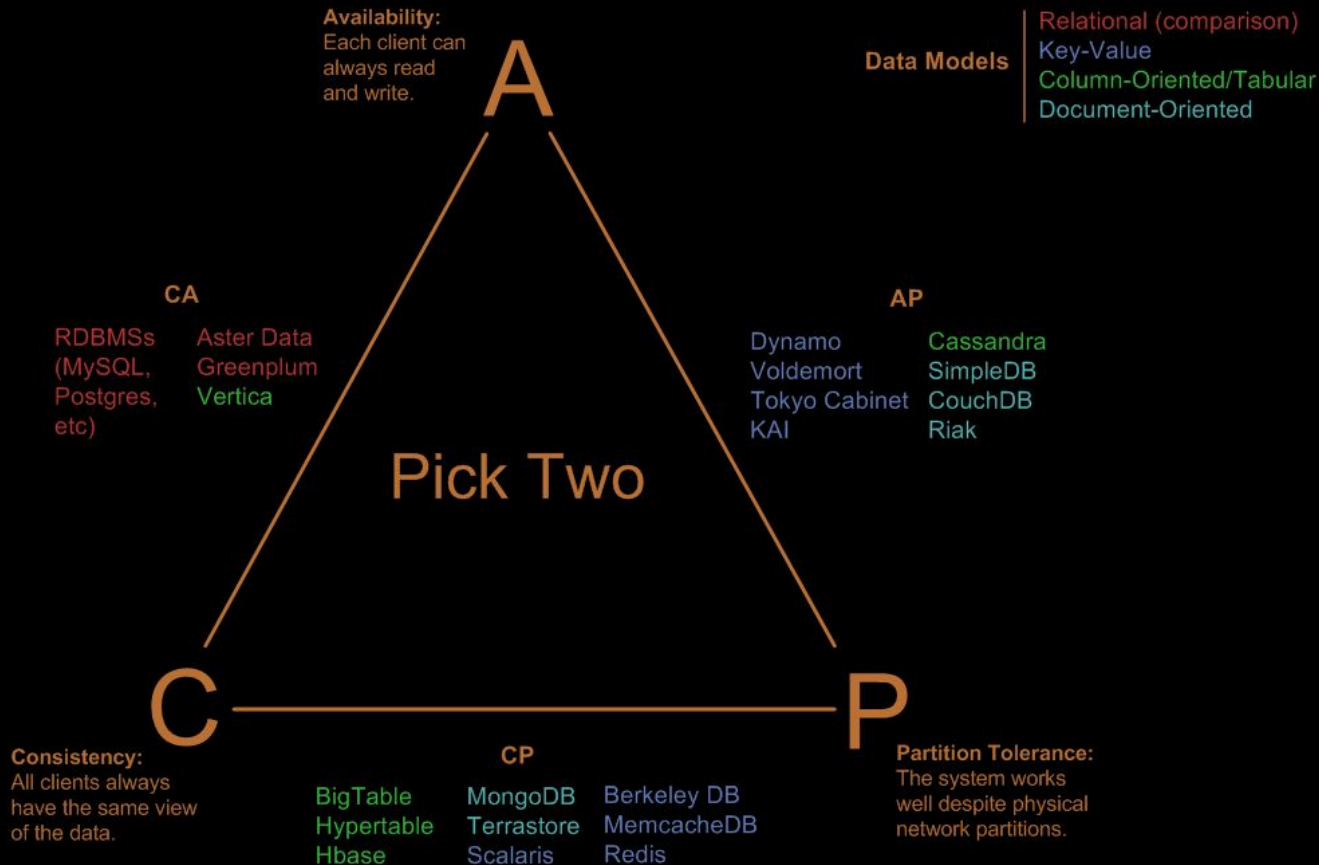
```
{
  '_id' : 1,
  'artistName' : { 'Iron Maiden' },
  'albums' : [
    {
      'albumname' : 'The Book of Souls',
      'datereleased' : 2015,
      'genre' : 'Hard Rock'
    }, {
      'albumname' : 'Killers',
      'datereleased' : 1981,
      'genre' : 'Hard Rock'
    }, {
      'albumname' : 'Powerslave',
      'datereleased' : 1984,
      'genre' : 'Hard Rock'
    }, {
      'albumname' : 'Somewhere in Time',
      'datereleased' : 1986,
      'genre' : 'Hard Rock'
    }
  ]
}
```

EJEMPLOS DOCUMENTOS















```
{  
  Nombre:"Pepe",  
  Dirección:"Plaza Mayor 5",  
  Profesión:"Panadero"  
}
```

```
{  
  Nombre:"Juliana",  
  Dirección:"Gran Vía 15",  
  Hijos:[  
    {Nombre:"Miguel", Edad:10},  
    {Nombre:"Jacinta", Edad:8},  
    {Nombre:"Sara", Edad:5},  
    {Nombre:"Elena", Edad:2}  
  ]  
}
```

Visual Guide to NoSQL Systems



DB-ENGINES

Rank			DBMS	Database Model	Score		
Apr 2024	Mar 2024	Apr 2023			Apr 2024	Mar 2024	Apr 2023
1.	1.	1.	MongoDB 	Document, Multi-model 	423.96	-0.57	-17.93
2.	2.	2.	Amazon DynamoDB 	Multi-model 	77.57	-0.15	+0.12
3.	3.	3.	Databricks 	Multi-model 	76.33	+1.99	+15.36
4.	4.	4.	Microsoft Azure Cosmos DB 	Multi-model 	29.85	-0.54	-5.23
5.	5.	5.	Couchbase 	Document, Multi-model 	18.46	-0.69	-5.30
6.	6.	6.	Firebase Realtime Database	Document	15.00	-0.07	-3.22
7.	7.	7.	CouchDB	Document, Multi-model 	10.26	-1.47	-4.36
8.	8.	8.	Google Cloud Firestore	Document	8.96	-1.01	-2.13
9.	9.	 10.	Realm	Document	7.71	+0.01	-0.57
10.	10.	 9.	MarkLogic	Multi-model 	6.50	-0.57	-1.84

<https://db-engines.com/en/ranking>



- Administración de contenidos (Blogs, plataformas de vídeo)
- Catálogos (Aplicaciones de e-commerce)

¿CUÁNDO SON UTILIZADAS?



<https://openwebinars.net/blog/empresas-que-usan-mongodb/>



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

MONGODB



MONGODB

Pretende combinar lo mejor de los almacenes clave/valor, bases de datos de documentos y RDBMS

Hace uso de JSON y tiene su propio lenguaje de consultas

Implementada en C++

Usada por SourceForge, Bit.ly, Foursquare o GitHub

URL: <http://www.mongodb.org/>





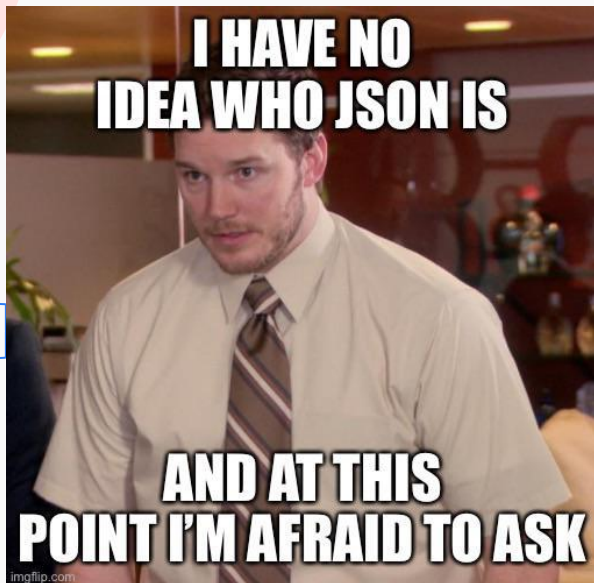
MongoDB (de la palabra en ingles “humongous” que significa enorme) es un sistema de base de datos NoSQL orientado a documentos.



MongoDB guarda estructuras de datos en documentos tipo **BSON** (*Binary JSON* - JSON Binario) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.



BSON –JSON ?



JSON (JavaScript Object Notation)

- Es un formato de archivo y de transferencia de información
- human-readable
- parejas clave-valor y colecciones
- Independiente del lenguaje, la mayoría de lenguajes provee librerías para generar y leer información en formato JSON
- Los archivos con este formato tienen extensión `.json`

MODELO DE DATOS – RDBMS VS MONGODB

RDBMS	MongoDB
Base de datos	Base de datos
Tabla	Colección
Índice	Índice
Fila	Documento JSON
Columna	Campo del documento
Join	Documentos embebidos y búsqueda



CARACTERÍSTICAS PRINCIPALES

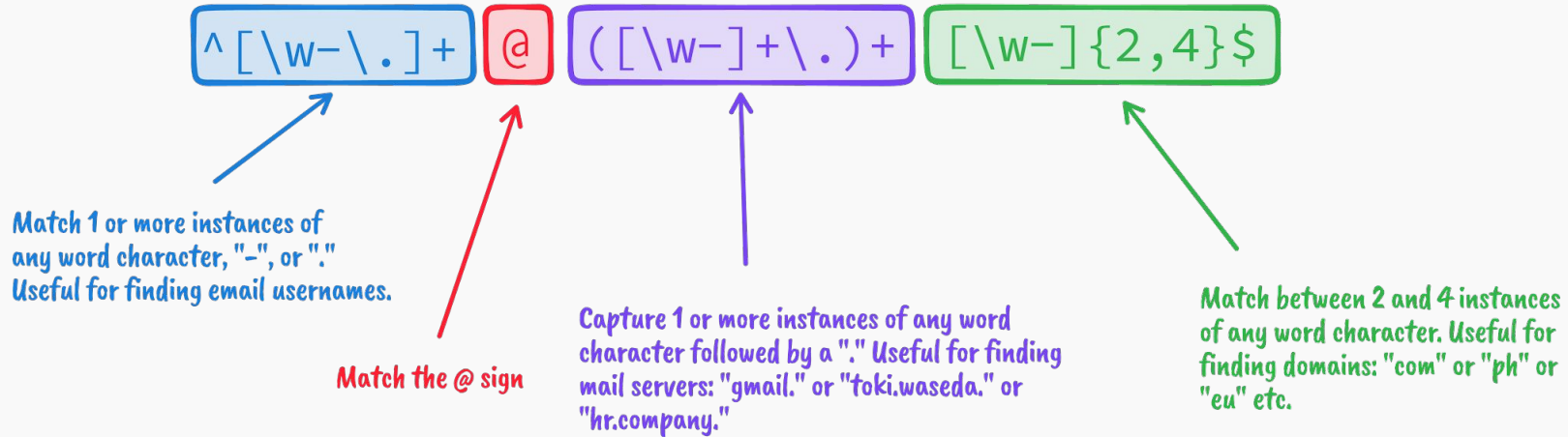
Consultas Ad-hoc

- MongoDB soporta la búsqueda por campos, consultas de rangos y expresiones regulares.
- Las consultas pueden devolver un campo específico del documento pero también puede ser una función JavaScript definida por el usuario.

Indexación

- Cualquier campo en un documento de MongoDB puede ser indexado, al igual que es posible hacer índices secundarios.
- El concepto de índices en MongoDB es similar a los encontrados en base de datos relacionales.

Expresiones Regulares



- Una expresión regular es una secuencia de caracteres que define un patrón de búsqueda.
- Es una herramienta utilizada por los programadores para buscar, validar y manipular texto de manera eficiente.
- Permite realizar tareas como la búsqueda de coincidencias de caracteres específicos, la validación de formatos y la extracción de información.
- Es útil en la programación para realizar operaciones complejas en texto de forma rápida y precisa.

MANIPULACIÓN DE DATOS: COLECCIONES Y DOCUMENTOS

- MongoDB guarda la estructura de los datos en documentos tipo JSON.
- Los elementos de los datos son llamados **documentos** y se guardan en **colecciones**.
- Una colección puede tener un número indeterminado de documentos.
- Las colecciones son como tablas y los documentos como filas.
- Cada documento en una colección puede tener diferentes campos.
- La estructura de un documento es simple y está compuesta por “key-value pairs”.
- Como valor se pueden usar números, cadenas o datos binarios como imágenes o cualquier otro “key-value pairs”.



object id

- `_id` Field: Cada objeto que se inserta en una base de datos MongoDB obtiene un campo `_id` generado automáticamente.
- Es único para cada documento insertado en la colección.

`ObjectId(' 5b7d297cc718bc133212aa94')`

5b7d297c

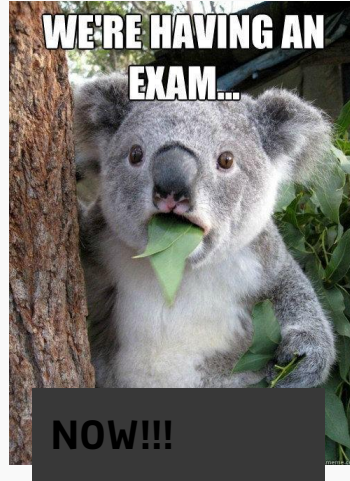
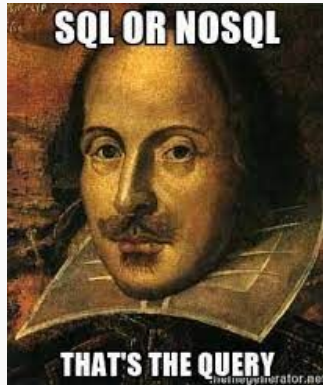
UNIX Timestamp
4 Bytes

c718bc1332

Random Value
5 Bytes

12aa94

Count
3 bytes



QUIZ MODELOS NOSQL

PRÁCTICA MONGODB INSTRUCCIONES BÁSICAS [LINK]



CREAR UN BASE DE DATOS

Relacional SQL

```
CREATE DATABASE  
base_datos;
```

```
> db //muestra la BD actual
```

MongoDB

```
use base_datos;
```



Nombre de la
base de datos

INSERTAR UN DOCUMENTO EN UNA COLECCIÓN

MongoDB

Relacional SQL

TABLA + INSERT

```
CREATE TABLE users (  
  id int primary key,  
  nombre varchar(255) not null,
```

**Poco flexible,
Ligado a la estructura!**

```
);
```

```
INSERT INTO users(id,nombre, apellido, edad, estatura)  
VALUES (1,"Maria", "Fernandez", 38, 1.72)
```

```
usuario = { nombre : "Monica", apellido : "Rojas",  
edad : 43, estatura : 1.60 };
```

Colección

db.users.insert(usuario);

BD actual

Documento

Cada uno inserte un documento con sus datos. Incluyendo el género: femenino o masculino. Tenga en cuenta la estructura y nombres de las claves actualmente de los documentos en la colección.

SELECCIONAR

Relacional SQL

```
SELECT * FROM users;
```

MongoDB

Colección



```
db.users.find ();
```



BD actual



Operador

CONEXIÓN MONGODB CON MONGODB COMPASS



Relacional SQL

```
SELECT top 1  
FROM users;
```

Identificador único autogenerado

**¿Con base en qué se
genera el Identificador
Único?**

MongoDB

```
var usuario = db.users.findOne();
```

BD actual

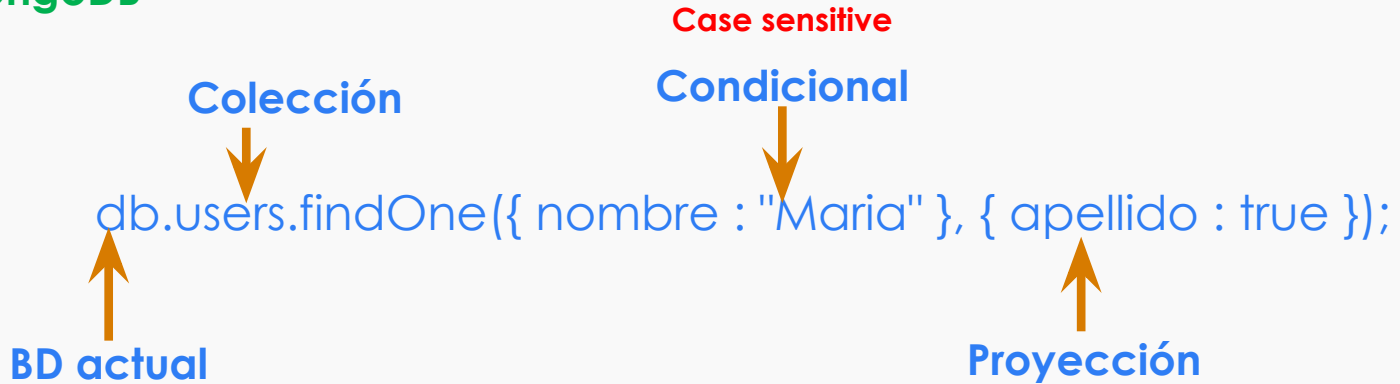
Colección

Operador

➔
{ "_id" : ObjectId("59c3331d45953192e2928d50"),
 nombre: "Maria", apellido: "Fernandez", edad: 38,
 estatura: 1.72
}

SELECT top 1 apellido FROM USERS WHERE nombre = "Maria"

MongoDB



```
> db.usuarios.findOne({nombre:"Maria"}, {apellido:true})
{ "_id" : ObjectId("5f6a5f44a5f0c5bd29483fa8"), "apellido" : "Fernandez" }
>
```

MONGODB COMPASS

base_datos.users

DOCUMENTS 10

STORAGE SIZE 20.5KB

AVG. SIZE 94B

INDEXES 1

Documents

Aggregations

Schema

Explain Plan

Indexes

Validation

FILTER { nombre : "Maria" }

PROJECT { apellido : true }

SORT { field: -1 } or [['field', -1]]

COLLATION { locale: 'simple' }

MAX TIME MS 60000

SKIP 0

LIMIT 0

OPTIONS

FIND

VIEW

Displaying documents 1 - 1 of 1

```
_id: ObjectId("6234b094efe50c37f9da527b")
apellido: "Andrade"
```

SELECT * FROM USERS WHERE nombre <> "Maria";

MongoDB

Colección

Operador
desigualdad

var usuario = db.users.find({nombre: {\$ne : "Maria"}});

BD actual

Operador

```
> db.usuarios.find({nombre: {$ne : "Maria"}});
{ "_id" : ObjectId("5f6a5f05a5f0c5bd29483fa7"), "nombre" : "Monica", "apellido" : "Rojas", "edad" : 43, "estatura" : 1.6 }
>
```

base_datos.users

DOCUMENTS 10

STORAGE SIZE
20.5KB

AVG. SIZE
94B

INDEXES

Documents

Aggregations

Schema

Explain Plan

Indexes

Validation

FILTER {nombre: {\$ne : "Maria"}}

▼ OPTIONS

FIND

PROJECT { field: 0 }

SORT { field: -1 } or [['field', -1]]

MAX TIME MS 60000

COLLATION { Locale: 'simple' }

SKIP 0

LIMIT 0

ADD DATA ▼



VIEW



Displaying documents 1 - 9 of 9 <

```
_id: ObjectId("6234b094efe50c37f9da527a")
nombre: "Monica"
apellido: "Rojas"
genero: "femenino"
edad: 43
estatura: "1.60"
```

```
_id: ObjectId("6234b094efe50c37f9da527c")
nombre: "Luis"
apellido: "Ruiz"
edad: 48
estatura: "1.70"
```

OPERADORES

Operador	Expresión
Igual	\$eq
Diferente	\$ne
Mayor que	\$gt
Mayor o igual que	\$gte
Menor que	\$lt
Menor o igual que	\$lte
Existencia en array	\$in
Inexistencia en array	\$nin

CONSULTA OPERADOR AND

```
SELECT * FROM users  
WHERE nombre = 'Maria' AND apellido = 'Andrade';
```

MongoDB

Colección



```
db.users.find({nombre:"Maria", apellido: "Andrade"});
```



BD actual

MONGODB COMPASS

FILTER

{nombre:"Maria", apellido: "Andrade"}

OPTIONS

PROJECT

{ field: 0 }

SORT

{ field: -1 } or [['field', -1]]

MAX TIME MS

60000

COLLATION

{ locale: 'simple' }

SKIP

0

LIMIT

0

ADD DATA

VIEW

Displaying documents 1 - 1 of 1

```
_id: ObjectId("6234b094efe50c37f9da527b")
nombre: "Maria"
apellido: "Andrade"
edad: 18
estatura: "1.50"
```

CONSULTA OPERADOR OR

```
SELECT * FROM users  
WHERE nombre = 'Maria' OR apellido = 'Cardenas'
```

MongoDB

Colección



```
db.users.find({$or:[{nombre:"Maria"},{apellido:"Cardenas"}]});
```



BD actual

MONGODB COMPASS

FILTER

`{ $or: [{ nombre: "Maria" }, { apellido: "Cardenas" }] }`

PROJECT

`{ field: 0 }`

SORT

`{ field: -1 } or [['field', -1]]`

COLLATION

`{ locale: 'simple' }`

MAX TIME MS

60000

SKIP

0

LIMIT

0

OPTIONS

FIND

ADD DATA

VIEW

Displaying documents 1 - 2 of 2

```
_id: ObjectId("6234b094efe50c37f9da527b")
nombre: "Maria"
apellido: "Andrade"
edad: 18
estatura: "1.50"
```

```
_id: ObjectId("6234b094efe50c37f9da5281")
nombre: "Lucia"
apellido: "Cardenas"
edad: 24
estatura: "1.68"
```

CLÁUSULA ORDER BY

```
SELECT * FROM USERS ORDER BY nombre, edad ASC;
```

```
SELECT * FROM USERS ORDER BY nombre, edad DESC;
```

MongoDB

Colección



```
db.users.find().sort({"nombre":1, "edad":-1})
```



BD actual

1 para ordenar Ascendente
-1 para ordenar Descendente

MONGODB COMPASS

FILTER { field: 'value' }

▼ OPTIONS

FIND

RESET



PROJECT { field: 0 }

SORT [{"nombre":1, "edad":-1}]

MAX TIME MS 60000

COLLATION { locale: 'simple' }

SKIP 0

LIMIT 0

ADD DATA ▼



VIEW



Displaying documents 1 - 10 of 10



REFRESH

```
_id: ObjectId("6234b094efe50c37f9da5280")
nombre: "Alejandro"
apellido: "Sanchez"
edad: 10
estatura: "1.40"
```

```
_id: ObjectId("6234b094efe50c37f9da527f")
nombre: "Camila"
apellido: "Jurado"
genero: "femenino"
edad: 12
estatura: "1.50"
```

```
_id: ObjectId("6234b094efe50c37f9da5283")
nombre: "Diego"
```

LIKE: CONTIENE

SELECT * FROM TABLE users WHERE nombre LIKE '%Leon%'

db.users.find({nombre: /Leon/});

```
Atlas atlas-mrr86e-shard-0 [primary] users_infrati> db.users.find({nombre: /Leon/});
[
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e1"),
    nombre: 'Leonor',
    apellido: 'Restrepo',
    edad: 78
  },
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e2"),
    nombre: 'Leonardo',
    apellido: 'Jurado',
    edad: 44,
    estatura: '1.80'
  }
]
```

FILTER	{nombre: /Leon/}	OPTIONS	FIND
PROJECT	{ field: 0 }		
SORT	{ field: -1 } or [['field', -1]]	MAX TIME MS 60000	
COLLATION	{ locale: 'simple' }	SKIP 0 LIMIT 0	

LIKE: COMIENZA

SELECT * FROM TABLE users WHERE nombre LIKE 'L%'

db.users.find({nombre: /^L/});

```
Atlas atlas-mrr86e-shard-0 [primary] users_infrati> db.users.find({nombre: /^L/});
[
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e0"),
    nombre: 'Luis',
    apellido: 'Ruiz',
    edad: 48,
    estatura: '1.70'
  },
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e1"),
    nombre: 'Leonor',
    apellido: 'Restrepo',
    edad: 78
  },
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e2"),
    nombre: 'Leonardo',
    apellido: 'Jurado',
    edad: 44,
    estatura: '1.80'
  },
  {
    _id: ObjectId("6346dd3dc060d51f1b9c73e5"),
    nombre: 'Lucia',
    apellido: 'Cardenas',
    edad: 24,
    estatura: '1.60'
  }
]
```

FILTER	{nombre: /^L/}	OPTIONS	FIND
PROJECT	{ field: 0 }		
SORT	{ field: -1 } or [['field', -1]]	MAX TIME MS	60000
COLLATION	{ locale: 'simple' }	SKIP	0
		LIMIT	0

Like: termina

```
SELECT * FROM TABLE users WHERE nombre LIKE '%o'
```

```
db.users.find({nombre: /o$/});
```

FILTER	{nombre: /o\$/}	OPTIONS	FIND
PROJECT	{ field: 0 }		
SORT	{ field: -1 } or [['field', -1]]	MAX TIME MS 60000	
COLLATION	{ locale: 'simple' }	SKIP 0 LIMIT 0	

BETWEEN

```
SELECT * FROM TABLE users WHERE edad BETWEEN 20 AND 38;
```

```
db.users.find({edad: {$gte:20,$lte:38}});
```

```
SELECT * FROM TABLE users WHERE edad>=20 AND edad <38;
```

```
db.users.find({edad: {$gte:20,$lt:38}});
```

FILTER	<code>{edad: {\$gte:20,\$lte:38}}</code>	OPTIONS
PROJECT	<code>{ field: 0 }</code>	
SORT	<code>{ field: -1 } or [['field', -1]]</code>	MAX TIME MS 60000
COLLATION	<code>{ locale: 'simple' }</code>	SKIP 0 LIMIT 0

TALLER

1. Consulte todos los documentos de la colección users
2. Muestre la cantidad de documentos que hay en la colección users
3. Muestre los nombres de todos los usuarios
4. Muestre los nombres y apellidos de los users mayores de 18 años
5. Muestre los nombres y apellidos de las usuarias menores de 40 años
6. De cuántas manera diferentes escribieron el género de un usuario

REFERENCIAS

Cielen, D., Meysman, A., & Ali, M. (2016). Introducing Data Science: Big Data. Machine Learning and More, Using Python Tools. Manning, Shelter Island, US, 322.

MOOC BigData: Sistemas gestores de bases de datos orientados a documentos III
<https://www.youtube.com/watch?v=eYiebokW2hg>

Manual MongoDB
<https://docs.mongodb.com/manual/replication/>