



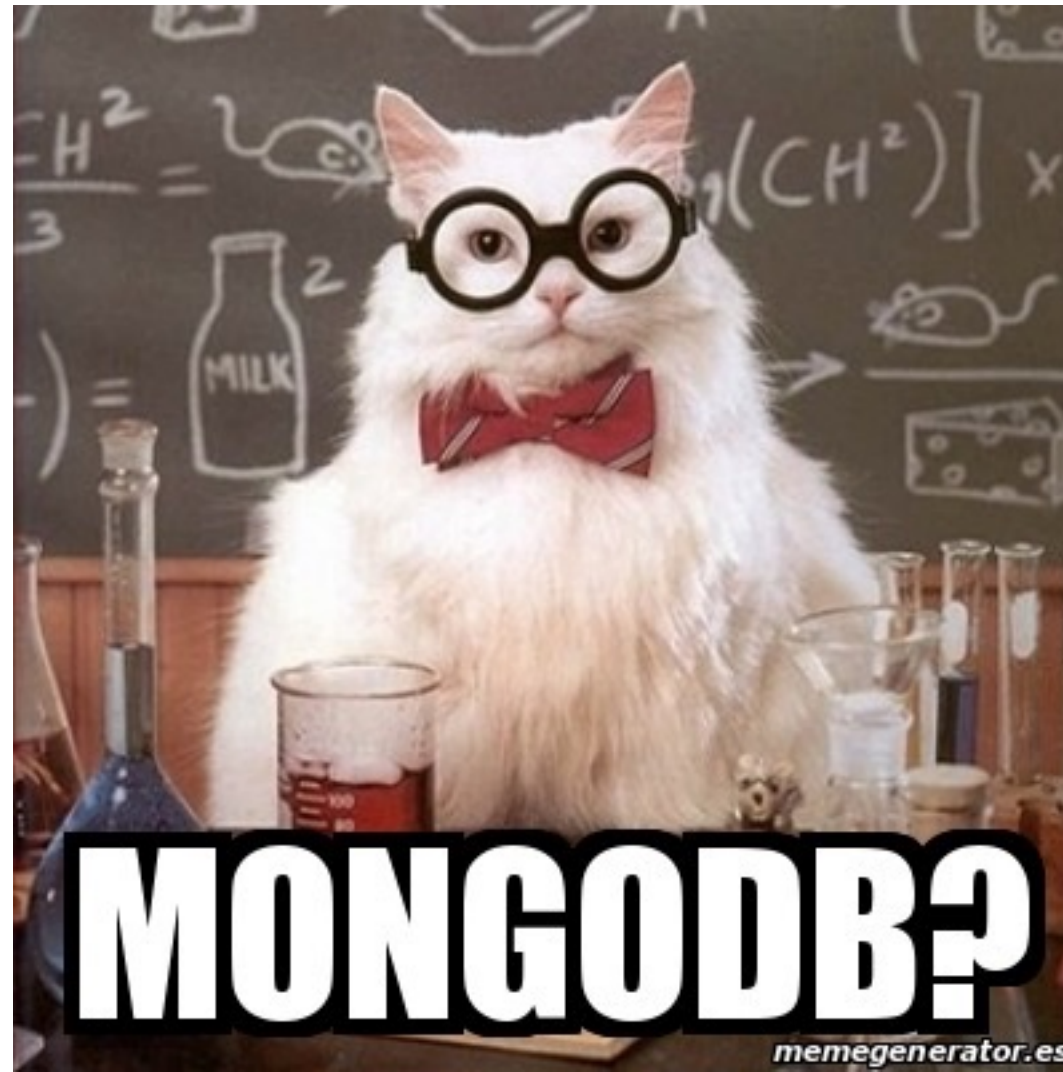
Primeros Pasos con *MongoDB*

Alejandro Romero me@alejo8591.co



MongoDB (from “*humongous*”) is scalable, high-performance, open source, schema-free, document-oriented database.

—*mongodb.org*



¿Quién utiliza *MongoDB*?



Check-in's



BOSCH

IoT: Big Data



github
SOCIAL CODING

Bug Tracking & Analytics



CMS

The
New York
Times

Image submission



Tracking Points



Imagen: <https://goo.gl/KjRXd1>



Imagen: <https://goo.gl/6b4LtM>

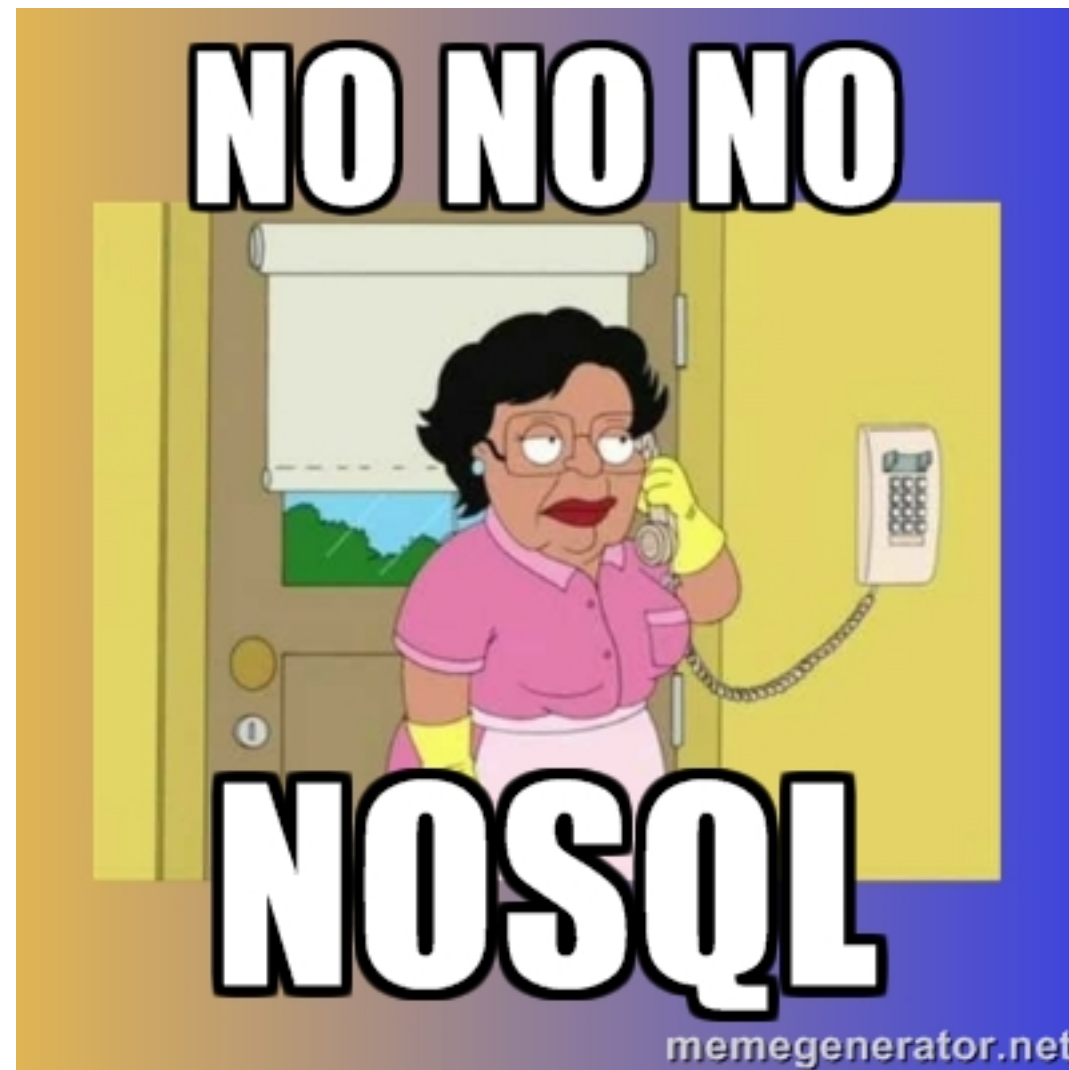


Imagen: <https://goo.gl/gQ9N3E>



Imagen: <https://goo.gl/ssZSDI>

¿Quién más utiliza *MongoDB*?

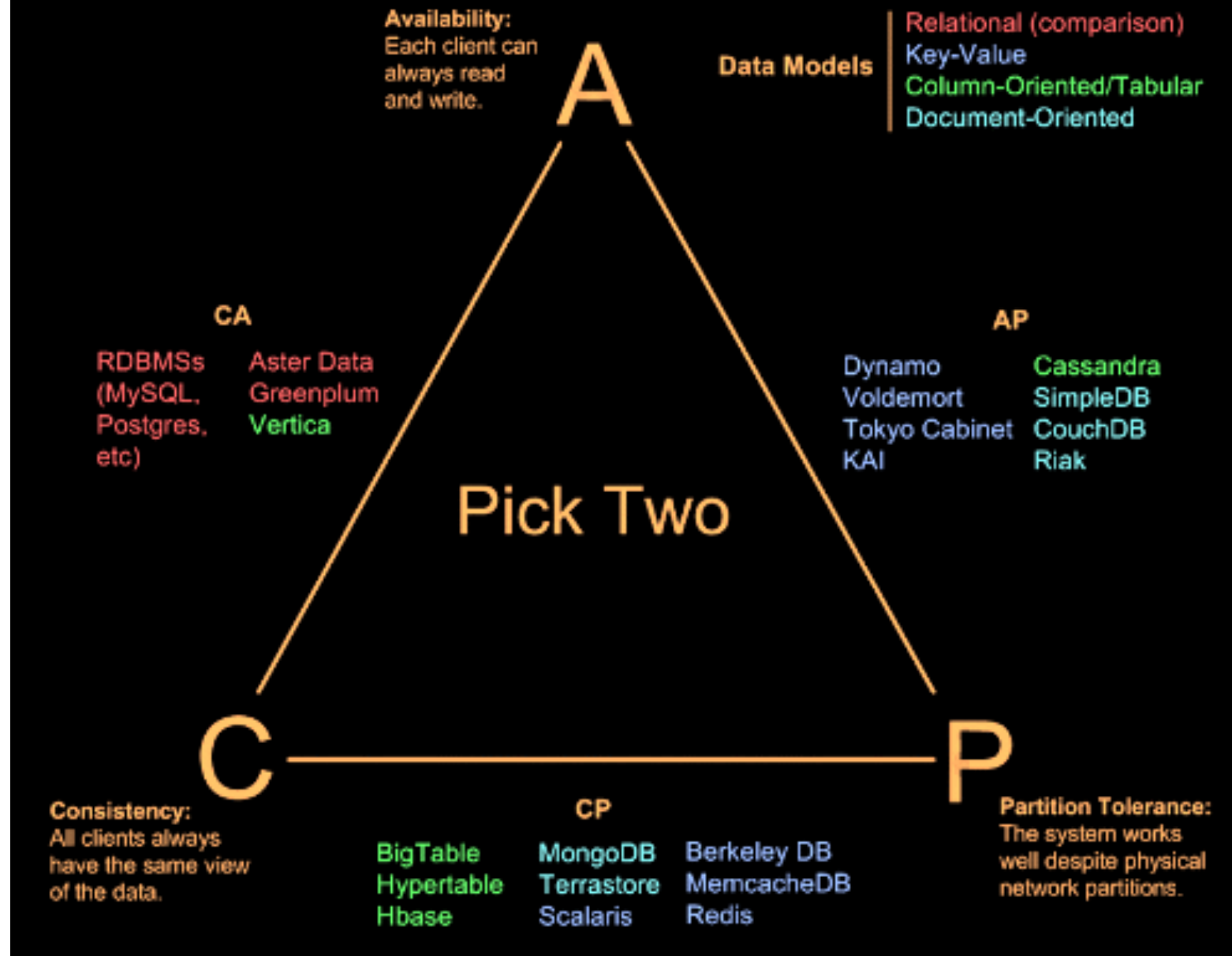


MongoDB es una base de
datos *NoSQL!!!*



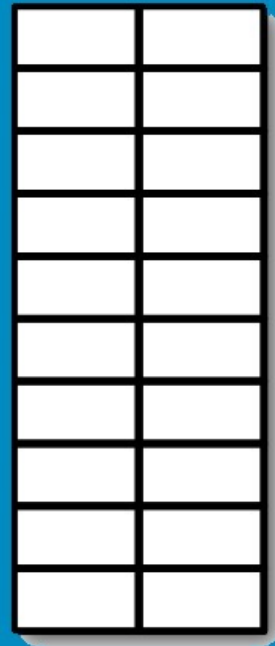
Filosofía

Visual Guide to NoSQL Systems



Filosofía

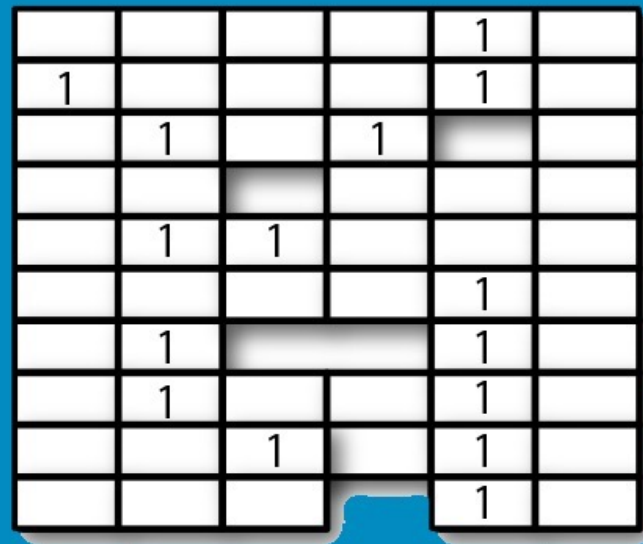
Key-Value



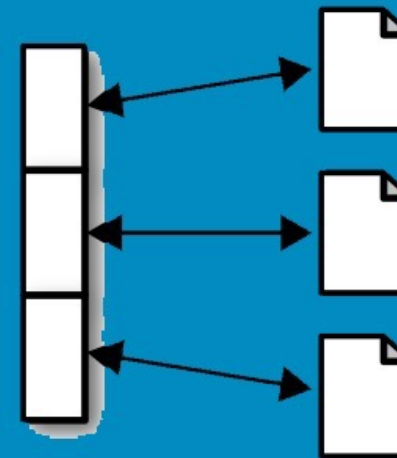
Graph DB



Column Family



Document



Tipos de NoSQL

Data Model	Performance	Scalabilty	Flexibility	Complexity	Functionality
Relational	variable	variable	low	moderate	relational algebra
Key—Value Stores	high	high	high	none	variable
Graph Database	variable	variable	high	high	graph theory
Document Store	high	variable (high)	high	low	variable (low)
Column Store	high	high	moderate	low	minimal

Clasificación por características



Ok!, Volvamos con ***MongoDB***



Características

MongoDB

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características *MongoDB*

- De propósito general
- **Escalabilidad**
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características ***MongoDB***

- De propósito general
- Escalabilidad
- **Replicación**
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características ***MongoDB***

- De propósito general
- Escalabilidad
- Replicación
- **Auto balanceo de carga**
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características ***MongoDB***

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- **Estándares como *GeoJSON***
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características ***MongoDB***

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- **No existen los “Joins”**
- Esquemas flexibles
- Aggregation Framework
- Map-Reduce

Características *MongoDB*

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- **Esquemas flexibles**
- Aggregation Framework
- Map-Reduce

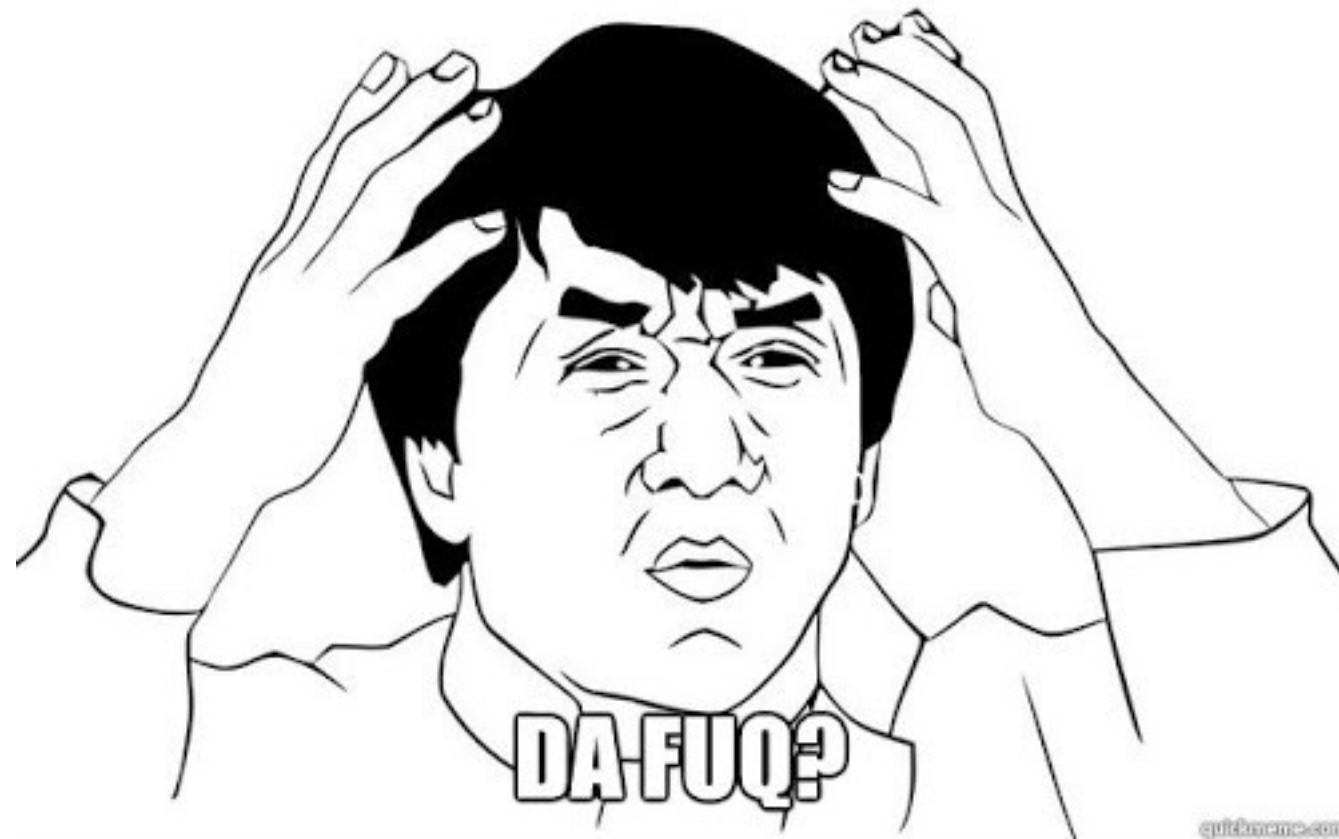
Características *MongoDB*

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- **Aggregation Framework**
- Map-Reduce

Características *MongoDB*

- De propósito general
- Escalabilidad
- Replicación
- Auto balanceo de carga
- Estándares como *GeoJSON*
- No existen los “*Joins*”
- Esquemas flexibles
- Aggregation Framework
- **Map-Reduce**

Características *MongoDB*



Terminología y Conceptos en *MongoDB*

SQL	MongoDB
Database	Database
Table	Collection
row	document
Column	Field
Table joins	embedded document and linking
Primary Key	Primary Key
Aggregation (group by)	Aggregation Mapping Chart

T y C en *MongoDB*

SQL Schema

MongoDB Schema

```
CREATE TABLE users (  
  id MEDIUMINT NOT NULL  
    AUTO_INCREMENT,  
  user_id Varchar(30),  
  age Number,  
  status char(1),  
  PRIMARY KEY (id)  
)
```

```
db.users.insert( {  
  user_id: "abc123",  
  age: 55,  
  status: "A"  
} )
```

T y C en *MongoDB*

SQL Schema

MongoDB Schema

```
ALTER TABLE users  
ADD join_date DATETIME
```

```
db.users.update(  
  { },  
  { $set: { join_date: new Date() } },  
  { multi: true }  
)
```

T y C en *MongoDB*

SQL Schema

MongoDB Schema

```
ALTER TABLE users  
DROP COLUMN join_date
```

```
db.users.update(  
  { },  
  { $unset: { join_date: "" } },  
  { multi: true }  
)
```

T y C en *MongoDB*

SQL Schema

MongoDB Schema

```
CREATE INDEX idx_user_id_asc  
ON users(user_id)
```

```
CREATE INDEX  
    idx_user_id_asc_age_desc  
ON users(user_id, age DESC)
```

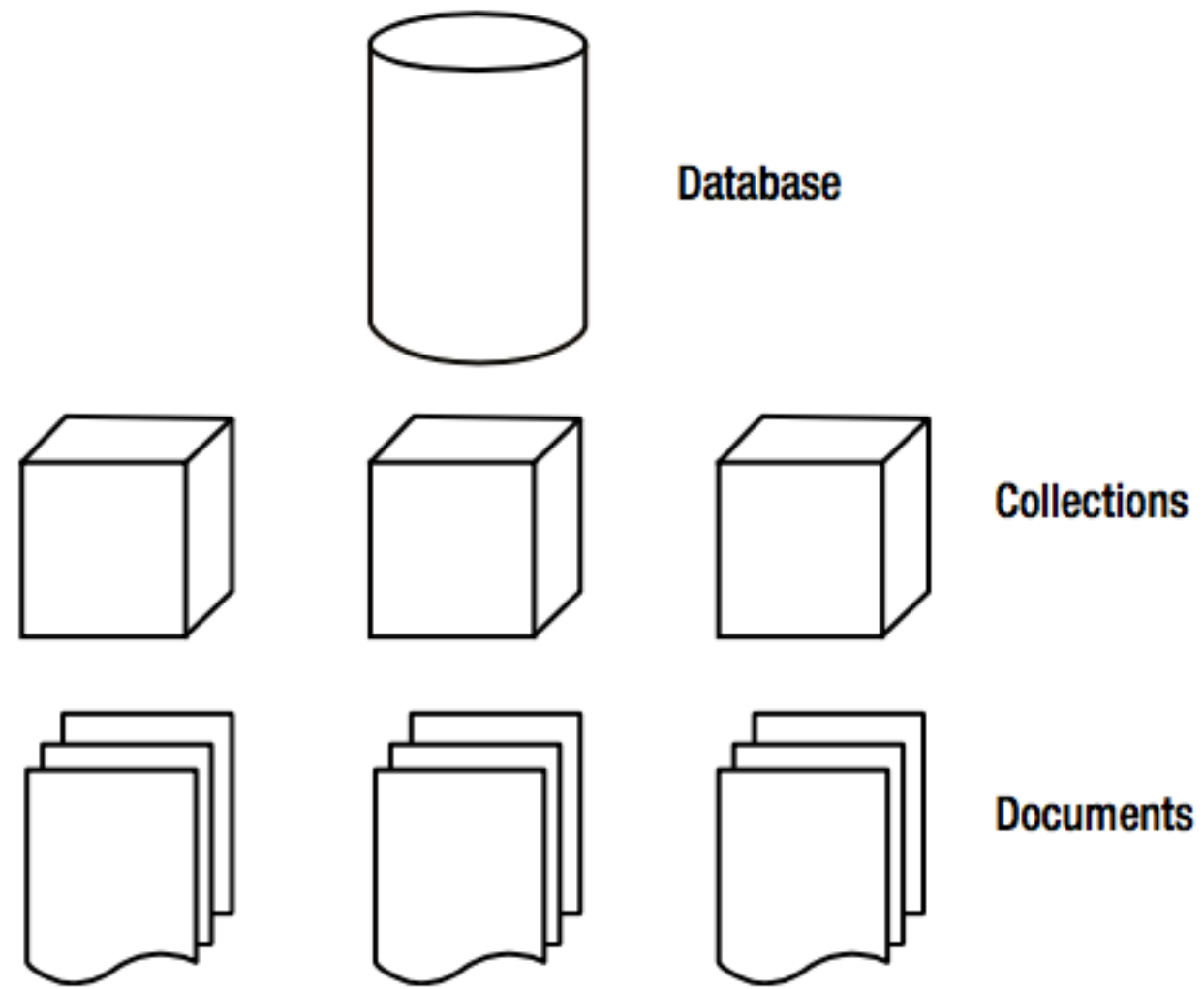
```
DROP TABLE users
```

```
db.users.createIndex( { user_id: 1 } )
```

```
db.users.createIndex( { user_id: 1, age: -1 } )
```

```
db.users.drop()
```

T y C en *MongoDB*

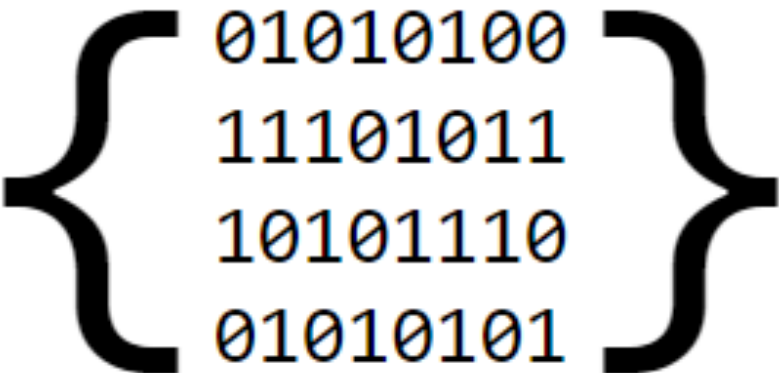


T y C en *MongoDB*

JSON

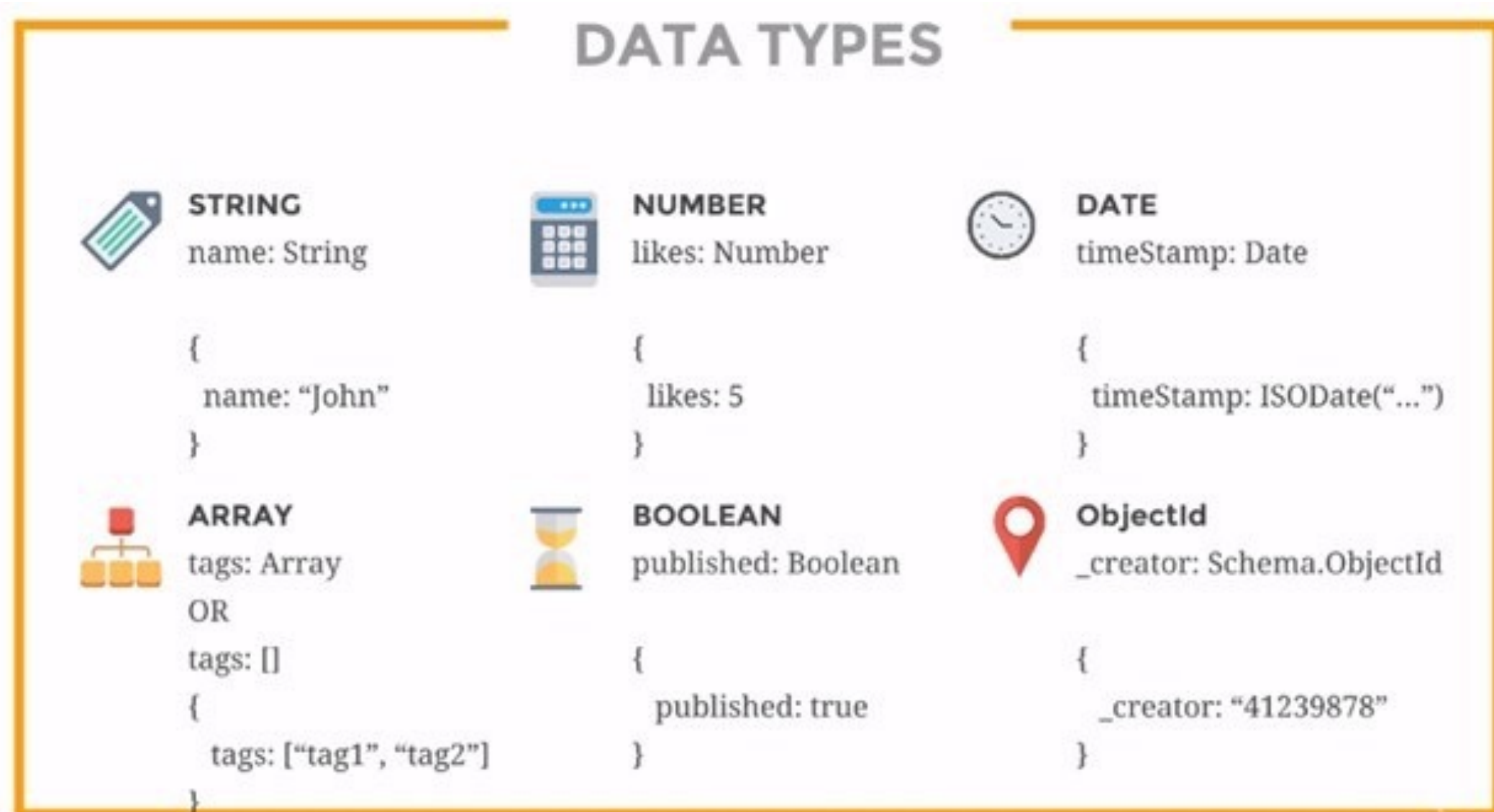
```
{  
  "array": [  
    1,  
    2,  
    3  
  ],  
  "boolean": true,  
  "null": null,  
  "number": 123,  
  "object": {  
    "a": "b",  
    "c": "d",  
    "e": "f"  
  },  
  "string": "Hello World"  
}
```

BSON



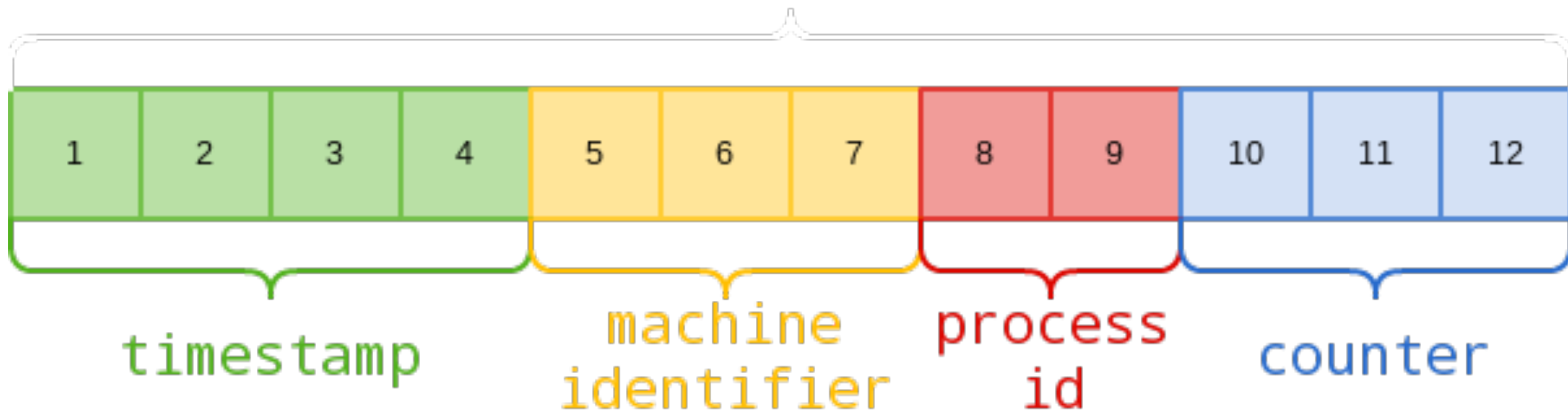
```
01010100  
11101011  
10101110  
01010101
```

T y C en *MongoDB*



T y C en *MongoDB*

id
ObjectId



T y C en *MongoDB*



T y C en *MongoDB*