# **MongoDB Fundamentos**

## **Motores NoSQL**

NoSQL es una nueva forma de diseñar soluciones de base de datos a escala de Internet. No es un producto o tecnología, sino un término que define un conjunto de tecnologías de base de datos que no se basan en los principios tradicionales de los motores Relacionales.

En los últimos años con el auge de las aplicaciones web se recopilan cada vez más datos y cada vez más usuarios desean acceder a estos datos al mismo tiempo, lo que significa que la escalabilidad y el rendimiento se han convertido en un gran reto para las bases de datos relacionales basadas en esquemas.

Existen varios sistemas de bases de datos de código abierto siendo los más populares Hypertable, Cassandra, MongoDB, DynamoDB, Hbase y Redis.

### **Características**

* Alta escalabilidad
* Fácil gestion y administración
* Bajo costo
* Modelos de datos flexibles

## **MongoDB**

Es una base de datos **orientada a documentos.** Esta diseñado para trabajar con documentos sin necesidad de tener columnas predefinidas o tipos de datos, lo que hace que el modelo de datos sea extremadamente flexible.

Estos documentos se almacenan en un formato llamado JSON binario (también conocido como BSON). Los documentos pueden tener diferentes esquemas, lo que significa que el esquema puede cambiar a medida que la aplicación evoluciona. MongoDB está construido para escalabilidad, rendimiento y alta disponibilidad.

### **¿Cómo es un documento JSON?**

Un documento JSON siempre comienza y termina con llaves. Los campos y sus valores están separados por comas, con el nombre de campo siendo siempre un string y el valor de cualquier tipo de data que va de string, number, date, array, otro documento JSON, etc.

No proporciona soporte para operaciones JOIN, pero permite al usuario almacenar todos los datos relevante juntos en un solo documento.

MongoDB no proporciona soporte para transacciones de la misma manera que SQL. Sin embargo, garantiza la atomicidad a nivel de documento ya que utiliza un operador de aislamiento para operaciones de escritura que afectan a varios documentos.

### **¿Por qué usar MongoDB?**

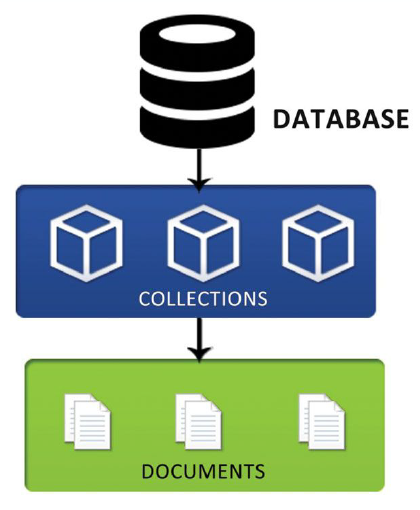
* Almacenamiento orientado a documentos.
* Índice sobre cualquier atributo.
* Replicación y alta disponibilidad.
* Auto-sharding
* Consultas enriquecidas.

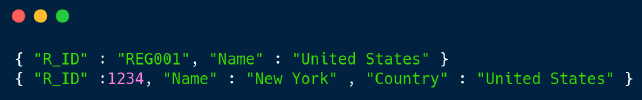
### **¿Dónde utilizar MongoDB?**

* Big Data
* Gestión de contenido y entrega
* Infraestructura móvil y social
* Gestión de datos de usuario
* Centro de datos

## **Modelado de Datos en MongoDB**

Una implementación de MongoDB puede tener muchas bases de datos. Cada base de datos es un conjunto de colecciones, las cuales no necesitan conocer con anterioridad el esquema (modelo de datos) y el mismo es **dinámico**. Cada colección puede tener múltiples documentos.

Esquema dinámico significa que los documentos dentro de la misma colección pueden tener conjuntos iguales o diferentes de campos o estructura, e incluso campos comunes pueden almacenar diferentes tipos de valores en los documentos.

Ejemplo:

Los documentos de una misma colección pueden tener esquemas completamente diferentes. Le corresponde a nuestra aplicación almacenar documentos en una colección particular juntos o para tener múltiples colecciones.

**Las colecciones son similares al concepto de tablas en RDBMS.**

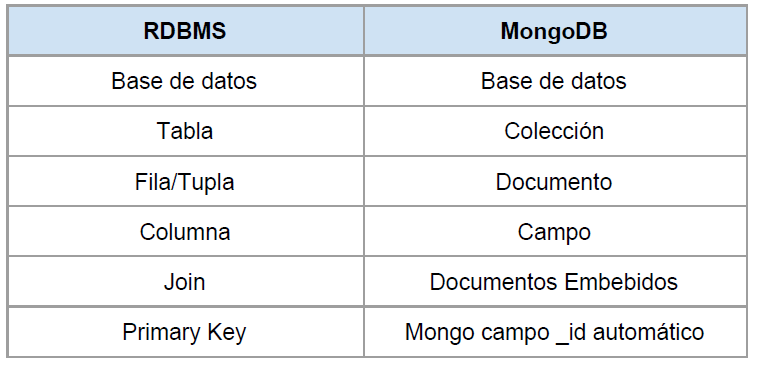
### **Identificador \_id**

Hemos visto que MongoDB almacena datos en documentos. Los documentos se componen de pares clave-valor. Una colección debe tener una clave que identifique de forma única cada documento. A esto nos referimos como identificador \_id en MongoDB. Si no se especifica un valor para una clave, se generará automáticamente un valor único por MongoDB. Este valor clave es inmutable y puede ser de cualquier tipo de datos, excepto los arrays.

### **Algunas consideraciones al diseñar el esquema en MongoDB**

* Diseña tu esquema de acuerdo a los requerimientos del usuario.
* Combine los objetos en un solo documento si los usará juntos. De lo contrario, sepárelos (pero asegúrese de que no haya necesidad de uniones).
* Duplique los datos (pero limitado).
* Optimice su esquema para los casos de uso más frecuentes.

## **RDBMS vs. MongoDB**



## **Componentes**

* Mongod (motor de base de datos)
* Mongo (cliente de línea de comandos)
* Mongo Compass (cliente gráfico)

Debemos ejecutar “mongod” para levantar el motor, el administrador de los datos. Cuando logremos levantarlo podemos conectarnos con “mongo” el cliente de línea de comandos para realizar operaciones en la base de datos.