

# NoSQL vs SQL






## Introducción a MongoDB

Acceso a Datos - 2º DAM

**¿Todas las aplicaciones necesitan bases de datos relacionales?**



# El mundo de los datos ha cambiado

-  Millones de usuarios simultáneos
-  Datos distribuidos globalmente
-  Volúmenes masivos (Big Data)
-  Estructuras de datos variables
-  Necesidad de respuesta inmediata

# Bases de Datos Relacionales (SQL)

## Características

- Tablas con filas y columnas
- Esquema fijo y predefinido
- Relaciones con claves foráneas
- Transacciones ACID
- Lenguaje SQL estándar

## Ejemplos

- MySQL
- PostgreSQL
- Oracle
- SQL Server

## Ejemplo: Tabla de Usuarios

| TABLA: usuarios |              |                    |         |        |
|-----------------|--------------|--------------------|---------|--------|
| id              | nombre       | email              | dept_id | activo |
| 1               | Ana García   | ana@empresa.com    | 1       | true   |
| 2               | Carlos López | carlos@empresa.com | 1       | true   |
| 3               | María Ruiz   | maria@empresa.com  | 2       | false  |

dept\_id —  FOREIGN KEY a tabla departamentos

## ¿Cuándo SQL puede ser limitante?

- ⚠ Esquemas muy cambiantes → Migraciones constantes
- ⚠ Datos heterogéneos → Muchos campos NULL
- ⚠ Escalado horizontal → Complejo y costoso
- ⚠ Datos anidados → Múltiples JOINS
- ⚠ Alta velocidad de escritura → Bloqueos

# NoSQL: "Not Only SQL"

Familia de bases de datos diseñadas para casos de uso específicos donde las bases relacionales no son la mejor opción.

**No es un reemplazo, es un complemento**



# Tipos de Bases de Datos NoSQL

## DOCUMENTOS

MongoDB

CouchDB

## CLAVE-VALOR

Redis

DynamoDB

Memcached

## COLUMNAS

Cassandra







HBase

## GRAFOS

# MongoDB - La BD Documental más Popular



## Ventajas:

-  Líder del mercado en BD documentales
-  Gran comunidad y documentación
-  Fácil de aprender viniendo de JSON
-  Escalable horizontalmente
-  Flexible: sin esquema fijo
-  Driver oficial para Java

# Traduciendo Conceptos

| SQL           | MongoDB            |
|---------------|--------------------|
| Base de datos | Base de datos      |
| Tabla         | Colección          |
| Fila          | Documento          |
| Columna       | Campo              |
| PRIMARY KEY   | _id                |
| JOIN          | Embedding/\$lookup |

# Anatomía de un Documento MongoDB

```
{
  "_id": ObjectId("507f1f77bcf86cd799439011"), // ← Identificador único automático
  "nombre": "Ana García",
  "email": "ana.garcia@empresa.com",
  "departamento": "IT",
  "activo": true,
  "fechaAlta": ISODate("2024-01-15"),
  "skills": ["Java", "Spring", "MongoDB"], // ← ¡Array embebido!
  "direccion": { // ← ¡Objeto anidado!
    "ciudad": "Madrid",
    "pais": "España"
  }
}
```

# El Mismo Usuario: SQL vs MongoDB

## SQL

```
TABLA usuarios:  
id=1, nombre="Ana", dept_id=1
```

```
TABLA skills:  
user_id=1, skill="Java"  
user_id=1, skill="Spring"
```

```
TABLA direcciones:  
user_id=1, ciudad="Madrid"
```

**3 tablas, 2 JOINS necesarios**

## MongoDB

```
{  
  "nombre": "Ana",  
  ...  
}
```

# Flexibilidad de Esquema

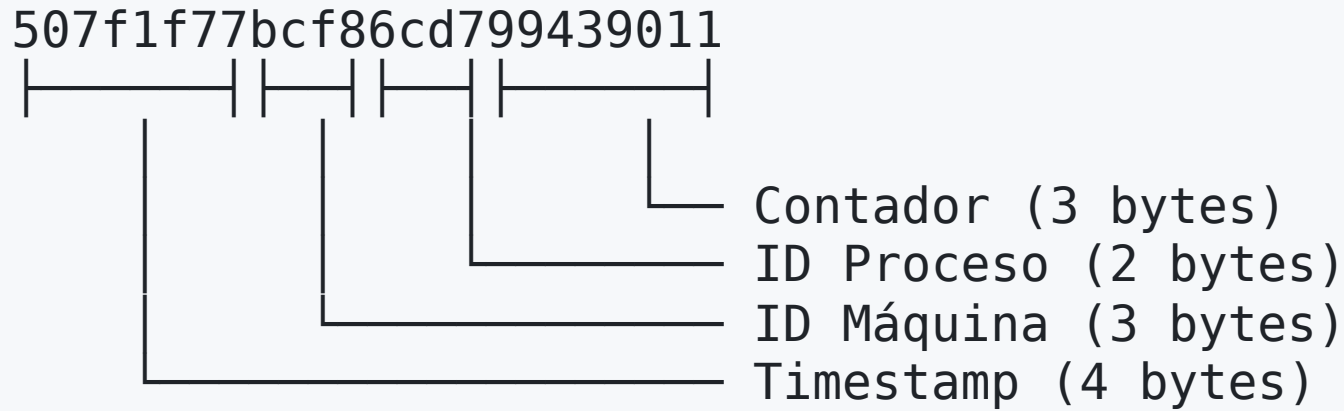
## Documento 1

```
{  
  "nombre": "Ana",  
  "email": "ana@test.com"  
}
```

## Documento 2

```
{  
  "nombre": "Carlos",  
  "email": "carlos@test.com",  
  "telefono": "612345678",  
  "linkedin": "linkedin.com/carlos"  
}
```

# El Identificador \_id (ObjectId)



## Puntos clave:

- 12 bytes, representado como 24 caracteres hex
- Generado automáticamente si no se especifica
- Ordenable cronológicamente
- Único globalmente

## Operaciones CRUD

| Operación | SQL               | MongoDB                   |
|-----------|-------------------|---------------------------|
| Create    | INSERT INTO...    | db.users.insertOne({...}) |
| Read      | SELECT * FROM...  | db.users.find({...})      |
| Update    | UPDATE ... SET... | db.users.updateOne({...}) |
| Delete    | DELETE FROM...    | db.users.deleteOne({...}) |



# Buscar usuarios de IT activos

## SQL

```
SELECT * FROM usuarios  
WHERE departamento = 'IT'  
AND activo = true;
```

## MongoDB Shell

```
db.usuarios.find({  
  departamento: "IT",  
  activo: true  
})
```

## MongoDB Java

```
collection.find(  
  Filters.and(  

```

# MongoDB es Ideal Para...

## **USAR**

- Catálogos de productos
- Gestión de contenido (CMS)
- Datos de IoT / sensores
- Perfiles de usuario
- Logs y analytics
- Aplicaciones móviles
- Prototipado rápido

## **EVITAR**

- Transacciones bancarias complejas
- Sistemas con muchas relaciones

# **Empresas que Usan MongoDB**

**Netflix Uber Adobe eBay**

**Forbes Cisco Bosch SAP**

**EA Verizon Toyota Expedia**



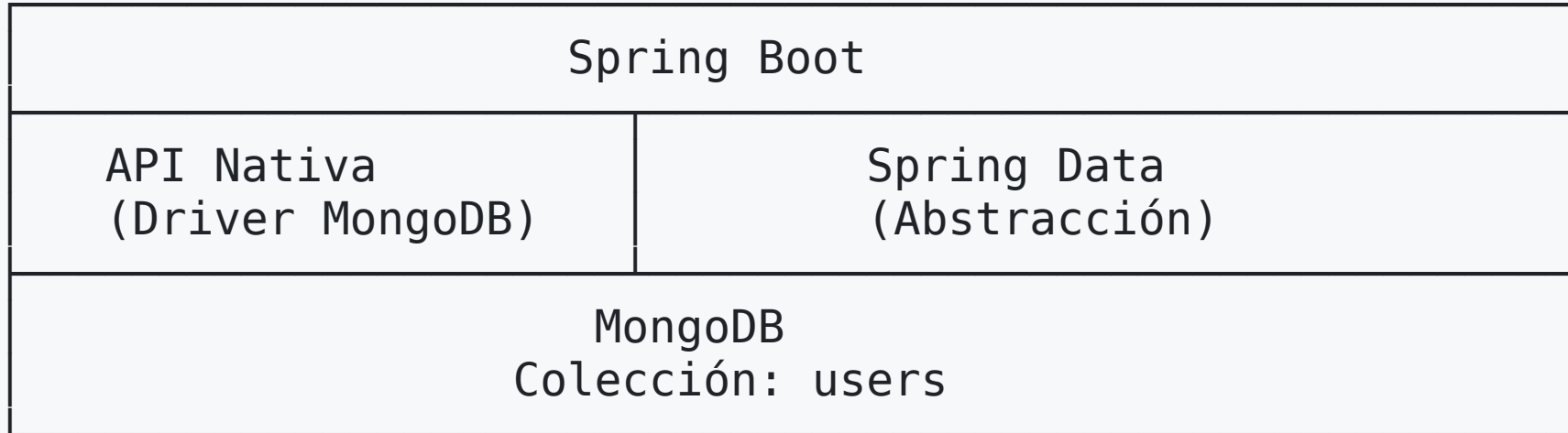
**Más de 46,000 empresas usan MongoDB**

# SQL vs MongoDB - Resumen

| Aspecto       | SQL                 | MongoDB                  |
|---------------|---------------------|--------------------------|
| Modelo        | Relacional          | Documental               |
| Esquema       | Rígido              | Flexible                 |
| Escalado      | Vertical            | Horizontal               |
| Relaciones    | JOINS               | Embedding                |
| Transacciones | Nativas ACID        | Limitadas*               |
| Consultas     | SQL                 | JSON/BSON                |
| Ideal para    | Datos estructurados | Datos semi-estructurados |

\*MongoDB soporta transacciones multi-documento desde v4.0, pero no es su punto fuerte.

# Nuestro Proyecto: Gestión de Usuarios



## Lo que aprenderemos:

- Operaciones CRUD de dos formas diferentes
- Consultas con filtros dinámicos
- Agregaciones básicas
- Comparar ambos enfoques






# Reflexión

**Si tuvieras que desarrollar una app de e-commerce con millones de productos...**

¿Usarías SQL, MongoDB, o ambos?

¿Para qué parte usarías cada uno?





## ¿Qué Viene Ahora?

1.  Poner en marcha el proyecto
2.  Explorar Swagger UI
3.  Conocer la estructura del código
4.  Analizar los métodos ya implementados
5.  Identificar los TODOs a completar

**¡Manos a la obra!** 

# Para Saber Más

## Recursos:

-  [docs.mongodb.com](https://docs.mongodb.com) - Documentación oficial
-  [university.mongodb.com](https://university.mongodb.com) - Cursos gratuitos
-  [db-engines.com](https://db-engines.com) - Rankings de BD
-  [mongodb.com/try](https://mongodb.com/try) - MongoDB Atlas (cloud)

## En el proyecto:

- `README.md` - Guía completa
- `ARQUITECTURA.md` - Diseño técnico



# ¿Preguntas?



Acceso a Datos - 2º DAM