



Bases de datos NoSQL

2da Unidad

CONTENIDO

1. Comprender cómo funciona la estructura flexible de documentos en MongoDB.
2. Aplicar consultas usando operadores relacionales y lógicos.
3. Manipular datos complejos: arrays, subdocumentos, fechas.
4. Usar filtros anidados y jerárquicos para extraer información útil

Estructura flexible de datos

- MongoDB permite documentos sin esquema rígido.
- Cada documento puede tener su propia estructura.
- Ideal para datos dinámicos y anidados.

```
{  
  "nombre": "Ana",  
  "edad": 30,  
  "direccion": {  
    "ciudad": "Valdivia",  
    "codigoPostal": 5090000  
  }  
}
```

Operadores relacionales (comparación)

Los **operadores relacionales** permiten hacer comparaciones entre valores numéricos, de fecha o texto.

 **Ejemplo práctico:**

```
db.productos.find({ precio: { $gte: 30000 } })
```

Busca todos los productos con precio mayor o igual a 30.000.

Operador	Descripción	Ejemplo
\$eq	Igual a	{ edad: { \$eq: 25 } }
\$ne	Distinto de	{ edad: { \$ne: 18 } }
\$gt	Mayor que	{ precio: { \$gt: 10000 } }
\$gte	Mayor o igual que	{ edad: { \$gte: 18 } }
\$lt	Menor que	{ stock: { \$lt: 5 } }
\$lte	Menor o igual que	{ stock: { \$lte: 10 } }

Operadores lógicos

Los **operadores lógicos** permiten combinar múltiples condiciones en una sola consulta.

Operador	Descripción	Ejemplo
\$and	Todas las condiciones deben cumplirse	{ \$and: [{ edad: { \$gt: 18 } }, { ciudad: "Osorno" }] }
\$or	Al menos una condición debe cumplirse	{ \$or: [{ stock: 0 }, { categoria: "Obsoletos" }] }
\$not	Niega una condición (se usa junto a otro operador)	{ stock: { \$not: { \$gt: 0 } } }
\$nor	Ninguna condición debe cumplirse	{ \$nor: [{ ciudad: "Temuco" }, { ciudad: "Valdivia" }] }

```
db.clientes.find({
  $or: [
    { edad: { $lt: 18 } },
    { ciudad: "Puerto Montt" }
  ]
})
```

Arrays y subdocumentos

Arrays

MongoDB permite almacenar listas de valores en un solo campo.

```
{ "nombre": "Carlos", "etiquetas": ["activo", "vip", "newsletter"] }
```

 Puedes consultar con:

```
db.usuarios.find({ etiquetas: "vip" })
```

 Busca documentos donde el array etiquetas contenga "vip".

Subdocumentos

Un subdocumento es un objeto dentro de otro documento.

```
{ "nombre": "Sofía", "direccion": { "ciudad": "Osorno", "region": "Los Lagos" } }
```

 Consulta con notación de puntos (dot notation):

```
db.usuarios.find({ "direccion.ciudad": "Osorno" })
```


Consultas sobre arrays

Operador	Qué hace	Ejemplo
\$in	Coincide si el campo tiene alguno de los valores	{ categoria: { \$in: ["Electrónica", "Audio"] } }
\$all	Coincide si el array contiene todos los elementos	{ etiquetas: { \$all: ["nuevo", "oferta"] } }
\$size	Coincide con la cantidad exacta de elementos del array	{ etiquetas: { \$size: 3 } }
\$elemMatch	Coincide con al menos un elemento que cumpla múltiples condiciones	{ compras: { \$elemMatch: { producto: "Notebook", precio: { \$lt: 500000 } } } }

Manejo de fechas

MongoDB usa el tipo ISODate para trabajar con fechas.

 **Insertar una fecha:**

```
{ fechaRegistro: ISODate("2024-03-30") }
```

 **Comparar fechas:**

```
db.registros.find({ fechaRegistro: { $gte: ISODate("2024-01-01") } })
```

 **Filtrar por rango:**

```
db.eventos.find({ fecha: { $gte: ISODate("2024-05-01"), $lte: ISODate("2024-05-31") } })
```

Filtros y búsquedas anidadas

 Puedes consultar campos dentro de subdocumentos usando "campo.subcampo".

Ejemplo:

```
db.alumnos.find({ "contacto.email": { $ne: null } })
```

 También puedes usar condiciones combinadas en subdocumentos:

```
db.alumnos.find({ "direccion.region": "Los Lagos", "direccion.comuna": "Puerto Montt" })
```

Estructuras jerárquicas (arrays de subdocumentos)

Ejemplo de estructura:

```
{ "nombre": "Pablo", "compras": [ { "producto": "Monitor", "precio": 120000 },  
  { "producto": "Teclado", "precio": 30000 } ] }
```


 Consulta con \$elemMatch:

```
db.usuarios.find({ compras: { $elemMatch: { producto: "Teclado", precio: { $lt: 40000 }  
  } } })
```


Ejercicio 1


 Mostrar todos los usuarios que tengan más de 30 años.

Ejercicio 1

 Mostrar todos los usuarios que tengan más de 30 años.

```
db.usuarios.find({ edad: { $gt: 30 } })
```

Ejercicio 2  Mostrar los usuarios cuya ciudad sea "Valdivia".

Ejercicio 2  Mostrar los usuarios cuya ciudad sea "Valdivia".


```
db.usuarios.find({ "direccion.ciudad":  
"Valdivia" })
```

Ejercicio 3  Mostrar usuarios con la etiqueta "vip" en su campo etiquetas.

Ejercicio 3  Mostrar usuarios con la etiqueta "vip" en su campo etiquetas.


```
db.usuarios.find({ etiquetas: "vip" })
```

Ejercicio 4 📌 Mostrar los usuarios que se registraron después del 1 de febrero de 2024.

Ejercicio 4  Mostrar los usuarios que se registraron después del 1 de febrero de 2024.

```
db.usuarios.find({  
  fechaRegistro: { $gt: ISODate("2024-  
02-01") }  
})
```

Ejercicio 5  Mostrar usuarios con edad entre 25 y 35 años.


Ejercicio 5  Mostrar usuarios con edad entre 25 y 35 años.

```
db.usuarios.find({  
  edad: { $gte: 25, $lte: 35 }  
})
```


Ejercicio 6 📌 Mostrar usuarios que no tengan compras registradas.


Ejercicio 6  Mostrar usuarios que no tengan compras registradas.


```
db.usuarios.find({ compras:  
{ $size: 0 } })
```

Ejercicio 7  Buscar usuarios que
hayan comprado un producto llamado
"Notebook".

Ejercicio 7 📌 Buscar usuarios que hayan comprado un producto llamado "Notebook".


```
db.usuarios.find({  
  compras: { $elemMatch: { producto:  
    "Notebook" } }  
})
```

Ejercicio 8  Mostrar los usuarios que viven en la región "Los Lagos" y tienen etiqueta "newsletter".


Ejercicio 8  Mostrar los usuarios que viven en la región "Los Lagos" y tienen etiqueta "newsletter".

```
db.usuarios.find({  
  "direccion.region": "Los Lagos",  
  etiquetas: "newsletter"  
})
```

Ejercicio 9


 Mostrar los usuarios que tienen más de una etiqueta.

Ejercicio 9

 Mostrar los usuarios que tienen más de una etiqueta.

```
db.usuarios.find({  
  etiquetas: { $exists: true },  
  $expr: { $gt: [ { $size: "$etiquetas" }, 1 ] }  
})
```

Ejercicio 10

 Mostrar usuarios que hayan comprado algún producto con precio mayor a 100.000.

Ejercicio 10

 Mostrar usuarios que hayan comprado algún producto con precio mayor a 100.000.

```
db.usuarios.find({
  compras: {
    $elemMatch: { precio: { $gt: 100000 } }
  }
})
```

MUCHAS GRACIAS!



inacap.cl