

Trabajo Práctico – Queries y Operadores en MongoDB

Parte Teórica

1. Operadores de comparación

1. Explicar la diferencia entre **\$eq** y **\$ne**.

\$eq: selecciona documentos donde el valor de un campo es igual al especificado.

\$ne: selecciona documentos donde el valor de un campo NO es igual al especificado.

2. ¿Qué devuelven los operadores **\$gt** y **\$gte**? Ejemplificar con un caso.

gt (greater than): valores mayores que.

\$gte (greater than or equal): valores mayores o iguales que.

Productos con precio > 100

```
db.productos.find({ precio: { $gt: 100 } });
```

Productos con precio >= 100

```
db.productos.find({ precio: { $gte: 100 } });
```

3. ¿Cuándo conviene usar **\$in** y cuándo **\$nin**?

\$in: busca documentos donde el campo coincida con cualquiera de los valores de una lista.

\$nin: busca documentos donde el campo NO esté en la lista de valores.

2. Operadores lógicos

1. ¿Cuál es la diferencia entre **\$and** y **\$or**?

\$and: devuelve documentos que cumplen todas las condiciones.

\$or: devuelve documentos que cumplen al menos una de las condiciones.

2. ¿Qué operador usarías para obtener documentos que no cumplan cierta condición?

Se usa **\$not**.

3. Operadores de arrays

1. ¿Qué hace **\$all** y en qué se diferencia de **\$in**?

\$all: selecciona documentos donde el array contiene todos los valores especificados.

\$in: selecciona documentos donde el array contiene al menos uno de los valores.

2. ¿Qué devuelve \$size y sobre qué tipo de campos se aplica?

Devuelve documentos donde un array tiene un tamaño específico.
Solo se aplica sobre campos de tipo array.

4. Operadores de actualización

1. Explicar la diferencia entre \$set y \$unset.

\$set: asigna un valor a un campo (lo crea si no existe).
\$unset: elimina un campo del documento.

2. ¿Cómo funciona \$inc? ¿Se puede usar con valores negativos?

\$inc: incrementa (o decrementa) el valor numérico de un campo.
Sí, se puede usar con valores negativos para restar.

3. ¿Cuál es la diferencia entre \$push y \$addToSet?

\$push: agrega un valor a un array (aunque ya exista).
\$addToSet: agrega un valor a un array solo si no existe (evita duplicados).

Datos para usar

Colección **alumnos**

```
[
  { "nombre": "Ana", "edad": 20, "carrera": "Ingeniería", "materias": ["BD", "Algoritmos"], "promedio": 8.5 },
  { "nombre": "Luis", "edad": 22, "carrera": "Medicina", "materias": ["Anatomía", "Biología"], "promedio": 7.2 },
  { "nombre": "María", "edad": 19, "carrera": "Derecho", "materias": ["Derecho Penal", "Derecho Civil"], "promedio": 6.8 },
  { "nombre": "Pedro", "edad": 25, "carrera": "Ingeniería", "materias": ["BD", "Sistemas"], "promedio": 9.1 },
  { "nombre": "Sofía", "edad": 21, "carrera": "Arquitectura", "materias": ["Diseño", "Historia"], "promedio": 7.9 },
  { "nombre": "Martín", "edad": 18, "carrera": "Ingeniería", "materias": ["Algoritmos"], "promedio": 5.5 }
]
```

Colección **productos**

```
[
  { "nombre": "Notebook", "precio": 1200, "stock": 5, "categorias": ["Electrónica", "Computación"] },
  { "nombre": "Mouse", "precio": 20, "stock": 50, "categorias": ["Electrónica", "Accesorios"] },
  { "nombre": "Teclado", "precio": 35, "stock": 15, "categorias": ["Electrónica", "Accesorios"] },
  { "nombre": "Monitor", "precio": 300, "stock": 10, "categorias": ["Electrónica", "Pantallas"] },
]
```

```
{ "nombre": "Impresora", "precio": 200, "stock": 0, "categorias": ["Electrónica", "Oficina"] },  
{ "nombre": "Tablet", "precio": 600, "stock": 8, "categorias": ["Electrónica", "Computación"] }  
]
```

Parte Práctica

Ejercicios con **alumnos**

1. Buscar todos los alumnos **mayores de 20 años**.

```
db.alumnos.find({edad:{$gte: 20}})
```

```
tpmongo> db.alumnos.find({edad:{$gte: 20}})  
[  
  {  
    _id: ObjectId('68b06820004c7efc9aee4af'),  
    nombre: 'Ana',  
    edad: 20,  
    carrera: 'Ingeniería',  
    materias: [ 'BD', 'Algoritmos' ],  
    promedio: 8.5  
  },  
  {  
    _id: ObjectId('68b06820004c7efc9aee4b0'),  
    nombre: 'Luis',  
    edad: 22,  
    carrera: 'Medicina',  
    materias: [ 'Anatomía', 'Biología' ],  
    promedio: 7.2  
  },  
  {  
    _id: ObjectId('68b06820004c7efc9aee4b2'),  
    nombre: 'Pedro',  
    edad: 25,  
    carrera: 'Ingeniería',  
    materias: [ 'BD', 'Sistemas' ],  
    promedio: 9.1  
  },  
  {  
    _id: ObjectId('68b06820004c7efc9aee4b3'),  
    nombre: 'Sofía',  
    edad: 21,  
    carrera: 'Arquitectura',  
    materias: [ 'Diseño', 'Historia' ],  
    promedio: 7.9  
  }  
]  
tpmongo>
```

2. Listar alumnos cuya carrera sea **Ingeniería o Medicina**.

```
db.alumnos.find({ carrera: {$in:["Ingeniería", "Medicina"]} })
```

```
tpmongo> db.alumnos.find({ carrera: {$in:["Ingeniería", "Medicina"]} })
[
  {
    _id: ObjectId('68b06820004c7efc9aee4af'),
    nombre: 'Ana',
    edad: 20,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Algoritmos' ],
    promedio: 8.5
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b0'),
    nombre: 'Luis',
    edad: 22,
    carrera: 'Medicina',
    materias: [ 'Anatomía', 'Biología' ],
    promedio: 7.2
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b2'),
    nombre: 'Pedro',
    edad: 25,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Sistemas' ],
    promedio: 9.1
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b4'),
    nombre: 'Martín',
    edad: 18,
    carrera: 'Ingeniería',
    materias: [ 'Algoritmos' ],
    promedio: 5.5
  }
]
tpmongo>
```

3. Buscar alumnos que tengan un **promedio mayor o igual a 8**.

```
db.alumnos.find({promedio: {$gte: 8}})
```

```
tpmongo> db.alumnos.find({promedio: {$gte: 8}})
[
  {
    _id: ObjectId('68b06820004c7efc9aee4af'),
    nombre: 'Ana',
    edad: 20,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Algoritmos' ],
    promedio: 8.5
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b2'),
    nombre: 'Pedro',
    edad: 25,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Sistemas' ],
    promedio: 9.1
  }
]
tpmongo>
```

4. Listar alumnos con exactamente **2 materias**.

```
tpmongo> db.alumnos.find({materias:{$size: 2}})
```

```
tpmongo> db.alumnos.find({materias:{$size: 2}})
[
  {
    _id: ObjectId('68b06820004c7efc9aee4af'),
    nombre: 'Ana',
    edad: 20,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Algoritmos' ],
    promedio: 8.5
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b0'),
    nombre: 'Luis',
    edad: 22,
    carrera: 'Medicina',
    materias: [ 'Anatomía', 'Biología' ],
    promedio: 7.2
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b1'),
    nombre: 'Maria',
    edad: 19,
    carrera: 'Derecho',
    materias: [ 'Derecho Penal', 'Derecho Civil' ],
    promedio: 6.8
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b2'),
    nombre: 'Pedro',
    edad: 25,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Sistemas' ],
    promedio: 9.1
  },
  {
    _id: ObjectId('68b06820004c7efc9aee4b3'),
    nombre: 'Sofia',
    edad: 21,
    carrera: 'Arquitectura',
    materias: [ 'Diseño', 'Historia' ],
    promedio: 7.9
  }
]
tpmongo> _
```

5. Actualizar la edad de **Ana** sumando 1 año.

```
db.alumnos.updateOne({nombre: "Ana"}, {$inc: {edad: 1}})
```

```
tpmongo> db.alumnos.updateOne({nombre: "Ana"}, {$inc: {edad: 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
tpmongo> db.alumnos.find().pretty()
[
  {
    _id: ObjectId('68b06820004c7efc9aee4af'),
    nombre: 'Ana',
    edad: 21,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Algoritmos' ],
    promedio: 8.5
  }
]
```

6. Agregar la materia "MongoDB" a Pedro, pero solo si no la tiene.

```
db.alumnos.updateOne(
  { nombre: "Pedro" },
  { $addToSet: { materias: "MongoDB" } }
)
```

```
tpmongo> db.alumnos.updateOne(
...   { nombre: "Pedro" },
...   { $addToSet: { materias: "MongoDB" } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
tpmongo> db.alumnos.find({ nombre: "Pedro" })
...
[
  {
    _id: ObjectId('68b06820004c7efc9aee4b2'),
    nombre: 'Pedro',
    edad: 25,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Sistemas', 'MongoDB' ],
    promedio: 9.1
  }
]
tpmongo> _
```

7. Eliminar el campo promedio de Martín.

```
db.alumnos.updateOne(

  { nombre: "Martín" },

  { $unset: { promedio: "" } } )
```

```
tpmongo> db.alumnos.updateOne(
...   { nombre: "Martín" },
...   { $unset: { promedio: "" } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
tpmongo> db.alumnos.find({ nombre: "Martín" })
[
  {
    _id: ObjectId('68b06820004c7efc9aee4b4'),
    nombre: 'Martín',
    edad: 18,
    carrera: 'Ingeniería',
    materias: [ 'Algoritmos' ]
  }
]
tpmongo>
```

Ejercicios con **productos**

1. Buscar productos con **precio menor a 100**.

```
db.productos.find({ precio: { $lt: 100 } })
```

```
tpmongo> db.productos.find({ precio: { $lt: 100 } })
...
[
  {
    _id: ObjectId('68b06857004c7efc9aee4b6'),
    nombre: 'Mouse',
    precio: 20,
    stock: 50,
    categorias: [ 'Electrónica', 'Accesorios' ]
  },
  {
    _id: ObjectId('68b06857004c7efc9aee4b7'),
    nombre: 'Teclado',
    precio: 35,
    stock: 15,
    categorias: [ 'Electrónica', 'Accesorios' ]
  }
]
```

2. Mostrar nombre y precio de productos en la categoría "Computación".

```
db.productos.find( { categorias: "Computación" }, { _id: 0, nombre: 1, precio: 1 })
```

```
tpmongo> db.productos.find(
...   { categorias: "Computación" },
...   { _id: 0, nombre: 1, precio: 1 }
... )
...
[
  { nombre: 'Notebook', precio: 1200 },
  { nombre: 'Tablet', precio: 600 }
]
tpmongo>
```

3. Encontrar productos con **stock igual a 0**.

```
db.productos.find({ stock: 0 })
```

```
tpmongo> db.productos.find({ stock: 0 })
...
[
  {
    _id: ObjectId('68b06857004c7efc9aee4b9'),
    nombre: 'Impresora',
    precio: 200,
    stock: 0,
    categorias: [ 'Electrónica', 'Oficina' ]
  }
]
tpmongo>
```

4. Buscar productos con **precio entre 100 y 1000**.

db.productos.find({ precio: { \$gte: 100, \$lte: 1000 } })

```
tpmongo> db.productos.find({ precio: { $gte: 100, $lte: 1000 } })
...
[
  {
    _id: ObjectId('68b06857004c7efc9aeec4b8'),
    nombre: 'Monitor',
    precio: 300,
    stock: 10,
    categorias: [ 'Electrónica', 'Pantallas' ]
  },
  {
    _id: ObjectId('68b06857004c7efc9aeec4b9'),
    nombre: 'Impresora',
    precio: 200,
    stock: 0,
    categorias: [ 'Electrónica', 'Oficina' ]
  },
  {
    _id: ObjectId('68b06857004c7efc9aeec4ba'),
    nombre: 'Tablet',
    precio: 600,
    stock: 8,
    categorias: [ 'Electrónica', 'Computación' ]
  }
]
tpmongo>
```

5. Aumentar en un **10%** el precio de la Notebook.

db.productos.updateOne({ nombre: "Notebook" }, { \$mul: { precio: 1.10 } })

```
tpmongo> db.productos.updateOne(
...   { nombre: "Notebook" },
...   { $mul: { precio: 1.10 } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
tpmongo> db.productos.find({ nombre: "Notebook" })
[
  {
    _id: ObjectId('68b06857004c7efc9aeec4b5'),
    nombre: 'Notebook',
    precio: 1320,
    stock: 5,
    categorias: [ 'Electrónica', 'Computación' ]
  }
]
tpmongo>
```

6. Disminuir en 5 unidades el stock de la Tablet.

```
db.productos.updateOne(
  { nombre: "Tablet" },
  { $inc: { stock: -5 } }
)
```



```
tpmongo> db.productos.updateOne(
...   { nombre: "Tablet" },
...   { $inc: { stock: -5 } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
tpmongo> db.productos.find({ nombre: "Tablet" })
[
  {
    _id: ObjectId('68b06857004c7efc9aee4ba'),
    nombre: 'Tablet',
    precio: 600,
    stock: 3,
    categorias: [ 'Electrónica', 'Computación' ]
  }
]
tpmongo> _
```

7. Agregar la categoría "Ofertas" a los productos con precio menor a 50.

```
db.productos.updateMany(
  { precio: { $lt: 50 } },
  { $addToSet: { categorias: "Ofertas" } }
)
```

```
tpmongo> db.productos.updateMany(
...   { precio: { $lt: 50 } },
...   { $addToSet: { categorias: "Ofertas" } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
tpmongo> db.productos.find({ precio: { $lt: 50 } })
...
[
  {
    _id: ObjectId('68b06857004c7efc9aee4b6'),
    nombre: 'Mouse',
    precio: 20,
    stock: 50,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ]
  },
  {
    _id: ObjectId('68b06857004c7efc9aee4b7'),
    nombre: 'Teclado',
    precio: 35,
    stock: 15,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ]
  }
]
tpmongo> _
```

8. Agregar un campo **ultimaActualizacion** con la fecha actual a todos los productos.

```
db.productos.updateMany(
  {},
  { $currentDate: { ultimaActualizacion: true } })
```

```
tpmongo> db.productos.updateMany(
...   {},
...   { $currentDate: { ultimaActualizacion: true } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 6,
  modifiedCount: 6,
  upsertedCount: 0
}
tpmongo> db.productos.find().pretty()
[
  {
    _id: ObjectId('68b06857004c7efc9aeec4b5'),
    nombre: 'Notebook',
    precio: 1320,
    stock: 5,
    categorias: [ 'Electrónica', 'Computación' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.637Z')
  },
  {
    _id: ObjectId('68b06857004c7efc9aeec4b6'),
    nombre: 'Mouse',
    precio: 20,
    stock: 50,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z')
  },
  {
    _id: ObjectId('68b06857004c7efc9aeec4b7'),
    nombre: 'Teclado',
    precio: 35,
    stock: 15,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z')
  },
  {
    _id: ObjectId('68b06857004c7efc9aeec4b8'),
    nombre: 'Monitor',
    precio: 300,
    stock: 10,
    categorias: [ 'Electrónica', 'Pantallas' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z')
  },
]
```

Ejercicios Desafío (armar la query de búsqueda)

1. Buscar alumnos que:
 - Estudien Ingeniería
 - Y tengan promedio **mayor a 7**
 - O sean menores de 20 años

```
db.alumnos.find({
  $or: [
    { $and: [ { carrera: "Ingeniería" }, { promedio: { $gt: 7 } } ] },
    { edad: { $lt: 20 } }
  ]
})
```

```
tpmongo> db.alumnos.find({
...   $or: [
...     { $and: [ { carrera: "Ingeniería" }, { promedio: { $gt: 7 } } ] },
...     { edad: { $lt: 20 } }
...   ]
... })
...
[
  {
    _id: ObjectId('68b06820004c7efc9aeec4af'),
    nombre: 'Ana',
    edad: 21,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Algoritmos' ],
    promedio: 8.5
  },
  {
    _id: ObjectId('68b06820004c7efc9aeec4b1'),
    nombre: 'Maria',
    edad: 19,
    carrera: 'Derecho',
    materias: [ 'Derecho Penal', 'Derecho Civil' ],
    promedio: 6.8
  },
  {
    _id: ObjectId('68b06820004c7efc9aeec4b2'),
    nombre: 'Pedro',
    edad: 25,
    carrera: 'Ingeniería',
    materias: [ 'BD', 'Sistemas', 'MongoDB' ],
    promedio: 9.1
  },
  {
    _id: ObjectId('68b06820004c7efc9aeec4b4'),
    nombre: 'Martin',
    edad: 18,
    carrera: 'Ingeniería',
    materias: [ 'Algoritmos' ]
  }
]
tpmongo> █
```

2. Actualizar productos que:

- Tengan **stock = 0**
- O un precio menor a 50
- Para que se les agregue **descontinuado: true**

```
db.productos.updateMany(
  { $or: [ { stock: 0 }, { precio: { $lt: 50 } } ] },
  { $set: { descontinuado: true } }
)
```

```
tpmongo> db.productos.updateMany(
...   { $or: [ { stock: 0 }, { precio: { $lt: 50 } } ] },
...   { $set: { descontinuado: true } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
tpmongo> db.productos.find({
...   $or: [
...     { stock: 0 },
...     { precio: { $lt: 50 } }
...   ]
... })
...
[
  {
    _id: ObjectId('68b06857004c7efc9aee4b6'),
    nombre: 'Mouse',
    precio: 20,
    stock: 50,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z'),
    descontinuado: true
  },
  {
    _id: ObjectId('68b06857004c7efc9aee4b7'),
    nombre: 'Teclado',
    precio: 35,
    stock: 15,
    categorias: [ 'Electrónica', 'Accesorios', 'Ofertas' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z'),
    descontinuado: true
  },
  {
    _id: ObjectId('68b06857004c7efc9aee4b9'),
    nombre: 'Impresora',
    precio: 200,
    stock: 0,
    categorias: [ 'Electrónica', 'Oficina' ],
    ultimaActualizacion: ISODate('2025-08-29T02:09:08.641Z'),
    descontinuado: true
  }
]
```

3.

- Listar usuarios de Barcelona entre 20 y 40 años con suscripción "gold" o "premium".

```
db.usuarios.find({
  ciudad: "Barcelona",
  edad: { $gte: 20, $lte: 40 },
  suscripciones: { $in: ["gold", "premium"] }
})
```

- Obtener usuarios que **no vivan** en Madrid y **no tengan suscripciones**.

```
db.usuarios.find({
  ciudad: { $ne: "Madrid" },
  $or: [
    { suscripciones: { $exists: false } },
    { suscripciones: { $size: 0 } }
  ]
})
```

- Buscar usuarios con "newsletter" en sus suscripciones o cuyo email termine en @gmail.com.

```
db.usuarios.find({
  $or: [
    { suscripciones: "newsletter" },
    { email: /@gmail\.com$/i }
  ]
})
```

- Aumentar en 1 año la edad de todos los usuarios de Sevilla.

```
db.usuarios.updateMany(
  { ciudad: "Sevilla" },
  { $inc: { edad: 1 } }
)
```

- Eliminar el campo telefono de los usuarios de Barcelona.

```
db.usuarios.updateMany(
  { ciudad: "Barcelona" },
  { $unset: { telefono: "" } }
)
```

- Agregar "ofertas" a las suscripciones de los usuarios con menos de 20 años.

```
db.usuarios.updateMany(
  { edad: { $lt: 20 } },
  { $addToSet: { suscripciones: "ofertas" } }
)
```